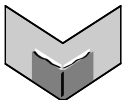
The Journal of Machine Learning Research
Print-Archive Edition

Volume 20
Issues 1–92



Microtome Publishing
Brookline, Massachusetts
www.mtome.com

The Journal of Machine Learning Research
Print-Archive Edition
Volume 20
Issues 1–92

The Journal of Machine Learning Research (JMLR) is an open access journal. All articles published in JMLR are freely available via electronic distribution. This Print-Archive Edition is published annually as a means of archiving the contents of the journal in perpetuity. The contents of this edition are articles published electronically in JMLR in 2019.

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

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

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

Collection copyright © 2019 The Journal of Machine Learning Research, Inc. and Microtome Publishing.
Individual articles copyright © 2019 by their respective authors and distributed under a Creative Commons Attribution 4.0 International License.

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

JMLR Editorial Board

Editor-in-Chief
Francis Bach, Centre de Recherche INRIA de Paris

Editor-in-Chief
David Blei, Columbia University

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

Managing Editor
Fabian Pedregosa, Google Reseach

Editorial Assistant
Karin Bierig, MPI for Intelligent Systems

Production Editor
Alp Kucukelbir, Fero Labs and Columbia University

Production Staff
Trevor Campbell, Massachusetts Institute of Technology **Behzad Tabibian**, MPI for Intelligent Systems **Hoon Cho**, Massachusetts Institute of Technology

Web Master
Alp Kucukelbir, Fero Labs and Columbia University **Fabian Pedregosa**, Google Reseach

JMLR Action Editors
Ryan Adams, Princeton University, USA **Shivani Agarwal**, University of Pennsylvania, USA **Edoardo M. Airoldi**, Harvard University, USA **Genevera Allen**, Rice University, USA **Anima Anandkumar**, California Institute of Technology, USA **Peter Auer**, University of Leoben, Austria **Samy Bengio**, Google Research, USA **Yoshua Bengio**, Université de Montréal, Canada **Jeff Bilmes**, University of Washington, USA **Karsten Borgwardt**, ETH Zurich, Switzerland **Miguel A. Carreira-Perpinan**, University of California, Merced, USA **Alexander Clark**, King's College London, UK **Corinna Cortes**, Google Research, USA **Koby Crammer**, Technion, Israel **John P. Cunningham**, Columbia University, USA **Arnak Dalalyan**, ENSAE / CREST, France **Inderjit S. Dhillon**, University of Texas, Austin, USA **Tina Eliassi-Rad**, Northeastern University **Gal Elidan**, Hebrew University, Israel **Charles Elkan**, University of California at San Diego, USA **Barbara Engelhardt**, Princeton University, USA **Rob Fergus**, New York University, USA **Kenji Fukumizu**, The Institute of Statistical Mathematics, Japan **Amir Globerson**, Tel Aviv University, Israel **Moises Goldszmidt**, Microsoft Research, USA **Russ Greiner**, University of Alberta, Canada **Arthur Gretton**, University College London, UK **Maya Gupta**, Google Research, USA **Isabelle Guyon**, ClopiNet, USA **Zaid Harchaoui**, University of Washington **Moritz Hardt**, Google Research, USA **Philipp Hennig**, University of Tübingen **Bert Huang**, Virginia Tech, USA **Aapo Hyvärinen**, University of Helsinki, Finland **Tommi Jaakkola**, Massachusetts Institute of Technology, USA **Prateek Jain**, Microsoft Research, India **Stefanie Jegelka**, Massachusetts Institute of Technology, USA **Samuel Kaski**, Aalto University, Finland **Sathiya Keerthi**, Microsoft Research, USA **Emtiyaz**

Khan, RIKEN Center for Advanced Intelligence, Japan **George Konidaris**, Duke University, USA **Andreas Krause**, ETH Zurich, Switzerland **Sanjiv Kumar**, Google Research, USA **Simon Lacoste-Julien**, Quebec Institute of Artificial Intelligence, Canada **Christoph Lampert**, Institute of Science and Technology, Austria **Honglak Lee**, Google and University of Michigan, Ann Arbor **Daniel Lee**, University of Pennsylvania, USA **Qiang Liu**, Dartmouth College, USA **Gábor Lugosi**, Pompeu Fabra University, Spain **Michael Mahoney**, University of California at Berkeley, USA **Vikash K. Mansinghka**, Massachusetts Institute of Technology, USA **Julien Mairal**, INRIA **Shie Mannor**, Technion, Israel **Jon McAuliffe**, University of California at Berkeley, USA **Robert E. McCulloch**, University of Chicago, USA **Chris Meek**, Microsoft Research, USA **Qiaozhu Mei**, University of Michigan, USA **Vahab Mirrokni**, Google Research, USA **Mehryar Mohri**, New York University, USA **Joris Mooij**, University of Amsterdam, Netherlands **Sayan Mukherjee**, Duke University, USA **Boaz Nadler**, Weizmann Institute of Science, Israel **Long Nguyen**, University of Michigan, USA **Sebastian Nowozin**, Google Research **Una-May O'Reilly**, Massachusetts Institute of Technology, USA **Laurent Orseau**, Google Deepmind, USA **Jie Peng**, University of California, Davis, USA **Jan Peters**, Technische Universitaet Darmstadt, Germany **Avi Pfeffer**, Charles River Analytics, USA **Joelle Pineau**, McGill University, Canada **Massimiliano Pontil**, Istituto Italiano di Tecnologia (Italy), University College London (UK) **Pushmeet Kohli**, DeepMind **Luc de Raedt**, Katholieke Universiteit Leuven, Belgium **Garvesh Raskutti**, University of Wisconsin-Madison, USA **Ben Recht**, University of California, Berkeley, USA **Lorenzo Rosasco**, Massachusetts Institute of Technology, USA **Ruslan Salakhutdinov**, University of Toronto, Canada **Sujay Sanghavi**, University of Texas, Austin, USA **Marc Schoenauer**, INRIA Saclay, France **Ohad Shamir**, Weizmann Institute of Science, Israel **John Shawe-Taylor**, University College London, UK **Xiaotong Shen**, University of Minnesota, USA **David Sontag**, Massachusetts Institute of Technology **Peter Spirtes**, Carnegie Mellon University, USA **Suvrit Sra**, Massachusetts Institute of Technology, USA **Ingo Steinwart**, University of Stuttgart, Germany **Amos Storkey**, University of Edinburgh, UK **Erik Sudderth**, University of California, Irvine, USA **Csaba Szepesvari**, University of Alberta, Canada **Olivier Teytaud**, INRIA Saclay, France **Ambuj Tewari**, University of Michigan **Ivan Titov**, University of Amsterdam, Netherlands **Ryan Tibshirani**, Carnegie Mellon University **Ryota Tomioka**, Microsoft Research Cambridge, UK **Koji Tsuda**, National Institute of Advanced Industrial Science and Technology, Japan **Nicolas Vayatis**, ENS Cachan, France **Jean-Philippe Vert**, Google Research, France **S V N Vishwanathan**, Purdue University, USA **Manfred Warmuth**, University of California at Santa Cruz, USA **Kilian Weinberger**, Cornell University, USA **David Wipf**, Microsoft Research Asia, China **Daniela Witten**, University of Washington **Frank Wood**, University of British Columbia **Tong Zhang**, Baidu Inc, China **Zhihua Zhang**, Peking University, China

JMLR MLOSS Editors

Alexandre Gramfort, INRIA, Université Paris-Saclay, France **Antti Honkela**, University of Helsinki, Finland **Balázs Kégl**, CNRS / Université Paris-Saclay, France **Andreas C. Müller**, Columbia University, USA **Cheng Soon Ong**, Australian National University, Australia

JMLR Editorial Board

Naoki Abe, IBM TJ Watson Research Center, USA **Yasemin Altun**, Google Inc, Switzerland **Jean-Yves Audibert**, CERTIS, France **Jonathan Baxter**, Australia National University, Australia **Richard K. Belew**, University of California at San Diego, USA **Kristin Bennett**, Rensselaer Polytechnic Institute, USA **Christopher M. Bishop**, Microsoft Research, Cambridge, UK **Lashon Booker**, The Mitre Corporation, USA **Henrik Boström**, Stockholm University/KTH, Sweden **Craig Boutilier**,

Google Research, USA **John Patrick Cunningham**, Columbia University, USA **Nello Cristianini**, University of Bristol, UK **Peter Dayan**, University College, London, UK **Dennis DeCoste**, eBay Research, USA **Thomas Dietterich**, Oregon State University, USA **Saso Dzeroski**, Jozef Stefan Institute, Slovenia **Ran El-Yaniv**, Technion, Israel **Peter Flach**, Bristol University, UK **Dan Geiger**, Technion, Israel **Claudio Gentile**, Università degli Studi dell'Insubria, Italy **Sally Goldman**, Google Research, USA **Thore Graepel**, Google DeepMind and University College London, UK **Tom Griffiths**, University of California at Berkeley, USA **Carlos Guestrin**, University of Washington, USA **Stefan Harmeling**, University of Düsseldorf, Germany **David Heckerman**, Microsoft Research, USA **Katherine Heller**, Duke University, USA **Philipp Hennig**, MPI for Intelligent Systems, Germany **Larry Hunter**, University of Colorado, USA **Jens Kober**, Delft University of Technology, Netherlands **Risi Kondor**, University of Chicago, USA **Aryeh Kontorovich**, Ben-Gurion University of the Negev, Israel **Samory Kpotufe**, Princeton University, USA **John Lafferty**, University of Chicago, USA **Erik Learned-Miller**, University of Massachusetts, Amherst, USA **Fei Fei Li**, Stanford University, USA **Yi Lin**, University of Wisconsin, USA **Wei-Yin Loh**, University of Wisconsin, USA **Richard Maclin**, University of Minnesota, USA **Sridhar Mahadevan**, University of Massachusetts, Amherst, USA **Vikash Mansingkha**, Massachusetts Institute of Technology, USA **Yishay Mansour**, Tel-Aviv University, Israel **Jon McAuliffe**, University of California, Berkeley, USA **Andrew McCallum**, University of Massachusetts, Amherst, USA **Raymond J. Mooney**, University of Texas, Austin, USA **Klaus-Robert Muller**, Technical University of Berlin, Germany **Kevin Murphy**, Google, USA **Guillaume Obozinski**, Ecole des Ponts - ParisTech, France **Pascal Poupart**, University of Waterloo, Canada **Konrad Rieck**, University of Göttingen, Germany **Cynthia Rudin**, Massachusetts Institute of Technology, USA **Suchi Saria**, Johns Hopkins University, USA **Robert Schapire**, Princeton University, USA **Fei Sha**, University of Southern California, USA **Shai Shalev-Shwartz**, Hebrew University of Jerusalem, Israel **Padhraic Smyth**, University of California, Irvine, USA **Bharath Sriperumbudur**, Pennsylvania State University, USA **Alexander Statnikov**, New York University, USA **Jean-Philippe Vert**, Mines ParisTech, France **Martin J. Wainwright**, University of California at Berkeley, USA **Chris Watkins**, Royal Holloway, University of London, UK **Max Welling**, University of Amsterdam, Netherlands **Stefan Wrobel**, Fraunhofer IAIS and University of Bonn, Germany **Chris Williams**, University of Edinburgh, UK **Alice Zheng**, GraphLab, USA

JMLR Advisory Board

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

Journal of Machine Learning Research

Volume 20

Part A

20(1):1–30 **Adaptation Based on Generalized Discrepancy**
Corinna Cortes, Mehryar Mohri, Andrés Muñoz Medina

20(2):1–52 **Transport Analysis of Infinitely Deep Neural Network**
Sho Sonoda, Noboru Murata

20(3):1–44 **Parsimonious Online Learning with Kernels via Sparse Projections in Function Space**
Alec Koppel, Garrett Warnell, Ethan Stump, Alejandro Ribeiro

20(4):1–45 **Convergence Rate of a Simulated Annealing Algorithm with Noisy Observations**
Clément Bouttier, Ioana Gavra

20(5):1–37 **Non-Convex Projected Gradient Descent for Generalized Low-Rank Tensor Regression**
Han Chen, Garvesh Raskutti, Ming Yuan

20(6):1–22 **scikit-multilearn: A Python library for Multi-Label Classification**
Piotr Szymański, Tomasz Kajdanowicz

20(7):1–45 **Scalable Approximations for Generalized Linear Problems**
Murat Erdogdu, Mohsen Bayati, Lee H. Dicker

20(8):1–39 **Forward-Backward Selection with Early Dropping**
Giorgos Borboudakis, Ioannis Tsamardinos

20(9):1–49 **Dynamic Pricing in High-dimensions**
Adel Javanmard, Hamid Nazerzadeh

20(10):1–44 **Graphical Lasso and Thresholding: Equivalence and Closed-form Solutions**
Salar Fattahi, Somayeh Sojoudi

20(11):1–23 **An Approach to One-Bit Compressed Sensing Based on Probably Approximately Correct Learning Theory**
Mehmet Eren Ahsen, Mathukumalli Vidyasagar

20(12):1–49 **Scalable Kernel K-Means Clustering with Nystrom Approximation: Relative-Error Bounds**
Shusen Wang, Alex Gittens, Michael W. Mahoney

20(13):1–34 **Train and Test Tightness of LP Relaxations in Structured Prediction**
Ofer Meshi, Ben London, Adrian Weller, David Sontag

20(14):1–37 **Approximations of the Restless Bandit Problem**
Steffen Grünewälder, Azadeh Khaleghi

20(15):1–38 **Automated Scalable Bayesian Inference via Hilbert Coresets**
Trevor Campbell, Tamara Broderick

20(16):1–24 **Smooth neighborhood recommender systems**
Ben Dai, Junhui Wang, Xiaotong Shen, Annie Qu

20(17):1–38 **Delay and Cooperation in Nonstochastic Bandits**
Nicolò Cesa-Bianchi, Claudio Gentile, Yishay Mansour

20(18):1–29 **Multiplicative local linear hazard estimation and best one-sided cross-validation**
Maria Luz Gámiz, María Dolores Martínez-Miranda, Jens Perch Nielsen

20(19):1–5 **spark-crowd: A Spark Package for Learning from Crowdsourced Big Data**
Enrique G. Rodrigo, Juan A. Aledo, José A. Gámez

20(20):1–33 **Accelerated Alternating Projections for Robust Principal Component Analysis**
HanQin Cai, Jian-Feng Cai, Ke Wei

20(21):1–55 **Spectrum Estimation from a Few Entries**
Ashish Khetan, Sewoong Oh

20(22):1–36 **Random Feature-based Online Multi-kernel Learning in Environments with Unknown Dynamics**
Yanning Shen, Tianyi Chen, Georgios B. Giannakis

20(23):1–38 **Determining the Number of Latent Factors in Statistical Multi-Relational Learning**
Chengchun Shi, Wenbin Lu, Rui Song

20(24):1–29 **Joint PLDA for Simultaneous Modeling of Two Factors**
Luciana Ferrer, Mitchell McLaren

20(25):1–49 **Group Invariance, Stability to Deformations, and Complexity of Deep Convolutional Representations**
Alberto Bietti, Julien Mairal

20(26):1–6 **TensorLy: Tensor Learning in Python**
Jean Kossaifi, Yannis Panagakis, Anima Anandkumar, Maja Pantic

20(27):1–42 **Monotone Learning with Rectified Wire Networks**
Veit Elser, Dan Schmidt, Jonathan Yedidia

20(28):1–6 **Pyro: Deep Universal Probabilistic Programming**
Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, Noah D. Goodman

20(29):1–28 **Iterated Learning in Dynamic Social Networks**
Bernard Chazelle, Chu Wang

20(30):1–34 **Exact Clustering of Weighted Graphs via Semidefinite Programming**
Aleksis Pirinen, Brendan Ames

20(31):1–45 **Kernels for Sequentially Ordered Data**
Franz J. Kiraly, Harald Oberhauser

20(32):1–50 **NetSDM: Semantic Data Mining with Network Analysis**
Jan Kralj, Marko Robnik-Sikonja, Nada Lavrac

20(33):1–38 **The Relationship Between Agnostic Selective Classification, Active Learning and the Disagreement Coefficient**
Roei Gelbhart, Ran El-Yaniv

20(34):1–44 **Matched Bipartite Block Model with Covariates**
Zahra S. Razaee, Arash A. Amini, Jingyi Jessica Li

20(35):1–93 **Optimal Policies for Observing Time Series and Related Restless Bandit Problems**
Christopher R. Dance, Tomi Silander

20(36):1–37 **A New Approach to Laplacian Solvers and Flow Problems**
Patrick Rebeschini, Sekhar Tatikonda

20(37):1–59 **A Well-Tempered Landscape for Non-convex Robust Subspace Recovery**
Tyler Maunu, Teng Zhang, Gilad Lerman

20(38):1–27 **Approximation Hardness for A Class of Sparse Optimization Problems**
Yichen Chen, Yinyu Ye, Mengdi Wang

20(39):1–40 **A Bootstrap Method for Error Estimation in Randomized Matrix Multiplication**
Miles E. Lopes, Shusen Wang, Michael W. Mahoney

20(40):1–47 **Stochastic Modified Equations and Dynamics of Stochastic Gradient Algorithms I: Mathematical Foundations**
Qianxiao Li, Cheng Tai, Weinan E

20(41):1–57 **Decontamination of Mutual Contamination Models**
Julian Katz-Samuels, Gilles Blanchard, Clayton Scott

20(42):1–56 **Utilizing Second Order Information in Minibatch Stochastic Variance Reduced Proximal Iterations**
Jialei Wang, Tong Zhang

20(43):1–58 **DSCOVR: Randomized Primal-Dual Block Coordinate Algorithms for Asynchronous Distributed Optimization**
Lin Xiao, Adams Wei Yu, Qihang Lin, Weizhu Chen

20(44):1–5 **Picasso: A Sparse Learning Library for High Dimensional Data Analysis in R and Python**
Jason Ge, Xingguo Li, Haoming Jiang, Han Liu, Tong Zhang, Mengdi Wang, Tuo Zhao

20(45):1–41 **Robust Frequent Directions with Application in Online Learning**
Luo Luo, Cheng Chen, Zhihua Zhang, Wu-Jun Li, Tong Zhang

20(46):1–36 **Boosted Kernel Ridge Regression: Optimal Learning Rates and Early Stopping**
Shao-Bo Lin, Yunwen Lei, Ding-Xuan Zhou

20(47):1–47 **Analysis of spectral clustering algorithms for community detection: the general bipartite setting**
Zhixin Zhou, Arash A.Amini

20(48):1–23 **Efficient augmentation and relaxation learning for individualized treatment rules using observational data**
Ying-Qi Zhao, Eric B. Laber, Yang Ning, Sumona Saha, Bruce E. Sands

20(49):1–24 **Using Simulation to Improve Sample-Efficiency of Bayesian Optimization for Bipedal Robots**
Akshara Rai, Rika Antonova, Franziska Meier, Christopher G. Atkeson

20(50):1–24 **No-Regret Bayesian Optimization with Unknown Hyperparameters**
Felix Berkenkamp, Angela P. Schoellig, Andreas Krause

20(51):1–18 **Bayesian Combination of Probabilistic Classifiers using Multivariate Normal Mixtures**
Gregor Pirš, Erik Štrumbelj

20(52):1–24 **Thompson Sampling Guided Stochastic Searching on the Line for Deceptive Environments with Applications to Root-Finding Problems**
Sondre Glimsdal, Ole-Christoffer Granmo

20(53):1–32 **Tunability: Importance of Hyperparameters of Machine Learning Algorithms**
Philipp Probst, Anne-Laure Boulesteix, Bernd Bischl

20(54):1–31 **Deep Reinforcement Learning for Swarm Systems**
Maximilian Hüttenrauch, Adrian Šošić, Gerhard Neumann

20(55):1–21 **Neural Architecture Search: A Survey**
Thomas Elsken, Jan Hendrik Metzen, Frank Hutter

20(56):1–23 **Near Optimal Frequent Directions for Sketching Dense and Sparse Matrices**
Zengfeng Huang

20(57):1–31 **Multi-class Heterogeneous Domain Adaptation**
Joey Tianyi Zhou, Ivor W. Tsang, Sinno Jialin Pan, Mingkui Tan

20(58):1–49 **The Common-directions Method for Regularized Empirical Risk Minimization**
Po-Wei Wang, Ching-pei Lee, Chih-Jen Lin

20(59):1–36 **Kernel Approximation Methods for Speech Recognition**
Avner May, Alireza Bagheri Garakani, Zhiyun Lu, Dong Guo, Kuan Liu, Aurélien Bellet, Linxi Fan, Michael Collins, Daniel Hsu, Brian Kingsbury, Michael Picheny, Fei Sha

20(60):1–49 **Robust Estimation of Derivatives Using Locally Weighted Least Absolute Deviation Regression**
WenWu Wang, Ping Yu, Lu Lin, Tiejun Tong

20(61):1–42 **The Sup-norm Perturbation of HOSVD and Low Rank Tensor Denoising**
Dong Xia, Fan Zhou

20(62):1–37 **Multi-scale Online Learning: Theory and Applications to Online Auctions and Pricing**
Sébastien Bubeck, Nikhil R. Devanur, Zhiyi Huang, Rad Niazadeh

20(63):1–17 **Nearly-tight VC-dimension and Pseudodimension Bounds for Piecewise Linear Neural Networks**
Peter L. Bartlett, Nick Harvey, Christopher Liaw, Abbas Mehrabian

20(64):1–32 **A Representer Theorem for Deep Kernel Learning**
Bastian Bohn, Michael Griebel, Christian Rieger

20(65):1–50 **Active Learning for Cost-Sensitive Classification**
Akshay Krishnamurthy, Alekh Agarwal, Tzu-Kuo Huang, Hal Daumé III, John Langford

20(66):1–38 **Proximal Distance Algorithms: Theory and Practice**
Kevin L. Keys, Hua Zhou, Kenneth Lange

20(67):1–28 **Learnability of Solutions to Conjunctive Queries**
Hubie Chen, Matthew Valeriote

20(68):1–55 **Variance-based Regularization with Convex Objectives**
John Duchi, Hongseok Namkoong

20(69):1–39 **On Consistent Vertex Nomination Schemes**
Vince Lyzinski, Keith Levin, Carey E. Priebe

20(70):1–33 **Semi-Analytic Resampling in Lasso**
Tomoyuki Obuchi, Yoshiyuki Kabashima

20(71):1–42 **Lazifying Conditional Gradient Algorithms**
Gábor Braun, Sebastian Pokutta, Daniel Zink

20(72):1–47 **Redundancy Techniques for Straggler Mitigation in Distributed Optimization and Learning**
Can Karakus, Yifan Sun, Suhas Diggavi, Wotao Yin

20(73):1–46 **Analysis of Langevin Monte Carlo via Convex Optimization**
Alain Durmus, Szymon Majewski, Błażej Miasojedow

20(74):1–25 **Deep Optimal Stopping**
Sebastian Becker, Patrick Cheridito, Arnulf Jentzen

20(75):1–42 **Fairness Constraints: A Flexible Approach for Fair Classification**
Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, Krishna P. Gummadi

20(76):1–70 **Generalized Score Matching for Non-Negative Data**
Shiqing Yu, Mathias Drton, Ali Shojaie

20(77):1–33 **Nonuniformity of P-values Can Occur Early in Diverging Dimensions**
Yingying Fan, Emre Demirkaya, Jinchi Lv

20(78):1–39 **Prediction Risk for the Horseshoe Regression**
Anindya Bhadra, Jyotishka Datta, Yunfan Li, Nicholas G. Polson, Brandon Willard

20(79):1–33 **Quantification Under Prior Probability Shift: the Ratio Estimator and its Extensions**
Afonso Fernandes Vaz, Rafael Izbicki, Rafael Bassi Stern

20(80):1–37 **Learning to Match via Inverse Optimal Transport**
Ruilin Li, Xiaojing Ye, Haomin Zhou, Hongyuan Zha

20(81):1–8 **Tight Lower Bounds on the VC-dimension of Geometric Set Systems**
Mónika Csikós, Nabil H. Mustafa, Andrey Kupavskii

20(82):1–5 **SMART: An Open Source Data Labeling Platform for Supervised Learning**
Rob Chew, Michael Wenger, Caroline Kery, Jason Nance, Keith Richards, Emily Hadley, Peter Baumgartner

20(83):1–28 **On the optimality of the Hedge algorithm in the stochastic regime**
Jaouad Mourtada, Stéphane Gaïffas

20(84):1–40 **Differentiable Game Mechanics**
Alistair Letcher, David Balduzzi, Sébastien Racanière, James Martens, Jakob Foerster, Karl Tuyls, Thore Graepel

20(85):1–35 **Bayesian Space-Time Partitioning by Sampling and Pruning Spanning Trees**
Leonardo V. Teixeira, Renato M. Assunção, Rosangela H. Loschi

20(86):1–62 **Streaming Principal Component Analysis From Incomplete Data**
Armin Eftekhari, Gregory Ongie, Laura Balzano, Michael B. Wakin

20(87):1–30 **An asymptotic analysis of distributed nonparametric methods**
Botond Szabó, Harry van Zanten

20(88):1–26 **Model Selection via the VC Dimension**
Merlin Mpoudeu, Bertrand Clarke

20(89):1–43 **Dependent relevance determination for smooth and structured sparse regression**
Anqi Wu, Oluwasanmi Koyejo, Jonathan Pillow

20(90):1–56 **A Particle-Based Variational Approach to Bayesian Non-negative Matrix Factorization**
Muhammad A Masood, Finale Doshi-Velez

20(91):1–39 **Best Arm Identification for Contaminated Bandits**
Jason Altschuler, Victor-Emmanuel Brunel, Alan Malek

20(92):1–6 **AffectiveTweets: a Weka Package for Analyzing Affect in Tweets**
Felipe Bravo-Marquez, Eibe Frank, Bernhard Pfahringer, Saif M. Mohammad

Part B

20(93):1–8 **iNNvestigate Neural Networks!**
Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T. Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, Pieter-Jan Kindermans

20(94):1–34 **Simultaneous Private Learning of Multiple Concepts**
Mark Bun, Kobbi Nissim, Uri Stemmer

20(95):1–41 **High-Dimensional Poisson Structural Equation Model Learning via ℓ_1-Regularized Regression**
Gunwoong Park, Sion Park

20(96):1–7 **PyOD: A Python Toolbox for Scalable Outlier Detection**
Yue Zhao, Zain Nasrullah, Zheng Li

20(97):1–22 **Relative Error Bound Analysis for Nuclear Norm Regularized Matrix Completion**
Lijun Zhang, Tianbao Yang, Rong Jin, Zhi-Hua Zhou

20(98):1–63 **Adaptive Geometric Multiscale Approximations for Intrinsically Low-dimensional Data**
Wenjing Liao, Mauro Maggioni

20(99):1–51 **Change Surfaces for Expressive Multidimensional Changepoints and Counterfactual Prediction**
William Herlands, Daniel B. Neill, Hannes Nickisch, Andrew Gordon Wilson

20(100):1–31 **Hamiltonian Monte Carlo with Energy Conserving Subsampling**
Khue-Dung Dang, Matias Quiroz, Robert Kohn, Minh-Ngoc Tran, Mattias Villani

20(101):1–43 **Low Permutation-rank Matrices: Structural Properties and Noisy Completion**
Nihar B. Shah, Sivaraman Balakrishnan, Martin J. Wainwright

20(102):1–56 **Non-Convex Matrix Completion and Related Problems via Strong Duality**
Maria-Florina Balcan, Yingyu Liang, Zhao Song, David P. Woodruff, Hongyang Zhang

20(103):1–68 **Regularization via Mass Transportation**
Soroosh Shafieezadeh-Abadeh, Daniel Kuhn, Peyman Mohajerin Esfahani

20(104):1–34 **Complete Search for Feature Selection in Decision Trees**
Salvatore Ruggieri

20(105):1–23 **Optimal Transport: Fast Probabilistic Approximation with Exact Solvers**
Max Sommerfeld, Jörn Schrieber, Yoav Zemel, Axel Munk

20(106):1–25 **Solving the OSCAR and SLOPE Models Using a Semismooth Newton-Based Augmented Lagrangian Method**
Ziyan Luo, Defeng Sun, Kim-Chuan Toh, Naihua Xiu

20(107):1–34 **Scalable Interpretable Multi-Response Regression via SEED**
Zemin Zheng, M. Taha Bahadori, Yan Liu, Jinchi Lv

20(108):1–30 **Maximum Likelihood for Gaussian Process Classification and Generalized Linear Mixed Models under Case-Control Sampling**
Omer Weissbrod, Shachar Kaufman, David Golan, Saharon Rosset

20(109):1–30 **Learning Unfaithful K-separable Gaussian Graphical Models**
De Wen Soh, Sekhar Tatikonda

20(110):1–30 **A Representer Theorem for Deep Neural Networks**
Michael Unser

20(111):1–40 **An Efficient Two Step Algorithm for High Dimensional Change Point Regression Models Without Grid Search**
Abhishek Kaul, Venkata K. Jandhyala, Stergios B. Fotopoulos

20(112):1–49 **Measuring the Effects of Data Parallelism on Neural Network Training**
Christopher J. Shallue, Jaehoon Lee, Joseph Antognini, Jascha Sohl-Dickstein, Roy Frostig, George E. Dahl

20(113):1–41 **Distributed Inference for Linear Support Vector Machine**
Xiaozhou Wang, Zhuoyi Yang, Xi Chen, Weidong Liu

20(114):1–34 **Sharp Restricted Isometry Bounds for the Inexistence of Spurious Local Minima in Nonconvex Matrix Recovery**
Richard Y. Zhang, Somayeh Sojoudi, Javad Lavaei

20(115):1–58 **Learning Attribute Patterns in High-Dimensional Structured Latent Attribute Models**
Yuqi Gu, Gongjun Xu

20(116):1–42 **Graph Reduction with Spectral and Cut Guarantees**
Andreas Loukas

20(117):1–63 **Generic Inference in Latent Gaussian Process Models**
Edwin V. Bonilla, Karl Krauth, Amir Dezfouli

20(118):1–34 **Binarsity: a penalization for one-hot encoded features in linear supervised learning**
Mokhtar Z. Alaya, Simon Bussy, Stéphane Gaïffas, Agathe Guilloux

20(119):1–38 **Layer-Wise Learning Strategy for Nonparametric Tensor Product Smoothing Spline Regression and Graphical Models**
Kean Ming Tan, Junwei Lu, Tong Zhang, Han Liu

20(120):1–49 **Ivanov-Regularised Least-Squares Estimators over Large RKHSs and Their Interpolation Spaces**
Stephen Page, Steffen Grünewälder

20(121):1–39 **Scaling Up Sparse Support Vector Machines by Simultaneous Feature and Sample Reduction**
Bin Hong, Weizhong Zhang, Wei Liu, Jieping Ye, Deng Cai, Xiaofei He, Jie Wang

20(122):1–55 **Approximate Profile Maximum Likelihood**
Dmitri S. Pavlichin, Jiantao Jiao, Tsachy Weissman

20(123):1–26 **ADMMBO: Bayesian Optimization with Unknown Constraints using ADMM**
Setareh Ariafar, Jaume Coll-Font, Dana Brooks, Jennifer Dy

20(124):1–62 **Deep Exploration via Randomized Value Functions**
Ian Osband, Benjamin Van Roy, Daniel J. Russo, Zheng Wen

20(125):1–5 **ORCA: A Matlab/Octave Toolbox for Ordinal Regression**
Javier Sánchez-Monedero, Pedro A. Gutiérrez, María Pérez-Ortiz

20(126):1–45 **Learning Representations of Persistence Barcodes**
Christoph D. Hofer, Roland Kwitt, Marc Niethammer

20(127):1–32 **Causal Learning via Manifold Regularization**
Steven M. Hill, Chris J. Oates, Duncan A. Blythe, Sach Mukherjee

20(128):1–73 **Unsupervised Basis Function Adaptation for Reinforcement Learning**
Edward Barker, Charl Ras

20(129):1–30 **Time-to-Event Prediction with Neural Networks and Cox Regression**
Håvard Kvamme, Ørnulf Borgan, Ida Scheel

20(130):1–47 **Logical Explanations for Deep Relational Machines Using Relevance Information**
Ashwin Srinivasan, Lovekesh Vig, Michael Bain

20(131):1–27 **Decoupling Sparsity and Smoothness in the Dirichlet Variational Autoencoder Topic Model**
Sophie Burkhardt, Stefan Kramer

20(132):1–59 **More Efficient Estimation for Logistic Regression with Optimal Subsamples**
HaiYing Wang

20(133):1–34 **Spurious Valleys in One-hidden-layer Neural Network Optimization Landscapes**
Luca Venturi, Afonso S. Bandeira, Joan Bruna

20(134):1–47 **Stochastic Variance-Reduced Cubic Regularization Methods**
Dongruo Zhou, Pan Xu, Quanquan Gu

20(135):1–36 **Gaussian Processes with Linear Operator Inequality Constraints**
Christian Agrell

20(136):1–37 **Local Regularization of Noisy Point Clouds: Improved Global Geometric Estimates and Data Analysis**
Nicolás García Trillos, Daniel Sanz-Alonso, Ruiyi Yang

20(137):1–68 **Multiclass Boosting: Margins, Codewords, Losses, and Algorithms**
Mohammad Saberian, Nuno Vasconcelos

20(138):1–29 **Generalized Maximum Entropy Estimation**
Tobias Sutter, David Sutter, Peyman Mohajerin Esfahani, John Lygeros

20(139):1–62 **Decentralized Dictionary Learning Over Time-Varying Digraphs**
Amir Daneshmand, Ying Sun, Gesualdo Scutari, Francisco Facchinei, Brian M. Sadler

20(140):1–81 **Nonparametric Bayesian Aggregation for Massive Data**
Zuofeng Shang, Botao Hao, Guang Cheng

20(141):1–43 **Provably Accurate Double-Sparse Coding**
Thanh V. Nguyen, Raymond K. W. Wong, Chinmay Hegde

20(142):1–39 **Model-free Nonconvex Matrix Completion: Local Minima Analysis and Applications in Memory-efficient Kernel PCA**
Ji Chen, Xiaodong Li

20(143):1–57 **Minimal Sample Subspace Learning: Theory and Algorithms**
Zhenyue Zhang, Yuqing Xia

20(144):1–30 **Convergence of Gaussian Belief Propagation Under General Pairwise Factorization: Connecting Gaussian MRF with Pairwise Linear Gaussian Model**
Bin Li, Yik-Chung Wu

20(145):1–30 **Bayesian Optimization for Policy Search via Online-Offline Experimentation**
Benjamin Letham, Eytan Bakshy

20(146):1–33 **Characterizing the Sample Complexity of Pure Private Learners**
Amos Beimel, Kobbi Nissim, Uri Stemmer

20(147):1–50 **Robustifying Independent Component Analysis by Adjusting for Group-Wise Stationary Noise**
Niklas Pfister, Sebastian Weichwald, Peter Bühlmann, Bernhard Schölkopf

20(148):1–43 **Collective Matrix Completion**
Mokhtar Z. Alaya, Olga Klopp

20(149):1–24 **On Asymptotic and Finite-Time Optimality of Bayesian Predictors**
Daniil Ryabko

20(150):1–75 **Learning Optimized Risk Scores**
Berk Ustun, Cynthia Rudin

20(151):1–49 **Nonparametric Estimation of Probability Density Functions of Random Persistence Diagrams**
Vasileios Maroulas, Joshua L Mike, Christopher Oballe

20(152):1–44 **High-dimensional Varying Index Coefficient Models via Stein's Identity**
Sen Na, Zhuoran Yang, Zhaoran Wang, Mladen Kolar

20(153):1–33 **Approximation Algorithms for Stochastic Clustering**
David G. Harris, Shi Li, Thomas Pensyl, Aravind Srinivasan, Khoa Trinh

20(154):1–52 **Convergence Guarantees for a Class of Non-convex and Non-smooth Optimization Problems**
Koulik Khamaru, Martin J. Wainwright

20(155):1–35 **Quantifying Uncertainty in Online Regression Forests**
Theodore Vasiloudis, Gianmarco De Francisci Morales, Henrik Boström

20(156):1–8 **SimpleDet: A Simple and Versatile Distributed Framework for Object Detection and Instance Recognition**
Yuntao Chen, Chenxia Han, Yanghao Li, Zehao Huang, Yi Jiang, Naiyan Wang, Zhaoxiang Zhang

20(157):1–28 **Simultaneous Phase Retrieval and Blind Deconvolution via Convex Programming**
Ali Ahmed, Alireza Aghasi, Paul Hand

20(158):1–7 **GraSPy: Graph Statistics in Python**
Jaewon Chung, Benjamin D. Pedigo, Eric W. Bridgeford, Bijan K. Varjavand, Hayden S. Helm, Joshua T. Vogelstein

20(159):1–31 **Optimal Convergence Rates for Convex Distributed Optimization in Networks**
Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, Laurent Massoulié

20(160):1–56 **Learning by Unsupervised Nonlinear Diffusion**
Mauro Maggioni, James M. Murphy

20(161):1–44 **Sparse Kernel Regression with Coefficient-based ℓ_q-regularization**
Lei Shi, Xiaolin Huang, Yunlong Feng, Johan A.K. Suykens

20(162):1–56 **A Kernel Multiple Change-point Algorithm via Model Selection**
Sylvain Arlot, Alain Celisse, Zaid Harchaoui

20(163):1–42 **Two-Layer Feature Reduction for Sparse-Group Lasso via Decomposition of Convex Sets**
Jie Wang, Zhanqiu Zhang, Jieping Ye

20(164):1–31 **The Reduced PC-Algorithm: Improved Causal Structure Learning in Large Random Networks**
Arjun Sondhi, Ali Shojaie

20(165):1–37 **On the Convergence of Gaussian Belief Propagation with Nodes of Arbitrary Size**
Francois Kamper, Sarel J. Steel, Johan A. du Preez

20(166):1–40 **Unsupervised Evaluation and Weighted Aggregation of Ranked Classification Predictions**
Mehmet Eren Ahsen, Robert M Vogel, Gustavo A Stolovitzky

20(167):1–46 **Stochastic Canonical Correlation Analysis**
Chao Gao, Dan Garber, Nathan Srebro, Jialei Wang, Weiran Wang

20(168):1–70 **Determinantal Point Processes for Coresets**
Nicolas Tremblay, Simon Barthelmé, Pierre-Olivier Amblard

20(169):1–26 **Embarrassingly Parallel Inference for Gaussian Processes**
Michael Minyi Zhang, Sinead A. Williamson

20(170):1–50 **DBSCAN: Optimal Rates For Density-Based Cluster Estimation**
Daren Wang, Xinyang Lu, Alessandro Rinaldo

20(171):1–37 **Shared Subspace Models for Multi-Group Covariance Estimation**
Alexander M. Franks, Peter Hoff

20(172):1–59 **Optimization with Non-Differentiable Constraints with Applications to Fairness, Recall, Churn, and Other Goals**
Andrew Cotter, Heinrich Jiang, Maya Gupta, Serena Wang, Taman Narayan, Seungil You, Karthik Sridharan

20(173):1–27 **Fast Automatic Smoothing for Generalized Additive Models**
Yousra El-Bachir, Anthony C. Davison

20(174):1–42 **Learning Overcomplete, Low Coherence Dictionaries with Linear Inference**
Jesse A. Livezey, Alejandro F. Bujan, Friedrich T. Sommer

20(175):1–6 **DataWig: Missing Value Imputation for Tables**
Felix Biessmann, Tammo Rukat, Phillipp Schmidt, Prathik Naidu, Sebastian Schelter, Andrey Taptunov, Dustin Lange, David Salinas

20(176):1–49 **New Convergence Aspects of Stochastic Gradient Algorithms**
Lam M. Nguyen, Phuong Ha Nguyen, Peter Richtárik, Katya Scheinberg, Martin Takáč, Marten van Dijk

20(177):1–81 **All Models are Wrong, but Many are Useful: Learning a Variable's Importance by Studying an Entire Class of Prediction Models Simultaneously**
Aaron Fisher, Cynthia Rudin, Francesca Dominici

20(178):1–29 **Morpho-MNIST: Quantitative Assessment and Diagnostics for Representation Learning**
Daniel C. Castro, Jeremy Tan, Bernhard Kainz, Ender Konukoglu, Ben Glocker

20(179):1–62 **Differentiable reservoir computing**
Lyudmila Grigoryeva, Juan-Pablo Ortega

20(180):1–7 **DPPy: DPP Sampling with Python**
Guillaume Gautier, Guillermo Polito, Rémi Bardenet, Michal Valko

20(181):1–23 **Neural Empirical Bayes**
Saeed Saremi, Aapo Hyvärinen

20(182):1–46 **Model Selection in Bayesian Neural Networks via Horseshoe Priors**
Soumya Ghosh, Jiayu Yao, Finale Doshi-Velez

20(183):1–42 **Log-concave sampling: Metropolis-Hastings algorithms are fast**
Raaz Dwivedi, Yuansi Chen, Martin J. Wainwright, Bin Yu

20(184):1–25 **Why do deep convolutional networks generalize so poorly to small image transformations?**
Aharon Azulay, Yair Weiss

Journal of Machine Learning Research 20 (2019) 1-30 Submitted 5/15; Revised 10/15; Published 1/19

Adaptation Based on Generalized Discrepancy

Corinna Cortes corinna@google.com
Google Research, 111 8th ave, New York, NY 10011

Mehryar Mohri mohri@cims.nyu.edu
Andrés Muñoz Medina munoz@cims.nyu.edu
Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, NY 10012

Editor: Multi Task Learning

Abstract

We present a new algorithm for domain adaptation improving upon a discrepancy minimization algorithm, (DM), previously shown to outperform a number of algorithms for this problem. Unlike many previously proposed solutions for domain adaptation, our algorithm does not consist of a fixed reweighting of the losses over the training sample. Instead, the reweighting depends on the hypothesis sought. The algorithm is derived from a less conservative notion of discrepancy than the DM algorithm called *generalized discrepancy*. We present a detailed description of our algorithm and show that it can be formulated as a convex optimization problem. We also give a detailed theoretical analysis of its learning guarantees which helps us select its parameters. Finally, we report the results of experiments demonstrating that it improves upon discrepancy minimization.

Keywords: domain adaptation, learning theory

1. Introduction

A standard assumption in statistical learning theory and PAC learning is that training and test samples are drawn from the same distribution (Vapnik, 1998; Valiant, 1984). In practice, however, this assumption often does not hold: the source and target distributions may somewhat differ. This problem is known as *domain adaptation* and arises in a variety of applications such as natural language processing and computer vision (Dredze et al., 2007; Blitzer et al., 2007; Jiang and Zhai, 2007; Leggetter and Woodland, 1995; Martínez, 2002; Hoffman et al., 2014). The domain adaptation problem may appear when the distributions over the instance space differ, the so-called *covariate shift* problem, or when the labeling functions associated with each domain disagree. In practice, a combination of both issues occurs and, for adaptation to succeed, the divergence between the two domains needs to be relatively small. This is clear for the labeling functions since, if the learner receives source labels that are vastly different from the target ones, no learning algorithm can generalize well to the target domain. The same holds when input distributions largely differ.

This intuition was formalized by Ben-David et al. (2010) and Ben-David and Urner (2012) who showed that even in the favorable scenario where the source and target distribution admit the same support, a sample of size in the order of that of the support is needed in order to solve the domain adaptation problem. As the authors point out, the domain adaptation problem becomes intractable when the labeling function for the training data is vastly different from the labeling function used for testing. On the other hand, when some similarity between domains exist, it has been empirically and theoretically shown that adaptation algorithms can be beneficial and in fact a large number of algorithms for this task have been proposed over the past decade. The large majority of them fall in one of the following paradigms:

1. **Learning a new feature representation**. The core idea behind these algorithms is to map the source and target data into a new feature space where the difference between source and target distributions is reduced. Transfer Component Analysis (TCA) (Pan et al., 2011) and the work on Frustratingly Easy Domain Adaptation (FE) (Daumé III, 2007) belong to this family of algorithms. Whereas some empirical evidence of the effectiveness of these algorithms exists in the literature, to the best of our knowledge, no work has been done to provide learning guarantees for these algorithms.
2. **Reweighting**. Originated in the Statistics literature on sample bias correction, these techniques attempt to correct the difference between distributions by multiplying the loss at each training example by a positive weight. Most of the classical algorithms such as KMM (Huang et al., 2006), KLIEP (Sugiyama et al., 2007) and a two-step algorithm by Bickel et al. (2007) fall in this category.

The main focus of this work will be on the latter. A common trait shared by most algorithms in this category is that their reweighting schemes are based on the minimization of a divergence measure between the empirical source and target distributions. For instance, the KL-divergence in the case of KLIEP and the Maximum Mean Discrepancy (MMD) (Gretton et al., 2012) for KMM. The guarantees of these algorithms are therefore given as a function of the chosen divergence. The main drawback of these measures is that they do not take into account the hypothesis set or the loss function, both crucial components of any learning algorithm. In contrast, the *discrepancy* introduced by Mansour et al. (2009) and further studied by Cortes and Mohri (2011) is a measure of the divergence between distributions tailored to domain adaptation that precisely takes into account both the loss function and the hypothesis set. The $d_{\mathcal{A}}$-distance, introduced by Devroye et al. (1996)[pp. 271-272] under the name of *generalized Kolmogorov-Smirnov distance*, later by Ben-David et al. (2006), coincides with the discrepancy when the binary loss function is used. The discrepancy is a pivotal concept used in the analysis of several adaptation scenarios: the $\mathcal{Y}$-discrepancy or *integral probability metric* (Zhang et al., 2012) was successfully used by Mohri and Muñoz (2012) to provide tight learning guarantees for the related task of learning with drifting distributions, whereas a modified version of the discrepancy was used by Germain et al. (2013) to study the problem of domain adaptation in a PAC-Bayesian setting. The discrepancy-based generalization bounds given by Mansour et al. (2009) motivated a discrepancy minimization (DM) algorithm (Cortes and Mohri, 2013), which attempts to minimize said bounds. Besides its favorable theoretical guarantees, this algorithm was shown to perform well in a number of adaptation tasks and to match or outperform several other algorithms such as KMM, KLIEP and the aforementioned two stage algorithm by Bickel et al. (2007).

One shortcoming of the DM algorithm, however, is that it seeks to reweight the loss on the training samples to minimize a quantity defined as the maximum over *all* pairs of hypotheses, including hypotheses that the learning algorithm might not ever consider as candidates. Thus, the algorithm tends to be too conservative on its choice of weights. We

©2019 Corinna Cortes, Mehryar Mohri, Andrés Muñoz Medina.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/15-192.html.

present an alternative theoretically well founded algorithm for domain adaptation that is based on minimizing a finer quantity, the *generalized discrepancy*, and that seeks to improve upon DM. Unlike the DM algorithm, our algorithm does not consist of a *fixed* reweighting of the losses over the training sample. Instead, the weights assigned to training sample losses vary as a function of the hypothesis h. This helps us ensure that for every hypothesis, h, the empirical loss on the source distribution is as close as possible to the empirical loss on the target distribution for that particular h.

We describe the learning scenario considered (Section 2), then present a detailed description of our algorithm and show that it can be formulated as a convex optimization problem (Section 3). Next, we analyze the theoretical properties of our algorithm, which guide us in choosing the surrogate hypothesis set defining our algorithm (Section 4). In Section 5, we further analyze the optimization problem defining our algorithm and derive an equivalent form that can be handled by a standard convex optimization solver. In Section 6, we report the results of experiments demonstrating that our algorithm improves upon the DM algorithm in several tasks.

2. Learning Scenario

This section defines the learning scenario of domain adaptation we consider, which coincides with that of Ben-David et al. (2006) or Mansour et al. (2009); Cortes and Mohri (2013). We first introduce the definitions and concepts needed for the following sections. For the most part, we follow the definitions and notation of Cortes and Mohri (2013).

Let $\mathcal{X}$ denote the input space and $\mathcal{Y} \subseteq \mathbb{R}$ the output space. We define a *domain* as a pair formed by a distribution over $\mathcal{X}$ and a target labeling function mapping from $\mathcal{X}$ to $\mathcal{Y}$. Throughout the paper, (Q, f_Q) denotes the *source domain* and (P, f_P) the *target domain* with Q the source and P the target distribution over $\mathcal{X}$ and with $f_Q, f_P \colon \mathcal{X} \to \mathcal{Y}$ the source and target labeling functions, respectively.

In the scenario of *domain adaptation* we consider, the learner receives two samples: a labeled sample of m points from the source domain $\mathcal{S} = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$ with $x_1, \ldots, x_m$ drawn i.i.d. according to Q and $y_i = f_Q(x_i)$ for $i \in [1, m]$; and an unlabeled sample $\mathcal{T} = (x'_1, \ldots, x'_n) \in \mathcal{X}^n$ of size n drawn i.i.d. according to the target distribution P. We denote by $\widehat{Q}$ the empirical distribution corresponding to the (unlabeled) sample $\mathcal{S}_{\mathcal{X}} = (x_1, \ldots, x_m)$ and by $\widehat{P}$ the empirical distribution corresponding to $\mathcal{T}$. We will be in fact more interested in the scenario commonly encountered in practice where, in addition to these two samples, the learner receives a small amount of labeled data $\mathcal{T}' = ((x''_1, y''_1), \ldots, (x''_s, y''_s)) \in (\mathcal{X} \times \mathcal{Y})^s$ from the target domain.

We consider a loss function $L \colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ jointly convex in its two arguments. The L_p losses commonly used in regression and defined by $L_p(y, y') = |y' - y|^p$ for $p \geq 1$ are special instances of this definition. For any two functions $h, h' \colon \mathcal{X} \to \mathcal{Y}$ and any distribution D over $\mathcal{X}$, we denote by $\mathcal{L}_D(h, h')$ the expected loss of $h(x)$ and $h'(x)$: $\mathcal{L}_D(h, h') = \mathrm{E}_{x \sim D}[L(h(x), h'(x))]$. The learning problem consists of selecting a hypothesis h out of a hypothesis set H with a small expected loss $\mathcal{L}_P(h, f_P)$ with respect to the target domain. We further extend this notation to arbitrary functions $\mathsf{q} \colon \mathcal{X} \to \mathbb{R}$ with a finite support as follows: $\mathcal{L}_{\mathsf{q}}(h, h') = \sum_{x \in \mathcal{X}} q(x) L(h(x), h'(x))$.

$\mathcal{X}$	Input space	$\mathcal{Y}$	Output space
P	Target distribution	Q	Source distribution
$\widehat{P}$	Empirical target distribution	$\widehat{Q}$	Empirical source distribution
$\mathcal{T}$	Target unlabeled sample	$\mathcal{S}$	Labeled source sample
$\mathcal{T}'$	Small target labeled sample	$\mathcal{S}_{\mathcal{X}}$	Unlabeled source sample
f_P	Target labeling function	f_Q	Source labeling function
$\mathcal{L}_P(h, f_P)$	Expected target loss	$\mathcal{L}_Q(h, f_Q)$	Expected source loss
$\mathcal{L}_{\widehat{P}}(h, f_P)$	Empirical target loss	$\mathcal{L}_{\widehat{Q}}(h, f_Q)$	Empirical source loss
$\mathrm{disc}(P,Q)$	Discrepancy	$\mathrm{DISC}(\widehat{P},\mathrm{U})$	Generalized discrepancy
$\mathrm{disc}_{H''}(P,Q)$	Local Discrepancy	$\mathrm{disc}_{\mathcal{Y}}(P,Q)$	$\mathcal{Y}$-discrepancy
$\mathsf{q}_{\min}$	DM solution	Q_h	GDM solution

Table 1: Notation table.

3. Algorithm

In this section, we describe our new adaptation algorithm. We first review some related previous work. Next, we present the key idea behind our algorithm and derive its general form, and finally, formulate it as a convex optimization problem.

3.1. Previous Work

It was shown by Mansour et al. (2009) and Cortes and Mohri (2011) (see also the $d_{\mathcal{A}}$-*distance* (Ben-David et al., 2006) in the case of binary loss for classification) that a key measure of the difference of two distributions in the context of adaptation is the *discrepancy*. Given a hypothesis set H, the discrepancy, disc, between two distributions P and Q over $\mathcal{X}$ is defined by:

$$\mathrm{disc}(P, Q) = \max_{h, h' \in H} \left| \mathcal{L}_P(h', h) - \mathcal{L}_Q(h', h) \right|. \tag{1}$$

The discrepancy has several advantages over other common divergence measures such as the L_1 distance. We refer the reader to (Medina, 2015) for a detailed discussion on this subject. Several generalization bounds for adaptation in terms of the discrepancy have been given in the past (Ben-David et al., 2006; Mansour et al., 2009; Cortes and Mohri, 2011, 2013). including pointwise guarantees in the case of kernel-based regularization algorithms, which includes algorithms such as support vector machines (SVM), kernel ridge regression, or support vector regression (SVR). The bounds given in (Mansour et al., 2009) motivated a *discrepancy minimization* algorithm. Given a positive semi-definite (PSD) kernel K, the hypothesis returned by the algorithm is the solution of the following optimization problem

$$\min_{h \in \mathbb{H}} \quad \lambda \|h\|_K^2 + \mathcal{L}_{\mathsf{q}_{\min}}(h, f_Q), \tag{2}$$

where $\|\cdot\|_K$ is the norm in the reproducing Hilbert space $\mathbb{H}$ induced by the kernel K and $\mathsf{q}_{\min}$ is a distribution over the support of $\widehat{Q}$ such that $\mathsf{q}_{\min} = \mathrm{argmin}_{\mathsf{q} \in \mathcal{Q}} \mathrm{disc}(\mathsf{q}, \widehat{P})$, where $\mathcal{Q} = [0, 1]^{\mathcal{S}_{\mathcal{X}}}$ is the set of all distributions defined over the support of $\widehat{Q}$. Besides its theoretical motivation, this algorithm has been shown to outperform several other algorithms in a series of experiments carried out by Cortes and Mohri (2013).

Observe that, by definition, the objective function optimized by $\mathsf{q}_{\min}$ corresponds to a maximum over all pairs of hypotheses. But, the maximizing pair of hypotheses may not be among the candidates ever considered by the learning algorithm. Thus, a learning algorithm based on discrepancy minimization tends to be too conservative.

3.2. Main Idea

From here on we assume the algorithm selected by the learner is an instance of a regularized risk minimization algorithm over the Hilbert space $\mathbb{H}$ induced by a PSD kernel K. With knowledge of the target labels, these algorithms return a hypothesis h^* solution of $\min_{h\in\mathbb{H}} F(h)$ where

$$F(h) = \lambda\|h\|_K^2 + \mathcal{L}_{\widehat{P}}(h, f_P), \tag{3}$$

where $\lambda \geq 0$ is a regularization parameter. Thus, h^* can be viewed as the *ideal hypothesis*.

In view of that, we can formulate our objective, in the *presence* of a domain adaptation problem, as that of finding a hypothesis h whose loss $\mathcal{L}_P(h, f_P)$ with respect to the target domain is as close as possible to $\mathcal{L}_P(h^*, f_P)$. To do so, we will seek in fact a hypothesis h that is as close as possible to h^*, which would imply the closeness of the losses with respect to the target domains. We do not have access to f_P and can only access the labels of the training sample $\mathcal{S}$. Thus, we must resort to using in our objective function, instead of $\mathcal{L}_{\widehat{P}}(h, f_P)$, a reweighted empirical loss over the training sample $\mathcal{S}$. The main idea behind our algorithm is to define, for any $h \in \mathbb{H}$, a reweighting function $\mathsf{Q}_h \colon \mathcal{S}_{\mathcal{X}} = \{x_1, \ldots, x_m\} \to \mathbb{R}$ such that the objective function G defined for all $h \in \mathbb{H}$ by

$$G(h) = \lambda\|h\|_K^2 + \mathcal{L}_{\mathsf{Q}_h}(h, f_Q) \tag{4}$$

is uniformly close to F, thereby resulting in close minimizers. Since the first term of (3) and (4) coincide, the idea consists equivalently of seeking Q_h such that $\mathcal{L}_{\mathsf{Q}_h}(h, f_Q)$ and $\mathcal{L}_{\widehat{P}}(h, f_P)$ be as close as possible. Observe that this departs from the standard reweighting methods: instead of reweighting the training sample with some fixed set of weights, we allow the weights to vary as a function of the hypothesis h. Note that we have further relaxed the condition commonly adopted by reweighting techniques that the weights must be non-negative and sum to one.

Of course, searching for Q_h to directly minimize $|\mathcal{L}_{\mathsf{Q}_h}(h, f_Q) - \mathcal{L}_{\widehat{P}}(h, f_P)|$ is in general not possible since we do not have access to f_P, but it is instructive to consider the imaginary case where the average loss $\mathcal{L}_{\widehat{P}}(h, f_P)$ is known to us for any $h \in \mathbb{H}$. Q_h could then be determined via

$$\mathsf{Q}_h = \operatorname*{argmin}_{\mathsf{q}\in\mathcal{F}(\mathcal{S}_{\mathcal{X}},\mathbb{R})} |\mathcal{L}_{\mathsf{q}}(h, f_Q) - \mathcal{L}_{\widehat{P}}(h, f_P)|, \tag{5}$$

where $\mathcal{F}(\mathcal{S}_{\mathcal{X}}, \mathbb{R})$ is the set of real-valued functions defined over $\mathcal{S}_{\mathcal{X}}$. For any h, we can in fact select Q_h such that $\mathcal{L}_{\mathsf{Q}_h}(h, f_Q) = \mathcal{L}_{\widehat{P}}(h, f_P)$ since $\mathcal{L}_{\mathsf{q}}(h, f_Q)$ is a linear function of q. Thus, the optimization problem (5) reduces to solving a simple linear equation. With this choice of Q_h, the objective functions F and G coincide and by minimizing G we can recover the ideal solution h^*. Note that, in general, the DM algorithm could not recover that ideal solution. Even a finer discrepancy minimization algorithm exploiting the knowledge of $\mathcal{L}_{\widehat{P}}(h, f_P)$ for all h and seeking a distribution $\mathsf{q}'_{\min}$ minimizing $\max_{h\in H} |\mathcal{L}_{\mathsf{q}}(h, f_Q) - \mathcal{L}_{\widehat{P}}(h, f_P)|$ could not,

in general, recover the ideal solution since we could not have $\mathcal{L}_{\mathsf{q}'_{\min}}(h, f_Q) = \mathcal{L}_{\widehat{P}}(h, f_P)$ for all $h \in \mathbb{H}$.

Of course, $\mathcal{L}_{\widehat{P}}(h, f_P)$ is not accessible since the sample $\mathcal{T}$ is unlabeled. Instead, we will consider a non-empty convex set of candidate hypotheses $H'' \subseteq H$ that could contain a good approximation of f_P. Using H'' as a set of surrogate labeling functions leads to the following definition of Q_h instead of (5):

$$\mathsf{Q}_h = \operatorname*{argmin}_{\mathsf{q}\in\mathcal{F}(\mathcal{S}_{\mathcal{X}},\mathbb{R})} \max_{h''\in H''} |\mathcal{L}_{\mathsf{q}}(h, f_Q) - \mathcal{L}_{\widehat{P}}(h, h'')|. \tag{6}$$

The choice of the subset H'' is of course key. Our choice will be based on the theoretical analysis of Section 4. Nevertheless, we now present the formulation of the optimization problem for an arbitrary choice of the convex subset H''.

Proposition 1 *For any $h \in \mathbb{H}$, let Q_h be defined by* (6)*. Then, the following identity holds for any $h \in \mathbb{H}$:*

$$\mathcal{L}_{\mathsf{Q}_h}(h, f_Q) = \frac{1}{2}\Big(\max_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'') + \min_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'')\Big).$$

Proof For any $h \in \mathbb{H}$, the equation $\mathcal{L}_{\mathsf{q}}(h, f_Q) = l$ with $l \in \mathbb{R}$ admits a solution $\mathsf{q} \in \mathcal{F}(\mathcal{S}_{\mathcal{X}}, \mathbb{R})$. Thus, $\{\mathcal{L}_{\mathsf{q}}(h, f_Q) : \mathsf{q} \in \mathcal{F}(\mathcal{S}_{\mathcal{X}}, \mathbb{R})\} = \mathbb{R}$ and for any $h \in \mathbb{H}$, we can write

$$\begin{aligned}
\mathcal{L}_{\mathsf{Q}_h}(h, f_Q) &= \operatorname*{argmin}_{l\in\{\mathcal{L}_{\mathsf{q}}(h,f_Q):\, \mathsf{q}\in\mathcal{F}(\mathcal{S}_{\mathcal{X}},\mathbb{R})\}} \max_{h''\in H''} |l - \mathcal{L}_{\widehat{P}}(h, h'')| \\
&= \operatorname*{argmin}_{l\in\mathbb{R}} \max_{h''\in H''} |l - \mathcal{L}_{\widehat{P}}(h, h'')| \\
&= \operatorname*{argmin}_{l\in\mathbb{R}} \max_{h''\in H''} \max\Big\{\mathcal{L}_{\widehat{P}}(h, h'') - l, l - \mathcal{L}_{\widehat{P}}(h, h'')\Big\} \\
&= \operatorname*{argmin}_{l\in\mathbb{R}} \max\Big\{\max_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'') - l, l - \min_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'')\Big\} \\
&= \frac{1}{2}\Big(\max_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'') + \min_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'')\Big),
\end{aligned}$$

since the minimizing l is obtained for $\max_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'') - l = l - \min_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'')$. ■

In view of this proposition, with our choice of Q_h based on (6), the objective function G of our algorithm (4) can be equivalently written for all $h \in \mathbb{H}$ as follows:

$$G(h) = \lambda\|h\|_K^2 + \frac{1}{2}\Big(\max_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'') + \min_{h''\in H''} \mathcal{L}_{\widehat{P}}(h, h'')\Big). \tag{7}$$

Using the fact the $\mathcal{L}_{\widehat{P}}$ is a jointly convex function, it is easy to show (see for instance Boyd and Vandenberghe, 2004) that G is in fact a convex function too.

4. Learning Guarantees

Here, we present two different types of guarantees: a tight learning bound based on the Rademacher complexity and a pointwise bound derived from a stability analysis. We further show that our algorithm is in fact minimizing this pointwise bound. As in previous work, we assume that the loss function L is μ-*admissible*.

Definition 2 *A loss function L is μ-admissible if there exists $\mu > 0$ such that the inequality*

$$|L(h(x), y) - L(h'(x), y)| \leq \mu |h(x) - h'(x)| \tag{8}$$

holds for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$ and $h', h \in H$.

The L_p losses commonly used in regression, $p \geq 1$, verify this condition (see Appendix C).

4.1. Rademacher Complexity Bounds

Definition 3 *Let $\mathcal{Z}$ be any set and G be a family of functions mapping $\mathcal{Z}$ to $\mathbb{R}$. Given a sample $\mathcal{S} = \{z_1, \ldots, z_n\} \subset \mathcal{Z}$, the empirical Rademacher complexity of G is denoted by $\widehat{\mathfrak{R}}_{\mathcal{S}}(G)$ and defined by*

$$\widehat{\mathfrak{R}}_{\mathcal{S}}(G) = \frac{1}{n} \mathop{\mathrm{E}}_{\sigma} \left[\sup_{g \in G} \sum_{i=1}^{n} \sigma_i g(z_i) \right],$$

where σ_is, called Rademacher variables, are independent random variables distributed according to the uniform distribution over $\{-1, 1\}$. The Rademacher complexity of G is defined as

$$\mathfrak{R}_n(G) = \mathop{\mathrm{E}}_{\mathcal{S}} \left[\widehat{\mathfrak{R}}_{\mathcal{S}}(G) \right].$$

Our first generalization bound is given in terms of the $\mathcal{Y}$-*discrepancy*, which is a generalization of the discrepancy distance. The $\mathcal{Y}$-discrepancy was first introduced by Mohri and Muñoz (2012) in the context of learning with drifting distributions.

Definition 4 *The $\mathcal{Y}$-discrepancy between two domains (P, f_P) and (Q, f_Q) is defined by*

$$\mathrm{disc}_{\mathcal{Y}}(P, Q) = \sup_{h \in H} \left| \mathcal{L}_Q(h, f_Q) - \mathcal{L}_P(h, f_P) \right|.$$

Note that the definition depends on the labeling functions f_P and f_Q. We do not explicitly indicate that dependency for the sake of simplicity of the notation.

We follow the analysis of (Mohri and Muñoz, 2012) to derive the following tight generalization bounds based on the notion of $\mathcal{Y}$-discrepancy.

Proposition 5 *Let $\mathcal{H}_Q$ and $\mathcal{H}_P$ be the families of functions defined as follows: $\mathcal{H}_Q := \{x \mapsto L(h(x), f_Q(x)) \colon h \in H\}$ and $\mathcal{H}_P := \{x \mapsto L(h(x), f_P(x)) \colon h \in H\}$. Define M_Q and M_P as $M_Q = \sup_{x \in \mathcal{X}, h \in H} L(h(x), f_Q(x))$ and $M_P = \sup_{x \in \mathcal{X}, h \in H} L(h(x), f_P(x))$. Then, for any $\delta > 0$,*

1. *with probability at least $1 - \delta$ over the choice of a labeled sample $\mathcal{S}$ of size m, the following inequality holds for all $h \in H$:*

$$\mathcal{L}_P(h, f_P) \leq \mathcal{L}_{\widehat{Q}}(h, f_Q) + \mathrm{disc}_{\mathcal{Y}}(P, Q) + 2\mathfrak{R}_m(\mathcal{H}_Q) + M_Q \sqrt{\frac{\log(\frac{1}{\delta})}{2m}}; \tag{9}$$

2. *with probability at least $1 - \delta$ over the choice of a sample $\mathcal{T}$ of size n, the following inequality holds for all $h \in H$ and any distribution q over a sample $\mathcal{S}_{\mathcal{X}}$:*

$$\mathcal{L}_P(h, f_P) \leq \mathcal{L}_{\mathsf{q}}(h, f_Q) + \mathrm{disc}_{\mathcal{Y}}(\widehat{P}, \mathsf{q}) + 2\mathfrak{R}_n(\mathcal{H}_P) + M_P \sqrt{\frac{\log(\frac{1}{\delta})}{2n}}. \tag{10}$$

Proof Let $\Phi(\mathcal{S})$ denote $\sup_{h \in H} \mathcal{L}_{\widehat{Q}}(h, f_Q) - \mathcal{L}_P(h, f_P)$. Changing one point in $\mathcal{S}$ changes $\Phi(\mathcal{S})$ by at most $\frac{M_Q}{m}$. Thus, by McDiarmid's inequality, we have $\mathbb{P}\big(\Phi(\mathcal{S}) - \mathrm{E}[\Phi(\mathcal{S})] > \epsilon\big) \leq e^{-\frac{2m\epsilon^2}{M_Q^2}}$. Therefore, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $h \in H$:

$$\mathcal{L}_P(h, f_P) \leq \mathcal{L}_{\widehat{Q}}(h, f_Q) + \mathrm{E}[\Phi(\mathcal{S})] + M_Q \sqrt{\frac{\log(\frac{1}{\delta})}{2m}}.$$

Next, we can bound $\mathrm{E}[\Phi(\mathcal{S})]$ as follows:

$$\begin{aligned}
\mathrm{E}[\Phi(\mathcal{S})] &= \mathrm{E}\left[\sup_{h \in H} \mathcal{L}_{\widehat{Q}}(h, f_Q) - \mathcal{L}_P(h, f_P) \right] \\
&\leq \mathrm{E}\left[\sup_{h \in H} \mathcal{L}_{\widehat{Q}}(h, f_Q) - \mathcal{L}_Q(h, f_Q) \right] + \sup_{h \in H} \mathcal{L}_Q(h, f_Q) - \mathcal{L}_P(h, f_P) \\
&\leq 2\mathfrak{R}_m(\mathcal{H}_Q) + \mathrm{disc}_{\mathcal{Y}}(P, Q),
\end{aligned}$$

where the last inequality follows from a standard symmetrization inequality in terms of the Rademacher complexity and the definition of $\mathrm{disc}_{\mathcal{Y}}(P, Q)$.

For the second bound we have, starting with a standard Rademacher complexity bound for $\mathcal{H}_P$, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $h \in H$:

$$\begin{aligned}
\mathcal{L}_P(h, f_P) &\leq \mathcal{L}_{\widehat{P}}(h, f_P) + 2\mathfrak{R}_n(\mathcal{H}_P) + M_P \sqrt{\frac{\log(\frac{1}{\delta})}{2n}} \\
&\leq \mathcal{L}_{\mathsf{q}}(h, f_Q) + \mathcal{L}_{\widehat{P}}(h, f_P) - \mathcal{L}_{\mathsf{q}}(h, f_Q) + 2\mathfrak{R}_n(\mathcal{H}_P) + M_P \sqrt{\frac{\log(\frac{1}{\delta})}{2n}}.
\end{aligned} \tag{11}$$

Moreover, by definition $\mathcal{L}_{\widehat{P}}(h, f_P) - \mathcal{L}_{\mathsf{q}}(h, f_Q) \leq \mathrm{disc}_{\mathcal{Y}}(\widehat{P}, \mathsf{q})$ for any q. Replacing this bound in (11) yields the result. ■

Observe that these bounds are tight as a function of the divergence measure (discrepancy) we use: in the absence of adaptation, the following standard Rademacher complexity learning bound holds:

$$\mathcal{L}_{\widehat{P}}(h, f_P) \leq \mathcal{L}_{\widehat{P}}(h, f_P) + 2\mathfrak{R}_n(\mathcal{H}_P) + M_P \sqrt{\frac{\log(\frac{1}{\delta})}{2n}}.$$

Our second adaptation bound differs from this inequality only by the fact that $\mathcal{L}_{\widehat{P}}(h, f_P)$ is replaced with $\mathcal{L}_{\mathsf{q}}(h, f_Q) + \mathrm{disc}_{\mathcal{Y}}(\widehat{P}, \mathsf{q})$. But, by definition of $\mathcal{Y}$-discrepancy, there exists an $h \in H$ such that $|\mathcal{L}_{\widehat{P}}(h, f_P) - \mathcal{L}_{\mathsf{q}}(h, f_Q)| = \mathrm{disc}_{\mathcal{Y}}(\widehat{P}, \mathsf{q})$. A similar analysis shows that our first bound is also tight.

Given a labeled sample $\mathcal{S}$ from the source domain, Proposition 5 suggests choosing a distribution q with support $\mathcal{S}_{\mathcal{X}}$ that minimizes the right-hand side of (10). However, the quantity $\mathrm{disc}_{\mathcal{Y}}(\widehat{P}, \mathsf{q})$ depends, by definition, on the unknown labels from the target domain and therefore cannot be minimized. Thus, we will instead upper bound the $\mathcal{Y}$-discrepancy in terms of quantities that can be estimated.

Let $\mathcal{A}(H)$ denote the set of all functions $\mathsf{U}\colon h \mapsto \mathsf{U}_h$ mapping H to $\mathcal{F}(\mathcal{S}_\mathcal{X}, \mathbb{R})$ such that for all $h \in H$, $h \mapsto \mathcal{L}_{\mathsf{U}_h}(h, f_Q)$ is a convex function. Thus, for any $h \in H$, U_h is a reweighting function defined over $\mathcal{S}_\mathcal{X}$. $\mathcal{A}(H)$ contains all constant functions U such that $\mathsf{U}_h = \mathsf{q}$ for all $h \in H$, where q is a distribution over $\mathcal{S}_\mathcal{X}$. We will abuse the notation and denote this functions also by q. By Proposition 1, $\mathcal{A}(H)$ also includes the function $\mathsf{Q}\colon h \to \mathsf{Q}_h$ used by our algorithm.

Definition 6 (Generalized discrepancy) *For any $\mathsf{U} \in \mathcal{A}(H)$, the* generalized discrepancy between $\widehat{P}$ and U *is denoted by* $\mathrm{DISC}(\widehat{P}, \mathsf{U})$ *and is defined by*

$$\mathrm{DISC}(\widehat{P}, \mathsf{U}) = \sup_{h \in H, h'' \in H''} |\mathcal{L}_{\widehat{P}}(h, h'') - \mathcal{L}_{\mathsf{U}_h}(h, f_Q)|. \tag{12}$$

We also denote by $d_1(f_P, H'')$ the L_1 distance of f_P to H'':

$$d_1(f_P, H'') = \min_{h_0 \in H''} \mathop{\mathrm{E}}_{\widehat{P}} |h_0(x) - f_P(x)|. \tag{13}$$

The following theorem gives an upper bound on the $\mathcal{Y}$-discrepancy in terms of the generalized discrepancy and $d_1(f_P, H'')$.

Proposition 7 *For any distribution q over $\mathcal{S}_\mathcal{X}$ and any set H'', the following inequality holds:*

$$\mathrm{disc}_\mathcal{Y}(\widehat{P}, \mathsf{q}) \leq \mathrm{DISC}(\widehat{P}, \mathsf{q}) + \mu\, d_1(f_P, H'').$$

Proof Let $h_0 \in H''$, by the triangle inequality, we can write

$$\begin{aligned}
\mathrm{disc}_\mathcal{Y}(\widehat{P}, \mathsf{q}) &= \sup_{h \in H} |\mathcal{L}_\mathsf{q}(h, f_Q) - \mathcal{L}_{\widehat{P}}(h, f_P)| \\
&\leq \sup_{h \in H} |\mathcal{L}_\mathsf{q}(h, f_Q) - \mathcal{L}_{\widehat{P}}(h, h_0)| + \sup_{h \in H} |\mathcal{L}_{\widehat{P}}(h, h_0) - \mathcal{L}_{\widehat{P}}(h, f_P)| \\
&\leq \sup_{h \in H} \max_{h'' \in H''} |\mathcal{L}_\mathsf{q}(h, f_Q) - \mathcal{L}_{\widehat{P}}(h, h'')| + \sup_{h \in H} |\mathcal{L}_{\widehat{P}}(h, h_0) - \mathcal{L}_{\widehat{P}}(h, f_P)|.
\end{aligned}$$

The hypothesis h_0 will later be chosen to minimize the distance of f_P to H''. By the μ-admissibility of the loss, the last term can be bounded as follows:

$$\sup_{h \in H} |\mathcal{L}_{\widehat{P}}(h, h_0) - \mathcal{L}_{\widehat{P}}(h, f_P)| \leq \mu \mathop{\mathrm{E}}_{\widehat{P}} |f_P(x) - h_0(x)|.$$

Using this inequality and minimizing over $h_0 \in H''$ yields:

$$\begin{aligned}
\mathrm{disc}_\mathcal{Y}(\widehat{P}, \mathsf{q}) &\leq \sup_{h \in H} \max_{h'' \in H''} |\mathcal{L}_\mathsf{q}(h, f_Q) - \mathcal{L}_{\widehat{P}}(h, h'')| + \mu\, d_1(f_P, H'') \\
&= \mathrm{DISC}(\widehat{P}, \mathsf{q}) + \mu\, d_1(f_P, H''),
\end{aligned}$$

which completes the proof. ■

We can also bound the $\mathcal{Y}$-discrepancy in terms of the discrepancy measure and the following measure of the difference of the source and target labeling functions:

$$\eta_H(f_P, f_Q) = \min_{h_0 \in H} \Big(\max_{x \in \mathrm{supp}(\widehat{P})} |f_P(x) - h_0(x)| + \max_{x \in \mathrm{supp}(\widehat{Q})} |f_Q(x) - h_0(x)| \Big).$$

Proposition 8 *The following inequality holds for all distributions q over $\mathcal{S}_\mathcal{X}$:*

$$\mathrm{disc}_\mathcal{Y}(\widehat{P}, \mathsf{q}) \leq \mathrm{disc}(\widehat{P}, \mathsf{q}) + \mu\, \eta_H(f_P, f_Q).$$

Proof By the triangle inequality and the μ-admissibility of the loss, the following inequality holds for all $h_0 \in H$:

$$\begin{aligned}
&\mathrm{disc}_\mathcal{Y}(\widehat{P}, \mathsf{q}) \\
&= \sup_{h \in H} |\mathcal{L}_\mathsf{q}(h, f_Q) - \mathcal{L}_{\widehat{P}}(h, f_P)| \\
&\leq \sup_{h \in H} \Big(|\mathcal{L}_{\widehat{P}}(h, h_0) - \mathcal{L}_{\widehat{P}}(h, f_P)| + |\mathcal{L}_\mathsf{q}(h, f_Q) - \mathcal{L}_\mathsf{q}(h, h_0)| \Big) + \sup_{h \in H} |\mathcal{L}_\mathsf{q}(h, h_0) - \mathcal{L}_{\widehat{P}}(h, h_0)| \\
&\leq \mu \Big(\sup_{x \in \mathrm{supp}(\widehat{P})} |h_0(x) - f_P(x)|] + \sup_{x \in \mathrm{supp}(\widehat{Q})} [|f_Q(x) - h_0(x)|] \Big) + \mathrm{disc}(\widehat{P}, \mathsf{q}).
\end{aligned}$$

Minimizing over all $h_0 \in H$ gives $\mathrm{disc}_\mathcal{Y}(\widehat{P}, \mathsf{q}) \leq \mu\, \eta_H(f_P, f_Q) + \mathrm{disc}(\widehat{P}, \mathsf{q})$ and completes the proof. ■

The following learning guarantees are immediate consequences of Propositions 5, 7 and 8.

Corollary 9 *Let $H'' \subset H$ be a convex set and q a distribution over $\mathcal{S}_\mathcal{X}$. Then, for any $\delta > 0$, each of the following inequalities holds with probability at least $1 - \delta$ for all $h \in H$:*

$$\mathcal{L}_P(h, f_P) \leq \mathcal{L}_\mathsf{q}(h, f_Q) + \mathrm{DISC}(\widehat{P}, \mathsf{q}) + \mu\, d_1(f_P, H'') + 2\mathfrak{R}_n(\mathcal{H}_P) + M_P \sqrt{\frac{\log(\frac{1}{\delta})}{2n}}, \tag{14}$$

$$\mathcal{L}_P(h, f_P) \leq \mathcal{L}_\mathsf{q}(h, f_Q) + \mathrm{disc}(\widehat{P}, \mathsf{q}) + \mu\, \eta_H(f_P, f_Q) + 2\mathfrak{R}_n(\mathcal{H}_P) + M_P \sqrt{\frac{\log(\frac{1}{\delta})}{2n}}. \tag{15}$$

In general, the bounds (14) and (15) are not comparable. However, when L is an L_P loss for some $p \geq 1$, we can show the existence of a set H'' for which (14) is a tighter bound than (15). The result is expressed in terms of the *local discrepancy* defined by:

$$\mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) = \sup_{h \in H, h'' \in H''} |\mathcal{L}_{\widehat{P}}(h, h'') - \mathcal{L}_\mathsf{q}(h, h'')|,$$

which is a finer measure than the standard discrepancy for which the supremum is defined over a pair of hypotheses *both* in $H \supseteq H''$.

Theorem 10 *Let L be the L_P loss for some $p \geq 1$. Let $\mathcal{H} := \{B(r)\colon r \geq 0\}$ be a set of all balls $B(r) = \{h'' \in H | \mathcal{L}_\mathsf{q}(h'', f_Q) \leq r^p\}$. Then, for any distribution q over $\mathcal{S}_\mathcal{X}$, there exists $H'' \in \mathcal{H}$ such that the following holds:*

$$\mathrm{DISC}(\widehat{P}, \mathsf{q}) + \mu\, d_1(f_P, H'') \leq \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) + \mu\, \eta_H(f_P, f_Q).$$

Proof Fix a distribution q over $\mathcal{S}_{\mathcal{X}}$. Let h_0^* be an element of $\operatorname{argmin}_{h_0 \in H} \left(\mathcal{L}_{\widehat{P}}(h_0, f_P)^{\frac{1}{p}} + \mathcal{L}_{\mathsf{q}}(h_0, f_Q)^{\frac{1}{p}}\right)$. Choose $H'' \in \mathcal{H}$ as $H'' = \{h'' \in H | \mathcal{L}_{\mathsf{q}}(h'', f_Q) \leq r^p\}$ with $r = \mathcal{L}_{\mathsf{q}}(h_0^*, f_Q)^{\frac{1}{p}}$. Then, by definition, h_0^* is in H''. For the L_p loss, it is not hard to show that for all $h, h'' \in H$, $|\mathcal{L}_{\mathsf{q}}(h, h'') - \mathcal{L}_{\mathsf{q}}(h, f_Q)| \leq \mu[\mathcal{L}_{\mathsf{q}}(h'', f_Q)]^{\frac{1}{p}}$ (see Appendix C). In view of this inequality, we can write:

$$\begin{aligned}
\mathrm{DISC}(\widehat{P}, \mathsf{q}) &= \sup_{h \in H, h'' \in H''} |\mathcal{L}_{\widehat{P}}(h, h'') - \mathcal{L}_{\mathsf{q}}(h, f_Q)| \\
&\leq \sup_{h \in H, h'' \in H''} |\mathcal{L}_{\widehat{P}}(h, h'') - \mathcal{L}_{\mathsf{q}}(h, h'')| + \sup_{h \in H, h'' \in H''} |\mathcal{L}_{\mathsf{q}}(h, h'') - \mathcal{L}_{\mathsf{q}}(h, f_Q)| \\
&\leq \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) + \max_{h'' \in H''} \mu[\mathcal{L}_{\mathsf{q}}(h'', f_Q)]^{\frac{1}{p}} \\
&= \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) + \mu r = \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) + \mu \mathcal{L}_{\mathsf{q}}(h_0^*, f_Q)^{\frac{1}{p}}.
\end{aligned}$$

Using this inequality, Jensen's inequality, and the fact that h_0^* is in H'', we can write

$$\begin{aligned}
&\mu\, d_1(f_P, H'') + \mathrm{DISC}(\widehat{P}, \mathsf{q}) \\
&\leq \mu \min_{h_0 \in H''} \mathop{\mathrm{E}}_{x \in \widehat{P}}[|f_P(x) - h_0(x)|] + \mu \mathcal{L}_{\mathsf{q}}(h_0^*, f_Q)^{\frac{1}{p}} + \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) \\
&\leq \mu \min_{h_0 \in H''} \mathop{\mathrm{E}}_{x \in \widehat{P}}[|f_P(x) - h_0(x)|^p]^{\frac{1}{p}} + \mu \mathcal{L}_{\mathsf{q}}(h_0^*, f_Q)^{\frac{1}{p}} + \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) \\
&\leq \mu\, \mathcal{L}_{\widehat{P}}(h_0^*, f_P)^{\frac{1}{p}} + \mu \mathcal{L}_{\mathsf{q}}(h_0^*, f_Q)^{\frac{1}{p}} + \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}).
\end{aligned}$$

Moreover, by definition of h_0^* the last expression is equal to

$$\begin{aligned}
&\mu \min_{h_0 \in H} \left(\mathcal{L}_{\widehat{P}}(h_0, f_P)^{\frac{1}{p}} + \mathcal{L}_{\mathsf{q}}(h_0, f_Q)^{\frac{1}{p}}\right) + \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) \\
&\quad \leq \mu \min_{h_0 \in H} \Big(\max_{x \in \mathrm{supp}(\widehat{P})} |f_P(x) - h_0(x)| + \max_{x \in \mathrm{supp}(\widehat{Q})} |f_Q(x) - h_0(x)| \Big) + \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) \\
&\quad = \mu\, \eta_H(f_P, f_Q) + \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}).
\end{aligned}$$

which concludes the proof. ■

Theorem 10 shows that the generalized discrepancy can provide a finer measure of the difference between two domains for some choices of H''. Therefore, for a good choice of H'', an algorithm minimizing the right-hand side of (14) would benefit from better theoretical guarantees than the DM algorithm. However, the optimization problem defined by (14) is not jointly convex in q and h. Instead, we propose to first minimize the generalized discrepancy and then use this reweighting function as input to our learning algorithm. Further motivation for this two-stage algorithm is given in the following section.

4.2. Pointwise Guarantees

Similar to the guarantee presented by Cortes and Mohri (2013), we will seek to bound the difference between an *ideal solution* h^* and the solution obtained by our algorithm. We begin by stating the following bound motivating the DM algorithm.

Theorem 11 (Cortes and Mohri, 2013) *Let q be an arbitrary distribution over $\mathcal{S}_{\mathcal{X}}$ and let h^* and h_{q} be the hypotheses minimizing $\lambda\|h\|_K^2 + \mathcal{L}_{\widehat{P}}(h, f_P)$ and $\lambda\|h\|_K^2 + \mathcal{L}_{\mathsf{q}}(h, f_Q)$ respectively. Then, the following inequality holds:*

$$\lambda\|h^* - h_{\mathsf{q}}\|_K^2 \leq \mu\, \eta_H(f_P, f_Q) + \mathrm{disc}(\widehat{P}, \mathsf{q}). \tag{16}$$

Notice that the solution of DM minimizes the right-hand side of (16), that is $\mathrm{disc}(\widehat{P}, \mathsf{q})$. The following theorem provides an analogous bound for our algorithm.

Theorem 12 *Let U be an arbitrary element of $\mathcal{A}(H)$ and let h^* and h_{U} be the hypotheses minimizing $\lambda\|h\|_K^2 + \mathcal{L}_{\widehat{P}}(h, f_P)$ and $\lambda\|h\|_K^2 + \mathcal{L}_{\mathsf{U}_h}(h, f_Q)$ respectively. Then, the following inequality holds for any convex set $H'' \subseteq H$:*

$$\lambda\|h^* - h_{\mathsf{U}}\|_K^2 \leq \mu\, d_1(f_P, H'') + \mathrm{DISC}(\widehat{P}, \mathsf{U}). \tag{17}$$

Proof Fix $\mathsf{U} \in \mathcal{A}(H)$ and let $G_{\widehat{P}}$ denote $h \mapsto \mathcal{L}_{\widehat{P}}(h, f_P)$ and G_{U} the function $h \mapsto \mathcal{L}_{\mathsf{U}_h}(h, f_Q)$. Since $h \mapsto \lambda\|h\|_K^2 + G_{\widehat{P}}(h)$ is convex and differentiable and since h^* is its minimizer, the gradient is zero at h^*, that is $2\lambda h^* = -\nabla G_{\widehat{P}}(h^*)$. Similarly, since $h \mapsto \lambda\|h\|_K^2 + G_{\mathsf{U}}(h)$ is convex, it admits a sub-differential at any $h \in \mathbb{H}$. Since h_{U} is a minimizer, its sub-differential at h_{U} must contain 0. Thus, there exists a sub-gradient $g_0 \in \partial G_{\mathsf{U}}(h_{\mathsf{U}})$ such that $2\lambda h_{\mathsf{U}} = -g_0$, where $\partial G_{\mathsf{U}}(h_{\mathsf{U}})$ denotes the sub-differential of G_{U} at h_{U}. Using these two equalities we can write

$$\begin{aligned}
2\lambda\|h^* - h_{\mathsf{U}}\|_K^2 &= \langle h^* - h_{\mathsf{U}}, g_0 - \nabla G_{\widehat{P}}(h^*)\rangle = \langle g_0, h^* - h_{\mathsf{U}}\rangle - \langle \nabla G_{\widehat{P}}(h^*), h^* - h_{\mathsf{U}}\rangle \\
&\leq G_{\mathsf{U}}(h^*) - G_{\mathsf{U}}(h_{\mathsf{U}}) + G_{\widehat{P}}(h_{\mathsf{U}}) - G_{\widehat{P}}(h^*) \\
&= \mathcal{L}_{\widehat{P}}(h_{\mathsf{U}}, f_P) - \mathcal{L}_{\mathsf{U}_h}(h_{\mathsf{U}}, f_Q) + \mathcal{L}_{\mathsf{U}_h}(h^*, f_Q) - \mathcal{L}_{\widehat{P}}(h^*, f_P) \\
&\leq 2 \sup_{h \in H} |\mathcal{L}_{\widehat{P}}(h, f_P) - \mathcal{L}_{\mathsf{U}_h}(h, f_Q)|,
\end{aligned}$$

where we used for the first inequality the convexity of G_{U} combined with the sub-gradient property of $g_0 \in \partial G_{\mathsf{U}}(h_{\mathsf{U}})$, and the convexity of $G_{\widehat{P}}$. For any $h \in H$, using the μ-admissibility of the loss, we can upper bound the operand of the max operator as follows:

$$\begin{aligned}
|\mathcal{L}_{\widehat{P}}(h, f_P) - \mathcal{L}_{\mathsf{U}_h}(h, f_Q)| &\leq |\mathcal{L}_{\widehat{P}}(h, f_P) - \mathcal{L}_{\widehat{P}}(h, h_0)| + |\mathcal{L}_{\widehat{P}}(h, h_0) - \mathcal{L}_{\mathsf{U}_h}(h, f_Q)| \\
&\leq \mu \mathop{\mathrm{E}}_{x \sim \widehat{P}} |f_P(x) - h_0(x)| + \max_{h'' \in H''} |\mathcal{L}_{\widehat{P}}(h, h'') - \mathcal{L}_{\mathsf{U}_h}(h, f_Q)|,
\end{aligned}$$

where h_0 is an arbitrary element of H''. Since this bound holds for all $h_0 \in H''$, it follows immediately that

$$\lambda\|h^* - h_{\mathsf{U}}\|_K^2 \leq \mu \min_{h_0 \in H''} \mathop{\mathrm{E}}_{\widehat{P}} |f_P(x) - h_0(x)| + \sup_{h \in H} \max_{h'' \in H''} |\mathcal{L}_{\widehat{P}}(h, h'') - \mathcal{L}_{\mathsf{U}_h}(h, f_Q)|,$$

which concludes the proof. ■

Note that our choice of $\mathsf{Q}\colon h \mapsto \mathsf{Q}_h$ minimizes the right-hand side of (17) among all functions $\mathsf{U} \in \mathcal{A}(H)$ since, for any U, we can write

$$\begin{aligned}
\mathrm{DISC}(\widehat{P}, \mathsf{U}) &= \sup_{h \in H} \max_{h'' \in H''} |\mathcal{L}_{\widehat{P}}(h, h'') - \mathcal{L}_{\mathsf{U}_h}(h, f_Q)| \geq \sup_{h \in H} \min_{\mathsf{q} \in \mathcal{F}(\mathcal{S}_{\mathcal{X}})} \max_{h'' \in H''} |\mathcal{L}_{\widehat{P}}(h, h'') - \mathcal{L}_{\mathsf{q}}(h, f_Q)| \\
&= \sup_{h \in H} \max_{h'' \in H''} |\mathcal{L}_{\widehat{P}}(h, h'') - \mathcal{L}_{\mathsf{Q}_h}(h, f_Q)| = \mathrm{DISC}(\widehat{P}, \mathsf{Q}).
\end{aligned}$$

Thus, in view of Theorem 10, for any constant function $\mathsf{U} \in \mathcal{A}(H)$ with $\mathsf{U}_h = \mathsf{q}$ for some fixed distribution q over $\mathcal{S}_{\mathcal{X}}$, the right-hand side of the bound of Theorem 11 is lower bounded by the right-hand side of the bound of Theorem 12, since the local discrepancy is a finer quantity than the discrepancy: $\mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}) \leq \mathrm{disc}(\widehat{P}, \mathsf{q})$. Thus, as expected from the discussion after Theorem 10, our algorithm benefits from a more favorable guarantee than the DM algorithm for some particular choices of H'', especially since, our choice of Q is based on the minimization over all elements in $\mathcal{A}(H)$ and not just the subset of constant functions mapping to a distribution. The following pointwise guarantee follows directly from Theorem 12.

Corollary 13 *Let h^* be a minimizer of $\lambda\|h\|_K^2 + \mathcal{L}_{\widehat{P}}(h, f_P)$ and h_{Q} a minimizer of $\lambda\|h\|_K^2 + \mathcal{L}_{\mathsf{Q}_h}(h, f_Q)$. Then, the following holds for any convex set $H'' \subseteq H$ and for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$:*

$$|L(h_{\mathsf{Q}}(x), y) - L(h^*(x), y)| \leq \mu R \sqrt{\frac{\mu\, d_1(f_P, H'') + \mathrm{DISC}(\widehat{P}, \mathsf{Q})}{\lambda}}, \tag{18}$$

where $R^2 = \sup_{x \in \mathcal{X}} K(x, x)$.

Proof By the μ-admissibility of the loss, the reproducing property of $\mathbb{H}$, and the Cauchy-Schwarz inequality, the following holds for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$:

$$\begin{aligned}
|L(h_{\mathsf{Q}}(x), y) - L(h^*(x), y)| &\leq \mu |h_{\mathsf{Q}}(x) - h^*(x)| \\
&= \mu |\langle h_{\mathsf{Q}} - h^*, K(x, \cdot)\rangle_K| \\
&\leq \mu \|h_{\mathsf{Q}} - h^*\|_K \sqrt{K(x, x)} \leq R \|h_{\mathsf{Q}} - h^*\|_K.
\end{aligned}$$

Upper bounding $\|h_{\mathsf{Q}} - h^*\|_K$ using Theorem 12 and using the fact that $\mathsf{Q} \colon h \to \mathsf{Q}_h$ is a minimizer of the bound over all choices of $\mathsf{U} \in \mathcal{A}(H)$ yields the desired result. ■

The pointwise loss guarantee just presented can be directly used to bound the difference of the expected loss of h^* and h_{Q} in terms of the same upper bounds, e.g.,

$$\mathcal{L}_P(h_{\mathsf{Q}}, f_P) \leq \mathcal{L}_P(h^*, f_P) + \mu R \sqrt{\frac{\mu\, d_1(f_P, H'') + \mathrm{DISC}(\widehat{P}, \mathsf{Q})}{\lambda}}. \tag{19}$$

Similarly, Theorem 10 directly implies the following Corollary.

Corollary 14 *Let h^* be a minimizer of $\lambda\|h\|_K^2 + \mathcal{L}_{\widehat{P}}(h, f_P)$ and h_{Q} a minimizer of $\lambda\|h\|_K^2 + \mathcal{L}_{\mathsf{Q}_h}(h, f_Q)$. Let $\sup_{x \in \mathcal{X}} K(x, x) = R^2$. Then, there exists a choice of $H'' \in \mathcal{H}$ for which the following inequality holds uniformly over over $(x, y) \in \mathcal{X} \times \mathcal{Y}$:*

$$|L(h_{\mathsf{Q}}(x), y) - L(h^*(x), y)| \leq \mu R \sqrt{\frac{\mu \eta_H(f_P, f_Q) + \mathrm{disc}_{H''}(\widehat{P}, \mathsf{q}_{min})}{\lambda}},$$

where q_{min} is the solution of the DM algorithm.

The choice of the set H'' defining our algorithm is strongly motivated by the theoretical results of this section. In view of Theorem 10, we restrict our choice of H'' to the family $\mathcal{H}$, parametrized only by the radius r. Since the generalized discrepancy DISC is a function of the set H'' which in turn depends only on r, the radius r is chosen to minimize (19). This can be done by using as a validation set a small amount of labeled data from the target domain which is typically available in practice. In particular, as the size of the unlabeled sample $\mathcal{T}'$ increases, our estimate of the optimal radius r becomes more accurate. We provide a detailed description of our algorithm's implementation in Section 5.

4.3. Comparison against Other Learning Bounds

We now compare the learning bounds just derived for our algorithm with those of some common reweighting techniques. In particular, we compare our bounds with those of Cortes et al. (2008) for the KMM algorithm. A similar comparison however can be derived for other algorithms based on importance weighting such as KLIEP or uLSIF.

Assume P and Q admit densities p and q respectively. For every $x \in \mathcal{X}$ we denote by $\beta(x) = \frac{p(x)}{q(x)}$ the importance ratio and by $\overline{\beta} = \beta|_{\mathcal{S}_{\mathcal{X}}}$ its restriction to $\mathcal{S}_{\mathcal{X}}$. We also let $\widehat{\beta}$ be the solution to the optimization problem solved by the KMM algorithm. Let $h_{\overline{\beta}}$ denote the solution to

$$\min_{h \in \mathbb{H}} \lambda\|h\|^2 + \mathcal{L}_{\overline{\beta}}(h, f_Q), \tag{20}$$

and $h_{\widehat{\beta}}$ be the solution to

$$\min_{h \in \mathbb{H}} \lambda\|h\|^2 + \mathcal{L}_{\widehat{\beta}}(h, f_Q). \tag{21}$$

The following proposition due to Cortes et al. (2008) relates the error of these hypotheses. The proposition requires the kernel K to be a strictly positive definite universal kernel, with Gram matrix $\mathbf{K}$ given by $\mathbf{K}_{ij} = K(x_i, x_j)$.

Proposition 15 *Assume $L(h(x), y) \leq 1$ for all $(x, y) \in \mathcal{X} \times \mathcal{Y}, h \in H$. For any $\delta > 0$, with probability at least $1 - \delta$ we have:*

$$\begin{aligned}
|\mathcal{L}_P(h_{\overline{\beta}}, f_P) - \mathcal{L}_P(h_{\widehat{\beta}}, f_P)| \leq \frac{\mu^2 R^2 \lambda_{\max}^{\frac{1}{2}}(\mathbf{K})}{\lambda} \Big(& \frac{\epsilon B'}{\sqrt{m}} \\
& + \frac{\kappa^{1/2}}{\lambda_{\min}^{1/2}(\mathbf{K})} \sqrt{\frac{B'^2}{m} + \frac{1}{n}} \Big(1 + \sqrt{2 \log \frac{2}{\delta}}\Big) \Big),
\end{aligned} \tag{22}$$

where ϵ and B' are the hyperparameters defining the KMM algorithm and $\lambda_{\max}(\mathbf{K}), \lambda_{\min}(\mathbf{K})$ denote the largest and smallest eigenvalues of $\mathbf{K}$ respectively.

This bound and the one obtained in (19) are of course not comparable since the dependence on μ, R and λ is different. In some cases this dependency can be more favorable in (22) whereas for other values of these parameters (19) provides a better bound. Moreover, (22) depends on the condition number of $\mathbf{K}$ which can become really large in practice. However, the most important difference between these bounds is that (19) is given in terms of the ideal hypothesis h^* while (22) is given in terms of $h_{\overline{\beta}}$, which, in view of the results of Cortes et al. (2010) is not guaranteed to have a good performance on the target distribution. Therefore (22) does not, in general, provide an informative bound.

4.4. Scenario of Additional Labeled Data

Here, we consider a rather common scenario in practice where, in addition to the labeled sample $\mathcal{S}$ drawn from the source domain and the unlabeled sample $\mathcal{T}$ from the target domain, the learner receives a small amount of labeled data from the target domain $\mathcal{T}' = ((x'_1, y'_1), \ldots, (x'_s, y'_s)) \in (\mathcal{X} \times \mathcal{Y})^s$. This sample is typically too small to be used solely to train an algorithm and achieve a good performance. However, it can be useful in at least two ways that we discuss here.

One important benefit of $\mathcal{T}'$ is to serve as a validation set to determine the parameter r that defines the convex set H'' used by our algorithm. The sample $\mathcal{T}'$ can also be used to enhance the discrepancy minimization algorithm as we now show. Let $\widehat{P}'$ denote the empirical distribution associated with $\mathcal{T}'$. To take advantage of $\mathcal{T}'$, the DM algorithm can be trained on the sample of size $(m+s)$ obtained by combining $\mathcal{S}$ and $\mathcal{T}'$, which corresponds to the new empirical distribution $\widehat{Q}' = \frac{m}{m+s}\widehat{Q} + \frac{s}{m+s}\widehat{P}'$. Note that for a fixed value m and large values of s, $\widehat{Q}'$ essentially ignores the points from the source distribution Q, which corresponds to the standard supervised learning scenario in the absence of adaptation. Let $\mathsf{q}'_{\min}$ denote the discrepancy minimization solution when using $\widehat{Q}'$. Since $\mathrm{supp}(\widehat{Q}') \supseteq \mathrm{supp}(\widehat{Q})$, the discrepancy using $\mathsf{q}'_{\min}$ is a lower bound on the discrepancy using $\mathsf{q}_{\min}$:

$$\mathrm{disc}(\mathsf{q}'_{\min}, \widehat{P}) = \min_{\mathrm{supp}(\mathsf{q}) \subseteq \mathrm{supp}(\widehat{Q}')} \mathrm{disc}(\widehat{P}, \mathsf{q}) \leq \min_{\mathrm{supp}(\mathsf{q}) \subseteq \mathrm{supp}(\widehat{Q})} \mathrm{disc}(\widehat{P}, \mathsf{q}) = \mathrm{disc}(\mathsf{q}_{\min}, \widehat{P}).$$

5. Optimization Solution

As shown in Section 3.2, the function G defining our algorithm is convex and the problem of minimizing the expression (7) is a convex optimization problem. Nevertheless, the problem is not straightforward to solve, in particular because evaluating a term like $\max_{h'' \in H''} \mathcal{L}_{\widehat{P}}(h, h'')$ that it contains requires solving a non-convex optimization problem. Here, we present an exact solution in the case of the L_2 loss by solving a semi-definite programming (SDP) problem.

5.1. SDP Formulation

As discussed in Section 4, the choice of H'' is a key component of our algorithm. In view of Corollary 14, we will consider the set $H'' = \{h'' \,|\, \mathcal{L}_{\mathsf{q}_{\min}}(h'', f_Q) \leq r^2\}$. Equivalently, as a result of the reproducing property of $\mathbb{H}$ and the representer theorem, H'' may be defined as $\{\mathbf{a} \in \mathbb{R}^m | \sum_{j=1}^m \mathsf{q}_{\min}(x_j)(\sum_{i=1}^m a_i \mathsf{q}_{\min}(x_i)^{1/2} K(x_i, x_j) - y_j)^2 \leq r^2\}$. Also by the representer theorem, the solution to (7) will be of the form $h = n^{-1/2} \sum_{i=1}^n b_i K(x'_i, \cdot)$. Therefore, given *normalized* kernel matrices $\mathbf{K}_t$, $\mathbf{K}_s$, $\mathbf{K}_{st}$ defined respectively as $\mathbf{K}_t^{ij} = n^{-1} K(x'_i, x'_j)$, $\mathbf{K}_s^{ij} = \mathsf{q}_{\min}(x_i)^{1/2} \mathsf{q}_{\min}(x_j)^{1/2} K(x_i, x_j)$ and $\mathbf{K}_{st}^{ij} = n^{-1/2} \mathsf{q}_{\min}(x_j)^{1/2} K(x'_i, x_j)$, problem (7) is equivalent to

$$\min_{\mathbf{b} \in \mathbb{R}^n} \lambda \mathbf{b}^\top \mathbf{K}_t \mathbf{b} + \frac{1}{2}\left(\max_{\substack{\mathbf{a} \in \mathbb{R}^m \\ \|\mathbf{K}_s \mathbf{a} - \mathbf{y}\|^2 \leq r^2}} \|\mathbf{K}_{st}\mathbf{a} - \mathbf{K}_t \mathbf{b}\|^2 + \min_{\substack{\mathbf{a} \in \mathbb{R}^m \\ \|\mathbf{K}_s \mathbf{a} - \mathbf{y}\|^2 \leq r^2}} \|\mathbf{K}_{st}\mathbf{a} - \mathbf{K}_t \mathbf{b}\|^2 \right), \quad (23)$$

where $\mathbf{y} = (\mathsf{q}_{\min}(x_1)^{1/2} y_1, \ldots, \mathsf{q}_{\min}(x_m)^{1/2} y_m)$ is the vector of normalized labels.

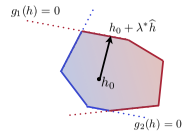


Figure 1: Illustration of the sampling process on the set H''.

Lemma 16 *The Lagrangian dual of the problem*

$$\max_{\substack{\mathbf{a} \in \mathbb{R}^m \\ \|\mathbf{K}_s \mathbf{a} - \mathbf{y}\|^2 \leq r^2}} \frac{1}{2}\|\mathbf{K}_{st}\mathbf{a}\|^2 - \mathbf{b}^\top \mathbf{K}_t \mathbf{K}_{st} \mathbf{a},$$

is given by

$$\min_{\eta \geq 0, \gamma} \gamma$$
$$s.t. \begin{pmatrix} -\frac{1}{2}\mathbf{K}_{st}^\top \mathbf{K}_{st} + \eta \mathbf{K}_s^2 & \frac{1}{2}\mathbf{K}_{st}^\top \mathbf{K}_t \mathbf{b} - \eta \mathbf{K}_s \mathbf{y} \\ \frac{1}{2}\mathbf{b}^\top \mathbf{K}_t \mathbf{K}_{st} - \eta \mathbf{y}^\top \mathbf{K}_s & \eta(\|\mathbf{y}\|^2 - r^2) + \gamma \end{pmatrix} \succeq 0.$$

Furthermore, the duality gap for these problems is zero.

The proof of the lemma is given in Appendix A. The lemma helps us derive the following equivalent SDP formulation for our original optimization problem. Its solution can be found in polynomial time using standard convex optimization solvers.

Proposition 17 *The optimization problem* (23) *is equivalent to the following SDP:*

$$\max_{\alpha, \beta, \nu, \mathbf{Z}, \mathbf{z}} \frac{1}{2}\mathrm{Tr}(\mathbf{K}_{st}^\top \mathbf{K}_{st} \mathbf{Z}) - \beta - \alpha$$
$$s.\ t \begin{pmatrix} \nu \mathbf{K}_s^2 + \frac{1}{2}\mathbf{K}_{st}^\top \mathbf{K}_{st} - \frac{1}{4}\widetilde{\mathbf{K}} & \nu \mathbf{K}_s \mathbf{y} + \frac{1}{4}\widetilde{\mathbf{K}}\mathbf{z} \\ \nu \mathbf{y}^\top \mathbf{K}_s + \frac{1}{4}\mathbf{z}^\top \widetilde{\mathbf{K}} & \alpha + \nu(\|\mathbf{y}\|^2 - r^2) \end{pmatrix} \succeq 0 \quad \begin{pmatrix} \lambda \mathbf{K}_t + \mathbf{K}_t^2 & \frac{1}{2}\mathbf{K}_t \mathbf{K}_{st} \mathbf{z} \\ \frac{1}{2}\mathbf{z}^\top \mathbf{K}_{st}^\top \mathbf{K}_t & \beta \end{pmatrix} \succeq 0$$
$$\begin{pmatrix} \mathbf{Z} & \mathbf{z} \\ \mathbf{z}^\top & 1 \end{pmatrix} \succeq 0 \wedge \nu \geq 0 \wedge \mathrm{Tr}(\mathbf{K}_s^2 \mathbf{Z}) - 2\mathbf{y}^\top \mathbf{K}_s \mathbf{z} + \|\mathbf{y}\|^2 \leq r^2,$$

where $\widetilde{\mathbf{K}} = \mathbf{K}_{st}^\top \mathbf{K}_t (\lambda \mathbf{K}_t + \mathbf{K}_t^2)^\dagger \mathbf{K}_t \mathbf{K}_{st}$, *and* $\mathbf{A}^\dagger$ *denotes the pseudo-inverse of matrix* A.

In the following section we derive a more efficient approximate solution to the optimization problem using sampling, which helps reducing the problem to a simple QP.

5.2. QP Formulation

The SDP formulation described in the previous section is applicable for a specific choice of H''. In this section, we present an analysis that holds for an arbitrary compact, convex set H''. First, notice that the problem of minimizing G (expression (7)) is related to the

minimum enclosing ball (MEB) problem. For a set $D \subseteq \mathbb{R}^d$, the MEB problem is defined as follows:

$$\min_{\mathbf{u}\in\mathbb{R}^d} \max_{\mathbf{v}\in D} \|\mathbf{u}-\mathbf{v}\|^2.$$

Omitting the regularization and the min term from (7) leads to a problem similar to the MEB. Thus, we could benefit from the extensive literature and algorithmic study available for this problem (Kumar et al., 2003; Schőnherr, 2002; Yildirim, 2008). However, to the best of our knowledge, there is currently no solution available to this problem in the case of an infinite set D, as in the case of our problem. Instead, we present a solution for solving an approximation of (7) based on sampling.

Let $\{h_1, \ldots, h_k\}$ be a set of hypotheses on the boundary of H'', $\partial H''$ and let $\mathcal{C} = \mathcal{C}(h_1, \ldots, h_k)$ denote their convex hull. The following is the sampling-based approximation of (7) that we consider:

$$\min_{h\in\mathbb{H}} \lambda\|h\|_K^2 + \frac{1}{2}\max_{i=1,\ldots,k} \mathcal{L}_{\widehat{P}}(h, h_i) + \frac{1}{2}\min_{h'\in\mathcal{C}} \mathcal{L}_{\widehat{P}}(h, h'). \tag{24}$$

Proposition 18 *Let $\mathbf{Y} = (Y_{ij}) \in \mathbb{R}^{n\times k}$ be the matrix defined by $Y_{ij} = n^{-1/2}h_j(x_i')$ and $\mathbf{y}' = (y_1', \ldots, y_k')^\top \in \mathbb{R}^k$ the vector defined by $y_i' = n^{-1}\sum_{j=1}^n h_i(x_j')^2$. Then, the dual problem of (24) is given by*

$$\max_{\boldsymbol{\alpha},\boldsymbol{\gamma},\beta} -\Big(\mathbf{Y}\boldsymbol{\alpha}+\frac{\boldsymbol{\gamma}}{2}\Big)^\top \mathbf{K}_t\Big(\lambda\mathbf{I}+\frac{1}{2}\mathbf{K}_t\Big)^{-1}\Big(\mathbf{Y}\boldsymbol{\alpha}+\frac{\boldsymbol{\gamma}}{2}\Big) - \frac{1}{2}\boldsymbol{\gamma}^\top\mathbf{K}_t\mathbf{K}_t^\dagger\boldsymbol{\gamma} + \boldsymbol{\alpha}^\top\mathbf{y}' - \beta \tag{25}$$
$$s.t.\ \mathbf{1}^\top\boldsymbol{\alpha} = \frac{1}{2}, \qquad \mathbf{1}\beta \geq -\mathbf{Y}^\top\boldsymbol{\gamma}, \qquad \boldsymbol{\alpha} \geq 0,$$

where $\mathbf{1}$ is the vector in $\mathbb{R}^k$ with all components equal to 1. Furthermore, the solution h of (24) can be recovered from a solution $(\boldsymbol{\alpha}, \boldsymbol{\gamma}, \beta)$ of (25) by $\forall x, h(x) = \sum_{i=1}^n a_i K(x_i, x)$, where $\boldsymbol{a} = \big(\lambda\mathbf{I} + \frac{1}{2}\mathbf{K}_t\big)^{-1}\big(\mathbf{Y}\boldsymbol{\alpha} + \frac{1}{2}\boldsymbol{\gamma}\big)$.

The proof of the proposition is given in Appendix B. The result shows that, given a finite sample $h_1, \ldots, h_k$ on the boundary of H'', (24) is in fact equivalent to a standard QP. Hence, a solution can be found efficiently with one of the many off-the-shelf algorithms for quadratic programming.

We now describe the process of sampling from the boundary of the set H'', which is a necessary step for defining problem (24). We consider compact sets of the form $H'' := \{h'' \in \mathbb{H} \mid g_i(h'') \leq 0\}$, where the functions g_i are continuous and convex. For instance, we could consider the set H'' defined in the previous section. More generally, we can consider a family of sets $H_p'' = \{h'' \in H| \mid \sum_{i=1}^m \mathsf{q}_{\min}(x_i)|h(x_i) - y_i|^p \leq r^p\}$.

Assume that there exists h_0 satisfying $g_i(h_0) < 0$. Our sampling process is illustrated by Figure 1 and works as follows: pick a random direction $\widehat{h}$ and define λ_i to be the minimal solution to the system

$$(\lambda \geq 0) \wedge (g_i(h_0 + \lambda\widehat{h}) = 0).$$

Set $\lambda_i = \infty$ if no solution is found and define $\lambda^* = \min_i \lambda_i$. By the convexity and compactness of H'' we can guarantee that $\lambda^* < \infty$. The hypothesis $h = h_0 + \lambda^*\widehat{h}$ satisfies $h \in H''$ and $g_j(h) = 0$ for j such that $\lambda_j = \lambda^*$. The latter is straightforward. To verify the former,

assume that $g_i(h_0 + \lambda^*\widehat{h}) > 0$ for some i. The continuity of g_i would imply the existence of λ_i' with $0 < \lambda_i' < \lambda^* \leq \lambda_i$ such that $g_i(h_0 + \lambda_i'\widehat{h}) = 0$. This would contradict the choice of λ_i, thus, the inequality $g_i(h_0 + \lambda^*\widehat{h}) \leq 0$ must hold for all i.

Since a point h_0 with $g_i(h_0) < 0$ can be obtained by solving a convex program and solving the equations defining λ_i is, in general, simple, the process described provides an efficient way of sampling points from the convex set H''.

5.3. Implementation for the L_2 Loss

We now describe how to fully implement our sampling-based algorithm for the case where L is equal to the L_2 loss. In view of the results of Section 4, we let $H'' = \{h''|\|h''\|_K \leq \Lambda \ \wedge \ \mathcal{L}_{\mathsf{q}}(h'', f_Q) \leq r^2\}$. We first describe the steps needed to find a point $h_0 \in H''$. Let h_Λ be such that $\|h_\Lambda\|_K = \Lambda$ and $\lambda_r \in \mathbb{R}_+$ be such that the solution h_r to the optimization problem

$$\min_{h\in\mathbb{H}} \lambda_r\|h\|^2 + \mathcal{L}_{\mathsf{q}}(h, f_Q),$$

satisfies $\mathcal{L}_{\mathsf{q}}(h_r, f_Q) = r^2$. It is easy to verify that the existence of λ_r is guaranteed for $\min_{h\in H}\mathcal{L}_{\mathsf{q}}(h, f_Q) \leq r^2 \leq \sum_{i=1}^m \mathsf{q}(x_i)y_i^2$. It is clear that the point $h_0 = \frac{1}{2}(h_r + h_\Lambda)$ is in the interior of H''. Of course, finding λ_r with the desired properties is not straightforward. However, since r is chosen via validation, we do not need to find λ_r as a function of r. Instead, we can simply select λ_r through validation too.

In order to complete the sampling process, we must have an efficient way of selecting a random direction $\widehat{h}$. If $H \subset \mathbb{R}^d$ is a set of linear hypotheses, a direction $\widehat{h}$ can be sampled uniformly by letting $\widehat{h} = \frac{\xi}{\|\xi\|}$, where ξ is a standard Gaussian random variable in $\mathbb{R}^d$. If H is a subset of a RKHS, by the representer theorem, we only need to consider hypotheses of the form $h = \sum_{i=1}^m \alpha_i K(x_i, \cdot)$. Therefore, we can sample a direction $\widehat{h} = \sum_{i=1}^m \alpha_i' K(x_i, \cdot)$, where the vector $\boldsymbol{\alpha'} = (\alpha_1', \ldots, \alpha_m')$ is drawn uniformly from the unit sphere in $\mathbb{R}^m$. A full implementation of our algorithm thus consists of the following steps:

- find the distribution $\mathsf{q}_{\min} = \operatorname{argmin}_{\mathsf{q}\in\mathcal{Q}} \operatorname{disc}(\mathsf{q}, \widehat{P})$. This can be done by using the smooth approximation algorithm of Cortes and Mohri (2013);
- sample points from the set H'' using the sampling process described above;
- solve the QP introduced in Section 5.2.

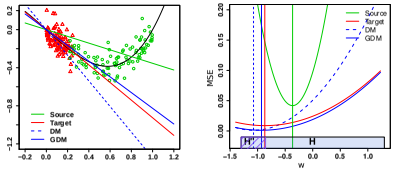
Notice that our algorithm only requires solving a simple QP and therefore its complexity is the same as other adaptation algorithms such as KMM, KLIEP and DM.

6. Experiments

Here, we report the results of extensive comparisons between GDM and several other adaptation algorithms which demonstrate the benefits of our algorithm. We use the implementation described in the previous section. The source code for our algorithm as well as all other baselines described in this section can be found at `http://cims.nyu.edu/~munoz`.

6.1. Synthetic Data Set

To compare the performances of the GDM and DM algorithms, we considered the following synthetic one-dimensional task, which is similar to the one considered by Huang et al.



(a) (b)

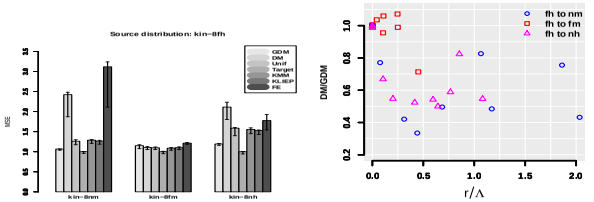
Figure 2: (a) Hypotheses obtained by training on source (green circles), target (red triangles) and using DM (dashed blue) and GDM algorithms (solid blue). (b) Objective functions for source and target distribution as well as GDM and DM algorithms. Sets H and surrogate hypothesis set $H'' \subseteq H$ are shown at the bottom. The vertical lines represent the minimizing hypothesis for each loss.

(2006): the source domain examples were sampled from the uniform distribution over the interval $[.2, 1]$ and target ones sampled uniformly over $[0, .25]$. The labels were given by the map $x \mapsto -x + x^3 + \xi$, where ξ is a Gaussian random variable with mean 0 and standard deviation 0.1. Our hypothesis set was defined by the family of linear functions without an offset. Figure 2(a) shows the regression hypotheses obtained by training the DM and GDM algorithm as well as those obtained by training on the source and target distributions. The ideal hypothesis is shown in red. Notice how the GDM solution gives a closer approximation than DM to the ideal solution. In order to better understand the difference between the solutions of these algorithms, Figure 2(b) depicts the objective function minimized by each algorithm as a function of the slope w of the linear function, the only variable of the hypothesis. The vertical lines show the value of the minimizing hypothesis for each loss. Keeping in mind that the regularization parameter λ used in ridge regression corresponds to a Lagrange multiplier for the constraint $w^2 \leq \Lambda^2$ for some Λ (Cortes and Mohri, 2013) [Lemma 1], the hypothesis set $H = \{w | w^2 \leq \Lambda^2\}$ is depicted at the bottom of this plot. The shaded region represents the set $H'' = H \cap \{h'' | \mathcal{L}_{\mathbf{q}_{\min}}(h'') \leq r\}$. It is clear from this plot that DM helps approximate the target loss function. Nevertheless, only GDM seems to uniformly approach it. This should come as no surprise since our algorithm was precisely designed to achieve that.

6.2. Adaptation Data Sets

We now present the results of evaluating our algorithm against several other adaptation algorithms. GDM is compared against DM and training on the uniform distribution. The following baselines were also used:

1. The KMM algorithm (Huang et al., 2006), which reweights examples from the source distribution in order to match the mean of the source and target data in a feature space induced by a universal kernel. The hyper-parameters of this algorithm were set to the recommended values of $B = 1000$ and $\epsilon = \frac{\sqrt{m}}{\sqrt{m}-1}$.
2. KLIEP (Sugiyama et al., 2007). This algorithm estimates the importance ratio of the source and target distribution by modeling this ratio as a mixture of basis functions and learning the mixture coefficients from the data. Gaussian kernels were used as basis functions for this algorithm and KMM. The bandwidth for the kernel was selected from the set $\{\sigma d \colon \sigma = 2^{-5}, \ldots, 2^5\}$ via validation on the *test* set, where d is the mean distance between points sampled from the source domain.
3. FE (Daumé III, 2007). This algorithm maps source and target data into a common high-dimensional feature space where the difference of the distributions is expected to reduce.



(a) (b)

Figure 3: (a)MSE performance for different adaptation algorithms when adapting from `kin-8fh` to the three other `kin-8xy` domains. (b) Relative error of DM over GDM as a function of the ratio $\frac{r}{\Lambda}$.

Since the two-stage algorithm of Bickel et al. (2007) was already shown to perform similarly to KMM and KLIEP (Cortes and Mohri, 2013), for the sake of readability of our results, we omitted the results of comparison with this algorithm. Finally, we compare our algorithm with the ideal hypothesis h^* returned by training on the target sample $\mathcal{T}$, which we denote by `Tar`. Notice that in practice, this is impossible as $\mathcal{T}$ is unlabeled so we use this result only to show the best attainable performance.

We selected the set of linear functions as our hypothesis set. The learning algorithm used for all tasks was ridge regression and the performance evaluated by the mean squared error. We follow the setup of Cortes and Mohri (2011) and for all adaptation algorithms we selected the parameter λ via 10-fold cross validation over the training data by using a grid search over the set of values $\lambda \in \{2^{-10}, \ldots, 2^{10}\}$. The results of training on the target distribution are presented for a parameter λ tuned via 10-fold cross validation over the target data. We used the QP implementation of our algorithm with the sampling set H'' and the sampling mechanism defined at the end of Section 5.2, where the parameter λ_r was chosen from the same set as λ via validation on a small amount of data from the target distribution. Whereas there are other methods such as transfer cross validation (Zhong et al., 2010) to select the parameters for our algorithm, these methods require the use of importance weighting which as shown in (Cortes et al., 2010) is not theoretically justified.

In order to achieve a fair comparison, all other algorithms were allowed to use the small amount of labeled data too. Since, with the exception of FE, all other baselines do not propose a way of dealing with labeled data from the target distribution, we simply added this data to the training set and ran the algorithms on the extended source data as discussed in Section 4.4.

Task: Sentiment								
S	T	GDM	DM	Unif	Tar	KMM	KLIEP	F
B	K	**0.763±(0.222)**	1.056±(0.289)	1.00	0.517±(0.152)	3.328±(0.845)	3.494±(1.144)	0.942±(0.093)
	E	**0.574±(0.211)**	1.018±(0.206)	1.00	0.367±(0.124)	3.018±(0.319)	3.022±(0.318)	0.857±(0.135)
	D	0.936±(0.256)	1.215±(0.255)	1.00	0.623±(0.152)	2.842±(0.492)	2.764±(0.446)	**0.936±(0.110)**
K	B	**0.854±(0.119)**	1.258±(0.117)	1.00	0.665±(0.085)	2.784±(0.244)	2.642±(0.218)	1.047±(0.047)
	E	0.975±(0.131)	1.460±(0.633)	1.00	0.653±(0.201)	2.408±(0.582)	2.157±(0.255)	**0.969±(0.131)**
	D	**0.884±(0.101)**	1.174±(0.140)	1.00	0.665±(0.071)	2.771±(0.157)	2.620±(0.210)	1.111±(0.059)
E	B	**0.723±(0.138)**	1.016±(0.187)	1.00	0.551±(0.109)	3.433±(0.694)	3.290±(0.583)	1.035±(0.059)
	K	1.030±(0.312)	1.277±(0.283)	1.00	0.636±(0.176)	2.173±(0.249)	2.223±(0.293)	**0.955±(0.199)**
	D	**0.731±(0.171)**	1.005±(0.166)	1.00	0.518±(0.117)	3.363±(0.402)	3.231±(0.483)	0.974±(0.102)
D	B	0.992±(0.191)	1.026±(0.090)	1.00	0.740±(0.138)	2.571±(0.616)	2.475±(0.400)	**0.986±(0.041)**
	K	**0.870±(0.212)**	1.062±(0.318)	1.00	0.557±(0.137)	2.755±(0.375)	2.741±(0.347)	0.940±(0.087)
	E	**0.674±(0.135)**	0.994±(0.171)	1.00	0.478±(0.098)	2.939±(0.501)	2.878±(0.418)	0.907±(0.081)

Table 2: Adaptation from `books` (B), `kitchen` (K), `electronics` (E) and `dvd` (D) to all other domains. Normalized results: MSE of training on the unweighted source data is equal to 1. Results in bold represent the algorithm with the lowest MSE.

The first task we considered is given by the 4 `kin-8xy` Delve data sets (Rasmussen et al., 1996). These data sets are variations of the same model: a realistic simulation of the forward dynamics of an 8 link all-revolute robot arm. The task in all data sets consists of predicting the distance of the end-effector from a target. The data sets differ by the degree of non-linearity (fairly linear, `x=f`, or non-linear, `x=n`) and the amount of noise in the output (moderate, `y=m`, or high, `y=h`). The data set defines 4 different domains, that is 12 pairs of different distributions and labeling functions. A sample of 200 points from each domain was used for training and 10 labeled points from the target distribution were used to select H''. The experiment was carried out 10 times and the results of testing on a sample of 400 points from the target domain are reported in Figure 3(a). The bars represent the median performance of each algorithm. The error bars are the low and high 25% quartiles respectively. All results were normalized in such a way that the median performance of training on the source is equal to 1. Notice that the performance of all algorithms is comparable when adapting to `kin8-fm` since both labeling functions are fairly linear, yet only GDM is able to reasonably adapt to the two data sets with different labeling functions. In order to better understand the advantages of GDM over DM we plot the relative error of DM against GDM as a function of the ratio r/Λ in Figure 3(b), where r is the radius defining H'' and is selected through cross validation. Notice that when the ratio r/Λ is small then both algorithms behave similarly which is most of the times for the adaptation task `fh` to `fm`. On the other hand, a better performance of GDM can be obtained when the ratio is larger. This is due to the fact that r/Λ measures the effective size of the set H''. A small ratio means that the size of H'' is small and therefore the hypothesis returned by GDM will be close to that of DM where as if H'' is large then GDM has the possibility of finding a better hypothesis.

For our next experiment we considered the cross-domain sentiment analysis data set of Blitzer et al. (2007). This data set consists of consumer reviews from 4 different domains: `books, kitchen, electronics` and `dvds`. We used the top 1000 uni-grams and bi-grams

Task: Images								
S	T	GDM	DM	Unif	Tar	KMM	KLIEP	F
C	I	**0.927±(0.051)**	1.005±(0.010)	1.00	0.879±(0.048)	2.752±(3.820)	0.936±(0.016)	0.959±(0.035)
	S	0.938±(0.064)	0.993±(0.018)	1.00	0.840±(0.057)	**0.827±(0.017)**	0.835±(0.020)	0.947±(0.025)
	B	**0.909±(0.040)**	1.003±(0.013)	1.00	0.886±(0.052)	0.945±(0.022)	0.942±(0.017)	0.947±(0.019)
I	C	1.011±(0.015)	**0.951±(0.011)**	1.00	0.802±(0.040)	0.989±(0.036)	1.009±(0.042)	0.971±(0.024)
	S	1.006±(0.030)	0.992±(0.016)	1.00	0.871±(0.030)	**0.930±(0.018)**	0.936±(0.016)	0.973±(0.017)
	B	**0.987±(0.022)**	1.009±(0.010)	1.00	0.986±(0.028)	1.011±(0.028)	1.011±(0.028)	0.994±(0.018)
S	C	1.022±(0.037)	0.982±(0.035)	1.00	0.759±(0.033)	1.172±(0.043)	1.201±(0.038)	**0.938±(0.036)**
	I	**0.924±(0.049)**	0.998±(0.030)	1.00	0.831±(0.047)	3.868±(4.231)	1.227±(0.039)	0.947±(0.028)
	B	**0.898±(0.072)**	1.003±(0.044)	1.00	0.821±(0.053)	1.240±(0.039)	1.248±(0.041)	0.945±(0.021)
B	C	1.010±(0.014)	**0.956±(0.017)**	1.00	0.777±(0.031)	1.028±(0.033)	1.032±(0.031)	0.980±(0.019)
	I	1.012±(0.010)	1.004±(0.007)	1.00	0.966±(0.009)	2.785±(3.803)	**0.981±(0.018)**	1.000±(0.004)
	S	1.009±(0.018)	0.988±(0.010)	1.00	0.850±(0.035)	**0.930±(0.022)**	0.934±(0.024)	0.983±(0.013)

Table 3: Adaptation from `caltech256` (C), `imagenet` (I), `sun` (S) and `bing` (B).

as the features for this task. For each pair of adaptation tasks we sampled 700 points from the source distribution and 700 unlabeled points from the target. Only 50 labeled points from the target distribution were used to tune the parameter r of our algorithm. The final evaluation is done on a test set of 1000 points. The mean results and standard deviations of this task are shown in Table 2 where the MSE values have been normalized in such a way that the performance of training on the source without reweighting is always 1.

Finally, we considered a novel domain adaptation task (Tommasi et al., 2014) of paramount importance in the computer vision community. The domains correspond to 4 well known collections of images: `bing, caltech256, sun` and `imagenet`. These data sets have been standardized so that they all share the same feature representation and labeling function (Tommasi et al., 2014). We sampled 800 labeled points from the source distribution and 800 unlabeled points from the target distribution as well as 50 labeled target points to be used for validation of r. The results of testing on 1000 points from the target domain are presented in Table 3 where, again, the results were normalized in such a way that the performance of training on the source data is always 1.

After analyzing the results of this section we notice that the GDM algorithm consistently outperforms DM and achieves similar or better performance than all other common adaptation algorithms. It is worth noticing that in some cases, other algorithms perform even worse than training on the unweighted sample. This deficiency of the KLIEP algorithm had already been pointed out by Sugiyama et al. (2007) but here we observe that this problem can also affect the KMM algorithm. Finally, let us point out that even though the FE algorithm also achieved performances similar to GDM on the sentiment and image adaptation, its performance was far from optimal adapting on the `kin-8xy` task. Since there is a lack of theoretical understanding for this algorithm, it is hard to characterize the scenarios where FE would perform better than GDM.

7. Conclusion

We presented a new theoretically well-founded domain adaptation algorithm seeking to minimize a less conservative quantity than the DM algorithm. We presented an SDP so-

lution for the particular case of the L_2 loss which can be solved in polynomial time. Our empirical results show that our new algorithm always performs better than or is on par with the otherwise state-of-the-art DM algorithm. We also provided tight generalization bounds for the domain adaptation problem based on the $\mathcal{Y}$-discrepancy. As pointed out in Section 4, an algorithm that minimizes the $\mathcal{Y}$-discrepancy would benefit from the best possible guarantees. However, the lack of labeled data from the target distribution makes this algorithm not viable. This suggests analyzing a richer scenario where the learner is allowed to ask for a limited number of labels from the target distribution. This setup, which is related to active learning, seems to be in fact the closest one to real-life applications and has started to receive attention from the research community (Berlind and Urner, 2015). We believe that the discrepancy disc will play a central role in the analysis of that scenario as well.

Acknowledgments

We would like to thank the reviewers of KDD and JMLR for their suggestions to improve this paper. This work was partly funded by the NSF awards IIS-1117591 and CCF-1535987.

Appendix A. SDP Formulation

Lemma 19 *The Lagrangian dual of the problem*

$$\max_{\substack{\mathbf{a}\in\mathbb{R}^m \\ \|\mathbf{K}_s\mathbf{a}-\mathbf{y}\|^2\leq r^2}} \frac{1}{2}\|\mathbf{K}_{st}\mathbf{a}\|^2 - \mathbf{b}^\top\mathbf{K}_t\mathbf{K}_{st}\mathbf{a}, \tag{26}$$

is given by

$$\begin{array}{ll}\min\limits_{\eta\geq 0,\gamma} & \gamma \\ s.\ t. & \begin{pmatrix} -\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}+\eta\mathbf{K}_s^2 & \frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b}-\eta\mathbf{K}_s\mathbf{y} \\ \frac{1}{2}\mathbf{b}^\top\mathbf{K}_t\mathbf{K}_{st}-\eta\mathbf{y}^\top\mathbf{K}_s & \eta(\|\mathbf{y}\|^2-r^2)+\gamma \end{pmatrix} \succeq 0.\end{array}$$

Furthermore, the duality gap for these problems is zero.

Proof For $\eta\geq 0$ the Lagrangian of (26) is given by

$$\begin{aligned} L(\mathbf{a},\eta) &= \frac{1}{2}\|\mathbf{K}_{st}\mathbf{a}\|^2 - \mathbf{b}^\top\mathbf{K}_t\mathbf{K}_{st}\mathbf{a} - \eta(\|\mathbf{K}_s\mathbf{a}-\mathbf{y}\|^2 - r^2) \\ &= \mathbf{a}^\top\Big(\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st} - \eta\mathbf{K}_s^2\Big)\mathbf{a} + (2\eta\mathbf{K}_s\mathbf{y} - \mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b})^\top\mathbf{a} - \eta(\|\mathbf{y}\|^2 - r^2). \end{aligned}$$

Since the Lagrangian is a quadratic function of $\mathbf{a}$ and that the conjugate function of a quadratic can be expressed in terms of the pseudo-inverse, the dual is given by

$$\begin{array}{ll}\min\limits_{\eta\geq 0} & \frac{1}{4}(2\eta\mathbf{K}_s\mathbf{y} - \mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b})^\top\Big(\eta\mathbf{K}_s^2 - \frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}\Big)^\dagger(2\eta\mathbf{K}_s\mathbf{y} - \mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b}) - \eta(\|\mathbf{y}\|^2 - r^2) \\ \text{s. t.} & \eta\mathbf{K}_s^2 - \frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st} \succeq 0.\end{array}$$

Introducing the variable γ to replace the objective function yields the equivalent problem

$$\begin{array}{ll}\min\limits_{\eta\geq 0,\gamma} & \gamma \\ \text{s. t.} & \eta\mathbf{K}_s^2 - \frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st} \succeq 0 \\ & \gamma - \frac{1}{4}(2\eta\mathbf{K}_s\mathbf{y} - \mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b})^\top\Big(\eta\mathbf{K}_s^2 - \frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}\Big)^\dagger(2\eta\mathbf{K}_s\mathbf{y} - \mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b}) + \eta(\|\mathbf{y}\|^2 - r^2) \geq 0\end{array}$$

Finally, by the properties of the Schur complement (Boyd and Vandenberghe, 2004), the two constraints above are equivalent to

$$\begin{pmatrix} -\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}+\eta\mathbf{K}_s^2 & \frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b}-\eta\mathbf{K}_s\mathbf{y} \\ \Big(\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b}-\eta\mathbf{K}_s\mathbf{y}\Big)^\top & \eta(\|\mathbf{y}\|^2-r)+\gamma \end{pmatrix} \succeq 0.$$

Since duality holds for a general QCQP with only one constraint (Boyd and Vandenberghe, 2004)[Appendix B], the duality gap between these problems is 0. ■

Proposition 20 *The optimization problem* (23) *is equivalent to the following SDP:*

$$\begin{array}{ll}\max\limits_{\alpha,\beta,\nu,\mathbf{Z},\mathbf{z}} & \frac{1}{2}\operatorname{Tr}(\mathbf{K}_{st}^\top\mathbf{K}_{st}\mathbf{Z}) - \beta - \alpha \\ s.\ t. & \begin{pmatrix} \nu\mathbf{K}_s^2+\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}-\frac{1}{4}\widetilde{\mathbf{K}} & \nu\mathbf{K}_s\mathbf{y}+\frac{1}{4}\widetilde{\mathbf{K}}\mathbf{z} \\ \nu\mathbf{y}^\top\mathbf{K}_s+\frac{1}{4}\mathbf{z}^\top\widetilde{\mathbf{K}} & \alpha+\nu(\|\mathbf{y}\|^2-r^2) \end{pmatrix} \succeq 0 \quad\wedge\quad \begin{pmatrix} \mathbf{Z} & \mathbf{z} \\ \mathbf{z}^\top & 1 \end{pmatrix} \succeq 0 \\ & \begin{pmatrix} \lambda\mathbf{K}_t+\mathbf{K}_t^2 & \frac{1}{2}\mathbf{K}_t\mathbf{K}_{st}\mathbf{z} \\ \frac{1}{2}\mathbf{z}^\top\mathbf{K}_{st}^\top\mathbf{K}_t & \beta \end{pmatrix} \succeq 0 \ \wedge\ \operatorname{Tr}(\mathbf{K}_s^2\mathbf{Z}) - 2\mathbf{y}^\top\mathbf{K}_s\mathbf{z} + \|\mathbf{y}\|^2 \leq r^2 \ \wedge\ \nu\geq 0,\end{array}$$

where $\widetilde{\mathbf{K}} = \mathbf{K}_{st}^\top\mathbf{K}_t(\lambda\mathbf{K}_t+\mathbf{K}_t^2)^\dagger\mathbf{K}_t\mathbf{K}_{st}$.

Proof

By Lemma 16, we may rewrite (23) as

$$\begin{array}{ll}\min\limits_{\mathbf{a},\gamma,\eta,\mathbf{b}} & \mathbf{b}^\top(\lambda\mathbf{K}_t+\mathbf{K}_t^2)\mathbf{b} + \frac{1}{2}\mathbf{a}^\top\mathbf{K}_{st}^\top\mathbf{K}_{st}\mathbf{a} - \mathbf{a}^\top\mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b} + \gamma \\ \text{s. t.} & \begin{pmatrix} -\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}+\eta\mathbf{K}_s^2 & \frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b}-\eta\mathbf{K}_s\mathbf{y} \\ \frac{1}{2}\mathbf{b}^\top\mathbf{K}_t\mathbf{K}_{st}-\eta\mathbf{y}^\top\mathbf{K}_s & \eta(\|\mathbf{y}\|^2-r^2)+\gamma \end{pmatrix} \quad\wedge\quad \eta\geq 0 \\ & \|\mathbf{K}_s\mathbf{a}-\mathbf{y}\|^2 \leq r^2.\end{array} \tag{27}$$

Let us apply the change of variables $\mathbf{b} = \frac{1}{2}(\lambda\mathbf{K}_t+\mathbf{K}_t^2)^\dagger\mathbf{K}_t\mathbf{K}_{st}\mathbf{a}+\mathbf{v}$. The following equalities can be easily verified.

$$\begin{aligned} \mathbf{b}^\top(\lambda\mathbf{K}_t+\mathbf{K}_t^2)\mathbf{b} &= \frac{1}{4}\mathbf{a}^\top\mathbf{K}_{st}^\top\mathbf{K}_t(\lambda\mathbf{K}_t+\mathbf{K}_t^2)^\dagger\mathbf{K}_t\mathbf{K}_{st}\mathbf{a} + \mathbf{v}^\top\mathbf{K}_t\mathbf{K}_{st}\mathbf{a} + \mathbf{v}^\top(\lambda\mathbf{K}_t+\mathbf{K}_t^2)\mathbf{v}. \\ \mathbf{a}^\top\mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{b} &= \frac{1}{2}\mathbf{a}^\top\mathbf{K}_{st}^\top\mathbf{K}_t(\lambda\mathbf{K}_t+\mathbf{K}_t^2)^\dagger\mathbf{K}_t\mathbf{K}_{st}\mathbf{a} + \mathbf{v}^\top\mathbf{K}_t\mathbf{K}_{st}\mathbf{a}. \end{aligned}$$

Thus, replacing $\mathbf{b}$ on (27) yields

$$\min_{\mathbf{a},\mathbf{v},\gamma,\eta} \ \mathbf{v}^\top(\lambda\mathbf{K}_t+\mathbf{K}_t^2)\mathbf{v}+\ \mathbf{a}^\top\Big(\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}-\frac{1}{4}\widetilde{\mathbf{K}}\Big)\mathbf{a}+\gamma$$
$$\text{s. t.} \quad \begin{pmatrix} -\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}+\eta\mathbf{K}_s^2 & \frac{1}{4}\widetilde{\mathbf{K}}\mathbf{a}+\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{v}-\eta\mathbf{K}_s\mathbf{y} \\ \frac{1}{4}\mathbf{a}^\top\widetilde{\mathbf{K}}+\frac{1}{2}\mathbf{v}^\top\mathbf{K}_t\mathbf{K}_{st}-\eta\mathbf{y}^\top\mathbf{K}_s & \eta(\|\mathbf{y}\|^2-r^2)+\gamma \end{pmatrix}\succeq 0 \quad\wedge\quad \eta\geq 0$$
$$\|\mathbf{K}_s\mathbf{a}-\mathbf{y}\|^2\leq r^2.$$

Introducing the scalar multipliers $\mu,\nu\geq 0$ and the matrix

$$\begin{pmatrix} \mathbf{Z} & \mathbf{z} \\ \mathbf{z}^\top & \widetilde{z}, \end{pmatrix}\succeq 0$$

as a multiplier for the matrix constraint, we can form the Lagrangian:

$$\mathfrak{L}:=\mathbf{v}^\top(\lambda\mathbf{K}_t+\mathbf{K}_t^2)\mathbf{v}+\mathbf{a}^\top\Big(\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}-\frac{1}{4}\widetilde{\mathbf{K}}\Big)\mathbf{a}+\gamma-\mu\eta+\nu(\|\mathbf{K}_s\mathbf{a}-\mathbf{y}\|^2-r^2)$$
$$-\operatorname{Tr}\left(\begin{pmatrix} \mathbf{Z} & \mathbf{z} \\ \mathbf{z} & \widetilde{z} \end{pmatrix}\begin{pmatrix} -\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}+\eta\mathbf{K}_s^2 & \frac{1}{4}\widetilde{\mathbf{K}}\mathbf{a}+\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{v}-\eta\mathbf{K}_s\mathbf{y} \\ \frac{1}{4}\mathbf{a}^\top\widetilde{\mathbf{K}}+\frac{1}{2}\mathbf{v}^\top\mathbf{K}_t\mathbf{K}_{st}-\eta\mathbf{y}^\top\mathbf{K}_s & \eta(\|\mathbf{y}\|^2-r^2)+\gamma \end{pmatrix}\right).$$

The KKT conditions $\frac{\partial\mathfrak{L}}{\partial\eta}=\frac{\partial\mathfrak{L}}{\partial\gamma}=0$ trivially imply $\widetilde{z}=1$ and $\operatorname{Tr}(\mathbf{K}_s^2\mathbf{Z})-2\mathbf{y}^\top\mathbf{K}_s\mathbf{z}+\|\mathbf{y}\|^2-r^2+\mu=0$. These constraints on the dual variables guarantee that the primal variables η and γ will vanish from the Lagrangian, thus yielding

$$\mathfrak{L}=\frac{1}{2}\operatorname{Tr}(\mathbf{K}_{st}^\top\mathbf{K}_{st}\mathbf{Z})+\nu(\|\mathbf{y}\|^2-r^2)+\mathbf{v}^\top(\lambda\mathbf{K}_t+\mathbf{K}_t^2)\mathbf{v}^\top-\mathbf{z}^\top\mathbf{K}_{st}^\top\mathbf{K}_t\mathbf{v}$$
$$+\mathbf{a}^\top\Big(\nu\mathbf{K}_s^2+\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}-\frac{1}{4}\widetilde{\mathbf{K}}\Big)\mathbf{a}-\Big(2\nu\mathbf{K}_s\mathbf{y}+\frac{1}{2}\widetilde{\mathbf{K}}\mathbf{z}\Big)^\top\mathbf{a}.$$

This is a quadratic function on the primal variables $\mathbf{a}$ and $\mathbf{v}$ with minimizing solutions

$$\mathbf{a}=\frac{1}{2}\Big(\nu\mathbf{K}_s^2+\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}-\frac{1}{4}\widetilde{\mathbf{K}}\Big)^\dagger\Big(2\nu\mathbf{K}_s\mathbf{y}+\frac{1}{2}\widetilde{\mathbf{K}}\mathbf{z}\Big)\qquad\text{and}\qquad\mathbf{v}=\frac{1}{2}(\lambda\mathbf{K}_t+\mathbf{K}_t^2)^\dagger\mathbf{K}_t\mathbf{K}_{st}\mathbf{z},$$

and optimal value equal to the objective of the Lagrangian dual:

$$\frac{1}{2}\operatorname{Tr}(\mathbf{K}_{st}^\top\mathbf{K}_{st}\mathbf{Z})+\nu(\|\mathbf{y}\|^2-r^2)-\frac{1}{4}\mathbf{z}^\top\widetilde{\mathbf{K}}\mathbf{z}$$
$$-\frac{1}{4}\Big(2\nu\mathbf{K}_s\mathbf{y}+\frac{1}{2}\widetilde{\mathbf{K}}\mathbf{z}\Big)^\top\Big(\nu\mathbf{K}_s^2+\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}-\frac{1}{4}\widetilde{\mathbf{K}}\Big)^\dagger\Big(2\nu\mathbf{K}_s\mathbf{y}+\frac{1}{2}\widetilde{\mathbf{K}}\mathbf{z}\Big).$$

As in Lemma 16, we apply the properties of the Schur complement to show that the dual is given by

$$\max_{\alpha,\beta,\nu,\mathbf{Z},\mathbf{z}} \ \frac{1}{2}\operatorname{Tr}(\mathbf{K}_{st}^\top\mathbf{K}_{st}\mathbf{Z})-\beta-\alpha$$
$$\text{s. t}\ \begin{pmatrix} \nu\mathbf{K}_s^2+\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}-\frac{1}{4}\widetilde{\mathbf{K}} & \nu\mathbf{K}_s\mathbf{y}+\frac{1}{4}\widetilde{\mathbf{K}}\mathbf{z} \\ \nu\mathbf{y}^\top\mathbf{K}_s+\frac{1}{4}\mathbf{z}^\top\widetilde{\mathbf{K}} & \alpha+\nu(\|\mathbf{y}\|^2-r^2) \end{pmatrix}\succeq 0 \quad\wedge\quad \begin{pmatrix} \mathbf{Z} & \mathbf{z} \\ \mathbf{z}^\top & 1 \end{pmatrix}\succeq 0$$
$$\operatorname{Tr}(\mathbf{K}_s^2\mathbf{Z})-2\mathbf{y}^\top\mathbf{K}_s\mathbf{z}+\|\mathbf{y}\|^2\leq r^2 \quad\wedge\quad \beta\geq\frac{1}{4}\mathbf{z}^\top\widetilde{\mathbf{K}}\mathbf{z} \quad\wedge\quad \nu\geq 0.$$

Finally, recalling the definition of $\widetilde{\mathbf{K}}$ and using the Schur complement one more time we arrive to the final SDP formulation

$$\max_{\alpha,\beta,\nu,\mathbf{Z},\mathbf{z}} \ \frac{1}{2}\operatorname{Tr}(\mathbf{K}_{st}^\top\mathbf{K}_{st}\mathbf{Z})-\beta-\alpha$$
$$\text{s. t}\ \begin{pmatrix} \nu\mathbf{K}_s^2+\frac{1}{2}\mathbf{K}_{st}^\top\mathbf{K}_{st}-\frac{1}{4}\widetilde{\mathbf{K}} & \nu\mathbf{K}_s\mathbf{y}+\frac{1}{4}\widetilde{\mathbf{K}}\mathbf{z} \\ \nu\mathbf{y}^\top\mathbf{K}_s+\frac{1}{4}\mathbf{z}^\top\widetilde{\mathbf{K}} & \alpha+\nu(\|\mathbf{y}\|^2-r^2) \end{pmatrix}\succeq 0 \quad\wedge\quad \begin{pmatrix} \mathbf{Z} & \mathbf{z} \\ \mathbf{z}^\top & 1 \end{pmatrix}\succeq 0$$
$$\begin{pmatrix} \lambda\mathbf{K}_t+\mathbf{K}_t^2 & \frac{1}{2}\mathbf{K}_t\mathbf{K}_{st}\mathbf{z} \\ \frac{1}{2}\mathbf{z}^\top\mathbf{K}_{st}^\top\mathbf{K}_t & \beta \end{pmatrix}\succeq 0 \ \wedge\ \operatorname{Tr}(\mathbf{K}_s^2\mathbf{Z})-2\mathbf{y}^\top\mathbf{K}_s\mathbf{z}+\|\mathbf{y}\|^2\leq r^2 \ \wedge\ \nu\geq 0.$$

■

Appendix B. QP Formulation

Proposition 21 *Let $\mathbf{Y}=(Y_{ij})\in\mathbb{R}^{n\times k}$ be the matrix defined by $Y_{ij}=n^{-1/2}h_j(x_i')$ and $\mathbf{y}'=(y_1',\ldots,y_k')^\top\in\mathbb{R}^k$ the vector defined by $y_i'=n^{-1}\sum_{j=1}^n h_i(x_j')^2$. Then, the dual problem of (24) is given by*

$$\max_{\boldsymbol{\alpha},\boldsymbol{\gamma},\beta} \ -\Big(\mathbf{Y}\boldsymbol{\alpha}+\frac{\boldsymbol{\gamma}}{2}\Big)^\top\mathbf{K}_t\Big(\lambda\mathbf{I}+\frac{1}{2}\mathbf{K}_t\Big)^{-1}\Big(\mathbf{Y}\boldsymbol{\alpha}+\frac{\boldsymbol{\gamma}}{2}\Big)-\frac{1}{2}\boldsymbol{\gamma}^\top\mathbf{K}_t\mathbf{K}_t^\dagger\boldsymbol{\gamma}+\boldsymbol{\alpha}^\top\mathbf{y}'-\beta \qquad (28)$$
$$s.t.\ \mathbf{1}^\top\boldsymbol{\alpha}=\frac{1}{2},\qquad \mathbf{1}\beta\geq-\mathbf{Y}^\top\boldsymbol{\gamma},\qquad \boldsymbol{\alpha}\geq 0,$$

where $\mathbf{1}$ is the vector in $\mathbb{R}^k$ with all components equal to 1. Furthermore, the solution h of (24) can be recovered from a solution $(\boldsymbol{\alpha},\boldsymbol{\gamma},\beta)$ of (28) by $\forall x, h(x)=\sum_{i=1}^n a_iK(x_i,x)$, where $\boldsymbol{a}=\big(\lambda\mathbf{I}+\frac{1}{2}\mathbf{K}_t\big)^{-1}(\mathbf{Y}\boldsymbol{\alpha}+\frac{1}{2}\boldsymbol{\gamma})$.

We will first prove a simplified version of the proposition for the case of linear hypotheses, i.e. we can represent hypotheses in $\mathbb{H}$ and elements of $\mathcal{X}$ as vectors $\mathbf{w},\mathbf{x}\in\mathbb{R}^d$ respectively. Define $\mathbf{X}'=n^{-1/2}(\mathbf{x}_1',\ldots,\mathbf{x}_n')$ to be the matrix whose columns are the normalized sample points from the target distribution. Let also $\{\mathbf{w}_1,\ldots,\mathbf{w}_k\}$ be a sample taken from $\partial H''$ and define $\boldsymbol{W}:=(\mathbf{w}_1,\ldots,\mathbf{w}_k)\in\mathbb{R}^{d\times k}$. Under this notation, problem (24) may be rewritten as

$$\min_{\mathbf{w}\in\mathbb{R}^d}\lambda\|\mathbf{w}\|^2+\frac{1}{2}\max_{i=1,\ldots,k}\|\mathbf{X}'^\top(\mathbf{w}-\mathbf{w}_i)\|^2+\frac{1}{2}\min_{\mathbf{w}'\in\mathcal{C}}\|\mathbf{X}'^\top(\mathbf{w}-\mathbf{w}')\|^2. \qquad (29)$$

Lemma 22 *The Lagrange dual of problem (29) is given by*

$$\max_{\boldsymbol{\alpha},\boldsymbol{\gamma},\beta}-\Big(\mathbf{Y}\boldsymbol{\alpha}+\frac{\boldsymbol{\gamma}}{2}\Big)^\top\mathbf{X}'^\top\Big(\lambda\mathbf{I}+\frac{\mathbf{X}'\mathbf{X}'^\top}{2}\Big)^{-1}\mathbf{X}'\Big(\boldsymbol{Y}\boldsymbol{\alpha}+\frac{\boldsymbol{\gamma}}{2}\Big)-\frac{1}{2}\boldsymbol{\gamma}^\top\mathbf{X}'^\top(\mathbf{X}'\mathbf{X}'^\top)^\dagger\mathbf{X}'\boldsymbol{\gamma}+\boldsymbol{\alpha}^\top\mathbf{y}'-\beta$$
$$s.\ t.\ \mathbf{1}^\top\boldsymbol{\alpha}=\frac{1}{2}\qquad\mathbf{1}\beta\geq-\mathbf{Y}^\top\boldsymbol{\gamma}\qquad\boldsymbol{\alpha}\geq 0,$$

where $\mathbf{Y}=\mathbf{X}'^\top\boldsymbol{W}$ and $\mathbf{y}_i'=\|\mathbf{X}'^\top\mathbf{w}_i\|^2$.

Proof By applying the change of variable $\mathbf{u} = \mathbf{w}' - \mathbf{w}$, problem (29) is can be made equivalent to

$$\min_{\mathbf{w}\in\mathbb{R}^d \mathbf{u}\in\mathcal{C}-\mathbf{w}} \lambda\|\mathbf{w}\|^2 + \frac{1}{2}\|\mathbf{X}'^\top\mathbf{w}\|^2 + \frac{1}{2}\|\mathbf{X}'^\top\mathbf{u}\|^2 + \frac{1}{2}\max_{i=1,\ldots,k}\|\mathbf{X}'^\top\mathbf{w}_i\|^2 - 2\mathbf{w}_i^\top\mathbf{X}'\mathbf{X}'^\top\mathbf{w}.$$

By making the constraints on $\mathbf{u}$ explicit and replacing the maximization term with the variable r the above problem becomes

$$\begin{aligned}\min_{\mathbf{w},\mathbf{u},r,\boldsymbol{\mu}} \quad & \lambda\|\mathbf{w}\|^2 + \frac{1}{2}\|\mathbf{X}'^\top\mathbf{w}\|^2 + \frac{1}{2}\|\mathbf{X}'^\top\mathbf{u}\|^2 + \frac{1}{2}r\\ \text{s. t.} \quad & \mathbf{1}r \geq \mathbf{y}' - 2\mathbf{Y}^\top\mathbf{X}'^\top\mathbf{w} \quad \wedge \quad \mathbf{1}^\top\boldsymbol{\mu} = 1 \quad \wedge \quad \boldsymbol{\mu} \geq 0 \quad \wedge \quad \boldsymbol{W}\boldsymbol{\mu} - \mathbf{w} = \mathbf{u}.\end{aligned}$$

For $\boldsymbol{\alpha}, \boldsymbol{\delta} \geq 0$, the Lagrangian of this problem is defined as

$$\begin{aligned}\mathfrak{L}(\mathbf{w},\mathbf{u},\boldsymbol{\mu},r,\boldsymbol{\alpha},\beta,\boldsymbol{\delta},\boldsymbol{\gamma}') = {} & \lambda\|\mathbf{w}\|^2 + \frac{1}{2}\|\mathbf{X}'^\top\mathbf{w}\|^2 + \frac{1}{2}\|\mathbf{X}'^\top\mathbf{u}\|^2 + \frac{1}{2}r + \beta(\mathbf{1}^\top\boldsymbol{\mu} - 1)\\ & + \boldsymbol{\alpha}^\top(\mathbf{y}' - 2(\mathbf{X}'\mathbf{Y})^\top\mathbf{w} - \mathbf{1}r) - \boldsymbol{\delta}^\top\boldsymbol{\mu} + \boldsymbol{\gamma}'^\top(\boldsymbol{W}\boldsymbol{\mu} - \mathbf{w} - \mathbf{u}).\end{aligned}$$

Minimizing with respect to the primal variables yields the following KKT conditions:

$$\mathbf{1}^\top\boldsymbol{\alpha} = \frac{1}{2} \qquad \mathbf{1}\beta = \boldsymbol{\delta} - \boldsymbol{W}^\top\boldsymbol{\gamma}'. \tag{30}$$

$$\mathbf{X}'\mathbf{X}'^\top\mathbf{u} = \boldsymbol{\gamma}' \qquad 2\left(\lambda\mathbf{I} + \frac{\mathbf{X}'\mathbf{X}'^\top}{2}\right)\mathbf{w} = 2(\mathbf{X}'\boldsymbol{Y})\boldsymbol{\alpha} + \boldsymbol{\gamma}'. \tag{31}$$

Condition (30) implies that the terms involving r and $\boldsymbol{\mu}$ will vanish from the Lagrangian. Furthermore, the first equation in (31) implies that any feasible $\boldsymbol{\gamma}'$ must satisfy $\boldsymbol{\gamma}' = \mathbf{X}'\boldsymbol{\gamma}$ for some $\gamma \in \mathbb{R}^n$. Finally, it is immediate that $\boldsymbol{\gamma}'^\top\mathbf{u} = \mathbf{u}^\top\mathbf{X}'\mathbf{X}'^\top\mathbf{u}$ and $2\mathbf{w}^\top\left(\lambda\mathbf{I} + \frac{\mathbf{X}'\mathbf{X}'^\top}{2}\right)\mathbf{w} = 2\boldsymbol{\alpha}^\top(\mathbf{X}'\mathbf{Y})^\top\mathbf{w} + \boldsymbol{\gamma}'^\top\mathbf{w}$. Thus, at the optimal point, the Lagrangian becomes

$$\begin{aligned}& -\mathbf{w}^\top\left(\lambda\mathbf{I} + \frac{1}{2}\mathbf{X}'\mathbf{X}'^\top\right)\mathbf{w} - \frac{1}{2}\mathbf{u}^\top\mathbf{X}'\mathbf{X}'^\top\mathbf{u} + \boldsymbol{\alpha}^\top\mathbf{y}' - \beta\\ \text{s. t.} \quad & \mathbf{1}^\top\boldsymbol{\alpha} = \frac{1}{2} \qquad \mathbf{1}\beta = \boldsymbol{\delta} - \boldsymbol{W}^\top\boldsymbol{\gamma}' \qquad \boldsymbol{\alpha} \geq 0 \wedge \boldsymbol{\delta} \geq 0.\end{aligned}$$

The positivity of $\boldsymbol{\delta}$ implies that $\mathbf{1}\beta \geq -\boldsymbol{W}^\top\boldsymbol{\gamma}'$. Solving for $\mathbf{w}$ and $\mathbf{u}$ on (31) and applying the change of variable $\mathbf{X}'\boldsymbol{\gamma} = \boldsymbol{\gamma}'$ we obtain the final expression for the dual problem:

$$\begin{aligned}\max_{\boldsymbol{\alpha},\boldsymbol{\gamma},\beta} \; & -\left(\mathbf{Y}\boldsymbol{\alpha} + \frac{\boldsymbol{\gamma}}{2}\right)^\top\mathbf{X}'^\top\left(\lambda\mathbf{I} + \frac{\mathbf{X}'\mathbf{X}'^\top}{2}\right)^{-1}\mathbf{X}'\left(\mathbf{Y}\boldsymbol{\alpha} + \frac{\boldsymbol{\gamma}}{2}\right) - \frac{1}{2}\boldsymbol{\gamma}^\top\mathbf{X}'^\top(\mathbf{X}'\mathbf{X}'^\top)^\dagger\mathbf{X}'\boldsymbol{\gamma} + \boldsymbol{\alpha}^\top\mathbf{y}' - \beta\\ \text{s. t.} \; & \mathbf{1}^\top\boldsymbol{\alpha} = \frac{1}{2} \qquad \mathbf{1}\beta \geq -\mathbf{Y}^\top\boldsymbol{\gamma} \qquad \boldsymbol{\alpha} \geq 0,\end{aligned}$$

where we have used the fact that $\mathbf{Y}^\top\boldsymbol{\gamma} = \mathbf{W}\mathbf{X}'^\top\boldsymbol{\gamma}$ to simplify the constraints. Notice also that we can recover the solution $\mathbf{w}$ of problem (29) as $\mathbf{w} = (\lambda\mathbf{I} + \frac{1}{2}\mathbf{X}'^\top\mathbf{X}')^{-1}\mathbf{X}'(\mathbf{Y}\boldsymbol{\alpha} + \frac{1}{2}\boldsymbol{\gamma})$ ■

The proof of Proposition 18 follows from a straightforward application of the well known matrix identities $\mathbf{X}'(\lambda\mathbf{I} + \mathbf{X}'^\top\mathbf{X}')^{-1} = (\lambda\mathbf{I} + \mathbf{X}'\mathbf{X}'^\top)^{-1}\mathbf{X}'$ and $\mathbf{X}'^\top\mathbf{X}'(\mathbf{X}'^\top\mathbf{X}')^\dagger = \mathbf{X}'^\top(\mathbf{X}'\mathbf{X}'^\top)^\dagger\mathbf{X}'$, and by the fact that the kernel matrix $\mathbf{K}_t$ is equal to $\mathbf{X}'^\top\mathbf{X}'$.

Appendix C. μ-admissibility

Lemma 23 *Assume that $L_p(h(x), y) \leq M$ for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, then L_p is μ-admissible with $\mu = pM^{p-1}$.*

Proof Since $x \mapsto x^p$ is p-Lipschitz over $[0, 1]$ we can write

$$\begin{aligned}|L(h(x), y) - L(h'(x), y)| &= M^p\left|\left(\frac{|h(x) - y|}{M}\right)^p - \left(\frac{|h'(x) - y|}{M}\right)^p\right|\\ &\leq pM^{p-1}|h(x) - y + y - h'(x)| = pM^{p-1}|h(x) - h'(x)|.\end{aligned}$$

■

Lemma 24 *Let L be the L_p loss for some $p \geq 1$ and let h, h', h'' be functions satisfying $L_p(h(x), h'(x)) \leq M$ and $L_p(h''(x), h'(x)) \leq M$ for all $x \in \mathcal{X}$, for some $M \geq 0$. Then, for any distribution $\mathcal{D}$ over $\mathcal{X}$, the following inequality holds:*

$$|\mathcal{L}_\mathcal{D}(h, h') - \mathcal{L}_\mathcal{D}(h'', h')| \leq pM^{p-1}[\mathcal{L}_\mathcal{D}(h, h'')]^{\frac{1}{p}}. \tag{32}$$

Proof Proceeding as in the proof of Lemma 23, we obtain

$$\begin{aligned}|\mathcal{L}_\mathcal{D}(h, h') - \mathcal{L}_\mathcal{D}(h'', h')| &= |\operatorname*{E}_{x\in\mathcal{D}}\left[L_p(h(x), h'(x)) - L_p(h''(x), h'(x)\right]|\\ &\leq pM^{p-1}\operatorname*{E}_{x\in\mathcal{D}}\left[|h(x) - h''(x)|\right].\end{aligned}$$

Since $p \geq 1$, by Jensen's inequality, we can write $\operatorname{E}_{x\in\mathcal{D}}\left[|h(x) - h''(x)|\right] \leq \operatorname{E}_{x\in\mathcal{D}}\left[|h(x) - h''(x)|^p\right]^{1/p} = [\mathcal{L}_\mathcal{D}(h, h'')]^{\frac{1}{p}}$. ■

References

Shai Ben-David and Ruth Urner. On the hardness of domain adaptation and the utility of unlabeled target samples. In *Proceedings of ALT*, pages 139–153, 2012.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Proceedings of NIPS*, pages 137–144, 2006.

Shai Ben-David, Tyler Lu, Teresa Luu, and Dávid Pál. Impossibility theorems for domain adaptation. *JMLR - Proceedings Track*, 9:129–136, 2010.

Christopher Berlind and Ruth Urner. Active nearest neighbors in changing environments. In *Proceedings of ICML*, pages 1870–1879, 2015.

Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of ICML*, pages 81–88, 2007.

John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL*, 2007.

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

Corinna Cortes and Mehryar Mohri. Domain adaptation in regression. In *Proceedings of ALT*, 2011.

Corinna Cortes and Mehryar Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theoretical Computer Science*, 9474, 2013.

Corinna Cortes, Mehryar Mohri, Michael Riley, and Afshin Rostamizadeh. Sample selection bias correction theory. In *Proceedings of ALT*, pages 38–53, 2008.

Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. In *Proceedings of NIPS*, pages 442–450, 2010.

Hal Daumé III. Frustratingly easy domain adaptation. In *Proceedings of ACL*, 2007.

Luc Devroye, Lázló Györfi, and Gábor Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

Mark Dredze, John Blitzer, Partha Pratim Talukdar, Kuzman Ganchev, João Graça, and Fernando Pereira. Frustratingly hard domain adaptation for dependency parsing. In *EMNLP-CoNLL*, 2007.

Pascal Germain, Amaury Habrard, François Laviolette, and Emilie Morvant. A PAC-Bayesian approach for domain adaptation with specialization to linear classifiers. In *Proceedings of ICML*, 2013.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. A kernel two-sample test. *JMLR*, 13:723–773, 2012.

Judy Hoffman, Trevor Darrell, and Kate Saenko. Continuous manifold based adaptation for evolving visual domains. In *Proceedings of IEEE CVPR*, pages 867–874, 2014.

Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Correcting sample selection bias by unlabeled data. In *Proceedings of NIPS*, volume 19, pages 601–608, 2006.

Jing Jiang and ChengXiang Zhai. Instance Weighting for Domain Adaptation in NLP. In *Proceedings of ACL*, pages 264–271, 2007.

Piyush Kumar, Joseph S. B. Mitchell, and E. Alper Yildirim. Computing core-sets and approximate smallest enclosing hyperspheres in high dimensions. In *ALENEX, Lecture Notes Comput. Sci*, pages 45–55, 2003.

C. J. Leggetter and Philip C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech & Language*, 9(2):171–185, 1995.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *Proceedings of COLT*. Omnipress, 2009.

Aleix M. Martínez. Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class. *IEEE Trans. Pattern Anal.*, 24(6), 2002.

Andrés Muñoz Medina. *Learning Theory and Algorithms for Auctioning and Adaptation Problems*. PhD thesis, New York University, 2015.

Mehryar Mohri and Andres Muñoz. New analysis and algorithm for learning with drifting distributions. In *Proceedings of ALT*. Springer, 2012.

Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Trans. on Neural Networks*, 22(2):199–210, 2011.

Carl Edward Rasmussen, Radford M. Neal, Geoffrey Hinton, Drew van Camp, Michael Revow Zoubin Ghahramani, Rafal Kustra, and Rob Tibshirani. The delve project. `http://www.cs.toronto.edu/~delve/data/datasets.html`, 1996. version 1.0.

Sven Schőnherr. *Quadratic Programming in Geometric Optimization: Theory, Implementation, and Applications*. PhD thesis, Swiss Federal Institute of Technology, 2002.

Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul von Bünau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Proceedings of NIPS*, pages 1433–1440, 2007.

Tatiana Tommasi, Tinne Tuytelaars, and Barbara Caputo. A testbed for cross-dataset analysis. *CoRR*, abs/1402.5923, 2014. URL `http://arxiv.org/abs/1402.5923`.

Leslie G. Valiant. *A Theory of the Learnable*. ACM Press New York, NY, USA, 1984.

Vladimir N. Vapnik. *Statistical Learning Theory*. J. Wiley & Sons, 1998.

E. Alper Yildirim. Two algorithms for the minimum enclosing ball problem. *SIAM Journal on Optimization*, 19(3):1368–1391, 2008.

Chao Zhang, Lei Zhang, and Jieping Ye. Generalization bounds for domain adaptation. In *Proceedings of NIPS*, pages 1790–1798. MIT Press, 2012.

Erheng Zhong, Wei Fan, Qiang Yang, Olivier Verscheure, and Jiangtao Ren. Cross validation framework to choose amongst models and datasets for transfer learning. In *Proceedings of ECML PKDD 2010 Part III*, pages 547–562, 2010.

Journal of Machine Learning Research 20 (2019) 1-52 Submitted 5/16; Revised 10/18; Published 2/19

Transport Analysis of Infinitely Deep Neural Network

Sho Sonoda SHO.SONODA@RIKEN.JP
Center for Advanced Intelligence Project
RIKEN
1–4–1 Nihonbashi, Chuo-ku, Tokyo 103–0027, Japan

Noboru Murata NOBORU.MURATA@EB.WASEDA.AC.JP
School of Advanced Science and Engineering
Waseda University
3–4–1 Okubo, Shinjuku-ku, Tokyo 169–8555, Japan

Editor: Yoshua Bengio

Abstract

We investigated the feature map inside deep neural networks (DNNs) by tracking the transport map. We are interested in the *role of depth*—why do DNNs perform better than shallow models?—and the *interpretation* of DNNs—what do intermediate layers do? Despite the rapid development in their application, DNNs remain analytically unexplained because the hidden layers are nested and the parameters are not faithful. Inspired by the *integral representation* of shallow NNs, which is the continuum limit of the width, or the hidden unit number, we developed the *flow representation* and *transport analysis* of DNNs. The flow representation is the continuum limit of the depth, or the hidden layer number, and it is specified by an ordinary differential equation (ODE) with a vector field. We interpret an ordinary DNN as a *transport map* or an Euler broken line approximation of the flow. Technically speaking, a dynamical system is a natural model for the nested feature maps. In addition, it opens a new way to the coordinate-free treatment of DNNs by avoiding the redundant parametrization of DNNs. Following *Wasserstein geometry*, we analyze a flow in three aspects: dynamical system, continuity equation, and Wasserstein gradient flow. A key finding is that we specified a series of transport maps of the *denoising autoencoder* (DAE), which is a cornerstone for the development of deep learning. Starting from the shallow DAE, this paper develops three topics: the transport map of the deep DAE, the equivalence between the stacked DAE and the composition of DAEs, and the development of the double continuum limit or the integral representation of the flow representation. As partial answers to the research questions, we found that deeper DAEs converge faster and the extracted features are better; in addition, a deep Gaussian DAE transports mass to decrease the Shannon entropy of the data distribution. We expect that further investigations on these questions lead to the development of an interpretable and principled alternatives to DNNs.

Keywords: representation learning, denoising autoencoder, flow representation, continuum limit, backward heat equation, Wasserstein geometry, ridgelet analysis

1. Introduction

Despite the rapid development in their application, deep neural networks (DNN) remain analytically unexplained. We are interested in the *role of depth—why do DNNs perform better than shallow models?*—and the *interpretation* of DNNs—*what do intermediate layers do?* To the best of our knowledge, thus far, traditional theories, such as the statistical learning theory (Vapnik, 1998), have not succeeded in completely answering the above questions (Zhang et al., 2018). Existing DNNs lack interpretability; hence, a DNN is often called a *blackbox*. In this study, we propose the *flow representation* and *transport analysis* of DNNs, which provide us with insights into why DNNs can perform better and facilitate our understanding of what DNNs do. We expect that these lines of study lead to the development of an interpretable and principled alternatives to DNNs.

Compared to other *shallow models*, such as kernel methods (Shawe-Taylor and Cristianini, 2004) and ensemble methods (Schapire and Freund, 2012), DNNs have at least two specific technical issues: the *function composition* and the *redundant and complicated parametrization*. First, a DNN is formally a composite $\boldsymbol{g}_L \circ \cdots \circ \boldsymbol{g}_0$ of intermediate maps $\boldsymbol{g}_\ell (\ell = 0, \dots, L)$. Here, each $\boldsymbol{g}_\ell$ corresponds to the ℓ-th hidden layer. Currently, our understanding of learning machines is based on *linear algebra*, i.e., the *basis and coefficients* (Vapnik, 1998). Linear algebra is compatible with shallow models because a shallow model is a linear combination of basis functions. However, it has poor compatibility with deep models because the function composition $(\boldsymbol{f}, \boldsymbol{g}) \mapsto \boldsymbol{f} \circ \boldsymbol{g}$ is not assumed in the standard definition of the linear space. Therefore, we should move to spaces where the function composition is defined, such as monoids, semigroups, and *dynamical systems*. Second, the standard parametrization of the NN, such as $\boldsymbol{g}_\ell(\boldsymbol{x}) = \sum_{j=1}^{p} \boldsymbol{c}_j^\ell \sigma(\boldsymbol{a}_j^\ell \cdot \boldsymbol{x} - b_j^\ell)$, is redundant because there exist different sets of parameters that specify the same function, which causes technical problems, such as local minima. Furthermore, it is complicated because the interpretation of parameters is usually impossible, which results in the blackbox nature of DNNs. Therefore, we need a new parametrization that is concise in the sense that different parameters specify different functions and simple in the sense that it is easy to understand.

For shallow NNs, the *integral representation theory* (Murata, 1996; Candès, 1998; Sonoda and Murata, 2017a) provides a concise and simple reparametrization. The integral representation is derived by a continuum limit of the *width* or the number of hidden units. Owing to the *ridgelet transform* or a pseudo-inverse operator of the integral representation operator, it is concise and simple (see Section 1.3.2 for further details on the ridgelet transform). Furthermore, in the integral representation, we can compute the parameters of the shallow NN that attains the *global minimum* of the backpropagation training (Sonoda et al., 2018). In the integral representation, thus far, the shallow NNs is no longer a blackbox, and the training is principled. However, the integral representation is again based on linear algebra, the scope of which does not include DNNs.

Inspired by the integral representation theory, we introduced the *flow representation* and developed the *transport analysis* of DNNs. The flow representation is derived by a continuum limit of the *depth* or the number of hidden layers. In the flow representation, we formulate a DNN as a flow of an ordinary differential equation (ODE) $\dot{\boldsymbol{x}}_t = \boldsymbol{v}_t(\boldsymbol{x}_t)$ with vector field $\boldsymbol{v}_t$. In addition, we introduced the *transport map* by which we call a discretization $\boldsymbol{x} \mapsto \boldsymbol{x} + \boldsymbol{f}_t(\boldsymbol{x})$ of the flow. Specifically, we regard the intermediate map $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^n$ of an ordinary DNN as a transport map that transfers the mass at $\boldsymbol{x} \in \mathbb{R}^m$ toward $\boldsymbol{g}(\boldsymbol{x}) \in \mathbb{R}^n$. Since the flow and transport map are independent of coordinates, they enable us the coordinate-free treatment of DNNs. In the transport analysis, following *Wasserstein geometry* (Villani, 2009), we track a flow by analyzing the three profiles of the

©2019 Sho Sonoda and Noboru Murata.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/16-243.html.

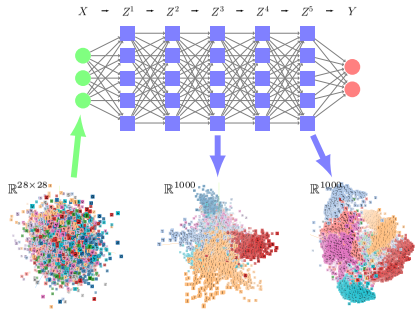


Figure 1: Mass transportation in a deep neural network that classifies images of digits. In the final hidden layer, the feature vectors have to be linearly separable because the output layer is just a linear classifier. Hence, through the network, the same digits gradually accumulate and different digits gradually separate.

flow: *dynamical system*, *pushforward measure*, and *Wasserstein gradient flow* (Ambrosio et al., 2008) (see Section 2 for further details).

We note that when the input and the output differ in dimension, i.e., $m \neq n$, we simply consider that both the input space and the output space are embedded in a common high-dimensional space. As a composite of transport maps leads to another transport map, the transport map has compatibility with deep structures. In this manner, transportation is a universal characteristic of DNNs. For example, let us consider a digit recognition problem with DNNs. We can expect the feature extractor in the DNN to be a transport map that separates the feature vectors of different digits, similar to the separation of *oil and water* (see Figure 1 for example). At the time of the initial submission in 2016, the flow representation seemed to be a novel viewpoint of DNNs. At present, it is the mainstream of development. For example, two important DNNs—residual network (ResNet) (He et al., 2016) and generative adversarial net (GAN) (Goodfellow et al., 2014)—are now considered to be transport maps (see Section 1.2 for a more detailed survey). Instead of directly investigating DNNs in terms of the redundant and complex parametrization, we perform transport analysis associated with the flow representation. We consider that the flow representation is potentially concise and simple because the flow is independent of parametrization, and it is specified by a single vector field $\boldsymbol{v}$.

In this study, we demonstrate transport analysis of the *denoising autoencoder (DAE)*. The DAE was introduced by Vincent et al. (2008) as a heuristic modification to enhance the robustness of the traditional autoencoder. The traditional autoencoder is an NN that is trained as an identity map $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x}$. The hidden layer of the network is used as a feature map, which is often called the "code" because the activation pattern appears to be random, but it surely encodes some information about the input data. On the other hand, the DAE is an NN that is trained as a "denoising" map $\boldsymbol{g}(\widetilde{\boldsymbol{x}}) \approx \boldsymbol{x}$ of deliberately corrupted inputs $\widetilde{\boldsymbol{x}}$. The DAE is a cornerstone for the development of deep learning or representation learning (Bengio et al., 2013a). Although the *corrupt and denoise* principle is simple, it is

successful and has inspired many representation learning algorithms (see Section 1.3.1 for example). Furthermore, we investigate *stacking* (Bengio et al., 2007) of DAEs. Because *stacked DAE* (Vincent et al., 2010) runs DAEs on the codes in the hidden layer, it has been less investigated, so far.

The key finding is that when the corruption process is additive, i.e., $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}$ with some noise $\boldsymbol{\varepsilon}$, then the DAE $\boldsymbol{g}$ is given by the sum of the traditional autoencoder $\widetilde{\boldsymbol{x}} \mapsto \widetilde{\boldsymbol{x}}$ and a certain denoising term $\widetilde{\boldsymbol{x}} \mapsto \boldsymbol{f}_t(\widetilde{\boldsymbol{x}})$ parametrized by noise variance t:

$$\boldsymbol{g}_t(\widetilde{\boldsymbol{x}}) = \widetilde{\boldsymbol{x}} + \boldsymbol{f}_t(\widetilde{\boldsymbol{x}}). \tag{1}$$

From the statistical viewpoint, this equation is reasonable because the DAE amounts to an estimation problem of the mean parameter. Obviously, (1) is a transport map because the denoising term $\boldsymbol{f}_t$ is a displacement vector from the origin $\widetilde{\boldsymbol{x}}$ and the noise variance t is the transport time. Starting from the shallow DAE, this paper develops three topics: the transport map of the deep DAE, the equivalence between the stacked DAE and the composition of DAEs, and the development of the double continuum limit, or the integral representation of the flow representation.

1.1. Contributions of This Study

In this paper, we introduce the flow representation of DNNs and develop the transport analysis of DAEs. The contributions of this paper are listed below.

- We introduced the flow representation, which can avoid the redundancy and complexity of the ordinary parametrization of DNNs.
- We specified the transport maps of shallow, deep, and infinitely deep DAEs, and provided their statistical interpretations. The shallow DAE is an estimator of the mean, and the deep DAE transports data points to decrease the Shannon entropy of the data distribution. According to analytic and numerical experiments, we showed that deep DAEs can extract much more information than shallow DAEs.
- We proved the equivalence between the stacked DAE and the composition of DAEs. Because of the peculiar construction, it is difficult to formulate and understand stacking. Nevertheless, by tracking the flow, we succeeded in formulating the stacked DAE. Consequently, we can interpret the effect of the *pre-training* as a regularization of hidden layers.
- We provided a new direction for the mathematical modeling of DNNs: the double continuum limit or the integral representation of the flow representation. We presented some examples of the double continuum limit of DAEs. In the integral representation, the shallow NNs is no longer a blackbox, and the training is principled. We consider that further investigations on the double continuum limit lead to the development of an interpretable and principled alternatives to DNNs.

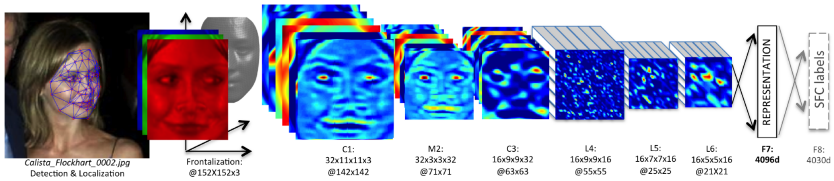


Figure 2: The activation patterns in DeepFace gradually changes (Taigman et al., 2014).

1.2. Related Work

1.2.1. Why Deep?

Before the success of deep learning, traditional theories were skeptical of the depth concept. According to approximation theory, (not only NNs but also) various shallow models can approximate any function (Pinkus, 2005). According to estimation theory, various shallow models can attain the minimax optimal ratio (Tsybakov, 2009). According to optimization theory, the depth does nothing but increase the complexity of loss surfaces unnecessarily (Boyd and Vandenberghe, 2004). In reality, of course, DNNs perform overwhelmingly better than shallow models. Thus far, the learning theory has not succeeded in explaining the gap between theory and reality (Zhang et al., 2017).

In recent years, these theories have changed drastically. For example, many authors claim that the depth increases the expressive power in the exponential order while the width does so in the polynomial order (Telgarsky, 2016; Eldan and Shamir, 2016; Cohen et al., 2016; Yarotsky, 2017), and that DNNs can attain the minimax optimal ratio in wider classes of functions (Schmidt-Hieber, 2017; Imaizumi and Fukumizu, 2019). Radical reviews of the shape of loss surfaces (Dauphin et al., 2014; Choromanska et al., 2015; Kawaguchi, 2016; Soudry and Carmon, 2016), the implicit regularization by stochastic gradient descent (Neyshabur, 2017), and the acceleration effect by over-parametrization (Nguyen and Hein, 2017; Arora et al., 2018) are ongoing. Besides the recent trends toward the rationalization of deep learning, neutral yet interesting studies have been published (Ba and Caruana, 2014; Lin et al., 2017; Poggio et al., 2017). In this study, we found that deep DAEs converge faster and that the extracted features are different from each other.

1.2.2. What Do Deep Layers Do?

Traditionally, DNNs are said to construct the hierarchy of meanings (Hinton, 1989). In convolutional NNs for image recognition, such hierarchies are empirically observed (Lee, 2010; Krizhevsky et al., 2012; Zeiler and Fergus, 2014). The hierarchy hypothesis seems to be acceptable, but it lacks explanations as to how the hierarchy is organized.

Taigman et al. (2014) reported an interesting phenomenon whereby the activation patterns in the hidden layers change by gradation from face-like patterns to codes. Inspired by Figure 2, we came up with the idea of regarding the activation pattern as a coordinate and the depth as the transport time.

1.2.3. Flow Inside Neural Networks

At the time of the initial submission in 2016, the flow representation, especially the continuum limit of the depth and collaboration with Wasserstein geometry, seemed to be a novel viewpoint of DNNs. At present, it is the mainstream of development.

Alain and Bengio (2014) was the first to derive a special case of (1), which motivated our study. Then, Alain et al. (2016) developed the generative model as a probabilistic reformulation of DAE. The generative model was a new frontier at that time; now, it is widely used in variational autoencoders (Kingma and Welling, 2014), generative adversarial nets (GANs) (Goodfellow et al., 2014), minimum probability flows (Sohl-Dickstein et al., 2015), and normalizing flows (Rezende and Mohamed, 2015). Generative models have high compatibility with transport analysis because they are formulated as Markov processes. In particular, the *generator* in GANs is exactly a transport map because it is a change-of-distribution $\boldsymbol{g} : M \to N$ from a normal distribution to a data distribution. From this viewpoint, Arjovsky et al. (2017) succeeded in stabilizing the training process of GANs by introducing Wasserstein geometry.

The *skip connection* in the residual network (ResNet) (He et al., 2016) is considered to be a key structure for training a super-deep network with more than $1,000$ layers. Formally, the skip connection is a transport map because it has an expression $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{f}(\boldsymbol{x})$. From this viewpoint, Nitanda and Suzuki (2018) reformulated the ResNet as a functional gradient and estimated the generalization error, and Lu et al. (2018) unified various ResNets as ODEs. In addition, Chizat and Bach (2018) proved the global convergence of stochastic gradient descent (SGD) using Wasserstein gradient flow. Novel deep learning methods have been proposed by controlling the flow (Ioffe and Szegedy, 2015; Gomez et al., 2017; Haber and Ruthotto, 2018; Li and Hao, 2018; Chen et al., 2018).

We remark that in shrinkage statistics, the expression of the transport map $\boldsymbol{x} + \boldsymbol{f}(\boldsymbol{x})$ is known as Brown's representation of the posterior mean (George et al., 2006). Liu and Wang (2016) analyzed it and proposed a Bayesian inference algorithm, apart from deep learning.

1.3. Background

1.3.1. Denoising Autoencoders

The *denoising autoencoder (DAE)* is a fundamental model for representation learning, the objective of which is to capture a good representation of the data. Vincent et al. (2008) introduced it as a heuristic modification of traditional autoencoders for enhancing robustness. In the setting of traditional autoencoders, we train an NN as an identity map $\boldsymbol{x} \mapsto \boldsymbol{x}$ and extract the hidden layer to obtain the so-called "code." On the other hand, the DAE is trained as a denoising map $\widetilde{\boldsymbol{x}} \mapsto \boldsymbol{x}$ of deliberately corrupted inputs $\widetilde{\boldsymbol{x}}$. Although the *corrupt and denoise* principle is simple, it has inspired many next-generation models. In this study, we analyze DAE variants such as shallow DAE, deep DAE (or composition of DAEs), infinitely deep DAE (or continuous DAE), and stacked DAE. Stacking (Bengio et al., 2007) was proposed in the early stages of deep learning, and it remains a mysterious treatment because it runs DAEs on codes in the hidden layer.

The theoretical justifications and extensions follow from at least five standpoints: manifold learning (Rifai et al., 2011; Alain and Bengio, 2014), generative modeling (Vincent et al., 2010; Bengio et al., 2013b, 2014), infomax principle (Vincent et al., 2010), learning

dynamics (Erhan et al., 2010), and score matching (Vincent, 2011). The first three standpoints were already mentioned in the original paper (Vincent et al., 2008). According to these standpoints, a DAE extracts one of the following from the data set: a manifold on which the data are arranged (manifold learning); the latent variables, which often behave as nonlinear coordinates in the feature space, that generate the data (generative modeling); a transformation of the data distribution that maximizes the mutual information (infomax); good initial parameters that allow the training to avoid local minima (learning dynamics); or the data distribution (score matching). A turning point appears to be the finding of the score matching aspect (Vincent, 2011), which reveals that score matching with a special form of the energy function coincides with a DAE. Thus, a DAE is a density estimator of the data distribution μ. In other words, it extracts and stores information as a function of μ. Since then, many researchers have avoided stacking deterministic autoencoders and have developed generative density estimators (Bengio et al., 2013b, 2014) instead.

1.3.2. Integral Representation Theory and Ridgelet Analysis

The flow representation is inspired by the integral representation theory (Murata, 1996; Candès, 1998; Sonoda and Murata, 2017a).

The integral representation

$$S[\gamma](\boldsymbol{x}) = \int \gamma(\boldsymbol{a}, b)\sigma(\boldsymbol{a}\cdot\boldsymbol{x} - b)\mathrm{d}\lambda(\boldsymbol{a}, b) \tag{2}$$

is a continuum limit of a shallow NN $g_p(\boldsymbol{x}) = \sum_{j=1}^p c_j\sigma(\boldsymbol{a}_j\cdot\boldsymbol{x}-b_j)$ as the hidden unit number $p\to\infty$. In $S[\gamma]$, every possible nonlinear parameter $(\boldsymbol{a}, b)$ is "integrated out," and only linear parameters c_j remain as a coefficient function $\gamma(\boldsymbol{a}, b)$. Therefore, we do not need to select which $(\boldsymbol{a}, b)$'s to use, which amounts to a non-convex optimization problem. Instead, the coefficient function $\gamma(\boldsymbol{a}, b)$ automatically selects the $(\boldsymbol{a}, b)$'s by weighting them. Similar reparametrization techniques have been proposed for Bayesian NNs (Radford M. Neal, 1996) and convex NNs (Bengio et al., 2006; Bach, 2017a). Once a coefficient function γ is given, we can obtain an ordinary NN g_p that approximates $S[\gamma]$ by numerical integration. We also remark that the integral representation $S[\gamma_p]$ with a singular coefficient $\gamma_p := \sum_{j=1}^p c_j\delta_{(\boldsymbol{a}_j,b_j)}$ leads to an ordinary NN g_p.

The advantage of the integral representation is that the solution operator—the *ridgelet transform*—to the integral equation $S[\gamma] = f$ and the optimization problem of $L[\gamma] := \|S[\gamma]-f\|^2+\beta\|\gamma\|^2$ is known. The ridgelet transform with an admissible function ρ is given by

$$R[f](\boldsymbol{a}, b) := \int_{\mathbb{R}^m} f(\boldsymbol{x})\overline{\rho(\boldsymbol{a}\cdot\boldsymbol{x} - b)}\mathrm{d}\boldsymbol{x}. \tag{3}$$

The integral equation $S[\gamma] = f$ is a traditional form of learning, and the ridgelet transform $\gamma = R[f]$ satisfies $S[\gamma] = S[R[f]] = f$ (Murata, 1996; Candès, 1998; Sonoda and Murata, 2017a). The optimization problem of $L[\gamma]$ is a modern form of learning, and a modified version of the ridgelet transform gives the global optimum (Sonoda et al., 2018). These studies imply that a shallow NN is *no longer a blackbox* but a ridgelet transform of the data set. Traditionally, the integral representation has been developed to estimate the

approximation and estimation error bounds of shallow NNs g_p (Barron, 1993; Kůrková, 2012; Klusowski and Barron, 2017, 2018; Suzuki, 2018). Recently, the numerical integration methods for $R[f]$ and $S[R[f]]$ were developed (Candès, 1998; Sonoda and Murata, 2014; Bach, 2017b) with various f, including the MNIST classifier. Hence, by computing the ridgelet transform of the data set, we can obtain the global minimizer without gradient descent.

Thus far, the integral representation is known as an efficient reparametrization method to facilitate understanding of the hidden layers, to estimate the approximation and estimation error bounds of shallow NNs, and to calculate the hidden parameters. However, it is based on linear algebra, i.e., it starts by regarding c_j and $\sigma(\boldsymbol{a}_j\cdot\boldsymbol{x} - b_j)$ as coefficients and basis functions, respectively. Therefore, the integral representation for DNNs is not trivial at all.

1.3.3. Optimal Transport Theory and Wasserstein Geometry

The optimal transport theory (Villani, 2009) originated from the practical requirement in the 18th century to transport materials at the minimum cost. At the end of the 20th century, it was transformed into *Wasserstein geometry,* or the geometry on the space of probability distributions. Recently, Wasserstein geometry has attracted considerable attention in statistics and machine learning. One of the reasons for its popularity is that the *Wasserstein distance* can capture the difference between two singular measures, whereas the traditional Kullback-Leibler distance cannot (Arjovsky et al., 2017). Another reason is that it gives a unified perspective on a series of function inequalities, including the concentration inequality. Computation methods for the Wasserstein distance and Wasserstein gradient flow have also been developed (Peyré and Cuturi, 2018; Nitanda and Suzuki, 2018; Zhang et al., 2018). In this study, we employ *Wasserstein gradient flow* (Ambrosio et al., 2008) for the characterization of DNNs.

Given a density μ of materials in $\mathbb{R}^m$, a density ν of final destinations in $\mathbb{R}^m$, and a cost function $c : \mathbb{R}^m\times\mathbb{R}^m\to\mathbb{R}$ associated with the transportation, under some regularity conditions, there exist some optimal transport map(s) $\boldsymbol{g} : \mathbb{R}^m\to\mathbb{R}^m$ that attain the minimum transportation cost. Let $W(\mu,\nu)$ denote the minimum cost of the transportation problem from μ to ν. Then, it behaves as the distance between two probability densities μ and ν, and it is called the *Wasserstein distance*, which is the start point of Wasserstein geometry.

When the cost function c is given by the ℓ^p-distance, i.e., $c(\boldsymbol{x},\boldsymbol{y}) = |\boldsymbol{x}-\boldsymbol{y}|_p$, the corresponding Wasserstein distance is called the L^p-Wasserstein distance $W_p(\mu,\nu)$. Let $\mathcal{P}_p(\mathbb{R}^m)$ be the space of probability densities on $\mathbb{R}^m$ that have at least the p-th moment. The distance space $\mathcal{P}_p(\mathbb{R}^m)$ equipped with L^p-Wasserstein distance W_p is called the *L^p-Wasserstein space*. Furthermore, the L^2-Wasserstein space $(\mathcal{P}_2, W_2)$ admits the *Wasserstein metric* $\mathfrak{g}_2$, which is an infinite-dimensional Riemannian metric that induces the L^2-Wasserstein distance as the geodesic distance. Owing to $\mathfrak{g}_2$, the L^2-Wasserstein space is an infinite-dimensional manifold. On $\mathcal{P}_2$, we can introduce the tangent space $T_\mu\mathcal{P}_2$ at $\mu\in\mathcal{P}_2$, and the gradient operator grad, which are fundamentals to define *Wasserstein gradient flow*. See Section 2 for more details.

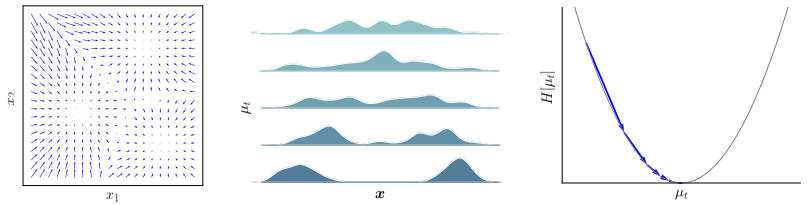


Figure 3: Three profiles of a flow analyzed in the transport analysis: dynamical system in $\mathbb{R}^m$ described by vector field (or transport map) (**left**), pushforward measure described by continuity equation in $\mathbb{R}^m$ (**center**), and Wasserstein gradient flow in $\mathcal{P}_2(\mathbb{R}^m)$ (**right**).

Organization of This Paper

In Section 2, we describe the framework of transport analysis, which combines a quick introduction to dynamical systems theory, optimal transport theory, and Wasserstein gradient flow. In Section 3 and 4, we specify the transport maps of shallow, deep, and infinitely deep DAEs, and we give their statistical interpretations. In Section 5, we present analytic examples and the results of numerical experiments. In Section 6, we prove the equivalence between the stacked DAE and the composition of DAEs. In Section 7, we develop the integral representation of the flow representation.

Remark

After the initial submission of the manuscript in 2016, the present manuscript has been substantially reorganized and updated. The authors presented the digests of some results from Section 3, 4 and 7 in two workshops (Sonoda and Murata, 2017b,c).

2. Transport Analysis of Deep Neural Networks

In the transport analysis, we regard a deep neural network as a transport map, and we track the flow in three scales: microscopic, mesoscopic, and macroscopic. Wasserstein geometry provides a unified framework for bridging these three scales. In each scale, we analyze three profiles of the flow: dynamical system, pushforward measure, and Wasserstein gradient flow.

First, on the microscopic scale, we analyze the transport map $\boldsymbol{g}_t : \mathbb{R}^m \to \mathbb{R}^m$, which simply describes the transportation of every point. In continuum mechanics, this viewpoint corresponds to the Eulerian description. The transport map $\boldsymbol{g}_t$ is often associated with a velocity field $\boldsymbol{v}_t$ that summarizes all the behavior of $\boldsymbol{g}_t$ by an ODE or the continuous dynamical system: $\partial_t \boldsymbol{g}_t(\boldsymbol{g}_t(\boldsymbol{x})) = \boldsymbol{v}_t(\boldsymbol{g}_t(\boldsymbol{x}))$. We note that, as suggested by chaos theory, it is generally difficult to track a continuous dynamics.

Second, on the mesoscopic scale, we analyze the pushforward μ_t or the time evolution of the data distribution. In continuum mechanics, this viewpoint corresponds to the Lagrangian description. When the transport map is associated with a vector field $\boldsymbol{v}_t$, then the corresponding distributions evolve according to a partial differential equation (PDE) or the

continuity equation $\partial_t \mu_t = -\nabla \cdot [\boldsymbol{v}_t \mu_t]$. We note that, as suggested by fluid dynamics, it is generally difficult to track a continuity equation.

Finally, on the macroscopic scale, we analyze the Wasserstein gradient flow or the trajectories of time evolution of μ_t in the space $\mathcal{P}(\mathbb{R}^m)$ of probability distributions on $\mathbb{R}^m$. When the transport map is associated with a vector field $\boldsymbol{v}_t$, then there exists a time-independent potential functional F on $\mathcal{P}(\mathbb{R}^m)$ such that an evolution equation or the Wasserstein gradient flow $\dot{\mu}_t = -\mathsf{grad}\, F[\mu_t]$ coincides with the continuity equation. We remark that tracking a Wasserstein gradient flow may be easier compared to the two above-mentioned cases, because the potential functional is independent of time.

2.1. Transport Map and Flow

In the broadest sense, a *transport map* is simply a measurable map $\boldsymbol{g} : M \to N$ between two probability spaces M and N (see Definition 1.2 in Villani, 2009, for example). In this study, we use the term as an update rule. Depending on the context, we distinguish the term "flow" from "transport map." While a flow is associated with a continuous dynamical system, a transport map is associated with a discrete dynamical system. We understand that a transport map arises as a discretization of a flow. An ordinary DNN coincides with a transport map, and the depth continuum limit coincides with a flow.

Definition 1 *A transport map* $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^m$ *is a measurable map given by*

$$\begin{cases} \boldsymbol{g}_t(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{f}_t(\boldsymbol{x}), & \boldsymbol{x} \in \mathbb{R}^m,\ t > 0 \\ \boldsymbol{g}_0(\boldsymbol{x}) = \boldsymbol{x}, & \boldsymbol{x} \in \mathbb{R}^m,\ t = 0, \end{cases} \tag{4}$$

with an update vector $\boldsymbol{f}_t$.

Definition 2 *A flow* $\boldsymbol{\varphi}_t$ *is given by an ordinary differential equation (ODE),*

$$\begin{cases} \dot{\boldsymbol{\varphi}}_t(\boldsymbol{x}) = \boldsymbol{v}_t(\boldsymbol{\varphi}_t(\boldsymbol{x})), & \boldsymbol{x} \in \mathbb{R}^m,\ t > 0 \\ \boldsymbol{\varphi}_0(\boldsymbol{x}) = \boldsymbol{x}, & \boldsymbol{x} \in \mathbb{R}^m,\ t = 0, \end{cases} \tag{5}$$

with a velocity field $\boldsymbol{v}_t$.

In particular, we are interested in the case when the update rule (4) is a tangent line approximation of a flow (5). i.e., $\boldsymbol{g}_t$ satisfies

$$\lim_{t \to 0} \frac{\boldsymbol{g}_t(\boldsymbol{x}) - \boldsymbol{x}}{t} = \boldsymbol{v}_0(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^m \tag{6}$$

for some $\boldsymbol{v}_t$. In this case, the velocity field $\boldsymbol{v}_t$ is the only parameter that determines the transport map.

2.2. Pushforward Measure and Continuity Equation

In association with the mass transportation $\boldsymbol{x} \mapsto \boldsymbol{g}_t(\boldsymbol{x})$, the data distribution μ_0 itself changes its shape to, say, μ_t (see Figure 4, for example). Technically speaking, μ_t is called (the density of) the *pushforward measure* of μ_0 by $\boldsymbol{g}_t$, and it is denoted by $\boldsymbol{g}_{t\sharp}\mu_0$.

Definition 3 *Let μ be a Borel measure on M and $\boldsymbol{g} : M \to N$ be a measurable map. Then, $\boldsymbol{g}_\sharp\mu$ denotes the image measure (or pushforward) of μ by $\boldsymbol{g}$. It is a measure on N, defined by $(\boldsymbol{g}_\sharp\mu)(B) = \mu \circ \boldsymbol{g}^{-1}(B)$ for every Borel set $B \subset N$.*

The pushforward μ_t is calculated by the change-of-variables formula. In particular, the following extended version by Evans and Gariepy (2015, Theorem 3.9) from geometric measure theory is useful.

Fact 1 *Let $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^n$ be Lipschitz continuous, $m \leq n$, and μ be a probability density on $\mathbb{R}^m$. Then, the pushforward $\boldsymbol{g}_\sharp\mu$ satisfies*

$$\boldsymbol{g}_\sharp\mu \circ \boldsymbol{g}(\boldsymbol{x})[\nabla \boldsymbol{g}](\boldsymbol{x}) = \mu(\boldsymbol{x}), \quad \text{a.e. } \boldsymbol{x}. \tag{7}$$

Here, the Jacobian is defined by

$$[\nabla \boldsymbol{g}] = \sqrt{\det |(\nabla \boldsymbol{g})^* \circ (\nabla \boldsymbol{g})|}. \tag{8}$$

The continuity equation describes the one-to-one relation between a flow and the pushforward.

Fact 2 *Let $\boldsymbol{\varphi}_t$ be the flow of an ODE (5) with vector field $\boldsymbol{v}_t$. Then, the pushforward μ_t of the initial distribution μ_0 evolves according to the* continuity equation

$$\partial_t \mu_t(\boldsymbol{x}) = -\nabla \cdot [\mu_t(\boldsymbol{x}) \boldsymbol{v}_t(\boldsymbol{x})], \quad \boldsymbol{x} \in \mathbb{R}^m, t \geq 0. \tag{9}$$

Here, $\nabla\cdot$ denotes the divergence operator in $\mathbb{R}^m$.

The continuity equation is also known as the *conservation of mass formula*, and this relation between the partial differential equation (PDE) (9) and the ODE (5) is a well-known fact in continuum physics (Villani, 2009, pp.19). See Appendix B for a sketch of the proof and Ambrosio et al. (2008, § 8) for more detailed discussions.

2.3. Wasserstein Gradient Flow Associated with Continuity Equation

In addition to the ODE and PDE in $\mathbb{R}^m$, we introduce the third profile: the *Wasserstein gradient flow* or the evolution equation in the space of the probability densities on $\mathbb{R}^m$. The Wasserstein gradient flow has a distinct advantage that the potential functional F of the gradient flow is independent of time t; on the other hand, the vector field $\boldsymbol{v}_t$ is usually time-dependent. Furthermore, it often facilitates the understanding of transport maps because we will see that both the Boltzmann entropy and the Renyi entropy are examples of F.

Let $\mathcal{P}_2(\mathbb{R}^m)$ be the L^2-Wasserstein space defined in Section 1.3.3, and let $\mu_t \in \mathcal{P}_2(\mathbb{R}^m)$ be the solution of the continuity equation (9) with initial distribution $\mu_0 \in \mathcal{P}_2(\mathbb{R}^m)$. Then, the map $t \mapsto \mu_t$ plots a curve in $\mathcal{P}_2(\mathbb{R}^m)$. According to the Otto calculus (Villani, 2009, § 23), this curve coincides with a functional gradient flow in $\mathcal{P}_2(\mathbb{R}^m)$, called the Wasserstein gradient flow, with respect to some *potential functional* $F : \mathcal{P}_2(\mathbb{R}^m) \to \mathbb{R}$.

Specifically, we further assume that the vector field $\boldsymbol{v}_t$ is given by the gradient vector field ∇V_t of a potential function $V_t : \mathbb{R}^m \to \mathbb{R}$.

Fact 3 *Assume that μ_t satisfies the continuity equation with the gradient vector field,*

$$\partial_t \mu_t = -\nabla \cdot [\mu_t \nabla V_t], \tag{10}$$

and that we have found F that satisfies the following equation:

$$\frac{\mathrm{d}}{\mathrm{d}t} F[\mu_t] = \int_{\mathbb{R}^m} \nabla V_t(\boldsymbol{x}) [\partial_t \mu_t](\boldsymbol{x}) \mathrm{d}\boldsymbol{x}. \tag{11}$$

Then, the Wasserstein gradient flow

$$\frac{\mathrm{d}}{\mathrm{d}t} \mu_t = -\mathsf{grad}\, F[\mu_t], \tag{12}$$

coincides with the continuous equation.

Here, grad denotes the gradient operator on L^2-Wasserstein space $\mathcal{P}_2(\mathbb{R}^m)$ explained in Section 1.3.3. While (12) is an evolution equation or an ODE in $\mathcal{P}_2(\mathbb{R}^m)$, (9) is a PDE in $\mathbb{R}^m$. Hence, we use different notations for the time derivatives, $\frac{\mathrm{d}}{\mathrm{d}t}$ and ∂_t.

3. Denoising Autoencoder

We formulate the denoising autoencoder (DAE) as a variational problem, and we show that the minimizer $\boldsymbol{g}^*$ or the training result is a transport map. Even though the term "DAE" refers to a training procedure of neural networks, we refer to the minimizer of DAE also as a "DAE." We further investigate the initial velocity vector field $\partial_t \boldsymbol{g}_{t=0}$ for mass transportation, and we show that the data distribution μ_t evolves according to the continuity equation.

For the sake of simplicity, we assume that the hidden unit number of NNs is sufficiently large (or infinite), and thus the NNs can always attain the minimum. Furthermore, we assume the the size of data set is sufficiently large (or infinite). In the case when the hidden unit number and the size of data set are both finite, we understand the DAE $\boldsymbol{g}$ is composed of the minimizer $\boldsymbol{g}^*$ and the residual term $\boldsymbol{h}$. Namely, $\boldsymbol{g} = \boldsymbol{g}^* + \boldsymbol{h}$. However, theoretical investigations on the approximation and estimation error $\boldsymbol{h}$ remain as our future work.

3.1. Training Procedure of DAE

Let $\boldsymbol{x}$ be an m-dimensional random vector that is distributed according to the data distribution μ_0, and let $\widetilde{\boldsymbol{x}}$ be its corruption defined by

$$\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \nu_t$$

where ν_t denotes the noise distribution parametrized by variance $t \geq 0$. A basic example of ν_t is the Gaussian noise with mean 0 and variance t, i.e., $\nu_t = N(0, tI)$.

The DAE is a function that is trained to remove corruption $\widetilde{\boldsymbol{x}}$ and restore it to the original $\boldsymbol{x}$; this is equivalent to finding a function $\boldsymbol{g}$ that minimizes an objective function, i.e.,

$$L[\boldsymbol{g}] := \mathbb{E}_{\boldsymbol{x},\widetilde{\boldsymbol{x}}} |\boldsymbol{g}(\widetilde{\boldsymbol{x}}) - \boldsymbol{x}|^2. \tag{13}$$

Note that as long as $\boldsymbol{g}$ is a universal approximator and can thus attain the minimum, it need not be a neural network. Specifically, our analysis in this section and the next section is applicable to a wide range of learning machines. Typical examples of $\boldsymbol{g}$ include neural networks with a sufficiently large number of hidden units, splines (Wahba, 1990), kernel machines (Shawe-Taylor and Cristianini, 2004) and ensemble models (Schapire and Freund, 2012).

3.2. Transport Map of DAE

Theorem 4 *(Modification of Theorem 1 by Alain and Bengio, 2014). The global minimum $\boldsymbol{g}_t^*$ of $L[\boldsymbol{g}]$ is attained at*

$$\boldsymbol{g}_t^*(\widetilde{\boldsymbol{x}}) = \frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} \boldsymbol{x}\nu_t(\widetilde{\boldsymbol{x}} - \boldsymbol{x})\mu_0(\boldsymbol{x})\mathrm{d}\boldsymbol{x}, \tag{14}$$

$$= \widetilde{\boldsymbol{x}} - \underbrace{\frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} \boldsymbol{\varepsilon}\nu_t(\boldsymbol{\varepsilon})\mu_0(\widetilde{\boldsymbol{x}} - \boldsymbol{\varepsilon})\mathrm{d}\boldsymbol{\varepsilon}}_{=: \boldsymbol{f}_t(\widetilde{\boldsymbol{x}})}, \tag{15}$$

where $*$ denotes the convolution operator.

Here, the second equation is simply derived by changing the variable $\boldsymbol{x} \leftarrow \widetilde{\boldsymbol{x}} - \boldsymbol{\varepsilon}$ (see Appendix A for the complete proof, where we used the calculus of variations). Note that this calculation first appeared in Alain and Bengio (2014, Theorem 1), where the authors obtained (14).

The DAE $\boldsymbol{g}_t^*(\boldsymbol{x})$ is composed of the identity term $\boldsymbol{x}$ and the denoising term $\boldsymbol{f}_t(\boldsymbol{x})$. If we assume that $\nu_t \to \delta_t$ as $t \to 0$, then in the limit $t \to 0$, the denoising term $\boldsymbol{f}_t(\boldsymbol{x})$ vanishes and DAE reduces to a traditional autoencoder. We reinterpret the DAE $\boldsymbol{g}_t^*(x)$ as a *transport map with transport time t* that transports the mass at $\boldsymbol{x} \in \mathbb{R}^m$ toward $\boldsymbol{x} + \boldsymbol{f}_t(\boldsymbol{x}) \in \mathbb{R}^m$ with displacement vector $\boldsymbol{f}_t(\boldsymbol{x})$.

3.3. Statistical Interpretation of DAE

In statistics, (15) is known as Brown's representation of the posterior mean (George et al., 2006). This is not just a coincidence, because the DAE $\boldsymbol{g}_t^*$ is an estimator of the mean. Recall that a DAE is trained to retain the original vector $\boldsymbol{x}$, given its corruption $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}$. At least in principle, this is nonsense because to retain $\boldsymbol{x}$ from $\widetilde{\boldsymbol{x}}$ means to reverse the random walk $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}$ (in Figure 4, the multimodal distributions $\mu_{0.5}$ and $\mu_{1.0}$ indicate its difficulty). Obviously, this is an inverse problem or a statistical estimation problem of the latent vector $\boldsymbol{x}$, given the noised observation $\widetilde{\boldsymbol{x}}$ with the observation model $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}$. According to a fundamental fact of estimation theory, the minimum mean squared error (MMSE) estimator of $\boldsymbol{x}$ given $\widetilde{\boldsymbol{x}}$ is given by the posterior mean $\mathbb{E}[\boldsymbol{x}|\widetilde{\boldsymbol{x}}]$. In our case, the posterior mean equals $\boldsymbol{g}_t^*$.

$$\mathbb{E}[\boldsymbol{x}|\widetilde{\boldsymbol{x}}] = \frac{\int_{\mathbb{R}^m} \boldsymbol{x}p(\widetilde{\boldsymbol{x}} \mid \boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}}{\int_{\mathbb{R}^m} p(\widetilde{\boldsymbol{x}} \mid \boldsymbol{x}')p(\boldsymbol{x}')\mathrm{d}\boldsymbol{x}'} = \frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} \boldsymbol{x}\nu_t(\widetilde{\boldsymbol{x}} - \boldsymbol{x})\mu_0(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \boldsymbol{g}_t^*(\widetilde{\boldsymbol{x}}). \tag{16}$$

Similarly, we can interpret the denoising term $\boldsymbol{f}_t(\widetilde{\boldsymbol{x}})$ as the posterior mean $\mathbb{E}[\boldsymbol{\varepsilon}|\widetilde{\boldsymbol{x}}]$ of noise $\boldsymbol{\varepsilon}$ given observation $\widetilde{\boldsymbol{x}}$.

3.4. Examples: Gaussian DAE

When the noise distribution is Gaussian with mean 0 and covariance tI, i.e.,

$$\nu_t(\boldsymbol{\varepsilon}) = \frac{1}{(2\pi t)^{m/2}} e^{-|\boldsymbol{\varepsilon}|^2/2t},$$

the transport map is calculated as follows.

Theorem 5 *The transport map $\boldsymbol{g}_t^*$ of Gaussian DAE is given by*

$$\boldsymbol{g}_t^*(\widetilde{\boldsymbol{x}}) = \widetilde{\boldsymbol{x}} + t\nabla \log[\nu_t * \mu_0](\widetilde{\boldsymbol{x}}). \tag{17}$$

Proof The proof is straightforward by using Stein's identity,

$$-t\nabla\nu_t(\boldsymbol{\varepsilon}) = \boldsymbol{\varepsilon}\,\nu_t(\boldsymbol{\varepsilon}),$$

which is known to hold only for Gaussians.

$$\begin{aligned}
\boldsymbol{g}_t^*(\widetilde{\boldsymbol{x}}) &= \widetilde{\boldsymbol{x}} - \frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} \boldsymbol{\varepsilon}\nu_t(\boldsymbol{\varepsilon})\mu_0(\widetilde{\boldsymbol{x}} - \boldsymbol{\varepsilon})\mathrm{d}\boldsymbol{\varepsilon} \\
&= \widetilde{\boldsymbol{x}} + \frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} t\nabla\nu_t(\boldsymbol{\varepsilon})\mu_0(\widetilde{\boldsymbol{x}} - \boldsymbol{\varepsilon})\mathrm{d}\boldsymbol{\varepsilon} \\
&= \widetilde{\boldsymbol{x}} + \frac{t\nabla\nu_t * \mu_0(\widetilde{\boldsymbol{x}})}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \\
&= \widetilde{\boldsymbol{x}} + t\nabla \log[\nu_t * \mu_0(\widetilde{\boldsymbol{x}})].
\end{aligned}$$

■

Theorem 6 *At the initial moment $t \to 0$, the pushforward μ_t of Gaussian DAE satisfies the* backward heat equation

$$\partial_t \mu_{t=0}(\boldsymbol{x}) = -\triangle\mu_0(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^m, \tag{18}$$

where $\triangle$ denotes the Laplacian.

Proof The initial velocity vector is given by the *Fisher score*

$$\partial_t \boldsymbol{g}_{t=0}^*(\boldsymbol{x}) = \lim_{t\to 0} \frac{\boldsymbol{g}_t^*(\boldsymbol{x}) - \boldsymbol{x}}{t} = \nabla \log \mu_0(\boldsymbol{x}). \tag{19}$$

Hence, by substituting the score (19) in the continuity equation (9), we have

$$\partial_t \mu_{t=0}(\boldsymbol{x}) = -\nabla \cdot [\mu_0(\boldsymbol{x})\nabla \log \mu_0(\boldsymbol{x})] = -\nabla \cdot [\nabla\mu_0(\boldsymbol{x})] = -\triangle\mu_0(\boldsymbol{x}).$$

■

The backward heat equation (BHE) rarely appears in nature. However, of course, the present result is not an error. As mentioned in Section 3.3, the DAE solves an estimation problem. Therefore, in the sense of the mean, the DAE behaves as time reversal. We remark that, as shown by Figure 4, a training result of a DAE with a real NN on a finite data set does not converge to a perfect time reversal of a diffusion process.

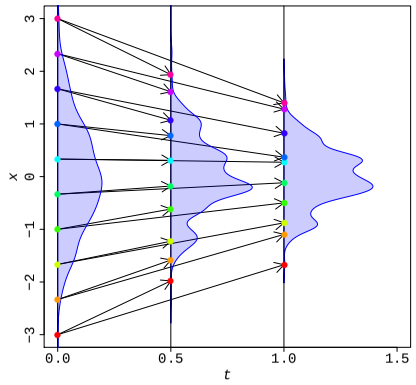


Figure 4: Shallow Gaussian DAE, which is one of the most fundamental versions of DNNs, transports mass, from the left to the right, to decrease the Shannon entropy of data. The x-axis represents the 1-dimensional input/output space, the t-axis represents the variance of the Gaussian noise, and t is the transport time. The leftmost distribution depicts the original data distribution $\mu_0 = N(0,1)$. The middle and rightmost distributions depict the pushforward $\mu_t = \boldsymbol{g}_{t\sharp}\mu_0$, associated with the transportation by two DAEs with noise variance $t = 0.5$ and $t = 1.0$, respectively. As t increases, the variance of the pushforward decreases.

4. Deep DAEs

We introduce the composition $\boldsymbol{g}_L \circ \cdots \circ \boldsymbol{g}_0$ of DAEs $\boldsymbol{g}_\ell : \mathbb{R}^m \to \mathbb{R}^m$ and its continuum limit: the continuous DAE $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$. We can understand the composition of DAEs as the *Euler scheme* or the *broken line approximation* of a continuous DAE.

For the sake of simplicity, we assume that the hidden unit number of NNs is infinite, and that the size of data set is infinite.

4.1. Composition of DAEs

We write $0 = t_0 < t_1 < \cdots < t_{L+1} = t$. We assume that the input vector $\boldsymbol{x}_0 \in \mathbb{R}^m$ is subject to a data distribution μ_0. Let $\boldsymbol{g}_0 : \mathbb{R}^m \to \mathbb{R}^m$ be a DAE that is trained on μ_0 with noise variance $t_1 - t_0$. Then, let $\boldsymbol{x}_1 := \boldsymbol{g}_0(\boldsymbol{x}_0)$, which is a random vector in $\mathbb{R}^m$ that is subject to the pushforward $\mu_1 := \boldsymbol{g}_{0\sharp}\mu_0$. We train another DAE $\boldsymbol{g}_1 : \mathbb{R}^m \to \mathbb{R}^m$ on μ_1 with noise variance $t_2 - t_1$. By repeating the procedure, we obtain $\boldsymbol{g}_\ell(\boldsymbol{x}_\ell)$ from $\boldsymbol{x}_{\ell-1}$ that is subject to $\mu_\ell := \boldsymbol{g}_{(\ell-1)\sharp}\mu_{\ell-1}$.

For the sake of generality, we assume that each component DAE is given by

$$\boldsymbol{g}_\ell(\boldsymbol{x}) = \boldsymbol{x} + (t_{\ell+1} - t_\ell)\nabla V_{t_\ell}(\boldsymbol{x}), \quad (\ell = 0, \dots, L) \tag{20}$$

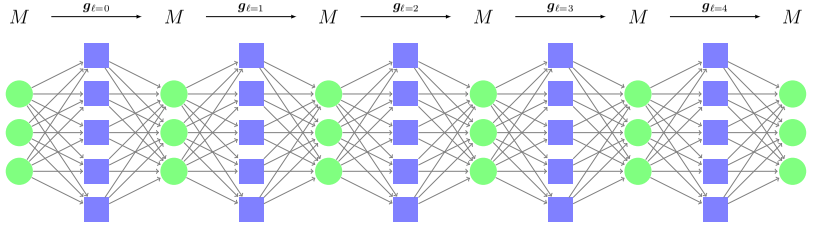


Figure 5: Composition of DAEs $\boldsymbol{g}^t_{0:4} : M \to M$, or the composite of five shallow DAEs $M \to M$, where $M = \mathbb{R}^3$

where V_{t_ℓ} denotes a certain potential function. For example, the Gaussian DAE satisfies the requirement because $V_{t_\ell} = \log[\nu_{t_\ell} * \mu_{t_\ell}]$.

We abbreviate the composition of DAEs by

$$\boldsymbol{g}^t_{0:L}(\boldsymbol{x}) := \boldsymbol{g}_L \circ \cdots \circ \boldsymbol{g}_0(\boldsymbol{x}). \tag{21}$$

By definition, the "velocity" of a composition of DAEs coincides with the vector field

$$\frac{\boldsymbol{g}^{t_{\ell+1}}_{0:\ell}(\boldsymbol{x}) - \boldsymbol{g}^{t_\ell}_{0:(\ell-1)}(\boldsymbol{x})}{t_{\ell+1} - t_\ell} = \nabla V_{t_\ell}(\boldsymbol{x}). \tag{22}$$

4.2. Continuous DAE

We fix the total time t, take the limit $L \to \infty$ of the layer number L, and introduce the continuous DAE as the limit of the "infinite composition of DAEs" $\lim_{L\to\infty} \boldsymbol{g}^t_{0:L}$.

Definition 4 *We call the solution operator or flow $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$ of the following dynamical systems as the* continuous DAE *associated with vector field ∇V_t.*

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) = \nabla V_t(\boldsymbol{x}(t)), \quad t \geq 0. \tag{23}$$

Proof According to the Cauchy-Lipschitz theorem or the Picard-Lindelöf theorem, when the vector field ∇V_t is continuous in t and Lipschitz in $\boldsymbol{x}$, the limit $\lim_{L\to\infty} \boldsymbol{g}_{0:L}$ converges to a continuous DAE (23) because the trajectory $t \mapsto \boldsymbol{g}_{0:L}(x_0)$ corresponds to a broken line approximation of the integral curve $t \mapsto \boldsymbol{\varphi}_t(x)$. ■

The following properties are immediate from Fact 2 and Fact 3. Let $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$ be the continuous DAE associated with vector field ∇V_t. Given the data distribution μ_0, the pushforward $\mu_t := (\boldsymbol{\varphi}_t)_\sharp\mu_0$ evolves according to the continuity equation

$$\partial_t \mu_t(\boldsymbol{x}) = -\nabla \cdot [\mu_t(\boldsymbol{x})\nabla V_t(\boldsymbol{x})], \quad t \geq 0 \tag{24}$$

and the Wasserstein gradient flow

$$\frac{\mathrm{d}}{\mathrm{d}t}\mu_t = -\mathsf{grad}\, F[\mu_t], \quad t \geq 0 \tag{25}$$

where F is given by (11).

4.3. Example: Gaussian DAE

We consider a continuous Gaussian DAE $\boldsymbol{\varphi}_t$ trained on $\mu_0 \in \mathcal{P}_2(\mathbb{R}^m)$. Specifically, it satisfies

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) = \nabla \log[\mu_t(\boldsymbol{x}(t))], \quad t \geq 0 \tag{26}$$

with $\mu_t := \boldsymbol{\varphi}_{t\sharp}\mu_0$.

Theorem 7 *The pushforward $\mu_t := \boldsymbol{\varphi}_{t\sharp}\mu_0$ of the continuous Gaussian DAE $\boldsymbol{\varphi}_t$ is the solution to the initial value problem of the backward heat equation (BHE)*

$$\partial_t \mu_t(\boldsymbol{x}) = -\triangle \mu_t(\boldsymbol{x}), \quad \mu_{t=0}(\boldsymbol{x}) = \mu_0(\boldsymbol{x}). \tag{27}$$

The proof is immediate from Theorem 6.

As mentioned after Theorem 6, the BHE appears because the DAE solves an estimation problem. We remark that the BHE is equivalent to the following *final value problem* for the ordinary heat equation:

$$\partial_t u_t(\boldsymbol{x}) = \triangle u_t(\boldsymbol{x}), \quad u_{t=T}(\boldsymbol{x}) = \mu_0(\boldsymbol{x}) \quad \text{for some } T$$

where u_t denotes a probability measure on $\mathbb{R}^m$. Indeed, $\mu_t(\boldsymbol{x}) = u_{T-t}(\boldsymbol{x})$ solves (27). In other words, the backward heat equation describes the time reversal of an ordinary diffusion process.

According to Wasserstein geometry, an ordinary heat equation corresponds to a Wasserstein gradient flow that *increases* the Shannon entropy functional $H[\mu] := -\int \mu(\boldsymbol{x}) \log \mu(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}$ (Villani, 2009, Th. 23.19). Consequently, we can conclude that the continuous Gaussian DAE is a transport map that *decreases* the Shannon entropy of the data distribution.

Theorem 8 *The pushforward $\mu_t := \boldsymbol{\varphi}_{t\sharp}\mu_0$ evolves according to the Wasserstein gradient flow with respect to the Shannon entropy*

$$\frac{\mathrm{d}}{\mathrm{d}t}\mu_t = -\mathsf{grad}\, H[\mu_t], \quad \mu_{t=0} = \mu_0. \tag{28}$$

Proof When $F = H$, then $V_t = -\log \mu_t$; thus,

$$\mathsf{grad}\, H[\mu_t] = \nabla \cdot [\mu_t \nabla \log \mu_t] = \nabla \cdot [\nabla \mu_t] = \triangle \mu_t,$$

which means that the continuity equation reduces to the backward heat equation. ■

4.4. Example: Renyi Entropy

Similarly, when F is the Renyi entropy

$$H^\alpha[\mu] := \int_{\mathbb{R}^m} \frac{\mu^\alpha(\boldsymbol{x}) - \mu(\boldsymbol{x})}{\alpha - 1} \mathrm{d}\boldsymbol{x},$$

then $\mathsf{grad}\, H^\alpha[\mu_t] = \triangle \mu_t^\alpha$ (see Ex. 15.6 in Villani, 2009, for the proof) and thus the continuity equation reduces to the *backward porous medium equation*

$$\partial_t \mu_t(\boldsymbol{x}) = -\triangle \mu_t^\alpha(\boldsymbol{x}). \tag{29}$$

5. Further Investigations on Shallow and Deep DAEs through Examples

5.1. Analytic Examples

We list analytic examples of shallow and continuous DAEs (see Appendix D for further details, including proofs). In all the settings, the continuous DAEs attain a singular measure at some finite $t > 0$ with various singular supports that reflect the initial data distribution μ_0, while the shallow DAEs accept any $t > 0$ and degenerate to a point mass as $t \to \infty$.

5.1.1. Univariate Normal Distribution

When the data distribution is a univariate normal distribution $N(m_0, \sigma_0)$, the transport map and pushforward for the *shallow* DAE are given by

$$g_t(x) = \frac{\sigma_0^2}{\sigma_0^2 + t} x + \frac{t}{\sigma_0^2 + t} m_0, \tag{30}$$

$$\mu_t = N\left(m_0, \frac{\sigma_0^2}{(1 + t/\sigma_0^2)^2}\right), \tag{31}$$

and those of the *continuous* DAE are given by

$$g_t(x) = \sqrt{1 - 2t/\sigma_0^2}(x - m_0) + m_0, \tag{32}$$

$$\mu_t = N(m_0, \sigma_0^2 - 2t). \tag{33}$$

5.1.2. Multivariate Normal Distribution

When the data distribution is a multivariate normal distribution $N(\boldsymbol{m}_0, \Sigma_0)$, the transport map and pushforward for the *shallow* DAE are given by

$$\boldsymbol{g}_t(\boldsymbol{x}) = (I + t\Sigma_0^{-1})^{-1}\boldsymbol{x} + (I + t^{-1}\Sigma_0)^{-1}\boldsymbol{m}_0, \tag{34}$$

$$\mu_t = N(\boldsymbol{m}_0, \Sigma_0(I + t\Sigma_0^{-1})^{-2}), \tag{35}$$

and those of the *continuous* DAE are given by

$$\boldsymbol{g}_t(\boldsymbol{x}) = \sqrt{I - 2t\Sigma_0^{-1}}(\boldsymbol{x} - \boldsymbol{m}_0) + \boldsymbol{m}_0, \tag{36}$$

$$\mu_t = N(\boldsymbol{m}_0, \Sigma_0 - 2tI). \tag{37}$$

5.1.3. Mixture of Multivariate Normal Distributions

When the data distribution is a mixture of multivariate normal distributions $\sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k)$ with the assumption that it is *well separated*, the transport map and pushforward for the *shallow* DAE are given by

$$\boldsymbol{g}_t(\boldsymbol{x}) = \sum_{k=1}^K \gamma_{kt}(\boldsymbol{x}) \left\{ (I + t\Sigma_k^{-1})^{-1}\boldsymbol{x} + (I + t^{-1}\Sigma_k)^{-1}\boldsymbol{m}_k \right\}, \tag{38}$$

$$\mu_t \approx \sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k(I + t\Sigma_k^{-1})^{-2}), \tag{39}$$

with responsibility function

$$\gamma_{kt}(\boldsymbol{x}) := \frac{w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k + tI)}{\sum_{k=1}^K w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k + tI)}, \tag{40}$$

and those of the *continuous* DAE are given by

$$\boldsymbol{g}_t(\boldsymbol{x}) \approx \sqrt{I - 2t\Sigma_k^{-1}}(\boldsymbol{x} - \boldsymbol{m}_k) + \boldsymbol{m}_k, \tag{41}$$

$$\mu_t = \sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k - 2tI), \tag{42}$$

with responsibility function

$$\gamma_{kt}(\boldsymbol{x}) := \frac{w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k - 2tI)}{\sum_{k=1}^K w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k - 2tI)}. \tag{43}$$

Here, we say that the mixture $\sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k)$ is well separated when for every cluster center $\boldsymbol{m}_k$, there exists a neighborhood Ω_k of $\boldsymbol{m}_k$ such that $N(\Omega_k; \boldsymbol{m}_k, \Sigma_k) \approx 1$ and $\gamma_{kt} \approx \mathbf{1}_{\Omega_k}$.

5.2. Numerical Example of Trajectories

We employed 2-dimensional examples, in order to visualize the difference of vector fields between the shallow and deep DAEs. In the examples below, every trajectories are drawn into attractors, however the shape of the attractors and the speed of trajectories are significantly different between shallow and deep.

5.2.1. Bivariate Normal Distribution

Figure 6 compares the trajectories of four DAEs trained on the common data distribution

$$\mu_0 = N\left([0,0], \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}\right). \tag{44}$$

The transport maps for computing the trajectories are given by (34) for the shallow DAE and composition of DAEs, and by (36) for the continuous DAE. Here, we applied (34) multiple times for the composition of DAEs.

The continuous DAE converges to an attractor lying on the x-axis at $t = 1/2$. By contrast, the shallow DAE slows down as $t \to \infty$ and never attains the singularity in finite time. As L tends to infinity, $\boldsymbol{g}_{0:L}^t$ plots a trajectory similar to that of the continuous DAE $\boldsymbol{\varphi}_t$; the curvature of the trajectory changes according to Δt.

5.2.2. Mixture of Bivariate Normal Distributions

Figure 7, 8, and 9 compare the trajectories of four DAEs trained on the three common data distributions

$$\mu_0 = 0.5\, N\left([-1,0], \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) + 0.5\, N\left([1,0], \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right), \tag{45}$$

$$\mu_0 = 0.2\, N\left([-1,0], \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) + 0.8\, N\left([1,0], \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right), \tag{46}$$

$$\mu_0 = 0.2\, N\left([-1,0], \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) + 0.8\, N\left([1,0], \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}\right). \tag{47}$$

respectively.

The transport maps for computing the trajectories are given by (38) for the shallow DAE and composition of DAEs. For the continuous DAE, we compute the trajectories by numerically solving the definition of the continuous Gaussian DAE: $\dot{\boldsymbol{x}} = \nabla \log \mu_t(\boldsymbol{x})$.

In any case, the continuous DAE converges to an attractor at some $t > 0$, but the shape of the attractors and the basins of attraction change according to the initial data distribution. The shallow DAE converges to the origin as $t \to \infty$, and the composition of DAEs plots a curve similar to that of the continuous DAE as L tends to infinity, $\boldsymbol{g}_{0:L}^t$. In particular, in Figure 8, some trajectories of the continuous DAE intersect, which implies that the velocity vector field $\boldsymbol{v}_t$ is time-dependent.

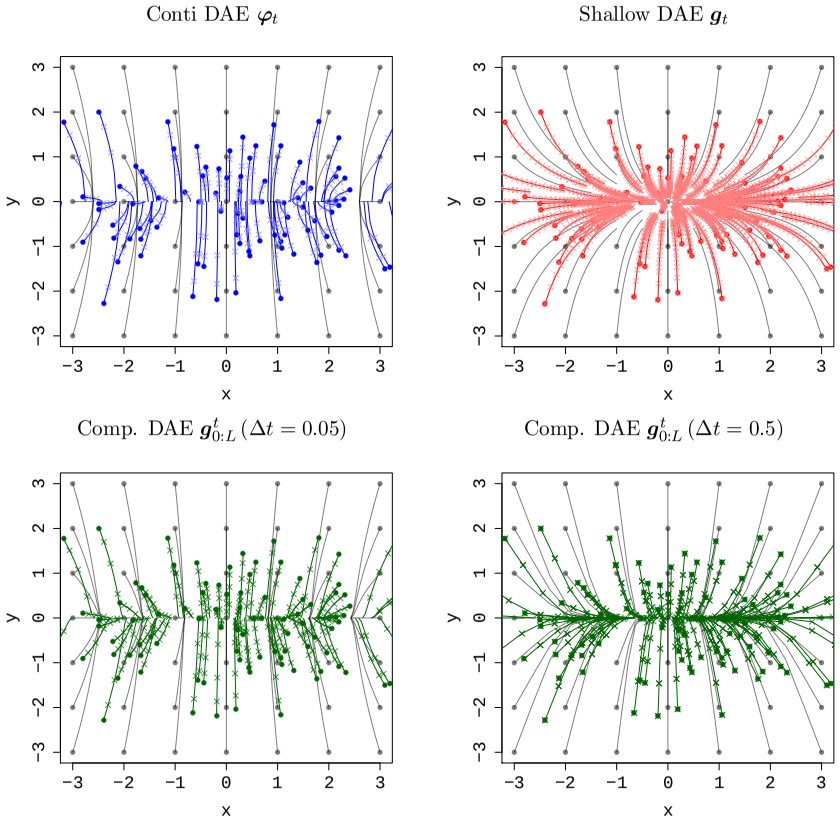


Figure 6: Trajectories of DAEs trained on the common data distribution (44) ($\mu_0 = N([0,0], \mathsf{diag}\,[2,1])$). The **gray lines** start from the regular grid. The **colored lines** start from the samples drawn from μ_0. The **midpoints** are plotted every $\Delta t = 0.2$. Every lines are drawn into attractors.

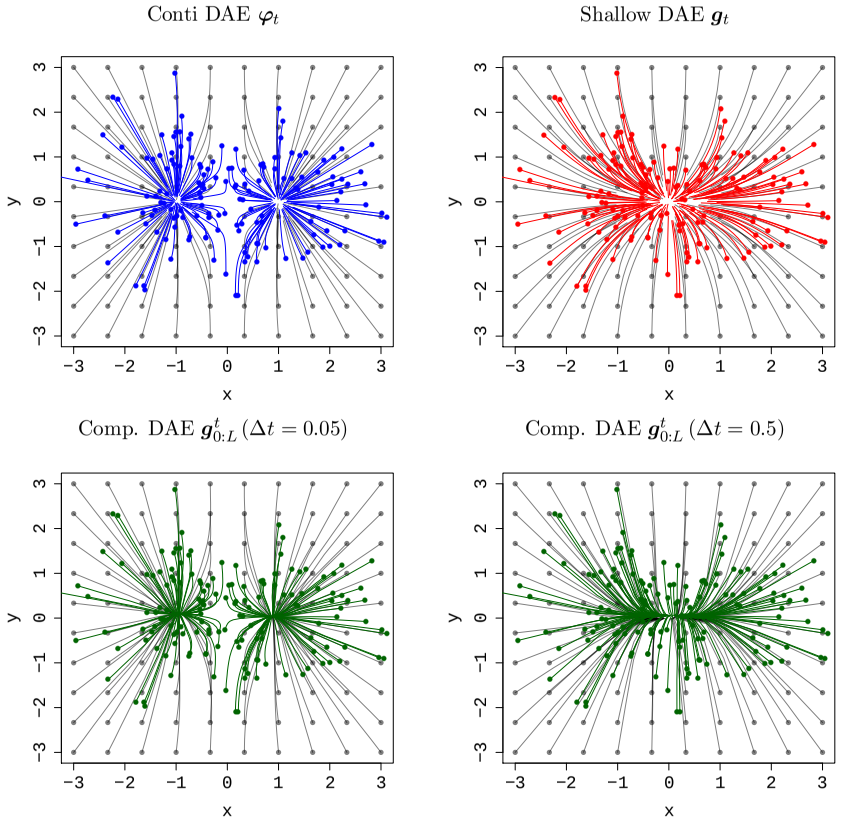


Figure 7: Trajectories of DAEs trained on the common data distribution (45) (a GMM with uniform weight and covariance). The **gray lines** start from the regular grid. The **colored lines** start from the samples drawn from μ_0. Every lines are drawn into attractors.

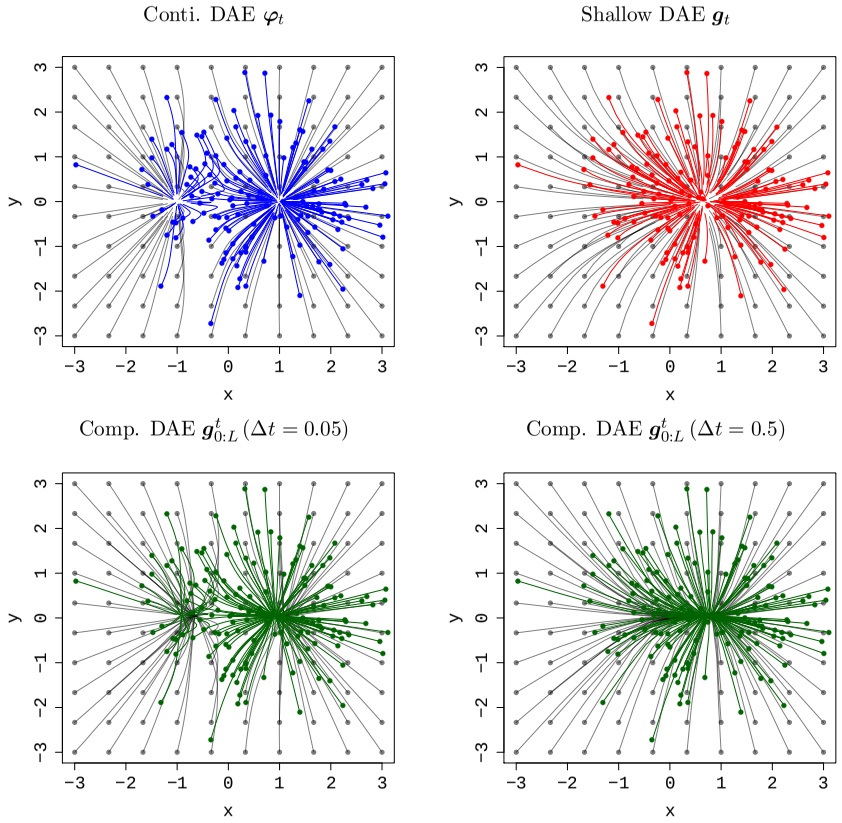


Figure 8: Trajectories of DAEs trained on the common data distribution (46) (a GMM with non-uniform weight and uniform covariance). The **gray lines** start from the regular grid. The **colored lines** start from the samples drawn from μ_0. Every lines are drawn into attractors.

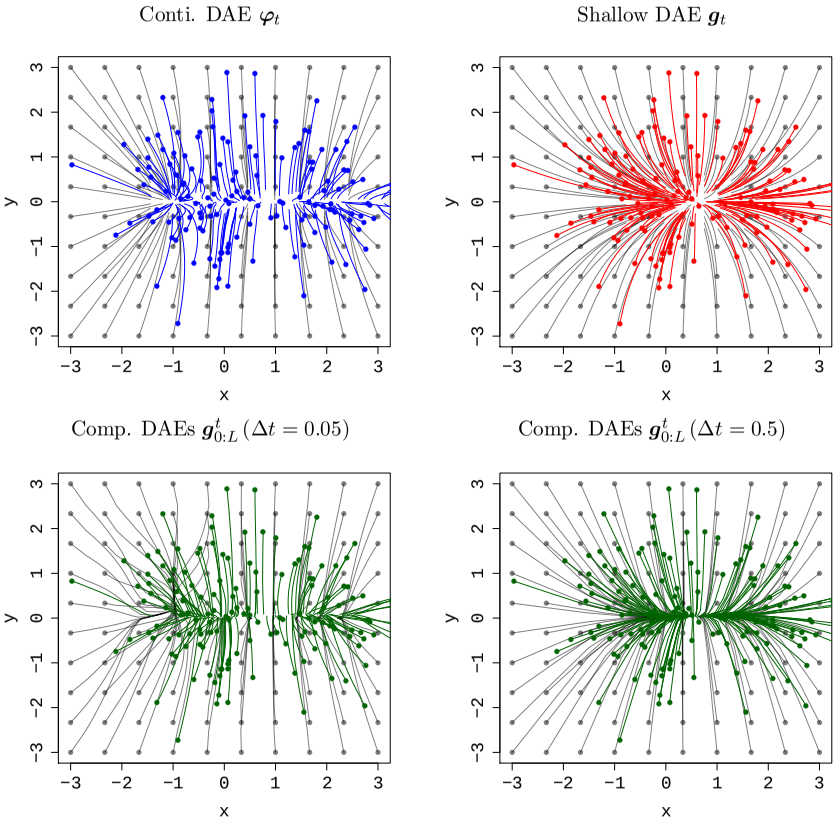


Figure 9: Trajectories of DAEs trained on the common data distribution (47) (a GMM with non-uniform weight and covariance). The **gray lines** start from the regular grid. The **colored lines** start from the samples drawn from μ_0. Every lines are drawn into attractors.

5.3. Numerical Example of Trajectories in Wasserstein Space

We consider the space $\mathcal{Q}$ of bivariate Gaussians:

$$\mathcal{Q} := \left\{ N\left([0,0], \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}\right) \,\middle|\, \sigma_1, \sigma_2 > 0 \right\}. \tag{48}$$

Obviously, $\mathcal{Q}$ is a 2-dimensional subspace of L^2-Wasserstein space, and it is closed in the actions of the continuous DAE and shallow DAE because the pushforwards are given by (37) and (35), respectively.

We employ (σ_1, σ_2) as the coordinate of $\mathcal{Q}$. This is reasonable because, in this coordinate, the L^2-Wasserstein distance $W_2(\mu, \nu)$ between two points $\mu = (\sigma_1, \sigma_2)$ and $\nu = (\tau_1, \tau_2)$ is simply given by the "Euclidean distance" $W_2(\mu, \nu) = \sqrt{(\sigma_1 - \tau_1)^2 + (\sigma_2 - \tau_2)^2}$ (see Takatsu, 2011, for the proof). The Shannon entropy is given by

$$H(\sigma_1, \sigma_2) = (1/2) \log |\mathsf{diag}\,[\sigma_1^2, \sigma_2^2]| + const. = \log \sigma_1 + \log \sigma_2 + const. \tag{49}$$

Figure 10 compares the trajectories of the pushforward by DAEs in $\mathcal{Q}$. In the left, we calculated the theoretical trajectories according to the analytic formulas (37) and (35). In the right, we trained real NNs as the composition of DAEs according to the training procedure described in Section 4.1. Even though we always assumed the infinite number of hidden units and the infinite size of data set, the results suggest that our calculus is a good approximation to finite settings.

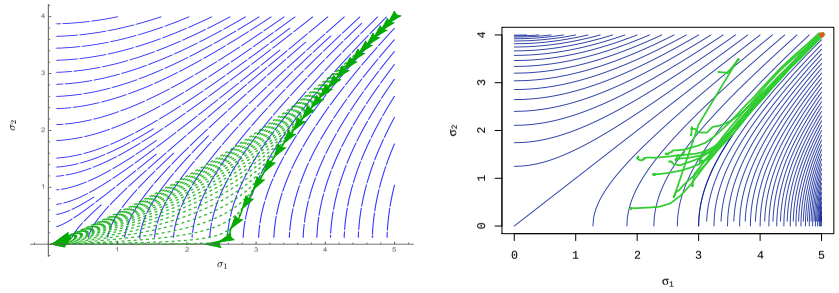


Figure 10: Trajectories of pushforward measures in a space $\mathcal{Q}$ of bivariate Gaussians $N([0,0], \mathsf{diag}\,[\sigma_1^2, \sigma_2^2])$. In **both** sides, the **blue** lines represent the Wasserstein gradient flow with respect to the Shannon entropy. The continuous Gaussian DAE $t \mapsto \boldsymbol{\varphi}_{t\sharp}\mu_0$ always coincides with the blue lines. In the **left-hand side**, the **dashed green** lines represent theoretical trajectories of the shallow DAE $t \mapsto \boldsymbol{g}_{t\sharp}\mu_0$ and the **solid green** line represents a theoretical trajectory of the composition of DAEs $t \mapsto \boldsymbol{g}^t_{0:L\sharp}\mu_0$. Both the green lines gradually leave the gradient flow. In the **right-hand side**, the **solid green** lines represent the trajectories of the composition of DAEs calculated by training real NNs (10 trials). In particular, in the early stage, the trajectories are parallel to the gradient flow.

6. Equivalence between Stacked DAE and Compositions of DAEs

As an application of transport analysis, we shed light on the equivalence of the stacked DAE (SDAE) and the composition of DAEs (CDAE), provided that the definition of DAEs is generalized to *L-DAE*, which is defined below. In SDAE, we apply the DAE to the features vectors obtained from the hidden layer of an NN to obtain higher-order feature vectors. Therefore, the feature vectors obtained from the SDAE and CDAE are different from each other. Nevertheless, we can prove that the trajectories generated by the SDAE and CDAE are *topologically conjugate*, which means that there exists a homeomorphism between the trajectories. Moreover, we can transform the trajectory of an SDAE into that of a CDAE by using a *linear map*, which is obtained from the decoder of the SDAE. Thus, we can synthesize the feature vectors of the SDAE by using CDAEs.

6.1. Definitions

To begin with, we introduce a generalized version of shallow DAE.

Definition 5 (L-DAE) *Let L be an elliptic operator on the domain Ω in $\mathbb{R}^m$, μ be a probability density on Ω, and D be a positive definite matrix. The L-DAE with diffusion coefficient D and initial data μ is defined by*

$$\mathsf{id} + tD\nabla \log e^{tL}\mu, \quad t > 0. \tag{50}$$

Here, e^{tL} is the semigroup generated by the elliptic operator L. Specifically, let $\mu_t := e^{tL}\mu$; then, μ_t satisfies the parabolic equation $\partial_t \mu_t = L\mu_t$. The original Gaussian DAE corresponds to a special case when $D \equiv I$ and $L = \triangle$.

By dae, we denote a DAE realized by a shallow NN (Figure 11). Specifically,

$$\mathsf{dae}(\boldsymbol{x}) = \sum_{j=1}^{p} \boldsymbol{c}_j \sigma(\boldsymbol{a}_j \cdot \boldsymbol{x} - b_j). \tag{51}$$

By enc and dec, we denote the encoder and decoder of dae, respectively. Specifically,

$$\mathsf{enc}_j(\boldsymbol{x}) = \sigma(\boldsymbol{a}_j \cdot \boldsymbol{x} - b_j), \quad j = 1, \ldots, p \tag{52}$$

$$\mathsf{dec}(\boldsymbol{z}) = \sum_{j=1}^{p} \boldsymbol{c}_j z_j, \tag{53}$$



Figure 11: enc and dec correspond to the hidden layer and output layer, respectively.

where z_j denotes the j-th element of $\boldsymbol{z} = \mathsf{enc}(\boldsymbol{x})$. Obviously, $\mathsf{dae} = \mathsf{dec} \circ \mathsf{enc}$.

For the sake of simipicity, even though we introduced the finite number p of hidden units, we assume that p is large, and thus dae approximately equals L-DAE for some L.

6.2. Training Procedure of Stacked DAE (SDAE)

Let $M := \mathbb{R}^m$ be the space of input vectors with probability density μ, and let $\mathsf{dae} : M \to M$ be a shallow NN with p hidden units. We assume that dae is trained as the Gaussian DAE with μ, and it thus approximates the DAE $\mathsf{id} + t\nabla \log[e^{t\triangle}\mu]$. Let $H := \mathbb{R}^p$. Then, the encoder and decoder of dae are the maps $\mathsf{enc} : M \to H$ and $\mathsf{dec} : H \to M$, respectively.

In the SDAE, we apply the DAE to $\boldsymbol{z}$. Specifically, let $\widetilde{\mu}$ be the density of hidden feature vectors $\boldsymbol{z} = \mathsf{enc}(\boldsymbol{x})$, and let $\widetilde{\mathsf{dae}} : H \to H$ be a shallow NN with $\widetilde{p}$ hidden units,

$$\widetilde{\mathsf{dae}}(\boldsymbol{z}) := \sum_{\widetilde{\jmath}=1}^{\widetilde{p}} \widetilde{\boldsymbol{c}}_{\widetilde{\jmath}} \sigma(\widetilde{\boldsymbol{a}}_{\widetilde{\jmath}} \cdot \boldsymbol{z} - \widetilde{b}_{\widetilde{\jmath}}).$$

We train $\widetilde{\mathsf{dae}}$ by using the Gaussian DAE with $\widetilde{\mu}$, where the network is decomposed as $\widetilde{\mathsf{dae}} = \widetilde{\mathsf{dec}} \circ \widetilde{\mathsf{enc}}$ with $\widetilde{\mathsf{enc}} : H \to \widetilde{H}$ and $\widetilde{\mathsf{dec}} : \widetilde{H} \to H$, and we obtain the feature vectors $\widetilde{\boldsymbol{z}} := \widetilde{\mathsf{enc}}(\boldsymbol{z}) \in \widetilde{H} = \mathbb{R}^{\widetilde{p}}$. By iterating the stacking procedure, we can obtain more abstract feature vectors (Figure 12).

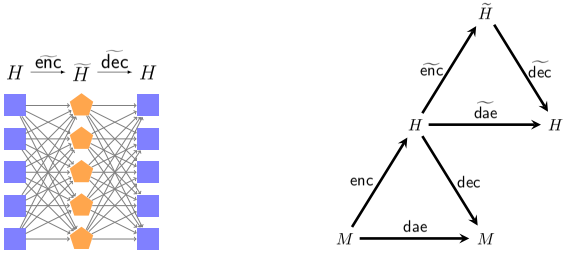


Figure 12: The (feature map of) SDAE $\widetilde{\mathsf{enc}} \circ \mathsf{enc}$ is built on the hidden layer.

Technically speaking, $\widetilde{\mu}$ is (the density of) the pushforward $\mathsf{dae}_\sharp\mu$, and its support is contained in the image $\widetilde{M} := \mathsf{enc}(M)$. In general, we assume that $\dim \widetilde{M} (= \dim M) \leq \dim H$; thus, the support of $\widetilde{\mu}$ is singular (i.e., the density vanishes outside $\widetilde{M}$) (see Fact 1 for further details).

6.3. Topological Conjugacy

The transport map of the feature vector $\widetilde{\mathsf{enc}} \circ \mathsf{enc} : M \to H \to \widetilde{H}$ is somewhat unclear. According to Theorem 9 and 10, the transport map of $\widetilde{\mathsf{enc}} \circ \mathsf{enc}$ can be transformed or projected to the ground space M by applying $\mathsf{dec} \circ \widetilde{\mathsf{dec}}$ (Figure 13). Specifically, there exists an L-DAE $\mathsf{dae}' : M \to M$ such that

$$\mathsf{dec} \circ \widetilde{\mathsf{dec}} \circ \widetilde{\mathsf{enc}} \circ \mathsf{enc} = \mathsf{dae}' \circ \mathsf{dae}. \tag{54}$$

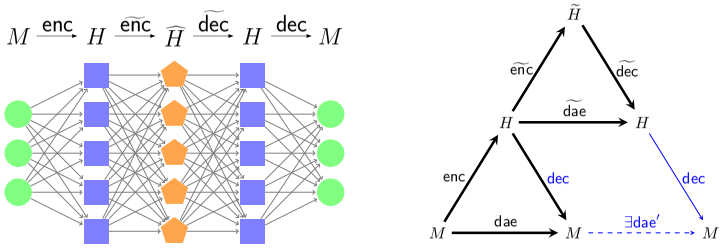


Figure 13: By reusing dec, we can transform the SDAE $\widetilde{\mathsf{enc}} \circ \mathsf{enc}$ into a CDAE $\mathsf{dae}' \circ \mathsf{dae}$.

Theorem 9 *Let H and $\widetilde{H}$ be vector spaces, $\dim H \geq \dim \widetilde{H}$, let M_0 be an m-dimensional smooth Riemannian manifold embedded in H, and let μ_0 be a C^2 probability density on M_0. Let $\boldsymbol{f} : H \to H$ be an L_t-DAE:*

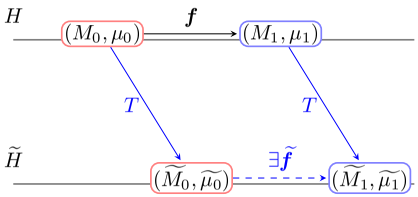
$$\boldsymbol{f} := \mathsf{id}_H + tD\nabla \log e^{tL_t}\mu_0,$$

with diffusion coefficient D and time-dependent elliptic operator L_t on H, where ∇ is the gradient operator in H.

Let $T : H \to \widetilde{H}$ be a linear map. If $T|_M$ is injective, then there exists an $\widetilde{L}_t$-DAE $\widetilde{\boldsymbol{f}} : \widetilde{H} \to \widetilde{H}$ with diffusion coefficient $\widetilde{D}$ such that

$$T \circ \boldsymbol{f}|_M = \widetilde{\boldsymbol{f}} \circ T|_M. \tag{55}$$

In other words, the following diagram commutes. Here we denoted $M_1 := \boldsymbol{f}(M_0)$ and



$\mu_1 := \boldsymbol{f}_\sharp\mu_0$. See Appendix C for the proof. The statement is general in that the choice of a linear map T is independent of the DAEs, as long as it is injective.

We note that the trajectory of the equivalent DAE $\widetilde{\boldsymbol{f}}$ may be complicated, because the "equivalence" we mean here is simply the topological conjugacy. Actually, as the proof suggests, $\widetilde{D}$ and $\widetilde{L}_t$ contain the non-linearity of activation functions via the pseudo-inverse $T^\dagger$ of T. Nevertheless, $\widetilde{\boldsymbol{f}}$ may not be much complicated because it is simply a linear projection of the high-dimensional trajectory of L_t-DAE. According to Theorem 6, a Gaussian DAE solves backward heat equation (at least when $t \to 0$). Hence, its projection to low dimension should also solve backward heat equation in low dimension spaces.

6.4. Equivalence between SDAE and CDAE

To clarify the statement, we prepare the notation. Figure 14 summarizes the symbols and procedures.

First, we rewrite the input vector as $\boldsymbol{z}^0$ instead of $\boldsymbol{x}$, the input space as $H^0 = M_0^0 (= \mathbb{R}^m)$ instead of M, and the density as μ_0^0 instead of μ. We iteratively train the ℓ-th NN $\mathsf{dae}_\ell^\ell : H^\ell \to H^\ell$ with a data distribution μ_ℓ^ℓ, obtain the encoder $\mathsf{enc}^\ell : H^\ell \to H^{\ell+1}$ and decoder $\mathsf{dec}^\ell : H^{\ell+1} \to H^\ell$, and update the feature $\boldsymbol{z}^{\ell+1} := \mathsf{enc}^\ell(\boldsymbol{z}^\ell)$, the image $M_{\ell+1}^{\ell+1} := \mathsf{enc}^\ell(M_\ell^\ell) \subset H^{\ell+1}$, and the distribution $\mu_{\ell+1}^{\ell+1} := (\mathsf{enc}^\ell)_\sharp \mu_\mu^\ell$.

For simplicity, we abbreviate

$$\mathsf{enc}^{\ell:n} := \mathsf{enc}^n \circ \cdots \circ \mathsf{enc}^\ell,$$
$$\mathsf{dec}^{n:\ell} := \mathsf{dec}^\ell \circ \cdots \circ \mathsf{dec}^n.$$

In addition, we introduce auxiliary objects.

$$M_{\ell+1}^n := \mathsf{dec}^{\ell:n}(M_{\ell+1}^{\ell+1}), \quad n = 0, \cdots, \ell$$
$$\mu_{\ell+1}^n := \mathsf{dec}_\sharp^{\ell:n} \mu_{\ell+1}^{\ell+1}, \quad n = 0, \cdots, \ell.$$

By construction, M_n^ℓ is an at most m-dimensional submanifold in H^ℓ, and the support of μ_n^ℓ is in M_n^ℓ.

Finally, we denote the map $\mathsf{dae}_n^\ell : M_n^\ell \to M_{n+1}^\ell$ that is (not "trained by DAE" but) defined by

$$\mathsf{dae}_n^\ell := (\mathsf{dec}^{n:\ell} \circ \mathsf{enc}^{0:n}) \circ (\mathsf{dec}^{(n-1):\ell} \circ \mathsf{enc}^{0:(n-1)})^{-1} : M_n^\ell \to M_{n+1}^\ell.$$

By Theorem 9, if $\mathsf{dae}_n^{\ell+1}$ is an $L_n^{\ell+1}$-DAE, then dae_n^ℓ exists and it is an L_n^ℓ-DAE.

Theorem 10 *If every* $\mathsf{enc}^\ell|_{M_\ell^\ell}$ *is a continuous injection and every* $\mathsf{dec}^\ell|_{M_n^{\ell+1}}$ *is an injection, then*

$$\mathsf{dec}^{L:0} \circ \mathsf{enc}^{0:L} = \mathsf{dae}_L^0 \circ \cdots \circ \mathsf{dae}_0^0. \tag{56}$$

Proof By repeatedly applying the topological conjugacy in Theorem 9,

$$\mathsf{dec}^\ell \circ \mathsf{dae}_n^{\ell+1} = \mathsf{dae}_n^\ell \circ \mathsf{dec}^\ell,$$

we have

$$\begin{aligned}
&\mathsf{dec}^{L:0} \circ \mathsf{enc}^{0:L} \\
&= \mathsf{dec}^{(L-2):0} \circ \mathsf{dec}^{L-1} \circ \mathsf{dae}_L^L \circ \mathsf{enc}^{L-1} \circ \mathsf{enc}^{0:(L-2)} \\
&= \mathsf{dec}^{(L-2):0} \circ \mathsf{dae}_L^{L-1} \circ \mathsf{dec}^{L-1} \circ \mathsf{enc}^{L-1} \circ \mathsf{enc}^{0:(L-2)} \\
&= \mathsf{dec}^{(L-2):0} \circ \mathsf{dae}_L^{L-1} \circ \mathsf{dae}_{L-1}^{L-1} \circ \mathsf{enc}^{0:(L-2)} \\
&\cdots \\
&= \mathsf{dae}_L^0 \circ \mathsf{dae}_{L-1}^0 \circ \cdots \circ \mathsf{dae}_0^0.
\end{aligned}$$

■

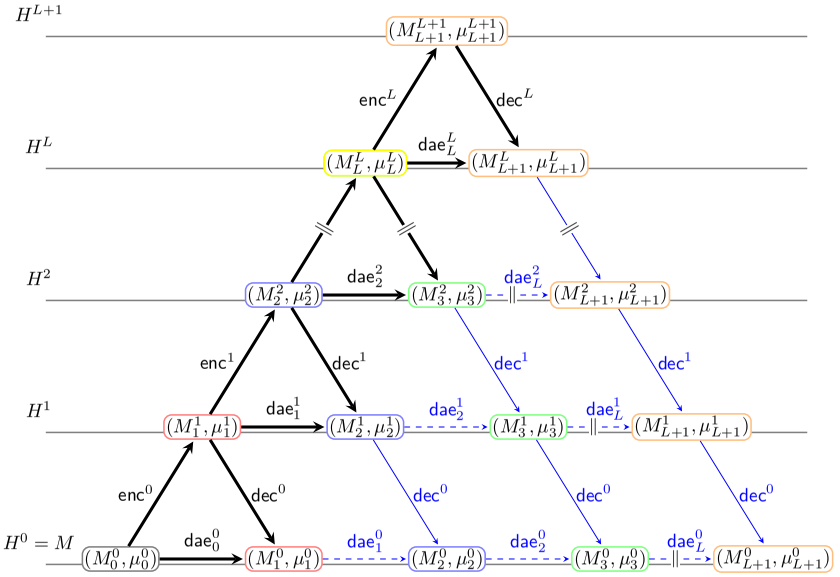


Figure 14: By using decoders, an SDAE is transformed or projected into a CDAE. The leftmost arrows correspond to the SDAE $\mathsf{enc}^{0:L}$, the rightmost arrows correspond to the decoders $\mathsf{dec}^{L:0}$, and the bottom arrows correspond to the CDAE $\mathsf{dae}_L^0 \circ \cdots \circ \mathsf{dae}_0^0$.

6.5. Numerical Example

Figure 15 compares the transportation results of the 2-dimensional swissroll data by the DAEs. In both the cases, the swissroll becomes thinner by the action of transportation. We remark that to test the topological conjugacy by numerical experiments is difficult. Here, we display Figure 15 to see typical trajectories by an SDAE and a CDAE.

In the left-hand side, we trained an SDAE $\mathsf{enc}^1 \circ \mathsf{enc}^0$ by using real NNs. Specifically, we first trained a shallow DAE dae_0^0 on the swissroll data $\boldsymbol{x}_0$. Second, writing $\mathsf{dae}_0^0 = \mathsf{dec}_0 \circ \mathsf{enc}_0$ and letting $\boldsymbol{z}^1 := \mathsf{enc}_0(\boldsymbol{x}_0)$, we trained a shallow DAE dae_1^1 on the feature vectors $\boldsymbol{z}^1$. Then, writing $\mathsf{dae}_1^1 = \mathsf{dec}_1 \circ \mathsf{enc}_1$, we obtained $\boldsymbol{x}_1 := \mathsf{dae}_0^0(\boldsymbol{x}_0)$ and $\boldsymbol{x}_2 := \mathsf{dec}^0 \circ \mathsf{dec}^1 \circ \mathsf{enc}^1 \circ \mathsf{enc}^0$. The black points represent the input vectors $\boldsymbol{x}_0$, and the red and blue points represent the first and second transportation results $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively. In other words, the distribution of $\boldsymbol{x}_0, \boldsymbol{x}_1$ and $\boldsymbol{x}_2$ correspond to μ_0^0, μ_1^0 and μ_2^0 in Figure 14, respectively.

In the right-hand side, we trained a CDAE $\mathsf{dae}_0^1 \circ \mathsf{dae}_0^0$ by using real NNs. Specifically, we first trained a shallow DAE dae_0^0 on the swissroll data $\boldsymbol{x}_0$. Second, writing $\boldsymbol{x}_1 := \mathsf{dae}_0^0(\boldsymbol{x}_0)$, we trained a shallow DAE dae_1^0 on the transported vectors $\boldsymbol{x}_1^0$. Then, we obtained $\boldsymbol{x}_2 := \mathsf{dae}_0^1(\boldsymbol{x}_1) = \mathsf{dae}_0^1 \circ \mathsf{dae}_0^0(\boldsymbol{x}_0)$. The black points represent the input vectors $\boldsymbol{x}_0$, and the red and blue points represent the first and second transportation results $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively.

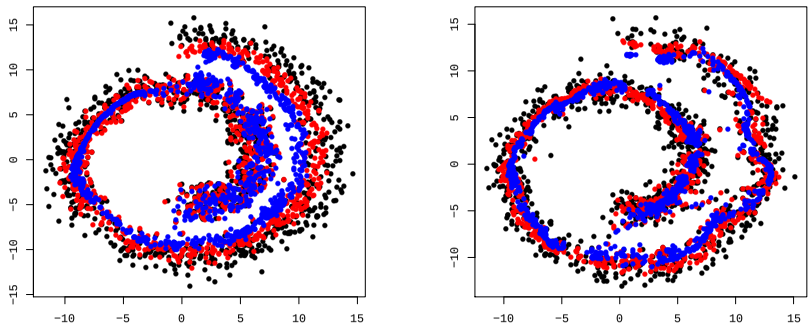


Figure 15: Typical transportation results of the 2-dimensional swissroll data by an SDAE (left) and a CDAE (right). In **both** the sides, the **black** points represent the input vectors $\boldsymbol{x}_0 \in \mathbb{R}^2$, and the **red** and **blue** points represent the first and second transportation results $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively.

7. Integral Representation of the Flow Representation

In this section, we aim to develop the double continuum limit: a combination of the depth continuum limit, or the flow representation, and the width continuum limit, or the integral representation.

To facilitate visualization, we write the hidden parameters as $\boldsymbol{\theta}$ instead of $(\boldsymbol{a}, b)$, the k-th element of the coefficient function as $\gamma(\boldsymbol{\theta}, k)$ or $\gamma_k(\boldsymbol{\theta})$ instead of the boldface $\boldsymbol{\gamma}(\boldsymbol{\theta})$, and the integral representation as

$$S[\gamma_k](\boldsymbol{x}) = \int \gamma(\boldsymbol{\theta}, k)\sigma(\boldsymbol{x}; \boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}. \tag{57}$$

Furthermore, by using a singular measure $\gamma_k^p(\boldsymbol{\theta}) := \sum_{j=1}^p c_{jk}\delta_{\boldsymbol{\theta}_j}(\boldsymbol{\theta})$, we write an ordinary shallow NN as

$$S[\gamma_k^p](\boldsymbol{x}) = \int \gamma^p(\boldsymbol{\theta}, k)\sigma(\boldsymbol{x}; \boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta} = \sum_{j=1}^p c_{jk}\sigma(\boldsymbol{x}; \boldsymbol{\theta}_j). \tag{58}$$

If there is no risk of confusion, we omit writing the superscript p. Specifically, we write "$S[\gamma_k]$" without distinction between an infinite NN (57) and a finite NN (58).

7.1. Encoder and Decoder in the Integral Representation

First, we consider a finite case. Suppose that a shallow DAE is realized by a finite NN $\sum_{j=1}^p c_{jk}\sigma(\boldsymbol{x}; \boldsymbol{\theta}_j)$. Then, the encoder is given by

$$z(\boldsymbol{\theta}_j) = \mathsf{enc}(\boldsymbol{x}, \boldsymbol{\theta}_j) = \sigma(\boldsymbol{x}; \boldsymbol{\theta}_j), \quad j = 1, \dots, p;$$

and the decoder is given by

$$\mathsf{dec}(\boldsymbol{z}, k) = \sum_{j=1}^p c_{jk} z(\boldsymbol{\theta}_j).$$

Therefore, supposing that a shallow DAE is realized by $S[\gamma]$, the encoder and decoder in the integral representation are given by

$$\mathsf{enc}(\boldsymbol{x}, \boldsymbol{\theta}) := \sigma(\boldsymbol{x}; \boldsymbol{\theta}), \tag{59}$$

$$\mathsf{dec}(\boldsymbol{z}, k) := \int \gamma(\boldsymbol{\theta}, k) z(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}, \tag{60}$$

where "the $\boldsymbol{\theta}$-th element" of $\boldsymbol{z}$ is given by $z(\boldsymbol{\theta})$.

Next, we consider the stacked DAE built on $\boldsymbol{z}$. Suppose that the stacked DAE is realized by $S[\widetilde{\gamma}_{\boldsymbol{\theta}}](\boldsymbol{z}) = \int \widetilde{\gamma}(\boldsymbol{\omega}, \boldsymbol{\theta})\sigma(\boldsymbol{z}; \boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}$; then, the encoder and decoder are given by

$$\widetilde{\mathsf{enc}}(\boldsymbol{z}, \boldsymbol{\omega}) := \sigma(\boldsymbol{z}; \boldsymbol{\omega}), \tag{61}$$

$$\widetilde{\mathsf{dec}}(\boldsymbol{u}, \boldsymbol{\theta}) := \int \widetilde{\gamma}(\boldsymbol{\omega}, \boldsymbol{\theta}) u(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}, \tag{62}$$

where the $\boldsymbol{\omega}$-th element of $\boldsymbol{u}$ is given by $u(\boldsymbol{\omega})$, and the $\boldsymbol{\theta}$-th element of $\boldsymbol{\omega}$ is given by $\omega(\boldsymbol{\theta})$.

In this notation, for example, the topological conjugacy (55) claims that there exists γ' such that

$$\int \gamma(\boldsymbol{\theta}, k) \int \widetilde{\gamma}(\boldsymbol{\omega}, \boldsymbol{\theta})\sigma(\sigma(\boldsymbol{x}; \cdot); \boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\mathrm{d}\boldsymbol{\theta} = \int \gamma'(\boldsymbol{\theta}', k)\sigma\left(\int \gamma(\boldsymbol{\theta}, \cdot)\sigma(\boldsymbol{x}; \boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}; \boldsymbol{\theta}'\right)\mathrm{d}\boldsymbol{\theta}'. \tag{63}$$

7.2. Ridgelet Transform of Flows

Let $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$ be a flow that satisfies $\boldsymbol{\varphi}_t \circ \boldsymbol{\varphi}_s = \boldsymbol{\varphi}_{t+s}$. Then, the following formula holds:

$$\int R[\boldsymbol{\varphi}_t](\boldsymbol{\theta}, k)\sigma\left(\int R[\boldsymbol{\varphi}_s](\boldsymbol{\theta}, \cdot)\sigma(\boldsymbol{x}; \boldsymbol{\theta}')\mathrm{d}\boldsymbol{\theta}'\right)\mathrm{d}\boldsymbol{\theta} = \int R[\boldsymbol{\varphi}_{t+s}](\boldsymbol{\theta}, k)\sigma(\boldsymbol{x}; \boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}. \tag{64}$$

In other words, $S[R[\boldsymbol{\varphi}_t]] \circ S[R[\boldsymbol{\varphi}_s]] = S[R[\boldsymbol{\varphi}_{t+s}]]$. According to Barron's bound (Kůrková, 2012, Cor.5.4), the discretization error $\|S[\gamma] - S[\gamma^p]\|_2$ between $S[\gamma]$ and $S[\gamma^p]$ is bounded by $\|\gamma\|_1/\sqrt{p}$. Hence, $\|R[\boldsymbol{\varphi}_t]\|_1 + \|R[\boldsymbol{\varphi}_s]\|_1 \leq \|R[\boldsymbol{\varphi}_{t+s}]\|_1$ for some t and s, which implies the expressive efficiency of the DNN.

Consider a special case when $\boldsymbol{\varphi} : \mathbb{R}^m \to \mathbb{R}^m$ is given by the gradient of a potential function V. Specifically, $\boldsymbol{\varphi} = \nabla V$. We note that according to the polar decomposition theorem by Brenier (1991), any optimal transport map $\boldsymbol{\varphi}_t : [0, 1] \times \mathbb{R}^m \to \mathbb{R}^m$ can be written as $\boldsymbol{\varphi}_t = \mathsf{id} + t\nabla U$ with some potential function U. Hence, by letting $V = |\cdot|^2/2 + U$, we can understand $\boldsymbol{\varphi} := \boldsymbol{\varphi}_1 = \nabla V$ as an optimal transport map.

Then, we have an integration-by-parts formula for the vector ridgelet transform.

Theorem 11 *Let $K \subset \mathbb{R}^m$ be a compact set with smooth boundary ∂K. Given that a smooth scalar potential V is supported in K, the ridgelet transform of the potential vector field ∇V is calculated by*

$$R_\rho[\nabla V](\boldsymbol{a}, b) = -\boldsymbol{a} R_{\rho'}[V](\boldsymbol{a}, b). \tag{65}$$

Here, R_ρ and $R_{\rho'}$ denote the ridgelet transform with respect to ρ and ρ', respectively.

Proof

$$\begin{aligned} R_\rho[\nabla V](\boldsymbol{a}, b) &= \int_K \nabla V(\boldsymbol{x}) \overline{\rho(\boldsymbol{a} \cdot \boldsymbol{x} - b)} \mathrm{d}\boldsymbol{x} \\ &= \left[\int_{\partial K} V(\boldsymbol{x}) \overline{\rho(\boldsymbol{a} \cdot \boldsymbol{x} - b)} \boldsymbol{n}(\boldsymbol{x}) \mathrm{d}S - \boldsymbol{a} \int_K V(\boldsymbol{x}) \overline{\rho'(\boldsymbol{a} \cdot \boldsymbol{x} - b)} \mathrm{d}\boldsymbol{x} \right] \\ &= 0 - \boldsymbol{a}\, R_{\rho'}[V](\boldsymbol{a}, b). \end{aligned}$$

■

The left-hand side (LHS) of (65) denotes a vector ridgelet transform defined by element-wise mapping, whereas the right-hand side (RHS) consists of a scalar ridgelet transform. We can understand the RHS given that the network shares common knowledge among element-wise tasks.

7.3. Example: Autoencoder

As the most fundamental transport map, we consider a smooth "truncated" autoencoder $\mathsf{id}_{r,\delta}$. We denote by $\mathbb{B}^m(\boldsymbol{z}; r)$ a closed ball in $\mathbb{R}^m$ with center $\boldsymbol{z}$ and radius r. We assume that $\mathsf{id}_{r,\delta}$ is (1) smooth, (2) equal to the identity map id when it is restricted to $\mathbb{B}^m(r)$, and (3) truncated to be supported in $\mathbb{B}^m(r+\delta)$ with a small positive number $\delta > 0$. Let $\nabla V_{r,\delta}$ be a smooth function that satisfies

$$V_{r,\delta}(\boldsymbol{x}) := \begin{cases} \frac{1}{2}|\boldsymbol{x}|^2 & \boldsymbol{x} \in \mathbb{B}^m(0; r), \\ (\text{smooth map}) & \boldsymbol{x} \in \mathbb{B}(0; r+\delta) \setminus \mathbb{B}(0; r), \\ 0 & \boldsymbol{x} \notin \mathbb{B}^m(0; r+\delta), \end{cases}$$

and let

$$\mathsf{id}_{r,\delta} := \nabla V_{r,\delta}.$$

Note that we can construct $\mathsf{id}_{r,\delta}$ and $\nabla V_{r,\delta}$ by using mollifiers; thus, such maps exist.

The ridgelet transform of the truncated autoencoder is given by

$$R_\rho[\mathsf{id}_{r,\delta}](\boldsymbol{a}, b) \approx -K \boldsymbol{a} \overline{\rho'(-b)} \quad \text{as} \quad \delta \to 0 \tag{66}$$

with a certain constant K (see Appendix E for the proof).

8. Discussion

We performed transport analysis of denoising autoencoders by introducing the flow representation. The flow representation $\boldsymbol{\varphi}_t$ is the depth continuum limit of a DNN, specified by an ODE with vector field $\boldsymbol{v}_t$. We interpreted an ordinary DNN $\boldsymbol{g}_t$ as a transport map or an Euler broken line approximation of $\boldsymbol{\varphi}_t$. The advantages of the flow representation are that it provides the coordinate-free treatment of DNNs, avoiding the redundancy of the ordinary parametrization of DNNs, and that it facilitates our understanding of what DNNs do—it is the mass transportation controlled by $\boldsymbol{v}_t$. In addition, the advantage of the interpretation as mass transportation is that it can handle function composition. In the transport analysis, we analyzed a flow in three aspects: a dynamical system described by a transport map or vector field, a pushforward measure described by a continuity equation, and Wasserstein gradient flow. From the results in Wasserstein geometry, these aspects are closely connected, and the hyperparameter $\boldsymbol{v}_t$ plays a central role as an intermediary. For example, in the transport analysis of continuous DAEs, the potential functional of the Wasserstein gradient flow often facilitates our understanding of the flow because it is the Shannon entropy, which is a fundamental quantity in statistics and machine learning.

In Section 3 and 4, we specified the transport maps of shallow, deep, and infinitely deep DAEs, and we gave their statistical interpretations. The shallow DAE is an estimator of the mean, while the deep DAE transports data points to decrease the Shannon entropy of the data distribution, which gives a partial answer to our research question "what do hidden layers do?" In Section 5, according to analytic and numerical experiments, we showed that deep DAEs converge faster and that the extracted features are different from each other, which gives a partial answer to the other question "why do DNNs perform better?" In Section 6, we proved the equivalence between the stacked DAE and the composition of DAEs. Because of the peculiar construction, it is difficult to formulate and understand stacking. Nevertheless, by tracking the flow, we succeeded in formulating the stacked DAE. In Section 7, we developed the double continuum limits, or the width continuum limit of the depth continuum limit. We presented some examples of the integral representation of the flow, such as encoder, decoder, and traditional autoencoder.

As a consequence of the equivalence, we can understand the so-called *pre-training* and *fine-tuning* strategy (Bengio et al., 2007; Erhan et al., 2010) as an *optimal control* problem. Namely, write a DNN as a composite $\boldsymbol{\psi} \circ \boldsymbol{\varphi}_t$ of classifier $\boldsymbol{\psi} : \mathbb{R}^m \to [0,1]^n$ and flow $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$. If $\boldsymbol{\varphi}_t$ stays closer to the identity, $\boldsymbol{\psi}$ has to be more complex—and vice versa. The pre-training regularizes the behavior of hidden layers by

$$\frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{\varphi}_t(\boldsymbol{x}) = \boldsymbol{v}_t(\boldsymbol{\varphi}_t(\boldsymbol{x})), \quad \boldsymbol{x} \in \mathbb{R}^m,\, t > 0 \tag{67}$$

and the fine-tuning specifies the relation between input and output by

$$\text{Minimize} \quad \mathbb{E}_{X,Y} |Y - \boldsymbol{\psi} \circ \boldsymbol{\varphi}_{t=1}(X)|^2 \quad \text{w.r.t NN } \boldsymbol{\psi} \circ \boldsymbol{\varphi}_{t=1}. \tag{68}$$

Overall, we can understand the strategy as the control problem of system (67) under restriction (68). Owing to ridgelet transform, shallow NNs are interpretable and principled. Development of a "solution operator" to the control problem in the flow representation would open the way to the interpretable and principled alternative to DNNs.

Acknowledgments

The authors thank the editor and reviewers for their supportive and insightful comments, which have improved the clarity of the argument significantly. The authors also acknowledge fruitful discussions with Dr. Shotaro Akaho, Dr. Kohei Yatabe, Dr. Keisuke Yano, and Mr. Kentaro Minami. This work was supported by JSPS KAKENHI (15J07517 and 18K18113).

Appendix A. Proof of Theorem 4

By $L^1_{\mathrm{loc}}(\mathbb{R}^m)$ and $C^\infty_c(\mathbb{R}^m)$, we denote the spaces of locally integrable functions and compactly supported smooth functions, respectively. We assume that $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^m$ is locally integrable (L^1_{loc}).

Proof The proof follows from the calculus of variations. Let

$$\begin{aligned} L[\boldsymbol{g}] &= \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} |\boldsymbol{g}(\boldsymbol{x}+\boldsymbol{\varepsilon}) - \boldsymbol{x}|^2 \mu_0(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\ &= \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{g}(\boldsymbol{x}') - \boldsymbol{x}' + \boldsymbol{\varepsilon}|^2 \mu_0(\boldsymbol{x}' - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x}', \quad \boldsymbol{x}' \leftarrow \boldsymbol{x} + \boldsymbol{\varepsilon}. \end{aligned}$$

Here, $L[\boldsymbol{g}]$ always exists because $\boldsymbol{g} \in L^1_{\mathrm{loc}}(\mathbb{R}^m) \subset L^2(\mu * \nu)$. Then, for an arbitrary function $\boldsymbol{h} \in C^\infty_c(\mathbb{R}^m)$, the first variation $\delta L[\boldsymbol{h}]$ is given by

$$\begin{aligned} \delta L[\boldsymbol{h}] &= \frac{d}{ds} L[\boldsymbol{g} + s\boldsymbol{h}]\Big|_{s=0} \\ &= \int_{\mathbb{R}^m} \frac{\partial}{\partial s} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{g}(\boldsymbol{x}) + s\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{x} + \boldsymbol{\varepsilon}|^2 \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x}\Big|_{s=0} \\ &= 2 \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [(\boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{x} + \boldsymbol{\varepsilon}) \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \boldsymbol{h}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}. \end{aligned}$$

At a critical point $\boldsymbol{g}^*$ of L, $\delta L[\boldsymbol{h}] \equiv 0$ for every $\boldsymbol{h}$. Hence,

$$\mathbb{E}_{\boldsymbol{\varepsilon}} [(\boldsymbol{g}^*(\boldsymbol{x}) - \boldsymbol{x} + \boldsymbol{\varepsilon}) \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] = 0, \quad \mathsf{a.e.}\, \boldsymbol{x},$$

by the fundamental lemma of calculus of variations for integrable functions, and we have

$$\begin{aligned} \boldsymbol{g}^*(\boldsymbol{x}) &= \frac{\mathbb{E}_{\boldsymbol{\varepsilon}} [(\boldsymbol{x} - \boldsymbol{\varepsilon}) \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})]}{\mathbb{E}_{\boldsymbol{\varepsilon}} [\mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})]} = (14) \\ &= \boldsymbol{x} - \frac{\mathbb{E}_{\boldsymbol{\varepsilon}} [\boldsymbol{\varepsilon} \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})]}{\mathbb{E}_{\boldsymbol{\varepsilon}} [\mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})]} = (15). \end{aligned}$$

Note that $\boldsymbol{g}^*$ attains the global minimum, because, for every function $\boldsymbol{h}$,

$$\begin{aligned} L[\boldsymbol{g}^* + \boldsymbol{h}] &= \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{\varepsilon} - \mathbb{E}_t[\boldsymbol{\varepsilon}|\boldsymbol{x}] + \boldsymbol{h}(\boldsymbol{x})|^2 \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x} \\ &= \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{\varepsilon} - \mathbb{E}_t[\boldsymbol{\varepsilon}|x]|^2 \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x} + \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{h}(\boldsymbol{x})|^2 \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x} \\ &\quad + 2 \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [(\boldsymbol{\varepsilon} - \mathbb{E}_t[\boldsymbol{\varepsilon}|\boldsymbol{x}]) \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \boldsymbol{h}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\ &= L[\boldsymbol{g}^*] + L[\boldsymbol{h}] + 2 \cdot 0 \geq L[\boldsymbol{g}^*]. \end{aligned}$$

■

Appendix B. Proof of Fact 2

For simplicity, we assume that $\boldsymbol{g}$, $\boldsymbol{v}$, and μ are smooth. See Ambrosio et al. (2008, § 8.1) for more generalized conditions on the continuity equation.

Proof To facilitate visualization, we write $\boldsymbol{g}(\boldsymbol{x}, t)$, $\boldsymbol{v}(\boldsymbol{x}, t)$, and $\mu(\boldsymbol{x}, t)$ instead of $\boldsymbol{g}_t(\boldsymbol{x})$, $\boldsymbol{v}_t(\boldsymbol{x})$, and $\mu_t(\boldsymbol{x})$, respectively.

By definition,

$$\begin{cases} \partial_t \boldsymbol{g}(\boldsymbol{g}(\boldsymbol{x}, t), t) = \boldsymbol{v}(\boldsymbol{g}(\boldsymbol{x}, t), t), & \boldsymbol{x} \in \mathbb{R}^m,\ t > 0 \\ \boldsymbol{g}(\boldsymbol{x}, 0) = 0, & \boldsymbol{x} \in \mathbb{R}^m. \end{cases}$$

In particular,

$$\nabla \boldsymbol{g}(\boldsymbol{x}, 0) = I.$$

According to the change-of-variables formula, for any $\boldsymbol{x} \in \mathbb{R}^m$ and $t > s > 0$,

$$\mu(\boldsymbol{g}(\boldsymbol{x}, t), t) \cdot |\nabla \boldsymbol{g}(\boldsymbol{x}, t)| = \mu(\boldsymbol{x}, s),$$

where $|\cdot|$ denotes the determinant.

Take the logarithm on both sides and then differentiate with respect to t. Then, the RHS vanishes and the LHS is calculated as follows:

$$\begin{aligned} \partial_t \log[\mu(\boldsymbol{g}(\boldsymbol{x}, t), t) \cdot |\nabla \boldsymbol{g}(\boldsymbol{x}, t)|] &= \frac{\partial_t [\mu(\boldsymbol{g}(\boldsymbol{x}, t), t)]}{\mu(\boldsymbol{g}(\boldsymbol{x}, t), t)} + \partial_t \log |\nabla \boldsymbol{g}(\boldsymbol{x}, t)| \\ &= \frac{(\nabla \mu)(\boldsymbol{g}(\boldsymbol{x}, t), t) \cdot \partial_t \boldsymbol{g}(\boldsymbol{x}, t) + (\partial_t \mu)(\boldsymbol{g}(\boldsymbol{x}, t), t)}{\mu(\boldsymbol{g}(\boldsymbol{x}, t), t)} \\ &\quad + \mathsf{tr}\,[(\nabla \boldsymbol{g}(\boldsymbol{x}, t))^{-1} \nabla \partial_t \boldsymbol{g}(\boldsymbol{x}, t)], \end{aligned}$$

where the second term follows a differentiation formula by Petersen and Pedersen (2012, Eq. 43)

$$\partial \log |J| = \mathsf{tr}\,[J^{-1} \partial J].$$

By letting $t \to s + 0$,

$$\frac{\nabla \mu(\boldsymbol{x}, t) \cdot \boldsymbol{v}(\boldsymbol{x}, t) + (\partial_t \mu)(\boldsymbol{x}, t)}{\mu(\boldsymbol{x}, t)} + \mathsf{tr}\,[\nabla \boldsymbol{v}(\boldsymbol{x}, t)] = 0,$$

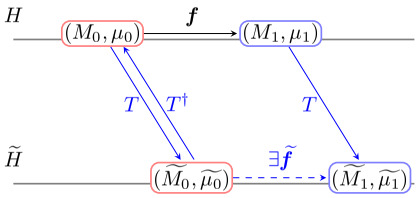
which gives

$$\partial_t \mu(\boldsymbol{x}, t) = -\nabla \cdot [\mu(\boldsymbol{x}, t) \boldsymbol{v}(\boldsymbol{x}, t)].$$

■

Appendix C. Proof of Theorem 9

We show that the diagram commutes. Observe that $\boldsymbol{f} = \mathsf{id} + tD\nabla \log e^{tL_t} \mu$ is the sum of the present position id and the gradient ∇V of potential $V = \log e^{tL_t} \mu$. We calculate the pushforward $\widetilde{\nabla} \widetilde{V}$ and show that it coincides with $\widetilde{L}_t$-DAE.



Proof We suppose that L_t is expressed as

$$L_t u := \boldsymbol{a}_t^\top (\nabla^2 u) \boldsymbol{a}_t + \boldsymbol{b}_t^\top \nabla u + c_t u, \quad u \in C^2(H) \tag{69}$$

and T is expressed as

$$T(\boldsymbol{z}) = A\boldsymbol{z} \tag{70}$$

with a matrix A.

By the assumption that the restriction $T|_{M_0}$ is injective, it has a left inverse $T^\dagger$ such that $T^\dagger \circ T|_{M_0} = \mathsf{id}_{M_0}$. Note that it is not a linear map but an abstract nonlinear map, which means that there is no matrix A that realizes $T^\dagger$.

Step. 1

We show that

$$T \circ \boldsymbol{f} \circ T^\dagger = \mathsf{id} + t\widetilde{D}\widetilde{\nabla}\widetilde{V} \quad \text{in } \widetilde{M}_0 \tag{71}$$

where $\widetilde{D} = ADA^\top$ and $\widetilde{V} = V \circ T^\dagger$.

For an arbitrary $U \in C^2(M_0)$, write $T_* U := U \circ T^\dagger \in C^2(\widetilde{M}_0)$, and

$$\nabla U(T^\dagger(\boldsymbol{x})) = A^\top \widetilde{\nabla} T_* U(\boldsymbol{x}), \quad \boldsymbol{x} \in \widetilde{M}_0 \tag{72}$$

because the i-th element of $\widetilde{\nabla} T_* U$ is given by

$$\frac{\partial U \circ T^\dagger}{\partial x_i}(\boldsymbol{x}) = \sum_p \frac{\partial U}{\partial z_p}(T^\dagger(\boldsymbol{x})) \frac{\partial T_p^\dagger}{\partial x_i}(\boldsymbol{x}).$$

Thus, the q-th element of $A^\top \widetilde{\nabla} T_* U$ is given by

$$\begin{aligned}\sum_i A_{iq} \frac{\partial U \circ T^\dagger}{\partial x_i}(\boldsymbol{x}) &= \sum_p \frac{\partial U}{\partial z_p}(T^\dagger(\boldsymbol{x})) \sum_i A_{iq} \frac{\partial T_p^\dagger}{\partial x_i}(\boldsymbol{x}) \\ &= \sum_p \frac{\partial U}{\partial z_p}(T^\dagger(\boldsymbol{x})) \delta_{pq} \\ &= \frac{\partial U}{\partial z_q}(T^\dagger(\boldsymbol{x})).\end{aligned}$$

Therefore, by substituting U with $V = \log e^{tL_t}\mu_0$,

$$\begin{aligned}T \circ \boldsymbol{f} \circ T^\dagger(\boldsymbol{x}) &= \boldsymbol{x} + A(tD\nabla V(T^\dagger(\boldsymbol{x}))) \\ &= \boldsymbol{x} + tADA^\top \widetilde{\nabla} T_* V(\boldsymbol{x})) \\ &= \boldsymbol{x} + t\widetilde{D}\widetilde{\nabla}\widetilde{V}(\boldsymbol{x}).\end{aligned}$$

Step. 2

We show that

$$\widetilde{V} = \log e^{t\widetilde{L}_t}\widetilde{\mu}_0 + (const.), \quad \text{in } M_0 \tag{73}$$

where

$$\widetilde{L}_t \widetilde{u} := \widetilde{\boldsymbol{a}}_t^\top (\widetilde{\nabla}^2 \widetilde{u}) \widetilde{\boldsymbol{a}}_t + \widetilde{\boldsymbol{b}}_t^\top \widetilde{\nabla} \widetilde{u} + \widetilde{c}_t \widetilde{u}, \quad \widetilde{u} \in C^2(\widetilde{H}) \tag{74}$$

with $\widetilde{\boldsymbol{a}}_t = A\boldsymbol{a}_t \circ T^\dagger, \widetilde{\boldsymbol{b}}_t = A\boldsymbol{b}_t \circ T^\dagger$, and $\widetilde{c}_t = c_t \circ T^\dagger$.

Let

$$u_t := e^{tL_t}\mu_0. \tag{75}$$

By the definition of semigroup e^{tL_t}, $u_0 = \mu_0$ and $\partial_t u_t = L_t u_t$ (however, u_1 is different from μ_1).

Given u_t, let

$$\widetilde{u}_t := T_\sharp u_t. \tag{76}$$

According to the change-of-variables formula (7),

$$\widetilde{u}_t = [A]^{-1} T_* u_t, \tag{77}$$

where $[A] := \sqrt{\det |A^\top A|}$ and $T_* u_t := u_t \circ T^\dagger$. In particular, $\widetilde{u}_0 = \widetilde{\mu}_0$ and $\log \widetilde{u}_t = \widetilde{V}$.

Furthermore,

$$\partial_t \widetilde{u}_t = \widetilde{L}_t \widetilde{u}_t, \quad \text{in } \widetilde{M}_0, \tag{78}$$

because

$$\begin{aligned}\partial_t \widetilde{u}_t(\boldsymbol{x}) &= [A]^{-1} \partial_t [u_t(T^\dagger(\boldsymbol{x}))] \\ &= [A]^{-1} L_t[u_t](T^\dagger(\boldsymbol{x})),\end{aligned}$$

and

$$\begin{aligned}&[A]^{-1} \boldsymbol{a}_t(T^\dagger(\boldsymbol{x}))^\top (\nabla^2 u_t(T^\dagger(\boldsymbol{x}))) \boldsymbol{a}_t(T^\dagger(\boldsymbol{x})) \\ &\quad = \boldsymbol{a}_t(T^\dagger(\boldsymbol{x}))^\top (A^\top \widetilde{\nabla}^2 [[A]^{-1} T_* u_t](\boldsymbol{x}) A) \boldsymbol{a}_t(T^\dagger(\boldsymbol{x})) \\ &\quad = \widetilde{\boldsymbol{a}}_t(\boldsymbol{x})^\top (\widetilde{\nabla}^2 [\widetilde{u}_t](\boldsymbol{x})) \widetilde{\boldsymbol{a}}_t(\boldsymbol{x}), \\ &[A]^{-1} \boldsymbol{b}_t(T^\dagger(\boldsymbol{x}))^\top \nabla u_t(T^\dagger(\boldsymbol{x})) \\ &\quad = \boldsymbol{b}_t(T^\dagger(\boldsymbol{x}))^\top A^\top \widetilde{\nabla} [[A]^{-1} T_* u_t](\boldsymbol{x}) \\ &\quad = \widetilde{\boldsymbol{b}}_t(\boldsymbol{x})^\top \widetilde{\nabla} \widetilde{u}_t(\boldsymbol{x}), \\ &[A]^{-1} c_t(T^\dagger(\boldsymbol{x})) u_t(T^\dagger(\boldsymbol{x})) \\ &\quad = \widetilde{c}_t(\boldsymbol{x}) \widetilde{u}_t(\boldsymbol{x}).\end{aligned}$$

Thus,

$$\partial_t \widetilde{u}_t(\boldsymbol{x}) = [A]^{-1} L_t[u_t](T^\dagger(\boldsymbol{x})) = \widetilde{L}_t \widetilde{u}_t(\boldsymbol{x}).$$

Hence, $\widetilde{u}_t$ is the solution of the initial value problem $\partial_t \widetilde{u}_t = \widetilde{L}_t \widetilde{u}_t$ with $\widetilde{u}_0 = \widetilde{\mu}_0$. By the uniqueness of the solution, $\widetilde{u}_t = e^{t\widetilde{L}_t}\widetilde{\mu}_0$. On the other hand, $\log \widetilde{u}_t = \widetilde{V}$. Therefore, $\widetilde{V} = \log \widetilde{u}_t = e^{t\widetilde{L}_t}\widetilde{\mu}_0$.

To sum up the two steps,

$$T \circ \boldsymbol{f} \circ T^\dagger = \mathsf{id} + t\widetilde{D}\widetilde{\nabla} \log e^{t\widetilde{L}_t}\widetilde{\mu}_0 =: \widetilde{\boldsymbol{f}},$$

and we have the topological conjugacy

$$T \circ \boldsymbol{f} = \widetilde{\boldsymbol{f}} \circ T. \qquad \blacksquare$$

Appendix D. Proofs for Analytic Examples

D.1. Univariate Normal Distribution

We calculate the case for a univariate normal distribution $N(m_0, \sigma_0^2)$.

D.1.1. Shallow DAE

We show that

$$g_t(x) = \frac{\sigma_0^2}{\sigma_0^2 + t} x + \frac{t}{\sigma_0^2 + t} m_0, \tag{30}$$

$$\mu_t = N\left(m_0, \frac{\sigma_0^2}{(1 + t/\sigma_0^2)^2}\right). \tag{31}$$

Proof The proof is immediate from (17). First, write $\phi_t(x, y) = (4\pi t)^{-1/2} \exp(-|x-y|^2/4t)$,

$$\phi_{t/2} * N(m_0, \sigma_0^2) = N(m_0, \sigma_0^2 + t).$$

Hence,

$$g_t(x) = x + t\nabla \log[N(m_0, \sigma_0^2 + t)] = \frac{\sigma_0^2}{\sigma_0^2 + t} x + \frac{t}{\sigma_0^2 + t} m_0.$$

As g_t is affine, the pushforward is immediate. $\blacksquare$

D.1.2. Continuous DAE

We show that

$$g_t(x) = \sqrt{1 - 2t/\sigma_0^2}(x - m_0) + m_0, \tag{32}$$

$$\mu_t = N(m_0, \sigma_0^2 - 2t), \quad 0 \le t < \sigma_0^2/2. \tag{33}$$

Proof [μ_t] Write the pushforward as $N(m_t, \sigma_t^2)$. By using the heat kernel $\phi_t(\boldsymbol{x}, \boldsymbol{y}) = (4\pi t)^{-m/2} \exp(-|\boldsymbol{x} - \boldsymbol{y}|^2/4t)$, for some $T > 0$,

$$\begin{aligned} N(m_t, \sigma_t^2) &= \phi_{T-t} * N(m_T, \sigma_T^2) \\ &= N(m_T, \sigma_T^2 + 2(T - t)). \end{aligned}$$

By eliminating T by the initial conditions, we have

$$N(m_t, \sigma_t^2) = N(m_0, \sigma_0^2 - 2t).$$

By the positivity of σ_t^2, we can determine the largest possible T as $T = \sigma_0^2/2$. $\blacksquare$

Proof [g_t] Fix an arbitrary point x_0. Write $x_t := g_t(x_0)$ and $\dot{x}_t := \partial_t g_t(x_0)$. Recall that $\dot{m}_t \equiv 0$, because m_t is a constant. According to (24),

$$\dot{x}_t = -\frac{x_t - m_t}{\sigma_t^2}.$$

By dividing both sides by x_t and integrating them,

$$\begin{aligned} \log\left|\frac{x_t - m_t}{x_0 - m_0}\right| &= -\int_0^t \frac{\mathrm{d}s}{\sigma_s^2} \\ &= \frac{1}{2}\int_0^t \frac{\mathrm{d}s}{s - T} \\ &= \frac{1}{2}\log\left|\frac{T - t}{T}\right|, \end{aligned}$$

which concludes the proof. $\blacksquare$

D.2. Multivariate Normal Distribution

We calculate the case for a multivariate normal distribution $N(\boldsymbol{m}_0, \Sigma_0)$.

D.2.1. Shallow DAE

We show that

$$\boldsymbol{g}_t(\boldsymbol{x}) = (I + t\Sigma_0^{-1})^{-1}\boldsymbol{x} + (I + t^{-1}\Sigma_0)^{-1}\boldsymbol{m}_0, \tag{34}$$

$$\mu_t = N(\boldsymbol{m}_0, \Sigma_0(I + t\Sigma_0^{-1})^{-2}). \tag{35}$$

Proof Calculate (17) directly as in the univariate case. First, by writing $\phi_t(\boldsymbol{x}, \boldsymbol{y}) = (4\pi t)^{-m/2} \exp(-|\boldsymbol{x} - \boldsymbol{y}|^2/4t)$,

$$\phi_{t/2} * N(\boldsymbol{m}_0, \Sigma_0) = N(\boldsymbol{m}_0, \Sigma_0 + tI).$$

Hence,

$$\begin{aligned}\boldsymbol{g}_t(\boldsymbol{x}) &= \boldsymbol{x} + t\nabla \log[N(\boldsymbol{m}_0, \Sigma_0 + tI)] \\ &= \boldsymbol{x} + t\nabla\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m}_0)^\top(\Sigma_0+tI)^{-1}(\boldsymbol{x}-\boldsymbol{m}_0)\right] \\ &= (I + t\Sigma_0^{-1})^{-1}\boldsymbol{x} + (I + t^{-1}\Sigma_0)^{-1}\boldsymbol{m}_0.\end{aligned}$$

As $\boldsymbol{g}_t$ is affine, the pushforward is immediate. ■

D.2.2. Continuous DAE

We show that

$$\boldsymbol{g}_t(\boldsymbol{x}) = \sqrt{I - 2t\Sigma_0^{-1}}(\boldsymbol{x}-\boldsymbol{m}_0) + \boldsymbol{m}_0, \tag{36}$$

$$\mu_t = N(\boldsymbol{m}_0, \Sigma_0 - 2tI). \tag{37}$$

Proof Write $\phi_t(\boldsymbol{x}, \boldsymbol{y}) = (4\pi t)^{-m/2}\exp(-|\boldsymbol{x}-\boldsymbol{y}|^2/4t)$, and recall that $\phi_t * N(\boldsymbol{m}, \Sigma) = N(\boldsymbol{m}, \Sigma + 2tI)$. Thus, the pushforward $N(\boldsymbol{m}_t, \Sigma_t)$ is obtained as follows in a manner similar to the univariate case.

$$N(\boldsymbol{m}_t, \Sigma_t) = N\left(\boldsymbol{m}_0, \Sigma_0 - 2tI\right).$$

Suppose that $\boldsymbol{g}_t(\boldsymbol{x})$ is an affine transform $A_t(\boldsymbol{x}-\boldsymbol{m}_0)+\boldsymbol{m}_0$ analogous to the univariate case. Recall that, if $X \sim N(\boldsymbol{m}, \Sigma)$, then $AX+b \sim N(A\boldsymbol{m}+b, A\Sigma A^\top)$. Hence, for our case, $\Sigma_t = A_t\Sigma_0 A_t^\top$ and we can determine

$$A_t = \sqrt{\Sigma_t\Sigma_0^{-1}} = \sqrt{I - 2t\Sigma_0^{-1}}.$$

Finally, we check whether $\boldsymbol{g}_t$ satisfies (24). As Σ_0 is symmetric, we can always diagonalize $\Sigma_0 = UD_0U^\top$ with an orthogonal matrix U and a diagonal matrix D_0. Observe that with the same U, we can simultaneously diagonalize Σ_t and A_t as

$$\begin{aligned}\Sigma_t &= UD_tU^\top, \quad D_t := D_0 - 2tI \\ A_t &= UD_t^{1/2}D_0^{-1/2}U^\top.\end{aligned}$$

Without loss of generality, we can assume that $U = I$; therefore, Σ_t and A_t are diagonal and $\boldsymbol{m}_t \equiv 0$. Fix an index j and denote the j-th diagonal element of Σ_t and A_t by σ_t^2 and a_t, respectively. Then, our goal is reduced to showing that $\partial_t[a_t x] = \nabla \log \mu_t(a_t x)$ for every fixed $x \in \mathbb{R}$.

By definition,

$$\begin{aligned}\sigma_t^2 &= \sigma_0^2 - 2t, \\ a_t &= \sigma_t\sigma_0^{-1} = \sqrt{1 - 2t\sigma_0^{-2}}.\end{aligned}$$

Thus, the LHS is

$$\partial_t[a_t x] = -\frac{1}{\sigma_0\sqrt{\sigma_0^2 - 2t}}x = -\sigma_0^{-1}\sigma_t^{-1}x,$$

and the RHS is

$$\nabla \log \mu_t(a_t x) = -\frac{a_t x}{\sigma_t^2} = -\sigma_0^{-1}\sigma_t^{-1}x.$$

Hence, the LHS equals the RHS. ■

D.3. Mixture of Multivariate Normal Distributions

We calculate the case for the mixture of multivariate normal distributions $\sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k)$, with the assumption that it is *well separated* (see Section 5.1.3 for the definition).

D.3.1. Shallow DAE

We show that

$$\boldsymbol{g}_t(\boldsymbol{x}) = \sum_{k=1}^K \gamma_{kt}(\boldsymbol{x})\left\{(I + t\Sigma_k^{-1})^{-1}\boldsymbol{x} + (I + t^{-1}\Sigma_k)^{-1}\boldsymbol{m}_k\right\}, \tag{38}$$

$$\mu_t \approx \sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k(I + t\Sigma_k^{-1})^{-2}), \quad \text{if well separated} \tag{39}$$

with the responsibility function

$$\gamma_{kt}(\boldsymbol{x}) := \frac{w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k + tI)}{\sum_{k=1}^K w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k + tI)}. \tag{40}$$

Proof Directly calculate (17). By the linearity of the heat kernel,

$$\begin{aligned}\boldsymbol{g}_t &:= \mathsf{id} + t\sum_{k=1}^K \frac{w_k \nabla N(\boldsymbol{m}_k, \Sigma_k + tI)}{\sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k + tI)}, \\ &= \mathsf{id} + \sum_{k=1}^K \frac{w_k N(\boldsymbol{m}_k, \Sigma_k + tI)}{\sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k + tI)} \cdot t\nabla \log N(\boldsymbol{m}_k, \Sigma_k + tI), \\ &= \mathsf{id} + \sum_{k=1}^K \gamma_{kt}(\boldsymbol{g}_{kt} - \mathsf{id}), \\ &= \sum_{k=1}^K \gamma_{kt}\boldsymbol{g}_{kt},\end{aligned}$$

where $\boldsymbol{g}_{kt}$ exactly coincides with the flow induced by the individual k-th component.

To calculate the pushforward, we introduce some auxiliary variables. Write $w(k) := w_k$, $\gamma(k \mid \cdot) := \gamma_{kt}(\cdot)$ and

$$\mu_t(\cdot \mid k) := N(\boldsymbol{m}_k, \Sigma_k + tI),$$
$$\mu_t := \sum_k w(k)\mu_t(\cdot \mid k).$$

Let $\tau_k(\cdot \mid \boldsymbol{x})$ be a probability measure that satisfies

$$\int_M \tau_k(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x} \mid k)\mathrm{d}\boldsymbol{x} = \mu_t(y \mid k).$$

Note that τ_k is not unique. Recall that by definition, if $X \sim \mu_0(\cdot \mid k)$, then $Y = \boldsymbol{g}_{kt}(X) \sim \mu_t(\cdot \mid k)$. Hence, τ_k is a stochastic alternative to $\boldsymbol{g}_{kt}$.

Consider a probability measure

$$\sigma(\cdot \mid \boldsymbol{x}) := \sum_{k=1}^K \gamma(k \mid \boldsymbol{x})\tau_k(\cdot \mid \boldsymbol{x}).$$

Clearly, this is a stochastic alternative to $\boldsymbol{g}_t$. We show that

$$\int_M \sigma(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \approx \mu_t(y).$$

The LHS is reduced to

$$\begin{aligned}\int_M \sigma(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x})\mathrm{d}\boldsymbol{x} &= \int_M \sum_{k=1}^K \gamma(k \mid \boldsymbol{x})\tau_k(y \mid \boldsymbol{x}) \sum_\ell w(\ell)\mu_0(\boldsymbol{x} \mid \ell)\mathrm{d}\boldsymbol{x} \\ &= \sum_\ell w(\ell) \sum_{k=1}^K \int_M \gamma(k \mid \boldsymbol{x})\tau_k(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x} \mid \ell)\mathrm{d}\boldsymbol{x}. \end{aligned} \tag{79}$$

Suppose that $\gamma(k \mid \boldsymbol{x})$ is an indicator function of a domain Ω_k, where $\int_{\Omega_k} \mu_0(\cdot \mid k) \approx 1$. Then,

$$\begin{aligned}(79) &\approx \sum_\ell w(\ell) \int_{\Omega_\ell} \tau_k(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x} \mid \ell)\mathrm{d}\boldsymbol{x} \\ &\approx \sum_\ell w(\ell)\mu_t(y \mid \ell) = \mu_t(y).\end{aligned}$$

This concludes the claim. ■

D.3.2. Continuous DAE

We show that

$$\boldsymbol{g}_t(\boldsymbol{x}) \approx \sqrt{I - 2t\Sigma_k^{-1}}\,(\boldsymbol{x} - \boldsymbol{m}_k) + \boldsymbol{m}_k, \quad \boldsymbol{x} \in \Omega_k, \text{ if well separated} \tag{41}$$

$$\mu_t = \sum_{k=1}^K w_k N\left(\boldsymbol{m}_k, \Sigma_k - 2tI\right), \tag{42}$$

with the responsibility function

$$\gamma_{kt}(\boldsymbol{x}) := \frac{w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k - 2tI)}{\sum_{k=1}^K w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k - 2tI)}. \tag{43}$$

Proof The pushforward is immediate by the linearity of the heat kernel. The dynamical system (24) for our case is reduced to

$$\partial_t \boldsymbol{g}_t(\boldsymbol{x}) = -\sum_{k=1}^K \gamma_{kt} \circ \boldsymbol{g}_t(\boldsymbol{x})(\Sigma_k - 2tI)^{-1}(\boldsymbol{g}_t(\boldsymbol{x}) - \boldsymbol{m}_k).$$

By the assumption that μ_0 is well separated, we can take an open neighborhood Ω_k of $\boldsymbol{m}_k$ and an open time interval I that contains t such that $\gamma_{kt} \circ \boldsymbol{g}_t(\boldsymbol{x}) \equiv 1$ for every $(\boldsymbol{x}, t) \in \Omega_k \times I$. In this restricted domain, the dynamical system is reduced to a single-component version:

$$\partial_t \boldsymbol{g}_t(\boldsymbol{x}) = -(\Sigma_k - 2tI)^{-1}(\boldsymbol{g}_t(\boldsymbol{x}) - \boldsymbol{m}_k), \quad (\boldsymbol{x}, t) \in \Omega_k \times I.$$

According to the previous results, we have exactly

$$\boldsymbol{g}_t(\boldsymbol{x}) = \sqrt{I - 2t\Sigma_k^{-1}}(\boldsymbol{x} - \boldsymbol{m}_k) + \boldsymbol{m}_k, \quad (\boldsymbol{x}, t) \in \Omega_k \times I. \qquad ■$$

Appendix E. Proof of (66)

Let $\delta \to 0$. Then, the ridgelet transform of the truncated autoencoder $\mathsf{id}_{r,\delta}$ is given by

$$R_\rho[\mathsf{id}_{r,0}](\boldsymbol{a}, b) = -\frac{A_{m-1}}{2(m+1)} \int_{|p|<r} (r^2 - p^2)^{\frac{m-1}{2}} \left\{ \frac{2}{m-1}p^2 + r^2 \right\} \overline{\rho'(|\boldsymbol{a}|p - b)}\boldsymbol{a}\mathrm{d}p \tag{80}$$

$$\approx -K\boldsymbol{a}\overline{\rho'(-b)}, \tag{81}$$

where $A_{m-1} := \frac{2\pi^{\frac{m-1}{2}}}{\Gamma(\frac{m-1}{2})}$ is the surface area of $\mathbb{S}^{m-1}$, and K is given by (87).

Proof Let $\delta \to 0$. Then, the connecting annulus $\mathbb{B}(0; r+\delta) \setminus \mathbb{B}(0; r)$ vanishes as follows:

$$\begin{aligned}R_\rho[\mathsf{id}_{r,\delta}](\boldsymbol{a}, b) &= -\boldsymbol{a}R_{\rho'}[V_{r,\delta}](\boldsymbol{a}, b) \\ &\to -\boldsymbol{a}\int_{\mathbb{B}^m(r)} \frac{1}{2}|\boldsymbol{x}|^2\overline{\rho'(\boldsymbol{a}\cdot\boldsymbol{x} - b)}\mathrm{d}\boldsymbol{x} \\ &= -\boldsymbol{a}R_{\rho'}[V_{r,0}](\boldsymbol{a}, b).\end{aligned}$$

Hence, we omit considering the annulus.

In the following, we use a spherical coordinate defined by

$$\boldsymbol{u} := \boldsymbol{a}/|\boldsymbol{a}|, \quad \alpha := 1/|\boldsymbol{a}|, \quad \beta := b/|\boldsymbol{a}|,$$

where $\boldsymbol{u} \in \mathbb{S}^{m-1}$ denotes the direction, $\alpha \in \mathbb{R}_+$ denotes the scale, and $\beta \in \mathbb{R}$ denotes the (scaled) shift parameters.

The ridgelet transform in the spherical coordinate (Sonoda and Murata, 2017a) is given by

$$R_\rho f(\boldsymbol{u}/\alpha, \beta/\alpha) = \int_{\mathbb{R}} \mathrm{Rad}[f](\boldsymbol{u}, p)\overline{\rho_\alpha(p-\beta)}\mathrm{d}p,$$

where $\mathrm{Rad}[f](\boldsymbol{u}, p)$ denotes the Radon transform

$$\mathrm{Rad}[f](\boldsymbol{u}, p) \coloneqq \int_{(\mathbb{R}\boldsymbol{u})^\perp} f(p\boldsymbol{u}+\boldsymbol{y})\mathrm{d}\boldsymbol{y}$$

of the function $f \in L^1(\mathbb{R}^m)$ at direction $\boldsymbol{u} \in \mathbb{S}^{m-1}$ and position $p \in \mathbb{R}$, and

$$\rho_\alpha(p) \coloneqq \rho(p/\alpha).$$

The Radon transform $\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p)$ for $|p| < r$ is calculated as follows. Because $V_{r,\delta}$ is a radial function, $\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p)$ does not depend on the direction $\boldsymbol{u}$. Hence, it is sufficient to consider a special case when $(\mathbb{R}\boldsymbol{u})^\perp = \mathbb{R}^{m-1}$. Therefore,

$$\begin{aligned}
\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p) &= \int_{\mathbb{R}^{m-1}} V_{r,0}(p\boldsymbol{u}+y)\mathrm{d}\boldsymbol{y}, \quad \boldsymbol{u} \perp \boldsymbol{y} \\
&= \int_{\mathbb{R}^{m-1}} \frac{1}{2}|p\boldsymbol{u}+\boldsymbol{y}|^2 \mathbf{1}_{\mathbb{B}^m(0;r)}(p\boldsymbol{u}+\boldsymbol{y})\mathrm{d}\boldsymbol{y} \\
&= \frac{1}{2}\int_{\mathbb{B}^{m-1}\left(0;\sqrt{r^2-p^2}\right)} \left\{p^2+|\boldsymbol{y}|^2\right\}\mathrm{d}\boldsymbol{y}, \qquad (82)
\end{aligned}$$

where the third equation follows by the orthogonality $|p\boldsymbol{u}+\boldsymbol{y}|_m^2 = p^2 + |\boldsymbol{y}|_{m-1}^2$ and a geometric consideration as follows:

$$\begin{aligned}
\int_{\mathbb{R}^{m-1}} [\,\cdot\,]\mathbf{1}_{\mathbb{B}^m(0;r)}(p\boldsymbol{u}+\boldsymbol{y})\mathrm{d}\boldsymbol{y} &= \int_{\mathbb{R}^{m-1}} [\,\cdot\,]\mathbf{1}_{\mathbb{B}^m(-p\boldsymbol{u};r)}(\boldsymbol{y})\mathrm{d}\boldsymbol{y} \\
&= \int_{\mathbb{R}^{m-1}\cap\mathbb{B}^m(-p\boldsymbol{u};r)} [\,\cdot\,]\mathrm{d}\boldsymbol{y} \\
&= \int_{\mathbb{B}^{m-1}(0;\sqrt{r^2-p^2})} [\,\cdot\,]\mathrm{d}\boldsymbol{y}.
\end{aligned}$$

The first integral in (82) is calculated as follows:

$$\begin{aligned}
\int_{\mathbb{B}^{m-1}\left(0;\sqrt{r^2-p^2}\right)} p^2\mathrm{d}\boldsymbol{y} &= p^2\,\mathsf{vol}\left[\mathbb{B}^{m-1}(0;\sqrt{r^2-p^2})\right] \\
&= \frac{\pi^{\frac{m-1}{2}}}{2\Gamma\left(\frac{m-1}{2}+1\right)} p^2(r^2-p^2)^{\frac{m-1}{2}}. \qquad (83)
\end{aligned}$$

The second integral in (82) is calculated as follows:

$$\begin{aligned}
\int_{\mathbb{B}^{m-1}\left(0;\sqrt{r^2-p^2}\right)} |\boldsymbol{y}|^2\mathrm{d}\boldsymbol{y} &= \int_{\mathbb{S}^{m-2}}\int_0^{\sqrt{r^2-p^2}} |\rho\omega|^2\rho^{m-2}\mathrm{d}\rho\mathrm{d}\omega \\
&= \int_{\mathbb{S}^{m-2}}\mathrm{d}\omega \int_0^{\sqrt{r^2-p^2}} \rho^m\mathrm{d}\rho \\
&= \frac{\pi^{\frac{m-1}{2}}}{(m+1)\Gamma\left(\frac{m-1}{2}\right)}(r^2-p^2)^{\frac{m+1}{2}}. \qquad (84)
\end{aligned}$$

Hence, by combining the first and second integrals, we have

$$\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p) = \begin{cases} \frac{A_{m-1}}{2(m+1)}(r^2-p^2)^{\frac{m-1}{2}}\left\{\frac{2}{m-1}p^2+r^2\right\} & |p| < r \\ 0 & |p| \geq r. \end{cases} \qquad (85)$$

The ridgelet transform $R_{\rho'}[V_{r,0}]$ is given by

$$R_{\rho'}[V_{r,0}](\boldsymbol{u}/\alpha, \beta/\alpha) = \int_{|p|<r} k(p)\overline{\rho'_\alpha(p-\beta)}\mathrm{d}p, \qquad (86)$$

where we define

$$k(p) \coloneqq \mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p).$$

Recall that $\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p)$ does not depend on the direction $\boldsymbol{u}$; thus, the definition of k is reasonable. According to (85), k is a compactly supported bump function. Consequently, k is summable; thus, the integral

$$K \coloneqq \int_{\mathbb{R}} k(p)\mathrm{d}p \qquad (87)$$

always exists. Recall that the convolution results in smoothing, i.e.,

$$\int_{|p|<r} k(p)\overline{\rho'_\alpha(p-\beta)}\mathrm{d}p \approx K\overline{\rho'_\alpha(-\beta)}. \qquad (88)$$

In summary, we have presented the following:

$$R_\rho[\mathsf{id}_{r,0}](\boldsymbol{a}, b) = -\boldsymbol{a}R_{\rho'}[V_{r,0}](\boldsymbol{a}, b) \approx -K\boldsymbol{a}\overline{\rho'(-b)}. \qquad \blacksquare$$

References

Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data generating distribution. *Journal of Machine Learning Research*, 15(Nov):3743–3773, 2014.

Guillaume Alain, Yoshua Bengio, Li Yao, Jason Yosinski, Eric Thibodeau-Laufer, Saizheng Zhang, and Pascal Vincent. GSNs: Generative stochastic networks. *Information and Inference*, 5(2):210–249, 2016.

Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient Flows in Metric Spaces and in the Space of Probability Measures*. Birkhäuser, 2008.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of The 34th International Conference on Machine Learning*, volume 70, pages 214–223, Sydney, Australia, 2017.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *The 36th International Conference on Machine Learning*, volume 80, pages 244–253, Stockholm, Sweden, 2018.

Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems 27*, pages 2654–2662, Montréal, BC, 2014.

Francis Bach. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 18(19):1–53, 2017a.

Francis Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research*, 18(21):1–38, 2017b.

Andrew R Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

Yoshua Bengio, Nicolas Le Roux, Pascal Vincent, Olivier Delalleau, and Patrice Marcotte. Convex neural networks. In *Advances in Neural Information Processing Systems 18*, pages 123–130, Vancouver, BC, 2006.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems 19*, pages 153–160, Vancouver, BC, 2007.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013a.

Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems 26*, pages 899–907, Lake Tahoe, USA, 2013b.

Yoshua Bengio, Éric Thibodeau-Laufer, Guillaume Alain, and Jason Yosinski. Deep generative stochastic networks trainable by backprop. In *Proceedings of The 31st International Conference on Machine Learning*, volume 32, pages 226–234, Beijing, China, 2014.

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

Yann Brenier. Polar factorization and monotone rearrangement of vector-valued functions. *Communications on Pure and Applied Mathematics*, 44(4):375–417, 1991.

Emmanuel Jean Candès. *Ridgelets: Theory and Applications*. PhD thesis, Standford University, 1998.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems 31*, pages 6572—-6583, Montréal, BC, 2018.

Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in Neural Information Processing Systems 32*, Montréal, BC, 2018.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Proceedings of The 18th International Conference on Artificial Intelligence and Statistics 2015*, volume 38, pages 192–204, San Diego, USA, 2015.

Nadav Cohen, Or Sharir, and Amnon Shashua. On the expressive power of deep learning: A tensor analysis. In *29th Annual Conference on Learning Theory*, volume 49, pages 1–31, 2016.

Yann Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems 27*, pages 2933–2941, Montréal, BC, 2014.

Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *29th Annual Conference on Learning Theory*, volume 49, pages 1–34, 2016.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

Lawrence C. Evans and Ronald F. Gariepy. *Measure Theory and Fine Properties of Functions*. CRC Press, revised edition, 2015.

Edward I. George, Feng Liang, and Xinyi Xu. Improved minimax predictive densities under Kullback-Leibler loss. *Annals of Statistics*, 34(1):78–91, 2006.

Aidan N Gomez, Mengye Ren, Raquel Urtasun, and Roger B. Grosse. The reversible residual network: Backpropagation without storing activations. In *Advances in Neural Information Processing Systems 30*, pages 2214–2224, Long Beach, USA, 2017.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680, Montréal, BC, 2014.

Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):1–22, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, Las Vegas, USA, 2016.

Geoffrey E. Hinton. Connectionist learning procedures. *Artificial Intelligence*, 40(1-3): 185–234, 1989.

Masaaki Imaizumi and Kenji Fukumizu. Deep neural networks learn non-smooth functions effectively. In *Proceedings of The 22nd International Conference on Artificial Intelligence and Statistics 2019*, Okinawa, Japan, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of The 32nd International Conference on Machine Learning*, volume 37, pages 448–456, Lille, France, 2015.

Kenji Kawaguchi. Deep learning without poor local minima. In *Advances in Neural Information Processing Systems 29*, pages 586–594, Barcelona, Spain, 2016.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations 2014*, pages 1–14, Banff, BC, 2014.

Jason M. Klusowski and Andrew R. Barron. Minimax lower bounds for ridge combinations including neural nets. In *2017 IEEE International Symposium on Information Theory*, pages 1376–1380, 2017.

Jason M. Klusowski and Andrew R. Barron. Approximation by combinations of ReLU and squared ReLU ridge functions with ℓ^1 and ℓ^0 controls. *IEEE Transactions on Information Theory*, 64(12):7649–7656, 2018.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105, Lake Tahoe, USA, 2012.

Věra Kůrková. Complexity estimates based on integral transforms induced by computational units. *Neural Networks*, 33:160–167, 2012.

Honglak Lee. *Unsupervised Feature Learning via Sparse Hierarchical Representations*. PhD thesis, Stanford University, 2010.

Qianxiao Li and Shuji Hao. An optimal control approach to deep learning and applications to discrete-weight neural networks. In *Proceedings of The 35th International Conference on Machine Learning*, volume 80, pages 2985–2994, Stockholm, Sweden, 2018.

Henry W. Lin, Max Tegmark, and David Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168(6):1223–1247, 2017.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances in Neural Information Processing Systems 29*, pages 2378–2386, Barcelona, Spain, 2016.

Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *Proceedings of The 35th International Conference on Machine Learning*, volume 80, pages 3276–3285, Stockholm, Sweden, 2018.

Noboru Murata. An integral representation of functions using three-layered betworks and their approximation bounds. *Neural Networks*, 9(6):947–956, 1996.

Behnam Neyshabur. *Implicit Regularization in Deep Learning*. PhD thesis, Toyota Technological Institute at Chicago, 2017.

Quynh Nguyen and Matthias Hein. The loss surface of deep and wide neural networks. In *Proceedings of The 34th International Conference on Machine Learning*, volume 70, pages 2603–2612, Sydney, Australia, 2017.

Atsushi Nitanda and Taiji Suzuki. Functional gradient boosting based on residual network perception. In *Proceedings of The 35th International Conference on Machine Learning*, volume 80, pages 3819–3828, Stockholm, Sweden, 2018.

Kaare Brandt Petersen and Michael Syskind Pedersen. The matrix cookbook, version: November 15, 2012. Technical report, Technical University of Denmark, 2012.

Gabriel Peyré and Marco Cuturi. Computational Optimal Transport. 2018.

Allan Pinkus. Density in approximation theory. *Surveys in Approximation Theory*, 1:1–45, 2005.

Tomaso Poggio, Hrushikesh Mhaskar, Lorenzo Rosasco, Brando Miranda, and Qianli Liao. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14(5):503–519, 2017.

Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag New York, 1996.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of The 32nd International Conference on Machine Learning*, volume 37, pages 1530–1538, Lille, France, 2015.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of The 28th International Conference on Machine Learning*, pages 833–840, Belleview, USA, 2011.

Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. MIT Press, 2012.

Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with ReLU activation function. 2017.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of The 32nd International Conference on Machine Learning*, volume 37, pages 2256–2265, Lille, France, 2015.

Sho Sonoda and Noboru Murata. Sampling hidden parameters from oracle distribution. In *24th International Conference on Artificial Neural Networks*, pages 539–546, Hamburg, Germany, 2014.

Sho Sonoda and Noboru Murata. Neural network with unbounded activation functions is universal approximator. *Applied and Computational Harmonic Analysis*, 43(2):233–268, 2017a.

Sho Sonoda and Noboru Murata. Double continuum limit of deep neural networks. In *ICML 2017 Workshop on Principled Approaches to Deep Learning*, pages 1–5, Sydney, Australia, 2017b.

Sho Sonoda and Noboru Murata. Transportation analysis of denoising autoencoders: A novel method for analyzing deep neural networks. In *NIPS 2017 Workshop on Optimal Transport & Machine Learning*, pages 1–10, Long Beach, USA, 2017c.

Sho Sonoda, Isao Ishikawa, Masahiro Ikeda, Kei Hagihara, Yoshihiro Sawano, Takuo Matsubara, and Noboru Murata. Integral representation of shallow neural network that attains the global minimum. 2018.

Daniel Soudry and Yair Carmon. No bad local minima: Data independent training error guarantees for multilayer neural networks. 2016.

Taiji Suzuki. Fast generalization error bound of deep learning from a kernel perspective. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, volume 84, pages 1397–1406, Playa Blanca, Lanzarote, Canary Islands, 2018.

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. DeepFace: Closing the gap to human-level performance in face verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, Columbus, USA, 2014.

Asuka Takatsu. Wasserstein geometry of Gaussian measures. *Osaka Journal of Mathematics*, 48(4):1005–1026, 2011.

Matus Telgarsky. Benefits of depth in neural networks. In *29th Annual Conference on Learning Theory*, pages 1–23, 2016.

Alexandre B Tsybakov. *Introduction to Nonparametric Estimation*. Springer-Verlag New York, 2009.

Vladimir Naumovich Vapnik. *Statistical Learning Theory*. Wiley, 1998.

Cédric Villani. *Optimal Transport: Old and New*. Springer-Verlag Berlin Heidelberg, 2009.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of The 25th International Conference on Machine Learning*, pages 1096–1103, Helsinki, Finland, 2008.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

Grace Wahba. *Spline Models for Observational Data*. SIAM, 1990.

Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833, Zurich, Switzerland, 2014.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations 2017*, Toulon, France, 2017.

Ruiyi Zhang, Changyou Chen, Chunyuan Li, and Lawrence Carin. Policy optimization as Wasserstein gradient flows. In *Proceedings of The 35th International Conference on Machine Learning*, volume 80, pages 5737–5746, Stockholm, Sweden, 2018.

Journal of Machine Learning Research 20 (2019) 1-44 Submitted 11/16; Revised 1/19; Published 1/19

Parsimonious Online Learning with Kernels via Sparse Projections in Function Space*

Alec Koppel AKOPPEL@SEAS.UPENN.EDU
Computational and Information Sciences Directorate
U.S. Army Research Laboratory
Adelphi, MD 20783, USA

Garrett Warnell GARRETT.A.WARNELL.CIV@MAIL.MIL
Computational and Information Sciences Directorate
U.S. Army Research Laboratory
Adelphi, MD 20783, USA

Ethan Stump ETHAN.A.STUMP2.CIV@MAIL.MIL
Computational and Information Sciences Directorate
U.S. Army Research Laboratory
Adelphi, MD 20783, USA

Alejandro Ribeiro ARIBEIRO@SEAS.UPENN.EDU
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA 19104, USA

Editor: Le Song

Abstract

Despite their attractiveness, popular perception is that techniques for nonparametric function approximation do not scale to streaming data due to an intractable growth in the amount of storage they require. To solve this problem in a memory-affordable way, we propose an online technique based on functional stochastic gradient descent in tandem with supervised sparsification based on greedy function subspace projections. The method, called *parsimonious online learning with kernels* (POLK), provides a controllable tradeoff between its solution accuracy and the amount of memory it requires. We derive conditions under which the generated function sequence converges almost surely to the optimal function, and we establish that the memory requirement remains finite. We evaluate POLK for kernel multi-class logistic regression and kernel hinge-loss classification on three canonical data sets: a synthetic Gaussian mixture model, the MNIST hand-written digits, and the Brodatz texture database. On all three tasks, we observe a favorable trade-off of objective function evaluation, classification performance, and complexity of the nonparametric regressor extracted by the proposed method.

Keywords: kernel methods, online learning, stochastic optimization, supervised learning, orthogonal matching pursuit, nonparametric regression

*. This work in this paper is supported by NSF CCF-1017454, NSF CCF-0952867, ONR N00014-12-1-0997, ARL MAST CTA, and ASEE SMART. Part of the results in this paper appeared in Koppel et al. (2017).

©2019 Alec Koppel, Garrett Warnell, Ethan Stump and Alejandro Ribeiro.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/16-585.html.

1. Introduction

Reproducing kernel Hilbert spaces (RKHS) provide the ability to approximate functions using nonparametric functional representations. Although the structure of the space is determined by the choice of kernel, the set of functions that can be represented is still sufficiently rich so as to permit close approximation of a large class of functions. This resulting expressive power makes RKHS an appealing choice in many learning problems where we want to estimate an unknown function that is specified as optimal with respect to some empirical risk. When learning these optimal function representations in a RKHS, the representer theorem is used to transform the search over functions into a search over parameters, where the number of parameters grows with each new observation that is processed (Wheeden et al., 1977; Norkin and Keyzer, 2009). This growth is what endows the representation with expressive power. However, this growth also results in function descriptions that are as complex as the number of processed observations, and, more importantly, in training algorithms that exhibit a cost per iteration that grows with each new iterate (Pontil et al., 2005; Kivinen et al., 2004). The resulting unmanageable training cost renders RKHS learning approaches inefficient for large data sets and outright inapplicable in streaming applications. This is a well-known limitation which has motivated the development of several heuristics to reduce the growth in complexity. These heuristics typically result in suboptimal functional approximations (Richard et al., 2009).

This paper proposes a new technique for learning nonparametric function approximations in a RKHS that respects optimality and ameliorates the complexity issues described above. We accomplish this by: (i) shifting the goal from that of finding an approximation that is optimal to that of finding an approximation that is optimal within a class of parsimonious (sparse) kernel representations; (ii) designing a training method that follows a trajectory of intermediate representations that are also parsimonious. The proposed technique, *parsimonious online learning with kernels* (POLK), provides a controllable tradeoff between complexity and optimality and we provide theoretical guarantees that neither factor becomes untenable.

Formally, we propose solving expected risk minimization problems, where the goal is to learn a regressor that minimizes a loss function quantifying the merit of a statistical model averaged over a data set. We focus on the case when the number of training examples, N, is either very large, or the training examples arrive sequentially. Further, we assume that these input-output examples, $(\mathbf{x}_n, \mathbf{y}_n)$, are i.i.d. realizations drawn from a stationary joint distribution over the random pair $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}$. This problem class is popular in many fields and particularly ubiquitous in text (Sampson et al., 1990), image (Mairal et al., 2007), and genomic (Taşan et al., 2014) processing. Here, we consider finding regressors that are not vector valued parameters, but rather functions $f \in \mathcal{H}$ in a hypothesized function class $\mathcal{H}$. This function estimation task allows one to learn nonlinear statistical models and is known to yield better results in applications where linearity of a given statistical model is overly restrictive such as computer vision and object recognition (Mukherjee and Nayar, 1996; Li et al., 2014). The adequacy of the regressor function f is evaluated by the convex loss function $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that quantifies the merit of the estimator $f(\mathbf{x})$ evaluated at feature vector $\mathbf{x}$. This loss is averaged over all possible training examples to define the statistical loss $L(f) := \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), y)]$, which we combine with a Tikhonov regularizer to

construct the regularized loss $R(f) := \operatorname{argmin}_{f\in\mathcal{H}} L(f) + (\lambda/2)\|f\|^2_{\mathcal{H}}$ (Shalev-Shwartz et al., 2010; Evgeniou et al., 2000). We then define the optimal function as

$$f^* = \operatorname*{argmin}_{f\in\mathcal{H}} R(f) := \operatorname*{argmin}_{f\in\mathcal{H}} \mathbb{E}_{\mathbf{x},\mathbf{y}}\Big[\ell(f(\mathbf{x}), y)\Big] + \frac{\lambda}{2}\|f\|^2_{\mathcal{H}} \tag{1}$$

The optimization problem in (1) is intractable in general. However, when $\mathcal{H}$ is equipped with a *reproducing kernel* $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, a nonparametric function estimation problem of the form (1) may be reduced to a parametric form via the representer theorem (Wheeden et al., 1977; Norkin and Keyzer, 2009). This theorem states that the optimal argument of (1) is in the span of kernel functions that are centered at points in the given training data set, and it reduces the problem to that of determining the N coefficients of the resulting linear combination of kernels (Section 2). This results in a function description that is data driven and flexible, alas very complex. As we consider problems with larger training sets, the representation of f requires a growing number of kernels (Norkin and Keyzer, 2009). In the case of streaming applications this number would grow unbounded and the kernel matrix as well as the coefficient vector would grow to infinite dimension and an infinite amount of memory would be required to represent f. It is therefore customary to reduce this complexity by forgetting training points or otherwise requiring that f^* admit a parsimonious representation in terms of a sparse subset of kernels. This overcomes the difficulties associated with a representation of unmanageable complexity but a steeper difficulty is the *determination* of this optimal parsimonious representation as we explain in the following section

1.1. Context

To understand the challenge in determining optimal parsimonious representations, recall that kernel optimization methods borrow techniques from vector valued (i.e., without the use of kernels) stochastic optimization in the sense that they seek to optimize (1) by replacing the descent direction of the objective with that of a stochastic estimate (Bottou, 1998; Robbins and Monro, 1951). Stochastic optimization is well understood in vector valued problems to the extent that recent efforts are concerned with improving convergence properties through the use of variance reduction (Schmidt et al., 2013; Johnson and Zhang, 2013; Defazio et al., 2014), or stochastic approximations of Newton steps (Schraudolph et al., 2007; Bordes et al., 2009; Mokhtari and Ribeiro, 2014, 2015). Stochastic optimization in kernel spaces, however, exhibits two peculiarities that make it more challenging:

i. The implementation of stochastic methods for expected risk minimization in a RKHS requires storage of kernel matrices and weight vectors that together are cubic in the iteration index. This is true even if we require that the solution f^* admit a sparse representation because, while it may be true that the asymptotic solution admits a sparse representation, the intermediate iterates are not necessarily sparse; see, e.g., Kivinen et al. (2004).

ii. The problem in (i) makes it necessary to use sparse approximations of descent directions. However, these sparse approximates are not guaranteed to be valid stochastic descent directions. Consequently, there are no guarantees that a path of sparse approximation learns the optimal sparse approximation.

Issue (i) is a key point of departure between kernel stochastic optimization and its vector valued counterpart. It implies that redefining f^* to encourage sparsity may make it easier to work with the RKHS representation after it has been learnt. However, the stochastic gradient iterates that need to be computed to find such representation have a complexity that grows with the order of the iteration index (Kivinen et al., 2004), precluding the use of sparsity-encouraging penalties (Fu, 1998). Works on stochastic optimization in a RKHS have variously ignored the intractable growth of the parametric representation of $f \in \mathcal{H}$ (Ying and Zhou, 2006; Liu et al., 2008; Pontil et al., 2005; Dieuleveut and Bach, 2014), addressed it indirectly through probabilistic (Rahimi and Recht, 2008, 2009; Dai et al., 2014) or computational (Williams and Seeger, 2001; Zhang et al., 2008) approximations of the kernel, or have augmented the learned function to limit the memory issues associated with kernelization using online sparsification procedures. These approaches focus on limiting the growth of the kernel dictionary through forgetting factors (Kivinen et al., 2004), random dropping (Zhang et al., 2013; Le et al., 2016b), compressive sensing (Engel et al., 2004; Richard et al., 2009; Honeine, 2012), greedy search (Bordes et al., 2005) and projection (Zhao et al., 2012; Zhao and Hoi, 2012; Wang et al., 2012). These approaches overcome Issue (i) but they do so at the cost of dropping optimality [cf. Issue (ii)]. This is because these sparsification techniques introduce a bias in the stochastic gradient which nullifies convergence guarantees, and empirically leads to instability.

Past works that have considered *supervised* sparsification (addressing issues (i)-(ii)) have only been developed for special cases such as online support vector machines (SVM) (Wang et al., 2012), off-line logistic regression Zhu and Hastie (2005), and off-line SVM (Joachims and Yu, 2009). The works perhaps most similar to ours, but developed only for SVM (Wang et al., 2012)) fixes the number of kernel dictionary elements, or the model order, in advance rather tuning the kernel dictionary to guarantee stochastic descent, i.e. determining which kernel dictionary elements are most important for representing f^*. Further, the analysis of the resulting bias induced by sparsification requires overly restrictive assumptions and is conducted in terms of time-average objective sub-optimality, a looser criterion than almost sure convergence. For specialized classes of loss functions, the bias of the descent direction induced by unsupervised sparsification techniques using random sub-sampling does not prevent the derivation of bounds on the time-average sub-optimality (regret) (Zhang et al., 2013); however, this analysis omits important cases such as support vector machines and kernel ridge regression.

1.2. Contributions

In this work, we build upon past works which combine functional generalizations of first-order stochastic optimization methods operating in tandem with supervised sparsification. In particular, descending along the gradient of the objective in (1) is intractable when the sample size N is not necessarily finite, and thus stochastic methods are necessary. In Section 3, we build upon Kivinen et al. (2004) in deriving the generalization of stochastic gradient method called functional SGD (Section 3.1). The complexity of this online functional iterative optimization is on the order of the iteration index, a complicating factor of kernel methods which is untenable for streaming settings.

Thus, we project the FSGD iterates onto sparse subspaces, which, rather than being fixed independent of the problem (Zhu et al., 2009), are adaptively constructed from the span of a small number of kernel dictionary elements (Section 3.2). To find these sparse subspaces of the RKHS, we make use of greedy sampling methods based on matching pursuit (Pati et al., 1993). The use of this technique is motivated by: (i) The fact that kernel matrices induced by arbitrary data streams will not, in general, satisfy requisite conditions for methods that enforce sparsity through convex relaxation (Candes, 2008); (ii) That having function iterates that exhibit small model order is of greater importance than exact recovery since SGD iterates are not the goal signal but just a noisy stepping stone to the optimal f^*. Therefore, we construct these instantaneous sparse subspaces by making use of kernel orthogonal matching pursuit (Vincent and Bengio, 2002), a greedy search routine which, given a function and an approximation budget ϵ, returns its a sparse approximation and guarantees its output to be in a specific Hilbert-norm neighborhood of its function input. Greedy sampling methods appear in prior works for constructing RKHS subspaces in recursive mini-batching procedures (Keerthi et al., 2006) or with FSGD (Zhao et al., 2012), but in a manner that avoids addressing how to merge greedy compression with obtaining optimal function representations.

To guarantee stochastic descent, we tie the size of the error neighborhood induced by sparse projections to the magnitude of the functional stochastic gradient and other problem parameters, thereby keeping only those kernel dictionary elements necessary for convergence (Section 4). The result is that we are able to conduct stochastic gradient descent using only sparse projections of the stochastic gradient, maintaining a convergence path of moderate complexity towards the optimal f^* (1). When the data and target domains ($\mathcal{X}$ and $\mathcal{Y}$, respectively) are compact, for a certain approximation budget depending on the stochastic gradient algorithm step-size, we show that the sparse stochastically projected FSGD sequence still converges almost surely to the optimum of (5) under both attenuating and constant learning rate schemes. Moreover, the model order of this sequence remains finite for a given choice of constant step-size and approximation budget, and is, in the worst-case, comparable to the covering number of the data domain (Zhou, 2002; Pontil, 2003).

In Section 5 we present numerical results on synthetic and empirical data for large-scale kernelized supervised learning tasks. We observe stable convergence behavior of POLK comparable to vector-valued first-order stochastic methods in terms of objective function evaluation, punctuated by a state of the art trade-off between test set error and number of samples processed. Further, the proposed method reduces the complexity of training kernel regressors by orders of magnitude. In Section 6 we discuss our main findings. In particular, we suggest that there is a path forward for kernel methods as an alternative to neural networks that provides a more interpretable mechanism for inference with nonlinear statistical models and that one may achieve high generalization capability without losing convexity, an essential component for efficient training.

Relative to (Koppel et al., 2017), this paper provides detailed derivations of the algorithms and proofs to all theorems, including auxiliary and technical lemmas. Moreover, a thorough accounting of computational complexity and extensive experimentation with several state of the art alternatives, both offline and online, are provided.

2. Statistical Optimization in Reproducing Kernel Hilbert Spaces

Supervised learning is often formulated as an optimization problem that computes a set of parameters $\theta \in \Theta$ to minimize the average of a loss function $l : \Theta \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ for training examples $(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{X} \times \mathcal{Y}$. When the number of training examples N is finite, this goal is referred to as *empirical risk minimization* (Tewari and Bartlett, 2014), and may be solved using batch optimization techniques. The optimal $\boldsymbol{\theta}$ is the one that minimizes the regularized average loss, $\tilde{R}(\theta; \{\mathbf{x}_n, y_n\}_{n=1}^N) = \frac{1}{N}\sum_{n=1}^N l(\boldsymbol{\theta}; (\mathbf{x}_n, y_n))$, over the set of training data $\mathcal{S} = \{\mathbf{x}_n, y_n\}_{n=1}^N$, i.e.,

$$\theta^* = \operatorname*{argmin}_{\boldsymbol{\theta}\in\Theta} \tilde{R}(\boldsymbol{\theta}; \mathcal{S}) = \operatorname*{argmin}_{\boldsymbol{\theta}\in\Theta} \frac{1}{N}\sum_{n=1}^N l(\boldsymbol{\theta}; \mathbf{x}_n, y_n) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2 \,. \tag{2}$$

We focus on the case when the inputs are vectors $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^p$ and the target domain $\mathcal{Y} \subseteq \{0, 1\}$ in the case of classification or $\mathcal{Y} \subseteq \mathbb{R}$ in the case of regression.

2.1. Supervised Kernel Learning

In the case of supervised kernel learning (Müller et al., 2013; Li et al., 2014), Θ is taken to be a Hilbert space, denoted here as $\mathcal{H}$. Elements of $\mathcal{H}$ are *functions*, $f : \mathcal{X} \to \mathcal{Y}$, that admit a representation in terms of elements of $\mathcal{X}$ when $\mathcal{H}$ has a special structure. In particular, equip $\mathcal{H}$ with a unique *kernel function*, $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, such that:

$$(i)\ \langle f, \kappa(\mathbf{x}, \cdot)\rangle_{\mathcal{H}} = f(\mathbf{x}) \quad \text{for all } \mathbf{x} \in \mathcal{X}\,, \qquad (ii)\ \mathcal{H} = \overline{\operatorname{span}\{\kappa(\mathbf{x}, \cdot)\}} \quad \text{for all } \mathbf{x} \in \mathcal{X}\,. \tag{3}$$

where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the Hilbert inner product for $\mathcal{H}$. We further assume that the kernel is positive semidefinite, i.e. $\kappa(\mathbf{x}, \mathbf{x}') \geq 0$ for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. Function spaces with this structure are called reproducing kernel Hilbert spaces (RKHS).

In (3), property (i) is called the reproducing property of the kernel, and is a consequence of the Riesz Representation Theorem (Wheeden et al., 1977). Replacing f by $\kappa(\mathbf{x}', \cdot)$ in (3) (i) yields the expression $\langle \kappa(\mathbf{x}', \cdot), \kappa(\mathbf{x}, \cdot)\rangle_{\mathcal{H}} = \kappa(\mathbf{x}, \mathbf{x}')$, which is the origin of the term "reproducing kernel." This property provides a practical means by which to access a nonlinear transformation of the input space $\mathcal{X}$. Specifically, denote by $\phi(\cdot)$ a nonlinear map of the feature space that assigns to each $\mathbf{x}$ the kernel function $\kappa(\cdot, \mathbf{x})$. Then the reproducing property of the kernel allows us to write the inner product of the image of distinct feature vectors $\mathbf{x}$ and $\mathbf{x}'$ under the map ϕ in terms of kernel evaluations only: $\langle \phi(\mathbf{x}), \phi(\mathbf{x}')\rangle_{\mathcal{H}} = \kappa(\mathbf{x}, \mathbf{x}')$. This is commonly referred to as the *kernel trick*, and it provides a computationally efficient tool for learning nonlinear functions.

Moreover, property (3) (ii) states that any function $f \in \mathcal{H}$ may be written as a linear combination of kernel evaluations. For kernelized and regularized empirical risk minimization, the Representer Theorem (Kimeldorf and Wahba, 1971; Schölkopf et al., 2001) establishes that the optimal f in the hypothesis function class $\mathcal{H}$ may be written as an expansion of kernel evaluations *only* at elements of the training set as

$$f(\mathbf{x}) = \sum_{n=1}^N w_n \kappa(\mathbf{x}_n, \mathbf{x}) \,. \tag{4}$$

where $\mathbf{w} = [w_1, \cdots, w_N]^T \in \mathbb{R}^N$ denotes a set of weights. The upper summand index N in (4) is henceforth referred to as the model order. Common choices κ include the polynomial kernel and the radial basis kernel, i.e., $\kappa(\mathbf{x}, \mathbf{x}') = \left(\mathbf{x}^T\mathbf{x}' + b\right)^c$ and $\kappa(\mathbf{x}, \mathbf{x}') = \exp\left\{-\frac{\|\mathbf{x}-\mathbf{x}'\|_2^2}{2c^2}\right\}$, respectively, where $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$.

We may now formulate the kernel variant of the empirical risk minimization problem as the one that minimizes the loss functional $L : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ plus a complexity-reducing penalty. The loss functional L may be written as an average over instantaneous losses $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, each of which penalizes the average deviation between $f(\mathbf{x}_n)$ and the associated output y_n over the training set $\mathcal{S}$. We denote the data loss and complexity loss as $R : \mathcal{H} \to \mathbb{R}$, and consider the problem

$$f^* = \operatorname*{argmin}_{f\in\mathcal{H}} R(f;\mathcal{S}) = \operatorname*{argmin}_{f\in\mathcal{H}} \frac{1}{N}\sum_{n=1}^{N} \ell(f(\mathbf{x}_n), y_n) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \,. \tag{5}$$

The above problem, referred to as Tikhonov regularization (Evgeniou et al., 2000), is one in which we aim to learn a general nonlinear relationship between $\mathbf{x}_n$ and y_n through a function f. Throughout, we assume ℓ is convex with respect to its first argument $f(\mathbf{x})$. By substituting the Representer Theorem expansion in (4) into (5), the optimization problem amounts to finding an optimal set of coefficients $\mathbf{w}$ as

$$\begin{aligned} f^* &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^N} \frac{1}{N}\sum_{n=1}^{N} \ell(\sum_{m=1}^{N} w_m\kappa(\mathbf{x}_m, \mathbf{x}_n), y_n) + \frac{\lambda}{2}\|\sum_{n,m=1}^{N} w_n w_m \kappa(\mathbf{x}_m, \mathbf{x}_n)\|_{\mathcal{H}}^2 \\ &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^N} \frac{1}{N}\sum_{n=1}^{N} \ell(\mathbf{w}^T\boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_n), y_n) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{K}_{\mathbf{X},\mathbf{X}}\mathbf{w} \end{aligned} \tag{6}$$

where we have defined the Gram matrix (variously referred to as the *kernel matrix*) $\mathbf{K}_{\mathbf{X},\mathbf{X}} \in \mathbb{R}^{N\times N}$, with entries given by the kernel evaluations between $\mathbf{x}_m$ and $\mathbf{x}_n$ as $[\mathbf{K}_{\mathbf{X},\mathbf{X}}]_{m,n} = \kappa(\mathbf{x}_m, \mathbf{x}_n)$. We further define the vector of kernel evaluations $\boldsymbol{\kappa}_{\mathbf{X}}(\cdot) = [\kappa(\mathbf{x}_1, \cdot) \dots \kappa(\mathbf{x}_N, \cdot)]^T$, which are related to the kernel matrix as $\mathbf{K}_{\mathbf{X},\mathbf{X}} = [\boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_1) \dots \boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_N)]$. The dictionary of training points associated with the kernel matrix is defined as $\mathbf{X} = [\mathbf{x}_1, \ \dots \ , \mathbf{x}_N]$.

Observe that by exploiting the Representer Theorem, we transform a nonparametric infinite dimensional optimization problem in $\mathcal{H}$ (5) into a finite N-dimensional parametric problem (6). Thus, for empirical risk minimization, the RKHS provides a principled framework to solve nonparametric regression problems as via search over $\mathbb{R}^N$ for an optimal set of coefficients. A motivating example is presented next to clarify the setting of supervised kernel learning.

Example 1 *(Kernel Logistic Regression)* Consider the case of *kernel logistic regression* (KLR), with feature vectors $\mathbf{x}_n \in \mathcal{X} \subseteq \mathbb{R}^p$ and binary class labels $y_n \in \{0, 1\}$. We seek to learn a function $f \in \mathcal{H}$ that allows us to best approximate the distribution of an unknown class label given a training example $\mathbf{x}$ under the assumed model

$$\mathbb{P}\left(y = 0 \mid \mathbf{x}\right) = \frac{\exp\left\{f(\mathbf{x})\right\}}{1 + \exp\left\{f(\mathbf{x})\right\}} \,. \tag{7}$$

In classical logistic regression, we assume that f is linear, i.e., $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + b$. In KLR, on the other hand, we instead seek a nonlinear function of the form given in (4). By making use of (7) and (4), we may formulate a maximum-likelihood estimation (MLE) problem to find the optimal function f on the basis of $\mathcal{S}$ by solving for the $\mathbf{w}$ that maximizes the λ-regularized average negative log likelihood over $\mathcal{S}$, i.e.,

$$\begin{aligned} f^* &= \operatorname*{argmin}_{f\in\mathcal{H}} \frac{1}{N}\sum_{n=1}^{N}\left[-\log\mathbb{P}(y = y_n \mid \mathbf{x} = \mathbf{x}_n) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2\right] \\ &= \operatorname*{argmin}_{f\in\mathcal{H}} \frac{1}{N}\sum_{n=1}^{N}\left[\log\left(1 + \exp\{f(\mathbf{x}_n)\}\right) - \mathbb{1}(y_n = 1) - f(\mathbf{x}_n)\mathbb{1}(y_n = 0) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2\right] \\ &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^N} \frac{1}{N}\sum_{n=1}^{N}\left[\log\left(1+\exp\{\mathbf{w}^T\boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_n)\}\right) - \mathbb{1}(y_n = 1) - \mathbf{w}^T\boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_n)\mathbb{1}(y_n = 0) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{K}_{\mathbf{X},\mathbf{X}}\mathbf{w}\right] , \end{aligned} \tag{8}$$

where $\mathbb{1}(\cdot)$ represents the indicator function. Solving (8) amounts to finding a function f that, given a feature vector $\mathbf{x}$ and the model outlined by (7), best represents the class-conditional probabilities that the corresponding label y is either 0 or 1.

2.2. Online Kernel Learning

The goal of this paper is to solve problems of the form (5) when training examples $(\mathbf{x}_n, \mathbf{y}_n)$ either become sequentially available or their total number is not necessarily finite. To do so, we consider the case where $(\mathbf{x}_n, \mathbf{y}_n)$ are independent realizations from a stationary joint distribution of the random pair $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}$ (Slavakis et al., 2013). In this case, the objective in (5) may be written as an expectation over this random pair as

$$\begin{aligned} f^* = \operatorname*{argmin}_{f\in\mathcal{H}} R(f) :&= \operatorname*{argmin}_{f\in\mathcal{H}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), y)] + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \\ &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^{\mathcal{I}}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\sum_{n\in\mathcal{I}} w_n\kappa(\mathbf{x}_n, \mathbf{x}), y)] + \frac{\lambda}{2}\|\sum_{n,m\in\mathcal{I}} w_n w_m\kappa(\mathbf{x}_m, \mathbf{x}_n)\|_{\mathcal{H}}^2 \,. \end{aligned} \tag{9}$$

where we define the average loss as $L(f) := \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), y)]$. In the second equality in (1), we substitute in the expansion of f given by the Representer Theorem generalized to the infinite sample-size case established in (Norkin and Keyzer, 2009), with $\mathcal{I}$ as some countably infinite indexing set.

3. Algorithm Development

We turn to deriving an algorithmic solution to the kernelized expected risk minimization problem stated in (1). To do so, two complexity bottlenecks must be overcome. The first is that in order to develop a numerical optimization scheme such as gradient descent, we must compute the functional gradient (Frechét derivative) of the expected risk $L(f)$ with respect to f, which requires infinitely many realizations of the random pair $(\mathbf{x}, y)$. This bottleneck is handled via stochastic approximation, as detailed in Section 3.1. The second issue is that when making use of the stochastic gradient method in the RKHS setting,

the resulting parametric updates require memory storage whose complexity is cubic in the iteration index (the curse of kernelization), which rapidly becomes unaffordable. To alleviate this memory explosion, we introduce our sparse stochastic projection scheme based upon kernel orthogonal matching pursuit in Section 3.2.

3.1. Functional Stochastic Gradient Descent

Following Kivinen et al. (2004), we derive the generalization of the stochastic gradient method for the RKHS setting. The resulting procedure is referred to as *functional stochastic gradient descent* (FSGD). First, given an independent realization $(\mathbf{x}_t, y_t)$ of the random pair $(\mathbf{x}, y)$, we compute the stochastic functional gradient (Frechét derivative) of $L(f)$, stated as

$$\nabla_f \ell(f(\mathbf{x}_t), y_t)(\cdot) = \frac{\partial \ell(f(\mathbf{x}_t), y_t)}{\partial f(\mathbf{x}_t)} \frac{\partial f(\mathbf{x}_t)}{\partial f}(\cdot) \tag{10}$$

where we have applied the chain rule. Now, define the short-hand notation $\ell'(f(\mathbf{x}_t), y_t) := \partial \ell(f(\mathbf{x}_t), y_t)/\partial f(\mathbf{x}_t)$ for the derivative of $\ell(\mathbf{f}(\mathbf{x}_t), y_t)$ with respect to its first scalar argument $f(\mathbf{x}_t)$ evaluated at $\mathbf{x}_t$. To evaluate the second term on the right-hand side of (10), differentiate both sides of the expression defining the reproducing property of the kernel [cf. (3)(i)] with respect to f to obtain

$$\frac{\partial f(\mathbf{x}_t)}{\partial f} = \frac{\partial \langle f, \kappa(\mathbf{x}_t, \cdot) \rangle_{\mathcal{H}}}{\partial f} = \kappa(\mathbf{x}_t, \cdot) \tag{11}$$

With this computation in hand, we present the stochastic gradient method for the kernelized λ-regularized expected risk minimization problem in (1) as

$$f_{t+1} = (1 - \eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t) = (1 - \eta_t \lambda) f_t - \eta_t \ell'(f_t(\mathbf{x}_t), y_t) \kappa(\mathbf{x}_t, \cdot) \, , \tag{12}$$

where $\eta_t > 0$ is an algorithm step-size either chosen as diminishing with $\mathcal{O}(1/t)$ or a small constant – see Section 4. We further require that, given $\lambda > 0$, the step-size satisfies $\eta_t < 1/\lambda$ and the sequence is initialized as $f_0 = 0 \in \mathcal{H}$. Given this initialization, by making use of the Representer Theorem (4), at time t, the function f_t may be expressed as an expansion in terms of feature vectors $\mathbf{x}_t$ observed thus far as

$$f_t(\mathbf{x}) = \sum_{n=1}^{t-1} w_n \kappa(\mathbf{x}_n, \mathbf{x}) = \mathbf{w}_t^T \boldsymbol{\kappa}_{\mathbf{X}_t}(\mathbf{x}) \, . \tag{13}$$

On the right-hand side of (13) we have introduced the notation $\mathbf{X}_t = [\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}] \in \mathbb{R}^{p \times (t-1)}$ and $\boldsymbol{\kappa}_{\mathbf{X}_t}(\cdot) = [\kappa(\mathbf{x}_1, \cdot), \ \ldots \ , \kappa(\mathbf{x}_{t-1}, \cdot)]^T$. Moreover, observe that the kernel expansion in (13), taken together with the functional update (12), yields the fact that performing the stochastic gradient method in $\mathcal{H}$ amounts to the following parametric updates on the kernel dictionary $\mathbf{X}$ and coefficient vector $\mathbf{w}$:

$$\mathbf{X}_{t+1} = [\mathbf{X}_t, \ \ \mathbf{x}_t], \quad \mathbf{w}_{t+1} = [(1 - \eta_t \lambda)\mathbf{w}_t, \ \ -\eta_t \ell'(f_t(\mathbf{x}_t), y_t)] \, , \tag{14}$$

Observe that this update causes $\mathbf{X}_{t+1}$ to have one more column than $\mathbf{X}_t$. We define the *model order* as number of data points M_t in the dictionary at time t (the number of columns of $\mathbf{X}_t$). FSGD is such that $M_t = t - 1$, and hence grows unbounded with iteration index t.

3.2. Model Order Control via Stochastic Projection

To mitigate the model order issue described above, we shall generate an approximate sequence of functions by orthogonally projecting functional stochastic gradient updates onto subspaces $\mathcal{H}_{\mathbf{D}} \subseteq \mathcal{H}$ that consist only of functions that can be represented using some dictionary $\mathbf{D} = [\mathbf{d}_1, \ \ldots, \ \mathbf{d}_M] \in \mathbb{R}^{p \times M}$, i.e., $\mathcal{H}_{\mathbf{D}} = \{f \ : \ f(\cdot) = \sum_{n=1}^{M} w_n \kappa(\mathbf{d}_n, \cdot) = \mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\} = \text{span}\{\kappa(\mathbf{d}_n, \cdot)\}_{n=1}^{M}$. For convenience we have defined $[\boldsymbol{\kappa}_{\mathbf{D}}(\cdot) = \kappa(\mathbf{d}_1, \cdot) \ldots \kappa(\mathbf{d}_M, \cdot)]$, and $\mathbf{K}_{\mathbf{D},\mathbf{D}}$ as the resulting kernel matrix from this dictionary. We will enforce parsimony in function representation by selecting dictionaries $\mathbf{D}$ that $M_t \ll t$.

We first show that, by selecting $\mathbf{D} = \mathbf{X}_{t+1}$ at each iteration, the sequence (12) derived in Section 3.1 may be interpreted as carrying out a sequence of orthogonal projections. To see this, rewrite (12) as the quadratic minimization

$$\begin{aligned} f_{t+1} &= \operatorname*{argmin}_{f \in \mathcal{H}} \left\| f - \Big((1 - \eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t) \Big) \right\|_{\mathcal{H}}^2 \\ &= \operatorname*{argmin}_{f \in \mathcal{H}_{\mathbf{X}_{t+1}}} \left\| f - \Big((1 - \eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t) \Big) \right\|_{\mathcal{H}}^2 , \end{aligned} \tag{15}$$

where the first equality in (15) comes from ignoring constant terms which vanish upon differentiation with respect to f, and the second comes from observing that f_{t+1} can be represented using only the points $\mathbf{X}_{t+1}$, using (14). Notice now that (15) expresses f_{t+1} as the orthogonal projection of the update $(1 - \eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t)$ onto the subspace defined by dictionary $\mathbf{X}_{t+1}$.

Rather than select dictionary $\mathbf{D} = \mathbf{X}_{t+1}$, we propose instead to select a different dictionary, $\mathbf{D} = \mathbf{D}_{t+1}$, which is extracted from the data points observed thus far, at each iteration. The process by which we select $\mathbf{D}_{t+1}$ will be discussed shortly, but is of dimension $p \times M_{t+1}$, with $M_{t+1} \ll t$. As a result, we shall generate a function sequence f_t that differs from the functional stochastic gradient method presented in Section 3.1. The function f_{t+1} is parameterized by dictionary $\mathbf{D}_{t+1}$ and weight vector $\mathbf{w}_{t+1}$. We denote columns of $\mathbf{D}_{t+1}$ as $\mathbf{d}_n$ for $n = 1, \ldots, M_{t+1}$, where the time index is dropped for notational clarity but may be inferred from the context.

To be specific, we propose replacing the update (15) in which the dictionary grows at each iteration by the stochastic projection of the functional stochastic gradient sequence onto the subspace $\mathcal{H}_{\mathbf{D}_{t+1}} = \text{span}\{\kappa(\mathbf{d}_n, \cdot)\}_{n=1}^{M_{t+1}}$ as

$$\begin{aligned} f_{t+1} &= \operatorname*{argmin}_{f \in \mathcal{H}_{\mathbf{D}_{t+1}}} \left\| f - \Big((1 - \eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t) \Big) \right\|_{\mathcal{H}}^2 \\ &:= \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}} \Big[(1 - \eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t) \Big] . \end{aligned} \tag{16}$$

where we define the projection operator $\mathcal{P}$ onto subspace $\mathcal{H}_{\mathbf{D}_{t+1}} \subset \mathcal{H}$ by the update (16).

Coefficient update The update (16), for a fixed dictionary $\mathbf{D}_{t+1} \in \mathbb{R}^{p \times M_{t+1}}$, may be expressed in terms of the parameter space of coefficients only. In order to do so, we first define the stochastic gradient update without projection, given function f_t parameterized by dictionary $\mathbf{D}_t$ and coefficients $\mathbf{w}_t$, as

$$\tilde{f}_{t+1} = (1 - \eta_t \lambda) f_t - \eta_t \nabla_f \ell(f_t; \mathbf{x}_t, \mathbf{y}_t). \tag{17}$$

This update may be represented using dictionary and weight vector

$$\tilde{\mathbf{D}}_{t+1} = [\mathbf{D}_t, \;\; \mathbf{x}_t], \quad \tilde{\mathbf{w}}_{t+1} = [(1-\eta_t\lambda)\mathbf{w}_t, \;\; -\eta_t \ell'(f_t(\mathbf{x}_t), y_t)] \; . \tag{18}$$

Observe that $\tilde{\mathbf{D}}_{t+1}$ has $\tilde{M} = M_t + 1$ columns, which is also the length of $\tilde{\mathbf{w}}_{t+1}$. For a fixed dictionary $\mathbf{D}_{t+1}$, the stochastic projection in (16) amounts to a least-squares problem on the coefficient vector. To see this, make use of the Representer Theorem to rewrite (16) in terms of kernel expansions, and that the coefficient vector is the only free parameter to write

$$\begin{aligned} &\operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^{M_{t+1}}} \frac{1}{2\eta_t} \Big\| \sum_{n=1}^{M_{t+1}} w_n \kappa(\mathbf{d}_n, \cdot) - \sum_{m=1}^{\tilde{M}} \tilde{w}_m \kappa(\tilde{\mathbf{d}}_m, \cdot) \Big\|_{\mathcal{H}}^2 \qquad (19) \\ &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^{M_{t+1}}} \frac{1}{2\eta_t} \left(\sum_{n,n'=1}^{M_{t+1}} w_n w_{n'} \kappa(\mathbf{d}_n, \mathbf{d}_{n'}) - 2 \sum_{n,m=1}^{M_{t+1},\tilde{M}} w_n \tilde{w}_m \kappa(\mathbf{d}_n, \tilde{\mathbf{d}}_m) + \sum_{m,m'=1}^{\tilde{M}} \tilde{w}_m \tilde{w}_{m'} \kappa(\tilde{\mathbf{d}}_m, \tilde{\mathbf{d}}_{m'}) \right) \\ &= \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^{M_{t+1}}} \frac{1}{2\eta_t} \Big(\mathbf{w}^T \mathbf{K}_{\mathbf{D}_{t+1},\mathbf{D}_{t+1}} \mathbf{w} - 2\mathbf{w}^T \mathbf{K}_{\mathbf{D}_{t+1},\tilde{\mathbf{D}}_{t+1}} \tilde{\mathbf{w}}_{t+1} + \tilde{\mathbf{w}}_{t+1} \mathbf{K}_{\tilde{\mathbf{D}}_{t+1},\tilde{\mathbf{D}}_{t+1}} \tilde{\mathbf{w}}_{t+1} \Big) := \mathbf{w}_{t+1} \; . \end{aligned}$$

In (19), the first equality comes from expanding the square, and the second comes from defining the cross-kernel matrix $\mathbf{K}_{\mathbf{D}_{t+1},\tilde{\mathbf{D}}_{t+1}}$ whose $(n,m)^{\text{th}}$ entry is given by $\kappa(\mathbf{d}_n, \tilde{\mathbf{d}}_m)$. The other kernel matrices $\mathbf{K}_{\tilde{\mathbf{D}}_{t+1},\tilde{\mathbf{D}}_{t+1}}$ and $\mathbf{K}_{\mathbf{D}_{t+1},\mathbf{D}_{t+1}}$ are similarly defined. Note that M_{t+1} is the number of columns in $\mathbf{D}_{t+1}$, while $\tilde{M} = M_t + 1$ is the number of columns in $\tilde{\mathbf{D}}_{t+1}$ [cf. (18)]. The explicit solution of (19) may be obtained by noting that the last term is a constant independent of $\mathbf{w}$, and thus by computing gradients and solving for $\mathbf{w}_{t+1}$ we obtain

$$\mathbf{w}_{t+1} = \mathbf{K}_{\mathbf{D}_{t+1}\mathbf{D}_{t+1}}^{-1} \mathbf{K}_{\mathbf{D}_{t+1}\tilde{\mathbf{D}}_{t+1}} \tilde{\mathbf{w}}_{t+1} \; , \tag{20}$$

Given that the projection of $\tilde{f}_{t+1}$ onto the stochastic subspace $\mathcal{H}_{\mathbf{D}_{t+1}}$, for a fixed dictionary $\mathbf{D}_{t+1}$, amounts to a simple least-squares multiplication, we turn to detailing how the kernel dictionary $\mathbf{D}_{t+1}$ is selected from the data sample path $\{\mathbf{x}_u, y_u\}_{u\le t}$.

Dictionary Update The selection procedure for the kernel dictionary $\mathbf{D}_{t+1}$ is based upon greedy sparse approximation, a topic studied extensively in the compressive sensing community (Needell et al., 2008). The function $\tilde{f}_{t+1} = (1-\eta_t) f_t - \eta_t \nabla_f \ell(f_t; \mathbf{x}_t, \mathbf{y}_t)$ defined by stochastic gradient method without projection is parameterized by dictionary $\tilde{\mathbf{D}}_{t+1}$ [cf. (18)], whose model order is $\tilde{M} = M_t + 1$. We form $\mathbf{D}_{t+1}$ by selecting a subset of M_{t+1} columns from $\tilde{\mathbf{D}}_{t+1}$ that are best for approximating $\tilde{f}_{t+1}$ in terms of error with respect to the Hilbert norm. As previously noted, numerous approaches are possible for seeking a sparse representation. We make use of *kernel orthogonal matching pursuit* (KOMP) (Vincent and Bengio, 2002) with allowed error tolerance ϵ_t to find a kernel dictionary matrix $\mathbf{D}_{t+1}$ based on the one which adds the latest sample point $\tilde{\mathbf{D}}_{t+1}$. This choice is due to the fact that we can tune its stopping criterion to guarantee stochastic descent, and guarantee the model order of the learned function remains finite – see Section 4 for details. Similar ideas have independently appeared in a prior work but without any analysis Zhao et al. (2012).

We now describe the variant of KOMP we propose using, called Destructive KOMP with Pre-Fitting (see Vincent and Bengio (2002), Section 2.3), which is summarized in

Algorithm 1 Destructive Kernel Orthogonal Matching Pursuit (KOMP)

Require: function $\tilde{f}$ defined by dict. $\tilde{\mathbf{D}} \in \mathbb{R}^{p\times\tilde{M}}$, coeffs. $\tilde{\mathbf{w}} \in \mathbb{R}^{\tilde{M}}$, approx. budget $\epsilon_t > 0$
initialize $f = \tilde{f}$, dictionary $\mathbf{D} = \tilde{\mathbf{D}}$ with indices $\mathcal{I}$, model order $M = \tilde{M}$, coeffs. $\mathbf{w} = \tilde{\mathbf{w}}$.

while candidate dictionary is non-empty $\mathcal{I} \neq \emptyset$ **do**
  for $j = 1, \ldots, \tilde{M}$ **do**
    Find minimal approximation error with dictionary element $\mathbf{d}_j$ removed

$$\gamma_j = \min_{\mathbf{w}_{\mathcal{I}\setminus\{j\}}\in\mathbb{R}^{M-1}} \|\tilde{f}(\cdot) - \sum_{k\in\mathcal{I}\setminus\{j\}} w_k \kappa(\mathbf{d}_k, \cdot)\|_{\mathcal{H}} \; .$$

  end for
  Find dictionary index minimizing approximation error: $j^* = \operatorname{argmin}_{j\in\mathcal{I}} \gamma_j$
  if minimal approximation error exceeds threshold $\gamma_{j^*} > \epsilon_t$
    stop
  else
    Prune dictionary $\mathbf{D} \leftarrow \mathbf{D}_{\mathcal{I}\setminus\{j^*\}}$
    Revise set $\mathcal{I} \leftarrow \mathcal{I} \setminus \{j^*\}$ and model order $M \leftarrow M - 1$.
    Compute updated weights $\mathbf{w}$ defined by current dictionary $\mathbf{D}$

$$\mathbf{w} = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^M} \|\tilde{f}(\cdot) - \mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\|_{\mathcal{H}}$$

  end
end while
return $f, \mathbf{D}, \mathbf{w}$ of model order $M \le \tilde{M}$ such that $\|f - \tilde{f}\|_{\mathcal{H}} \le \epsilon_t$

Algorithm 1. This flavor of KOMP takes as an input a candidate function $\tilde{f}$ of model order $\tilde{M}$ parameterized by its kernel dictionary $\tilde{\mathbf{D}} \in \mathbb{R}^{p\times\tilde{M}}$ and coefficient vector $\tilde{\mathbf{w}} \in \mathbb{R}^{\tilde{M}}$. The method then seeks to approximate $\tilde{f}$ by a parsimonious function $f \in \mathcal{H}$ with a lower model order. Initially, this sparse approximation is the original function $f = \tilde{f}$ so that its dictionary is initialized with that of the original function $\mathbf{D} = \tilde{\mathbf{D}}$, with corresponding coefficients $\mathbf{w} = \tilde{\mathbf{w}}$. Then, the algorithm sequentially removes dictionary elements from the initial dictionary $\tilde{\mathbf{D}}$, yielding a sparse approximation f of $\tilde{f}$, until the error threshold $\|f - \tilde{f}\|_{\mathcal{H}} \le \epsilon_t$ is violated, in which case it terminates.

At each stage of KOMP, a single dictionary element j of $\mathbf{D}$ is selected to be removed which contributes the least to the Hilbert-norm approximation error $\min_{f\in\mathcal{H}_{\mathbf{D}\setminus\{j\}}} \|\tilde{f} - f\|_{\mathcal{H}}$ of the original function $\tilde{f}$, when dictionary $\mathbf{D}$ is used. Since at each stage the kernel dictionary is fixed, this amounts to a computation involving weights $\mathbf{w} \in \mathbb{R}^{M-1}$ only; that is, the error of removing dictionary point $\mathbf{d}_j$ is computed for each j as $\gamma_j = \min_{\mathbf{w}_{\mathcal{I}\setminus\{j\}}\in\mathbb{R}^{M-1}} \|\tilde{f}(\cdot) - \sum_{k\in\mathcal{I}\setminus\{j\}} w_k \kappa(\mathbf{d}_k, \cdot)\|$. We use the notation $\mathbf{w}_{\mathcal{I}\setminus\{j\}}$ to denote the entries of $\mathbf{w} \in \mathbb{R}^M$ restricted to the sub-vector associated with indices $\mathcal{I} \setminus \{j\}$. Then, we define the dictionary element which contributes the least to the approximation error as $j^* = \operatorname{argmin}_j \gamma_j$. If the error associated with removing this kernel dictionary element exceeds the given approximation budget $\gamma_{j^*} > \epsilon_t$, the algorithm terminates. Otherwise, this dictionary element $\mathbf{d}_{j^*}$ is removed, the weights $\mathbf{w}$ are revised based on the pruned dictionary as

Algorithm 2 Parsimonious Online Learning with Kernels (POLK)

Require: $\{\mathbf{x}_t, \mathbf{y}_t, \eta_t, \epsilon_t\}_{t=0,1,2,\ldots}$

initialize $f_0(\cdot) = 0, \mathbf{D}_0 = [], \mathbf{w}_0 = []$, i.e. initial dictionary, coefficient vectors are empty

for $t = 0, 1, 2, \ldots$ **do**

Obtain independent training pair realization $(\mathbf{x}_t, y_t)$

Compute unconstrained functional stochastic gradient step [cf. (17)]

$$\tilde{f}_{t+1}(\cdot) = (1 - \eta_t\lambda) f_t - \eta_t \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot)$$

Revise dictionary $\tilde{\mathbf{D}}_{t+1} = [\mathbf{D}_t, \ \mathbf{x}_t]$ and weights $\tilde{\mathbf{w}}_{t+1} \leftarrow [(1-\eta_t\lambda)\mathbf{w}_t, \ -\eta_t\ell'(f_t(\mathbf{x}_t), y_t)]$

Compute sparse function approximation via Algorithm 1

$$(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \mathbf{KOMP}(\tilde{f}_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$$

end for

$\mathbf{w} = \operatorname{argmin}_{\mathbf{w}\in\mathbb{R}^M} \|\tilde{f}(\cdot) - \mathbf{w}^T\boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\|_{\mathcal{H}}$, and the process repeats as long as the current function approximation is defined by a nonempty dictionary.

With Algorithm 1 stated, we may summarize the key steps of the proposed method in Algorithm 2 for solving (1) while maintaining a finite model order, thus breaking the "curse of kernelization." The method, Parsimonious Online Learning with Kernels (POLK), executes the stochastic projection of the functional stochastic gradient iterates onto sparse subspaces $\mathcal{H}_{\mathbf{D}_{t+1}}$ stated in (16). The initial function is set to null $f_0 = 0$, meaning that it has empty kernel dictionary $\mathbf{D}_0 = []$ and coefficient vector $\mathbf{w}_0 = []$. The notation $[]$ is used to denote the empty matrix or vector respective size $p \times 0$ or 0. Then, at each step, given an independent training example $(\mathbf{x}_t, y_t)$ and step-size η_t, we compute the *unconstrained* functional stochastic gradient iterate $\tilde{f}_{t+1}(\cdot) = (1 - \eta_t\lambda)f_t - \eta_t\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot)$ which admits the parametric representation $\tilde{\mathbf{D}}_{t+1}$ and $\tilde{\mathbf{w}}_{t+1}$ as stated in (18). These parameters are then fed into KOMP with approximation budget ϵ_t, such that $(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \text{KOMP}(\tilde{f}_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$.

In the next section, we discuss the analytical properties of Algorithm 2 for solving online nonparametric regression problems of the form (1). We close here with an example algorithm derivation for the kernel logistic regression problem stated in Example 1 and a remark regarding the selection of KOMP to execute the greedy compression.

Example 2 *(Kernel Logistic Regression)* Returning to the case of kernel logistic regression stated in Example 1, with feature vectors $\mathbf{x}_n \in \mathcal{X} \subseteq \mathbb{R}^p$ and binary class labels $y_n \in \{0, 1\}$, we may perform sparse function estimation in $\mathcal{H}$ that fits a a training example $\mathbf{x}$ to its associated label y under the logistic model [cf. (7)] of the odds-ratio of the given class label. The associated λ-regularized maximum-likelihood estimation (MLE) is given as (8). Provided that a particular kernel map $\kappa(\cdot, \cdot)$, regularizer λ, and step-size η_t have been chosen, the only specialization of Algorithm 2 to this case is the computation of $\tilde{f}_t$, which requires computing the stochastic gradient of (7) with respect to an instantaneous training

example $(\mathbf{x}_t, y_t)$. Doing so specializes (17) to

$$\tilde{f}_{t+1}(\cdot) = (1 - \eta_t\lambda)f_t - \eta_t \frac{\exp\{-\tilde{f}_t(\mathbf{x}_t)\}}{[1 + \exp\{-\tilde{f}_t(\mathbf{x}_t)\}]^2}\kappa(\mathbf{x}_t, \cdot)\ . \tag{21}$$

The resulting dictionary and parameter updates implied by (21), given in (18), are then fed into KOMP (Algorithm 1) which returns their greedy sparse approximation for a fixed budget ϵ_t.

Remark 1 *(On the Choice of Compression Method)* Many penalty-based compression methods (Fu, 1998; Zou and Hastie, 2005) in principle could be used to reduce the complexity of the nonparametric regressor through the introduction of sparsity-promiting terms that are added to the objective in (1). However, given the infinite sample size setting of the problem considered in this work, it is not enough to sparsify the solution, since (1) must be solved with FSGD [cf. (12)]. The per-iteration complexity of FSGD [cf. (14)] that is necessary to actually solve the optimization problem means that it will still be untenably computationally burdensome even to find a sparsified solution. Thus, it is necessary to apply sparsification in an *online manner* to the FSGD sequence, akin to Algorithm 2. However, compressing the FSGD sequence may violate the descent properties of the optimization algorithm, which motivates us to address how to compress the parameterization while preserving stochastic descent. In POLK, this is done by identifying the relationship between the projected and un-projected stochastic gradient, and noting that the directional error in this relationship may be controlled by tying compression budget ϵ_t to the choice of algorithm learning rate η_t.

Therefore, to ensure that we actually attain the minimizer of the expected risk functional (1) almost surely, we need to sparsify FSGD while preserving the validity of the descent direction in the function space. In principle, it is possible to do this with *any method* that sparsifies FSGD while guaranteeing the function sequence moves in a Hilbert-norm error neighborhood of the true functional stochastic gradient. The reason we select KOMP in particular is that it easily admits a Hilbert-norm error neighborhood stopping criterion in conjunction with its compression procedure, and we were not aware of lower complexity alternatives that meet this stringent criterion. The result is that the complexity of the function sequence is then tailored to solve the learning task at hand and the model order is directly modified on the fly by that which is necessary to represent f^*, rather than arbitrarily fixed in advance of training.

Alternative online sparsification methods based on sliding windows (Kivinen et al., 2004), fixed-size subspace projections (Wang and Vucetic, 2010a; Wang et al., 2012; Zhao and Hoi, 2012), approximate linear dependence tests (Engel et al., 2004; Richard et al., 2009), or random dropping methods Zhang et al. (2013); Le et al. (2016b) cannot directly control the directional error/bias induced by sparsification so that the algorithm approximately behaves like FSGD. This is because the descent direction is intrinsically tied to its parameterization associated with a sequentially expanding kernel dictionary (14) (see Section 3.1), and these methods are proposed to reduce the complexity of the kernel representation *independently* of the optimization sequence to which they are applied.

4. Convergence Analysis

We turn to studying the theoretical performance of Algorithm 2 developed in Section 3. In particular, we establish that the method, when a diminishing step-size is chosen, is guaranteed to converge to the optimum of (1). We further obtain that when a sufficiently small constant step-size is chosen, the limit infimum of the iterate sequence is within a neighborhood of the optimum. In both cases, the convergence behavior depends on the approximation budget used in the online sparsification procedure detailed in Algorithm 1.

We also perform a worst-case analysis of the model order of the instantaneous iterates resulting from Algorithm 2, and show that asymptotically the model order depends on that of the optimal $f^* \in \mathcal{H}$. Before proceeding with the analysis, we define key quantities to simplify the analysis and introduce standard assumptions which are necessary to establish convergence. First, define the regularized stochastic functional gradient as

$$\hat{\nabla}_f \ell(f_t(\mathbf{x}_t), y_t) = \nabla_f \ell(f_t(\mathbf{x}_t), y_t) + \lambda f_t \tag{22}$$

Further define the projected stochastic functional gradient associated with the update in (16) as

$$\tilde{\nabla}_f \ell(f_t(\mathbf{x}_t), y_t) = \Big(f_t - \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}} \Big[f_t - \eta_t \hat{\nabla}_f \ell(f_t(\mathbf{x}_t), y_t) \Big] \Big) / \eta_t \tag{23}$$

such that the Hilbert space update of Algorithm 2 [cf. (16)] may be expressed as a stochastic descent using projected functional gradients

$$f_{t+1} = f_t - \eta_t \tilde{\nabla}_f \ell(f_t(\mathbf{x}_t), y_t) \; . \tag{24}$$

The definitions (23) - (22) will be used to analyze the convergence behavior of the algorithm. Before doing so, observe that the stochastic functional gradient in (22), based upon the fact that $(\mathbf{x}_t, y_t)$ are independent and identically distributed realizations of the random pair $(\mathbf{x}, y)$, is an unbiased estimator of the true functional gradient of the regularized expected risk $R(f)$ in (1), i.e.

$$\mathbb{E}[\hat{\nabla}_f \ell(f_t(\mathbf{x}_t), y_t) \,|\, \mathcal{F}_t] = \nabla_f R(f_t) \tag{25}$$

for all t. Next, we formally state technical conditions on the loss functions, data domain, and stochastic approximation errors that are necessary to establish convergence.

Assumption 1 *The feature space $\mathcal{X} \subset \mathbb{R}^p$ and target domain $\mathcal{Y} \subset \mathbb{R}$ are compact, and the reproducing kernel map may be bounded as*

$$\sup_{\mathbf{x} \in \mathcal{X}} \sqrt{\kappa(\mathbf{x}, \mathbf{x})} = X < \infty \tag{26}$$

Assumption 2 *The instantaneous loss $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is uniformly C-Lipschitz continuous for all $z \in \mathbb{R}$ for a fixed $y \in \mathcal{Y}$*

$$|\ell(z, y) - \ell(z', y)| \leq C|z - z'| \tag{27}$$

Assumption 3 *The loss function $\ell(f(\mathbf{x}), y)$ is convex and differentiable with respect to its first (scalar) argument $f(\mathbf{x})$ on $\mathbb{R}$ for all $\mathbf{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$.*

Assumption 4 *Let $\mathcal{F}_t$ denote the sigma algebra which measures the algorithm history for times $u < t$, i.e. $\mathcal{F}_t = \{\mathbf{x}_u, y_u, u_u\}_{u=1}^t$. The projected functional gradient of the regularized instantaneous risk in (22) has finite conditional second moments for each t, that is,*

$$\mathbb{E}[\|\tilde{\nabla}_f \ell(f_t(\mathbf{x}_t), y_t)\|_{\mathcal{H}}^2 \mid \mathcal{F}_t] \leq \sigma^2 \tag{28}$$

Assumption 1 holds in most practical settings by the data domain itself, and justifies the bounding of the loss in Assumption 2. Taken together, these conditions permit bounding the optimal function f^* in the Hilbert norm, and imply that the worst-case model order is guaranteed to be finite. Variants of Assumption 2 appear in the analysis of stochastic descent methods in the kernelized setting (Pontil et al., 2005; Ying and Zhou, 2006). Assumption 3 is satisfied for supervised learning problems such as logistic regression, support vector machines with the square-hinge-loss, the square loss, among others. Moreover, it is a standard condition in the analysis of descent methods (see Boyd and Vanderberghe (2004)). Assumption 4 is common in stochastic approximation literature, and ensures that the variance of the stochastic approximation error is finite. Under these conditions, we establish technical results in Appendix A which are used to establish our main convergence results in the following subsection.

4.1. Iterate Convergence

As is customary in the analysis of stochastic algorithms, we establish that under a diminishing algorithm step-size scheme (non-summable and square-summable), with the sparse approximation budget selection

$$\sum_{t=1}^{\infty} \eta_t = \infty \; , \quad \sum_{t=1}^{\infty} \eta_t^2 < \infty \; , \quad \epsilon_t = \eta_t^2 \; , \tag{29}$$

Algorithm 2 converges exactly to the optimal function f^* in stated in (1) almost surely. We use the abbreviation a.s. for an event that occurs almost surely, or with probability 1.

Theorem 2 *Consider the sequence generated $\{f_t\}$ by Algorithm 2 with $f_0 = 0$, and denote f^* as the minimizer of the regularized expected risk stated in (1). Let Assumptions 1-4 hold and suppose the step-size rules and approximation budget are diminishing as in (29) with regularizer such that $\eta_t < 1/\lambda$ for all t. Then the objective function error sequence converges to null in infimum almost surely as*

$$\liminf_{t \to \infty} R(f_t) - R(f^*) = 0 \qquad a.s. \tag{30}$$

Moreover, the sequence of functions $\{f_t\}$ converges almost surely to the optimum f^* as

$$\lim_{t \to \infty} \|f_t - f^*\|_{\mathcal{H}}^2 = 0 \qquad a.s. \tag{31}$$

Proof: See Appendix B. ■

The result in Theorem 2 states that when a diminishing algorithm step-size is chosen as, e.g. $\eta_t = \mathcal{O}(1/t)$, and the approximation budget that dictates the size of the sparse stochastic subspaces onto which the iterates are projected is selected as $\epsilon_t = \eta_t^2$, we obtain

exact convergence to the optimizer of the regularized expected risk in (1). However, in obtaining exact convergence behavior, we require the approximation budget to approach null asymptotically, which means that the model order of the resulting function sequence may grow arbitrarily, unless f^* is sparse and the magnitude of the stochastic gradient reduces sufficiently quickly, i.e., comparable to $\epsilon_t = \mathcal{O}(1/t^2)$.

If instead we consider a constant algorithm step-size $\eta_t = \eta$ and the approximation budget $\epsilon_t = \epsilon$ is chosen as a constant which satisfies $\epsilon_t = \epsilon = \mathcal{O}(\eta^{3/2})$, we obtain that the iterates converge in infimum to a neighborhood of the optimum, as we state next.

Theorem 3 *Denote $\{f_t\}$ as the sequence generated by Algorithm 2 with $f_0 = 0$, and denote f^* as the minimizer of the regularized expected risk stated in (1). Let Assumptions 1-4 hold, and given regularizer $\lambda > 0$, suppose a constant algorithm step-size $\eta_t = \eta$ is chosen such that $\eta < 1/\lambda$, and the sparse approximation budget satisfies $\epsilon = K\eta^{3/2} = \mathcal{O}(\eta^{3/2})$, where K is a positive scaler. Then the algorithm converges to a neighborhood almost surely as*

$$\liminf_{t\to\infty} \|f_t - f^*\|_{\mathcal{H}} \leq \frac{\sqrt{\eta}}{\lambda}\left(K + \sqrt{K^2 + \lambda\sigma^2}\right) = \mathcal{O}(\sqrt{\eta}) \qquad a.s. \tag{32}$$

Proof: See Appendix C. ■

Theorem 3 states that when a sufficiently small constant step-size is used together with a bias tolerance induced by sparsification chosen as $\epsilon = \mathcal{O}(\eta^{3/2})$, Algorithm 2 converges in infimum to a neighborhood of the optimum which depends on the chosen step-size, the parsimony constant K which scales the approximation budget ϵ, the regularization parameter λ, as well as the variance of the stochastic gradient σ^2. This result again is typical of convergence results in stochastic gradient methods. However, the use of a constant learning rate allows use to guarantee the model order of the resulting function sequence is always bounded, as we establish in the following subsection.

4.2. Model Order Control

In this subsection, we establish that the sequence of functions $\{f_t\}$ generated by Algorithm 2, when a constant algorithm step-size is selected, is parameterized by a kernel dictionary which is guaranteed to have finitely many elements, i.e., its the model order remains bounded. We obtain that the worst-case bound on the model order of f_t depends by the topological properties of the feature space $\mathcal{X}$, the Lipschitz constant of the instantaneous loss, and the radius of convergence $\Delta = (\sqrt{\eta}/\lambda)(K + \sqrt{K^2 + \lambda\sigma^2})$ defined in Theorem 3.

Theorem 4 *Denote f_t as the function sequence defined by Algorithm 2 with constant step-size $\eta_t = \eta < 1/\lambda$ and approximation budget $\epsilon = K\eta^{3/2}$ where $K > 0$ is an arbitrary positive scalar. Let M_t be the model order of f_t i.e., the number of columns of the dictionary $\mathbf{D}_t$ which parameterizes f_t. Then there exists a finite upper bound M^∞ such that, for all $t \geq 0$, the model order is always bounded as $M_t \leq M^\infty$. Consequently, the model order of the limiting function $f^\infty = \lim_t f_t$ is finite.*

Proof: See Appendix D.1. ■

The number of kernel dictionary elements in the function sequence f_t generated by Algorithm 2 is in the worst-case determined by the packing number of the kernel transformation

Table 1: Summary of convergence results for different parameter selections.

	Diminishing	Constant
Step-size/Learning rate	$\eta_t = \mathcal{O}(1/t)$	$\eta_t = \eta > 0$
Sparse Approximation Budget	$\epsilon_t = \eta_t^2$	$\epsilon = \mathcal{O}(\eta^{3/2})$
Regularization Condition	$\eta_t < 1/\lambda$	$\eta < 1/\lambda$
Convergence Result	$f_t \to f^*$ a.s.	$\liminf_t \|f_t - f^*\| = \mathcal{O}(\sqrt{\eta})$ a.s.
Model Order Guarantee	None	Finite

of the feature space $\phi(\mathcal{X}) = \kappa(\mathcal{X}, \cdot)$, as shown in the proof of Theorem 4. Moreover, the online sparsification procedure induced by KOMP reduces to a condition on the scale of the packing number of $\phi(\mathcal{X})$ as stated in (76). Specifically, as the radius $\frac{K\sqrt{\eta}}{C}$ increases, the packing number of the kernelized feature space decreases, and hence the required model order to fill $\phi(\mathcal{X})$ decreases. This radius depends on the constant K which scales the approximation budget selection η, the learning rate η, and the constant C bounding the gradient of the regularized instantaneous loss.

We have established that Algorithm 2 yields convergent behavior for the problem (1) in both diminishing and constant step-size regimes. When the learning rate η_t satisfies $\eta_t < 1/\lambda$, where $\lambda > 0$ is the regularization parameter, and is attenuating such that $\sum_t \eta_t = \infty$ and $\sum_t \eta_t^2 < \infty$, i.e., $\eta_t = \mathcal{O}(1/t)$, the approximation budget ϵ_t of Algorithm 1 must satisfy $\epsilon_t = \eta_t^2$ [cf. (29)]. Practically speaking, this means that asymptotically the iterates generated by Algorithm 2 may have a very large model order in the diminishing step-size regime, since the approximation budget is vanishing as $\epsilon_t = \mathcal{O}(1/t^2)$. On the other hand, when a constant algorithm step-size $\eta_t = \eta$ is chosen to satisfy $\eta < 1/\lambda$, then we only require the constant approximation budget $\epsilon_t = \epsilon$ to satisfy $\epsilon = \mathcal{O}(\eta^{3/2})$. This means that in the constant learning rate regime, we obtain a function sequence which converges to a neighborhood of the optimal f^* defined by (1) and is guaranteed to have a finite model order. These results are summarized in Table 1.

Remark 5 *(Sparsity of f^*)* Algorithm 2 provides a method to avoid keeping an unnecessarily large number of kernel dictionary elements along the convergence path towards f^* [cf. (1)], solving the classic scalability problem of kernel methods in stochastic programming. However, if the optimal function admits a low dimensional representation $|\mathcal{I}| \ll \infty$, then in addition to extracting memory efficient instantaneous iterates, POLK will obtain the optimal function exactly. In Section 5, we illustrate this property via a multi-class classification problem where the data is generated from Gaussian mixture models.

4.3. Complexity Analysis

We now study the per-iteration computational complexity of Algorithm 2. Throughout the following discussion, we shall use M_t to denote the model order at iteration t, and p to

denote the dimension of the input (feature) space. For convenience of analysis, Algorithm 2 can be thought of as operating in two distinct phases:

i. **Append**: augment the dictionary and weights so as to take the true FSGD step (i.e., such that they represent $\tilde{f}_{t+1}$).

ii. **Prune**: use matching pursuit to reduce the model order (i.e., compute the approximation f_{t+1}).

In analyzing the append phase, there are two major operations to consider. The first is the computation of the coefficient $\ell'(f(\mathbf{x}_t), y_t)$ for the new point, which requires M_t kernel evaluations, $\{\kappa(\mathbf{d}_j, \mathbf{x}_t)\}_{j=1}^{M_t}$, to be weighted and summed. Computing the kernel values exhibits complexity $\mathcal{O}(pM_t^2)$ (the complexity of a kernel evaluation scales at least linearly with p), and the weighting and summing exhibits complexity $\mathcal{O}(M_t)$. The second major operation that takes place in the append phase is that of augmenting the kernel matrix. However, for computational efficiency, a copy of the inverse of this kernel matrix is also kept, which can be updated in $O(M_t^2)$ time using classical rank-one, block-inverse update rules, based on the Sherman-Woodbury-Morrison identity (see, e.g., KRLS in Engel et al. (2004)). Therefore, the total computational complexity of the append phase is $\mathcal{O}(M_t^2)$.

To analyze the pruning phase, we first introduce some additional notation. Let Z_t denote the total number of the outer iterations of Algorithm 1 that actually take place at time t, i.e., the number of points removed from $\tilde{\mathbf{D}}$. Using an efficient implementation of Algorithm 1 inspired by Vincent and Bengio (2002), each of these Z_t iterations exhibits a computational complexity of $\mathcal{O}(M_t^2)$. After the stopping criterion has been met, the augmented function is projected onto the subspace spanned by the pruned dictionary, which also exhibits a complexity of $\mathcal{O}(M_t^2)$. Finally, the new kernel matrix and its inverse are updated, which, too, incurs a computational cost of $\mathcal{O}(M_t^2)$, bringing the overall complexity of the pruning phase to $\mathcal{O}(Z_t M_t^2)$. A pessimistic estimate of Z_t is M_t, although in practice it is always less than the mini-batch size B.

Combining the terms from each phase, the total complexity of Algorithm 2 at iteration t comes out to $O(Z_t M_t^2)$. In the steady-state, $Z = 1$ (i.e., $M_{t+1} = M_t$), which further reduces the expression to $\mathcal{O}(M_t^2)$, exactly the complexity of KRLS (Engel et al., 2004). Observe that the complexity of FSGD is $\mathcal{O}(t)$, which means that POLK is computationally advantageous whenever $t > Z_t M_t^2$. Since $\lim_t M_t$ is guaranteed to be finite, and usually $Z_t \leq B$, the mini-batch size.Thus, POLK is justified whenever t is large. This is typical of the online methods for kernel learning that we compare against experimentally in the following section.

Remark 6 *(Cheaper Projections)* Both theoretically and in practice, it is possible to cheapen the cost of projections by replacing the argmin in KOMP by a scheme that drops kernel dictionary elements *uniformly at random* until the function hits the boundary of a Hilbert-norm error neighborhood. Such a scheme's convergence would still be guaranteed by the preceding theoretical analysis, and the cost of pruning points would be significantly cheaper. Specifically, it would be at most linear in the dictionary size $\mathcal{O}(M_t Z_t)$. However, the price of pursuing this route would be lower quality estimates of f^*. We have left the investigation of these random approximated projections to future work. However, we note that several

alternatives in the literature adopt a version of this approach, but with *fixed* subspace sizes (Wang and Vucetic, 2010a; Wang et al., 2012; Zhao and Hoi, 2012; Zhang et al., 2013; Le et al., 2016b; Lu et al., 2016) rather than error-neighborhood budgets, as is done in this work, which is not enough to establish almost sure convergence.

5. Experiments

In this section, we evaluate POLK by considering its performance on two supervised learning tasks trained for three streaming data sets. The specific tasks we consider are those of (a) training a multi-class kernel logistic regressor (KLR), and (b) training a multi-class kernel support vector machine (KSVM). The three data sets we use are (i) `multidist`, a synthetic data set we constructed using two-dimensional Gaussian mixture models; (ii) `mnist`, the MNIST handwritten digits (Lecun and Cortes, 2009); and (iii) `brodatz`, image textures drawn from a subset of the Brodatz texture database (Brodatz, 1966).

Where possible, we compare our technique with competing methods. Specifically, for the online support vector machine case, we compare with budgeted stochastic gradient descent (BSGD) Wang et al. (2012), a fixed subspace projection method, which requires a maximum model order *a priori*; Dual Space Gradient Descent (Dual) (Le et al., 2016a), which executes a hybrid combination of functional stochastic gradient method together with a hybrid random feature representation of the regression function; nonparametric budgeted stochastic gradient descent (NPBSGD) Le et al. (2016b), which combines a fixed subspace projection with random dropping, Naive Online R_{reg} Minimization Algorithm (NORMA) (Kivinen et al., 2004), which does a greedy objective function value based truncation to a fixed budget, and an instantiation of Budgeted Passive-Aggressive (BPA) algorithm based on k-nearest neighbor merging of incoming points (Wang and Vucetic, 2010a). We note that all the aforementioned methods fix the model order except Dual (Le et al., 2016a). Further, for logistic regression, from the preceding list, we omit BPA and BSGD since they are derived specifically for the SVM setting. We compare all of these online methods on our first data set but then only compare POLK and BSGD on the later benchmark data domains, since the trends we observe for the former directly scale up to the later.

For off-line (batch) KLR, we compare with the import vector machine (IVM) (Zhu and Hastie, 2005), a sparse second-order method. We also compare with the batch techniques of LIBSVM (Chang and Lin, 2011), applicable to KSVM only, an L-BFGS solver (Nocedal, 1980), and methods which approximate the kernel matrix using either Fourier transforms (Rahimi and Recht, 2008, 2009) or Nyström approximation (Williams and Seeger, 2001; Zhang et al., 2008).

5.1. Tasks

The tasks we consider are those of multi-class classification, which is a problem that admits approaches based on probabilistic and geometric criteria. In what follows, we use $\mathbf{x}_n \in \mathcal{X} \subset \mathbb{R}^p$ to denote the n^{th} feature vector in a given data set, and $y_n \in \{1, \ldots, C\}$ to denote its corresponding label.

Multi-class Kernel Support Vector Machines (Multi-KSVM) The first task we consider is that of training a multi-class kernel support vector machine, in which the merit of a particular regressor is defined by its ability to maximize its classification margin. In

particular, define a set of class-specific activation functions $f_c : \mathcal{X} \to \mathbb{R}$, and denote them jointly as $\mathbf{f} \in \mathcal{H}^C$. In Multi-KSVM, points are assigned the class label of the activation function that yields the maximum response. KSVM is trained by taking the instantaneous loss ℓ to be the multi-class hinge function which defines the margin separating hyperplane in the kernelized feature space, i.e.,

$$\ell(\mathbf{f}, \mathbf{x}_n, y_n) = \max(0, 1 + f_r(\mathbf{x}_n) - f_{y_n}(\mathbf{x}_n)) + \lambda \sum_{c'=1}^{C} \|f_{c'}\|_{\mathcal{H}}^2 \,, \tag{33}$$

where $r = \operatorname{argmax}_{c' \neq y} f_{c'}(\mathbf{x})$. Further details may be found in Murphy (2012).

Multi-class Kernel Logistic Regression (Multi-KLR) The second task we consider is that of kernel logistic regression, wherein, instead of maximizing the margin which separates sample points in the kernelized feature space, we instead adopt a probabilistic model on the odds ratio that a sample point has a specific label relative to all others. Using the same notation as above for the class-specific activation functions, we adopt the probabilistic model:

$$P(y = c \,|\, \mathbf{x}) \triangleq \frac{\exp(f_c(\mathbf{x}))}{\sum_{c'} \exp(f_{c'}(\mathbf{x}))}. \tag{34}$$

which models the odds ratio of a given sample point being in class c versus all others. We use the negative log likelihood pertaining to the above model as the instantaneous loss (see, e.g., Murphy (2012)), i.e.,

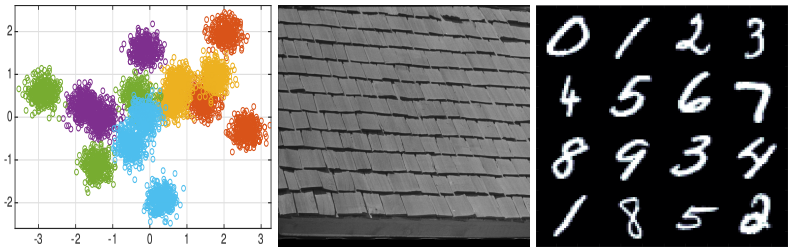
$$\begin{aligned} \ell(\mathbf{f}, \mathbf{x}_n, y_n) &= -\log P(y = y_n | \mathbf{x}_n) + \frac{\lambda}{2} \sum_c \|f_c\|_{\mathcal{H}}^2 \\ &= \log \left(\sum_{c'} \exp(f_{c'}(\mathbf{x}_n)) \right) - f_{y_n}(\mathbf{x}_n) + \frac{\lambda}{2} \sum_c \|f_c\|_{\mathcal{H}}^2 \,. \end{aligned} \tag{35}$$

Observe that the loss (35) substituted into the empirical risk minimization problem in Example 1 is its generalization to multi-class classification. For a given set of activation functions, the classification decision $\tilde{c}$ for $\mathbf{x}$ is given by the class that yields the maximum likelihood, i.e., $\tilde{c} = \operatorname{argmax}_{c \in \{1,\ldots,C\}} f_c(\mathbf{x})$.

5.2. Data Sets

We evaluate Algorithm 2 for the Multi-KLR and Multi-KSVM tasks described above using the `multidist`, `mnist`, `brodatz` data sets.

`multidist`
In a manner similar to (Zhu and Hastie, 2005), we generate the `multidist` data set using a set of Gaussian mixture models. The data set consists $N = 5000$ feature-label pairs for training and 2500 for testing. Each label y_n was drawn uniformly at random from the label set. The corresponding feature vector $\mathbf{x}_n \in \mathbb{R}^p$ was then drawn from a planar ($p = 2$), equitably-weighted Gaussian mixture model, i.e., $\mathbf{x} \,|\, y \sim (1/3) \sum_{j=1}^{3} \mathcal{N}(\boldsymbol{\mu}_{y,j}, \sigma_{y,j}^2 \mathbf{I})$ where $\sigma_{y,j}^2 = 0.2$ for all values of y and j. The means $\boldsymbol{\mu}_{y,j}$ are themselves realizations of their own Gaussian distribution with class-dependent parameters, i.e., $\boldsymbol{\mu}_{y,j} \sim \mathcal{N}(\boldsymbol{\theta}_y, \sigma_y^2 \mathbf{I})$, where



(a) `multidist` data (b) `brodatz` example texture (c) `mnist` examples

Figure 1: Visualizations of the data sets used in experiments.

$\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_C\}$ are equitably spaced around the unit circle, one for each class label, and $\sigma_y^2 = 1.0$. We fix the number of classes $C = 5$, meaning that the feature distribution has, in total, 15 distinct modes. The data points are plotted in Figure 1(a).

`mnist`
The `mnist` data set we use is the popular MNIST data set (Lecun and Cortes, 2009), which consists of $N = 60000$ feature-label pairs for training and 10000 for testing. Feature vectors are $p = 784$-dimensional, where each dimension captures a single grayscale pixel value (scaled to lie within the unit interval) that corresponds to a unique location in a 28-pixel-by-28-pixel image of a cropped, handwritten digit. Labels indicate which digit is written, i.e., there are $C = 10$ classes total, corresponding to digits $0, \ldots, 9$ – examples are given in Figure 1(c).

`brodatz`
We generated the `brodatz` data set using a subset of the images provided in Brodatz (1966). Specifically, we used 13 texture images (i.e., $C = 13$), and from them generated a set of 256 textons (Leung and Malik, 1999). Next, for each overlapping patch of size 24-pixels-by-24-pixels within these images, we took the feature to be the associated $p = 256$-dimensional texton histogram. The corresponding label was given by the index of the image from which the patch was selected. When then randomly selected $N = 10000$ feature-label pairs for training and 5000 for testing. An example texture image can be seen in Figure 1(b).

5.3. Parameter Selection

For the choice of kernel, some intuition regarding the data domain is helpful, but typically the Gaussian kernel is a good starting point, and therefore is our selection for all of the experiments. For bandwidth selection $\tilde{\sigma}^2$, we find that choosing it according to the sample variance of a cross-validation training subset to be reasonably effective, and this strategy was used in our implementation. The mini-batch size of 32 was selected based on reducing the noise of stochastic approximation error once steady state is achieved, and was done through trial and error by starting at batch sizes of 1 and doubling until the computational burden of a mini-batch exceeded its benefits in terms of noise reduction. We find the regularizer λ has little effect when the data does not have an untenably large variance, and therefore fix

it as a small constant. For problems with worse conditioning, however, larger regularizers may be useful.

The selection of the algorithm step-size and parsimony constant in our experiments is done through a simple trial and error procedure. We begin with a small step-size, between 0.01 and 0.1, to test whether the algorithm converges, with parsimony constant $K = 0.001$. Convergence, however, may occur while the model order grows untenably. So, we progressively increase the parsimony constant until the bias in the stochastic gradient becomes unmanageable and the algorithm diverges. We select the parsimony constant to be as large as possible while still preserving convergence of the algorithm. Then, we exploit knowledge of the trade-off between learning rate and step-size selection: smaller step-sizes yield slower learning and smaller asymptotic error. In light of this fact, we increment the step-size to increase the learning rate, and then ramp up the parsimony constant to its maximum value that preserves convergence. This trial and error procedure only needed to be repeated a few times for each data set/problem instance to obtain the POLK results presented in this section.

5.4. Results

For each task and data set described above, we implemented POLK (Algorithm 2) along with the competing methods described at the beginning of the section. For some of the tasks, only a subset of the competing methods are applicable, and in some cases such as online logistic regression, none are. Here, we shall describe the details of each experimental setting and the corresponding results.

`multidist` Results

Sparse SVM Due to the small size of our synthetic `multidist` data set, we were able to generate results for the Multi-KSVM task using each of the methods specified earlier except for IVM. For POLK, we used the following specific parameter values: we select the Gaussian/RBF kernel with bandwidth $\tilde{\sigma}^2 = 0.6$, constant learning rate $\eta = 6.0$, parsimony constant $K = 0.04$, and regularization constant $\lambda = 10^{-6}$. Further, we processed streaming samples in mini-batches of size 32. For BSGD, Dual, NORMA, NPBSGD, and BPA, we used the same $\tilde{\sigma}^2$ and λ, but achieved the best results with smaller constant learning rate $\eta = 1.0$ (perhaps due, in part, to the fact that it is unclear how to fold mini-batching into their implementation). For Dual, we set $k = 3$, which determines the how many model points we search over to omit from the dictionary and instead contribute to the random feature functional representation. For NPBSGD, we set the Bernoulli parameter $p = \min(1, \beta/t)$ with $\beta = (45/70)N$ as in the experiments of Le et al. (2016b). In order to compare with POLK, we set the competing methods' pre-specified model orders or budget parameters to be 16, i.e., the steady-state model orders of POLK parameterized with the value of K specified above.

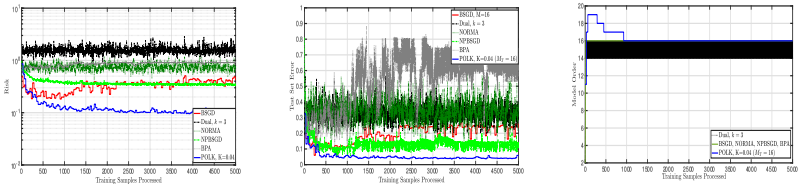
In Figure 2 we plot the empirical results of this experiment for POLK and its competitors, and observe that POLK outperforms many of its competitors by an order of magnitude in terms of objective evaluation (Fig. 2(a)) and test-set error rate (Fig 2(b)). The notable exception is NPBSGD which comes close in terms of objective evaluation but still falls short in terms of test error. Moreover, because the marginal feature density of `multidist` contains 15 modes, the optimal model order is $M^* = 15$, which is approximately *learned* by

	`multidist`	
Algorithm	Multi-KSVM (risk/error/model order)	Multi-Logistic (risk/error/model order)
LIBSVM	−/3.92/656	−/−/−
L-BFGS	0.0854/4.08/5000	0.0854/4.04/5000
IVM	−/−/−	0.0894/4.08/16
Fourier	−/3.88/5000	−/3.80/5000
Nyström	−/3.67/1000	−/3.72/1000
BSGD	0.385/21.8/16	−/−/−
Dual	1.598/30.5.8/15	1.417/48.4/15
NORMA	0.751/33.1/16	1.26/29.3/16
NPBSGD	0.334/11.6/16	0.101/3.92/16
BPA	0.947/64.1/16	−/−/−
POLK	**0.0919/3.98/16**	**0.120/4.36/16**

Table 2: Comparison of different methods on the `multidist` data set for extracting the optimally sparse functional representation, i.e., $K = 0.04$ for POLK and $M = 16$ for online competitors. Reported risk and error values for POLK and BSGD were averaged over the final 5% of processed training examples. Dashes indicate where the method could not be used to generate results because it is not defined for that task. LIBSVM is used as a baseline, but note that it uses a fundamentally different model for multi-class problems (a separate one-vs-all classifier is trained for each class, and then at test time, a majority vote is executed), and so a comparable risk value can not computed.

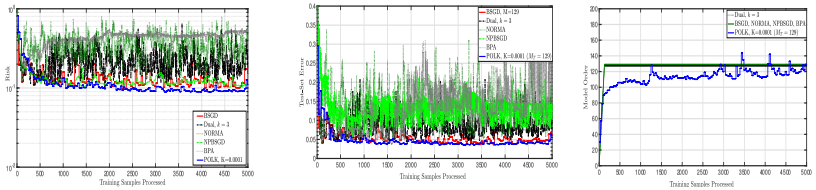
POLK for $K = 0.04$ (i.e., $M_T = 16$) (Fig. 2(c)). Several alternatives initialized with this parameter, on the other hand, do not converge. Observe that for this task POLK exhibits a state of the art trade off between test set accuracy and number of samples processed – reaching below 4% error after only 1249 samples. The final decision surface $\mathbf{f}_T$ of this trial of POLK is shown in Fig. 4(a), where it can be seen that the selected kernel dictionary elements concentrate near the modes of the marginal feature density.

We can also see from Table 2 that POLK compares favorably to the batch techniques for Multi-KSVM on the `multidist` data set. It achieves approximately the same error rate as LIBSVM with significantly fewer model points (support vectors) and even outperforms our (dense) L-BFGS batch solver in terms of test-set error, while adding the ability to process data in an online fashion. Compared to approximate batch solvers such as Random Fourier Features with $d = 1000$ features and all $N = 5000$ sample points, we observe POLK obtains a 0.1% error increase with a 99% reduction in model complexity. We also implement Nyström approximation with k-means sampled landmark points with $k = 1000$ (Zhang et al., 2008), and observe similar accuracy results to random features, which increased complexity associated with computing k means over the entire training set. We still report a model order equal to the sample size for these methods, however, since they require the full training set in order to implement their approximation procedure.



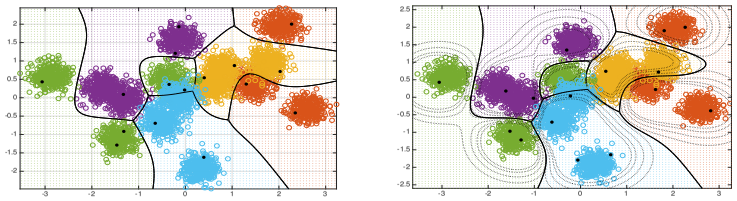
(a) Empirical risk $R(\mathbf{f}_t)$ (b) Error rate (c) Model order M_t

Figure 2: Comparison of POLK and its competitors on the `multidist` data set for the Multi-KSVM task for extracting a optimally sparse kernel regressor. Observe that POLK achieves lower risk and higher accuracy when we seek to learn a statistical model of comparable complexity to the class-conditional density ($M = 15$ total modes). Competing methods yield slower learning and higher test-set error rates.



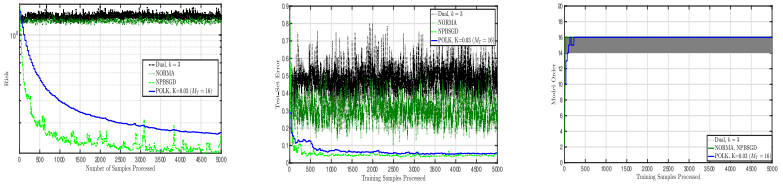
(a) Empirical risk $R(\mathbf{f}_t)$ (b) Error rate (c) Model order M_t

Figure 3: Comparison of POLK and its competitors on the `multidist` data set for the Multi-KSVM task when the model order parameter is fixed for the competitors at $M = 129$ and the parsimony constant of POLK is set to $K = 10^{-4}$. Observe that POLK achieves lower risk and higher accuracy than its competitors on this problem instance, but the gap is less relative to the sparser model extraction problem.



(a) $\mathbf{f}_T$ (hinge loss) (b) $\mathbf{f}_T$ (logistic loss)

Figure 4: Visualization of the decision surfaces yielded by POLK for the Multi-KSVM and Multi-Logisitic tasks on the `multidist` data set. Training examples from distinct classes are assigned a unique color. Grid colors represent the classification decision by $\mathbf{f}_T$. Bold black dots are kernel dictionary elements, which concentrate at the modes of the data distribution. Solid lines are drawn to denote class label boundaries, and dashed lines in 4(b) are drawn to denote confidence intervals.

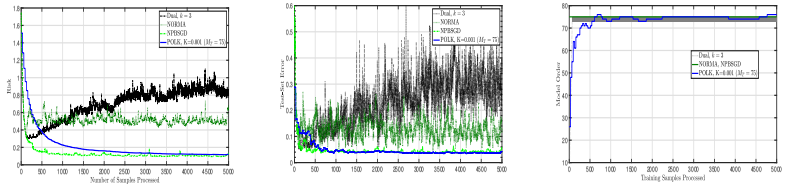


(a) Empirical risk $R(\mathbf{f}_t)$ (b) Error rate (c) Model order M_t

Figure 5: Empirical behavior of the POLK algorithm applied to the `multidist` data set for the Multi-Logistic task. Observe that the algorithm converges to a low risk value ($R(f_t) < 10^{-1}$) and achieves test set accuracy between 4% and 5% depending on choice of parsimony constant K, which respectively corresponds to a model order between 75 and 16.

Dense SVM We now repeat the previous experiment with the same parameters except for a different parsimony constant $K = 10^{-4}$ for POLK and a model order of $M = 129$ for its competitors, given that this is the value extracted by POLK for this parameter selection. The results of this experiment are given in Figure 3. Observe that NORMA and Dual appear to not converge for this experiment, while POLK converges stably, although its learning rate is exceeded by NPBSGD. This trend is observed both in terms of the risk functional (Fig. 3(a)) and test-set error (Fig 3(b)). We believe NPBSGD performs better for this case due to the smoothness of the logistic loss relative to the hinge loss. However, the test errors of these methods stabilize to similar levels of accuracy (below 4% error rate).

Sparse KLR For the Multi-Logisitic task on this data set, we were able to generate results for each method except BSGD, BPA, and LIBSVM, which are specifically tailored to the SVM task. For POLK, we used the following parameter values: Gaussian kernel with bandwidth $\tilde{\sigma}^2 = 0.6$, constant learning rate $\eta = 6.0$, parsimony constant $K = 0.03$, and



(a) Empirical risk $R(\mathbf{f}_t)$ (b) Error rate (c) Model order M_t

Figure 6: Empirical behavior of the POLK algorithm applied to the `multidist` data set for the Multi-Logistic task. Observe that the algorithm converges to a low risk value ($R(f_t) < 10^{-1}$) and achieves test set accuracy between 4% and 5% depending on choice of parsimony constant K, which respectively corresponds to a model order between 75 and 16.

regularization constant $\lambda = 10^{-6}$. As in Multi-KSVM, we processed the streaming samples in mini-batches of size 32. The empirical behavior of POLK for the Multi-Logistic task can be seen in Figure 5 and the final decision surface is presented in Figure 4(b). Observe that POLK is exhibits comparable convergence to the SVM problem, but a smoother descent due to the differentiability of the multi-logistic loss. The competing methods, initialized with a model order of $M = 16$ do not converge, except for NPBSGD, which attains comparable test error to POLK. In Table 2 we present final accuracy and risk values on the logistic task, and note that it performs comparably, or in some cases, favorably, to the batch techniques (IVM, L-BFGS, Random Features, Nyström with k-means sampled points), while processing streaming data. The final model f_T extracted by POLK attains comparable performance to off-line methods while reducing the model complexity by several orders of magnitude.

Dense KLR Now, we re-rerun the previous experiment but modify the parsimony constant of POLK as $K = 0.001$, which yields a limiting model of order $M = 75$. The competing methods are then run with this selection. The results of this experiment are given in Figure 6: observe that NORMA and Dual diverge, while NPBSGD converges slightly faster albeit more noisily than POLK. The later two methods attain comparable test accuracy near 96% by the end of training.

`mnist` and `brodatz` Results

By construction, the `multidist` data set above yields optimal activation functions that are themselves sparse (i.e., $\mathbf{f}^*$ has a low model order due to the marginal feature density). Here, we analyze the performance of POLK on more realistic data sets where the optimal solutions are not sparse, i.e., where one might desire a sparse approximation. Due to the increased size and dimensionality of these data sets, we were unable to generate results for `mnist` using the batch L-BFGS technique, and unable to generate results for either data set using IVM. With Random Fourier Features on `mnist`, we use a bandwidth of 0.0095 with $d = 1000$ randomly chosen features for both SVM and KLR. For `brodatz`, we use a larger bandwidth of $\tilde{\sigma}^2 = 4.0$, with $d = 10^4$ Random Features both for KSVM and KLR. For Nyström approximation, we set $k = 1000$ since larger choices of k cause memory problems.

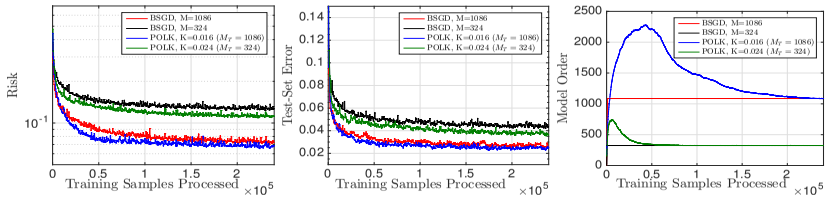
	`mnist`		`brodatz`	
Algorithm	Multi-KSVM (risk/error/model order)	Multi-Logistic (risk/error/model order)	Multi-KSVM (risk/error/model order)	Multi-Logistic (risk/error/model order)
LIBSVM	$-/1.50/16118$	$-/-/-$	$-/3.72/4777$	$-/-/-$
Fourier	$-/3.83/60000$	$-/3.87/60000$	$-/5.88/10000$	$-/6.18/10000$
Nyström	$-/2.89/60000$	$-/2.94/60000$	$-/5.13/10000$	$-/5.86/10000$
L-BFGS	$-/-/-$	$-/-/-$	$0.0319/4.44/10000$	$0.0572/4.00/10000$
BSGD	$0.0731/2.67/1086$	$-/-/-$	$0.0560/4.72/1171$	$-/-/-$
POLK	$\mathbf{0.0684/2.46/1086}$	$\mathbf{0.116/2.68/2326}$	$\mathbf{0.0507/4.53/1171}$	$\mathbf{0.0871/4.41/1833}$

Table 3: Comparison of POLK, BSGD, IVM, L-BFGS, and LIBSVM results on the `mnist` and `brodatz` data sets. Reported risk and error values for POLK and BSGD were averaged over the final 5% of processed training examples. Dashes indicate where the method could not be used to generate results either because it is not defined for the task or because the size of the problem was too large for that data set. For these reasons, IVM was not able to generate results for these data sets on either task, and so is omitted here. LIBSVM is used as a baseline, but note that it uses a fundamentally different model for multi-class problems (1v1 + majority vote), and so a comparable risk value can not be computed.

For Multi-KSVM on `mnist`, we used the following parameter values for POLK: Gaussian kernel with bandwidth $\tilde{\sigma}^2 = 4.0$, constant learning rate $\eta = 24.0$, parsimony constant $K \in \{0.16, 0.24\}$, and regularization constant $\lambda = 10^{-6}$. We again processed data in mini-batches of size 32. For `brodatz`, we used identical parameters except for changing the kernel bandwith $\tilde{\sigma}^2 = 0.1$ and parsimony constant $K \in \{0.01, 0.02\}$. For BSGD, we again found $\eta = 1.0$ to yield the best results on both datasets, and pre-specified model orders of $\{324, 1086\}$ and $\{305, 1171\}$ on `mnist` and `brodatz`, respectively, for comparison to POLK.

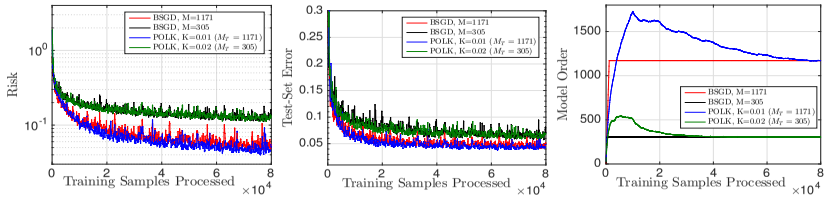
In Figures 7 and 8 we plot the empirical results of these experiments for POLK and BSGD. We observe that POLK is able to outperform the comparable BSGD trial in terms of convergence speed and steady-state risk and test-set error. The strength of the proposed technique is further demonstrated in Table 3, where we can see that POLK is able to achieve test-set error within 1-2% of LIBSVM while requiring a number of support vectors (model points) that is significantly-less than LIBSVM, *while* adding the ability to process streaming data.

For the Multi-Logistic task on `mnist`, we ran POLK using a Gaussian kernel with bandwidth $\tilde{\sigma}^2 = 4.0$, constant learning rate $\eta = 24.0$, parsimony constant $K \in \{0.08, 0.16\}$, and regularization constant $\lambda = 10^{-6}$. Data was processed in mini-batches of size 32 here as well. For `brodatz`, we again change the kernel bandwidth $\tilde{\sigma}^2 = 0.1$ and used different parsimony constants $K = \{0.005, 0.015\}$. The empirical behavior of POLK on this task can be seen in Figures 9 and 10. Observe that for this task the descent is smoother due to the differentiability of the logistic loss, although the asymptotic test accuracy is lower than that of KSVM.



(a) Expected risk $R(\mathbf{f}_t)$ (b) Error rate (c) Model order M_t

Figure 7: Comparison of POLK and BSGD on `mnist` data set for the Multi-KSVM task. Observe that POLK achieves lower risk and higher accuracy on this task, and extracts a model order directly from the feature space that yields convergence.



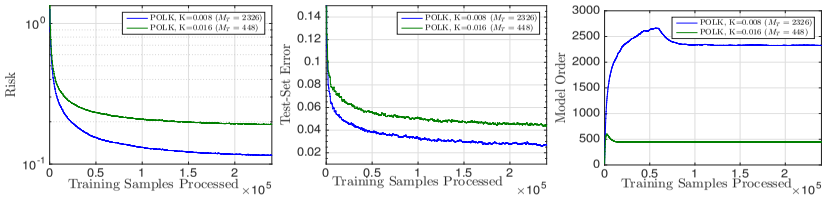
(a) Expected risk $R(\mathbf{f}_t)$ (b) Error rate (c) Model order M_t

Figure 8: Comparison of POLK and BSGD on `brodatz` data set for the Multi-KSVM task. We observe that POLK behaves similarly to BSGD for this task, stabilizing at an accuracy near 96%. For this dense data domain, larger model orders are needed to achieve convergence.

The overall performance is summarized in Table 3. Note that the only other technique that was able to generate results for this task was L-BFGS, and even there only on the `brodatz` data set, since the complexity bottleneck in the sample size for `mnist` is prohibitive for batch optimization. We see from this comparison that POLK yields a test-set error within 0.5% of the batch solution while using an order of magnitude fewer model points. Additionally, POLK is able to run *online*, with streaming data, whereas L-BFGS requires all the data points to be operated on at each step.

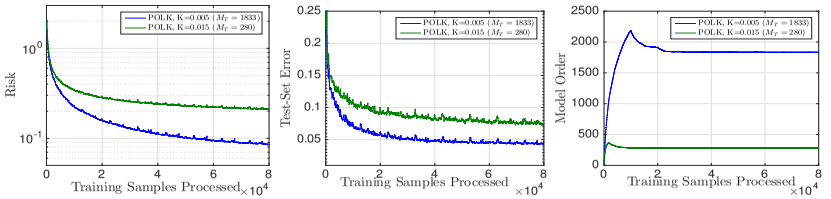
6. Discussion

Over the past several years, parametric function approximation has largely dominated the machine learning landscape. Deep learning is perhaps currently the most prominent parametric paradigm (Haykin, 1994). One must first specify a network structure, thereby fixing the parametric representation of the function to be learned, before proceeding to determine the coefficients linking neurons in different layers. Given this parametric representation, training techniques proceed by searching over the predefined parameter space for the op-



(a) Expected risk $R(\mathbf{f}_t)$ (b) Error rate (c) Model order M_t

Figure 9: Empirical behavior of the POLK algorithm applied to `mnist` data set for the Multi-Logistic task. The algorithm exhibits smoother convergence due to the differentiability of the logistic loss, and achieves asymptotic test error 2.6%. We again observe due to the dense data domain, larger model orders are needed to exhibit competitive classification performance.



(a) Expected risk $R(\mathbf{f}_t)$ (b) Error rate (c) Model order M_t

Figure 10: Empirical behavior of the POLK algorithm applied to the `brodatz` data set for the Multi-Logistic task. We observe convergent behavior, and a clear trade off between higher model order and increased accuracy. Due to this data domain being a more challenging task than the `mnist` digits, we observe asymptotic test accuracy of approximately 4.4%.

timal parameter values that minimize the error between the function and observed input-output pairs. The main reason for the popularity of parametric function approximation is its success in solving practical problems, but there are other factors that have fostered their adoption. One such factor is the availability of workable, if not necessarily efficient, optimization techniques for the determination of the optimal parameter values, in the form of stochastic gradient descent and its variants. Parametric stochastic gradient descent processes training examples sequentially and has a per-iteration complexity that is linear on the number of parameters but independent of the size of the training set.

Despite the success of parametric techniques, nonparametric function approximation has the advantage of expressive power in the sense that they are allowed to select the approximating function from a more general set of functions than those that admit a parametric form chosen *a priori*. This advantage is seen, e.g., in the improved classification accuracy of (nonparametric) kernel support vector machines (SVMs) relative to (parametric) linear SVMs (Evgeniou et al., 2000). This is not to say that nonparametric methods are necessarily better. Neural networks, e.g., have proven to be very adept parametric representations in image classification problems (Krizhevsky et al., 2012). However, it is nonetheless true that the better expressive power of nonparametric representations is of importance in some applications.

The importance of expressiveness notwithstanding, nonparametric approaches are relatively less popular. This is partly explained by the fact that, contrary to parametric approaches, workable algorithms for the minimization of functional costs are not as well-developed. Indeed, nonparametric models involve function representations that depend on an infinite number of parameters. This is a challenge not only because optimal function descriptions can become computationally intractable but, more importantly, because finding these optimal representations is itself intractable.

This work represents the first attempt at comprehensively addressing this intractability. In particular, we have proposed solving general convex expected risk minimization problems over a Hilbert space that defines nonparametric regression functions in a way that guarantees the model order of the learned function does not grow unnecessarily large. In doing so, we addressed challenges (i) - (ii) as follows: we considered the generalization of stochastic gradient descent to the kernelized expected risk setting and we compressed the learned decision function in a way that guarantees stochastic descent by tuning a greedy sparse approximation error criterion to the underlying optimization sequence. The result is an almost-sure convergent function sequence with moderate complexity that is able to operate in true online settings.

Indeed, our experiments have shown that POLK performs comparably to batch kernel methods in terms of accuracy, while its model complexity is reduced by orders of magnitude. Additionally, we observe state-of-the-art performance in terms of test-set accuracy relative to the number of samples processed. Such performance is key to achieving reasonable performance in many applications of interest, e.g., when learning on robotics platforms operating in unknown environments. In this case, the online nature of the problem is intrinsic and due to a lack of prior information on their operating domain (Koppel et al., 2016).

On the other hand, it must be noted that POLK, and even batch kernel methods, for certain large-scale supervised learning tasks, have not met the high bar of asymptotic

test set accuracy set forth by batch approaches to deep learning (Krizhevsky et al., 2012). We believe this discrepancy is on account of the single-layer nature of the nonparametric regressor, which is tied to the choice of reproducing kernel used in our experiments. Of course, more complicated multi-layer composite kernels may be used, based on the fact that a composition and positive linear combination of kernels is still a kernel (Theodoridis, 2015, Ch. 11). However, the scalable development of online nonparametric methods based on such composite kernels is not straight-forward, and left to future work.

Appendix A. Technical Results

Next we establish an auxiliary result needed to prove Theorems 2 and 3 which bounds the magnitude of the iterates of Algorithm 2 in the Hilbert norm.

Proposition 7 *Let Assumptions 1-4 hold and denote $\{f_t\}$ as the sequence generated by Algorithm 2 with $f_0 = 0$. Further denote f^* as the optimum defined by* (1)*. Both quantities are bounded by the constant $K := CX/\lambda$ in Hilbert norm for all t as*

$$\|f_t\|_{\mathcal{H}} \leq \frac{CX}{\lambda}\,, \qquad \|f^*\|_{\mathcal{H}} \leq \frac{CX}{\lambda} \tag{36}$$

Proof: First, since we repeatedly use the Cauchy-Schwartz inequality together with the reproducing kernel property in the following analysis, we here note that for all $g \in \mathcal{H}$, $|g(\mathbf{x}_t)| \leq |\langle g, \kappa(\mathbf{x}_t, \cdot)\rangle_{\mathcal{H}}| \leq X\|g\|_{\mathcal{H}}$. Now, consider the magnitude of f_1 in the Hilbert norm, given $f_0 = 0$

$$\begin{aligned}\|f_1\|_{\mathcal{H}} &= \left\|\mathcal{P}_{\mathcal{H}_{\mathbf{D}_1}}\left[\eta_0 \nabla_f \ell(0, y_0)\right]\right\|_{\mathcal{H}} \\ &\leq \eta_0 \|\nabla_f \ell(0, y_0)\|_{\mathcal{H}} \leq \eta_0 |\ell'(0, y_0)| \|\kappa(\mathbf{x}_0, \cdot)\|_{\mathcal{H}} \\ &\leq \eta_0 CX < \frac{CX}{\lambda}\end{aligned} \tag{37}$$

The first equality comes from substituting in $f_0 = 0$ and the second inequality comes from the definition of optimality condition of the projection operator and the homogeneity of the Hilbert norm, and the third uses the derivation of the functional stochastic gradient in (10) with the Cauchy-Schwartz inequality. Lastly, we make use of Assumptions 1 and 2 to bound the scalar derivative ℓ' using the Lipschitz constant, and the boundedness of the kernel map [cf. (26)]. The final strict inequality in (37) comes from applying the step-size condition $\eta_0 < 1/\lambda$.

Now we consider the induction step. Given the induction hypothesis $\|f_t\|_{\mathcal{H}} \leq CX/\lambda$, consider the magnitude of the iterate at the time $t+1$ as

$$\begin{aligned}\|f_{t+1}\|_{\mathcal{H}} &= \left\|\mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}}\left[(1-\eta_t\lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t)\right]\right\|_{\mathcal{H}} \\ &\leq \|(1-\eta_t\lambda) f_t - \eta_t \nabla_f \ell(f_t(\mathbf{x}_t), y_t)\|_{\mathcal{H}} \\ &\leq (1-\eta_t\lambda)\|f_t\| + \eta_t \|\nabla_f \ell(f_t(\mathbf{x}_t), y_t)\|_{\mathcal{H}}\,,\end{aligned} \tag{38}$$

where we have applied the non-expansion property of the projection operator for the first inequality on the right-hand side of (38), and the triangle inequality for the second. Now,

apply the induction hypothesis $\|f_t\|_{\mathcal{H}} \leq CX/\lambda$ to the first term on the right-hand side of (38), and the chain rule together with the triangle inequality to the second to obtain

$$\begin{aligned}\|f_{t+1}\|_{\mathcal{H}} &\leq (1-\eta_t\lambda)\frac{CX}{\lambda} + \eta_t|\ell'(f_t(\mathbf{x}_t),y_t)|\|\kappa(\mathbf{x}_t,\cdot)\|_{\mathcal{H}} \\ &\leq (\frac{1}{\lambda}-\eta_t)CX + \eta_t CX = \frac{CX}{\lambda} \end{aligned} \tag{39}$$

where we have made use of Assumptions 1 and 2 to bound the scalar derivative ℓ' using the Lipschitz constant, and the boundedness of the kernel map [cf. (26)] as in the base case for f_1, as well as the fact that $\eta_t < 1/\lambda$. The same bound holds for f^* by applying the result of Section V-B of Kivinen et al. (2004) with $m \to \infty$.

■

Next we introduce a proposition which quantifies the error due to our sparse stochastic projection scheme in terms of the ratio of the sparse approximation budget to the algorithm step-size.

Proposition 8 *Given independent identical realizations $(\mathbf{x}_t, y_t)$ of the random pair $(\mathbf{x}, y)$, the difference between the projected stochastic functional gradient and the stochastic functional gradient of the regularized instantaneous risk defined by* (23) *and* (22)*, respectively, is bounded for all t as*

$$\|\tilde{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t) - \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\|_{\mathcal{H}} \leq \frac{\epsilon_t}{\eta_t} \tag{40}$$

where $\eta_t > 0$ denotes the algorithm step-size and $\epsilon_t > 0$ is the approximation budget parameter of Algorithm 1.

Proof: Consider the square-Hilbert-norm difference of $\tilde{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)$ and $\hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)$ defined in (22) and (23), respectively,

$$\begin{aligned}&\|\tilde{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t) - \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\|_{\mathcal{H}}^2 \\ &\quad= \left\| \Big(f_t - \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}}\Big[f_t - \eta_t \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\Big]\Big)/\eta_t - \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\right\|_{\mathcal{H}}^2 \end{aligned} \tag{41}$$

Multiply and divide $\hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)$, the last term, by η_t, and reorder terms to write

$$\begin{aligned}&\left\| \Big(f_t - \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}}\Big[f_t - \eta_t \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\Big]\Big)/\eta_t - \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\right\|_{\mathcal{H}}^2 \\ &\quad= \left\| \frac{1}{\eta_t}\Big(f_t - \eta_t \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\Big) - \frac{1}{\eta_t}\mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}}\Big[f_t - \eta_t \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\Big]\right\|_{\mathcal{H}}^2 \\ &\quad= \frac{1}{\eta_t^2}\|\tilde{f}_{t+1} - f_{t+1}\|_{\mathcal{H}}^2 \end{aligned} \tag{42}$$

where we have substituted the definition of $\tilde{f}_{t+1}$ and f_{t+1} in (17) and (15), respectively, and pulled the nonnegative scalar η_t outside the norm. Now, observe that the KOMP residual stopping criterion in Algorithm 1 is $\|\tilde{f}_{t+1} - f_{t+1}\|_{\mathcal{H}} \leq \epsilon_t$, which we may apply to the last term on the right-hand side of (42) to conclude (40).

■

Lemma 9 *(Stochastic Descent) Consider the sequence generated $\{f_t\}$ by Algorithm 2 with $f_0 = 0$. Under Assumptions 1-4, the following expected descent relation holds.*

$$\mathbb{E}\left[\|f_{t+1} - f^*\|_{\mathcal{H}}^2 \,\middle|\, \mathcal{F}_t\right] \leq \|f_t - f^*\|_{\mathcal{H}}^2 - 2\eta_t[R(f_t) - R(f^*)] + 2\epsilon_t\|f_t - f^*\|_{\mathcal{H}} + \eta_t^2\sigma^2 \,. \tag{43}$$

Proof: Begin by considering the square of the Hilbert-norm difference between f_{t+1} and f^* defined by (1), and expand the square to write

$$\begin{aligned}\|f_{t+1} - f^*\|_{\mathcal{H}}^2 &= \|f_t - \eta_t\tilde{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\|_{\mathcal{H}}^2 \\ &= \|f_t - f^*\|_{\mathcal{H}}^2 - 2\eta_t\langle f_t - f^*, \tilde{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\rangle_{\mathcal{H}} + \eta_t^2\|\tilde{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\|_{\mathcal{H}}^2 \end{aligned} \tag{44}$$

Add and subtract the gradient of the regularized instantaneous risk $\hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)$ defined in (22) to the second term on the right-hand side of (44) to obtain

$$\begin{aligned}\|f_{t+1} - f^*\|_{\mathcal{H}}^2 = \|f_t - f^*\|_{\mathcal{H}}^2 &- 2\eta_t\langle f_t - f^*, \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\rangle_{\mathcal{H}} \\ &- 2\eta_t\langle f_t - f^*, \tilde{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t) - \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\rangle_{\mathcal{H}} + \eta_t^2\|\tilde{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\|_{\mathcal{H}}^2 \end{aligned} \tag{45}$$

We deal with the third term on the right-hand side of (45), which represents the directional error associated with the sparse stochastic projections, by applying the Cauchy-Schwartz inequality together with Proposition 8 to obtain

$$\begin{aligned}\|f_{t+1} - f^*\|_{\mathcal{H}}^2 \leq \|f_t - f^*\|_{\mathcal{H}}^2 &- 2\eta_t\langle f_t - f^*, \hat{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\rangle_{\mathcal{H}} \\ &+ 2\epsilon_t\|f_t - f^*\|_{\mathcal{H}} + \eta_t^2\|\tilde{\nabla}_f \ell(f_t(\mathbf{x}_t),y_t)\|_{\mathcal{H}}^2 \end{aligned} \tag{46}$$

Now compute the expectation of (46) conditional on the algorithm history $\mathcal{F}_t$

$$\mathbb{E}\left[\|f_{t+1} - f^*\|_{\mathcal{H}}^2 \,\middle|\, \mathcal{F}_t\right] \leq \|f_t - f^*\|_{\mathcal{H}}^2 - 2\eta_t\langle f_t - f^*, \nabla_f R(f_t)\rangle_{\mathcal{H}} + 2\epsilon_t\|f_t - f^*\|_{\mathcal{H}} + \eta_t^2\sigma^2 \,, \tag{47}$$

where we have applied the fact that the stochastic functional gradient in (22) is an unbiased estimator [cf. (25)] for the functional gradient of the expected risk in (1), as well as the fact that the variance of the functional projected stochastic gradient is finite stated in (28) (Assumption 4). Observe that since $R(f)$ is an expectation of a convex function, it is also convex, which allows us to write

$$R(f_t) - R(f^*) \leq \langle f_t - f^*, \nabla_f R(f_t)\rangle_{\mathcal{H}} \,, \tag{48}$$

which we substitute into the second term on the right-hand side of the relation given in (47) to obtain

$$\mathbb{E}\left[\|f_{t+1} - f^*\|_{\mathcal{H}}^2 \,\middle|\, \mathcal{F}_t\right] \leq \|f_t - f^*\|_{\mathcal{H}}^2 - 2\eta_t[R(f_t) - R(f^*)] + 2\epsilon_t\|f_t - f^*\|_{\mathcal{H}} + \eta_t^2\sigma^2 \,. \tag{49}$$

Thus the claim in Lemma 9 is valid. ■

Appendix B. Proof of Theorem 2

Apply the iterate bound stated in Proposition 7 to the third term on the right-hand side of (43) (Lemma 9) to write

$$\mathbb{E}\left[\|f_{t+1}-f^*\|_{\mathcal{H}}^2 \,\middle|\, \mathcal{F}_t\right] \leq \|f_t-f^*\|_{\mathcal{H}}^2 - 2\eta_t[R(f_t)-R(f^*)] + \eta_t^2\left(\frac{4CX}{\lambda}+\sigma^2\right) . \tag{50}$$

where we also have applied the approximation budget condition $\epsilon_t = \eta_t^2$. We use the relation in (50) to construct a martingale difference sequence. In particular, define the nonnegative stochastic processes α_t and β_t as

$$\alpha_t = \|f_t-f^*\|_{\mathcal{H}}^2 + \left(\frac{4CX}{\lambda}+\sigma^2\right)\sum_{u=t}^{\infty}\eta_u^2 , \qquad \beta_t = 2\eta_t[R(f_t)-R(f^*)] \tag{51}$$

Observe that α_t is finite almost surely, since $\sum_{u=t}^{\infty}\eta_u^2 \leq \sum_{u=0}^{\infty}\eta_u^2$. Given the definitions of α_t and β_t in (51), we may write

$$\mathbb{E}\left[\alpha_{t+1}\,\middle|\,\mathcal{F}_t\right] \leq \alpha_t - \beta_t , \tag{52}$$

together with the fact that α_t and β_t are nonnegative, whereby the conditions of the Supermartingale Convergence Theorem (Solo and Kong, 1995) are satisfied. Therefore, we obtain that (i) α_t has a finite limit almost surely; and (ii) the series $\sum_{t=1}^{\infty}\beta_t < \infty$ is almost surely finite. The later result, taken together with the non-summability of the step-sizes stated in (29), implies that the almost surely the limit infimum of $R(f_t)-R(f^*)$ is null, i.e.

$$\liminf_{t\to\infty} R(f_t)-R(f^*) = 0 \qquad \text{a.s.} \tag{53}$$

Now, using the consequence of the Supermartingale Convergence Theorem, α_t almost surely has a limit. Observe that the sum $\sum_{u=t}^{\infty}\eta_u^2$ is a deterministic quantity whose limit is null (we sum over less and less terms over time, asymptotically summing over zero terms). Taken with the finiteness of the limit of α_t, we conclude

$$\lim_{t\to\infty}\|f_t-f^*\|_{\mathcal{H}}^2 = 0 \qquad \text{a.s.} \tag{54}$$

Appendix C. Proof of Theorem 3

Proof: The use of the regularizing term $(\lambda/2)\|f\|_{\mathcal{H}}^2$ in (1) implies that the regularized expected risk is λ-strongly convex with respect to $f \in \mathcal{H}$, which allows us to write

$$\frac{\lambda}{2}\|f_t-f^*\|_{\mathcal{H}}^2 \leq R(f_t)-R(f^*) \tag{55}$$

Substituting the relation (55) into the second term on the right-hand side of the expected descent relation stated in Lemma 9, with constant step-size $\eta_t = \eta$ and approximation budget $\epsilon_t = \epsilon$, yields

$$\mathbb{E}\left[\|f_{t+1}-f^*\|_{\mathcal{H}}^2 \,\middle|\, \mathcal{F}_t\right] \leq (1-\eta\lambda)\|f_t-f^*\|_{\mathcal{H}}^2 + 2\epsilon\|f_t-f^*\|_{\mathcal{H}} + \eta^2\sigma^2 . \tag{56}$$

We use the expression in (56) to construct a stopping stochastic process, which tracks the suboptimality of $\|f_t-f^*\|_{\mathcal{H}}^2$ until it reaches a specific threshold. In doing so, we obtain convergence to a neighborhood. We aim to define a stochastic process δ_t that qualifies as a supermartingale, i.e. $\mathbb{E}\left[\delta_{t+1}\,\middle|\,\mathcal{F}_t\right] \leq \delta_t$. To do so, consider (56) and solve for the appropriate threshold by analyzing when the following holds true

$$\mathbb{E}\left[\|f_{t+1}-f^*\|_{\mathcal{H}}^2 \,\middle|\, \mathcal{F}_t\right] \leq (1-\eta\lambda)\|f_t-f^*\|_{\mathcal{H}}^2 + 2\epsilon\|f_t-f^*\|_{\mathcal{H}} + \eta^2\sigma^2 \leq \|f_t-f^*\|_{\mathcal{H}}^2 . \tag{57}$$

Re-arrange the above expression to obtain the sufficient condition

$$-\eta\lambda\|f_t-f^*\|_{\mathcal{H}}^2 + 2\epsilon\|f_t-f^*\|_{\mathcal{H}} + \eta^2\sigma^2 \leq 0 . \tag{58}$$

Observe that (58) defines a quadratic polynomial in $\|f_t-f^*\|_{\mathcal{H}}$, which, using the quadratic formula, has roots

$$\|f_t-f^*\|_{\mathcal{H}} = \frac{-2\epsilon \pm \sqrt{4\epsilon^2-(-4\lambda\eta)(\eta^2\sigma^2)}}{-2\lambda\eta} = \frac{\epsilon \pm \sqrt{\epsilon^2+\lambda\eta^3\sigma^2}}{\lambda\eta} \tag{59}$$

The quadratic polynomnial defined by (58) opens downward, and $\|f_t-f^*\|_{\mathcal{H}} \geq 0$, so we focus on the positive root, substituting the approximation budget selection $\epsilon = K\eta^{3/2}$ to define the radius of convergence as

$$\Delta := \frac{\epsilon+\sqrt{\epsilon^2+\lambda\eta^3\sigma^2}}{\lambda\eta} = \frac{\sqrt{\eta}}{\lambda}\left(K+\sqrt{K^2+\lambda\sigma^2}\right) \tag{60}$$

The definition (60) allows us to construct a stopping process. In particular, define the stochastic process δ_t as

$$\delta_t = \|f_t-f^*\|_{\mathcal{H}}\mathbb{1}\left\{\min_{u\leq t} -\eta\lambda\|f_u-f^*\|_{\mathcal{H}}^2 + 2\epsilon\|f_u-f^*\|_{\mathcal{H}} + \eta^2\sigma^2 > \Delta\right\} \tag{61}$$

where $\mathbb{1}\{E\}$ denotes the indicator process of event $E \in \mathcal{F}_t$. Note that $\delta_t \geq 0$ for all t, since both $\|f_t-f^*\|_{\mathcal{H}}$ and the indicator function are nonnegative. Observe that, given the definition (61), either $\min_{u\leq t} -\eta\lambda\|f_u-f^*\|_{\mathcal{H}}^2 + 2\epsilon\|f_u-f^*\|_{\mathcal{H}} + \eta^2\sigma^2 > \Delta$ holds, in which case we may compute the square root of the condition in (57) to write

$$\mathbb{E}[\delta_{t+1}\,|\,\mathcal{F}_t] \leq \delta_t \tag{62}$$

Alternatively, $\min_{u\leq t} -\eta\lambda\|f_u-f^*\|_{\mathcal{H}}^2 + 2\epsilon\|f_u-f^*\|_{\mathcal{H}} + \eta^2\sigma^2 \leq \Delta$, in which case the indicator function is null for all subsequent times, due to the use of the minimum inside the indicator in the definition of (61). Thus in either case, (62) holds, which implies that δ_t converges almost surely to null, which, as a consequence we obtain the fact that either $\lim_{t\to\infty}\|f_t-f^*\|_{\mathcal{H}} - \Delta = 0$ or the indicator function is null for large t, i.e. $\lim_{t\to\infty}\mathbb{1}\{\min_{u\leq t} -\eta\lambda\|f_u-f^*\|_{\mathcal{H}}^2 + 2\epsilon\|f_u-f^*\|_{\mathcal{H}} + \eta^2\sigma^2 > \Delta\} = 0$ almost surely. Therefore, we obtain that

$$\liminf_{t\to\infty}\|f_t-f^*\|_{\mathcal{H}} \leq \Delta = \frac{\sqrt{\eta}}{\lambda}\left(K+\sqrt{K^2+\lambda\sigma^2}\right) \qquad \text{a.s.} \tag{63}$$

which is as stated in Theorem 3. ■

Appendix D. Proofs Leading to Theorem 4

Before proving Theorem D, we present a lemma which allows us to relate the stopping criterion of our sparsification procedure to a Hilbert subspace distance.

Lemma 10 *Define the distance of an arbitrary feature vector* $\mathbf{x}$ *evaluated by the feature transformation* $\phi(\mathbf{x}) = \kappa(\mathbf{x}, \cdot)$ *to* $\mathcal{H}_{\mathbf{D}} = span\{\kappa(\mathbf{d}_n, \cdot)\}_{n=1}^{M}$*, the subspace of the Hilbert space spanned by a dictionary* $\mathbf{D}$ *of size* M*, as*

$$dist(\kappa(\mathbf{x}, \cdot), \mathcal{H}_{\mathbf{D}}) = \min_{f \in \mathcal{H}_{\mathbf{D}}} \|\kappa(\mathbf{x}, \cdot) - \mathbf{v}^T \boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\|_{\mathcal{H}} \,. \tag{64}$$

This set distance simplifies to following least-squares projection when $\mathbf{D} \in \mathbb{R}^{p \times M}$ *is fixed*

$$dist(\kappa(\mathbf{x}, \cdot), \mathcal{H}_{\mathbf{D}}) = \left\|\kappa(\mathbf{x}, \cdot) - [\mathbf{K}_{\mathbf{D},\mathbf{D}}^{-1} \boldsymbol{\kappa}_{\mathbf{D}}(\mathbf{x})]^T \boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\right\|_{\mathcal{H}} . \tag{65}$$

Proof: The distance to the subspace $\mathcal{H}_{\mathbf{D}}$ is defined as

$$\text{dist}(\kappa(\mathbf{x}, \cdot), \mathcal{H}_{\mathbf{D}_t}) = \min_{f \in \mathcal{H}_{\mathbf{D}}} \|\kappa(\mathbf{x}, \cdot) - \mathbf{v}^T \boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\|_{\mathcal{H}} = \min_{\mathbf{v} \in \mathbb{R}^M} \|\kappa(\mathbf{x}, \cdot) - \mathbf{v}^T \boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\|_{\mathcal{H}} \,, \tag{66}$$

where the first equality comes from the fact that the dictionary $\mathbf{D}$ is fixed, so $\mathbf{v} \in \mathbb{R}^M$ is the only free parameter. Now plug in the minimizing weight vector $\tilde{\mathbf{v}}^* = \mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t}^{-1} \boldsymbol{\kappa}_{\mathbf{D}_t}(\mathbf{x}_t)$ into (66) which is obtained in an analogous manner to the logic which yields (19) - (20). Doing so simplifies (66) to the following

$$\text{dist}(\kappa(\mathbf{x}_t, \cdot), \mathcal{H}_{\mathbf{D}_t}) = \left\|\kappa(\mathbf{x}_t, \cdot) - [\mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t}^{-1} \boldsymbol{\kappa}_{\mathbf{D}_t}(\mathbf{x}_t)]^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot)\right\|_{\mathcal{H}} . \tag{67}$$

■

D.1. Proof of Theorem 4

Proof: The proof proceeds by the following logic. We begin by considering the model order at two arbitrary subsequent iterates of Algorithm 2, and reduce model order growth at a given time to a criterion involving the approximation error γ_{M_t+1} associated with removing the most recent feature vector $\mathbf{x}_t$, and then analyze the conditions under which this simplified criterion is not satisfied for all subsequent times, meaning that the model order does not grow beyond a certain point. To do so, we prove that this quantity reduces to a weighted set distance to the Hilbert subspace $\mathcal{H}_{\mathbf{D}_t}$ defined by dictionary $\mathbf{D}_t$, and thus we are able to invoke point-set topological properties of the compact feature space $\mathcal{X}$, specifically, its packing number, which guarantee that the number of dictionary elements remains finite, in a manner similar to the proof of Theorem 3.1 in Engel et al. (2004).

Consider the model order of the function iterates f_t and f_{t+1} generated by Algorithm 2 denoted by M_t and M_{t+1}, respectively, at two arbitrary subsequent times t and $t+1$. Assume a constant algorithm step-size η has been chosen such that $\eta < 1/\lambda$ and the approximation budget ϵ satisfies $\epsilon = K\eta^{3/2}$ for some positive scalar $K > 0$. Suppose the model order of the function f_{t+1} is less than or equal to that of f_t, i.e. $M_{t+1} \leq M_t$. This relation holds when the stopping criterion of KOMP (Algorithm 1), stated as $\min_{j=1,\ldots,M_t+1} \gamma_j > \epsilon$, *is*

not satisfied for the kernel dictionary matrix with the newest sample point $\mathbf{x}_t$ appended: $\tilde{\mathbf{D}}_{t+1} = [\mathbf{D}_t; \mathbf{x}_t]$ [cf. (18)], which is of size $M_t + 1$. Thus, the negation of the termination condition of Algorithm 1 must hold for this case, stated as

$$\min_{j=1,\ldots,M_t+1} \gamma_j \leq \epsilon \,. \tag{68}$$

Observe that the left-hand side of (68) lower bounds the approximation error γ_{M_t+1} of removing the most recent feature vector $\mathbf{x}_t$ due to the minimization over j, that is, $\min_{j=1,\ldots,M_t+1} \gamma_j \leq \gamma_{M_t+1}$. Consequently, if $\gamma_{M_t+1} \leq \epsilon$, then (68) holds and the model order does not grow. Thus it suffices to consider γ_{M_t+1}.

The definition of γ_{M_t+1} with the substitution of $\tilde{f}_{t+1}$ in (17) allows us to write

$$\begin{aligned}\gamma_{M_t+1} &= \min_{\mathbf{u} \in \mathbb{R}^{M_t}} \left\|(1-\eta\lambda) f_t - \eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot) - \sum_{k \in \mathcal{I} \setminus \{M_t+1\}} u_k \kappa(\mathbf{d}_k, \cdot)\right\|_{\mathcal{H}} \\ &= \min_{\mathbf{u} \in \mathbb{R}^{M_t}} \left\|(1-\eta\lambda) \sum_{k \in \mathcal{I} \setminus \{M_t+1\}} w_k \kappa(\mathbf{d}_k, \cdot) - \eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot) - \sum_{k \in \mathcal{I} \setminus \{M_t+1\}} u_k \kappa(\mathbf{d}_k, \cdot)\right\|_{\mathcal{H}} ,\end{aligned} \tag{69}$$

where we denote the k^{th} column of $\mathbf{D}_t$ as $\mathbf{d}_k$. The minimal error is achieved by considering the square of the expression inside the minimization and expanding terms to obtain

$$\begin{aligned}&\left\|(1-\eta\lambda) \sum_{k \in \mathcal{I} \setminus \{M_t+1\}} w_k \kappa(\mathbf{d}_k, \cdot) - \eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot) - \sum_{k \in \mathcal{I} \setminus \{M_t+1\}} u_k \kappa(\mathbf{d}_k, \cdot)\right\|_{\mathcal{H}}^2 \\ &= (1-\eta\lambda)^2 \mathbf{w}^T \mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t} \mathbf{w} + \eta^2 \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)^2 \kappa(\mathbf{x}_t, \mathbf{x}_t) + \mathbf{u}^T \mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t} \mathbf{u} \\ &\quad - 2(1-\eta\lambda)\eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)^2 \mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\mathbf{x}_t) + 2\eta^2 \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t) \mathbf{u}^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\mathbf{x}_t) - 2(1-\eta\lambda) \mathbf{w}^T \mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t} \mathbf{u}.\end{aligned} \tag{70}$$

To obtain the minimum, we compute the stationary solution of (70) with respect to $\mathbf{u} \in \mathbb{R}^{M_t}$ and solve for the minimizing $\tilde{\mathbf{u}}^*$, which in a manner similar to the logic in (19) - (20), is given as

$$\tilde{\mathbf{u}}^* = (1-\eta\lambda)\mathbf{w} - \eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t) \mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t}^{-1} \boldsymbol{\kappa}_{\mathbf{D}_t}(\mathbf{x}_t) \,. \tag{71}$$

Plug $\tilde{\mathbf{u}}^*$ in (71) into the expression in (69) and using the short-hand notation $f_t(\cdot) = \mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot)$ and $\sum_k u_k \kappa(\mathbf{d}_k, \cdot) = \mathbf{u}^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot)$. Doing so simplifies (69) to

$$\begin{aligned}&\left\|(1-\eta\lambda)\mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot) - \eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot) - \mathbf{u}^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot)\right\|_{\mathcal{H}} \\ &= \Big\|(1-\eta\lambda)\mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot) - \eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot) \\ &\quad - [(1-\eta\lambda)\mathbf{w} - \eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t) \mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t}^{-1} \boldsymbol{\kappa}_{\mathbf{D}_t}(\mathbf{x}_t)]^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot)\Big\|_{\mathcal{H}}.\end{aligned} \tag{72}$$

The above expression may be simplified by cancelling like terms $(1-\eta\lambda)\mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot)$ and pulling out a common factor of $\eta|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)|$ outside the norm as

$$\begin{aligned}&\Big\| - \eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot) - \eta \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)[\mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t}^{-1} \boldsymbol{\kappa}_{\mathbf{D}_t}(\mathbf{x}_t)]^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot)\Big\|_{\mathcal{H}} \\ &\qquad = \eta|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)| \left\|\kappa(\mathbf{x}_t, \cdot) - [\mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t}^{-1} \boldsymbol{\kappa}_{\mathbf{D}_t}(\mathbf{x}_t)]^T \boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot)\right\|_{\mathcal{H}} .\end{aligned} \tag{73}$$

Notice that the right-hand side of (73) may be identified as the distance to the subspace $\mathcal{H}_{\mathbf{D}_t}$ in (67) defined in Lemma 10 scaled by a factor of $\eta|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)|$. We may upper-bound the right-hand side of (73) as

$$\eta|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)|\left\|\kappa(\mathbf{x}_t, \cdot) - [\mathbf{K}_{\mathbf{D}_t,\mathbf{D}_t}^{-1}\boldsymbol{\kappa}_{\mathbf{D}_t}(\mathbf{x}_t)]^T\boldsymbol{\kappa}_{\mathbf{D}_t}(\cdot)\right\|_{\mathcal{H}} \leq \eta|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)|\text{dist}(\kappa(\mathbf{x}_t, \cdot), \mathcal{H}_{\mathbf{D}_t}) \tag{74}$$

where we have applied (65) regarding the definition of the subspace distance on the right-hand side of (74) to replace the Hilbert-norm term. Now, when the KOMP stopping criterion is violated, i.e., (68) holds, which implies $\gamma_{M_t+1} \leq \epsilon$. Therefore, the right-hand side of (74) is upper-bounded by ϵ, which we select as $\epsilon = K\eta^{3/2}$.

$$\text{dist}(\kappa(\mathbf{x}_t, \cdot), \mathcal{H}_{\mathbf{D}_t}) \leq \frac{K\sqrt{\eta}}{|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)|}\ , \tag{75}$$

where we have divided both sides by $|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)|$ and cancelled out a common factor of η. Observe that if (75) holds, then $\gamma_{M_t+1} \leq \epsilon$ holds, but since $\gamma_{M_t+1} \geq \min_j \gamma_j$, we may conclude that (68) is satisfied. Consequently the model order at the subsequent step does not grow $M_{t+1} \leq M_t$ whenever (75) is valid.

Now, let's take the contrapositive of the preceding expressions to observe that growth in the model order ($M_{t+1} = M_t + 1$) implies that the condition

$$\text{dist}(\kappa(\mathbf{x}_t, \cdot), \mathcal{H}_{\mathbf{D}_t}) > \frac{K\sqrt{\eta}}{|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)|} \tag{76}$$

holds. Therefore, each time a new point is added to the model, the corresponding kernel function is guaranteed to be at least a distance of $\frac{K\sqrt{\eta}}{|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)|}$ from every other kernel function in the current model.

By the C-Lipschitz continuity of the instantaneous loss (Assumption 2): specifically $1/|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)| \geq 1/C$, we can lower-bound the threshold condition in (76) as

$$\frac{K\sqrt{\eta}}{|\ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)|} \geq \frac{K\sqrt{\eta}}{C} \tag{77}$$

Therefore, the KOMP stopping criterion is violated for the newest point whenever distinct dictionary points $\mathbf{d}_k$ and $\mathbf{d}_j$ for $j, k \in \{1, \ldots, M_t\}$, satisfy the condition $\|\phi(\mathbf{d}_j) - \phi(\mathbf{d}_k)\|_2 > \frac{K\sqrt{\eta}}{C}$. We shall now proceed in a manner similar to the proof of Theorem 3.1 in Engel et al. (2004). Since $\mathcal{X}$ is compact and κ is continuous, the range $\phi(\mathcal{X})$ (where $\phi(\mathbf{x}) = \kappa(\mathbf{x}, \cdot)$ for $\mathbf{x} \in \mathcal{X}$) of the kernel transformation of feature space $\mathcal{X}$ is compact. Therefore, the number of balls of radius δ (here, $\delta = \frac{K\sqrt{\eta}}{C}$) needed to cover the set $\phi(\mathcal{X})$ is finite (see, e.g., Anthony and Bartlett (2009)). Therefore, for some finite M^∞, if $M_t = M^\infty$, the left-hand side of (**??**) holds, which implies (68) is true for all t. We conclude that $M_t \leq M^\infty$ for all t. ■

References

Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

Antoine Bordes, Seyda Ertekin, Jason Weston, and Léon Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6(Sep):1579–1619, 2005.

Antoine Bordes, Léon Bottou, and Patrick Gallinari. Sgd-qn: Careful quasi-newton stochastic gradient descent. *The Journal of Machine Learning Research*, 10:1737–1754, 2009.

Léon Bottou. Online algorithms and stochastic approximations. In David Saad, editor, *Online Learning and Neural Networks*. Cambridge University Press, Cambridge, UK, 1998.

S. Boyd and L. Vanderberghe. *Convex Programming*. Wiley, New York, NY, 2004.

P. Brodatz. *Textures: A Photographic Album for Artists and Designers*. Dover, 1966.

Emmanuel J Candes. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346(9):589–592, 2008.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Bo Dai, Bo Xie, Niao He, Yingyu Liang, Anant Raj, Maria-Florina F Balcan, and Le Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pages 3041–3049, 2014.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pages 1646–1654, 2014.

Aymeric Dieuleveut and Francis Bach. Non-parametric stochastic approximation with large step sizes. *arXiv preprint arXiv:1408.0361*, 2014.

Y. Engel, S. Mannor, and R. Meir. The kernel recursive least-squares algorithm. *IEEE Transactions on Signal Processing*, 52(8):2275–2285, Aug 2004. ISSN 1053-587X. doi: 10.1109/TSP.2004.830985.

Theodoros Evgeniou, Massimiliano Pontil, and Tomaso Poggio. Regularization networks and support vector machines. *Advances in computational mathematics*, 13(1):1–50, 2000.

W. J. Fu. Penalized regressions: the bridge versus the lasso. *Journal of Computational and Graphical Statistics*, 7(3):397–416, 1998.

Simon Haykin. Neural networks: A comprehensive foundation. *Macmillan College Publishing Company*, 1994.

P. Honeine. Online kernel principal component analysis: A reduced-order model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1814–1826, 2012.

Thorsten Joachims and Chun-Nam John Yu. Sparse kernel svms via cutting-plane training. *Machine Learning*, 76(2-3):179–193, 2009.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

S Sathiya Keerthi, Olivier Chapelle, and Dennis DeCoste. Building support vector machines with reduced classifier complexity. *Journal of Machine Learning Research*, 7(Jul):1493–1515, 2006.

George Kimeldorf and Grace Wahba. Some results on tchebycheffian spline functions. *Journal of mathematical analysis and applications*, 33(1):82–95, 1971.

J. Kivinen, A. J. Smola, and R. C. Williamson. Online Learning with Kernels. *IEEE Transactions on Signal Processing*, 52:2165–2176, August 2004. doi: 10.1109/TSP.2004.830991.

A. Koppel, J. Fink, G. Warnell, E. Stump, and A. Ribeiro. Online learning for characterizing unknown environments in ground robotic vehicle models. In *Proc. Int. Conf. Intelligent Robots and Systems*, volume (to appear). Daejeon, Korea, Oct 9 - 14 2016.

Alec Koppel, Garrett Warnell, Ethan Stump, and A. Ribeiro. Parsimonious online kernel learning via sparse projections in function space. In *2017 IEEE Proc. Int. Conf. Acoust. Speech Signal Process.*, March 5-9 2017.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Trung Le, Tu Nguyen, Vu Nguyen, and Dinh Phung. Dual space gradient descent for online learning. In *Advances in Neural Information Processing Systems*, pages 4583–4591, 2016a.

Trung Le, Vu Nguyen, Tu Dinh Nguyen, and Dinh Phung. Nonparametric budgeted stochastic gradient descent. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 654–662, 2016b.

Yann Lecun and Corinna Cortes. The MNIST database of handwritten digits. *Database*, 2009. URL `http://yann.lecun.com/exdb/mnist/`.

T. Leung and J. Malik. Representing and Recognizing the Visual Appearence of Materials using Three-dimensional Textons. *International Journal of Computer Vision*, 43(1):29–44, 1999.

Jun-Bao Li, Shu-Chuan Chu, and Jeng-Shyang Pan. *Kernel Learning Algorithms for Face Recognition*. Springer, 2014.

Weifeng Liu, Puskal P Pokharel, and Jose C Principe. The kernel least-mean-square algorithm. *Signal Processing, IEEE Transactions on*, 56(2):543–554, 2008.

Jing Lu, Steven CH Hoi, Jialei Wang, Peilin Zhao, and Zhi-Yong Liu. Large scale online kernel learning. *The Journal of Machine Learning Research*, 17(1):1613–1655, 2016.

Julien Mairal, Julien Mairal, Michael Elad, Michael Elad, Guillermo Sapiro, and Guillermo Sapiro. Sparse representation for color image restoration. In *the IEEE Trans. on Image Processing*, pages 53–69. ITIP, 2007.

Aryan Mokhtari and Alejandro Ribeiro. Res: Regularized stochastic bfgs algorithm. *Signal Processing, IEEE Transactions on*, 62(23):6089–6104, 2014.

Aryan Mokhtari and Alejandro Ribeiro. Global convergence of online limited memory bfgs. *Journal of Machine Learning Research*, 16:3151–3181, 2015. URL `http://jmlr.org/papers/v16/mokhtari15a.html`.

Shayan Mukherjee and Shree K Nayar. Automatic generation of rbf networks using wavelets. *Pattern Recognition*, 29(8):1369–1383, 1996.

Klaus-Robert Müller, Tülay Adali, Kenji Fukumizu, Jose C. Principe, and Sergios Theodoridis. Special issue on advances in kernel-based learning for signal processing [from the guest editors]. *IEEE Signal Process. Mag.*, 30(4):14–15, 2013. doi: 10.1109/MSP.2013.2253031. URL `http://dx.doi.org/10.1109/MSP.2013.2253031`.

K. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT press, 2012.

Deanna Needell, Joel Tropp, and Roman Vershynin. Greedy signal recovery review. In *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*, pages 1048–1050. IEEE, 2008.

Jorge Nocedal. Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35(151):773–773, 1980. ISSN 0025-5718.

Vladimir Norkin and Michiel Keyzer. On stochastic optimization and statistical learning in reproducing kernel hilbert spaces by support vector machines (svm). *Informatica*, 20(2):273–292, 2009.

Y. Pati, R. Rezaiifar, and P.S. Krishnaprasad. Orthogonal Matching Pursuit: Recursive Function Approximation with Applications to Wavelet Decomposition. In *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, 1993.

Massimiliano Pontil. A note on different covering numbers in learning theory. *Journal of Complexity*, 19(5):665–671, 2003.

Massimiliano Pontil, Yiming Ying, and Ding xuan Zhou. Error analysis for online gradient descent algorithms in reproducing kernel hilbert spaces. Technical report, University College London, 2005.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*, pages 1313–1320, 2009.

Cédric Richard, José Carlos M Bermudez, and Paul Honeine. Online prediction of time series data with kernels. *IEEE Transactions on Signal Processing*, 57(3):1058–1067, 2009.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *Ann. Math. Statist.*, 22(3):400–407, 09 1951. doi: 10.1214/aoms/1177729586.

Geoffrey Sampson, Robin Haigh, and Eric Atwell. Natural language analysis by stochastic optimization: A progress report on project april. *J. Exp. Theor. Artif. Intell.*, 1(4): 271–287, October 1990. ISSN 0952-813X. doi: 10.1080/09528138908953710. URL http://dx.doi.org/10.1080/09528138908953710.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *arXiv preprint arXiv:1309.2388*, 2013.

Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. *Subseries of Lecture Notes in Computer Science Edited by JG Carbonell and J. Siekmann*, page 416, 2001.

Nicol N Schraudolph, Jin Yu, and Simon Günter. A stochastic quasi-newton method for online convex optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 436–443, 2007.

Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Learnability, stability and uniform convergence. *Journal of Machine Learning Research*, 11(Oct):2635–2670, 2010.

Konstantinos Slavakis, Pantelis Bouboulis, and Sergios Theodoridis. Online learning in reproducing kernel hilbert spaces. *Signal Processing Theory and Machine Learning*, pages 883–987, 2013.

Victor Solo and X. Kong. *Adaptive Signal Processing Algorithms: Stability and Performance*. Prentice-Hall information and system sciences series. Prentice Hall, 1995. ISBN 9780135012635. URL https://books.google.com/books?id=3AkfAQAAIAAJ.

Murat Taşan, Gabriel Musso, Tong Hao, Marc Vidal, Calum A MacRae, and Frederick P Roth. selecting causal genes from genome-wide association studies via functionally coherent subnetworks. *Nature methods*, 2014.

Ambuj Tewari and Peter L. Bartlett. Learning theory. In Paulo S.R. Diniz, Johan A.K. Suykens, Rama Chellappa, and Sergios Theodoridis, editors, *Academic Press Library in Signal Processing: Volume 1 Signal Processing Theory and Machine Learning*, volume 1 of *Academic Press Library in Signal Processing*, chapter 14, pages 775–816. Elsevier, 2014. URL http://dx.doi.org/10.1016/B978-0-12-396502-8.00014-0.

Sergios Theodoridis. *Machine learning: a Bayesian and optimization perspective*. Academic Press, 2015.

P. Vincent and Y. Bengio. Kernel matching pursuit. *Machine Learning*, 48(1):165–187, 2002.

Zhuang Wang and Slobodan Vucetic. Online passive-aggressive algorithms on a budget. In *International Conference on Artificial Intelligence and Statistics*, 2010a.

Zhuang Wang, Koby Crammer, and Slobodan Vucetic. Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale svm training. *The Journal of Machine Learning Research*, 13(1):3103–3131, 2012.

R. Wheeden, R.L. Wheeden, and A. Zygmund. *Measure and Integral: An Introduction to Real Analysis*. Chapman & Hall/CRC Pure and Applied Mathematics. Taylor & Francis, 1977. ISBN 9780824764999. URL https://books.google.com/books?id=YDkDmQ_hdmcC.

Christopher KI Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.

Y. Ying and D. X. Zhou. Online regularized classification algorithms. *IEEE Transactions on Information Theory*, 52(11):4775–4788, Nov 2006. ISSN 0018-9448. doi: 10.1109/TIT.2006.883632.

Kai Zhang, Ivor W Tsang, and James T Kwok. Improved nyström low-rank approximation and error analysis. In *Proceedings of the 25th international conference on Machine learning*, pages 1232–1239. ACM, 2008.

Lijun Zhang, Jinfeng Yi, Rong Jin, Ming Lin, and Xiaofei He. Online kernel learning with a near optimal sparsity bound. In *ICML (3)*, pages 621–629, 2013.

Peilin Zhao and Steven Hoi. Bduol: Double updating online learning on a fixed budget. *Machine Learning and Knowledge Discovery in Databases*, pages 810–826, 2012.

Shilei Zhao, Peng Wu, and Yupeng Liu. An online kernel learning algorithm based on orthogonal matching pursuit. *Journal of Software*, 7(Sep):2076–2082, 2012.

Ding-Xuan Zhou. The covering number in learning theory. *Journal of Complexity*, 18(3): 739–767, 2002.

J. Zhu and T. Hastie. Kernel Logistic Regression and the Import Vector Machine. *Journal of Computational and Graphical Statistics*, 14(1):185–205, 2005.

Zeyuan Allen Zhu, Weizhu Chen, Gang Wang, Chenguang Zhu, and Zheng Chen. P-packsvm: Parallel primal gradient descent kernel svm. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 677–686. IEEE, 2009.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

Journal of Machine Learning Research 20 (2019) 1-45 Submitted 11/16; Revised 3/18; Published 2/19

Convergence Rate of a Simulated Annealing Algorithm with Noisy Observations

Clément Bouttier CLEMENT.BOUTTIER@AIRBUS.COM
Aircraft Performance Departement
Airbus Operations
316 route de Blagnac, 31300 Toulouse, France

Ioana Gavra IOANA.GAVRA@MATH.UNIV-TOULOUSE.FR
Institut de Mathématiques de Toulouse
Université Toulouse III
118 route de Narbonne, 31062 Toulouse Cedex 9, France

Editor: Robert McCulloch

Abstract

In this paper we propose a modified version of the simulated annealing algorithm for solving a stochastic global optimization problem. More precisely, we address the problem of finding a global minimizer of a function with noisy evaluations. We provide a rate of convergence and its optimized parametrization to ensure a minimal number of evaluations for a given accuracy and a confidence level close to 1. This work is completed with a set of numerical experimentations and assesses the practical performance both on benchmark test cases and on real world examples.

Keywords: Simulated Annealing, Stochastic Optimization, Markov Process, Convergence Rate, Aircraft Trajectory Optimization

1. Introduction

We are interested in an algorithm that solves the stochastic global optimization problem:

$$\text{Find} \quad x^\star = \arg\min_{x \in E} \mathbb{E}_\omega(U(x,\omega)), \tag{*}$$

where x is a decision variable belonging to some large space E, ω is a random variable and U is the cost, a positive and bounded real valued function. We do not make any assumption on the regularity of U. We only expect it to be rapidly evaluable: typically, U is the result of some short numerical simulation. We do not make any distribution assumption for the random inputs ω themselves but only on the outputs U. We assume the code has some robustness property in the sense that, at some point x, it is either infinite for all ω or bounded uniformly in ω.
This problem is twofold: we must both estimate and minimize the expectation of the cost. A simple and general approach consists in the minimization of a sample average of Monte Carlo estimators:

$$\widehat{\mathbb{E}}_{\omega,N}(U(x,\omega)) := \frac{1}{N}\sum_{i=1}^{N} U(x,\omega_i),$$

©2019 Clément Bouttier and Ioana Gavra.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/16-588.html.

for any given i.i.d. sample $(\omega_i)_{1\leq i\leq N}$ of size N distributed according to the distribution $P_\Omega(x)$ of ω and for any $x \in E$. Such an estimator consistently estimates $\mathbb{E}_\omega(U(x,\omega))$ for any given x. Nevertheless, its accuracy is directly linked to N and thus to the computational effort. One can thus wonder if a computationally efficient procedure using that estimator can return a solution to the initial problem given a certain level of accuracy ϵ.

1.1. Previous Works: Different Types of Algorithms

There were many attempts to solve this stochastic problem across several research communities. We give a brief survey of them in what follows.
In the case E is finite and U takes its values in $[0,1]$, problem (*) is labelled as a "simple regret bandit optimization problem" by the bandit community. Indeed it can be seen as the problem of choosing, among a small finite set of slot machines providing random rewards, the one with the best expected reward by playing a minimal number of times. This is not the classical setting of bandit optimization which usually seeks for the "cumulative regret". As algorithm proposals for the simple regret context often extends cumulative regret concepts (Audibert and Bubeck, 2010), we focus on them first. The Upper Confidence Bound (UCB, Auer et al. 2002) algorithm aimed at building sequences of confidence bounds around the estimated expected cost of each element of the search space. If the space is too large this can be prohibitive. These were several attempts to bypass this issue by adding some assumptions on the regularity of the cost function around its optimum. We can mention HOO (Bubeck et al., 2011) that produced guarantees about the cumulative regret for a continuous Lipschitz cost function with known Lipschitz constant. In the same framework StoSOO (Munos, 2014) relaxed this last assumption. Both algorithms were however not very efficient in practice if the search space is multidimensional. Indeed they still required some uniform exploration of the state space in the first phase. This could lead to numerical difficulties when the dimension was growing. The Adaptive-treed bandit algorithm (Bull et al., 2015) partially solved this issue by adapting the exploration step using a Lipschitz constant per dimension.
These algorithms could all be viewed as stochastic variations around the classical *branch and bound* algorithm (Little et al., 1963), which was extensively studied by the optimization community. We can mention the very popular DIRECT algorithm (Jones et al., 1993), from which the StoSOO procedure was inspired.
Finally, let us mention the computer experiment community that introduced another popular global optimization method for dealing with the stochastic case, the so called Efficient Global Optimization (EGO, Jones et al. 1998) based on expected improvement. The convergence rate of this method was already investigated in Bull (2011) in a noise free context. This algorithm focused on minimizing the number of cost evaluations because it considered a setting where the cost evaluations were very time consuming. As a result, in order to select each evaluation point, it required a higher computational effort and memory storage per iteration than other optimization methods. Such a method could therefore turn out to under-perform in a setting where the computational cost ratio between selection and evaluation was inverted.
A typical algorithm that was known to perform well in the case of time-cheap cost evaluations was the simulated annealing (SA) as mentioned by Locatelli in Horst and Pardalos

(2013): *"The latter algorithms* (mainly EGO) *often outperform SA algorithms from the point of view of the number of function evaluations to reach a given relative accuracy on the standard test functions from Dixon and Szegö (1978), but usually require a higher computational effort per iteration. Typical advantages of SA algorithms are their very mild memory requirements and the small computational effort per iteration. If the cost of a function evaluation is very high, then even a considerable computational effort per iteration may be negligible with respect to the cost of a function evaluation, and algorithms which require few function evaluations are preferable; otherwise, also the computational effort per iteration should be taken into account, and from this point of view SA algorithms are often better than other algorithms."*

However SA algorithms have been designed and extensively studied in a context where the exact cost could be observed. We recall below some basic facts in the noiseless case (Section 1.2) and then present the noisy case which is the setting addressed in this paper (Section 1.3).

1.2. Simulated Annealing without Noise

Let E be some finite search space and $J : E \to \mathbb{R}_+$ a function that we want to minimize, called cost thereafter.

Simulated Annealing is a classical global optimization method. It aims at building a sequence of elements from E whose last element is drawn from a uniform probability law on the subset of *global* minima of J. In other words it aims at sampling from the following distribution

$$\mu^\star = \frac{\mathbb{1}_{S_{opt}}}{|S_{opt}|} ,$$

where $S_{opt} = \{x, J(x) = \min_{y\in E} J(y)\}$ and $|.|$ denotes the cardinality of a set. Such a sampling is of course not straightforward but one can notice that this distribution can be rewritten in the following form:

$$\forall x \in E,\ \mu^\star(x) = \lim_{T\to 0} \frac{e^{\frac{-J(x)}{T}}}{\sum_{y\in E} e^{\frac{-J(y)}{T}}} ,$$

and it is well-known that the Gibbs distributions of the form $\mu_T = e^{\frac{-J}{T}} / \sum e^{\frac{-J}{T}}$ are efficiently sampled for reasonably low temperatures $T \in \mathbb{R}_+$ using the Metropolis-Hastings algorithm (Aarts and Korst, 1988). A quite natural attempt is therefore to build a sequence of sequences obtained using Metropolis-Hastings algorithm for a set of decreasing temperatures. In particular, at a very low temperature, the Metropolis-Hastings algorithm generates exploratory moves that are accepted with very low probabilities, which makes it a very bad sampler. Therefore it is necessary to first encourage exploration by using a sampling at higher temperatures. A lower bound on the temperature at each step ensuring a convergence in probability of the algorithm has been provided by Hajek (1988). At the same time another proof of convergence using modern semi-group representation of Markov processes has been obtained by Holley and Stroock (1988). The obtained bounds are less explicit but contain information about the convergence rate and the proof scheme is much more general. We set our work in the continuity of this last work and use similar notations.

1.3. Simulated Annealing with Noisy Evaluations

As mentioned previously, our main interest is to extend such a method of simulated annealing to the stochastic case:

$$\text{Find} \quad x^\star = \arg\min_{x\in E} \mathbb{E}_\omega(U(x,\omega))$$

where ω is a random input of a bounded cost function U whose expectation can only be numerically estimated through Monte Carlo simulations. In other words, we consider $J(.) = \mathbb{E}_\omega(U(.,\omega))$. This question is not novel and several attempts were made to address this problem theoretically in the 90's. Gelfand and Mitter (1989) were probably the first ones to introduce the notion of simulated annealing with noisy measurements. They assumed an additive Gaussian noise independent of the evaluation point and gave a sufficient condition for the decrease of the variance σ_k^2 of this noise, to ensure convergence of the algorithm to the optimal set. Gutjahr and Pflug (1996) extended the results to distributions that are more peaked around zero than the Gaussian distribution. Their convergence result can be stated roughly as follows:

Theorem 1 (Gutjahr and Pflug 1996) *Let $(X_k)_{k\in\mathbb{N}}$ denote the sequence of states in E visited by the simulated annealing algorithm with Monte Carlo sampling of the noisy measurements. If:*

(i) the convergence conditions from (Hajek, 1988) are satisfied

(ii) $\exists\epsilon > 0$ such that the standard error of the noise at step k of the algorithm $\sigma_k^2 = \mathcal{O}(k^{-(2+\epsilon)})$

then $\forall x \in E$, $\lim_{k\to+\infty} \mathbb{P}(X_k = x) = \mu^\star(x)$, where $\mu^\star$ is the uniform distribution on the global minima of the expected cost. ■

This result provided a first answer to our question about the convergence of the algorithm in the stochastic case. However the convergence statement above did not give any information about the convergence rate of the algorithm. Following the noise-free proof of Aarts and Korst (1988), Homem-de Mello (2000) provided an extension of this statement to the noisy case with bounded variance and introducing a state dependent noise. He obtained the same constraint on the decrease of the variance and the same convergence statement. He also highlighted the need for an extended result concerning the rate of convergence and for numerical experiments. Indeed on this second point we can mention the works of Fink (1998) and Branke et al. (2008) that addressed this issue. Fink (1998) made a very interesting proposition in the framework of Gaussian noise. He proposed to use the noise of measurement to drive the simulated annealing, *i.e.*, accept a move if the estimated cost of the proposed solution is lower than the one of the current solution. Using an analogy with the Glauber acceptance mechanism, which is a symmetric alternative to the Metropolis-Hasting mechanism (Aarts and Korst, 1988), he proposed a far more efficient criteria for the variance decrease, *i.e.*, $\sigma_k = \mathcal{O}(log(k)^{-2})$. Unfortunately he only provided a few numerical examples to validate his statement and a theoretical proof is still missing.

1.4. Main Contributions

In this paper we consider a simulated annealing algorithm based on mini-batches of increasing size. More precisely, at each iteration, the expected cost is estimated by Monte Carlo sampling of increasing sizes. The estimated cost at step k of the algorithm is thus $\widehat{\mathbb{E}}_\omega(U(x_k,\omega)) = 1/N_k \sum_{i=1}^{N_k} U(x_k,\omega_i)$, where N_k is an increasing sequence and ω_i are i.i.d. random variables having the same law as ω. The cost can be written also as $\widehat{\mathbb{E}}_\omega(U(x_k,\omega)) = \mathbb{E}_\omega(U(x_k,\omega)) + \zeta_\omega(x_k)$, where $\zeta_\omega(x_k)$ is some bounded random variable. We denote $\sigma_k^2 := \mathrm{Var}(\zeta_\omega(x_k))$, the variance of post-sampling noise. As it is directly linked to the number of measurements made during the mini-batch, it can be tuned by the user.

Rate of convergence for all variances of polynomial decay. In the sequel we first show that theoretical guarantees of Theorem 1 can be extended to sub-Gaussian random variables (e.g., bounded noise distributions) with stronger convergence results for this algorithm. Indeed we show that convergence can be ensured if the number of measurements is chosen such that $\sigma_k = \mathcal{O}(k^{-(\alpha/2)})$ with $\alpha > 0$, which corresponds to N_k of the order k^α. One can observe that, as opposed to Gutjahr and Pflug (1996), the convergence still holds for $\alpha \leq 2$. This is summarized in Theorem 3.

We derive the rate of convergence of the procedure (Theorem 5) and optimize it (Corollary 1) with respect to the noisy simulated annealing algorithm parameters in order to provide a minimal total number of measurements at given accuracy and confidence requirements. This leads to the optimal value $\alpha = 2$ for which the number of cost evaluations increases fast enough to ensure almost the same convergence rate as in the noise-free case. This shows that the convergence rate is limited by the concentration speed of the Gibbs measure around its modes. According to our concentration result, increasing the estimation effort cannot increase the performance of the algorithm above this limit. On the other hand the convergence still holds for a decreased estimation effort ($\alpha < 2$) as soon as the cooling schedule is slowed consequently.

Computational cost in the general case. Finally, we derive an upper bound on the computational time-complexity of our simulated annealing algorithm (with noisy measurements). This quantity is roughly of the order of:

$$e^{\frac{C_1 \log \frac{1}{\delta}}{\epsilon}},$$

where C_1 is some constant depending on the cost function itself as detailed in Corollary 1. The provided bound exhibit an exponential dependency in $1/\epsilon$ and $\log 1/\delta$. This is comprehensive regarding the generality of the considered problem.

Computational cost in the absence of local minimum If the function has no local minimum apart from the global minimum (e.g., a convex function evaluated on a finite set) the temperature schedule can be adapted and the computational cost becomes of the order of:

$$\left(\frac{C_2 \log \frac{1}{\delta}}{\epsilon}\right)^3,$$

where C_2 is a constant detailed in Corollary 2. This second bound increases in a polynomial way with respect to $1/\epsilon$ and $\log 1/\delta$. This is a very positive result as it shows that the noisy simulated annealing algorithm recovers the state-of-the-art convergence guaranties if stronger hypotheses on the cost are considered.

Numerical experiments. We provide numerical evidence indicating that the numerically observed requirements in Fink (1998), *i.e.*, $\sigma_k = \mathcal{O}(\log(k))$, do not hold for a Metropolis-Hastings Acceptance criteria. We apply the noisy simulated annealing on classical non convex optimization test cases with different level of noise, but also perform a test on a real-world example, *i.e.*, an aircraft trajectory optimization problem using a black-box aircraft performance model.

1.5. Aircraft Trajectory Optimization

As a leading example for this setting, we consider the problem of optimizing commercial aircraft trajectories with respect to a combination of fuel consumption and flight duration:.

$$\begin{aligned} \text{Find } u^\star = \arg\min_u\ & g(x(t_f),t_f) + \int_{t_0}^{t_f} -\dot{m}(x(s),u(s))ds \\ s.t.\ \forall t > t_0\ & \dot{x}(t) = f(x(t),u(t)) \\ & x(t_0) = x_0 \\ & d(t_f) = d_f, \end{aligned}$$

where x is the state of the aircraft, m its mass, $\dot{m}$ its instantaneous fuel consumption, d the ground distance it has flown over, u the path control, f the instantaneous dynamic and g the terminal cost function. The path control u is the combination of the thrust rating δ_T and the lift coefficient C_l.

$$u = \begin{pmatrix} C_l \\ \delta_T \end{pmatrix}$$

Estimates of the cost of trajectories are usually obtained through numerical integration of the flight dynamic equations, f:

$$\dot{x} = \begin{pmatrix} \dot{V} \\ \dot{\gamma} \\ \dot{h} \\ \dot{d} \\ \dot{m} \end{pmatrix} = \begin{pmatrix} (T(h,V,\delta_T) - D(h,V,C_L))\frac{1}{m} - g\sin\gamma \\ (L(h,V,C_L) - mg\cos\gamma)\frac{1}{mV} \\ V\sin(\gamma) \\ V\cos(\gamma) \\ \eta T(h,V) \end{pmatrix}$$

where T is the thrust, D the drag, L the lift, η the specific fuel consumption, γ the path angle, V the speed of the aircraft and h its height. These equations involve some terms

like the aerodynamic drag coefficient (C_D) or maximal propulsion effort (T_{max}) who are estimated using interpolation of experimental local measurements.

$$T(h, V, \delta_T) = \delta_T T_{max}(h, V)$$
$$L(h, V, C_L) = \frac{1}{2}\rho(h, V)SV^2 C_L$$
$$D(h, V, C_L) = \frac{1}{2}\rho(h, V)SV^2 C_D(C_L, V)$$

No analytic solution is therefore available nor conceivable. Moreover the relation between cost and trajectory control parameters cannot reasonably be assumed to be convex. At last, the cost estimation relies on some predicted flight conditions including atmospheric ones. Hence, real-flight costs can thus deviate substantially from their predictions and some uncertainty propagation method must be applied to obtain an accurate estimate of the expected flight costs. In other words the function we want to minimize can only be evaluated with a certain random error, which corresponds exactly to the setting of this paper. Finally, the computational efficiency is a key ingredient as it must be performed only a few hours before the planned flight. For more information about aircraft trajectory optimization we refer to Betts (1998).
This example completely fits our requirement as the computation of the cost of one single complete trajectory is quite fast, *i.e.*, less than a second. Therefore, the EGO algorithm (Jones et al., 1998) would not be suited for this application. On the other hand formulations based on the DIRECT algorithm (Jones et al., 1993) would suffer strongly from the dimension of the problem. An additional element that motivates the use of simulated annealing is the fact that in the case of trajectory optimization the set of admissible controls is not known in advance as it is path dependent. We can only ensure that this set is connected. This implies in particular that no projection on the constraints can be performed and excludes the projected stochastic gradient descent for example. In the case of simulated annealing, a very simple step can bypass this issue. By setting the value of the cost to infinity when the trajectory evaluator returns an error we ensure staying in the admissible domain. Consequently, a feasible solution and a conservative approximation of the admissible domain are the only requirements to initiate the algorithm in this setting.

1.6. Outline of the Paper

Our paper is organized as follows. In Section 2 we present the noisy simulated algorithm and our main theoretical result. In Sections 3, 4 and 5 we provide the proof of this statement. More precisely, in Section 3 we compute the infinitesimal generator of the noisy simulated annealing algorithm. In Section 4 we compare it to the one of the noise-free simulated annealing algorithm from Holley and Stroock (1988). This enables us to derive a differential inequality for a L^2 distance between the distributions of the two previously mentioned processes. Integrating by applying Grönwall's Lemma Section 5, we obtain obtain our convergence result. In the same section, we show how to tune the parameters of the algorithm in order to optimize the performance bound and give the corresponding computational cost. In Section 6 we propose some numerical insight on synthetic and real data experiments.

Acknowledgments.

We thank Sébastien Gadat for introducing this topic to us, making this collaboration possible and for fruitful discussions and helpful insights, and Sébastien Gerchinovitz for all his constructive advice and ideas.

2. Noisy Simulated Annealing Algorithm: Statement and Convergence Result

We first present our extended version of the simulated annealing to the stochastic case, whose pseudo-code can be found in Algorithm 1.

2.1. Noisy Simulated Annealing algorithm (NSA)

Algorithm 1 Noisy Simulated Annealing

procedure NSA(**Inputs:** Neighbourhoods structure $(S_x)_{x\in S}$, Initial guess x_0, increasing function $\beta : \mathbb{R}_+ \to \mathbb{R}_+$, Function $t \mapsto n_t$)

Initialize time $t_0 = 0$
$\beta_0 = \beta(t_0)$

for k from 0 to Maximal number of iterations **do**
Draw one solution candidate: $\widetilde{x}_{t_k} \in S_{x_{t_k}}$ according to $q_0(x_k, \cdot)$
Draw $N_{t_k} \sim \mathcal{P}oisson(n_{t_k}) + 1$
Draw $2N_{t_k}$ simulation conditions independently:
$(\omega_1^k, ..., \omega_{N_{t_k}}^k) \sim (P_\Omega(x_{t_k}))^{\otimes N_{t_k}}$ and $(\widetilde{\omega}_1^k, ..., \widetilde{\omega}_{N_{t_k}}^k) \sim (P_\Omega(\widetilde{x}_{t_k}))^{\otimes N_{t_k}}$
Compute estimates $\widehat{J}(x_t)$ and $\widehat{J}(\widetilde{x}_t)$ using the N_{t_k} conditions:
$\widehat{J}(x_{t_k}) = \frac{1}{N_{t_k}} \sum_{i=1}^{N_{t_k}} U(x_{t_k}, \omega_i^k)$,
$\widehat{J}(\widetilde{x}_{t_k}) = \frac{1}{N_{t_k}} \sum_{i=1}^{N_{t_k}} U(\widetilde{x}_{t_k}, \widetilde{\omega}_i^k)$

Draw an exponential random variable ξ_{k+1} of parameter 1
Update time $t_{k+1} := t_k + \xi_{k+1}$

With probability $e^{-\beta_k \lfloor \widehat{J}(\widetilde{x}_{t_k}) - \widehat{J}(x_{t_k}) \rfloor_+}$: [1]
set $x_{t_{k+1}} := \widetilde{x}_{t_k}$
Otherwise set $x_{t_{k+1}} := x_{t_k}$

Increase the inverse of the temperature $\beta_{k+1} := \beta(t_{k+1})$
end for
return $x_{t_{k+1}}$
end procedure

[1] $\lfloor x \rfloor_+ = 0$ if $x \leq 0$ and x if not.

As in the deterministic setting, the algorithm requires an initial feasible solution x_{t_0}, a temperature schedule T_t (we will mostly use its inverse $\beta_t = 1/T_t$), and a good neighbourhood structure. What we mean by good will be specified in the definition of q_0. The algorithm explores the state space in the following manner. After k iterations, at time t_k, it selects a random neighbouring solution $\widetilde{x}_{t_k} \in S_{x_{t_k}}$ (S_{x_t} being the set of neighbours of x_{t_k}) according to a proposition law. Then it compares the estimate $\widehat{J}(\widetilde{x}_{t_k})$ of the cost of this new solution to the estimated cost $\widehat{J}(x_{t_k})$ of the current solution and then it decides to substitute (or not) the new to the current:

- if the estimated cost of the new state is lower than the current one, *i.e.*, $\widehat{J}(\widetilde{x}_{t_k}) \leq \widehat{J}(x_{t_k})$, the move is accepted, *i.e.*, $x_{t_{k+1}} \leftarrow \widetilde{x}_{t_k}$
- if not, it is only accepted with a probability $\exp(-\beta_{t_k}(\widehat{J}(\widetilde{x}_{t_k}) - \widehat{J}(x_{t_k}))$.

The time t is then updated using independent exponential random variables, enabling us to consider the NSA as a continuous time Markov process.

Remark 2 *As one can see in line 8 of Algorithm 1, the sample size at each step is random. This is a technical choice that will ease the convergence study of the underlying process. However in practice we did not observe any significant difference between using, at step k, a random sample size N_{t_k} and its expected value, namely $n_{t_k} + 1$.*

2.2. General Setting and Notations

To state the convergence of Algorithm 1, we first need to describe formally the framework we are working in. Notations introduced in this section are valid for the whole paper unless mentioned explicitly.

- Regarding the noise structure and the estimation procedure, we denote:

  $(\widehat{J})$ the estimated cost: $\widehat{J} : E \to \mathbb{R}_+$, such that $\forall\ x \in E, \widehat{J}(x) = \frac{1}{N}\sum_{i=1}^{N} U(x, \omega_i)$, where: $(\omega_1, ..., \omega_N)$ is a N i.i.d. vectors sequence drawn from distribution $P_\Omega(x)$

  (ξ_k) the time increments: $(\xi_k)_{k\in\mathbb{N}}$ is a sequence of i.i.d. exponential random variables of parameter 1

  (t_k) the jumping times: $\forall k \in\ \mathbb{N}, \quad t_k = \sum_{i=1}^{k} \xi_i$.

  (n_t) the samplig intensity: n_t a continuous increasing function.

  (N_{t_k}) the sample sizes: $N_{t_1}, N_{t_2}, \ldots N_{t_n}$ are independent for all $n \in \mathbb{N}$ and all $0 < t_1 < t_2 \ldots < t_n$ and

  $$N_{t_k} \sim \mathcal{P}oisson(n_{t_k}) + 1,$$

  We can make a few remarks about the different notations. The construction of N_t ensures that its value is a strictly positive integer at all times. The reason why we choose to have a randomly sized sample for the Monte Carlo estimation procedure is rather technical. It enables generating a continuous transition probability as it can be noticed in Equation (3) and ease the formulation of the infinitesimal generator (Equation (7)).

- About the state space, we denote:

  (E) a finite state space.

  (S) a neighbourhood structure such that E is connected with respect to it, *i.e.*, S is a connected graph containing all the points in E. For any x in E, we denote S_x the set of its direct neighbours.

  (μ_0) the initial distribution, a probability measure that charges every point of a subset of interest $E' \subset E$ defined more precisely in (U) ,

  (q_0) the proposition law, an irreducible and $\mu_0 - reversible$ transition probability, *i.e.*, $\forall x, y \in E$, $\sum_{n=0}^{\infty} q_0^{(n)}(x, y) = \infty$ and $\mu_0(x)q_0(x, y) = \mu_0(y)q_0(y, x)$. In addition we assume that for any x in E, we have $q_0(x, S_x) = 1$

  Considering a finite search space E enables us to easily derive the spectral gap inequality in Theorem 4 and overcome differentiation-under-the-integral-sign issues in Equation (15). Nevertheless, it could be replaced by coercivity assumptions on the function J, which could be more general but not really well suited for the application we are looking for. It is our most restrictive assumption. Nevertheless it is in line with previous works on noisy global optimization for example: Gutjahr and Pflug (1996), Homem-de Mello (2000) or Fink (1998). It corresponds to a historical use of simulated annealing for problems with huge finite search space like for the traveling salesman problem (Aarts and Korst, 1989). Mimicking Holley and Stroock (1988), we might however relax this assumption of finiteness. Nevertheless it requires more technicalities as in Aarts and Korst (1988) and this is left for future work.

  We assume that the algorithm can visit and start from every point in the solution space through the connection assumption S and the definition of μ_0. The proposition law q_0 defines the way a new solution $\widetilde{x}$ is proposed to the NSA at each iteration. The irreducibility of q_0 implies the fact that one can go from any state x to any other state y using the neighbourhood structure S, in a finite number of steps. The $\mu_0-reversibility$ is used to simplify the notations. A classical choice (Aarts and Korst, 1988) for q_0 and μ_0 is: $\forall\ x, y\ \in\ E,\ \mu_0(x) = \frac{1}{|E|}$ and $q_0(x, y) = \frac{1}{|S_x|}$, assuming every point in E to have the same number of neighbors. However there are other possible choices for μ_0 and q_0. This last two assumptions are inherited from the classical Metropolis-Hasting sampling algorithm which corresponds to the NSA algorithm with no cooling mechanism and no noise. They ensure that a run in this setting, starting from any point of the search space, converges to a stationary distribution which is the Gibbs measure associated to J.

- About the cost function, we consider:

  (U) the underlying cost: $\exists M > 0$ and $\exists E' \subset E$, such that U is bounded and non-negative on E', i.e. $\forall x\ \in\ E', \forall\omega$, $0 \leq U(x, \omega) \leq M$ and U is infinite on $E\backslash E'$, *i.e.*,$\forall x \in E\backslash E'$, $\forall\omega$, $U(x, \omega) = +\infty$.

The assumption about U being bounded is not restrictive. It reflects the practical setting where a simulation code crashes out of the definition domains. We associate infinite costs to crashes and thus (U) is rather a consequence of (E).

- About the algorithm parametrization, we denote

  (β_t) the inverse of the temperature: a positive increasing real function of t,

  (α) the sampling size (expected number of simulations): $\exists \alpha \in \mathbb{R}_+$ such that $n_t = (t+1)^\alpha$,

  β_t is usually chosen such that $\forall t \in \mathbb{R}_+$, $\frac{d\beta_t}{dt} = \frac{bd}{1+td}$ for some $b, d \in \mathbb{R}_+$, as it was shown by (Hajek, 1988) and (Holley and Stroock, 1988) to be a necessary condition to ensure the convergence of the simulated annealing algorithm for any cost function. There is no reason to expect that the noisy context would be more favorable than the deterministic one. As suggested by the definition of α, we choose a polynomial growth of the number of simulations for the cost estimation. We show later on in this work that this ensures the convergence of the noisy simulated annealing for a good choice of α and b.

2.3. Tool for the Analysis: the NSA Process

We now present the mathematical formalization of the NSA algorithm's underlying stochastic process. First, for pedagogical purposes, we omit the temperature evolution and noisy measurements. The NSA algorithm then becomes a simpler Markov chain exploring the state space E according to the Markovian transition matrix whose elements are of the form:

$$\mathbb{P}(x \to y) = q_\beta(x,y) = \begin{cases} q_0(x,y) e^{-\beta \lfloor J(y) - J(x) \rfloor_+} & \text{if } y \neq x \\ 1 - \sum\limits_{z \in E \setminus x} q_\beta(x,z) & \text{if } y = x, \end{cases} \tag{1}$$

This reflects the transition mechanism introduced at the beginning of this section. As the process is in fact a continuous one, we must also consider the time component. NSA jumps happen at stochastic times and the probability of acceptance depends on these times. Combining the law of the jumping times and the previous mechanism, we can make their joint transition probability explicit:

- Let $(\widetilde{\chi}_k, T_k)_{k \in \mathbb{N}}$ a $E \times \mathbb{R}_+$-valued Markov chain such that $\forall k \in \mathbb{N}$, $\forall y \in E$, $\forall u \in \mathbb{R}_+$:

$$\mathbb{P}(\widetilde{\chi}_{k+1} = y, T_{k+1} \geq u | \widetilde{\chi}_k, T_k) = \int_u^{+\infty} \widetilde{q}_{\beta_\tau}(\widetilde{\chi}_k, y) \mathbb{1}_{[T_k, +\infty[}(\tau) e^{-(\tau - T_k)} d\tau, \tag{2}$$

  where

$$\widetilde{q}_{\beta_t}(x,y) = \begin{cases} q_0(x,y) \mathbb{E}_{N_t} \mathbb{E}_{\omega_1, \ldots, \omega_{N_t}} \left(e^{-\frac{\beta_t}{N_t} \lfloor \sum_{i=1}^{N_t} U(y,\omega_i) - U(x,\omega_i) \rfloor_+} \right) & \text{if } y \neq x \\ 1 - \sum\limits_{x \neq z} \widetilde{q}_{\beta_t}(x,z) & \text{if } y = x \end{cases} \tag{3}$$

This is a similar construction to the one of the classical simulated annealing process (Hajek, 1988). The state transition mechanism must also reflect the estimation procedure, therefore the form of Equation (3) differs from Equation (1). As mentioned before the function $t \mapsto \beta_t$ represents the inverse of the temperature schedule and N_t is the random process described by N_{t_k}. The jumping times, or evaluation times of the process happen at times defined by the sequence T_k (*cf.* the definition of t_k).

The chain $(\widetilde{\chi}_k)_{k \geq 0}$ explores the state space E using a transition probability $\widetilde{q}_{\beta_t}$ constructed in a same way as the classical one, replacing the exact value of $-\beta_{T_k}(J(y) - J(x))_+$ by its Monte Carlo estimation. The expected value from the formula comes from the fact that, as mentioned in the definition of $\widehat{J}$ and in Algorithm 1, we use a random number of Monte Carlo shootings for the estimations.

Finally, we obtain the NSA process by associating the two sub-processes as follows:

- Let $\left(\widetilde{X}_t\right)_{t \geq 0}$ be the inhomogeneous Markov Process such that $\widetilde{X}_t = \widetilde{\chi}_k$ if $T_k \leq t < T_{k+1}$. One can see that this process is piecewise constant and jumps at exponential times from one candidate solution to another, in other words $(\widetilde{X}_t)_{t \geq 0}$ is just the continuous-time version of the noisy simulated annealing discrete time process,$(\widetilde{\chi}_k)_{k \geq 0}$.

  Note that, if $y \in E \setminus E'$ then $\forall x \in E'$, $\widetilde{q}_{\beta_t}(x,y) = 0$. Hence, if the initial solution $\widetilde{X}_0$ is chosen in E', then $\forall t \geq 0$, $\widetilde{X}_t \in E'$.

2.4. Convergence Result

We denote:

- $m^\star$ the maximum depth of a well not containing a fixed global minimum of the function J. To be more precise, we call a path from x to y any finite sequence $x_0 = x, x_1, \ldots, x_n = y$ such that for all i, $x_{i+1} \in S_{x_i}$. Let P_{xy} be the set of paths from x to y.
  For a given path $p \in P_{x,y}$, the elevation of the function J on p is $\max\limits_{z \in p} J(z)$. Minimizing this quantity over the set of possible paths $P_{x,y}$, gives us the elevation of the cheapest path going from x to y. Denote this elevation by:

$$H_{x,y} = \min_{p \in P_{xy}} \left\{ \max_{z \in p} J(z) \right\}$$

  Then

$$m^\star := \max_{x,y \in E} \{ H_{x,y} - \max(J(y), J(x)) \} \tag{4}$$

  As represented on Figure 1, $m^\star$ can also be understood as the highest energy barrier to climb to go from one point to another in the search space in the easiest direction. As mentioned before, it also represents the maximal depth of a well not containing a fixed global minimum. If $x^\star$ is a global minimum then:

$$m^\star = \max_{y \in E} \{ H_{x^\star, y} - J(y) \}.$$

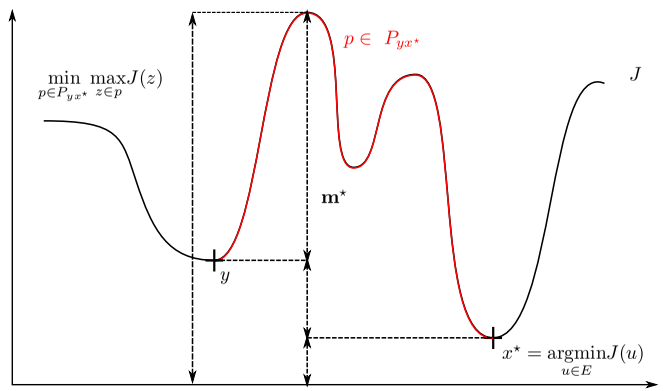


Figure 1: $m^\star$, maximal depth of local minima

The definition provided here is equivalent to the classical one, *i.e.*, the one provided in Hajek (1988) and Holley and Stroock (1988). A proof of this statement can be found in Appendix B.

- $\gamma(\beta)$ the spectral gap between 0 and the rest of the $L^2(\mu_\beta)$ spectrum of $-L_\beta$, where L_β is the generator of the classical simulated annealing (for more details about L_β see Section 3):

$$\gamma(\beta) := \inf\left\{-\int \phi L_\beta \phi d\mu_\beta \text{ s.t. } \int |\phi|^2 d\mu_\beta = 1 \text{ and } \int \phi d\mu_\beta = 0\right\} \tag{5}$$

Following Holley and Stroock (1988), we know that given E, μ_0 and U, there exists a constant c such that:

$$\forall \beta \geq 0, \quad \gamma(\beta) \geq c e^{-m^\star \beta}$$

Remark that this lower bound is mainly informative for small values of β. In addition set:

$$\forall x \in E,\ J(x) = \mathbb{E}_\omega(U(x,\omega)) \text{ and } J^\star = \min_{x\in E} \mathbb{E}_\omega(U(x,\omega)).$$

We define χ_ϵ the set of ϵ-optimal points in E, *i.e.*,

$$\chi_\epsilon = \{x : J(x) \leq J^\star + \epsilon\}, \tag{6}$$

and denote ${}^c\chi_\epsilon = E\backslash\chi_\epsilon$, its complementary in E. We also write $a \wedge b = \min a, b$.

Theorem 3 *Consider the settings of Section 2.2, if* $\beta_t = b\log(td+1)$ *and* $n_t = (t+1)^\alpha$, *with:*

$$\{m^\star b < 1 \wedge \alpha/2\} \text{ or } \{m^\star b = 1,\ \alpha > 2 \text{ and } d < 2cm^\star/M\},$$

then there exits $C > 0$ *such that,* $\forall t \in \mathbb{R}_+,\ \forall \epsilon > 0, \quad \mathbb{P}(\widetilde{X}_t \in {}^c\chi_\epsilon) \leq C(\mu_{\beta_t}({}^c\chi_\epsilon))^{1/2}.$ ■

This theorem is a natural extension of the result provided by Holley and Stroock (1988). There are two main interesting facts to point out. First, we obtain a balance between the expected number of Monte Carlo simulations at each step of the algorithm and the inverse of the temperature, *i.e.*, $\{m^\star b < 1 \wedge \alpha/2\}$ or $\{m^\star b = 1,\ \alpha > 2$ and $d < 2cm^\star/M\}$. Reducing the growth rate α of the number of simulations below the quadratic rate should be compensated by decreasing accordingly the temperature factor b. Second, the convergence is stated in terms of a bound on the probability of not returning an optimal solution. Using the concentration speed of the Gibbs measure one can deduce a rate of convergence of the algorithm. Also the theorem provides an insight on how the algorithm could be used in practice. A run of parallel noisy simulated annealing would have a probability of returning a bad solution that would decrease in the power of the number of runs. Nevertheless this benefit should be traded with an additional selection cost. Indeed, if we obtain K solutions retrieved by K parallel NSA realization, we still face the problem of selecting the best one. We only access estimates of the costs associated to each solution.

Remark 1 *Taking* $n_t = (t+1)^\alpha$, *implies that the sample size at step* k, *is of the order* t_k^α. *The value* $\alpha = 2$ *minimizes the number of measurements used by the algorithm in order to return an estimation of a given accuracy and confidence level (see Corollary 1). In practice generating samples of sizes growing at this rate might be overwhelming. However, for some well-behaved cost functions and low noise levels one can afford a sub-optimal evaluation number as shown in Figure 3. On the other hand, when this is not the case, using fewer measurements at each step can decrease the performance of the algorithm as suggested in Figure 5.*

Sketch of the proof The proof of this theorem is divided into three parts. First, in Section 3, we compute the infinitesimal generator of the classical (Equation (11)) and noisy simulated annealing (Equation (7)). Second, in Section 4, we compare them (Lemma 1) and third, in Section 5, we conclude about the convergence using the Grönwall lemma (Equation (18)) and the convergence of the classical simulated annealing (Equation (24)).

Convergence rate In the case $m^\star b < 1$ a finer bound can be deduced from Grönwall's lemma and one can obtain a more precise convergence rate for the algorithm (Theorem 5), which is roughly of the order of:

$$\mathbb{P}(\widetilde{X}_t \in {}^c\chi_\epsilon) \leq \Gamma\ t^{((m^\star-\epsilon)b - \min(1,\alpha/2))/2}$$

where Γ is some constant detailed in Theorem 5. In particular this implies that for fixed $\epsilon, \delta > 0$ we can find $T^\star$ such that $\mathbb{P}(\widetilde{X}_T^\star \in {}^c\chi_\epsilon) \leq \delta$. This leads to a bound (Lemma 6) on the computational complexity, $\mathbb{E}\left(N_{call}^{T^*}\right)$, of the order of:

$$\mathbb{E}\left(N_{call}^{T^*}\right) \leq \left(\frac{\Gamma}{\delta}\right)^{2(\alpha+1)/(\min(1,\alpha/2)-(m^\star-\epsilon)b)}.$$

3. Proof, Part 1: Infinitesimal Generator

In this section we use the semi-group characterization of the generator in order to prove that as soon as $\widetilde{q}_{\beta_t}$ defined in Equation (3) is continuous with respect to t then the infinitesimal generator $\widetilde{L}_{\beta_t}$ of the Markov process $\widetilde{X}_t$ can be written as:

$$\widetilde{L}_{\beta_t} f(x) = \sum_{y \in E} \Big(f(y) - f(x) \Big) \widetilde{q}_{\beta_t}(x, y). \tag{7}$$

We briefly recall the definition of the semi-group associated to a Markov process.

Definition 1 *The semi-group $(P_{t,t+s})_{t\geq 0, s\geq 0}$ associated to the Markov process $(X_t)_{t\geq 0}$ is a family of probability kernels such that for all non-negative borelian functions:*

$$\forall t, s \in \mathbb{R}_+ \quad P_{t,t+s} f(x) = \mathbb{E}(f(X_{t+s}) | X_t = x)$$

Let $(P_{t,t+s})_{t\geq 0, s\geq 0}$ be the semi-group associated to the Markov process $(X_t)_{t\geq 0}$. The semi-group characterization of its generator is given in the following definition:

Definition 2 *The infinitesimal generator L_t of the Markov process $(X_t)_{t\geq 0}$ is defined as the operator such that for any bounded function f:*

$$L_t f(x) = \lim_{s \to 0} \frac{P_{t,t+s} f(x) - P_{t,t} f(x)}{s}$$

We start by computing the infinitesimal generator L_{β_t} of the process associated to the SA algorithm, *i.e.*, with no measurement noise, and then deduce the infinitesimal generator of the NSA algorithm. Using similar notations to the ones of Section 2.3, we consider the noise free inhomogeneous Markov process, $(X_t)_{t\geq 0}$ constructed from the inhomogeneous Markov chain $(\chi_k)_{k\in\mathbb{N}}$ whose one step transition probability is:

$$\forall x, y \in E, \quad q_{\beta_{T_k}}(x, y) = \begin{cases} q_0(x, y) e^{-\beta_{T_k}(J(y) - J(x))_+} & \text{if } y \neq x \\ 1 - \sum\limits_{z \in E \setminus \{x\}} q_{\beta_{T_k}}(x, z) & \text{if y=x} \end{cases}$$

This is the natural extension of the simulated annealing process with discrete jumping times (Hajek, 1988) to the continuous time process. In this configuration, the jumping times are drawn from an i.i.d. sequence of exponential random variables of parameter 1. In the homogeneous configuration, *i.e.*, $\beta_t = \beta$, the infinitesimal generator has a classical form: $L_\beta = Q_\beta - Id$ where Q_β is the transition matrix associated to q_β and Id denotes the identity. The extension to the generator of the non-homogeneous process is not straightforward. Therefore we propose to detail the computations.

By definition, for any bounded function f:

$$\begin{aligned} L_{\beta_t} f(x) &= \lim_{s \to 0} \frac{P_{t,t+s} f(x) - P_{t,t} f(x)}{s} \\ &= \lim_{s \to 0} \frac{\sum\limits_{y \in E} f(y) \mathbb{P}(X_{t+s} = y | X_t = x) - f(x)}{s} \\ &= \lim_{s \to 0} \frac{\sum\limits_{y \in E} f(y) \mathbb{P}(X_{t+s} = y, H_{t+s} - H_t \geq 0 | X_t = x) - f(x)}{s}, \end{aligned} \tag{8}$$

where $H_t = \max\{k \in \mathbb{N} : T_k < t\}$ denotes the number of jumps before time t. Since T_k is a sum of independent exponential variables of parameter 1, one can remark that H_t is in fact a Poisson process of parameter 1.

In order to compute the above limit, we begin by calculating a more explicit form of the probabilities above. We can divide these computations into three parts according to the number of jumps between t and $t + s$:

$$\begin{aligned} \mathbb{P}(X_{t+s} = y, H_{t+s} - H_t \geq 0 | X_t = x) = \mathbb{P}(X_{t+s} = y, H_{t+s} - H_t = 0 | X_t = x) \\ + \mathbb{P}(X_{t+s} = y, H_{t+s} - H_t = 1 | X_t = x) \\ + \mathbb{P}(X_{t+s} = y, H_{t+s} - H_t \geq 2 | X_t = x). \end{aligned}$$

The first case is straightforward, if there is no jump between t and $t + s$, the process will not change its position and we thus have:

$$\mathbb{P}(X_{t+s} = y,\ H_{t+s} - H_t = 0 | X_t = x) = \delta_x(y) e^{-s}.$$

The second case is slightly more involved. Using the stationarity and the definition of Poisson processes, the event that the algorithm goes from x to y, having only one jump between t and $t + s$, can be written as:

$$\mathbb{P}(X_{t+s} = y, H_{t+s} - H_t = 1 | X_t = x) = \mathbb{P}(X_{t+s} = y, \xi_1' < s, s - \xi_1' < \xi_2' | X_t = x)$$

where ξ_1' and ξ_2' are two independent exponential random variables of parameter one.
Let $\xi = (\xi_1', \xi_2')$ and $D_s = \{(h_1, h_2) \in \mathbb{R}^2 | h_1 < s \text{ and } h_2 > s - h_1\}$. Also in what follows, for a random variable Y we denote f_Y its probability distribution. Using these notations and the fact that ξ is independent of X_t, we can write:

$$\begin{aligned} \mathbb{P}(X_{t+s} = y, \xi \in D_s | X_t = x) &= \int_{D_s} f_{(X_{t+s}, \xi) | X_t = x}(y, h) \mathrm{d}h \\ &= \int_{D_s} f_{X_{t+s} | \xi = h, X_t = x}(y) f_\xi(h) \mathrm{d}h \\ &= \int_0^s \int_{s - h_1}^{+\infty} q_{\beta_{t+h_1}}(x, y) e^{-h_1} e^{-h_2} \mathrm{d}h_1 \mathrm{d}h_2 \end{aligned}$$

The previous equality yields:

$$\mathbb{P}(X_{t+s} = y,\ H_{t+s} - H_t = 1 | X_t = x) = e^{-s} \int_0^s q_{\beta_{t+h_1}}(x, y) \mathrm{d}h_1. \tag{9}$$

In the following we use the classical $\mathcal{O}(.)$ and $o(.)$ notations: for all functions f and g defined on some subset of $\mathbb{R}$,

- $f(x) = \mathcal{O}(g(x))$ as $x \to 0^+ \iff \exists \sigma, x_0 > 0, |f(x)| \leq \ \sigma |g(x)|$ for all $0 < x \leq x_0$
- $f(x) = o(g(x))$ as $x \to 0^+ \iff \lim\limits_{x \to 0^+} \frac{f(x)}{g(x)} = 0.$

For the third term we can see that:

$$\mathbb{P}(X_{t+s} = y,\ H_{t+s} - H_t \geq 2 | X_t = x) \leq \mathbb{P}(H_{t+s} - H_t \geq 2) \leq \mathbb{P}(H_s \geq 2)$$

Since H_s is a Poisson Process of parameter 1, one can check that for all s close to zero we have that $\mathbb{P}(H_s \geq 2) = 1 - \mathbb{P}(H_s = 0) - \mathbb{P}(H_s = 1) = 1 - e^{-s} - se^{-s} = \mathcal{O}(s^2)$.

This implies that when s is close to zero, the probability that the process goes from x to y between t and $t+s$, with more than one jump is small in comparison to s:

$$\mathbb{P}(X_{t+s} = y,\ H_{t+s} - H_t \geq 2 | X_t = x) = \mathcal{O}(s^2) \tag{10}$$

Putting all the terms together and replacing them in Equation (2), we can rewrite the infinitesimal generator as follows:

$$\begin{aligned} L_{\beta_t} f(x) = \lim_{s\to 0} \frac{1}{s} \quad & \left[\sum_{y\in E} f(y) \left[\delta_x(y) e^{-s} + e^{-s} \int_0^s q_{\beta(t+\tau)}(x,y) d\tau \right] - f(x) \right] \\ + \lim_{s\to 0} \frac{1}{s} & \sum_{y\in E} f(y) \mathbb{P}(X_{t+s} = y, H_{t+s} - H_t \geq 2 | X_t = x) \end{aligned}$$

Using the fact that f is bounded, E finite and the upper bound given by Equation (10), one can easily check that the second term is zero. Hence we obtain:

$$\begin{aligned} L_{\beta_t} f(x) &= \lim_{s\to 0} \frac{1}{s} \quad \left[e^{-s} \left[f(x) + \sum_{y\in E} f(y) \int_0^s q_{\beta(t+\tau)}(x,y) d\tau \right] - f(x) \right] \\ &= \lim_{s\to 0} \frac{f(x)(e^{-s} - 1)}{s} + \lim_{s\to 0} \frac{e^{-s}}{s} \left(\sum_{y\in E} f(y) \int_0^s q_{\beta(t+\tau)}(x,y) d\tau \right) \end{aligned}$$

Noting the fact that q_{β_t} is continuous with respect to t and the following identity

$$e^{-s} = 1 - s + \mathcal{O}(s^2),$$

we easily obtain the simplest form for the infinitesimal generator of the inhomogeneous Markov chain:

$$L_{\beta_t} f(x) = \sum_{y\in E} \Big(f(y) - f(x) \Big) q_{\beta_t}(x,y). \tag{11}$$

We can remark that the explicit form of the transition probability q_{β_t} does not appear in the proof, hence the result is completely general. The only necessary property of this transition probability is its continuity with respect to t.

The fact that n_t and β_t are continuous functions ensures the continuity of transition probability $\widetilde{q}_{\beta_t}$, defined in Equation (3) . Therefore, following the same argument,one can deduce (7). Here we can see the relevance of the randomness of N_t. An increasing deterministic sequence would generate a discontinuous $\widetilde{q}_{\beta_t}$ and would make difficult the use of derivations above.

4. Proof, Part 2: Generators Comparison

The fact that for a temperature schedule that decreases slowly enough, the process generated by the classical Simulated Annealing converges to the set of global minima of J is well known. The Noisy Simulated Annealing is a similar algorithm, built on the same principles except that the values of the function J are replaced by an estimation each time its computation is needed. Therefore a tight relation exists between both approaches. Furthermore, as we will show in this section, for a well chosen couple (β_t, n_t) the generators of the two algorithms will be 'close' at large times. This is a key element of the proof as it will imply a first condition for the ratio β_t/n_t.

Using the relations given by Equation (11) and Equation (7), the quantity of interest is:

$$\widetilde{L}_{\beta_t} f(x) = L_{\beta_t} f(x) + \sum_{y\in E} (f(y) - f(x))(\widetilde{q_{\beta_t}} - q_{\beta_t})(x,y).$$

Hence quantifying the difference between the two generators can be reduced to bounding the difference between the two probability transitions q_{β_t} and $\widetilde{q}_{\beta_t}$. Thus the main result of this section is the following lemma.

Lemma 1 *Let $\beta_t/\sqrt{n_t} \underset{\infty}{\to} 0$. There exist two functions ϵ_t^- and ϵ_t^+ such that*

$$\forall\ t\ \in\ \mathbb{R}_+,\ \forall\ x\ \in\ E',\ \forall\ y\ \in\ E, \quad \epsilon_t^- q_{\beta_t}(x,y) \leq (\widetilde{q}_{\beta_t} - q_{\beta_t})(x,y) \leq \epsilon_t^+ q_{\beta_t}(x,y)$$

and

$$\lim_{t\to+\infty} \epsilon_t^- = \lim_{t\to+\infty} \epsilon_t^+ = 0.$$

Before going into the proof of this lemma, we present some preliminaries. First it can be noticed that for all $x, y \in E$, $x \neq y$ we have:

$$\begin{aligned} &(\widetilde{q_{\beta_t}} - q_{\beta_t})(x,y) \\ &= q_{\beta_t}(x,y) \left(\frac{\widetilde{q_{\beta_t}}}{q_{\beta_t}} - 1 \right)(x,y) \\ &= q_{\beta_t}(x,y) \left(\mathbb{E}_{N_t} \mathbb{E}_{\omega_1,\dots,\omega_{N_t}} \left(\frac{e^{-\frac{\beta}{N_t} \lfloor \sum_{i=1}^{N_t} U(y,\omega_i) - U(x,\omega_i) \rfloor_+}}{e^{-\beta_t \lfloor \mathbb{E}(U(y,\Omega)) - \mathbb{E}(U(x,\Omega)) \rfloor_+}} - 1 \right) \right). \end{aligned}$$

Unless specified otherwise, in this section we always consider $x \neq y$. The case $x = y$ is handled at the end of the section. To simplify the notations we denote $X_i^{x,y} := U(y,\omega_i) - U(x,\omega_i) - \mathbb{E}(U(y,\omega_i) - U(x,\omega_i))$ and $K^{x,y} = \mathbb{E}(U(y,\omega_i) - U(x,\omega_i))$. Hence,

$$\begin{aligned} &(\widetilde{q_{\beta_t}} - q_{\beta_t})(x,y) \\ &= q_{\beta_t}(x,y) \left(\mathbb{E}_{N_t} \mathbb{E}_{\omega_1,\dots,\omega_{N_t}} \left(e^{-\beta_t \left(\lfloor \frac{1}{N_t} \sum_{i=1}^{N_t} X_i^{x,y} + K^{x,y} \rfloor_+ + \lfloor K^{x,y} \rfloor_+ \right)} - 1 \right) \right). \end{aligned}$$

Noticing that,

$$\forall\ a, b \in \mathbb{R}, \quad -|a| \leq -\lfloor a + b \rfloor_+ + \lfloor b \rfloor_+ \leq |a|,$$

we obtain the following bounds for $(\widetilde{q_{\beta_t}} - q_{\beta_t})(x,y)$:

$$\mathbb{E}_{N_t}\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left(e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}-1\right) \geq (\frac{\widetilde{q_{\beta_t}} - q_{\beta_t}}{q_{\beta_t}})(x,y) \geq \mathbb{E}_{N_t}\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left(e^{-|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}-1\right). \tag{12}$$

In order to obtain a bound for the expectation of a function of N_t we need an estimation of the probability that N_t takes values 'far' from its expectation.

Lemma 2 *There exist $\delta \in (0,1)$ and $a = |(1-\delta)(1-\log(1-\delta))-1|$ such that for all $t>0$ we have:*

$$\mathbb{P}(N_t \leq (1-\delta)n_t) \leq e^{-an_t}.$$

Proof. We remind the reader that, as mentioned in the definition N_{t_k}, at a fixed time t the process N_t can be written as $1+H$, where H is a Poisson random variable of parameter n_t.

Fix $t \in \mathbb{R}$ and $\delta \in (0,1)$. We provide an upper bound for $\mathbb{P}(N_t \leq (1-\delta)n_t)$ using the Cramer-Chernoff method.

First we see that for any $\lambda > 0$, applying Markov's inequality we have:

$$\mathbb{P}(N_t \leq (1-\delta)n_t) = \mathbb{P}(e^{-\lambda N_t} > e^{\lambda(\delta-1)n_t}) \leq \frac{\mathbb{E}_{N_t}[e^{-\lambda N_t}]}{e^{\lambda(\delta-1)n_t}}..$$

For t fixed $N_t - 1$ has the distribution of a Poisson random variable of parameter n_t. Therefore by direct computations we have:

$$\begin{aligned}\mathbb{E}_{N_t}[e^{-\lambda N_t}] &= \sum_{k>0}(e^{-\lambda(k+1)}e^{-n_t}\frac{n_t^k}{k!}) \\ &= e^{-n_t-\lambda}\sum_{k>0}\frac{(e^{-\lambda}n_t)^k}{k!} \\ &= e^{-n_t-\lambda}e^{e^{-\lambda}n_t}\end{aligned}$$

Putting all these elements together yields:

$$\mathbb{P}(N_t \leq (1-\delta)n_t) \leq \exp([-\lambda(\delta-1)-1+e^{-\lambda}]n_t)$$

The idea is to choose λ and δ in order to obtain the smallest possible value for $-\lambda(\delta-1)-1+e^{-\lambda}$. For $\delta \in (0,1)$, the minimum is reached at $\lambda = -\log(1-\delta)$ which is strictly positive. For such λ and δ, we denote $a = |(1-\delta)(1-\log(1-\delta))-1|$ and conclude the proof. ■

Now we have what we need in order to start the proof of Lemma 1.

4.1. Proof of Lemma 1.

First, recall that if $y \in E\backslash E'$ then $\forall x \in E'$, $\widetilde{q}_{\beta_t}(x,y) = q_{\beta_t}(x,y) = 0$. Hence Lemma Equation (1) is trivially verified for $x \in E$ and $y \in E\backslash E'$.
Considering the inequalities given by Equation (12), the proof can be divided in two parts by studying separately the upper bound $\mathbb{E}_{N_t}\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left(e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}\right)$ and the lower bound $\mathbb{E}_{N_t}\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left(e^{-|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}\right)$.

4.1.1. Upper Bound

We have:

$$(\widetilde{q_{\beta_t}} - q_{\beta_t})(x,y) \leq q_{\beta_t}(x,y)\left(\mathbb{E}_{N_t}\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left(e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}-1\right)\right).$$

First we will provide an estimate of $\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left[e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}|N_t\right]$ for all N_t.

We start by rewriting this expectation as:

$$\begin{aligned}\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left[e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}|N_t\right] &= \int_{\mathbb{R}_+}\mathbb{P}\left(e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|} > u|N_t\right)du \\ &= \int_{\mathbb{R}_+}\mathbb{P}\left(\left|\sum_{i=1}^{N_t}X_i^{x,y}\right| > \frac{\log(u)N_t}{\beta_t}|N_t\right)du.\end{aligned}$$

Since $X_i^{x,y} = U(y,\omega_i) - U(x,\omega_i) - \mathbb{E}_{N_t}\mathbb{E}_{\omega_i}(U(y,\omega_i)-U(x,\omega_i))$ is a centered random variable and U is bounded on E' (see the definition of U) there exists σ such that $|X_i^{x,y}| \leq \sigma$ (for example set $\sigma = 2M$), almost surely for all i. Therefore $(X_i^{x,y})_{1\leq i\leq N_t}$ are sub-gaussian random variables ((Boucheron et al., 2013)) with variance factor σ^2, *i.e.*,

$$\forall u \geq 0,\ \max\left(\mathbb{P}(X_i^{x,y} > u), \mathbb{P}(-X_i^{x,y} > u)\right) \leq e^{-\frac{u^2}{2\sigma^2}}.$$

Considering that $(X_i^{x,y})_{i\leq N_t}$ is a sequence of independent sub-Gaussian variables, their sum is still a sub-Gaussian variable. As $\mathrm{Var}(X_i^{x,y}) \leq \sigma^2$ for all i we have that $\mathrm{Var}\left(\sum_{i=1}^{N_t}X_i^{x,y}\right) \leq N_t\sigma^2$ and therefore:

$$\forall u \geq 0,\ \max\left(\mathbb{P}\left(\sum_{i=1}^{N_t}X_i^{x,y} > u\right), \mathbb{P}\left(-\sum_{i=1}^{N_t}X_i^{x,y} > u\right)\right) \leq e^{-\frac{u^2}{2\sigma^2N_t}}.$$

For more details about sub-gaussian variables we refer to (Boucheron et al., 2013). We use this property of concentration in order to get an estimate of the expectation:

$$\begin{aligned}
&\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left[e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}|N_t\right] \\
&\quad=\int_0^1 \mathbb{P}\left(\left|\sum_{i=1}^{N_t}X_i^{x,y}\right|>\frac{\log(u)N_t}{\beta_t}|N_t\right)du+\int_1^{+\infty}\mathbb{P}\left(\left|\sum_{i=1}^{N_t}X_i^{x,y}\right|>\frac{\log(u)N_t}{\beta_t}|N_t\right)du \\
&\quad\leq 1+2\int_1^{+\infty}e^{-\frac{1}{2\sigma^2}\left(\frac{\log(u)\sqrt{N_t}}{\beta_t}\right)^2}du.
\end{aligned}$$

Using a simple variable substitution $s=\frac{\log(u)}{\lambda_t}$ with $\lambda_t=\frac{\sigma\beta_t}{\sqrt{N_t}}$, we get:

$$\begin{aligned}
\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left[e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}|N_t\right] &\leq 1+2\lambda_t\int_0^{+\infty}e^{\lambda_t s}e^{-\frac{s^2}{2}}du \\
&\leq 1+2\lambda_t e^{\frac{\lambda_t^2}{2}}\int_0^{+\infty}e^{-\frac{(s-\lambda_t)^2}{2}}ds \\
&\leq 1+2\lambda_t e^{\frac{\lambda_t^2}{2}}\int_{-\lambda_t}^{+\infty}e^{-\frac{u^2}{2}}du \\
&\leq 1+2\sqrt{2\pi}\lambda_t e^{\frac{\lambda_t^2}{2}}\mathbb{P}(G>-\lambda_t) \\
&\leq 1+2\sqrt{2\pi}\lambda_t e^{\frac{\lambda_t^2}{2}}(1-\mathbb{P}(G>\lambda_t)).
\end{aligned}$$

where G is a standard Gaussian. Thanks to the Taylor formula, we know that there exists a constant $0<\theta<1$, such that:

$$\begin{aligned}
\mathbb{P}(G>\lambda_t)&=\mathbb{P}(G>0)+\lambda_t\frac{e^{\frac{-(\theta\lambda_t)^2}{2}}}{\sqrt{2\pi}} \\
&=\frac{1}{2}+\lambda_t\frac{e^{\frac{-(\theta\lambda_t)^2}{2}}}{\sqrt{2\pi}} \\
&\geq\frac{1}{2}+\lambda_t\frac{e^{\frac{-(\lambda_t)^2}{2}}}{\sqrt{2\pi}}.
\end{aligned}$$

This leads to:

$$\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left[e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}|N_t\right]\leq 1+(\sqrt{2\pi}e^{\frac{\lambda_t^2}{2}})\lambda_t.$$

Thus, replacing λ_t by its definition we see that we need an estimate of:

$$\mathbb{E}_{N_t}\left[1+(\sqrt{2\pi}e^{\frac{\sigma^2\beta_t^2}{2N_t}})\frac{\sigma\beta_t}{\sqrt{N_t}}\right].$$

In order to simplify the notations we denote: $g_t=\sigma\beta_t$. Using the bound given by Lemma 2 we get:

$$\begin{aligned}
&\mathbb{E}_{N_t}\left[1+\sqrt{2\pi}e^{\frac{g_t^2}{2\sqrt{N_t}}}\frac{g_t}{\sqrt{N_t}}\right] \\
&\quad=1+\mathbb{E}_{N_t}\left[\sqrt{2\pi}e^{\frac{g_t^2}{2\sqrt{N_t}}}\frac{g_t}{\sqrt{N_t}}\right] \\
&\quad\leq 1+\mathbb{E}_{N_t}\left[\sqrt{2\pi}e^{\frac{g_t^2}{2\sqrt{N_t}}}\frac{g_t}{\sqrt{N_t}}(\mathbb{1}_{[1,(1-\delta)n_t]}+\mathbb{1}_{[(1-\delta)n_t,+\infty)})\right] \\
&\quad\leq 1+\sqrt{2\pi}g_t\left[\frac{e^{\frac{g_t^2}{2(1-\delta)n_t}}}{\sqrt{(1-\delta)n_t}}+e^{g_t^2-an_t}\right]
\end{aligned}$$

Finally we have obtained that under assumptions of Section 2.2:

$$\begin{aligned}
(\widetilde{q_{\beta_t}}-q_{\beta_t})(x,y)&\leq q_{\beta_t}(x,y)\left(\mathbb{E}_{N_t}\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left(e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}-1\right)\right) \\
&\leq q_{\beta_t}(x,y)\sqrt{2\pi}g_t\left[\frac{e^{\frac{g_t^2}{2(1-\delta)n_t}}}{\sqrt{(1-\delta)n_t}}+e^{g_t^2-an_t}\right]
\end{aligned}$$

Hence it is natural to define ϵ_t^+ as:

$$\epsilon_t^+=\sqrt{2\pi}\beta_t\sigma\left[\frac{e^{\frac{\beta_t^2\sigma^2}{2(1-\delta)n_t}}}{\sqrt{(1-\delta)n_t}}+e^{\beta_t^2\sigma^2-an_t}\right].$$

Since $\beta_t/\sqrt{n_t}\underset{\infty}{\to}0$ one can check that ϵ_t^+ goes to 0 when t goes to infinity. Here we can see once more the importance of the balance between the two parameters β_t and n_t.

4.1.2. Lower Bound

Considering the left-hand side of Equation (12) we have:

$$(\widetilde{q_{\beta_t}}-q_{\beta_t})(x,y)\geq q_{\beta_t}(x,y)\left(\mathbb{E}_{N_t}\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left(e^{-|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}-1\right)\right).$$

In a similar way we start by obtaining a lower bound for $\mathbb{E}_{\omega_1,\dots,\omega_{N_t}}\left(e^{-|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}|N_t\right)$ and after we improve it using the probabilistic properties of N_t. First observe that:

$$\begin{aligned}
\mathbb{E}_{\omega_1,..,\omega_{N_t}}\left[e^{-|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}|N_t\right] &= \int_{\mathbb{R}_+}\mathbb{P}\left(e^{-|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}>u|N_t\right)du\\
&= \int_{\mathbb{R}_+}\mathbb{P}\left(-\left|\sum_{i=1}^{N_t}X_i^{x,y}\right|>\frac{\log(u)N_t}{\beta_t}|N_t\right)du\\
&= \int_0^1\mathbb{P}\left(\left|\sum_{i=1}^{N_t}X_i^{x,y}\right|<\frac{-\log(u)N_t}{\beta_t}|N_t\right)du\\
&= \int_0^1 1-\mathbb{P}\left(\left|\sum_{i=1}^{N_t}X_i^{x,y}\right|>\frac{-\log(u)N_t}{\beta_t}|N_t\right)du\\
&\geq 1-2\int_0^1 e^{-\frac{N_t}{2\sigma^2}\left(\frac{-\log(u)}{\beta_t}\right)^2}du
\end{aligned}$$

Again, this is due to the fact that the sum of $X_i^{x,y}$ is sub-Gaussian with variance factor $N_t\sigma^2$.

Using the same variable substitution as above: $s=\frac{\log(u)}{\lambda_t}$ with $\lambda_t=\frac{\sigma\beta_t}{\sqrt{N_t}}$, we get:

$$\begin{aligned}
\mathbb{E}_{\omega_1,..,\omega_{N_t}}\left[e^{-|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}|N_t\right] &\geq 1-2\lambda_t\int_{-\infty}^0 e^{\lambda_t s}e^{-\frac{s^2}{2}}du\\
&\geq 1-2\lambda_t e^{\frac{\lambda_t^2}{2}}\int_{-\infty}^0 e^{-\frac{(s-\lambda_t)^2}{2}}ds\\
&\geq 1-2\lambda_t e^{\frac{\lambda_t^2}{2}}\int_{-\infty}^{-\lambda_t} e^{-\frac{u^2}{2}}du\\
&\geq 1-2\sqrt{2\pi}\lambda_t e^{\frac{\lambda_t^2}{2}}\mathbb{P}(G>\lambda_t)
\end{aligned}$$

where G is a standard $N(0,1)$ Gaussian. As seen before there exists some $0<\theta<1$, such that:

$$\mathbb{P}(G>\lambda_t)\geq\frac{1}{2}+\lambda_t\frac{e^{\frac{-(\lambda_t)^2}{2}}}{\sqrt{2\pi}}.$$

This leads to

$$\mathbb{E}_{\omega_1,..,\omega_{N_t}}\left[e^{|\frac{\beta_t}{N_t}\sum_{i=1}^{N_t}X_i^{x,y}|}|N_t\right]\geq 1-(\sqrt{2\pi}e^{\frac{\lambda_t^2}{2}})\lambda_t.$$

This expression has exactly the symmetric form to the one obtained in the upper bound part. Thus, the lower bound is obtained the same way as the upper bound. We directly get:

$$\begin{aligned}
&\mathbb{E}_{N_t}\left[1-(\sqrt{2\pi}e^{\frac{g_t^2}{2N_t}})\frac{g_t}{\sqrt{N_t}}\right]\\
&\quad\geq 1-\sqrt{2\pi}g_t\left[\frac{e^{\frac{g_t^2}{2(1-\delta)n_t}}}{\sqrt{(1-\delta)n_t}}+e^{g_t^2-an_t}\right]
\end{aligned}$$

Now we define ϵ_t^-:

$$\epsilon_t^- = -\sqrt{2\pi}\beta_t\sigma\left[\frac{e^{\frac{\beta_t^2\sigma^2}{2(1-\delta)n_t}}}{\sqrt{(1-\delta)n_t}}+e^{\beta_t^2\sigma^2-an_t}\right].$$

It is easy to see that ϵ_t^- goes to 0 when t goes to infinity as soon as $\beta_t/\sqrt{n_t}\underset{\infty}{\to}0$. This completes the proof of Lemma 1. ■

5. Proof, Last Part: Rate of Convergence in the General Case

We first complete the proof of convergence as stated in Theorem 3 and then deduce the convergence rate (Theorem 5) from it. This enables us to provide an upper bound on the minimal number of cost function evaluations in Section 5.3.

5.1. Proof of Theorem 3

The proof of Theorem 3 follows the roadmap of Holley and Strook (Holley and Stroock, 1988) and relies on the use of the Grönwall lemma. We derive a differential inequality for the $L^2_{\mu_{\beta_t}}$-norm of the density measure of the NSA process with respect to μ_{β_t} and deduce an integrated version of it using the lemma. We then show that bounding the $L^2_{\mu_{\beta_t}}$-norm of this density implies the convergence of the process to the optimal state space χ_ϵ.

Proof. **[Proof of Theorem 3]** Our goal is to show that when t goes to infinity, the noisy simulated annealing gets "close enough" to the classical simulated annealing. Therefore we denote by f_t the Radon-Nikodym derivative of the probability density of the noisy simulated annealing process $\widetilde{X}_t$ with respect to the Gibbs measure μ_{β_t}, *i.e.*:

$$f_t=\frac{\mathrm{d}m_t}{\mathrm{d}\mu_{\beta_t}} \tag{13}$$

where m_t is the distribution of $(\widetilde{X}_s)_{s\geq0}$ at time t. A first remark is that $f_t(x)=0$ for all $t\geq0$ and all $x\in E\setminus E'$, since our process, by construction does not accept states out of E'.

Using the results obtained in Section 3 one can see that $\mathbb{R}_+\ni t\to\widetilde{L}_{\beta_t}$ is continuous and therefore the semi-group $(P_{s,t})_{0\leq s\leq t}$ is smooth . Also by their definition the operators $(P_{s,t})_{0\leq s\leq t}$ are linear and have the following semi-group property: $P_{s,t+h}=P_{s,t}\circ P_{t,t+h}$, for all $0\leq s<t$ and $h>0$. Hence, for all $0\leq s\leq t$, we have:

$$\frac{\mathrm{d}}{\mathrm{d}t}P_{s,t}=P_{s,t}L_t. \tag{14}$$

For details about the infinitesimal generator see Section 1.4. of (Bakry et al., 2013).

As shown in Equation (23), bounding the L^2-norm of f_t w.r.t. μ_{β_t}, *i.e.*, $\|f_t\|_{\mu_{\beta_t}}$, ensures convergence of the NSA algorithm. However it does not provide enough information about

the convergence of m_t to μ_{β_t} to deduce a fine convergence rate. This is why we study the evolution of $\|f_t - 1\|_{\mu_{\beta_t}}$ which controls the distance between the two measures. If this quantity is bounded then we obtain the convergence of the NSA algorithm. If moreover it converges to zero, it implies a stronger convergence rate. In order to prove that, we deduce a differential inequality for $\|f_t - 1\|^2_{\mu_{\beta_t}}$. We start by computing its derivative:

$$\begin{aligned}\partial_t \|f_t - 1\|^2_{\mu_{\beta_t}} = \partial_t \|f_t\|^2_{\mu_{\beta_t}} =& \partial_t \sum_{x\in E} f_t^2(x)\mu_{\beta_t}(x) \\ &= 2\sum_{x\in E} f_t(x)\partial_t m_t(x) + \sum_{x\in E} f_t(x)\partial_t \mu_{\beta_t}(x).\end{aligned}$$

Using the backward Kolmogorov equation given by Equation (14), for the first term we have:

$$\begin{aligned}\sum_{x\in E} f_t(x)\partial_t\left[\frac{m_t}{\mu_{\beta_t}}\right](x)\mu_{\beta_t}(x) &= \sum_{x\in E} f_t(x)\partial_t\left[\frac{m_t}{\mu_{\beta_t}}\right](x)\mu_{\beta_t}(x) + \sum_{x\in E} f_t(x)\partial_t\mu_{\beta_t}(x) \qquad (15)\\ &= \sum_{x\in E}\left[\widetilde{L}_{\beta_t} f_t(x)\right] m_t(x) - \sum_{x\in E} f_t(x)\frac{m_t}{\mu_{\beta_t}}(x)\partial_t\mu_{\beta_t}(x).\end{aligned}$$

Denote $\langle J\rangle_{\mu_{\beta_t}} := \int J d\mu_{\beta_t}$ the mean of J with respect to μ_{β_t}. One can check that:

$$\partial_t \mu_{\beta_t}(x) = -\beta_t'\left[J(x) - \langle J\rangle_{\mu_{\beta_t}}\right]\mu_{\beta_t}(x).$$

Thus, we easily obtain the following equality:

$$\partial_t \|f_t - 1\|^2_{\mu_{\beta_t}} = 2\sum_{x\in E} f_t(x)(\widetilde{L}_{\beta_t} f_t)(x)\mu_{\beta_t}(x) + \beta_t'\sum_{x\in E}(J(x) - \langle J\rangle_{\mu_{\beta_t}})f_t^2(x)\mu_{\beta_t}. \qquad (16)$$

First, we focus on the first term of the right hand side of Equation (16). Since we try to control the generator of the noisy simulated annealing by the generator of the classical one, it is natural to write $\widetilde{L}_{\beta_t}$ as $L_{\beta_t} + \widetilde{L}_{\beta_t} - L_{\beta_t}$. This comparison leads to the computation:

$$\begin{aligned}&\sum_{x\in E} f_t(x)(\widetilde{L}_{\beta_t} f_t)(x)\mu_{\beta_t}(x)\\ &= \sum_x f_t(x)(L_{\beta_t} f_t)(x)\mu_{\beta_t}(x) + \sum_{x\in E} f_t(x)\left(\sum_{y\in E}(f_t(y) - f_t(x))(\widetilde{q_{\beta_t}} - q_{\beta_t})(x,y)\right)\mu_{\beta_t}(x).\end{aligned}$$

We rewrite the last part of the second term using Lemma 1:

$$\begin{aligned}&\sum_{y\in E}(f_t(y) - f_t(x))(\widetilde{q_{\beta_t}} - q_{\beta_t})(x,y)\\ &= \sum_{y\in E} f_t(y)(\widetilde{q_{\beta_t}} - q_{\beta_t})(x,y) - \sum_{y\in E} f_t(x))(\widetilde{q_{\beta_t}} - q_{\beta_t})(x,y)\\ &\le \epsilon_t^+\sum_{y\in E} f_t(y)q_{\beta_t}(x,y) - \epsilon_t^+\sum_{y\in E} f_t(x)q_{\beta_t}(x,y) + \epsilon_t^+\sum_{y\in E} f_t(x)q_{\beta_t}(x,y) - \epsilon_t^-\sum_{y\in E} f_t(x)q_{\beta_t}(x,y)\\ &\le \epsilon_t^+\sum_{y\in E}(f_t(y) - f_t(x))q_{\beta_t}(x,y) + (\epsilon_t^+ - \epsilon_t^-)f_t(x)\\ &\le \epsilon_t^+ L_{\beta_t} f_t(x) + (\epsilon_t^+ - \epsilon_t^-)f_t(x).\end{aligned}$$

Inserting this in the previous inequality, we get:

$$\begin{aligned}&\sum_{x\in E} f_t(x)(\widetilde{L}_{\beta_t} f_t)(x)\mu_{\beta_t}(x)\\ &\le (1+\epsilon_t^+)\sum_x f_t(x)(L_{\beta_t} f_t)(x)\mu_{\beta_t}(x) + (\epsilon_t^+ - \epsilon_t^-)\sum_{x\in E} f_t^2(x)\mu_{\beta_t}(x)\\ &\le (1+\epsilon_t^+)\sum_x f_t(x)(L_{\beta_t} f_t)(x)\mu_{\beta_t}(x) + (\epsilon_t^+ - \epsilon_t^-)\sum_{x\in E}(f_t^2(x) - 1)\mu_{\beta_t}(x) + (\epsilon_t^+ - \epsilon_t^-).\end{aligned}$$

Therefore, using Equation (16), we obtain the following inequality:

$$\begin{aligned}&\frac{d}{dt}\|f_t - 1\|^2_{\mu_{\beta t}}\\ &\le 2\left[(1+\epsilon_t^+)\sum_x f_t(x)(L_{\beta_t} f_t)(x)\mu_{\beta_t}(x) + (\epsilon_t^+ - \epsilon_t^-)\sum_{x\in E}(f_t^2(x) - 1)\mu_{\beta_t}(x) + (\epsilon_t^+ - \epsilon_t^-)\right]\\ &\quad + \beta_t'\sum_{x\in E}(J(x) - \langle J\rangle_{\mu_{\beta_t}})f_t^2(x)\mu_{\beta_t}\\ &\le 2\left[(1+\epsilon_t^+)\sum_x f_t(x)(L_{\beta_t} f_t)(x)\mu_{\beta_t}(x) + (\epsilon_t^+ - \epsilon_t^-)\sum_{x\in E}(f_t^2(x) - 1)\mu_{\beta_t}(x) + (\epsilon_t^+ - \epsilon_t^-)\right]\\ &\quad + \beta_t'\sum_{x\in E}(J(x) - \langle J\rangle_{\mu_{\beta_t}})(f_t(x) - 1)^2\mu_{\beta_t}(x) + 2\beta_t'\sum_{x\in E}(J(x) - \langle J\rangle_{\mu_{\beta_t}})(f_t(x) - 1)\mu_{\beta_t(x)}.\end{aligned}$$

In order to deal with the first sum we use an estimate of the spectral gap of L_{β_t}. This is provided by Theorem 2.1 of Holley and Strook(Holley and Stroock, 1988).

Theorem 4 (Holley and Strook 88) *Under assumptions of 2.2, there exist two positive constants $0 < c \le C < +\infty$ such that $\forall\ \beta \in \mathbb{R}_+$,*

$$ce^{-\beta m^\star} \le \gamma(\beta) \le Ce^{-\beta m^\star}$$

where $\gamma(\beta) = \inf\{-\int \phi L_\beta \phi\ d\mu_\beta : \|\phi\|_{\mu_\beta} = 1$ and $\int \phi d\mu_\beta = 0\}$ and $m^\star$ is the maximum depth of a well containing a local minimum defined in Equation(4). ■

Remark 2 *The constant $m^\star$ is always strictly positive as soon as the function has a strict local minimum that is not global. This is generally the case in our setting. Also, we always have $m^\star \leq M$.*

Following of the Proof. Using the definition of f_t, one can see that $\int f_t d\mu_{\beta_t} = 1$, hence applying the theorem for $\phi = \dfrac{f_t - 1}{\|f_t - 1\|_{\mu_{\beta_t}}}$ gives:

$$-\sum_x \phi(L_{\beta_t}\phi)(x)\mu_{\beta_t}(x) \geq ce^{-\beta_t m^\star}.$$

This and the definition of L_{β_t} imply:

$$\sum_x f_t(x)(L_{\beta_t}f_t)(x)\mu_{\beta_t}(x) \leq -ce^{-\beta_t m^\star}\|f_t - 1\|^2_{\mu_{\beta_t}}.$$

J is a positive function bounded by M on E'. For all $x \in E \setminus E'$, the only points where $J > M$, we have that $J(x) = +\infty$ and therefore $\mu_{\beta_t}(x) = 0$. This implies that for all measurable functions g,

$$\sum_{x\in E}(J(x) - \langle J\rangle_{\mu_{\beta_t}})g(x)\mu_{\beta_t} \leq M\|g\|_{\mu_{\beta_t}}.$$

Putting all these terms together gives:

$$\begin{aligned}\frac{d}{dt}\|f_t - 1\|^2_{\mu_{\beta_t}} \leq 2\left[-ce^{-\beta_t m^\star}(1+\epsilon_t^+) + (\epsilon_t^+ - \epsilon_t^-) + \frac{M}{2}\beta_t'\right]\|f_t - 1\|^2_{\mu_{\beta_t}} \\ + 2M\beta_t'\|f_t - 1\|_{\mu_{\beta_t}} \\ + 2(\epsilon_t^+ - \epsilon_t^-).\end{aligned} \tag{17}$$

We denote $u_t = \|f_t - 1\|^2_{\mu_{\beta_t}}$. Considering the fact that ϵ_t^+ is a positive function we have:

$$\begin{aligned}u_t' \leq 2\left[-ce^{-\beta_t m^\star} + (\epsilon_t^+ - \epsilon_t^-) + \frac{M}{2}\beta_t'\right]u_t \\ + 2M\beta_t'\sqrt{u_t} \\ + 2(\epsilon_t^+ - \epsilon_t^-).\end{aligned} \tag{18}$$

Using that, $\forall x \in \mathbb{R}$, $\frac{1}{4}x^2 + 1 \geq x$, we get:

$$\begin{aligned}u_t' \leq 2\left[-ce^{-\beta_t m^\star} + (\epsilon_t^+ - \epsilon_t^-) + (\frac{M}{2} + \frac{M}{4})\beta_t'\right]u_t \\ + 2M\beta_t' + 2(\epsilon_t^+ - \epsilon_t^-).\end{aligned} \tag{19}$$

Let $A_t = 2ce^{-\beta_t m^\star}$ and $B_t = 2(\epsilon_t^+ - \epsilon_t^-) + 2M\beta_t'$.

Applying Grönwall's Lemma for the previous relation gives:

$$u_t \leq u_0 e^{\int_0^t -A_s + B_s \mathrm{d}s} + \int_0^t B_s e^{\int_s^t -A_h + B_h \mathrm{d}h}\mathrm{d}s. \tag{20}$$

Under Assumptions 2.2, there exist $b, d > 0$ such that $\beta_t = b\log(1+td)$. This implies:

$$\beta_t' = \frac{bd}{1+td} \text{ and } e^{-m^\star\beta_t} = \left(\frac{1}{1+td}\right)^{m^\star b}.$$

Using the definition of $\epsilon_t^+, \epsilon_t^-$ and the fact that $n_t = (1+t)^\alpha$ one can check that:

$$A_t = \mathcal{O}\left(\frac{1}{t^{m^\star b}}\right) \text{ and } B_t = \mathcal{O}\left(\frac{1}{t} \vee \frac{\log t}{t^{\alpha/2}}\right)$$

We can see that if $B_t = o(A_t)$ the second term of Equation (20) is bounded and gives us a finite upper bound on u_t. This happens as soon as:

$$m^\star b < 1 \wedge \alpha/2 \tag{21}$$

However, the condition given by Equation (21) is sufficient yet not necessary. For $\alpha > 2$, B_t becomes of the order $\mathcal{O}(1/t)$ and thus we can choose d in a way that preserves a finite upper bound of Equation (20) even for $m^\star b = 1$. One can check by direct computation that this is true for any $d < cm^\star/M$.

Let β_t and n_t be chosen in order to comply to one of the two previously mentioned conditions. Then there exists a constant K' such that

$$u_t \leq K' \text{ for all } t \in \mathbb{R}^+ \tag{22}$$

To complete the proof of Theorem 3 one can observe that for all $t \in \mathbb{R}_+$, and all $\epsilon > 0$:

$$\mathbb{P}(\widetilde{X}_t \in {}^{\mathsf{c}}\chi_\epsilon) = \mathbb{E}(\mathbb{1}_{[J^\star+\epsilon,+\infty)}(J(\widetilde{X}_t))).$$

Using the Cauchy-Schwarz inequality and the upper bound given by Equation (22) we obtain:

$$\begin{aligned}\mathbb{E}(\mathbb{1}_{[J^\star+\epsilon,+\infty}(J(\widetilde{X}_t))) &= \int_{\mathbb{R}} \mathbb{1}_{[J^\star+\epsilon,+\infty}(J(x))f_t d\mu_{\beta_t}(x) \\ &\leq \left(\int_{\mathbb{R}}(f_t)^2 d\mu_{\beta_t}(x)\right)^{\frac{1}{2}}\left(\int_{\mathbb{R}}\mathbb{1}^2_{[J^\star+\epsilon,+\infty)}(J(x))d\mu_{\beta_t}(x)\right)^{\frac{1}{2}} \\ &\leq \|f_t\|_{L^2_{\mu_{\beta_t}}}(\mu_{\beta_t}({}^{\mathsf{c}}\chi_\epsilon))^{1/2} \\ &\leq K(\mu_{\beta_t}({}^{\mathsf{c}}\chi_\epsilon))^{1/2}\end{aligned} \tag{23}$$

with $K = \sqrt{K'+1}$. This completes the proof of Theorem 3. ■

5.2. Convergence Rate

A first rate of convergence can be deduced from Theorem 3 using the concentration speed of the Gibbs measure on χ_ϵ.

$$\begin{aligned}\mu_{\beta_t}({}^c\chi_\epsilon) &= \frac{\sum_{x\in{}^c\chi_\epsilon} e^{-\beta_t J(x)}}{\sum_{x\in E} e^{-\beta_t J(x)}} \\ &= \frac{\sum_{x\in{}^c\chi_\epsilon} e^{-\beta_t J(x)}}{\sum_{x\in{}^c\chi_\epsilon} e^{-\beta_t J(x)} + \sum_{x\in\chi_\epsilon} e^{-\beta_t J(x)}} \\ &\leq \frac{(|E|-|\chi_\epsilon|)e^{-\beta_t(J^\star+\epsilon)}}{0+|\chi_\epsilon|e^{-\beta_t J^\star}} \\ &\leq \left(\frac{|E|}{|\chi_\epsilon|}-1\right)(1+td)^{-b\epsilon} \end{aligned} \tag{24}$$

As the dependency of K (Theorem 3) in b and α is not explicit, we can however not deduce an optimal choice of (b,α) from this bound. This can be achieved if we assume that Equation (21) holds and distinguish the two cases $\alpha \leq 2$ and $\alpha > 2$. Indeed, we can then improve the bound on u_t and derive a more accurate convergence rate of the algorithm. This rate can then be optimized to obtain either an upper bound of the probability of convergence to χ_ϵ for a fixed computational budget or the minimal computational budget at a fixed risk of convergence out of χ_ϵ.

Theorem 5 *Under assumptions of Section 2.2, suppose:* $\beta_t = b\log(td+1)$, $n_t = (1+td)^\alpha$ *and* $m^\star b < \min(\alpha/2, 1)$*:*

- *if* $\alpha \geq 2$, *let* b *be such that* $m^\star b < 1$ *and let* $\gamma \in (0, \alpha/2 - m^\star b)$, *Then, there exist* $\Gamma_\gamma, \Gamma_2 > 0$ *such that for* t *large enough, for all* $\epsilon > 0$,

$$\mathbb{P}(\widetilde{X}_t \in {}^c\chi_\epsilon) \leq \Gamma_\gamma\Gamma_2(1+td)^{(m^\star b-1-b\epsilon)/2} + \Gamma_2(1+td)^{-b\epsilon}$$

- *if* $\alpha < 2$, *let* b *be such that* $m^\star b < \alpha/2$ *and let* $\gamma \in (0, \alpha/2 - m^\star b)$, *Then, there exist* $\Gamma_\gamma, \Gamma_2 > 0$ *such that for* t *large enough, for all* $\epsilon > 0$,

$$\mathbb{P}(\widetilde{X}_t \in {}^c\chi_\epsilon) \leq \Gamma_\gamma\Gamma_2(1+td)^{(m^\star b-\alpha/2+\gamma-b\epsilon)/2} + \Gamma_2(1+td)^{-b\epsilon}$$

■

Remark 3 γ *is not a new parameter of the NSA algorithm. This is a technical element that enables the tuning of the computational complexity bounds of Section 5.3. As shown in the Appendix A,* Γ_γ *is of the order of* $1/\gamma$.

Remark 4 *This two bounds display the trade off between the convergence rate of the Gibbs measure to the uniform distribution over the global minima and the rate of convergence of the NSA process to the Gibbs measure. For the first bound, considering* $\alpha > 2$ *we recover the classical rate of convergence of the simulated annealing in the noise free case. This corresponds to the result of Gutjahr and Pflug (1996). The second bound provides the rate of convergence for a choice of* $\alpha < 2$. *It can be seen that* b *will have to be reduced to ensure the convergence and thus this bound exhibits clearly the trade off between cooling and estimation.*

Proof. Under assumptions of Theorem 5, the following bound on u_t can be derived from Grönwall's Lemma (for details see Appendix A):

$$u_t \leq \begin{cases}\Gamma_\gamma(1+td)^{m^\star b-1} & \text{if } \alpha \geq 2 \\ \Gamma_\gamma(1+td)^{m^\star b-\alpha/2+\gamma} & \text{if } \alpha < 2\end{cases} \tag{25}$$

Thus we can compute a new bound on the probability that $\widetilde{X}_t$ does not belong to the optimal set χ_ϵ (*cf.* 6):

$$\begin{aligned}\mathbb{P}(\widetilde{X}_t \in {}^c\chi_\epsilon) &= \int_{\mathbb{R}} \mathbb{1}_{{}^c\chi_\epsilon}(J(x))f_t d\mu_{\beta_t}(x) \\ &= \int_{\mathbb{R}} \mathbb{1}_{{}^c\chi_\epsilon}(J(x))(f_t-1)d\mu_{\beta_t}(x) + \int_{\mathbb{R}_+} \mathbb{1}_{{}^c\chi_\epsilon}(J(x))d\mu_{\beta_t}(x) \\ &\leq \left(\int_{\mathbb{R}_+}(f_t-1)^2 d\mu_{\beta_t}(x)\int_{\mathbb{R}_+}\mathbb{1}^2_{{}^c\chi_\epsilon}(J(x))d\mu_{\beta_t}(x)\right)^{1/2} + \mu_{\beta_t}({}^c\chi_\epsilon) \\ &\leq \sqrt{u_t\mu_{\beta_t}({}^c\chi_\epsilon)} + \mu_{\beta_t}({}^c\chi_\epsilon) \end{aligned} \tag{26}$$

This means that if there exist (α, b) such that $u_t = \mathcal{O}(\mu_{\beta_t})$ the convergence rate in the noisy case will be of the same order as in the classical one, but for a smaller b.

Using the previous inequality, Equation (25) and the concentration rate of the Gibbs measure given by Equation (24) we have:

$$\mathbb{P}(\widetilde{X}_t \in {}^c\chi_\epsilon) \leq \begin{cases}\Gamma_\gamma\Gamma_2(1+td)^{\frac{m^\star b-1-b\epsilon}{2}} + \Gamma_2(1+td)^{-b\epsilon} & \text{if } \alpha \geq 2 \\ \Gamma_\gamma\Gamma_2(1+td)^{\frac{m^\star b-\alpha/2+\gamma-b\epsilon}{2}} + \Gamma_2(1+td)^{-b\epsilon} & \text{if } \alpha < 2\end{cases}$$

where $\Gamma_2 = \frac{|E|}{|\chi_\epsilon|} - 1$. ■

5.3. Computational Complexity of NSA

Given the convergence rate of the algorithm, we can define $T^\star$ such that the confidence inequality constraint is satisfied at time $T^\star$.

Let N^T_{call} be the number of cost function evaluations made by the NSA until time T. This is a random variable. We define the computational cost of the algorithm as the expectation of this random variable. It can be written as:

$$\mathbb{E}\left(N^T_{call}\right) = \mathbb{E}\left(\sum_{k\geq 1}\mathbb{1}_{T_k<T}N_{T_k}\right).$$

Lemma 6 *Let* $\delta, \epsilon > 0$, $\gamma \in (0, \alpha/2 - m^\star b)$ *and*

$$T^\star = \frac{1}{d}\left(\max\left(\left(\frac{2\Gamma_\gamma}{\delta}\right)^{2/(\min(1,\frac{\alpha}{2}-\gamma)-m^\star b+b\epsilon)}, \left(\frac{2\Gamma_2}{\delta}\right)^{1/b\epsilon}\right) - 1\right).$$

Then, for all $t \geq T^\star$, $\mathbb{P}(\widetilde{X}_t \in {}^c\chi_\epsilon) \leq \delta$ and the computational cost up to time $T^\star$ is bounded:

$$\mathbb{E}\left(N_{call}^{T^*}\right) \leq \frac{1}{d}\max\left(\left(\frac{2\Gamma_\gamma}{\delta}\right)^{2(\alpha+1)/(\min(1,\frac{\alpha}{2}-\gamma)-m^\star b+b\epsilon)}, \left(\frac{2\Gamma_2}{\delta}\right)^{(\alpha+1)/b\epsilon}\right).$$

■

Proof. In order to prove this statement, we use the inequalities from Theorem 5 treating each term separately.
We consider T_1, T_2 such that $\Gamma_\gamma\Gamma_2(1+dT_1)^{m^\star b-\min(1,\frac{\alpha}{2}-\gamma)-b\epsilon} = \delta/2$ and $\Gamma_2(1+dT_2)^{-b\epsilon} = \delta/2$. This implies:

$$1+dT_1 = \left(\frac{2\Gamma_\gamma}{\delta}\right)^{2/(\min(1,\frac{\alpha}{2}-\gamma)-m^\star b+b\epsilon)} \text{ and } 1+dT_2 = \left(\frac{2\Gamma_2}{\delta}\right)^{1/b\epsilon}. \tag{27}$$

Now we can define $T^\star$, the time after which the current state of the NSA belongs to χ_ϵ with probability at least $1-\delta$, *i.e.*, $\forall t > T^\star,\ \mathbb{P}(\widetilde{X}_t \in \chi_\epsilon) \geq 1-\delta$:

$$T^\star = \max(T_1, T_2).$$

We are interested in the computational cost up to time $T^\star$, more precisely the expected number of Monte Carlo simulations used up to $T^\star$. This is given by $\mathbb{E}\left(\sum_{k\geq 1}\mathbb{1}_{T_k<T^\star}N_{T_k}\right)$. The value of this quantity cannot be computed exactly, but it can easily be upper bounded.

$$\begin{aligned}\mathbb{E}\left(\sum_{k\geq 1}\mathbb{1}_{T_k<T^\star}N_{T_k}\right) =& \mathbb{E}\left(\mathbb{E}\left(\sum_{k\geq 1}\mathbb{1}_{T_k<T^\star}N_{T_k}\right)|(T_k)_{k=1\cdots+\infty}\right)\\ =& \mathbb{E}\left(\sum_{k\geq 1}\mathbb{1}_{T_k<T^\star}n_{T_k}\right)\\ \leq& \mathbb{E}\left(\sum_{k\geq 1}\mathbb{1}_{T_k<T^\star}\right)n_{T^\star}\end{aligned}$$

The last inequality is implied by the fact that n_t is an increasing function. Since $\sum_{k\geq 1}\mathbb{1}_{T_k<T^\star}$ is a Poisson variable of parameter $T^\star$, using the definition of n_t one can see that:

$$\begin{aligned}\mathbb{E}\left(\sum_{k\geq 1}\mathbb{1}_{T_k<T^\star}N_{T_k}\right) \leq& T^\star(1+dT^\star)^\alpha \\ \leq& \frac{1}{d}(1+dT^\star)^{\alpha+1}.\end{aligned} \tag{28}$$

We conclude using Equation (27). ■

The rate of growth of the total computation number is mainly driven by the exponent of $\frac{1}{\delta}$ in the cost function. We are looking for the couple (α, b) that minimizes this quantity and fulfills the requirements of Theorem 5. We can split the problem into two sub-problems:

Case 1: $\frac{\alpha}{2}-\gamma > 1$

$$\begin{aligned}&\min_{b,\alpha}\ \max\left(\frac{2(\alpha+1)}{1-m^\star b+b\epsilon}, \frac{\alpha+1}{b\epsilon}\right)\\ &s.t.\\ &\quad m^\star b < 1 \text{ and } \alpha - 2\gamma > 2\end{aligned} \tag{29}$$

Case 2: $\frac{\alpha}{2} \leq 1$

$$\begin{aligned}&\min_{b,\alpha}\ \max\left(\frac{2(\alpha+1)}{\alpha/2-\gamma-m^\star b+b\epsilon}, \frac{\alpha+1}{b\epsilon}\right)\\ &s.t.\\ &\quad 0 < \gamma < \frac{\alpha}{2} - m^\star b \text{ and } \alpha - 2\gamma \leq 2\end{aligned} \tag{30}$$

The solution of Equation (29) is obvious, the minimal value for α and the maximal for b, *i.e*, α must be as close to 2 as possible and $b = \frac{1}{m^\star+\epsilon}$. As for Equation (30), we consider two sub-cases. First suppose that:

$$\frac{2(\alpha+1)}{\alpha/2-\gamma-m^\star b+b\epsilon} \geq \frac{\alpha+1}{b\epsilon} \iff \alpha/2-\gamma-m^\star b \leq b\epsilon. \tag{31}$$

The function we want to minimize is strictly decreasing in α and strictly increasing in b, so its minimum value is attained for the maximal value of α and the minimal value of b, under the domain constraints given by Equation (30) and Equation (31), so the solution is:

$$\alpha = 2(1+\gamma) \text{ and } b > \frac{1}{m^\star+\epsilon}. \tag{32}$$

In the second sub-case, supposing that the inequality Equation (31) is inverted, the problem can be resumed at minimizing $(\alpha+1)/b\epsilon$, a decreasing function with respect to b, for

$$b \leq \frac{\alpha/2-\gamma}{(m^\star+\epsilon)} \text{ and } \alpha \leq 2.$$

Replacing b by its maximal value the objective function becomes a decreasing function in α, and therefore we obtain the same solution as before, defined in Equation (32). This is a quite comprehensive result, as it indicates that the lower the required accuracy in the solution space is, *i.e.*, ϵ increases and thus the size of χ_ϵ does too, the faster the temperature can decrease to zero. We need to explore less the state space.

Corollary 1 *For the optimal parameters choice defined in Equation (32), an ϵ-optimal solution is returned by NSA with probability $1-\delta$ at a computational cost at most :*

$$\frac{1}{d}\left(\frac{2\Gamma_\gamma}{\delta}\right)^{\frac{m^\star+\epsilon}{\epsilon}(3+2\gamma)},$$

where Γ_γ is defined in Theorem 5.

This is rather costly but represents a general bound with few constraints on the function J. However, if the function J has additional properties the bound can be significantly improved:

Corollary 2 *Suppose that J has no well containing a local minimum, apart from the one containing the global minimum, i.e. $m^\star = 0$, then an ϵ-optimal solution is returned by NSA with probability $1-\delta$ at a computational cost at most :*

$$\left(\frac{2\log\frac{1}{\delta}}{d\epsilon}\right)^3.$$

Remark 5 *We recover the polynomial dependency in $1/\epsilon$ and $\log 1/\delta$ of the state-of-the-art complexity results (c.f. Woodroofe 1972 and Nemirovski et al. 1982) which are of the order of $\epsilon^{-2}\log(\frac{1}{\delta})$ for strongly convex cost functions. As we relax this assumption and only consider cost functions with no local minimum, it seems coherent to observe a slight degradation of the complexity.*

Proof. In order to have an estimate of the computational cost in this setting we follow the same method as before and highlight only the main steps of the proof. First remark that in this case, Theorem 4 states that there exist $C, c > 0$ such that $\forall \beta \in \mathbb{R}^+$:

$$c \leq \gamma(\beta) \leq C \tag{33}$$

This changes the differential inequality obtained for $u_t = \|f_t - 1\|^2_{\mu_{\beta_t}}$ and thus Equation (19) becomes :

$$u'_t \leq 2\left[-c + (\epsilon_t^+ - \epsilon_t^-) + (\frac{M}{2} + \frac{M}{4})\beta'_t\right] u_t + 2M\beta'_t + 2(\epsilon_t^+ - \epsilon_t^-).$$

We can apply Grönwall's Lemma and obtain the same type of inequality as before:

$$u_t \leq u_0 e^{\int_0^t -A_s + B_s \mathrm{d}s} + \int_0^t B_s e^{\int_s^t -A_h + B_h \mathrm{d}h}\mathrm{d}s. \tag{34}$$

where B_t has the same form as before, $B_t = 2(\epsilon_t^+ - \epsilon_t^-) + 2M\beta'_t$ and $A_t = 2c$.

The convergence of u_t towards 0 can be proved now for a larger class of functions n_t, β_t, since:

$$A_t = \mathcal{O}(1) \text{ and } B_t = \mathcal{O}(\beta_t/\sqrt{n_t} \vee \beta'_t).$$

We no longer need to impose $\beta_t = \mathcal{O}(\log_t)$. Let $\alpha, b, d > 0$. Define

$$n_t = (1+t)^\alpha \text{ and } \beta_t = d(1+t)^b.$$

Using Equation (34) one can check that we have a finite upper bound on u_t as soon as:

$$\{b < \alpha/2 \wedge 1, d > 0\} \text{ or } \{b = 1, \alpha \geq 2, 0 < d < c\}.$$

This in particular implies that the NSA algorithm converges a.s. to the set of global minimums of J. Furthermore for the first set of conditions one can prove using the same technique as in Appendix A that :

$$u_t = \mathcal{O}(t^{\max(b-1, b-\alpha/2)}).$$

This means that there exits $\Gamma'_\gamma > 0$ such that for t large enough $u_t \leq \Gamma'_\gamma t^{-\gamma}$, where $\gamma = -\max(b-1, b-\alpha/2)$. Using this, Equation (26) and Equation (24), for t large enough, we get:

$$\begin{aligned}\mathbb{P}(\widetilde{X}_t \in {}^{\mathsf{c}}\chi_\epsilon) &\leq \Gamma'_\gamma t^{-\gamma/2} e^{-\epsilon d(1+t)^b/2} + \Gamma_2 e^{-\epsilon d(1+t)^b} \\ &\leq e^{-\epsilon d(1+t)^b/2}(\Gamma'_\gamma t^{-\gamma/2} + \Gamma_2 e^{-\epsilon d(1+t)^b/2}) \\ &\leq e^{-\epsilon d(1+t)^b/2}\end{aligned}$$

The last inequality is valid as soon as $t > \max\left[\left(\frac{2\log(2\Gamma_2)}{\epsilon d}\right)^{\frac{1}{b}} - 1, \left(2\Gamma'_\gamma\right)^{\frac{2}{\gamma}}\right]$. This is not a restrictive condition. Take for example the minimization of the $\|.\|_1$ over the subset subset $E = \{x \in \mathbb{Z}^p, \|x\|_\infty \leq n\}$ for some $n \in \mathbb{N}$. As $\Gamma_2 = \|E\| - 1 = (n+1)^p - 1$, the time for which the first part of the condition is fulfilled only grows linearly with the dimension of the search space. We show latter on that the optimal choice for b is one and thus the second part of the condition can be omitted.

Let $\delta > 0$ be a fixed. Using the previous inequality one can compute $T^\star$ such that the confidence inequality constraint is satisfied:

$$T^\star_{\epsilon,\delta} = \left(-\frac{2\log\delta}{d\epsilon}\right)^{1/b} - 1.$$

Regarding the computational cost we remind the reader that Equation (28) implies:

$$\mathbb{E}(N^{T^\star}_{call}) \leq n_{T^\star} T^\star \leq \left(-\frac{2\log\delta}{d\epsilon}\right)^{\frac{1+\alpha}{b}}.$$

We can optimize this bound with respect to α and b in the same way as for Corollary 1. This leads to $\alpha = 2$ and $b < 1$ and thus to the desired results:

$$\mathbb{E}(N^{T^\star}_{call}) \leq \left(-\frac{2\log\delta}{d\epsilon}\right)^3.$$

■

6. Numerical Experiments

In this section we first present some test cases, for which we use an additive Gaussian noise at each evaluation. We recover the theoretical results introduced by Gutjahr and Pflug (1996). In a second part we present some results for the aircraft trajectory optimization problem. In this case the solution of the problem is unknown. We can only observe the total cost improvement in comparison with a trajectory optimized for a similar but deterministic setting.

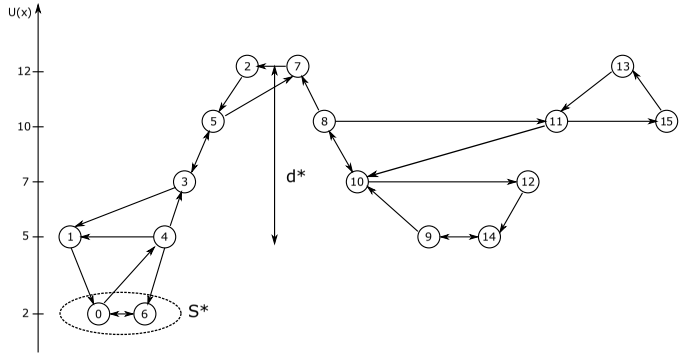


Figure 2: B. Hajek test case for the simulated annealing in a deterministic environment

6.1. Basic Exemple

The first experimental setting we consider, was introduced in (Hajek, 1988). The cost function and the neighbourhood structure are represented on Figure 2. This is of particular interest as the function has two basins from which it is hard to escape. B. Hajek has shown that the following holds:

Theorem 7 (Hajek 1988)
If $\beta_k = b\ log(k+2)$, then $b \leq d^\star \Leftrightarrow \lim_{k\to\infty} \mathbb{P}(X_k \in S^\star) = 1$, *where $d^\star$ is the maximum depth of a cup containing a local but not global minima. The depth of a cup is the maximal energy difference between two of it states and $(X_k)_{k\in\mathbb{N}}$ denotes the Markov chain generated by the classical simulated annealing.* ■

For a complete definition of $d^\star$, see Hajek (1988).

We add Gaussian noises to the cost function of Figure 2 with different variance levels to highlight the fact that if only one sampling is performed the simulated annealing performance becomes rapidly very poor as the variance increases. On the other hand it appears that the performance of the NSA for a linear increase of the mean number of samples is as good a quadratic one. These results are summarized on Figure 3.

6.2. Ackley Test Function

We introduce a second test case to further asses these observations. We consider the uniformly (2000 points) discretized version of the Ackley function in one dimension on $[-100, 100]$. This function has many local minima as shown on Figure 4.

Figures 5 displays the convergence results on this test case for different levels of variance of the added Gaussian noise, for each estimation schedule introduced in this paper. As before, the success ratio of the algorithm is established using 1000 Monte Carlo runs for each noise level and estimation schedule, by comparing at every run the returned value with

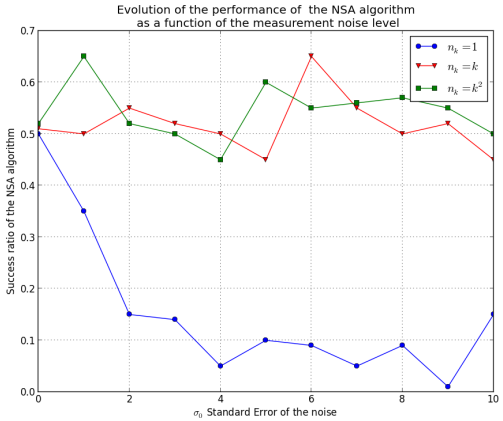


Figure 3: Convergence performance of NSA for the Hajek setting. Each line represents the evolution of the success ratio of the algorithm when the variance of the noise added to the cost function increases. The success ratio is measured over 1000 Monte Carlo simulations for each level of noise. A run is considered successful if the algorithm returns a true optimal value. One can see that the performance in the case where the mean sample number is generated using $n_k = k^2$ (green curve) or $n_k = k$ (red curve) is similar and quite stable, as opposed to case of only one sample used at each step (blue curve), where the success ratio decreases rapidly.

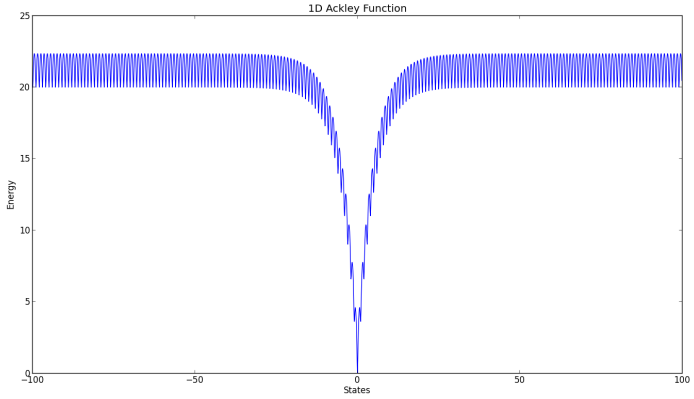


Figure 4: Ackley 1D Test Function

the true optimizer of the cost function. We observe that the only case where the convergence is not impacted by the noise variance increase is the $n_t = t^2$ case (corresponding to the use of approximately k^2 measurements at each step k). Although, when the variance of the noise is rather small (equal to 1), the algorithm's performance using the linear schedule ($n_t = t$) or the quadratic one ($n_t = t^2$) is quite similar, a clear gap is highlighted between the two schedules as soon as the variance increases.

These results also highlight the fact that a logarithmic sampling schedule is not appropriate for any cost function, even in the Gaussian case. This invalidates partially the hypotheses introduced in Fink (1998).

6.3. Aircraft Trajectory Optimization

We use a black box trajectory evaluator for a long range commercial aircraft. We consider a direct shooting method for optimizing the vertical part of the trajectory. As displayed on Figure 6, the vertical path is made of a sequence of flight segments at constant altitude called steps. The transitions between those steps are called step climbs. This has been put in place by the international authorities to ease the air traffic control. Aircraft can only fly at a finite set of altitudes. The steps climbs are transition phases that must be very short. The Figure 6 is a conceptual. It does not reflect the real scale of the different phases. Our optimization variables are the vectors of position of the steps and the vector of steps' altitude, denoted respectively x and h on Figure 6. The structure of this airspace strongly limits the number of steps. We will only consider the problem with an *a priori* number of steps. There are two main reasons why the aircraft might vary its altitude during a flight

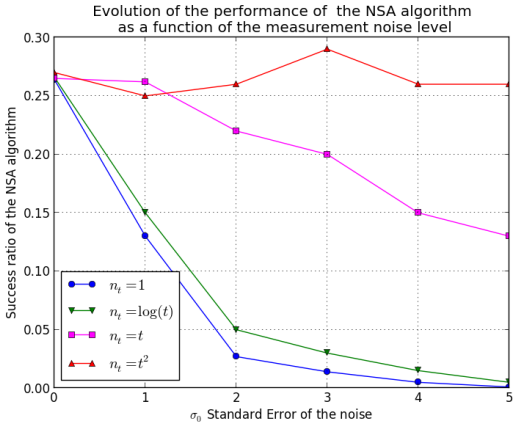


Figure 5: Performance of the NSA algorithm as a function of the level of noise on the evaluation of the cost function.

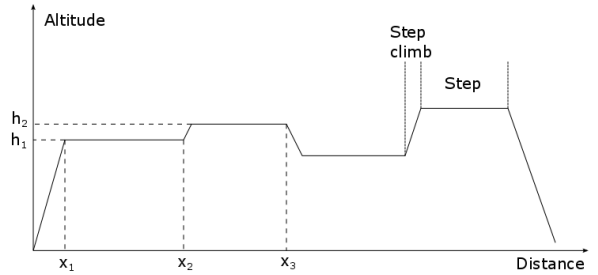


Figure 6: Aircraft trajectory, structure of the vertical path

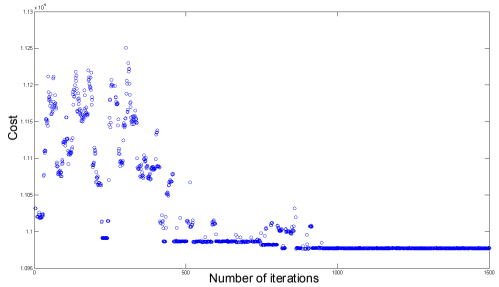


Figure 7: NSA descent: Aircraft trajectory optimization problem

(optimizing fuel consumption and air traffic control). Because of fuel consumption, the aircraft weight is decreasing during the flight. Analyzing the laws of flight physics, it can be shown that there exists an altitude at which the fuel consumption per flown distance unit is minimal. It can also be shown that this altitude increases as the weight decreases. This last statement is however only true if there is no wind. It is easily understandable that for some particular wind map configuration it might be preferable to target lower altitudes at lower weights.
The choice of the vertical path must be declared to the authorities before the flight to ensure traffic manageability. Airlines operating aircraft have therefore a stochastic optimization problem to solve. This is a stochastic problem for two main reasons. First, they only access predicted weather conditions that suffer some uncertainty. Second, the airspace is not empty and sometimes air traffic controllers might refuse some altitude changes because of the presence of other aircraft. As the weather, the traffic is not known in advance.
We applied NSA to the problem of finding an optimal 3 steps configuration. An example of the current solution cost evolution with respect to the number of iterations is displayed on Figure 7.

We observe a very quick convergence to a low cost trajectory. We do not claim it is a general behaviour. It might be due to the structure of the cost function. Figure 8, shows how the cost evolves with respect to the ground position of the first step. It is obviously not convex but has some regularity. We can observe some flat parts. This explains why gradient based methods would fail solving this problem. However, we are not aware of any other study solving the Aircraft trajectory optimization problem under uncertainty using a Black box aircraft model and are thus unable to provide more advance appreciation of the performance of such an approach.

As for the previous problems we have observed that the increase sampling condition must be satisfied to ensure a good behaviour of the algorithm.

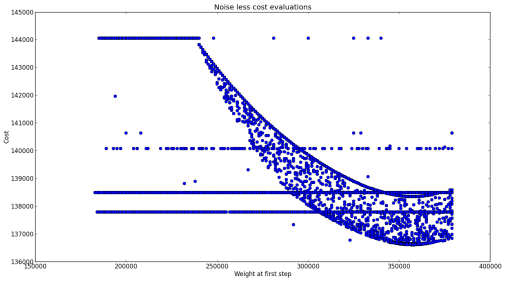


Figure 8: Sampling of the cost function along the first step position for a 3 step vertical path

Appendix A. Proof of Bound Equation (25)

Proof.[Proof of Theorem 5]
Let $\beta_t = b\log(td+1)$ and $n_t = (1+td)^\alpha$.
Recall Equation (18):

$$u'_t \leq 2\left[-ce^{-\beta_t m^\star} + (\epsilon^+ - \epsilon^-) + \left(\frac{M}{2} + \frac{M}{4}\right)\beta'_t\right] u_t + 2M\beta'_t + 2(\epsilon^+ - \epsilon^-)$$

Let $A_t = 2ce^{-\beta_t m^\star}$ and $B_t = 2(\epsilon^+ - \epsilon^-) + 2M\beta'_t$.

Applying Grönwall's Lemma for the previous relation gives:

$$u_t \leq u_0 e^{\int_0^t -A_s + B_s \mathrm{d}s} + \int_0^t B_s e^{\int_s^t -A_h + B_h \mathrm{d}h} \mathrm{d}s$$

Under Assumptions 2.2, there exist $b, d > 0$ such that $\beta_t = b\log(1+td)$. This implies:

$$\beta'_t = \frac{bd}{1+td} \text{ and } e^{-m^\star \beta_t} = \left(\frac{1}{1+td}\right)^{m^\star b}.$$

Using the definition of ϵ^-, ϵ^+ we have:

$$\epsilon^+ - \epsilon^-_t = 2\sqrt{2\pi}\beta_t\sigma\left[\frac{e^{\frac{\beta_t^2\sigma^2}{2(1-\delta)n_t}}}{\sqrt{(1-\delta)n_t}} + e^{\beta_t^2\sigma^2 - an_t}\right].$$

This implies that when t goes to infinity:

$$A_t = \mathcal{O}\left(\frac{1}{t^{m^\star b}}\right) \text{ and } B_t = \mathcal{O}\left(\frac{1}{t} \vee \frac{\log t}{t^{\alpha/2}}\right)$$

In order to highlight the mains ideas of the proof we will try to simplify the notations as much as possible. First observe that for all $\alpha > 0$ and all $0 < \gamma < \alpha/2$, $\frac{\log t}{t^{\alpha/2}} = o\left(\frac{1}{t^{\alpha/2-\gamma}}\right)$.

Hence we can assume there exist $A, B > 0$ and δ_1, δ_2 such that:

$$A_t = A\frac{d}{(1+td)^{\delta_1}} \text{ and } B_t \leq B\frac{d}{(1+td)^{\delta_2}}$$

where $\delta_1 = m^\star b$ and $\delta_2 = \min(1, -\gamma + \alpha/2)$. Since $\min(1, \alpha/2) > m^\star b$, and γ can be chosen arbitrarily close to 0, we choose it such that $\delta_1 < \delta_2$. This means that $0 < \gamma < \alpha/2 - m^\star b$.

Remark 6 *The choice of γ influences the choice of B. If $B_t = \mathcal{O}\left(\frac{\log t}{t^{\alpha/2}}\right)$, there exists $C_B > 0$ such that $B_t \leq C_B\frac{\log t}{t^{\alpha/2}}, \forall t$. The constant B is then such that $\forall t$, $C_B\frac{\log t}{t^\gamma} \leq B$. Hence we can choose:*

$$B = \frac{C_B}{e\gamma}. \tag{35}$$

Let $T_t^1 = u_0 e^{\int_0^t -A_s + B_s \mathrm{d}s}$ and $T_t^2 = \int_0^t B_s e^{\int_s^t -A_h + B_h \mathrm{d}h}\mathrm{d}s$.

The first term T_t^1 is always easy to deal with and one can check that under the theorem's assumptions we always have $T_t^1 = o(1/t^{\delta_1 - \delta_2})$ when t goes to infinity. As for the second term, using a substitution gives:

$$\begin{aligned} T_t^2 &\leq \int_1^{1+td} \frac{B}{s^{\delta_2}} e^{\int_s^{1+td} -\frac{A}{h^{\delta_1}} + \frac{B}{h^{\delta 2}}\mathrm{d}h}\mathrm{d}s \\ &\leq e^{-\frac{A(1+td)^{1-\delta_1}}{1-\delta_1} + \frac{B(1+td)^{1-\delta_2}}{1-\delta_2}} \int_1^{1+td} \frac{B}{s^{\delta_2}} e^{\frac{As^{1-\delta_1}}{1-\delta_1} - \frac{Bs^{1-\delta_2}}{1-\delta_2}}\mathrm{d}s \end{aligned} \tag{36}$$

For the last inequality we assume $\delta_2 \neq 1$ which corresponds to the case $\alpha \leq 2$.

Let $I_t = \int_1^{1+td} \frac{B}{s^{\delta_2}} e^{\frac{As^{1-\delta_1}}{1-\delta_1} - \frac{Bs^{1-\delta_2}}{1-\delta_2}}\mathrm{d}s$ and $f_s = \frac{As^{1-\delta_1}}{1-\delta_1} - \frac{Bs^{1-\delta_2}}{1-\delta_2}$. Let T_0 be such that for all $s \geq T_0$, $s^{\delta_2-\delta_1} \geq \frac{B+1}{A}$ (for instance $T_0 = (\frac{B}{A}+1)^{1/(\delta_2-\delta_1)}$). We can write I_t as follows:

$$\begin{aligned} I_t &= \int_1^{T_0} \frac{B}{s^{\delta_2}} e^{f_s}\mathrm{d}s + \int_{T_0}^{1+td} \frac{B}{s^{\delta_2}} e^{f_s}\mathrm{d}s \\ &= K_{T_0} + \int_{T_0}^{1+td} \frac{B}{s^{\delta_2}(As^{-\delta_1} - Bs^{-\delta_2})} e^{f_s} f_s' \mathrm{d}s \\ &= K_{T_0} + \left[\frac{B}{As^{\delta_2-\delta_1} - B} e^{f_s}\right]_{T_0}^{1+td} + \int_{T_0}^{1+td} \frac{AB(\delta_2-\delta_1)s^{\delta_2-\delta_1-1}}{(As^{\delta_2-\delta_1}-B)^2} e^{f_s}\mathrm{d}s \end{aligned}$$

Since $\delta_1 < \delta_2$, $\frac{A(\delta_2-\delta_1)s^{-\delta_1-1}}{(As^{\delta_2-\delta_1}-B)^2}$ goes to 0 when s goes to infinity. Moreover one can check that for all $s \geq T_0$ this quantity is smaller than $1/2$. Using this we get:

$$I_t \leq K_{T_0} + \left[\frac{B}{As^{\delta_2-\delta_1} - B} e^{f_s}\right]_{T_0}^{1+td} + \frac{1}{2} I_t$$

and therefore for all $t \geq T_0$:

$$I_t \leq 2\left[\frac{B}{As^{\delta_2-\delta_1} - B} e^{f_s}\right]_1^{1+td} + 2K_{T_0}$$

This gives:

$$\begin{aligned} T_t^2 &\leq 2e^{-f_{1+td}}\left(\left[\frac{B}{As^{\delta_2-\delta_1} - B} e^{f_s}\right]_1^{1+td} + K_{T_0}\right) \\ &\leq 2\left[\frac{B}{A(1+td)^{\delta_2-\delta_1} - B}\right] + 2\left(K_{T_0} - \frac{B}{A-B}e^{f_1}\right)e^{-f_{1+td}} \end{aligned}$$

Regrouping the terms we obtain for all $t \geq T_0$:

$$u_t \leq \left[u_0 + 2\left(K_{T_0} - \frac{B}{A-B}e^{f_1}\right)\right]e^{-f_{1+td}} + \frac{2B}{A(1+td)^{\delta_2-\delta_1} - B} \tag{37}$$

Since the first term is a $\mathcal{O}(e^{-f_t})$ and therefore a $o(t^{\delta_1-\delta_2})$ it is obvious that:

$$u_t = \mathcal{O}\left(\frac{1}{t^{\delta_2-\delta_1}}\right) \text{ when } t \to \infty \tag{38}$$

This means that for all $\gamma \in (0, \alpha/2 - m^\star b)$ there exists $\Gamma_\gamma > 0$ such that:

$$u_t \leq \Gamma_\gamma t^{-\alpha/2 + m^\star b + \gamma} \text{ for all } t \geq T_0. \tag{39}$$

Since $f_t = \mathcal{O}(t^{1-\delta_1})$, $e^{-f_t}t^{\delta_1-\delta_2}$ goes very fast to zero and therefore the size of Γ_γ is mainly driven by $\frac{2B}{A}$. Using Equation (35) one can see that:

$$\Gamma_\gamma \simeq \frac{1}{\gamma}.$$

If $\alpha > 2$, $\delta_2 = 1$ and Equation (36) becomes of the form:

$$T_t^2 \leq e^{-\frac{A(1+td)^{1-\delta_1}}{1-\delta_1} + B\log(1+td)} \int_1^{1+td} \frac{B}{s^{1+B}} e^{\frac{As^{1-\delta_1}}{1-\delta_1}}\mathrm{d}s.$$

Using a similar procedure one can check that in this case we also have an inequality similar to Equation (37) , T_2^t remains a $\mathcal{O}\left(\frac{1}{t^{1-\delta_1}}\right)$, and Equation (38) and Equation (39) still hold.

Remark 7 *In this case one can choose $\gamma = \alpha/2 - 1$, this way $\min(1, \frac{\alpha}{2} - \gamma) = 1$ and Γ_γ is minimal.*

■

Appendix B. Definition of $m^\star$

In this section we prove that the definition of $m^\star$, *i.e.* Equation (4), is equivalent to the definition provided by Holley and Stroock (1988).

Lemma 3 *Let,* $m^\star_{HS} := \max_{x,y\in E}\left\{\min_{p\in P_{xy}}\left\{\max_{z\in p} J(z)\right\} - J(y) - J(x) + \min_u J(u)\right\}$.
Then

$$m^\star = m^\star_{HS}$$

■

Proof. Let $x, y \in E$ and denote $H_{xy} := \min_{p\in P_{xy}}\left\{\max_{z\in p} J(z)\right\}$.
First it can be noticed that if x is a global minimum of J then we have

$$H_{x,y} - J(y) - J(x) + \min_u J(u) = H_{x,y} - J(y) \tag{40}$$

Thus $m^\star_{HS} \geq H_{x^\star,y} - J(y)$ for any y in E, where $x^\star$ is a global minimum of J.

Recall that $m^\star = \max_{x,y\in E}\{H_{xy} - \max(J(y), J(x))\}$. As the set of paths going from x to y containing a global minimum $x^\star$ is a subset of the paths going from x to y, we have:

$$H_{xy} \leq \max(H_{x^\star x}, H_{x^\star y})$$

Let $x, y \in E$ such that $m^\star = H_{xy} - \max(J(y), J(x))$,

$$\begin{aligned} m^\star &\leq \max(H_{x^\star x}, H_{x^\star y}) - \max(J(y), J(x)) \\ &\leq \max(H_{x^\star x} - J(x), H_{x^\star y} - J(y)) \\ &\leq m^\star_{HS} \end{aligned}$$

On the other hand, as $\forall\, x, y \in E$, we have $-\min(J(y), J(x)) + \min_u J(u) \leq 0$, so

$$H_{xy} - J(y) - J(x) + \min_u J(u) \leq H_{xy} - \max(J(y), J(x))$$

This implies $m^\star_{HS} \leq m^\star$, which completes the proof. ■

References

E. Aarts and J. Korst. Simulated annealing and Boltzmann machines. New York, NY; John Wiley and Sons Inc., 1988.

Emile HL Aarts and Jan HM Korst. Boltzmann machines for travelling salesman problems. European Journal of Operational Research, 39(1):79–95, 1989.

Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In COLT-23th Conference on Learning Theory-2010, pages 13–p, 2010.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. Machine learning, 47(2-3):235–256, 2002.

Dominique Bakry, Ivan Gentil, and Michel Ledoux. Analysis and geometry of Markov diffusion operators, volume 348. Springer Science & Business Media, 2013.

John T. Betts. Survey of numerical methods for trajectory optimization. Journal of Guidance,Control, and Dynamics, 21(2):193–207, March 1998.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. Concentration inequalities: A nonasymptotic theory of independence. Oxford university press, 2013.

J. Branke, S. Meisel, and C. Schmidt. Simulated annealing in the presence of noise. Journal of Heuristics, 14(6):627–654, 2008.

Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvári. X-armed bandits. Journal of Machine Learning Research, 12:1587–1627, 2011.

Adam D Bull. Convergence rates of efficient global optimization algorithms. The Journal of Machine Learning Research, 12:2879–2904, 2011.

Adam D Bull et al. Adaptive-treed bandits. Bernoulli, 21(4):2289–2307, 2015.

Laurence Charles Ward Dixon and Giorgio Philip Szegö. Towards global optimisation 2. North-Holland Amsterdam, 1978.

T.MA. Fink. Inverse protein folding, hierarchical optimisation and tie knots. PhD thesis, University of Cambridge, 1998.

S. B. Gelfand and S. K. Mitter. Simulated annealing with noisy or imprecise energy measurements. J. Optim. Theory Appl., 62(1):49–62, 1989. ISSN 0022-3239. doi: 10.1007/BF00939629. URL `http://dx.doi.org/10.1007/BF00939629`.

W.J. Gutjahr and G.Ch. Pflug. Simulated annealing for noisy cost functions. Journal of Global Optimization, 8(1):1–13, 1996.

B. Hajek. Cooling schedules for optimal annealing. Mathematics of operations research, 13 (2):311–329, 1988.

R. Holley and D. Stroock. Simulated annealing via Sobolev inequalities. Comm. Math. Phys., 115(4):553–569, 1988. ISSN 0010-3616. URL `http://projecteuclid.org/euclid.cmp/1104161084`.

Tito Homem-de Mello. Variable-sample methods and simulated annealing for discrete stochastic optimization. 2000.

Reiner Horst and Panos M Pardalos. Handbook of global optimization, volume 2. Springer Science & Business Media, 2013.

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

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. Journal of Global optimization, 13(4):455–492, 1998.

John DC Little, Katta G Murty, Dura W Sweeney, and Caroline Karel. An algorithm for the traveling salesman problem. Operations research, 11(6):972–989, 1963.

Rémi Munos. From bandits to monte-carlo tree search: The optimistic principle applied to optimization and planning. 2014.

Arkadi Nemirovski, D-B Yudin, and E-R Dawson. Problem complexity and method efficiency in optimization. 1982.

Michael Woodroofe. Normal approximation and large deviations for the robbins-monro process. Probability Theory and Related Fields, 21(4):329–338, 1972.

Journal of Machine Learning Research 20 (2019) 1-37 Submitted 12/16; Revised 1/18; Published 2/19

Non-Convex Projected Gradient Descent for Generalized Low-Rank Tensor Regression

Han Chen han.chen@wisc.edu

Garvesh Raskutti raskutti@stat.wisc.edu
Department of Statistics
University of Wisconsin–Madison
Madison, WI 53706, USA

Ming Yuan ming.yuan@columbia.edu
Department of Statistics
Columbia University
New York, NY 10027, USA

Editor: Sujay Sanghavi

Abstract

In this paper, we consider the problem of learning high-dimensional tensor regression problems with low-rank structure. One of the core challenges associated with learning high-dimensional models is computation since the underlying optimization problems are often non-convex. While convex relaxations could lead to polynomial-time algorithms they are often slow in practice. On the other hand, limited theoretical guarantees exist for non-convex methods. In this paper we provide a general framework that provides theoretical guarantees for learning high-dimensional tensor regression models under different low-rank structural assumptions using the projected gradient descent algorithm applied to a potentially non-convex constraint set Θ in terms of its *localized Gaussian width* (due to Gaussian design). We juxtapose our theoretical results for non-convex projected gradient descent algorithms with previous results on regularized convex approaches. The two main differences between the convex and non-convex approach are: (i) from a computational perspective whether the non-convex projection operator is computable and whether the projection has desirable contraction properties and (ii) from a statistical error bound perspective, the non-convex approach has a superior rate for a number of examples. We provide three concrete examples of low-dimensional structure which address these issues and explain the pros and cons for the non-convex and convex approaches. We supplement our theoretical results with simulations which show that, under several common settings of generalized low rank tensor regression, the projected gradient descent approach is superior both in terms of statistical error and run-time provided the step-sizes of the projected descent algorithm are suitably chosen.

Keywords: tensors, non-convex optimization, high-dimensional regression, low-rank;

1. Introduction

Parameter estimation in high-dimensional regression has received substantial interest over the past couple of decades. See, e.g., Buhlmann and van de Geer (2011); Hastie et al. (2015). One of the more recent advances in this field is the study of problems where the parameters and/or data take the form of a multi-way array or *tensor*. Such problems arise in many practical settings (see, e.g., Cohen and Collins, 2012; Li and Li, 2010; Semerci et al., 2014; Sidiropoulos and Nion, 2010) and present a number of additional challenges that do not arise in the vector or matrix setting. In particular, one of the challenges associated with high-dimensional tensor regression models is how to define low-dimensional structure since the notion of rank is ambiguous for tensors (see, e.g., Koldar and Bader, 2009). Different approaches on how to impose low-rank and sparsity structure that lead to implementable algorithms have been considered. See, e.g., Gandy et al. (2011); Mu et al. (2014); Raskutti and Yuan (2015); Tomioka et al. (2013); Yuan and Zhang (2014), and references therein. All of the previously mentioned approaches have relied on penalized convex relaxation schemes and in particular, many of these different approaches have been encompassed by Raskutti and Yuan (2015). The current work complements these earlier developments by studying the non-convex projected gradient descent (PGD) approaches to generalized low-rank tensor regression.

While convex approaches are popular since greater theoretical guarantees have been provided for them, non-convex approaches have gained popularity as recently more theoretical guarantees have been provided for specific high-dimensional settings. See, e.g., Fan and Li (2001); Jain et al. (2014, 2016); Loh and Wainwright (2015). Furthermore, even though non-convex problems do not in general lead to polynomial-time computable methods, they often work well in practice. In particular, inspired by the recent work of Jain et al. (2014, 2016) who demonstrated the effectiveness of non-convex projected gradient descent approaches for high-dimensional linear regression and matrix regression, we consider applying similar techniques to high-dimensional low-rank tensor regression problems with a generalized linear model loss function.

Low-rankness in higher order tensors may occur in a variety of ways (see e.g. Koldar and Bader (2009) for examples). To accommodate these different notions of low-rankness, we develop a general framework which provides theoretical guarantees for projected gradient descent algorithms applied to tensors residing in general low-dimensional subspaces. Our framework relies on two properties ubiquitous in low-rank tensor regression problems: (i) that the parameter space is a member of a class of subspaces super-additive when indexed over a partially ordered set; and (ii) there exists an approximate projection onto each subspace satisfying a contractive property. Assuming that the coefficient tensor lies in a low-dimensional subspace Θ satisfying these properties, we establish general risk bounds for non-convex projected gradient descent based methods applied to a generalized tensor regression model with covariates with Gaussian design. By developing this general framework in terms of these properties, we only need to verify that our set satisfies these properties and also demonstrate that these properties are fundamental to providing theoretical results for projected gradient descent.

Our main theoretical result shows that the Frobenius norm scales as $n^{-1/2} w_G[\Theta \cap \mathbb{B}_F(1)]$, where n is the sample size, $\mathbb{B}_F(1) := \{A : \|A\|_F \leq 1\}$ refers to the Frobenius-norm ball with radius 1 and $w_G[\Theta \cap \mathbb{B}_F(1)]$ refers to the *localized Gaussian width* of Θ. While statistical rates in terms of Gaussian widths are already established for convex regularization approaches (see, e.g., Chandrasekaran et al., 2012; Raskutti and Yuan, 2015), this to the best of our knowledge is the first general error bound for non-convex projected gradient descent in terms of a localized Gaussian width.

©2019 Han Chen, Garvesh Raskutti, Ming Yuan.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/16-607.html.

Another major contribution we make is to provide a comparison both in terms of statistical error rate and computation to existing convex approaches to low rank tensor regression. Using our statistical error bound for non-convex projected gradient descent which is stated in terms of the localized Gaussian width of Θ, we show explicitly that our error bound for the non-convex approach is no larger (up to a constant) than those for convex regularization schemes (see, e.g., Theorem 1 of Raskutti and Yuan, 2015), and in some cases is smaller. To make this comparison more concrete, we focus on three particular examples of low-rank tensor structure: (i) sum of ranks of each slice of a tensor being small; (ii) sparsity and low-rank structure for slices; and (iii) low Tucker rank. In case (i), both approaches are applicable and achieve the same rate of convergence. For case (ii), the non-convex approach is still applicable whereas a convex regularization approach is not naturally applicable. In case (iii) again both approaches are applicable but a superior statistical performance can be achieved via the non-convex method. We supplement our theoretical comparison with a simulation comparison. Our simulation results show that our non-convex projected gradient descent based approach compares favorably to the convex regularization approach using a generic `cvx` solver in terms of both run-time and statistical performance provided optimal step-size choices in the projected gradient descent and regularization parameters in the convex regularization approach are used. Furthermore the projected gradient descent scales to much larger-scale data than generic convex solvers.

1.1. Our contributions

To summarize, we make the following three contributions in our paper:

- Firstly, we provide general error bounds for projected gradient descent applied to generalized tensor regression problems in terms of the localized Gaussian width of the constraint set Θ. In particular, we provide three novel results. Theorem 1 provides an upper bound for projected gradient descent with tensor parameters and applies to any Θ satisfying the super-additivity and contractive properties described above and explained in greater detail in Section 3. Theorem 1 substantially generalizes prior results in Jain et al. (2014, 2016) which focus on sparse vectors and low-rank matrices. Theorem 2 and 3 apply the general result in Theorem 1 to generalized linear models and Gaussian linear models respectively. Significantly, Theorems 2 and 3 show that the localized Gaussian width for the constraint set for Θ play a crucial role in the mean-squared error bound. This is the first analysis we are aware of that expresses the statistical error of PGD in terms of localized Gaussian width which allows us to deal with PGD in a more unified manner.

- Using Theorem 3, our second major contribution is to provide a comparison in terms of mean-squared error to standard convex regularization schemes studied in Raskutti and Yuan (2015). We show using the comparison of Gaussian widths in the convex and non-convex cases that unlike for vector and matrix problems where convex regularization schemes provably achieve the same statistical error bounds as non-convex approaches, the more complex structure of tensors means that our non-convex approach could yield a superior statistical error bound in some examples compared to convex regularization schemes. We also prove that our non-convex error bound is no larger than the convex regularization bound developed in Raskutti and Yuan (2015).

- Lastly, we demonstrate in Section 5 the benefits of the non-convex approach compared to existing convex regularization schemes for various low-rank tensor regression problems. We also show through simulations the benefit of using low-rank tensor regularization schemes compared to using a low-rank matrix scheme.

The remainder of the paper is organized as follows: Section 2 introduces the basics of the low-rank tensor regression models we consider and introduces the projected gradient descent algorithm. Section 3 presents the general theoretical results for non-convex projected gradient descent and specific examples are discussed in Section 4. A simulation comparison between the convex and non-convex approach is provided in Section 5 and proofs are provided in Section 7.

2. Methodology

Consider a generalized tensor regression framework where the conditional distribution of a scalar response Y given a covariate tensor $X \in \mathbb{R}^{d_1 \times d_2 \times \ldots \times d_N}$ is given by

$$p(Y|X,T) = h(Y)\exp\left\{Y\langle X,T\rangle - a(\langle X,T\rangle)\right\}, \tag{1}$$

where $a(\cdot)$ is a strictly convex and differentiable log-partition function, $h(\cdot)$ is a nuisance parameter, and $T \in \mathbb{R}^{d_1 \times d_2 \times \ldots \times d_N}$ is the parameter tensor of interest. Typical examples of $a(\cdot)$ include $a(\theta) = \frac{1}{2}\theta^2$ leading to the usual normal linear regression, $a(\theta) = \log(1+e^\theta)$ corresponding to logistic regression, and $a(\theta) = e^\theta$ which can be identified with Poisson regression where θ is a scalar. The goal is to estimate tensor T based on the training data $\{(X^{(i)}, Y^{(i)}) : 1 \le i \le n\}$. For convenience we assume $(X^{(i)}, Y^{(i)})$'s are independent copies of (X,Y). Hence the negative log-likelihood risk objective, for any $A \in \mathbb{R}^{d_1 \times d_2 \times \ldots \times d_N}$, is:

$$\mathcal{L}(A) = \frac{1}{n}\sum_{i=1}^{n}\left[a(\langle X^{(i)}, A\rangle) - Y^{(i)}\langle X^{(i)}, A\rangle - \log h(Y^{(i)})\right]. \tag{2}$$

The notation $\langle\cdot,\cdot\rangle$ will refer throughout this paper to the standard inner product taken over appropriate Euclidean spaces. Hence, for $A \in \mathbb{R}^{d_1 \times \cdots \times d_N}$ and $B \in \mathbb{R}^{d_1 \times \cdots \times d_N}$:

$$\langle A, B\rangle = \sum_{j_1=1}^{d_1}\cdots\sum_{j_N=1}^{d_N} A_{j_1,\ldots,j_N}B_{j_1,\ldots,j_N} \in \mathbb{R}.$$

Using the standard notion of inner product, for a tensor A, $\|A\|_{\mathrm{F}} = \langle A, A\rangle^{1/2}$. And the empirical norm $\|\cdot\|_n$ for a tensor $A \in \mathbb{R}^{d_1 \times \cdots \times d_N}$ is define as:

$$\|A\|_n^2 := \frac{1}{n}\sum_{i=1}^{n}\langle A, X^{(i)}\rangle^2.$$

Also, for any linear subspace $\mathcal{A} \subset \mathbb{R}^{d_1 \times d_2 \times \ldots \times d_N}$, $A_{\mathcal{A}}$ denotes the projection of a tensor A onto $\mathcal{A}$. More precisely

$$A_{\mathcal{A}} := \arg\min_{M \in \mathcal{A}} \|A - M\|_{\mathrm{F}}.$$

2.1. Background on tensor algebra

One of the major challenges associated with low-rank tensors is that the notion of higher-order tensor decomposition and rank is ambiguous. See, e.g., Koldar and Bader (2009) for a review. There are two standard decompositions, the so-called canonical polyadic (CP) decomposition and the Tucker decomposition. In addition to that, there are various notions of tensor low-rankness considered in the application. In order to simplify notation, we focus our discussion on third-order tensors ($N = 3$) but point out that many of the notions generalize to $N > 3$. Our general theorem in Section 3 works for general $N \geq 3$.

The CP decomposition of a third-order tensor is defined as the smallest number r of rank-one tensors needed to represent a tensor $A \in \mathbb{R}^{d_1 \times d_2 \times d_3}$:

$$A = \sum_{k=1}^{r} u_{k,1} \otimes u_{k,2} \otimes u_{k,3} \tag{3}$$

where $u_{k,m} \in \mathbb{R}^{d_m}$, for $1 \leq k \leq r$ and $1 \leq m \leq 3$.

A second popular decomposition is the so-called Tucker decomposition. The Tucker decomposition of a tensor $A \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ is of the form:

$$A_{j_1 j_2 j_3} = \sum_{k_1=1}^{r_1} \sum_{k_2=1}^{r_2} \sum_{k_3=1}^{r_3} S_{k_1 k_2 k_3} U_{j_1 k_1,1} U_{j_2 k_2,2} U_{j_3 k_3,3}$$

so that $U_m \in \mathbb{R}^{d_m \times r_m}$ for $1 \leq m \leq 3$ are factors matrices (which are usually orthogonal) and $S \in \mathbb{R}^{r_1 \times r_2 \times r_3}$ is the so-called core tensor (see, e.g., Koldar and Bader, 2009). The vector (r_1, r_2, r_3) is referred to as the Tucker ranks of A. It is not hard to see that if (3) holds, then the Tucker ranks (r_1, r_2, r_3) can be equivalently interpreted as the dimensionality of the linear spaces spanned by $\{u_{k,1} : 1 \leq k \leq r\}$, $\{u_{k,2} : 1 \leq k \leq r\}$, and $\{u_{k,3} : 1 \leq k \leq r\}$ respectively.

A convenient way to represent low Tucker ranks of a tensor is through *matricization*. Denote by $\mathcal{M}_1(\cdot)$ the mode-1 matricization of a tensor, that is $\mathcal{M}_1(A)$ is the $d_1 \times (d_2 d_3)$ matrix whose column vectors are the mode-1 fibers of $A \in \mathbb{R}^{d_1 \times d_2 \times d_3}$. A fiber is defined by fixing every index but one, which is the first index in the case of mode-1 fiber. Here A has $d_2 d_3$ mode-1 fibers. $\mathcal{M}_2(\cdot)$, $\mathcal{M}_3(\cdot)$ are defined in the same fashion. By defining

$$\text{rank}(\mathcal{M}_m(A)) = r_m(A),$$

it follows that $(r_1(A), r_2(A), r_3(A))$ represent the Tucker ranks of A. For later discussion, define $\mathcal{M}_i^{-1}(\cdot)$ to be the inverse of mode-i matricization, so

$$\mathcal{M}_1^{-1} : \mathbb{R}^{d_1 \times (d_2 \cdot d_3)} \to \mathbb{R}^{d_1 \times d_2 \times d_3},$$
$$\mathcal{M}_2^{-1} : \mathbb{R}^{d_2 \times (d_1 \cdot d_3)} \to \mathbb{R}^{d_1 \times d_2 \times d_3},$$
$$\mathcal{M}_3^{-1} : \mathbb{R}^{d_3 \times (d_1 \cdot d_2)} \to \mathbb{R}^{d_1 \times d_2 \times d_3},$$

such that $\mathcal{M}_i^{-1}(\mathcal{M}_i(A)) = A$. Also, a tensor can be vectorized via collapsing all its dimensions sequentially, i.e. $\text{vec}(A) \in \mathbb{R}^{d_1 \cdot d_2 \cdot d_3}$ for $A \in \mathbb{R}^{d_1 \times d_2 \times d_3}$, and A's $(j_1, j_2, j_3)^{th}$ element is placed in the position of $\sum_{i=1}^{3}[(j_i - 1)\Pi_{l=i+1}^{3} d_l] + 1$ in $\text{vec}(A)$.

Further, we define *slices* of a tensor as follows. For an order-3 tensor A, the $(1, 2)$ slices of A are the collection of d_3 matrices of $d_1 \times d_2$

$$\{A_{\cdot\cdot j_3} := (A_{j_1 j_2 j_3})_{1 \leq j_1 \leq d_1, 1 \leq j_2 \leq d_2} : 1 \leq j_3 \leq d_3\}.$$

2.2. Low-dimensional structural assumptions

As mentioned earlier, there is not a unique way of defining tensor and there are multiple ways to impose low-rankness/low-dimensionality on a tensor space. Given the ambiguity of tensor low-rankness, one of the goals of the paper is to develop a general framework that applies to many different notions of low-rankness. To be more concrete, we focus on three specific examples of low-rank structure. These three examples fall in the general framework of our analysis of the PGD algorithm applied to low-dimensional tensor regression. Among the three, the first two are closely related to viewing the tensor as slices of matrix or lower order tensor (e.g. 4th order tensor has 'slices' of 3rd order tensor), and hence measuring hybrid of group sparsity and lower order low-rankness. The third is the maximum of the standard Tucker ranks. For simplicity, we focus on the case when $N = 3$ and then discuss potential generalization to higher order.

Firstly we place low-rank structure on the matrix slices. In particular first define:

$$\Theta_1(r) = \left\{ A \in \mathbb{R}^{d_1 \times d_2 \times d_3} : \sum_{j_3=1}^{d_3} \text{rank}(A_{\cdot\cdot j_3}) \leq r \right\}.$$

We can view $\Theta_1(r)$ as a specific rank-r or less subset of 3rd order tensor space with the rank defined to be the sum of the rank of the matrix slices. To connect to matrices, an alternative way to parameterize $\Theta_1(r)$ is as the set of $\mathbb{R}^{d_1 . d_3 \times d_2 . d_3}$ block diagonal matrices, where each block corresponds to the matrix slice $A_{\cdot\cdot j_3} \in \mathbb{R}^{d_1 \times d_2}$ and $1 \leq j_3 \leq d_3$. The rank constraint in $\Theta_1(r)$, corresponds exactly to placing a rank constraint on the corresponding $\mathbb{R}^{d_1 . d_3 \times d_2 . d_3}$ matrix.

Secondly we can impose a related notion where we bound the maximum of the rank of each slice and sparsity along the matrix slices.

$$\Theta_2(r, s) = \left\{ A \in \mathbb{R}^{d_1 \times d_2 \times d_3} : \max_{j_3} \text{rank}(A_{\cdot\cdot j_3}) \leq r, \sum_{j_3=1}^{d_3} \mathbb{I}(A_{\cdot\cdot j_3} \neq 0) \leq s \right\}.$$

We can view $\Theta_2(r, s)$ as a specific rank-(r,s) or less subset of the third-order tensor. One natural example where imposing this combination of low-rankness and sparsity for is vector auto-regressive (VAR) models (see e.g. Basu and Michailidis (2015)). For example, if we have an M-variate time series and consider a VAR(p) model, $d_1 = d_2 = M$ and $d_3 = p$. We want each auto-regressive matrix slice $A_{\cdot\cdot j_3}$ to have low-rank and the total number of lags involved in the problem to be sparse (e.g. to account for immediate effects and seasonal effects).

Finally, we impose the assumption that Tucker ranks are upper bounded:

$$\Theta_3(r_1, r_2, r_3) = \left\{ A \in \mathbb{R}^{d_1 \times d_2 \times d_3} : r_i(A) \leq r_i \text{ for } i = 1, 2, 3 \right\}.$$

Note that all these low-dimensional structural assumptions $\Theta_1(r)$, $\Theta_2(r,s)$ and $\Theta_3(r_1,r_2,r_3)$ are non-convex sets. Note that $\Theta_2(r,s)$ and $\Theta_3(r_1,r_2,r_3)$ have two parameters (r,s) and three parameters (r_1,r_2,r_3) respectively and our framework deals with this. In the next subsection we introduce a general projected gradient descent (PGD) algorithm for minimizing the generalized linear model objective (2) subject to the parameter tensor A belonging to a potentially non-convex constraint set Θ. Also note that it is straightforward to extend $\Theta_3(r_1,r_2,r_3)$ to $N>3$.

2.3. Projected Gradient Descent (PGD)

In this section we introduce the non-convex projected gradient descent (PGD) approaches developed in Jain et al. (2014, 2016) adapted to a general tensor space Θ. The problem we are interested in is minimizing the generalized linear model objective (2) subject to A belonging to a potentially non-convex set. The PGD algorithm for minimizing a general loss function $f(A)$ subject to the constraint $A\in\Theta$ is as follows:

Algorithm 1 Projected Gradient Descent

1: **Input :** data $\mathbf{Y},\mathbf{X}$, parameter space Θ, iterations K, step size η
2: **Initialize :** $k=0$, $\widehat{T}_0\in\Theta$
3: **for** $k=1,2,\ldots,K$ **do**
4: $\quad g_k=\widehat{T}_k-\eta\nabla f(\widehat{T}_k)$ (gradient step)
5: $\quad \widehat{T}_{k+1}=P_\Theta(g_k)$ or $\widehat{T}_{k+1}=\widehat{P}_\Theta(g_k)$ ((approximate) projection step)
6: **end for**
7: **Output :** $\widehat{T}_K$

The notation $\widehat{P}_\Theta(\cdot)$ refers to an approximate projection on to Θ if an exact projection is not implementable. The PGD algorithm has been widely used for both convex and non-convex objectives and constraint sets. In our setting, we choose the negative log-likelihood for the generalized linear model as the functions $f(A)$ to minimize.

An important question associated with projected gradients is whether the projection on to Θ is implementable. The projections we consider for $\Theta_1(r)$, $\Theta_2(r,s)$ and $\Theta_3(r_1,r_2,r_3)$ can all be implemented approximately as combinations of projections on to matrix or vector subspaces defined in Jain et al. (2014, 2016). By using sparsity and low-rankness projections in the vector and matrix respectively, the projections in Θ_1, Θ_2 and Θ_3 are implementable.

In particular, for a vector $v\in\mathbb{R}^d$, we define the projection operator $\tilde{P}_s(v)$ as the projection on to the set of s-sparse vectors by selecting the s largest elements of v in ℓ_2-norm. That is:

$$\tilde{P}_s(v):=\underset{\|z\|_{\ell_0}\le s}{\arg\min}\|z-v\|_{\ell_2}.$$

For a matrix $M\in\mathbb{R}^{d_1\times d_2}$, let $\bar{P}_r(M)$ denote the rank-r projection:

$$\bar{P}_r(M):=\underset{\mathrm{rank}(Z)\le r}{\arg\min}\|Z-M\|_{\mathrm{F}}.$$

As mentioned in Jain et al. (2014, 2016), both projections are computable. $\tilde{P}_s(v)$ is the thresholding operator which selects the s largest elements of v in ℓ_2-norm and $\bar{P}_r(M)$ can

be selecting the top r singular vectors of M. For the remainder of this paper we use both of these projection operators for vectors and matrices respectively.

3. Main Results

In this section we present our general theoretical results where we provide a statistical guarantee for the PGD algorithm applied to a low-dimensional space Θ.

3.1. Properties for Θ and its projection

To ensure the PGD algorithm converges for a given subspace Θ, we view it as a member of a collection of subspaces $\{\Theta(t):t\in\Xi\}$ for some $\Xi\subset\mathbb{Z}_+^k$ and require some general properties for this collection. The index t typically represents a sparsity and/or low-rank index and may be multi-dimensional. For example, $\Theta_1(r)$ is indexed by rank r where

$$\Xi=\{0,\ldots,d_3\cdot\min\{d_1,d_2\}\}.$$

Similarly, $\Theta_2(r,s)$ is indexed by $t=(r,s)$ so that

$$\Xi=\{(0,0),\ldots,(\min\{d_1,d_2\},d_3)\},$$

and $\Theta_3(r_1,r_2,r_3)$ is indexed by rank (r_1,r_2,r_3) so that

$$\Xi=\{(0,0,0),\ldots,(\min\{d_1,d_2d_3\},\min\{d_2,d_1d_3\},\min\{d_3,d_1d_2\})\}.$$

Note that the Ξ is partially ordered where $a\ge(\le,<,>)b$ for two vectors a and b of conformable dimension means the inequality holds in an element-wise fashion, i.e. the inequality holds for each element of a and its counterparty in b.

Definition 1. A set $\{\Theta(t):t\in\Xi\}$ is a *superadditive and partially ordered collection of symmetric cones* if

(1) each member $\Theta(t)$ is a *symmetric cone* in that if $z\in\Theta(t)$, then $cz\in\Theta(t)$ for any $c\in\mathbb{R}$;

(2) the set is *partially ordered* in that for any $t_1\le t_2$, $\Theta(t_1)\subset\Theta(t_2)$;

(3) the set is *superadditive* in that $\Theta(t_1)+\Theta(t_2)\subset\Theta(t_1+t_2)$.

The first two properties basically state that we have a set of symmetric cones in the tensor space with a partial ordering indexed by t. The last property requires that the collection of subspaces be superadditive in that the Minkowski sum of any two subspaces is contained in the subspace of dimension that is the sum of the two lower dimensions. All three properties are essential for deriving theoretical guarantees for the PGD algorithm. By relying on these properties alone, we obtain a general result that provides a unified way of dealing with $\Theta_1(r),\Theta_2(r,s)$ and $\Theta_3(r_1,r_2,r_3)$ as well as many other collections of subspaces.

Furthermore, we introduce the following property of contractive projection, for P_Θ or $\widehat{P}_\Theta$ in Algorithm 1, that is essential for the theoretical performance of the PGD algorithm. Again, we shall view these operators as members of a collection of operators $Q_{\Theta(t)}$:

$\cup_t \Theta(t) \mapsto \Theta(t)$. The contractive projection property says that, when $Q_{\Theta(t)} : \cup_t \Theta(t) \mapsto \Theta(t)$ are viewed as projections, projection onto a larger "dimension" incurs less approximation error *per dimension* compared to projection onto a smaller dimension, up to a constant factor. This property is adopted from earlier study on vector/matrix case. (see, e.g., Jain et al., 2014, 2016). The difference here is that in the tensor case, the constant factor δ is no longer one as in vector/matrix case. And later theorems will show that any finite δ can guarantee linear convergence with error bound expressed in Gaussian width for PGD in tensor case. This means that certain low-cost approximate low-rank projections in the tensor case are sufficient for PGD and later we will see Tucker rank is the case.

Definition 2. We say that a set $\{\Theta(t) : t \geq 0\}$ and corresponding operators $Q_{\Theta(t)} : \cup_t \Theta(t) \mapsto \Theta(t)$ satisfy the *contractive projection property* for some $\delta > 0$, denoted by CPP(δ), if for any $t_1 < t_2 < t_0$, $Y \in \Theta(t_1)$, and $Z \in \Theta(t_0)$:

$$\|Q_{\Theta(t_2)}(Z) - Z\|_{\mathrm{F}} \leq \delta \left\| \frac{t_0 - t_2}{t_0 - t_1} \right\|_{\ell_\infty}^{1/2} \cdot \|Y - Z\|_{\mathrm{F}}.$$

Here, when $\Theta(t)$ is indexed by multi-dimensional t, the division $\frac{t_0 - t_2}{t_0 - t_1}$ refers to a vector with the jth element to be $\frac{(t_0)_j - (t_2)_j}{(t_0)_j - (t_1)_j}$.

It is clear that $\Theta_1(r)$ is isomorphic to rank-r block diagonal matrices with diagonal blocks $A_{\cdot\cdot 1}$, $A_{\cdot\cdot 2}$,..., $A_{\cdot\cdot d_3}$ so that $\{\Theta_1(r)\}$ satisfies Definition 1. It is also easy to verify that $\{\Theta_1(r)\}$ and its projections $\{P_{\Theta_1(r)}\}$ which applies $\bar{P}_r(.)$ to the $d_1.d_3 \times d_2.d_3$ block-diagonal matrix described earlier obeys CPP(1). Later, we will see in Lemmas 2 and 3 that these two properties are also satisfied by $\{\Theta_2(r, s)\}$ and $\{\Theta_3(r_1, r_2, r_3)\}$, and their appropriate (approximate) projections.

3.2. Restricted strong convexity

Now we state some general requirements on the loss function, namely the restricted strong convexity and smoothness conditions (RSCS), that is another essential part for the guarantee of PGD performance (see, e.g., Jain et al., 2014, 2016). Recall that for $f(A)$, a function of tensor A, we can abuse the notation to view $f(A)$ as a function of vectorized tensor, i.e. $f(A) = f(\mathrm{vec}(A))$and use $\nabla^2 f(A)$ to denote the Hessian of function f on vectorized tensor. Please refer to Section 2.1 for the formal definition of vectorization of a tensor.

Definition 3. We say that a function f satisfies *restricted strong convexity and smoothness conditions* $RSCS(\Theta, C_l, C_U)$ for a set Θ, and $0 < C_l < C_u < \infty$ if for any $A \in \Theta$, $\nabla^2 f(A)$ is positive semidefinite such that for any $B \in \Theta$

$$C_l \cdot \|B\|_{\mathrm{F}} \leq \|\nabla^2 f(A) \cdot \mathrm{vec}(B)\|_{\ell_2} \leq C_u \cdot \|B\|_{\mathrm{F}},$$

for some constants $C_l < C_u$. Note that $RSCS(\Theta, C_l, C_U)$ reduces to restricted strong convexity and restricted smoothness assumptios for vectors and matrices (see e.g. Jain et al. (2014, 2016)) when A and B are vectors and matrices. We first state the following Theorem about the PGD performance under general loss function which is a tensor version of the results in Jain et al. (2014, 2016).

Theorem 1 ***(PGD Error Bound for General Loss Function)*** *Suppose that $\{\Theta(t) : t \geq 0\}$ is a superadditive and partially ordered collection of symmetric cones, together with operators $\{P_{\Theta(t)} : t \geq 0\}$ which obey CPP(δ) for some constant $\delta > 0$, and f satisfies $RSCS(\Theta(t_0), C_l, C_u)$ for some constants C_l and C_u. Let $\widehat{T}_K$ be the output from the Kth iteration of applying PGD algorithm with step size $\eta = 1/C_u$, and projection $P_{\Theta(t_1)}$ where*

$$t_1 = \left\lceil \frac{4\delta^2 C_u^2 C_l^{-2}}{1 + 4\delta^2 C_u^2 C_l^{-2}} \cdot t_0 \right\rceil.$$

Then

$$\sup_{T \in \Theta(t_0 - t_1)} \|\widehat{T}_K - T\|_{\mathrm{F}} \leq 4\eta C_u C_l^{-1} \cdot \sup_{A \in \Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)} \langle \nabla f(T), A \rangle + \epsilon,$$

for any

$$K \geq 2C_u C_l^{-1} \log\left(\frac{\|T\|_{\mathrm{F}}}{\epsilon} \right).$$

Basically, Theorem 1 guarantees that the PGD applied to general loss minimization problem with general low-rank constraint enjoys linear convergence rate with statistical error bounded by the restricted norm of the gradient of loss function evaluation at the true parameter. Note that we are taking a supremum over the set $\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)$ which relates to the local Gaussian width which we define shortly. More insight arises when we specialize f to the generalized linear model.

3.3. Generalized linear models

To use Theorem 1 for a specific f and collection of $\{\Theta(t) : t \geq 0\}$, we need to verify the conditions on $\{\Theta(t) : t \geq 0\}$, $\{P_{\Theta(t)} : t \geq 0\}$ and f satisfying $RSCS(\Theta, C_l, C_U)$, and choose the step-size in the PGD accordingly.

First we turn our attention to the covariate tensor $(X^{(i)})_{i=1}^n$ where $X^{(i)} \in \mathbb{R}^{d_1 \times d_2 \times \ldots \times d_N}$ and how it relates to the $RSCS(\Theta, C_l, C_U)$. With slight abuse of notation, write

$$\mathrm{vec}(X^{(i)}) \in \mathbb{R}^{d_1 d_2 \cdots d_N}$$

for $1 \leq i \leq n$ which is the vectorization of each tensor covariate $X^{(i)}$. For convenience let $D_N = d_1 d_2 \cdots d_N$. Further as mentioned for technical convenience we assume a Gaussian design of independent sample tensors $X^{(i)}$ s.t.

$$\mathrm{vec}(X^{(i)}) \sim \mathcal{N}(0, \Sigma) \text{ where } \Sigma \in \mathbb{R}^{D_N \times D_N}. \tag{4}$$

With more technical work our results may be extended beyond random Gaussian designs. We shall assume that Σ has bounded eigenvalues. Let $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ represent the smallest and largest eigenvalues of a matrix, respectively. In what follows, we shall assume that

$$c_\ell^2 \leq \lambda_{\min}(\Sigma) \leq \lambda_{\max}(\Sigma) \leq c_u^2, \tag{5}$$

for some constants $0 < c_\ell \leq c_u < \infty$. For our analysis of the non-convex projected gradient descent algorithm, we define the condition number $\kappa = c_u / c_l$.

A quantity that emerges from our analysis is the *Gaussian width* (see, e.g., Gordon, 1988) of a set $S \subset \mathbb{R}^{d_1 \times d_2 \times \ldots \times d_N}$ which is defined to be:

$$w_G(S) := \mathbb{E}\left(\sup_{A \in S} \langle A, G \rangle\right),$$

where $G \in \mathbb{R}^{d_1 \times d_2 \times \ldots \times d_N}$ is a tensor whose entries are independent $\mathcal{N}(0,1)$ random variables. The Gaussian width is a standard notion of size or complexity of a subset of tensors S.

In addition to the positive semi-definiteness of the Hessian in the GLM, the following Lemma extends a restricted upper and lower eigenvalue condition to the sample version of Σ and hence guarantees the RSCS for the GLM with Gaussian covariates with quite general covariance structure.

Lemma 1 *Assume that* (4) *and* (5) *hold. For any $\tau > 1$, there exist constants $c_1, c_2, c_3 > 0$ such that if $n \geq c_1 w_G^2[\Theta \cap \mathbb{B}_{\mathrm{F}}(1)]$, then with probability at least $1 - c_2 \exp(-c_3 w_G^2[\Theta \cap \mathbb{B}_{\mathrm{F}}(1)])$,*

$$\left(\tau^{-1} c_l\right)^2 \|A\|_{\mathrm{F}}^2 \leq \frac{1}{n} \sum_{i=1}^{n} \langle X^{(i)}, A \rangle^2 \leq (\tau c_u)^2 \|A\|_{\mathrm{F}}^2, \qquad \forall A \in \Theta.$$

Lemma 1 is a direct consequence of Theorem 6 in Banerjee et al. (2015). Using these definitions, we are in a position to state the main result for generalized linear models.

Theorem 2 ***(PGD Error Bound for Generalized Linear Model)*** *Suppose that $\{\Theta(t) : t \geq 0\}$ is a superadditive and partially ordered collection of symmetric cones, and together with operators $\{P_{\Theta(t)} : t \geq 0\}$ which obey CPP(δ) for some constant $\delta > 0$. Assume that $\{(X^{(i)}, Y^{(i)}) : i = 1, \ldots, n\}$ follow the generalized linear model* (1) *and $X^{(i)}$'s satisfy* (4) *and* (5)*, $\mathbb{E}|Y^{(i)}|^q \leq M_Y$ for some $q > 2$ and $M_Y > 0$, $1/\tau_0^2 \leq Var(Y^{(i)}) \leq \tau_0^2$ for $i = 1, \ldots, n$ and some $\tau_0 > 0$, and $n > c_1 w_G^2[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)]$ for some t_0 and $c_1 > 0$. Let $\widehat{T}_K$ be the output from the K^{th} iteration of applying PGD algorithm to* (2) *with step size $\eta = (\tau c_u)^{-2}$ and projection $P_{\Theta(t_1)}$ where*

$$t_1 = \left\lceil \frac{4\delta^2 \tau^8 \kappa^4}{1 + 4\delta^2 \tau^8 \kappa^4} \cdot t_0 \right\rceil,$$

for any given $\tau > \tau_0$. Then there exist constants $c_2, c_3, c_4, c_5 > 0$ such that

$$\sup_{T \in \Theta(t_0 - t_1)} \|\widehat{T}_K - T\|_{\mathrm{F}} \leq \frac{c_5 \eta \tau^4 \kappa^2 c_u M_Y^{1/q}}{\sqrt{n}} \cdot w_G[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)] + \epsilon,$$

with probability at least

$$1 - K c_2 \exp\left\{-c_3 w_G^2[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)]\right\} - K c_4 n^{-(q/2-1)} \log^q n,$$

for any

$$K \geq 2\tau^4 \kappa^2 \log\left(\frac{\|T\|_{\mathrm{F}}}{\epsilon}\right).$$

Notice that the statistical error we have is related to the Gaussian width of the intersection of a unit Frobenius ball and an (often non-convex) subset of low-dimensional structure $w_G[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)]$. The intersection of $\Theta(t_0)$ with $\mathbb{B}_{\mathrm{F}}(1)$ means we are *localizing* the Gaussian width to a unit Frobenius norm ball around T. *Localization* of the Gaussian width means a sharper statsitical error bound can be proven and the benefits of localization in empirical risk minimization have been previously discussed in Bartlett et al. (2005). Later we will see how the main result leads to sample complexity bounds applied to $\Theta_2(r,s)$ and $\Theta_3(r_1, r_2, r_3)$. To the best of our knowledge this is the first general result that provides statistical guarantees in terms of the local Gaussian width of $\Theta(t_0)$ for the projected gradient descent algorithm. Expressing the error bound in terms of the Gaussian width allows an easy comparison to already established error bounds for convex regularization schemes which we discuss in Section 3.4.

The moment conditions on the response in Theorem 2 are in place to ensure that the restricted strong convexity and restricted smoothness conditions are satisfied for a non-quadratic loss. When specialized to normal linear regression, these conditions could be further removed.

3.4. Gaussian model and comparison to convex regularization

Consider the Gaussian linear regression setting which corresponds to the GLM in Equation (1) with $a(\theta) = \frac{\theta^2}{2}$. In particular

$$Y^{(i)} = \langle X^{(i)}, T \rangle + \zeta^{(i)}, \tag{6}$$

where $\zeta^{(i)}$'s are independent $\mathcal{N}(0, \sigma^2)$ random variables. Furthermore, substituting $a(\theta) = \frac{\theta^2}{2}$ into the GLM objective (2), we have the least-squares objective:

$$f(A) = \frac{1}{2n} \sum_{i=1}^{n} (Y^{(i)} - \langle X^{(i)}, A \rangle)^2. \tag{7}$$

Now we state our main result for the normal linear regression.

Theorem 3 ***(PGD Error Bound for Normal Linear Regression)*** *Suppose that $\{\Theta(t) : t \geq 0\}$ is a superadditive and partially ordered collection of symmetric cones, and together with operators $\{P_{\Theta(t)} : t \geq 0\}$ which obey CPP(δ) for some constant $\delta > 0$. Assume that $\{(X^{(i)}, Y^{(i)}) : i = 1, \ldots, n\}$ follow the Gaussian linear model* (6) *where $n > c_1 w_G^2[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)]$ for some t_0 and $c_1 > 0$. Let $\widehat{T}_K$ be the output from the K^{th} iteration of applying PGD algorithm to* (2) *with step size $\eta = (\tau c_u)^{-2}$ and projection $P_{\Theta(t_1)}$ where*

$$t_1 = \left\lceil \frac{4\delta^2 \tau^8 \kappa^4}{1 + 4\delta^2 \tau^8 \kappa^4} \cdot t_0 \right\rceil,$$

for any given $\tau > 1$. Then there exist constants $c_2, c_3 > 0$ such that

$$\sup_{T \in \Theta(t_0 - t_1)} \|\widehat{T}_K - T\|_{\mathrm{F}} \leq \frac{8\eta \tau^4 \kappa^2 c_u \sigma}{\sqrt{n}} w_G[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)] + \epsilon,$$

with probability at least

$$1 - Kc_2 \exp\left\{-c_3 w_G^2[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)]\right\},$$

for any

$$K \geq 2\tau^4\kappa^2 \log\left(\frac{\|T\|_{\mathrm{F}}}{\epsilon}\right).$$

One of the contributions of this paper outlined in the introduction is to compare the non-convex PGD approach in tensor regression to the existing convex regularization approach analyzed in Raskutti and Yuan (2015) applied to the Gaussian linear model (6). In this section we first summarize the general result from Raskutti and Yuan (2015) and then provide a comparison to the error bound for the non-convex PGD approach. In particular, the following estimator for T is considered in Raskutti and Yuan (2015):

$$\widehat{T} \in \underset{A \in \mathbb{R}^{d_1 \times \cdots \times d_N}}{\arg\min} \left\{ \frac{1}{2n} \sum_{i=1}^{n} \|Y^{(i)} - \langle A, X^{(i)} \rangle\|_{\mathrm{F}}^2 + \lambda \mathcal{R}(A) \right\}, \tag{8}$$

where the convex regularizer $\mathcal{R}(\cdot)$ is a norm on $\mathbb{R}^{d_1 \times \cdots \times d_N}$, and $\lambda > 0$ is a tuning parameter. The *convex conjugate* for $\mathcal{R}$ (see e.g. Rockafellar (1970) for details) is given by:

$$\mathcal{R}^*(B) := \sup_{A \in \mathbb{B}_{\mathcal{R}}(1)} \langle A, B \rangle.$$

For example if $\mathcal{R}(A) = \|A\|_*$, then $\mathcal{R}^*(B) = \|B\|_s$. Following Negahban et al. (2012), for a subspace Θ of $\mathbb{R}^{d_1 \times \cdots \times d_N}$, define its compatibility constant $s(\Theta)$ as

$$s(\Theta) := \sup_{A \in \Theta / \{0\}} \frac{\mathcal{R}^2(A)}{\|A\|_{\mathrm{F}}^2},$$

which can be interpreted as a notion of low-dimensionality of Θ.

Raskutti and Yuan (2015) show that if $\widehat{T}$ is defined by (8) and the regularizer $\mathcal{R}(\cdot)$ is *decomposable* with respect to Θ, then if

$$\lambda \geq 2 w_G(\mathbb{B}_{\mathcal{R}}(1)), \tag{9}$$

where recall that $w_G(\mathbb{B}_{\mathcal{R}}(1)) = \mathbb{E}\big(\sup_{A \in \mathbb{B}_{\mathcal{R}}(1)} \langle A, G \rangle\big)$. Then according to Theorem 1 in Raskutti and Yuan (2015),

$$\max\left\{ \|\widehat{T} - T\|_n, \|\widehat{T} - T\|_{\mathrm{F}} \right\} \lesssim \frac{\sqrt{s(\Theta)}\lambda}{\sqrt{n}}. \tag{10}$$

with probability at least $1 - \exp(-cn)$ for some constant $c > 0$. In particular setting $\lambda = 2w_G(\mathbb{B}_{\mathcal{R}}(1))$,

$$\max\left\{ \|\widehat{T} - T\|_n, \|\widehat{T} - T\|_{\mathrm{F}} \right\} \lesssim \frac{\sqrt{s(\Theta)} w_G(\mathbb{B}_{\mathcal{R}}(1))}{\sqrt{n}}.$$

The error bound boils down to bounding two quantities, $s(\Theta)$ and $w_G(\mathbb{B}_{\mathcal{R}}(1))$, noting that for comparison pursposes the subpace Θ in the convex case refers to $\Theta(t_0)$ in the non-convex case. In the next section we provide a comparison between the error bound for the non-convex PGD approach and the convex regularization approach. To be clear, Raskutti and Yuan (2015) consider multi-response models where the response $Y^{(i)}$ can be a tensor which are not considered in this paper.

The error bound for the convex regularization scheme scales as $\sqrt{s(\Theta(t_0))} w_G[\mathbb{B}_{\mathcal{R}}(1)]/\sqrt{n}$ while we recall that the error bound we prove in this paper for the non-convex PGD approach scales as $w_G[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)]/\sqrt{n}$. Hence how the Frobenius error for the non-convex and convex approach scales depends on which of the quantities $\sqrt{s(\Theta(t_0))} w_G[\mathbb{B}_{\mathcal{R}}(1)]/\sqrt{n}$ or $w_G[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)]/\sqrt{n}$ is larger. It follows easily that $w_G[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)] \leq \sqrt{s(\Theta(t_0))} w_G[\mathbb{B}_{\mathcal{R}}(1)]$ since

$$\begin{aligned} w_G[\Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)] &= \mathbb{E}\big[\sup_{A \in \Theta(t_0), \|A\|_{\mathrm{F}} \leq 1} \langle A, G \rangle \big] \\ &\leq \mathbb{E}\big[\sup_{\mathcal{R}(A) \leq \sqrt{s(\Theta(t_0))}} \langle A, G \rangle \big] \\ &= \sqrt{s(\Theta(t_0))}\, \mathbb{E}\big[\sup_{\mathcal{R}(A) \leq 1} \langle A, G \rangle \big] = \sqrt{s(\Theta(t_0))} w_G[\mathbb{B}_{\mathcal{R}}(1)]. \end{aligned}$$

The first inequality follows from the subspace compatibility constant since for all $A \in \Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)$, $\mathcal{R}(A) \leq \sqrt{s(\Theta(t_0))} \|A\|_{\mathrm{F}} \leq \sqrt{s(\Theta(t_0))}$ and the final equality follows since $\mathcal{R}(\cdot)$ is a convex function. Therefore the non-convex error bound is always no larger than the convex error bound and the important question is whether there is a gap between the convex and non-convex bounds which implies a superior bound in the non-convex case. For examples involving sparse vectors and low-rank matrices as studied in e.g., Buhlmann and van de Geer (2011); Jain et al. (2014, 2016), these two quantities end up being identical up to a constant. On the other hand for tensors, as we see in this paper for $\Theta_3(r_1, r_2, r_3)$, the Gaussian width using the non-convex approach is smaller which presents an additional benefit for the non-convex projection approach.

In terms of implementation, the regularizer $\mathcal{R}(\cdot)$ needs to be defined in the convex approach and the important question is whether the convex regularizer is implementable for the low-dimensional structure of interest. For the non-convex approach, the important implementation issue is whether exact or approximate projection that satisfies the contractive projection property is implementable. These implementation issues have been resolved in the vector and matrix cases (see, e.g., Jain et al., 2014, 2016). In Section 5 in this paper, we focus on whether they apply in the low-rank tensor case under the low-dimensional structure Θ_1, Θ_2 and Θ_3.

4. Specific low rank structure

In this section, we apply Theorem 3 (and by extension Theorem 2) to $\Theta_1(r)$, $\Theta_2(r, s)$ and $\Theta_3(r_1, r_2, r_3)$ and compare our theoretical result to the theoretical bound achieved by the convex regularization approach. Recall that $\Theta_1(r)$ is isomorphic to rank-r block diagonal matrices with diagonal blocks $A_{\cdot\cdot 1}$, $A_{\cdot\cdot 2}, \ldots, A_{\cdot\cdot d_3}$ so that its treatment is identical to the case of low rank matrix estimation. See Jain et al. (2016) for further discussions. Hence we

will focus on $\Theta_2(r,s)$ and $\Theta_3(r_1,r_2,r_3)$. To prove error bounds using Theorem 3 we find an exact or approximate projection $P_{\Theta(t)}$, prove the contractive projection property and then find an error bound on the Gaussian width $w_G[\Theta(t)\cap\mathbb{B}_{\mathrm{F}}(1)]$.

4.1. Low-rank structure for matrix slices

Recall that

$$\Theta_2(r,s)=\left\{A\in\mathbb{R}^{d_1\times d_2\times d_3}:\max_{j_3}\operatorname{rank}(A_{\cdot\cdot j_3})\le r,\sum_{j_3=1}^{d_3}\mathbb{I}(A_{\cdot\cdot j_3}\neq 0)\le s\right\}.$$

For the projection, we define the two-step projection $P_{\Theta_2(r,s)}$:

(1) for each matrix slice $A_{\cdot\cdot j_3}$ where $1\le j_3\le d_3$, let $\tilde{A}_{\cdot\cdot j_3}:=\bar{P}_r(A_{\cdot\cdot j_3})$ be the best rank r approximation of $A_{\cdot\cdot j_3}$;

(2) to impose sparsity, retain s out of d_3 slices with the largest magnitude $\|\tilde{A}_{\cdot\cdot j_3}\|_{\mathrm{F}}$, and zero out all other slices.

As discussed earlier both steps are easily computable using thresholding and SVD operators as discussed in Jain et al. (2014, 2016).The following lemma proves that the contractive property of projection onto $\Theta_2(r,s)$ holds for our $P_{\Theta_2(r,s)}$.

Lemma 2 *Let the projection operator $P_{\Theta_2(r,s)}$ be defined above. Suppose $Z\in\Theta_2(r_0,s_0)$, and $r_1<r_2<r_0, s_1<s_2<s_0$. Then for any $Y\in\Theta_2(r_1,s_1)$, we have*

$$\|P_{\Theta_2(r_2,s_2)}(Z)-Z\|_{\mathrm{F}}\le(\alpha+\beta+\alpha\beta)\cdot\|Y-Z\|_{\mathrm{F}}.$$

where $\alpha=\sqrt{(s_0-s_2)/(s_0-s_1)}$, $\beta=\sqrt{(r_0-r_2)/(r_0-r_1)}$.

Consequently we have the following Theorem:

Theorem 4 *Let $\{X^{(i)},Y^{(i)}\}_{i=1}^n$ follow a Gaussian linear model as defined by (6) with $T\in\Theta_2(r,s)$ and*

$$n\ge c_1\cdot sr(d_1+d_2+\log d_3)$$

for some constant $c_1>0$. Then, applying the PGD algorithm with step size $\eta=(\tau c_u)^{-2}$ and projection $P_{\Theta(r',s')}$ where

$$s'=\lceil 36\tau^8\kappa^4 s\rceil,\qquad\text{and}\qquad r'=\lceil 36\tau^8\kappa^4 r\rceil,$$

guarantees that, with probability at least $1-Kc_2\exp\{-c_3\max(d_1,d_2,\log d_3)\}$, after $K\ge 2\tau^4\kappa^2\log(\|T\|_{\mathrm{F}}/\epsilon)$ iterations,

$$\|\widehat{T}_K-T\|_{\mathrm{F}}\le c_4\sigma\sqrt{\frac{sr\max\{d_1,d_2,\log(d_3)\}}{n}}+\epsilon$$

for any $\tau>1$, and some constants $c_2,c_3,c_4>0$.

The convex regularization approach defined by Raskutti and Yuan (2015) is not directly applicable for $\Theta_2(r,s)$ since there is no suitable choice of regularizer that imposes both low-rankness of each slice and sparsity. Therefore we discuss the convex regularization approach applied to the parameter space $\Theta_1(r)$ for which a natural choice of regularizer is:

$$\mathcal{R}_1(A)=\sum_{j_3=1}^{d_3}\|A_{\cdot\cdot j_3}\|_*,$$

where $\|\cdot\|_*$ refers to the standard nuclear norm of a matrix. Let $\widehat{T}$ be an estimator corresponding to the minimizer of the regularized least-squares estimator defined by (8) with regularizer $\mathcal{R}_1(A)$. Lemma 6 in Raskutti and Yuan (2015) proves that

$$\|\widehat{T}-T\|_{\mathrm{F}}\lesssim\sqrt{\frac{r\max(d_1,d_2,\log d_3)}{n}}.$$

Notice that both $\Theta_1(r)$ and $\Theta_2(r,s)$ focus on the low-rankness of matrix slices of a tensor, and actually $\Theta_1(\cdot)$ can be seen as relaxation of $\Theta_2(\cdot,\cdot)$ since $\Theta_2(s,r)\subset\Theta_1(sr)$. Theorem 4 guarantees that, under the restriction of sparse slices of low-rank matrices, PGD achieves the linear convergence rate with the statistical error of order

$$\sqrt{\frac{sr\max\{d_1,d_2,\log(d_3)\}}{n}}.$$

If we compare this result with the risk bound of the convex regularization approach where the true tensor parameter lies in $\Theta_1(r)$ we see that replacing r by sr yields the same rate which makes intuitive sense in light of the observation that $\Theta_2(s,r)\subset\Theta_1(sr)$.

4.2. Low Tucker rank

We now consider the general set of tensors with low Tucker rank:

$$\Theta_3(r_1,r_2,r_3)=\left\{A\in\mathbb{R}^{d_1\times d_2\times d_3}:r_i(A)\le r_i\text{ for }i=1,2,3\right\}.$$

Although we focus on $N=3$, note that $\Theta_3(r_1,r_2,r_3)$ can be easily extended to general N and we also consider $N=4$ in the simulations.

To define the projection $P_{\Theta_3(r_1,r_2,r_3)}$ on to $\Theta_3(r_1,r_2,r_3)$, we exploit the connection between Tucker ranks and ranks of different matricizations mentioned earlier. Recall that the matricization operator $\mathcal{M}_j$ maps a tensor to a matrix and the inverse operator $\mathcal{M}_j^{-1}$ maps a matrix back to a tensor. Let $\bar{P}_r(M)$ be the low-rank projection operator that maps a matrix M to its best rank r approximation. Then we can define the approximate projection $\widehat{P}_{\Theta_3(r_1,r_2,r_3)}$ as follows:

$$\widehat{P}_{\Theta_3(r_1,r_2,r_3)}(A):=(\mathcal{M}_3^{-1}\circ\bar{P}_{r_3}\circ\mathcal{M}_3)\circ(\mathcal{M}_2^{-1}\circ\bar{P}_{r_2}\circ\mathcal{M}_2)\circ(\mathcal{M}_1^{-1}\circ\bar{P}_{r_1}\circ\mathcal{M}_1)(A).\quad(11)$$

The order of which matricization is performed is nonessential. Similar to before, we have the following projection lemma to be essential in the analysis of PGD applied to the restricted parameter space Θ_3.

Lemma 3 *Suppose* $Z \in \Theta_3(r_1^{(0)}, r_2^{(0)}, r_3^{(0)})$, *and* $r_i^{(1)} < r_i^{(2)} < r_i^{(0)}$ *for* $i = 1, 2, 3$. *Then for any* $Y \in \Theta_3(r_1^{(1)}, r_2^{(1)}, r_3^{(1)})$, *we have*

$$\|\widehat{P}_{\Theta_3(r_1,r_2,r_3)}(Z) - Z\|_{\mathrm{F}} \leq [(\beta_1+1)(\beta_2+1)(\beta_3+1) - 1]\|Y - Z\|_{\mathrm{F}}$$

where $\beta_i = \sqrt{(r_i^{(0)} - r_i^{(2)})/(r_i^{(0)} - r_i^{(1)})}$.

This allows us to derive the following result for the PGD algorithm applied with projection operator $\widehat{P}_{\Theta_3(r_1',r_2',r_3')}(\cdot)$. Basically, the sequential matrix low-rank projection, as an approximate projection onto low Tucker rank subset, could achieve the same order error rate as the exact low Tucker rank projection which might involve expensive iterative computation.

Theorem 5 *Let* $\{X^{(i)}, Y^{(i)}\}_{i=1}^n$ *follow a Gaussian linear model as defined by* (6) *with* $T \in \Theta_3(r_1, r_2, r_3)$ *and*

$$n \geq c_1 \cdot \min\{r_1(d_1 + d_2d_3), r_2(d_2 + d_1d_3), r_3(d_3 + d_1d_2)\},$$

for some constant $c_1 > 0$. *Then, applying the PGD algorithm with step size* $\eta = (\tau c_u)^{-2}$ *and projection* $\widehat{P}_{\Theta_3(r_1',r_2',r_3')}$ *where*

$$r_i' = \lceil 196\tau^8\kappa^4 r_i \rceil \text{ for } i = 1, 2, 3$$

guarantees that, with probability at least $1 - Kc_2 \exp\{-c_3 \min(d_1+d_2d_3, d_2+d_1d_3, d_3+d_1d_2)\}$, *after* $K \geq 2\tau^4\kappa^2 \log(\|T\|_{\mathrm{F}}/\epsilon)$ *iterations,*

$$\|\widehat{T}_K - T\|_{\mathrm{F}} \leq c_4\sigma\sqrt{\frac{\min\{r_1(d_1 + d_2d_3), r_2(d_2 + d_1d_3), r_3(d_3 + d_1d_2)\}}{n}} + \epsilon$$

for any $\tau > 1$, *and some constants* $c_2, c_3, c_4 > 0$.

In Raskutti and Yuan (2015), the following convex low-rankness regularizer is considered for the space $\Theta_3(r)$:

$$\mathcal{R}_2(A) = \frac{1}{3}\left(\|\mathcal{M}_1(A)\|_* + \|\mathcal{M}_2(A)\|_* + \|\mathcal{M}_3(A)\|_*.\right).$$

Let $\widehat{T}$ be an estimator corresponding to the minimizer of the regularized least-squares estimator defined by (8) with regularizer $\mathcal{R}_2(A)$. An adaptation of the proof of Lemma 10 in Raskutti and Yuan (2015) to $\Theta_3(r_1, r_2, r_3)$ proves that:

$$\|\widehat{T} - T\|_{\mathrm{F}} \lesssim \sqrt{\frac{\max(r_1, r_2, r_3) \cdot \max(d_1 + d_2d_3, d_2 + d_1d_3, d_3 + d_1d_2)}{n}}.$$

This shows that convex relaxation in this particular case has greater mean-squared error since the minimum is replaced by the maximum. The underlying reason is that the non-convex PGD approach selects the optimal choice of matricization whereas the convex regularization approach takes an average of the three matricizations which is sub-optimal. While it may be argued that one could use the regularizer corresponding with only the optimal $\mathcal{M}_1(.)$, $\mathcal{M}_2(.)$, or $\mathcal{M}_3(.)$, since the Tucker ranks (r_1, r_2, r_3) are unknown, it is impossible to know which matricization to use in the convex case.

5. Simulations

In this section, we provide a simulation study that firstly verifies that the non-convex PGD algorithm performs well in solving least-squares, logistic and Poisson regression problems, compares the non-convex PGD approach with the convex regularization approach we discussed earlier, and also compares using tensor regularization and naive matricization schemes. Our simulation study includes both third and fourth order tensors. For the purpose of illustration, we consider the balanced-dimension situation where $d = d_1 = d_2 = d_3 (= d_4)$, and hence the number of elements is $p = d^3$ for a third order tensor and $p = d^4$ for a fourth order tensor.

5.1. Data generation

We first describe three different ways of generating random tensor coefficient T with different types of low tensor rank structure.

1. (Low CP rank) Generate three independent groups of r independent random vectors of unit length, $\{u_{k,1}\}_{k=1}^r$, $\{u_{k,2}\}_{k=1}^r$ and $\{u_{k,3}\}_{k=1}^r$. To do this we perform the SVD of a Gaussian random matrix three times and keep the r leading singular vectors, and then compute the outer-product

   $$T = \sum_{k=1}^r u_{k,1} \otimes u_{k,2} \otimes u_{k,3}.$$

   The T produced in this way is guaranteed to have CP rank at most r. This can easily be extended to $N = 4$.

2. (Low Tucker rank) Generate $M_{d\times d\times d}$ with i.i.d. $\mathcal{N}(0, 1)$ elements and then do approximate Tucker rank-r projection (successive low rank approximation of mode-1, mode-2 and mode-3 matricization) to get $T = \widehat{P}_{\Theta_3(r,r,r)}(M)$. The T produced in this way is guaranteed to have largest element of Tucker rank at most r. Once again this is easily extended to the $N = 4$ case.

3. (Sparse slices of low-rank matrices) In this case $N = 3$. Generate s slices of random rank-r matrices, (with eigenvalues all equal to one and random eigenvectors), and fill up the remaining $d - s$ slices with zero matrices to get $d \times d \times d$ tensor T. The T produced in this way is guaranteed to fall in $\Theta_2(r, s)$.

Then we generate covariates $\{X^{(i)}\}_{i=1}^n$ to be i.i.d random matrices filled with i.i.d $\mathcal{N}(0, 1)$ entries. Finally, we simulate three GLM model, the Gaussian linear model, logistic regression and Poisson regression as follows.

1. (Gaussian linear model) We simulated noise $\{\epsilon^{(i)}\}_{i=1}^n$ independently from $\mathcal{N}(0, \sigma^2)$ and we vary σ^2. The noisy observation is then

   $$Y^{(i)} = \langle X^{(i)}, T\rangle + \epsilon^{(i)}.$$

2. (Logistic regression) We simulated Binomial random variables:

   $$Y^{(i)} \sim \text{Binomial}(m, p_i), \text{ where } p_i = \text{logit}(\alpha \cdot \langle X^{(i)}, T\rangle).$$

3. (Poisson regression) We simulated

$$Y^{(i)} \sim \text{Poisson}(\lambda_i), \text{ where } \lambda_i = m \exp(\alpha \cdot \langle X^{(i)}, T \rangle).$$

5.2. Convergence of PGD under restricted tensor regression

The first set of simulations investigates the convergence performance of PGD under various constraints and step-size choices for three different types of low-rankness. One of the important challenges when using the projected gradient descent algorithm is choosing the step-sizes (just like selecting the regularization parameter for convex regularization schemes) and the step-size choices stated in Theorem 2 depend on non-computable parameters (e.g. $c_u, c_\ell, ...$). In practice, the step is very important in that large step size would lead to divergence and small step size could cause slow convergence. We suggest either start with relatively large step size and continuously decrease the step size if divergence behavior is observed until tolerable step size is found, or start with relatively small step size that can guarantee convergence and gradually increase step size along the way while going back to small step size when divergence behavior is observed, which is similar to warm start strategy usually used to speed up convergence. In all our simualtions, the step-size η is set as a constant specified in each plot.

5.2.1. Third-order tensors

In the first two cases (see cases below), PGD with approximate projection $\widehat{P}_{\Theta_3(r',r',r')}$ were applied with different choices of (r', η) while in the third case the PGD with exact projection $P_{\Theta_2(r',s')}$ were adopted with different choices of (r', s', η).

Case 1a: (Gaussian) Low CP Rank with $p = 50^3$, $n = 4000$, $r = 5$, $\sigma = 0.5$ (SNR ≈ 4.5);

Case 2a: (Gaussian) Low Tucker Rank with $p = 50^3$, $n = 4000$, $r = 5$, $\sigma = 5$ (SNR ≈ 7.2);

Case 3a: (Gaussian) Slices of Low-rank Matrices with $p = 50^3$, $n = 4000$, $r = 5$, $s = 5$, $\sigma = 1$ (SNR ≈ 5.2).

Figures 1, 2 and 3 plot normalized rooted mean squared error (rmse) $\|\widehat{T} - T\|_{\text{F}}/\|T\|_{\text{F}}$ versus number of iterations, showing how fast rmse decreases as the number of iterations increases, under different (r', η) or (r', s', η). Notice that here we plot the average rmse with error bar to be the standard deviation for ten runs for Cases 1a, 2a and 3a.

Overall, the plots show the convergence of rmse's, and that the larger the r' or s' is, the greater the converged rmse will be, meaning that misspecification of rank/sparsity will do harm to the performance of PGD. In terms of the choice of step size, the experiments inform us that if η is too large, the algorithm may not converge and the range of tolerable step-size choices varies in different cases. In general, the more misspecified the constraint parameter(s) is(are), the lower the tolerance for step size will be. On the other hand, as we can see in all cases, given η under a certain tolerance level, the larger the η is, the faster the convergence will be.

5.2.2. Fourth-order tensors

Although we have focused on third order tensor for brevity, our method applies straightforwardly to higher order tensors. For illustration, we considered the following two examples which focus on estimating fourth order low rank tensors.

Case 4a: (Gaussian) Low CP Rank with $p = 20^4$, $n = 4000$, $r = 5$, $\sigma = 0.5$ (SNR ≈ 4.4);

Case 5a: (Gaussian) Low Tucker Rank with $p = 20^4$, $n = 4000$, $r = 5$, $\sigma = 5$ (SNR ≈ 7.4).

Figure 4 plots average rmse (with s.d. as error bar) for ten runs vs number of iterations for Case 4a and Case 5a using $\eta = 0.2$ under various choices of low-rankness constraint parameter r'. In general the convergence behavior for Case 4a and Case 5a are similar to those for Case 1a and Case 2a.

5.2.3. Logistic and Poisson regression

In the next set of simulations, we study the convergence behavior of the PGD applied to logistic and Poisson regression situation.

Case 1b: (Logistic) Low CP Rank with $p = 50^3$, $n = 4000$, $r = 5$, $m = 10$, $\alpha = 0.1$ (SNR ≈ 3.8);

Case 1c: (Poisson) Low CP Rank with $p = 30^3$, $n = 4000$, $r = 5$, $m = 5$, $\alpha = 0.5$ (SNR ≈ 4.8).

The results presented in Figures 5 and 6 exhibit similar pattern of convergence as in Figure 1. We observe also that in the case of low Tucker rank and sparse slices of low-rank matrices, logistic and Poisson regression have similar convergence behavior to least-squares regression. In general, a relaxed projection step is inferior to using the true rank parameter for projection. Once again as the step-size increases, the convergence will speed up until the step size becomes too large to guarantee convergence.

5.3. Comparison of non-convex PGD to convex regularization

Next, we compare the PGD method with convex regularization methods (implemented via `cvx`). In general, the `cvx` based regularization algorithm is significantly slower than the PGD method. This is partly due to the fact that the infrastructure of generic `cvx` is not tailored to solve the specific convex optimization problems. On the other hand, the PGD is much easier to implement and enjoys fast rates of convergence, which may also contribute to its improved performance in terms of run-time. Besides, `cvx` cannot handle p as large as those in Cases 1a, 2a and 3a. Hence, in order to do comparison in terms of the estimation error, we resort to moderate p so that `cvx` runs to completion. The simulation setup is as follows:

Case 6a: (Gaussian) Low CP Rank with $p = 10^3$, $n = 1000$, $r = 5$, $\sigma = 0.5, 1,$ or 2 (SNR $\approx$ 4.8, 2.4 or 1.2);

Case 7a: (Gaussian) Low Tucker Rank with $p = 10^3$, $n = 1000$, $r = 5$, $\sigma = 2.5, 5,$ or 10 (SNR $\approx$ 7.2, 3.6, or 1.8);

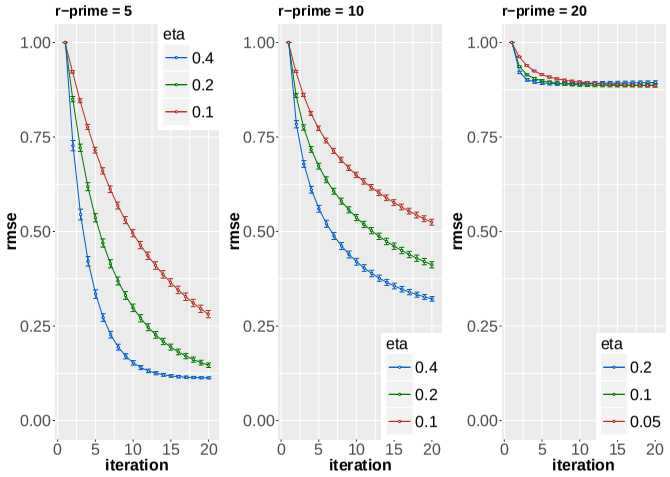


Figure 1: Case 1a: Low CP rank

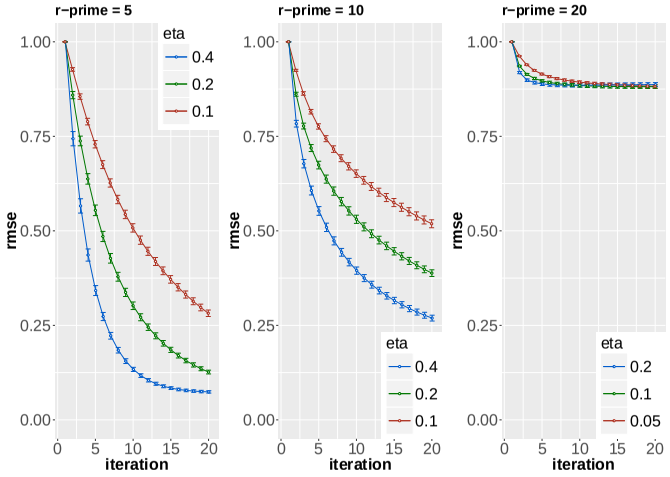


Figure 2: Case 2a: Low Tucker rank

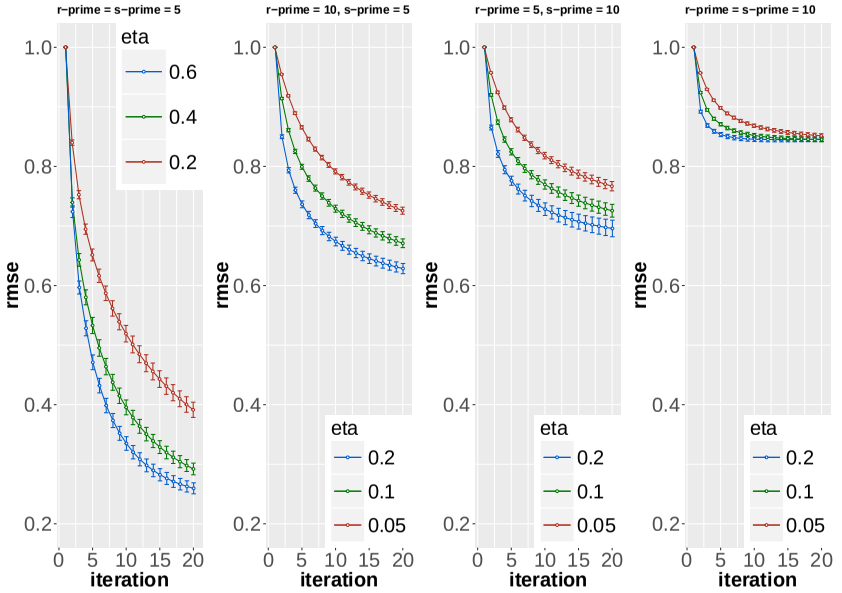


Figure 3: Case 3a: Sparse slices of low-rank matrices

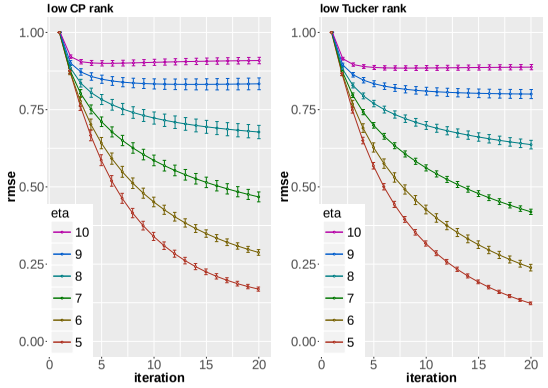


Figure 4: Case 4a, 5a: 4th order tensor

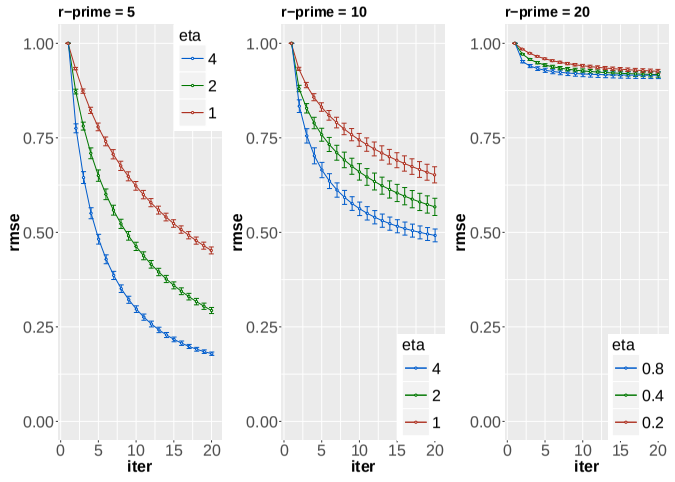


Figure 5: Case 1b: (Logistic) Low CP rank

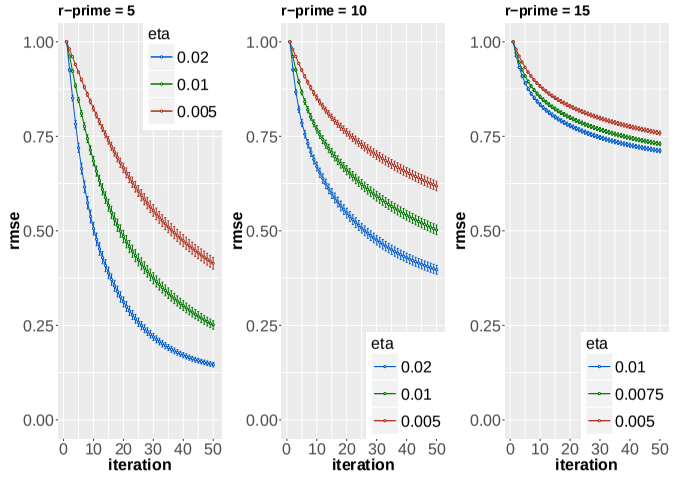


Figure 6: Case 1c: (Poisson) Low CP rank

Case 8a: (Gaussian) Slices of Low-rank Matrices with $p = 10^3$, $n = 1000$, $r = 5$, $s = 5$, $\sigma = 0.5, 1,$ or 2 (SNR $\approx$ 9.6, 4.8, or 2.4);

Case 6b: (Logistic) Low CP Rank with $p = 10^3$, $n = 1000$, $\alpha = 3.5$, $r = 5$, $m = 20, 5,$ or 1 (SNR $\approx$ 9.0, 4.5 or 2.0);

Case 7b: (Logistic) Low Tucker Rank with $p = 10^3$, $n = 1000$, $\alpha = 0.5$, $r = 5$, $m = 20, 5,$ or 1 (SNR $\approx$ 9.6 , 4.9 or 2.2);

Case 8b: (Logistic) Slices of Low-rank Matrices with $p = 10^3$, $n = 1000$, $\alpha = 1.2$, $r = 5$, $s = 5$, $m = 20, 5,$ or 1 (SNR $\approx$ 7.7 , 3.8 , or 1.7);

Case 6c: (Poisson) Low CP Rank with $p = 10^3$, $n = 1000$, $\alpha = 0.5$, $r = 5$, $m = 20, 5,$ or 1 (SNR $\approx$ 9.6 , 4.7, or 2.1);

Case 7c: (Poisson) Low Tucker Rank with $p = 10^3$, $n = 1000$, $\alpha = 0.06$, $r = 5$, $m = 20, 5,$ or 1 (SNR $\approx$ 9.0, 4.5 or 2.0);

Case 8c: (Poisson) Slices of Low-rank Matrices with $p = 10^3$, $n = 1000$, $\alpha = 0.25$, $r = 5$, $s = 5$, $m = 30, 10,$ or 5 (SNR $\approx$ 15.4, 8.8 or 6.2).

Cases 6(a,b,c), 7(a,b,c) and 8(a,b,c) were constructed to represent different types of tensor low-rankness structure in least-square, logistic and Poisson regression. In each case, three levels of SNR(high, moderate and low) are considered. For each setting, we simulated 50 groups of (T, ϵ, X) and run PGD and convex-regularization methods for the recovery of T to get average rmse with standard deviation for the two approaches respectively. Here we are comparing the best performance achieved by the PGD and convex regularization method respectively: for the PGD we use true parameter as the constraint parameter $r' = r$ (and $s' = s$); for convex regularization method, we do a grid search to choose the tuning parameter that yields the smallest rmse.

The results are summarized in Table 1. These results show that in general, the PGD method produces smaller rmse's than convex regularization methods regardless of the noise level of the data.

5.4. Comparison of tensor low-rankness approach and simple matricization approach

Finally, we display the value of considering low-rank tensors rather than a naive low-rank matricazation scheme. To begin, we consider the following low tensor ranks schemes.

Case 9a: (Gaussian) Low Tucker Rank with $p = 30^3$, $n = 4000$, $r = 5$, $\sigma = 5, 10,$ or 20 (SNR $\approx$ 5.9, 2.9 or 1.5);

Case 10a: (Gaussian) Low CP Rank with $p = 30^3$, $n = 4000$, $r = 5$, $\sigma = 0.5, 1, 2$ or (SNR $\approx$ 4.5, 2.2, or 1.1);

Case 11a: (Gaussian) Slices of Low-rank Matrices with $p = 30^3$, $n = 4000$, $r = 5$, $s = 5$, $\sigma = 1, 2,$ or 4 (SNR $\approx$ 5.0, 2.5 or 1.3).

rmse (sd)	SNR	PGD	Convex Regularization
Case 6a	High	**0.11 (0.01)**	0.28 (0.02)
	Moderate	**0.22 (0.01)**	0.47 (0.02)
	Low	**0.46 (0.03)**	0.69 (0.02)
Case 7a	High	**0.07 (0.01)**	0.18 (0.01)
	Moderate	**0.14 (0.01)**	0.32 (0.02)
	Low	**0.28 (0.02)**	0.51 (0.02)
Case 8a	High	**0.08 (0.01)**	0.12 (0.01)
	Moderate	**0.16 (0.01)**	0.23 (0.01)
	Low	**0.30 (0.01)**	0.41 (0.02)
Case 6b	High	**0.16 (0.01)**	0.44 (0.02)
	Moderate	**0.20 (0.01)**	0.54 (0.02)
	Low	**0.35 (0.02)**	0.66 (0.02)
Case 7b	High	**0.17 (0.01)**	0.46 (0.02)
	Moderate	**0.22 (0.01)**	0.55 (0.02)
	Low	**0.35 (0.01)**	0.67 (0.01)
Case 8b	High	**0.26 (0.01)**	0.37 (0.02)
	Moderate	**0.34 (0.02)**	0.50 (0.01)
	Low	**0.56 (0.04)**	0.68 (0.02)
Case 6c	High	**0.09 (0.01)**	0.57 (0.03)
	Moderate	**0.17 (0.01)**	0.61 (0.04)
	Low	**0.39 (0.04)**	0.71 (0.03)
Case 7c	High	**0.12 (0.01)**	0.74 (0.02)
	Moderate	**0.21 (0.02)**	0.75 (0.02)
	Low	**0.43 (0.06)**	0.80 (0.02)
Case 8c	High	**0.13 (0.01)**	0.79 (0.03)
	Moderate	**0.22 (0.03)**	0.81 (0.03)
	Low	**0.32 (0.03)**	0.83 (0.02)

Table 1: rmse of nonconvex PGD vs convex regularization

rmse (sd)	SNR	sequential low-rank projection	naive matricization
Case 9a	High	**0.22 (0.01)**	0.57 (0.01)
	Moderate	**0.24 (0.01)**	0.65 (0.01)
	Low	**0.31 (0.01)**	0.80 (0.01)
Case 10a	High	**0.10 (0.01)**	0.58 (0.01)
	Moderate	**0.35 (0.01)**	0.69 (0.01)
	Low	**0.35 (0.01)**	0.85 (0.01)
Case 11a	High	**0.15 (0.01)**	0.56 (0.01)
	Moderate	**0.29 (0.01)**	0.67 (0.01)
	Low	**0.57 (0.02)**	0.83 (0.01)

Table 2: rmse of approximate Tucker projection vs naive matricization

Case 9a and Case 10a is when the true tensor parameter has low Tucker rank or low CP rank. Case 11a is when the true tensor parameter is sparse slices of low-rank matrices. Three levels of SNR(high, moderate and low) are considered. For each setting, we simulated 50 groups of (T, ϵ, X) and run two approaches to recover T. The first approach we use is the PGD with our tensor low-rank (approximate) projection. In the Case 9a and 10a, tensor Tucker low-rankness is used via sequential matrix low rank projection, i.e. the approximate low Tucker rank projection $\hat{\Theta}_3(r', r', r')$ with $r' = 5$. While in Case 11a we exploit combination of group-sparsity and matrix low-rankness via $\Theta_2(r', s')$ with $r' = s' = 5$. The second approach is to naively use the matricization where we view the $p \times p \times p$ tensor simply as a $p \times p^2$ matrix and then impose low-rank structure. For naive matricization approach, we report the results of matrix rank-r' projection with $r' = 5$ which yields the best performance of its kind. Table 2 summarizes the average rmse with standard deviation of the two approaches for 50 random runs respectively. Hence we can see that if we partially discard the tensor structure by using only the low-rankness after matricization, the performance is greatly inferior to the low Tucker approximate projection approach. This applies to all three cases considered here, i.e. Tucker, CP and sparse slices of low-rank matrices.

6. Conclusion

In this paper, we provide a general framework that offers theoretical guarantees for learning high-dimensional tensor regression models under different low-rank structural assumptions using the PGD algorithm applied to a potentially non-convex constraint set Θ in terms of its *localized Gaussian width*. Our framework is the first general theory for PGD applied to tensor problems and given that the notion of low-rank structure is ambiguous for tensors, our general framework applies treats them in a unified way.

By providing statistical guarantees in terms of localized Gaussian width, we prove that the PGD approach has mean-squared error that is no worse than the convex regularization counter-part studied in Raskutti and Yuan (2015). We also provided three concrete examples Θ_1, Θ_2 and Θ_3 of low-dimensional tensor structure and provide implementable (approximate) projections and provide mean-squared error guarantees. For Θ_1 and Θ_2 we

provide a convex regularization scheme that achieves the same rate, while for Θ_3 we show that the non-convex PGD approach achieves superior mean-squared error.

We supplement our theoretical results with simulations which show that, under several common settings of generalized low rank tensor regression, the projected gradient descent approach is superior both in terms of statistical error and run-time compared to convex approaches provided the step-sizes of the projected descent algorithm are suitably chosen. Additional simulation results show that PGD with tensor low-rankness constraint also outperforms naive matricization approaches.

7. Proofs

7.1. Proof of general results

We first prove the results of Section 3: Theorems 1, 2 and 3. In particular, we first provide a proof for Theorem 1. For convenience we first state the proof for the Gaussian case (Theorem 3) and then describe the necessary changes needed for the more general GLM case (Theorem 2).

7.1.1. Proof of Theorem 1

The proof follows very similar steps to those developed in Jain et al. (2014, 2016). Recall that $\widehat{T}_{k+1} = P_{\Theta(t_1)}(g_k)$ where $g_k = \widehat{T}_k - \eta \nabla f(\widehat{T}_k)$. For $\widehat{T}_{k+1} \in \Theta(t_1)$ and any $T \in \Theta(t_0 - t_1)$, the superadditivity condition guarantees that there exists a linear subspace $\mathcal{A} = \{\alpha_1 \widehat{T}_{k+1} + \alpha_2 T \mid \alpha_1, \alpha_2 \in \mathbb{R}\}$ such that $\widehat{T}_{k+1} \in \mathcal{A}$, $T \in \mathcal{A}$ and $\mathcal{A} \subset \Theta(t_0)$.

The contractive projection property CPP(δ) implies that for any $T \in \Theta(t_0 - t_1)$,

$$\|(\widehat{T}_{k+1} - g_k)_{\mathcal{A}}\|_{\mathrm{F}} \leq \delta \left\| \frac{t_0 - t_1}{t_1} \right\|_{\ell_\infty}^{1/2} \cdot \|(T - g_k)_{\mathcal{A}}\|_{\mathrm{F}}.$$

Since $t_1 = \left\lceil \frac{4\delta^2 C_u^2 C_l^{-2}}{1+4\delta^2 C_u^2 C_l^{-2}} \cdot t_0 \right\rceil$,

$$\delta \left\| \frac{t_0 - t_1}{t_1} \right\|_{\ell_\infty}^{1/2} \leq (2C_u C_l^{-1})^{-1}.$$

Hence,

$$\begin{aligned}
\|\widehat{T}_{k+1} - T\|_{\mathrm{F}} &\leq \|\widehat{T}_{k+1} - g_k\|_{\mathrm{F}} + \|T - g_k\|_{\mathrm{F}} \\
&\leq \left(1 + \delta \left\| \frac{t_0 - t_1}{t_1} \right\|_{\ell_\infty}^{1/2}\right) \|(T - g_k)_{\mathcal{A}}\|_{\mathrm{F}} \\
&\leq (1 + (2C_u C_l^{-1})^{-1}) \|(T - g_k)_{\mathcal{A}}\|_{\mathrm{F}} \\
&\leq \left(1 + \frac{C_l}{2C_u}\right) \|[T - \widehat{T}_k - \eta(\nabla f(T) - \nabla f(\widehat{T}_k))]_{\mathcal{A}}\|_{\mathrm{F}} + 2\eta \|[\nabla f(T)]_{\mathcal{A}}\|_{\mathrm{F}},
\end{aligned}$$

where the final inequality follows from the triangle inequality. If we define the Hessian matrix of the function f of a vectorized tensor as

$$H(A) = \nabla^2 f(A),$$

the Mean Value Theorem implies that

$$\mathrm{vec}(\nabla f(T) - \nabla f(\widehat{T}_k)) = H(\widehat{T}_k + \alpha(T - \widehat{T}_k)) \cdot (T - \widehat{T}_k),$$

for some $0 < \alpha < 1$, and

$$\begin{aligned}
\|\widehat{T}_{k+1} - T\|_{\mathrm{F}} \leq (1 + (2C_u C_l^{-1})^{-1}) \|[(I - \eta H(\widehat{T}_k + \alpha(T - \widehat{T}_k)))\mathrm{vec}(\widehat{T}_k - T)]_{\mathrm{vec}(\mathcal{A})}\|_{\ell_2} \\
+ 2\eta \|[\nabla f(T)]_{\mathcal{A}}\|_{\mathrm{F}}.
\end{aligned}$$

We now appeal to the following lemma:

Lemma 4 *Suppose $\mathcal{S}$ is a linear subspace of $\mathbb{R}^d$, and H is an $d \times d$ positive semidefinite matrix. For any given $0 < c < 1$, if for any $x \in \mathcal{S}$,*

$$c x^\top x \leq x^\top H x \leq (2 - c) x^\top x, \tag{12}$$

then for any $z \in \mathcal{S}$, we have

$$\|[(I - H)z]_{\mathcal{S}}\|_{\ell_2} \leq (1 - c)\|z\|_{\ell_2},$$

$(\cdot)_{\mathcal{S}}$ stands for the projection onto the subspace $\mathcal{S}$.

Proof [Proof of Lemma 4] Suppose the orthonomal basis of $\mathcal{S}$ is $e_1 \ldots, e_q$, and then

$$\mathbb{R}^d = \{ce_1 | c \in \mathbb{R}\} \oplus \ldots \oplus \{ce_q | c \in \mathbb{R}\} \oplus \mathcal{S}^\perp$$

For positive semidefinite H, it can be decomposed as follows

$$H = D^\top D.$$

Hence we can decompose the rows of D to get

$$D = \sum_{i=1}^{q} \lambda_i e_i^\top + (y_1, \ldots, y_n)^\top$$

where $y_1, \ldots, y_n \in \mathcal{S}^\perp$, and $\lambda_i \in \mathbb{R}^n$ for $i = 1, \ldots, q$. Therefore,

$$D^\top D = \sum_{i=1}^{q}\sum_{j=1}^{q} (\lambda_i^\top \lambda_j) e_i e_j^\top + \sum_{k=1}^{n} y_k y_k^\top + (y_1, \ldots, y_n) \sum_{i=1}^{q} \lambda_i e_i^\top + \sum_{i=1}^{q} e_i \lambda_i^\top \cdot (y_1, \ldots, y_n)^\top.$$

Now for any $(\alpha_1, \ldots, \alpha_q)^\top \in \mathbb{R}^q$, we have $x = \sum_{i=1}^{q} \alpha_i e_i \in \mathcal{S}$, and hence

$$x^\top D^\top D x = (\sum_{i=1}^{q} \alpha_i e_i)^\top D^\top D (\sum_{i=1}^{q} \alpha_i e_i) = \sum_{i=1}^{q}\sum_{j=1}^{q} (\lambda_i^\top \lambda_j) \alpha_i \alpha_j.$$

The equation 12 then implies that the matrix

$$\Lambda = \{\Lambda_{i,j}\}_{i,j=1}^{q} \text{ where } \Lambda_{i,j} = \lambda_i^\top \lambda_j$$

has eigenvalues bounded by c from below and $2-c$ from above. Next, notice that for any $z \in \mathcal{S}$, we have $z = \sum_{i=1}^{q} \beta_i e_i$ for some $(\beta_1, \ldots, \beta_q)^\top \in \mathbb{R}^q$, and hence due to the fact that $y_1, \ldots y_n \in \mathcal{S}^\perp$

$$(I - D^\top D)z = \left(I - \sum_{i=1}^{q}\sum_{j=1}^{q} (\lambda_i^\top \lambda_j) e_i e_j^\top + (y_1, \ldots, y_n) \sum_{i=1}^{q} \lambda_i e_i^\top \right) \sum_{i=1}^{q} \beta_i e_i,$$

and furthermore

$$\begin{aligned} [(I - D^\top D)z]_{\mathcal{S}} &= \left(I - \sum_{i=1}^{q}\sum_{j=1}^{q} (\lambda_i^\top \lambda_j) e_i e_j^\top \right) \sum_{i=1}^{q} \beta_i e_i \\ &= (e_1, \ldots, e_q)(I_{q\times q} - \Lambda)(\beta_1, \ldots \beta_q)^\top, \end{aligned}$$

and

$$\|[(I - D^\top D)z]_{\mathcal{S}}\|_{\ell_2}^2 = (\beta_1, \ldots \beta_q)(I_{q\times q} - \Lambda)(\beta_1, \ldots \beta_q)^\top,$$

which completes the proof. ■

Condition $RSCS(\Theta(t_0), C_l, C_u)$ guarantees the condition of Lemma 4 is satisfied with $H = \eta H(\widehat{T}_k + \alpha(T - \widehat{T}_k))$, $c = (C_u C_l^{-1})^{-1}$ and $\mathcal{S} = \mathcal{A}$. Hence Lemma 4 implies that

$$\|\widehat{T}_{k+1} - T\|_{\mathrm{F}} \leq (1 + (2C_u C_l^{-1})^{-1}) \left(1 - (C_u C_l^{-1})^{-1}\right) \|\widehat{T}_k - T\|_{\mathrm{F}} + 2\eta \|[\nabla f(T)]_{\mathcal{A}}\|_{\mathrm{F}}.$$

Therefore for any k,

$$\|\widehat{T}_{k+1} - T\|_{\mathrm{F}} \leq (1 - (2C_u C_l^{-1})^{-1}) \|\widehat{T}_k - T\|_{\mathrm{F}} + 2\eta Q,$$

where

$$Q = \sup_{\mathcal{A}_0 \subset \Theta(t_0)} \|(\nabla f(T))_{\mathcal{A}_0}\|_{\mathrm{F}},$$

where $\mathcal{A}_0$ is any linear subspace of $\Theta(t_0)$.We then appeal to the following result.

Lemma 5 *Suppose $\mathcal{A}$ is a linear subspace of tensor space Ω. For any $L \in \Omega$,*

$$\|(L)_{\mathcal{A}}\|_{\mathrm{F}} = \sup_{A \in \mathcal{A} \cap \mathbb{B}_{\mathrm{F}}(1)} \langle A, L \rangle$$

Proof [Proof of Lemma 5] First, we are going to show that

$$\|(L)_{\mathcal{A}}\|_{\mathrm{F}} \leq \sup_{A \in \mathcal{A} \cap \mathbb{B}_{\mathrm{F}}(1)} \langle A, L \rangle$$

Suppose we have $(L)_{\mathcal{A}} = P \in \mathcal{A}$. Since for any $\alpha > -1$, $P + \alpha P \in \mathcal{A}$, and hence

$$\|P + \alpha P - L\|_{\mathrm{F}} = \|P - L\|_{\mathrm{F}} + \alpha^2 \|P\|_{\mathrm{F}} + \alpha \langle P, P - L \rangle \leq \|P - L\|_{\mathrm{F}}$$

we must have $\langle P, P - L \rangle = 0$, i.e. $\langle P, L \rangle = \langle P, P \rangle$. (otherwise α of small magnitude with the same sign of $\langle P, P - L \rangle$ will violate the inequality). Therefore,

$$\sup_{A \in \mathcal{A} \cap \mathbb{B}_{\mathrm{F}}(1)} \langle A, L \rangle \geq \left\langle \frac{P}{\|P\|_{\mathrm{F}}}, L \right\rangle = \|P\|_{\mathrm{F}} = \|(L)_{\mathcal{A}}\|_{\mathrm{F}}$$

What remains is to show

$$\|(L)_{\mathcal{A}}\|_{\mathrm{F}} \geq \sup_{A \in \mathcal{A} \cap \mathbb{B}_{\mathrm{F}}(1)} \langle A, L \rangle$$

For any $D \in \mathcal{A} \cap \mathbb{B}_{\mathrm{F}}(1)$, let D_α be the projection of L onto $\{\alpha D | \alpha \geq 0\}$, and hence

$$\langle D_\alpha, L \rangle = \langle D_\alpha, D_\alpha \rangle \leq \langle P, P \rangle.$$

Therefore, we have

$$\langle D, L \rangle \leq \left\langle \frac{D}{\|D\|_{\mathrm{F}}}, L \right\rangle = \left\langle \frac{D_\alpha}{\|D_\alpha\|_{\mathrm{F}}}, L \right\rangle \leq \|P\|_{\mathrm{F}}$$

which completes the proof. ■

Lemma 5 then implies

$$Q = \sup_{\mathcal{A}_0 \subset \Theta(t_0)} \|(\nabla f(T))_{\mathcal{A}_0}\|_{\mathrm{F}} = \sup_{\mathcal{A}_0 \subset \Theta(t_0)} \sup_{A \in \mathcal{A}_0 \cap \mathbb{B}_{\mathrm{F}}(1)} \langle \nabla f(T), A \rangle \leq \sup_{A \in \Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)} \langle \nabla f(T), A \rangle .$$

Therefore, after

$$K = \lceil 2C_u C_l^{-1} \log \frac{\|T\|_{\mathrm{F}}}{\epsilon} \rceil$$

iterations,

$$\|\widehat{T}_k - T\|_{\mathrm{F}} \leq \epsilon + 4\eta C_u C_l^{-1} \sup_{A \in \Theta(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)} \langle \nabla f(T), A \rangle ,$$

which completes the proof.

7.1.2. Proof of Theorem 3

Recall that the original least-squares objective is

$$f(A) = \frac{1}{2n} \sum_{i=1}^{n} (Y^{(i)} - \langle X^{(i)}, A \rangle)^2.$$

Hence, the gradient at true tensor coefficient T is :

$$\nabla f(T) = \frac{1}{n} \sum_{i=1}^{n} X^{(i)} \otimes [\langle T, X^{(i)} \rangle - Y^{(i)}] = -\frac{1}{n} \sum_{i=1}^{n} X^{(i)} \zeta^{(i)}$$

for the least-squares objective we consider. Further

$$\nabla^2 f(T) = \frac{1}{n} \sum_{i=1}^{n} X^{(i)} \otimes X^{(i)}.$$

Through vectorization, the Hessian matrix H is

$$H = \sum_{i=1}^{n} \text{vec}(X^{(i)})\text{vec}(X^{(i)})^\top.$$

Lemma 1 then implies that given $n \geq c_1 w_G^2[\Theta(t_0) \cap \mathbb{B}_{\text{F}}(1)]$, with probability at least

$$1 - c_2/2 \exp(-c_3 w_G^2[\Theta(t_0) \cap \mathbb{B}_{\text{F}}(1)]),$$

we have, for any $A \in \Theta(t_0)$,

$$\left(\tau^{-1}c_l\right)^2 \langle A, A\rangle \leq \frac{1}{n}\sum_{i=1}^{n}\langle X^{(i)}, A\rangle^2 \leq (\tau c_u)^2\langle A, A\rangle,$$

which guarantees the $RSCS(\Theta(t_0), C_l, C_u)$ condition with $C_u = \tau c_u$ and $C_l = \tau^{-1}c_l$. Thus Theorem 1 implies that

$$\|\widehat{T}_k - T\|_{\text{F}} \leq \epsilon + 4\eta\tau^2\kappa \sup_{A\in\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)} \left\langle \frac{1}{n}\sum_{i=1}^{n}\zeta^{(i)}X^{(i)}, A\right\rangle.$$

The last step is to show that

$$\sup_{A\in\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)} \left\langle \frac{1}{n}\sum_{i=1}^{n}\zeta^{(i)}X^{(i)}, A\right\rangle \leq 2c_u\sigma n^{-1/2} w_G[\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)],$$

with probability at least

$$1 - c_2/2\exp\left\{-c_3 w_G^2[\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)]\right\}.$$

This can be shown by simply applying Lemma 11 in Raskutti and Yuan (2015) and replacing $\{A|\mathcal{R}(A) \leq 1\}$ with $\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)$. Note that all the proof steps for Lemma 11 in Raskutti and Yuan (2015) are identical for $\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)$ since the sets $\Theta(t)$'s are symmetric.

7.1.3. Proof of Theorem 2

The proof follows the same flow as Theorem 3 but we requires an important concentration result from Mendelson (2014). Recall that in the GLM setting, according to (2),

$$f(A) = \frac{1}{n}\sum_{i=1}^{n}\left(a(\langle X^{(i)}, A\rangle) - Y^{(i)}\langle X^{(i)}, A\rangle\right).$$

Hence the gradient at true coefficient T is

$$\nabla f(T) = \frac{1}{n}\sum_{i=1}^{n}(\mu_i - Y^{(i)})X^{(i)},$$

where $\mu_i = a'(\langle X^{(i)}, T\rangle)$, and the Hessian matrix at vectorized tensor T is

$$\nabla^2 f(T) = \sum_{i=1}^{n} W_{ii}\text{vec}(X^{(i)})\text{vec}(X^{(i)})^\top.$$

where $W_{ii} = a''(\langle X^{(i)}, T\rangle)$.

Since $\text{Var}(Y^{(i)}) = a''(\langle X^{(i)}, T\rangle) = W_{ii}$, the moment assumption $1/\tau_0^2 \leq \text{Var}(Y^{(i)}) \leq \tau_0^2$ guarantees that

$$\frac{1}{\tau_0^2} \leq W_{ii} \leq \tau_0^2.$$

Plus, for any $\tau > \tau_0$, Lemma 1 guarantees that when $n > c_1 w_G^2[\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)]$,

$$\left((\tau/\tau_0)^{-1}c_l\right)^2\langle A, A\rangle \leq \frac{1}{n}\sum_{i=1}^{n}\langle X^{(i)}, A\rangle^2 \leq ((\tau/\tau_0)c_u)^2\langle A, A\rangle.$$

Therefore $RSCS(\Theta(t_0), C_l, C_u)$ condition holds with $C_l = \tau^{-1}c_l$ and $C_u = \tau c_u$. Thus Theorem 1 implies that

$$\|\widehat{T}_k - T\|_{\text{F}} \leq \epsilon + 4\eta\tau^2\kappa \sup_{A\in\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)} \left\langle \frac{1}{n}\sum_{i=1}^{n}(Y^{(i)} - \mu_i)X^{(i)}, A\right\rangle.$$

For the last step, by applying a concentration result on the following multiplier empirical process

$$\sup_{A\in\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)} \left\langle \frac{1}{n}\sum_{i=1}^{n}(Y^{(i)} - \mu_i)X^{(i)}, A\right\rangle = \sup_{A\in\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)} \frac{1}{n}\sum_{i=1}^{n}(Y^{(i)} - \mu_i)\left\langle X^{(i)}, A\right\rangle,$$

we can bound the quantity by the Gaussian width with large probability, up to some constant.

More specifically, denote

$$\omega^{(i)} = \Sigma^{-1/2}\text{vec}(X^{(i)})$$

then $\{\omega^{(i)}\}_{i=1}^n$ are i.i.d Gaussian random vectors and hence

$$\begin{aligned}
&\sup_{A\in\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)} \frac{1}{n}\sum_{i=1}^{n}(Y^{(i)} - \mu_i)\left\langle X^{(i)}, A\right\rangle \\
=\ &\sup_{F\in\text{vec}(\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1))} \frac{1}{n}\sum_{i=1}^{n}(Y^{(i)} - \mu_i)\text{vec}(X^{(i)})^\top F \\
=\ &\sup_{F\in\Sigma^{1/2}\cdot\text{vec}(\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1))} \frac{1}{n}\sum_{i=1}^{n}(Y^{(i)} - \mu_i)(\omega^{(i)})^\top F \\
\leq\ &c_5 M_Y^{1/q} \sup_{F\in\Sigma^{1/2}\cdot\text{vec}(\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1))} \frac{1}{n}\sum_{i=1}^{n}(\omega^{(i)})^\top F \\
=\ &c_5 M_Y^{1/q} \sup_{A\in\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)} \frac{1}{n}\sum_{i=1}^{n}\left\langle X^{(i)}, A\right\rangle \\
\leq\ &\frac{c_5 M_Y^{1/q} c_u}{\sqrt{n}} w_G^2[\Theta(t_0)\cap\mathbb{B}_{\text{F}}(1)],
\end{aligned}$$

where the first inequality follows from Theorem 1.9 of Mendelson (2014), and the second inequality holds in view of Lemma 11 of Raskutti and Yuan (2015), and both inequalities hold with probability greater than

$$1 - c_2\exp\left\{-c_3 w_G^2[\Theta(\theta' + \theta)\cap\mathbb{B}_{\text{F}}(1)]\right\} - c_4 n^{-(q/2-1)}\log^q n.$$

7.2. Proofs of results in Section 4

We now present the proofs for the two main examples $\Theta_2(r,s)$ and $\Theta_3(r)$. Our proofs involve: (i) proving that the projection properties hold for both sets of cones and (ii) finding an error bound for the Gaussian width $w_G[\Theta(t)\cap\mathbb{B}_{\mathrm{F}}(1)]$.

7.2.1. Proof of Theorem 4

First, it is straightforward to verify that $\{\Theta_2(r,s)\}$ is a superadditive family of symmetric cones. We then verify the contraction properties as stated by Lemma 2.

Proof [Proof of Lemma 2] We need to develop an error bound for $\|P_{\Theta_2(r_2,s_2)}(Z)-Z\|_{\mathrm{F}}$ for a general tensor $Z\in\Theta_2(r_0,s_0)$. Let $\tilde{Z}\in\mathbb{R}^{d_1\times d_2\times d_3}$ denote the tensor whose slices $\tilde{Z}_{\cdot\cdot j_3}$, $(j_3=1,\ldots,d_3)$ are the rank-r_2 approximation of the corresponding slices of Z. First, it follows from the contraction property of low rank matrix projector (see, e.g., Jain et al., 2016) that for all $1\le j_3\le d_3$, for any $Y_{\cdot\cdot j_3}$ such that $\mathrm{rank}(Y_{\cdot\cdot j_3})\le r_1$

$$\|\tilde{Z}_{\cdot\cdot j_3}-Z_{\cdot\cdot j_3}\|_{\mathrm{F}}\le\beta\|Y_{\cdot\cdot j_3}-Z_{\cdot\cdot j_3}\|_{\mathrm{F}}.$$

By summing over j_3 it follows that for any $Y\in\mathbb{R}^{d_1\times d_2\times d_3}$ where $\mathrm{rank}(Y_{\cdot\cdot j_3})\le r_1$ for all j_3

$$\|\tilde{Z}-Z\|_{\mathrm{F}}\le\beta\|Y-Z\|_{\mathrm{F}}.$$

The projection $P_{\Theta_2(r_2,s_2)}(Z)$ involves zeroing out the slices of $\tilde{Z}$ with the smallest magnitude. Let $v_{\tilde{Z}}:=\mathrm{vec}(\|\tilde{Z}_{\cdot\cdot 1}\|_{\mathrm{F}},\|\tilde{Z}_{\cdot\cdot 2}\|_{\mathrm{F}},\ldots,\|\tilde{Z}_{\cdot\cdot d_3}\|_{\mathrm{F}})$. As shown by Jain et al. (2014), for all Y where $v_Y=\mathrm{vec}(\|Y_{\cdot\cdot 1}\|_{\mathrm{F}},\|Y_{\cdot\cdot 2}\|_{\mathrm{F}},\ldots,\|Y_{\cdot\cdot d_3}\|_{\mathrm{F}})$ and $\|v_Y\|_{\ell_0}\le s_1$,

$$\|\tilde{P}_s(v_{\tilde{Z}})-v_{\tilde{Z}}\|_{\ell_2}\le\alpha\|v_Y-v_{\tilde{Z}}\|_{\ell_2}.$$

Therefore

$$\begin{aligned}\|P_{\Theta_2(r_2,s_2)}(Z)-\tilde{Z}\|_{\mathrm{F}} &\le \alpha\|Y-\tilde{Z}\|_{\mathrm{F}}\\ &\le \alpha(\|Y-Z\|_{\mathrm{F}}+\|\tilde{Z}-Z\|_{\mathrm{F}})\\ &\le (\alpha+\alpha\beta)\|Y-Z\|_{\mathrm{F}}.\end{aligned}$$

Hence using the triangle inequality:

$$\|P_{\Theta_2(r_2,s_2)}(Z)-Z\|_{\mathrm{F}}\le\|\tilde{Z}-Z\|_{\mathrm{F}}+\|P_{\Theta_2(r_2,s_2)}(Z)-\tilde{Z}\|_{\mathrm{F}}\le(\alpha+\beta+\alpha\beta)\cdot\|Y-Z\|_{\mathrm{F}},$$

which completes the proof. ■

Lemma 2 guarantees that $P_{\Theta_2(r,s)}$ satisfies the contractive projection property CPP(δ) with $\delta=3$. Hence, by setting $t_1=(r',s')$ and $t_0=(r'+r,s'+s)$, Theorem 3 directly implies the linear convergence rate result with statistical error of order

$$n^{-1/2}w_G[\Theta_2(r'+r,s'+s)\cap\mathbb{B}_{\mathrm{F}}(1)].$$

It remains to calibrate the Gaussian width. Recall the definition of the convex regularizer:

$$\mathcal{R}_1(A)=\sum_{j_3=1}^{d_3}\|A_{\cdot\cdot j_3}\|_*.$$

It is straightforward to show that

$$\Theta_2(r'+r,s'+s)\cap\mathbb{B}_{\mathrm{F}}(1)\subset\mathbb{B}_{\mathcal{R}_1}(\sqrt{(r'+r)(s'+s)}).$$

Then Lemma 5 of Raskutti and Yuan (2015) implies that

$$\begin{aligned}w_G[\Theta_2(r'+r,s'+s)\cap\mathbb{B}_{\mathrm{F}}(1)] &\le w_G[\mathbb{B}_{\mathcal{R}_1}(\sqrt{(r'+r)(s'+s)})]\\ &\le \sqrt{(s'+s)(r'+r)}\,w_G[\mathbb{B}_{\mathcal{R}_1}(1)]\\ &\le \sqrt{(s'+s)(r'+r)}\sqrt{6(d_1+d_2+\log d_3)}\end{aligned}$$

which completes the proof.

7.2.2. Proof of Theorem 5

Once again, it is straightforward to verify that $\{\Theta_3(r_1,r_2,r_3)\}$ is a superadditive family of symmetric cones. We now verify the contraction properties

Proof [Proof of Lemma 3] To develop an error bound for $\|\widehat{P}_{\Theta_3(r_1^{(2)},r_2^{(2)},r_3^{(2)})}(Z)-Z\|_{\mathrm{F}}$ for a general tensor $Z\in\Theta_3(r_1^{(0)},r_2^{(0)},r_3^{(0)})$, we introduce the following three tensors (recursively):

$$\begin{aligned}Z_{(1)} &:= (\mathcal{M}_1^{-1}\circ\bar{P}_{r_1}\circ\mathcal{M}_1)(Z)\\ Z_{(2)} &:= (\mathcal{M}_2^{-1}\circ\bar{P}_{r_2}\circ\mathcal{M}_2)(Z_{(1)})\\ Z_{(3)} &:= (\mathcal{M}_3^{-1}\circ\bar{P}_{r_3}\circ\mathcal{M}_3)(Z_{(2)}),\end{aligned}$$

where we recall that $\mathcal{M}_1(\cdot)$, $\mathcal{M}_2(\cdot)$ and $\mathcal{M}_3(\cdot)$ are the mode-1, mode-2 and mode-3 matricization operators. Therefore $\widehat{P}_{\Theta_3(r_1^{(2)},r_2^{(2)},r_3^{(2)})}(Z)=Z_{(3)}$ and:

$$\|\widehat{P}_{\Theta_3(r_1^{(2)},r_2^{(2)},r_3^{(2)})}(Z)-Z\|_{\mathrm{F}}\le\|\widehat{P}_{\Theta_3(r_1^{(2)},r_2^{(2)},r_3^{(2)})}(Z)-Z_{(2)}\|_{\mathrm{F}}+\|Z_{(2)}-Z_{(1)}\|_{\mathrm{F}}+\|Z_{(1)}-Z\|_{\mathrm{F}}.$$

Next note that

$$\|Z_{(1)}-Z\|_{\mathrm{F}}=\|(\mathcal{M}_1^{-1}\circ\bar{P}_{r_1}\circ\mathcal{M}_1)(Z)-Z\|_{\mathrm{F}}=\|\bar{P}_{r_1}(\mathcal{M}_1(Z))-\mathcal{M}_1(Z)\|_{\mathrm{F}}.$$

As shown by Jain et al. (2016), for any Y such that $r_1(Y)\le r_1^{(1)}$,

$$\|Z_{(1)}-Z\|_{\mathrm{F}}\le\beta_1\|Y-Z\|_{\mathrm{F}}.$$

Using a similar argument and the triangle inequality

$$\|Z_{(2)}-Z_{(1)}\|_{\mathrm{F}}\le\beta_2\|Y-Z_{(1)}\|_{\mathrm{F}}\le\beta_2(\|Y-Z\|_{\mathrm{F}}+\|Z_{(1)}-Z\|_{\mathrm{F}})\le(\beta_2+\beta_1\beta_2)\|Y-Z\|_{\mathrm{F}}.$$

Furthermore,

$$\begin{aligned}\|Z_{(3)}-Z_{(2)}\|_{\mathrm{F}} &\le \beta_3\|Y-Z_{(2)}\|_{\mathrm{F}}\\ &\le \beta_3(\|Y-Z\|_{\mathrm{F}}+\|Z_{(2)}-Z\|_{\mathrm{F}})\\ &\le \beta_3(1+\beta_1+\beta_2+\beta_1\beta_2)\|Y-Z\|_{\mathrm{F}}.\end{aligned}$$

Therefore for all $Y \in \Theta_3(r_1^{(1)}, r_2^{(1)}, r_3^{(1)})$

$$\begin{aligned}\|\widehat{P}_{\Theta_3(r_1^{(2)},r_2^{(2)},r_3^{(2)})}(Z) - Z\|_{\mathrm{F}} &= \|\widehat{P}_{\Theta_3(r_1^{(2)},r_2^{(2)},r_3^{(2)})}(Z) - Z_{(2)}\|_{\mathrm{F}} + \|Z_{(2)} - Z_{(1)}\|_{\mathrm{F}} + \|Z_{(1)} - Z\|_{\mathrm{F}} \\ &\leq [(\beta_1+1)(\beta_2+1)(\beta_3+1) - 1]\|Y - Z\|_{\mathrm{F}}.\end{aligned}$$

■

Lemma 2 guarantees the approximate projection $\widehat{P}_{\Theta_3(r_1,r_2,r_3)}$ fulfills the contractive projection property CPP(δ) with $\delta = 7$. And hence via setting $t_1 = (r_1', r_2', r_3')$ and $t_0 = (r_1' + r_1, r_2' + r_2, r_3' + r_3)$, Theorem 3 directly implies the linear convergence rate result with statistical error of order $n^{-1/2} w_G[\Theta_3(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)]$. To upper bound the Gaussian width, we define the following nuclear norms:

$$\mathcal{R}_{(i)}(A) = \|\mathcal{M}_i(A)\|_*,$$

where $1 \leq i \leq 3$ and $\|.\|_*$ is the standard nuclear norm. Then it clearly follows that

$$\Theta_3(t_0) \cap \mathbb{B}_{\mathrm{F}}(1) \subset \cap_{i=1}^3 \mathbb{B}_{\mathcal{R}_{(i)}}(\sqrt{r_i' + r_i}).$$

Lemma 5 in Raskutti and Yuan (2015) then implies that

$$\begin{aligned} w_G[\Theta_3(t_0) \cap \mathbb{B}_{\mathrm{F}}(1)] &\leq w_G[\cap_i \mathbb{B}_{\mathcal{R}_{(i)}}(\sqrt{r_i' + r_i})] \\ &\leq \min_i w_G[\mathbb{B}_{\mathcal{R}_{(i)}}(\sqrt{r_i' + r_i})] \\ &\leq \min_i \sqrt{r_i' + r_i}\, w_G[\mathbb{B}_{\mathcal{R}_{(i)}}(1)] \\ &\leq \sqrt{6 \min\{(r_1'+r_1)(d_1 + d_2 d_3), (r_2'+r_2)(d_2 + d_1 d_3), (r_3'+r_3)(d_3 + d_1 d_2)\}}\end{aligned}$$

which completes the proof.

Acknowledgement. Han Chen and Ming Yuan was supported in part by NSF FRG Grant DMS-1265202, and NIH Grant 1-U54AI117924-01. Ming Yuan was also supported by NSF Grant DMS-1803450 Garvesh Raskutti was supported in part by NSF Grants DMS-1407028, DMS-1811767, NIH Grant R01 GM131381-01, ARO Grant W911NF-17-1-0357 and NGA Grant HM0476-17-1-2003.

References

Arindam Banerjee, Sheng Chen, Farideh Fazayeli, and Vidyashankar Sivakumar. Estimation with norm regularization. Technical Report arXiv:1505.02294, November 2015.

P. Bartlett, O. Bousquet, and S. Mendelson. Local rademacher complexities. *The Annals of Statistics*, 33(4):1497–1537, 2005.

S. Basu and G. Michailidis. Regularized estimation in sparse high-dimensional time series models. *Annals of Statistics*, 43(4):1535–1567, 2015.

P. Buhlmann and S. van de Geer. *Statistical for High-Dimensional Data*. Springer Series in Statistics. Springer, New York, 2011.

V. Chandrasekaran, B. Recht, P. A. Parrilo, and A. S. Willsky. The convex algebraic geometry of linear inverse problems. *Foundations of Computational Mathematics*, 12: 805–849, 2012.

S. Cohen and M. Collins. Tensor decomposition for fast parsing with latent-variable pcfgs. In *Advances in Neural Information Processing Systems*, 2012.

J. Fan and R.Z. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of American Statistical Association*, (32):407–499, 2001.

S. Gandy, B. Recht, and I. Yamada. Tensor completion and low-n rank tensor recovery via convex optimization. *Inverse Problems*, 27, 2011.

Y. Gordon. On milmans inequality and random subspaces which escape through a mesh in $\mathbb{R}^n$. *Geometric aspects of functional analysis, Israel Seminar 1986-87, Lecture Notes*, 1317:84–106, 1988.

T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Monographs on Statistics and Applied Probability 143. CRC Press, New York, 2015.

P. Jain, A. Tewari, A. Nanopoulos, and P. Kar. On iterative hard thresholding methods for high-dimensional m-estimation. In *Proceedings of NIPS*, 2014.

P. Jain, N. Rao, and I. Dhillon. Structured sparse regression via greedy hard-thresholding. Technical Report arXiv:1602.06042, February 2016.

T. G. Koldar and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51:455–500, 2009.

N. Li and B. Li. Tensor completion for on-board compression of hyperspectral images. In *17th IEEE International Conference on Image Processing (ICIP)*, pages 517–520, 2010.

P. Loh and M. J. Wainwright. Regularized m-estimators with nonconvexity: Statistical and algorithmic theory for local optima. *Journal of Machine Learning Research*, 15:559-616, 2015.

Shahar Mendelson. Upper bounds on product and multiplier empirical processes. Technical Report arXiv:1410.8003, Technion, I.I.T, 2014.

C. Mu, B. Huang, J. Wright, and D. Goldfarb. Square deal: Lower bounds and improved relaxations for tensor recovery. In *International Conference on Machine Learning*, 2014.

S. Negahban, P. Ravikumar, M. J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Statistical Science*, 27(4):538–557, 2012.

G. Raskutti and M. Yuan. Convex regularization for high-dimensional tensor regression. Technical Report arXiv:1512.01215, University of Wisconsin-Madison, December 2015.

G. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, 1970.

O. Semerci, N. Hao, M. Kilmer, and E. Miller. Tensor based formulation and nuclear norm regularizatin for multienergy computed tomography. *IEEE Transactions on Image Processing*, 23:1678–1693, 2014.

N.D. Sidiropoulos and N. Nion. Tensor algebra and multi-dimensional harmonic retrieval in signal processing for mimo radar. *IEEE Transactions on Signal Processing*, 58:5693–5705, 2010.

R. Tomioka, T. Suzuki, K. Hayashi, and H. Kashima. Statistical performance of convex tensor decomposition. In *Advances in Neural Information Processing Systems*, pages 972–980, 2013.

M. Yuan and C-H. Zhang. On tensor completion via nuclear norm minimization. *Foundation of Computational Mathematics*, to appear, 2014.

Journal of Machine Learning Research 20 (2019) 1-22 Submitted 2/17; Revised 1/19; Published 1/19

scikit-multilearn: A scikit-based Python environment for performing multi-label classification

Piotr Szymański PIOTR.SZYMANSKI@{PWR.EDU.PL,ILLIMITES.EDU.PL}
Department of Computational Intelligence
Wrocław University of Science and Technology
Wrocław, Poland

illimites foundation
Wrocław, Poland

Tomasz Kajdanowicz TOMASZ.KAJDANOWICZ@PWR.EDU.PL
Department of Computational Intelligence
Wrocław University of Science and Technology
Wrocław, Poland

Editor: Alexandre Gramfort

Abstract

The scikit-multilearn is a Python library for performing multi-label classification. It is compatible with the scikit-learn and scipy ecosystems and uses sparse matrices for all internal operations; provides native Python implementations of popular multi-label classification methods alongside a novel framework for label space partitioning and division and includes modern algorithm adaptation methods, network-based label space division approaches, which extracts label dependency information and multi-label embedding classifiers. The library provides Python wrapped access to the extensive multi-label method stack from Java libraries and makes it possible to extend deep learning single-label methods for multi-label tasks. The library allows multi-label stratification and data set management. The implementation is more efficient in problem transformation than other established libraries, has good test coverage and follows PEP8. Source code and documentation can be downloaded from `http://scikit.ml` and also via `pip`. The project is BSD-licensed.

Keywords: Python, multi-label classification, label-space clustering, multi-label embedding, multi-label stratification

1. Introduction

The Python language with its machine learning library stack has grown to become one of the leading technologies of building models for the industry and developing new methods for the researchers. While the Python community boasts with the excellent culture of development, well-defined API traditions and well-performing implementations of methods from most machine learning areas it did not have a high-quality solution for multi-label classification.

In this paper we introduce scikit-multilearn, a well-tested, multi-platform, Python 3 compatible, BSD-licensed library with bleeding edge approaches for multi-label problems, which is actively developed and has a growing user base. We also show that it is faster than its competition in other languages. It is also entirely compatible with the scientific/machine learning ecosystem of well-known Python libraries which allows it to fill the niche of the missing multi-label classification solution, while also benefits from complementary solutions present in the scikit-learn community, while providing it with efficient implementations of missing classification methods, data stratification, dataset access and manipulation.

We start with a summary of what multi-label classification is in Section 2. Followed by Section 3 where we present related work and a description of the machine learning ecosystem that scikit-multilearn fits in. In Section 4 we describe what scikit-multilearn brings to the community. We evaluate how scikit-multilearn performs in comparison to libraries implemented in other languages in Section 5 and present conclusions and future ideas in Section 6.

2. Multi-label classification

Multi-label classification deals with the problem of assigning a subset of available labels to a given observation. Such problems appear in multiple domains - article and website classification, multimedia annotation, music categorization, the discovery of genomics functionalities and are performed either by transforming a problem into a single/multi-class classification problem or by adapting a single/multi-class method to take multi-label information into account. An excellent introduction to the field has been provided by Tsoumakas et al. (2009).

Madjarov et al. (2012) divide approaches multi-label classification into three groups: method adaptation, problem transformation and ensembles thereof. The first idea is to adapt single-label methods to multi-label classification by modifying label assigning fragments in a single-label method. An example of this is introducing a multi-label version of a decision function in Decision Trees (ex. Kocev et al., 2013).

Problem transformation concentrates on converting the multi-label problem to one or more single-label problems. The most prominent examples include classifying each label separately such as Binary Relevance or Classifier Chains (Read et al., 2009) and treating each label combination as a separate class in one multi-class problem as in the Label Powerset case.

Both of these approaches suffer not only from standard problems of machine learning such as over- and under-fitting but also from the issue of label imbalance (Sun et al., 2009) or numerical anomalies related to label ordering when Bayesian approaches are used for taking correlations into account. Ensemble methods aim to correct this by learning multiple classifiers trained on label subspaces (ex. RAkEL by Tsoumakas et al., 2011a), observation subsets with pruning and replacement, or analyzing various label orderings in chains (ex. Probabilistic Classifier Chains by Dembczynski et al., 2010).

Multi-label embedding techniques emerged as a response the need to cope with a large label space; these include label space dimensionality reduction techniques that turned Most multi-label embedding methods turn multi-label classification into multivariate regression problem followed by a rule-based or classifier-based correction step. Embedding methods also vary by the principle of how the embedding is performed, these include: Principle Label Space Transformation (Tai and Lin, 2012) based on Principal Component Analysis; Conditional Principal Label Space Transformation (Chen and Lin, 2012) with Canonical Com-

©2019 Piotr Szymański and Tomasz Kajdanowicz.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v20/17-100.html`.

ponent Analysis; Feature-aware Implicit Label Space Encoding (Lin et al., 2014) based on matrix decompositions; SLEEC (Bhatia et al., 2015) which uses k-means clustering and per cluster embedding; CLEMS (Huang and Lin, 2017) which performs Multi-Dimensional Scaling (see Kruskal, 1964) with extra multi-label quality measure optimization; and LNEMLC (Szymański et al., 2018) which embeds label networks using network embeddings such as LINE or node2vec. Most of these use linear, ridge or random forest regressors to predict embeddings for unseen samples and a variant of kNN to classify them with labels. CLEMS and LNEMLC are among the best performing multi-label embeddings at this time.

Multi-label advances in recent years also include developments in a sub-area called extreme multi-label where new methods emerged such as deep-learning based domain-specific approaches for: image classification frameworks (Zhao et al., 2015; Wang et al., 2016; Wei et al., 2016) or text (Liu et al., 2017; Yen et al., 2017). The field also includes tree-based (Prabhu and Varma, 2014) or embedding-based (Bhatia et al., 2015) approaches. While extreme multi-label is an interesting task it differs strongly from classical multi-label classification in performance expectations, benchmark datasets and quality measures and falls beyond the main focus of scikit-multilearn but its methods can be used to classify these scale of problems.

3. Related work

As with every applied science, research requires environments for performing experiments. The most prominent multi-label classification stack to date (regarding method count and popularity) is implemented in Java: MULAN (Tsoumakas et al., 2011b) and MEKA (Read et al., 2016). Both depend heavily on the famous WEKA library (Hall et al., 2009) which implements a plethora of classification and regression methods for single-label classification. MULAN and MEKA are large multi-purpose libraries which support not only multi-label classification tasks but also regression and multi-instance/multi-output tasks.

Python's scientific ecosystem's philosophy (Oliphant, 2007) is a different one, instead of having few large multi-purpose libraries, it provides a foundation of libraries that deliver most important approaches and a network of smaller and more specialized libraries that interact thanks to a well-defined API, coding and documentation standards set out for common machine learning tasks such as prediction or data transformation. With numpy (Oliphant, 2006), scipy (Jones et al., 2001) as numerical operation and data structure foundations. The machine-learning foundation provided by scikit-learn (Pedregosa et al., 2011) follows a different approach than WEKA as it does not aim at being a large mono-library where all functionality is gathered. Instead, it concentrates on implementing most cited well-established methods alongside a stable API design.

The rudimentary multi-label methods provided by scikit-learn do not support sparse representations of label matrices apart from the multi output classifier: algorithm adaptation approaches such as k nearest neighbors, CART trees and random forests and perceptrons; problem transformation methods Binary Relevance, Classifier Chains and one-vs-rest or one-vs-all usage of multi-class classifiers such as Support Vector Machines (Hearst et al., 1998). However all of these are the most rudimentary variants of listed methods and have been superseded by more modern approaches. The kNN approach from scikit-learn has been improved by ML-kNN (Zhang and Zhou, 2007), deep learning is used much more

often than perceptrons, and new tree-based classifiers like fastXML (Prabhu and Varma, 2014) outperform traditional random forests of multi-label trees. One-vs-all/rest multi-class classifiers can now be replaced with inherently multi-label variants of the methods, ex. SVM classification scheme has been enhanced by multi-label SVM approaches such as MLTSVM (Chen et al., 2016). Binary Relevance provides worse performance than data-driven label space division (Szymański et al., 2016) or stacking (Montañes et al., 2014). Classifier chains have been extended in multiple ways to overcome artifacts related to sample distribution changes between label orderings either by learning shorter chains via data-driven label space divisions or chain sampling methods (PCC by Dembczynski et al., 2010), (MCC by Read et al., 2013). However scikit-learn's classifiers are extremely useful when used as a base for more comprehensive methods in scikit-mutlilearn such as label space partitioning ensembles or multi-label embedding.

The scikit-learn project also forms a hub of a large network of complementary libraries for more specific tasks, new techniques or emergent sub-fields. Such libraries include among many others scikit-multiflow and imbalanced-learn.

The scikit-multiflow library (Montiel et al., 2018) concerns multi-label streams, which differs from the traditional multi-label setting where a method is trained once to predict data - in multi-label streams the methods are retrained or adopted as new samples arrive. Even though scikit-multiflow provides reimplementations selected methods of scikit-learn, the authors note that scikit-learn's implementations should be applied while using the library. Authors claim that the provided implementations are less optimal (kNN) or masks over the original scikit-learn code for internal purposes. more Modern version of classifier chains (PCC, MCC) and stream classifiers are the core contribution of the library. They are optimized towards partial updating as new evidence arrives. The library is well unit-tested and should be used for multi-label stream tasks which scikit-multilearn does not handle out of the box. However, scikit-multilearn provides more vibrant and various offering of state of the art methods optimized for the classic multi-label problem formulation.

The imbalanced-learn library (Lemaître et al., 2017) is dedicated to overcoming class imbalance problems in single-label tasks by a variety of under/over-sampling strategies. As many problem transformation approaches transform the problem to a multi-class problem imbalanced-learn can be used to improve learning capabilities of scikit-multilearn classifiers by fitting them to resampled data.

Domain and sub-field related deep neural networks with multi-label classification support are also available such as magpie[1], CNN-RNN for images or a variety of different models for biological data[2] for text classification.

Multi-label classification problems can also be addressed in general purse deep learning libraries such as Tensorflow (Abadi et al., 2015) or Keras (Chollet et al., 2015). However, deep neural networks often do not demonstrate an appropriate level of robustness and require significant differences in architecture to accommodate a concrete problem. This introduces an additional level of complication while applying deep learning to multi-label problems. While these libraries are instrumental in their areas, scikit-multilearn is dedicated to providing domain-independent solutions as efficient per domain specialization is rarely possible in one unified code base. However, scikit-multilearn does provide support for using

1. `https://github.com/inspirehep/magpie`
2. `https://qdata.github.io/deep4biomed-web/`

	Java		Python		R	Matlab Octave
	MULAN	MEKA	scikit-learn	scikit-multilearn	util.ml	MLC_toolbox
Algorithm-adaptation methods						
Multi-label SVM				✓		✓
Trees	✓	✓	✓			✓
Nearest Neighbors	✓	✓		✓		✓
Neural	✓	✓	✓	✓		✓
Problem transformation approaches						
To single-class	✓	✓	✓	✓	✓	✓
To multi-class	✓	✓		✓		✓
To deep learning models				✓		
Ensemble approaches						
Stacking	✓	✓	✓	✓	✓	✓
Label space partitioning	✓	✓		✓		✓
Overlapping models	✓	✓		✓		✓
Embeddings				✓		✓
Related tasks						
Multi-label stratification	✓	✓		✓		
Quality measures	✓	✓	✓	P	✓	✓
Multi-label dataset management	✓	✓	P	✓	✓	
Multi-label regression	✓	✓	✓			
Multi-output learning	✓					
Sparse label space representation			P	✓		
Multi-label ARFF support	✓	✓		✓	✓	
Project quality						
End-to-end use cases			✓	✓		
Developer documentation	✓		✓	✓		
API documentation	✓	✓	✓	✓	✓	
Unit-tests	✓	✓	✓	✓	✓	
Multi-platform	✓	✓	✓	✓	✓	P
Continuous integration on all platforms			✓	✓		

Table 1: Comparison of multi-label classification libraries and their features. Letter P denotes partial functionality: scikit-learn can generate multi-label datasets, but does not offer methods to lead them and manipulate them; sparse support is implemented only for Binary Relevance (MultiOutputClassifier). MLC toolbox offers some of its classifiers Windows-only. Output format from scikit-multilearn's classifiers is compatible with scikit-learn's quality measures.

Keras-compatible models for multi-label classification independent of the backend used by Keras (i.e. Tensorflow, CNTK) or PyTorch (via the skorch[3] library).

The scikit-multilearn project fits in the Python community similarly as to how MEKA or MULAN fit to the Java machine learning projects network, however, it is focused on multi-label classification problems to provide state of the art approaches and efficient implementation of more advanced methods.

Recently multi-label classification libraries were made available in R (R Core Team, 2013) and Matlab/Octave (Eaton et al., 2017): utiml (Rivolli and de Carvalho, 2018) R library built on top of the dataset management library mldr (Charte and Charte, 2015) and offers a larger variety of available methods than other R libraries and a more welcoming API; MLC Toolbox (Kimura et al., 2017) provides multi-label methods for Matlab/Octave communities.

Table 1 provides a general point of view on the functionality offered by multi-label classification methods in Java, Python, R and Matlab/Octave.

4. The scikit-multilearn library

The proposed library follows the idea of building a multi-label classification library on top of an existing classification solution. In the case of Python, the obvious choice for a base library is the scipy stack with scikit-learn. The concept of scikit-learn compatible projects (scikits in short, not to mistake with the old scipy notion of scikits), has been present for several years in many formats, most prominently exemplified in scikit-contrib. We follow the ideas of these communities, the scikit-learn API principles and licensing.

The primary goal of scikit-multilearn is to provide an efficient Python implementation of as many multi-label classification algorithms as possible both to the open source community and commercial users of the Python data science stack. With such focus in mind, we concentrate on delivering a library dedicated to provide domain-independent solutions for solving multi-label classification problems, allowing other toolkits to excel in their areas such as: scikit-multiflow when it comes to multi-label streams or imbalanced-learn when it comes to handling imbalance multi-class data or many deep learning approaches described in the previous section.

Extensions of scikit-learn. The scikit-multilearn library extends the multi-label classification offering of scikit-learn by:

- extending the family of classifiers available to the Python community with more advanced algorithm adaptation and problem transformation (label space division) approaches
- implementing multi-label embedding-based classification methods
- providing additional features too specific for scikit-learn such as the general framework for classification based on label space division or the wrapper for MEKA,
- implementing non-classification tools for multi-label classification, including multi-label dataset manipulation from ARFF, loading and saving sparse representations of

3. `https://github.com/dnouri/skorch`

these datasets and the only Python implementations of multi-label data stratification methods (see work by (Sechidis et al., 2011), (Szymański and Kajdanowicz, 2017))

Advanced Multi-label Adapted Algorithms. Within scikit-multilearn we provide implementations of recent multi-label algorithm adaptation methods, that do not meet the selectiveness criteria of scikit-learn due to their novelty. These include:

- ML-ARAM: the hierarchical multi-label adaptive resonance associative map (Benites and Sapozhnikova, 2015)
- BR-kNN and ML-kNN: Binary Relevance kNN (Spyromitros et al., 2008) and multi-label kNN (Zhang and Zhou, 2007) - their scikit-multilearn implementations are built on top of scikit-learn's NearestNeighbors multiclass classifier,
- MLTSVM: the multi-label twin support vector machine (Chen et al., 2016) which does not require a training quadratic number of SVM classifiers.

Problem Transformation with Label Space Division. One of the significant contributions of the library is a framework for performing ensemble classification with different label space division strategies. The library provides ways to divide the label space based on:

- clustering of the label matrix representations, using any of the scikit-learn compatible clusterers (i.e. classes that inherit scikit-learn's `ClusterMixin` base class). The clusterer is executed on a transposed label assignment matrix and detected clustering is used as the label space division for the ensemble method. An example of this approach is the k-means clustering of labels which has been applied by many approaches (Tsoumakas et al., 2008; Yu et al., 2017)).
- communities detected in a network which embeds information about relations between labels. In scikit-multilearn there is a general API for defining graph generators based on the label matrix, such graphs can exploit relations based on co-occurrence, correlation or any other pairwise measure defined over labels. The library supports community detection methods based on three popular graph/network packages in Python: (NetworkX, BSD by Hagberg et al., 2008)), (igraph, GPL by Csardi and Nepusz, 2006), (graphtool, GPL by Peixoto, 2014). An example of this approach is the label space division based on label co-occurrence graphs (Szymański et al., 2016).
- random division, present in the random k-label sets (RAkEL) approaches
- predefined fixed division obtained from expert knowledge

In scikit-multilearn a partitioning and a voting majority ensemble approach are provided, in both a sub-classifier is trained on every label subset from a label space partition. In the partitioning variant, the predicted results are the union of results from each of the classifiers. In the voting ensemble, the label is assigned if the majority of classifiers trained on a label subspace which contain the label have assigned it to a given sample.

Multi-Label Embeddings. Another significant contribution of the library is a framework for multi-label classification using multi-label embeddings and scikit-multilearn is the first Python library to provide a general multi-label embedding scheme with an embedding regressor and embedding-based classifier. Currently two state of the art embedding approaches are provided Cost-Sensitive Label Embedding with Multidimensional Scaling (CLEMS)(Huang and Lin, 2017), Label Network Embeddings for Multi-Label Classification (LNEMLC)(Szymański et al., 2018) alongside a general embedder that can use any scikit-learn manifold learning or dimension reduction to embed the output space.

Compatibility. The scikit-multilearn project aims to be compatible with the Python data science stack. It follows the scikit-learn API and requirements specified in check_estimator code. Multi-label classifiers in scikit-multilearn inherit scikit-learn's `BaseEstimator` and `ClassifierMixin` classes. They can be thus easily incorporated into scikit-learn pipelines and cross-validations, evaluation measures, and feature space transformators. It is easy to use scikit-learn's extensive feature-space manipulation methods, single-label classifiers and classification evaluation functions with scikit-multilearn.

BSD licensing. As innovations in machine learning happen in both academia and companies, it is important to create a library that allows both communities to grow and profit from common work. Thus, scikit-multilearn is released under the BSD license to permit the for-profit ecosystem to use the library while opening the possibility of sharing development and maintenance tasks. While most of scikit-multilearn's functionality is available via BSD licensed libraries, the package also allows using GPL-licensed libraries to be used for label space division such as igraph or graphtool. Importing from `skmultilearn.cluster.igraph` or `skmultilearn.cluster.graphtool` will require your code to submit to GPL licensing requirements.

Access to the Java stack. The library provides scikit-compatible wrapper to reference libraries such as MEKA, MULAN and WEKA through MEKA when a need arises to use methods that have not yet been implemented in Python natively. Using these libraries via scikit-multilearn wrapper does not induce a licensing (GPL-BSD) conflict. Support for MULAN is limited to methods available via MEKA's `meka.classifiers.multilabel.MULAN` classifier[4].

Using Deep Learning models. Scikit-multilearn provides a wrapper that allows using any Keras (Chollet et al., 2015) compatible backend such as Tensorflow (Abadi et al., 2015) CNTK (Seide and Agarwal, 2016) or PyTorch (Paszke et al., 2017), to provide a single-class or multi-class model that can be used to solve multi-label problems through problem transformation approaches.

Sparsity. Multi-label classification problems often do not provide cases for all possible label combinations - in most benchmark sets there are less than 5% labels per row on average. The output space is thus very sparse. As opposed to MULAN and MEKA, scikit-multilearn operates on sparse matrices internally yielding a large memory boost as displayed in Figure 1.

4. `http://meka.sourceforge.net/api-1.9/meka/classifiers/multilabel/MULAN.html#setMethod-java.lang.String-`

Using scikit-multilearn. The steps for using scikit-multilearn are very simple: loading the data, selecting the method and performing classification. The library supports loading matrices from all well-established formats thanks to scipy and operates on scipy/numpy representations internally, converting to dense matrices or lists of lists only if required by a scikit-base classifier, if one is employed in the problem transformation scenario.

Other output types and subproblems. Scikit-multilearn does not support multi-label regression or multi-output prediction at the moment. Multi-label regression is considered for future releases, while multi-output approaches will not be supported due to scikit-multilearn's concentration on efficient sparse matrix usage throughout the codebase. Extreme Multi-label Classification is not the main focus of scikit-multilearn but both the label network embedding methods and label space partitioning methods are capable of handling extreme multi-label datasets. Keras-based deep learning models can also be deployed with the label partitioning strategy to solve extreme multi-label problems.

Structure of scikit-multilearn. Scikit-multilearn provides three subpackages following the established categorization of multi-label methods: method adaptation approach (in `skmultilearn.adapt`), problem transformation approach (in `skmultilearn.pt`) and ensembles of the two (in `skmultilearn.ensemble`). The label space division methods and label relationship graph builders are present in `skmultilearn.cluster`. To use multi-label-adapted version of a single-label method one only needs to import it and instantiate an object of the class. A problem transformation approach takes a scikit-learn compatible base classifier and optional information whether the base classifier supports sparse input. Ensemble methods require a classifier and relevant method's parameters. The classifier is trained using the fit method, which takes the training input and output matrices, and classification is performed using the predict method on test observations - just as in scikit-learn. The MEKA wrapper provides a scikit-learn compatible classifier class in `skmultilearn.meka`. Dataset download, loading and manipulation tools are available in `skmultilearn.dataset`. Multi-label data stratification methods and dataset fold division quality measures are available in `skmultilearn.model_selection`. Multi-label embedding approaches are located in `skmultilearn.embeddings`.

4.1. Documentation and Availability

Scikit-multilearn is hosted on GitHub and managed via the fork and pull-request paradigm. Both user and developer documentation is available at `http://scikit.ml/`. After 4 years of development, in July 2018, the library has reached a stable 0.1.0 release. The newest version - 0.2.0 - was released in December 2018. The code is released under BSD licence using GitHub and releases are available via PyPI. Implementation follows PEP8 and is maintained with a high level of test coverage (82% for the 0.2.0 release). The development undergoes continuous integration on Windows, Ubuntu Linux and Mac OS X, with Python 2.7 and Python 3.3. At the moment of the 0.2.0 release scikit-multilearn has been starred 242 times and forked 72 times. Development is managed on the GitHub repository `scikit-multilearn`[5] and is accompanied with communication and discussion in the Slack channel[6]. All commits undergo continuous testing on Windows, Ubuntu and Mac OS X, both under Python 2.7 and 3.3. The library also has its tag[7] on StackOverflow.

5. `https://github.com/scikit-multilearn/scikit-multilearn`

5. Benchmark

We found that scikit-multilearn is faster than its competition. We have tested MEKA, MULAN and scikit-multilearn on 12 well-cited benchmark multi-label classification datasets using two comparison scenarios: Binary Relevance and Label Powerset. The two selected problem transformation approaches are widely used both in regular classification tasks and as the base for more sophisticated methods. We did not test algorithm adaptation methods as there are no algorithm adaptation methods present in all three libraries. We use these methods to illustrate two aspects of the classification performance of the libraries: the cost of using many classifiers with splitting operations performed on the label space matrix and the cost of using a single classifier which requires to access all label combinations to perform the transformation. To minimize the impact of base classifiers, we have decided to use a fast Random Forest base classifier with 10 trees. As Octave does not provide Matlab's random forest implementation, we used the one provided by the shogun toolbox (Sonnenburg et al., 2017). We have checked the classification quality and did not find significant differences between Hamming Loss, Jaccard and Accuracy scores between the outputs. The benchmark is performed with scikit-multilearn 0.1.0, MEKA 1.9.2, MULAN 1.5.0, scikit-learn 0.19.2, Octave 4.2.2, shogun 6.1.3, MLC Toolbox for Matlab/Octave code from GitHub master commit hash `e798779`, R 3.4.1, utiml 0.1.2. As utiml does not provide a Label Powerset implementation, we do not evaluate it in this subexperiment.

Figure 1 presents the time and memory required to perform classification, including loading the ARFF dataset (or Matlab export in case of the toolbox) and measuring errors, i.e. a complete classification use case scenario. As different datasets require a different amount of time and memory we decided to normalize the charts. The results on the chart are normalized for each dataset separately. To normalize we calculated the median of every library's time or memory performance and selected the highest of the medians in the dataset as the normalization point, i.e. 100% is the worst median performance. We present the best, median, and worst performance of each library per dataset, normalized performance with the worst median performance on that dataset. The closer the library is to point 0, the better it performed, thus the smaller area inside the library curve, the better.

All the libraries were forced to use a single core using the `taskset` command to minimize parallelization effects on the comparison. Time and memory results were obtained using the `time -v` command and represent User time, and Maximum resident set size respectively. All results taken into consideration reported that 100% of their CPU core had been assigned to the process which performed the classification scenario.

We notice that in both classification schemes scikit-multilearn always uses less or, in a few edge cases the same amount of, memory than MEKA or MULAN due to its sparse matrix support. In most cases scikit-multilearn also operates faster than MEKA, MULAN.

When it comes to label space division approaches modeled in the experiment by Binary Relevance, scikit-multilearn is the most efficient choice on every data set. With Label Pow-

6. `https://scikit-ml.slack.com`
7. `https://stackoverflow.com/tags/scikit-multilearn/info`

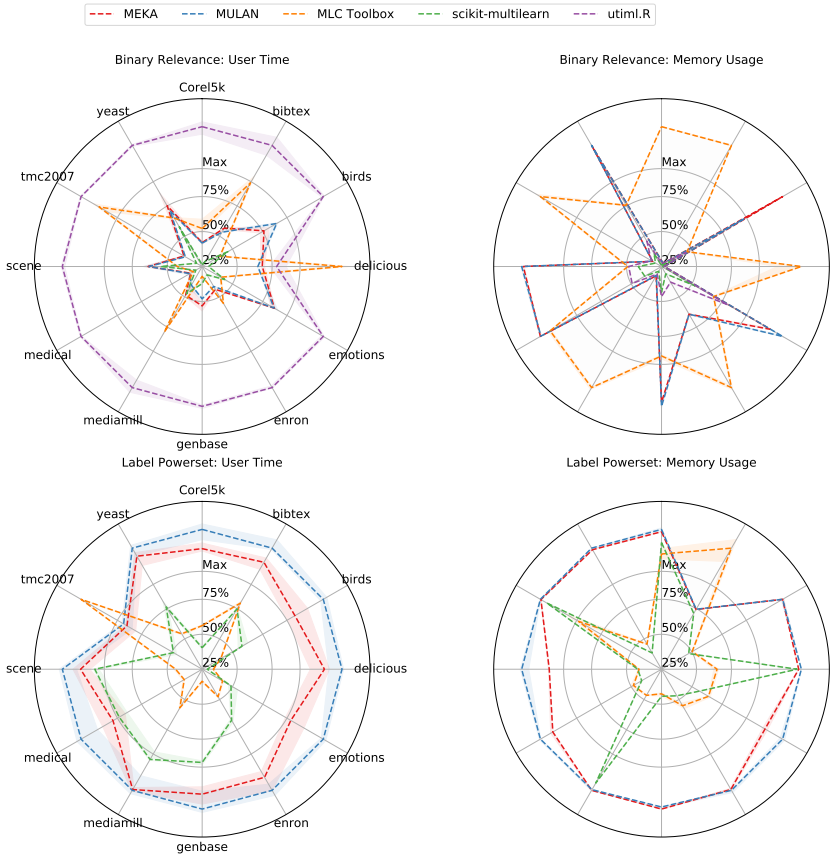


Figure 1: User running time (s) and memory usage ranks of utiml, MLC Toolbox, scikit-multilearn, MEKA and MULAN, with RandomForest, and two different multi-label classifiers.

erset multi-class transformation it is in most cases more efficient than MEKA and MULAN and outperforms the MLC Toolbox on datasets with larger numbers of label combinations while on smaller data sets the MLC Toolbox tends to be more efficient.

We have also compared Binary Relevance from scikit-learn to Binary Relevance from scikit-multilearn. The only difference between the two implementations is the fact that

scikit-learn does not convert the output matrix to a Compressed Sparse Column matrix[8] while scikit-multilearn does. The observed results were that scikit-learn's implementation was always slightly faster than scikit-multilearn's, while it always used a slightly larger amount of memory, both changes can be attributed to a difference between formats used for the output space. We, therefore, do not report this comparison in extensive detail.

As there is no statistical procedure that allows performing a non-parametric repeated measure tests on multiple measurements per test, we decided to compare the worst case time/memory usage of scikit-multilearn against the best case MEKA/MULAN/utiml/MLC

8. `https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.csc_matrix.html`

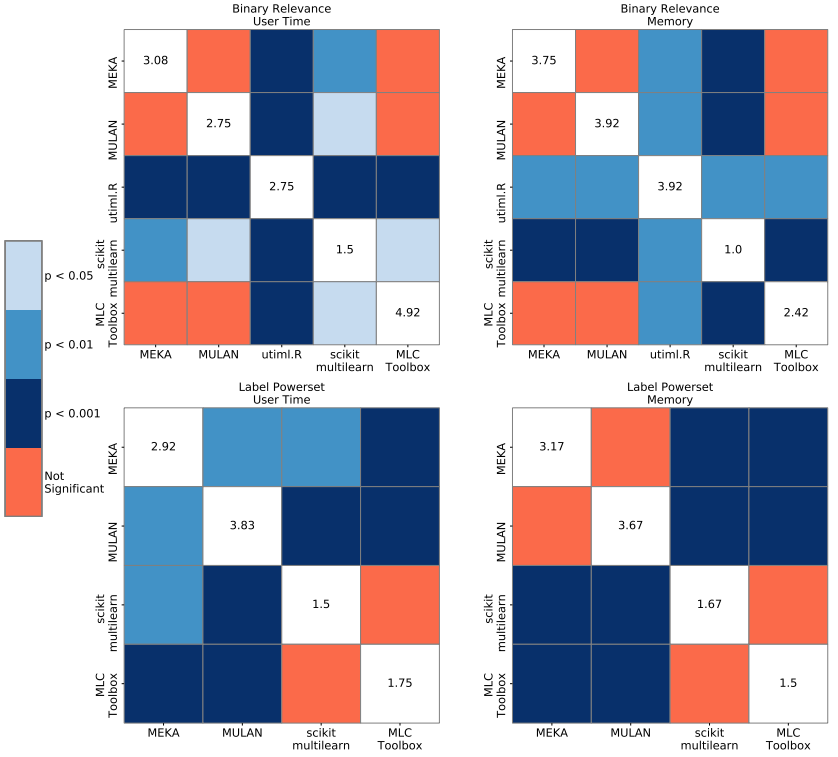


Figure 2: User running time (s) and memory usage ranks of utiml, MLC Toolbox, scikit-multilearn, MEKA and MULAN, with RandomForest, with RandomForest, and different multi-label classifiers. There are mean ranks on diagonal.

Toolbox performance. We used the Quade test with Bergmann-Hommel post hoc multiple hypothesis testing scenario (as laid out by García et al., 2010), the significances of rank difference and mean ranks of each libraries performance are provided in Figure 2.

We see that even in this negative averaging scenario scikit-multilearn is statistically significantly more efficient the other libraries when label space division approach (Binary Relevance) is deployed.

When Label Powerset multi-class transformation is used both scikit-multilearn and MLC Toolbox perform better than MULAN/MEKA with statistical significance. In terms of speed scikit-multilearn ranks slightly higher than the MLC Toolbox but consumes a little more memory. The differences between scikit-multilearn and the MLC Toolbox in this approach are not statistically significant.

6. Conclusions

The presented library - scikit-multilearn - is the most extensive scikit-learn compatible multi-label classification library. It provides implementations of both the most popular algorithms and new families of methods such as network-based label space division approaches. It is fast thanks to performing transformations on sparse matrices internally and using well-optimized methods from scikit-learn as base classifiers, and also integrates well with the Python data science stack of scipy. It also provides wrappers to Keras models which allows quick adaptation of deep learning approaches to multi-label classification via problem-transformation approaches, and MEKA/WEKA with parts of MULAN - the standard Java classification stack - and allows easy use of those methods with the rest of the Python stack.

Acknowledgments

We would like to thank Wojciech Stachowski for initial versions of BR/ML-kNN & Classifier Chains implementations, Christian Schulze for asking multiple questions, reporting and fixing bugs, Felipe Almeida for bug fixes and testing, Fernando Benites for providing the implementation of ML-ARAM (Brucker et al., 2011), and Jose Perez-Parras Toledano for his implementation of balanced k-means clustering and other contributions. The work was partially supported by The National Science Centre the research projects no. 2016/21/N/ST6/02382 and 2016/21/D/ST6/02948, by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 691152 (RENOIR); the Polish Ministry of Science and Higher Education fund for supporting internationally co-financed projects in 2016-2019 (agreement no. 3628/H2020/2016/2) and by the Faculty of Computer Science and Management, Wrocław University of Science and Technology statutory funds.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `http://tensorflow.org/`. Software available from tensorflow.org.

F. Benites and E. Sapozhnikova. Haram: A hierarchical aram neural network for large-scale text classification. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 847–854, Nov 2015. doi: 10.1109/ICDMW.2015.14.

Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. Sparse local embeddings for extreme multi-label classification. In *Advances in Neural Information Processing Systems*, pages 730–738, 2015.

Florian Brucker, Fernando Benites, and Elena Sapozhnikova. Multi-label classification and extracting predicted class hierarchies. *Pattern Recognition*, 44(3):724 – 738, 2011.

Francisco Charte and David Charte. Working with multilabel datasets in R: The mldr package. *The R Journal*, 7(2):149–162, December 2015. URL `https://journal.r-project.org/archive/2015-2/charte-charte.pdf`.

Wei-Jie Chen, Yuan-Hai Shao, Chun-Na Li, and Nai-Yang Deng. Mltsvm: a novel twin support vector machine to multi-label learning. *Pattern Recognition*, 52:61–74, 2016.

Yao-Nan Chen and Hsuan-Tien Lin. Feature-aware label space dimension reduction for multi-label classification. In *Advances in Neural Information Processing Systems*, pages 1529–1537, 2012.

François Chollet et al. Keras. `https://keras.io`, 2015.

Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006. URL `http://igraph.org`.

Krzysztof Dembczynski, Weiwei Cheng, and Eyke Hüllermeier. Bayes optimal multilabel classification via probabilistic classifier chains. In *ICML*, volume 10, pages 279–286, 2010.

John W. Eaton, David Bateman, Søren Hauberg, and Rik Wehbring. *GNU Octave version 4.2.1 manual: a high-level interactive language for numerical computations*, 2017.

Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, 180 (10):2044–2064, 2010.

Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009. ISSN 1931-0145.

Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

Kuan-Hao Huang and Hsuan-Tien Lin. Cost-sensitive label embedding for multi-label classification. *Machine Learning*, 106(9-10):1725–1746, 2017.

Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001. URL `http://www.scipy.org/`. Online; Accessed 2018-11-30.

Keigo Kimura, Lu Sun, and Mineichi Kudo. Mlc toolbox: A matlab/octave library for multi-label classification. *arXiv preprint arXiv:1704.02592*, 2017.

Dragi Kocev, Celine Vens, Jan Struyf, and Sašo Džeroski. Tree ensembles for predicting structured outputs. *Pattern Recognition*, 46(3):817–833, 2013.

Joseph B Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017. URL `http://jmlr.org/papers/v18/16-365.html`.

Zijia Lin, Guiguang Ding, Mingqing Hu, and Jianmin Wang. Multi-label classification via feature-aware implicit label space encoding. In *International conference on machine learning*, pages 325–333, 2014.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124. ACM, 2017.

Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Sašo Džeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9): 3084–3104, September 2012.

Elena Montañes, Robin Senge, Jose Barranquero, José Ramón Quevedo, Juan José del Coz, and Eyke Hüllermeier. Dependent binary relevance models for multi-label classification. *Pattern Recognition*, 47(3):1494–1508, 2014.

Jacob Montiel, Jesse Read, Albert Bifet, and Talel Abdessalem. Scikit-Multiflow: A Multi-output Streaming Framework. *CoRR*, abs/1807.04662, 2018.

Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

Travis E Oliphant. Python for scientific computing. *Computing in Science & Engineering*, 9(3), 2007.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

Tiago P. Peixoto. The graph-tool python library. *figshare*, 2014. doi: 10.6084/m9.figshare.1164194. URL `http://figshare.com/articles/graph_tool/1164194`.

Yashoteja Prabhu and Manik Varma. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 263–272. ACM, 2014.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 254–269. Springer, 2009.

Jesse Read, Luca Martino, and David Luengo. Efficient monte carlo optimization for multi-label classifier chains. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 3457–3461. IEEE, 2013.

Jesse Read, Peter Reutemann, Bernhard Pfahringer, and Geoff Holmes. MEKA: A multi-label/multi-target extension to Weka. *Journal of Machine Learning Research*, 17(21):1–5, 2016.

Adriano Rivolli and Andre C. P. L. F. de Carvalho. The utiml Package: Multi-label Classification in R. *The R Journal*, 2018. URL `https://journal.r-project.org/archive/2018/RJ-2018-041/index.html`.

Konstantinos Sechidis, Grigorios Tsoumakas, and Ioannis Vlahavas. On the stratification of multi-label data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 145–158. Springer, 2011.

Frank Seide and Amit Agarwal. Cntk: Microsoft's open-source deep-learning toolkit. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 2135–2135, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2945397. URL `http://doi.acm.org/10.1145/2939672.2945397`.

Soeren Sonnenburg, Heiko Strathmann, Sergey Lisitsyn, Viktor Gal, Fernando J. Iglesias García, Wu Lin, Soumyajit De, Chiyuan Zhang, frx, tklein23, Evgeniy Andreev, Jonas-Behr, sploving, Parijat Mazumdar, Christian Widmer, Pan Deng / Zora, Giovanni De Toni, Saurabh Mahindre, Abhijeet Kislay, Kevin Hughes, Roman Votyakov, khalednasr, Sanuj Sharma, Alesis Novik, Abinash Panda, Evangelos Anagnostopoulos, Liang Pang, Alex Binder, serialhex, and Björn Esser. shogun-toolbox/shogun: Shogun 6.1.0, November 2017.

Eleftherios Spyromitros, Grigorios Tsoumakas, and Ioannis Vlahavas. An empirical study of lazy multilabel classification algorithms. In *Hellenic conference on artificial intelligence*, pages 401–406. Springer, 2008.

Yanmin Sun, Andrew KC Wong, and Mohamed S Kamel. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(04): 687–719, 2009.

Piotr Szymański and Tomasz Kajdanowicz. A Network Perspective on Stratification of Multi-Label Data. In *First International Workshop on Learning with Imbalanced Domains: Theory and Applications, LIDTA 2017, 22 September 2017, ECML-PKDD, Skopje, Macedonia, Proceedings of Machine Learning Research*, volume 74, pages 22–35, 2017.

Piotr Szymański, Tomasz Kajdanowicz, and Kristian Kersting. How is a data-driven approach better than random choice in label space division for multi-label classification? *Entropy*, 18(8):282, 2016.

Piotr Szymański, Tomasz Kajdanowicz, and Nitesh Chawla. Lnemlc: Label network embeddings for multi-label classifiation, 2018.

Farbound Tai and Hsuan-Tien Lin. Multilabel classification with principal label space transformation. *Neural Computation*, 24(9):2508–2542, 2012.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proc. ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, volume 21, pages 53–59. sn, 2008.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Mining multi-label data. In *Data mining and knowledge discovery handbook*, pages 667–685. Springer, 2009.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7): 1079–1089, 2011a.

Grigorios Tsoumakas, Eleftherios Spyromitros-Xioufis, Jozef Vilcek, and Ioannis Vlahavas. Mulan: A java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011b.

Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2285–2294, 2016.

Yunchao Wei, Wei Xia, Min Lin, Junshi Huang, Bingbing Ni, Jian Dong, Yao Zhao, and Shuicheng Yan. Hcp: A flexible cnn framework for multi-label image classification. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1901–1907, 2016.

Ian EH Yen, Xiangru Huang, Wei Dai, Pradeep Ravikumar, Inderjit Dhillon, and Eric Xing. Ppdsparse: A parallel primal-dual sparse method for extreme classification. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 545–553. ACM, 2017.

Zhilou Yu, Hong Hao, Weipin Zhang, and Hongjun Dai. A classifier chain algorithm with k-means for multi-label classification on clouds. *Journal of Signal Processing Systems*, 86 (2-3):337–346, 2017.

Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048, 2007.

Fang Zhao, Yongzhen Huang, Liang Wang, and Tieniu Tan. Deep semantic ranking based hashing for multi-label image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1556–1564, 2015.

set	library	count	mean	std	min	25%	50%	75%	max
Corel5k	MEKA	10	329.35	6.93	311.28	328.73	330.65	331.59	336.41
	MLC-Toolbox	10	540.39	64.40	469.11	489.02	524.54	584.42	652.68
	MULAN	10	323.59	5.40	316.28	321.09	322.94	326.08	332.94
	scikit-multilearn	10	24.05	0.93	22.60	23.18	24.71	24.77	24.85
	utiml.R	10	1928.82	54.03	1825.97	1906.33	1929.56	1961.09	1995.39
bibtex	MEKA	10	317.91	7.41	307.67	311.13	319.67	325.21	326.59
	MLC-Toolbox	10	696.19	34.06	633.80	680.56	696.30	718.40	749.24
	MULAN	10	288.15	6.33	284.30	284.87	285.51	286.77	303.88
	scikit-multilearn	10	41.05	1.74	37.19	41.17	41.81	42.00	42.38
	utiml.R	10	1001.93	43.17	925.99	1005.15	1007.94	1010.66	1075.69
birds	MEKA	10	3.63	0.16	3.38	3.56	3.60	3.68	3.90
	MLC-Toolbox	10	1.05	0.03	1.02	1.02	1.04	1.08	1.10
	MULAN	10	4.30	0.10	4.09	4.27	4.34	4.38	4.40
	scikit-multilearn	10	1.50	0.05	1.45	1.46	1.50	1.54	1.58
	utiml.R	10	7.04	0.04	6.97	7.02	7.04	7.06	7.10
delicious	MEKA	10	5291.10	76.51	5193.85	5257.89	5288.31	5298.34	5483.73
	MLC-Toolbox	10	12294.52	519.71	11082.67	12141.24	12302.92	12510.35	13018.59
	MULAN	10	4902.66	37.00	4861.75	4879.77	4896.75	4901.54	4969.57
	scikit-multilearn	10	549.79	18.06	525.48	529.89	562.26	563.51	565.50
	utiml.R	10	6680.74	539.44	5930.92	6453.18	6549.90	7168.62	7381.27
emotions	MEKA	10	2.31	0.04	2.20	2.31	2.32	2.33	2.35
	MLC-Toolbox	10	0.60	0.02	0.58	0.59	0.60	0.62	0.63
	MULAN	10	2.32	0.06	2.25	2.28	2.31	2.34	2.43
	scikit-multilearn	10	0.75	0.06	0.67	0.72	0.72	0.74	0.86
	utiml.R	10	3.89	0.03	3.84	3.89	3.89	3.91	3.93
enron	MEKA	10	17.74	0.27	17.37	17.58	17.72	17.86	18.21
	MLC-Toolbox	10	28.62	0.27	28.31	28.45	28.53	28.76	29.19
	MULAN	10	15.53	0.20	15.32	15.35	15.48	15.69	15.84
	scikit-multilearn	10	6.40	0.23	5.82	6.44	6.48	6.52	6.57
	utiml.R	10	93.75	0.55	92.72	93.34	93.99	94.13	94.27
genbase	MEKA	10	11.94	0.60	11.19	11.71	11.80	12.18	13.32
	MLC-Toolbox	10	2.95	0.07	2.86	2.92	2.95	3.00	3.09
	MULAN	10	9.69	0.21	9.39	9.56	9.68	9.74	10.15
	scikit-multilearn	10	5.01	0.22	4.52	4.89	5.12	5.16	5.18
	utiml.R	10	41.89	0.33	41.48	41.64	41.88	41.97	42.49
mediamill	MEKA	10	607.94	79.02	448.97	624.48	638.55	655.44	662.81
	MLC-Toolbox	10	1446.38	73.95	1285.85	1419.56	1463.09	1504.47	1518.34
	MULAN	10	424.29	23.80	415.32	416.04	416.89	417.42	491.95
	scikit-multilearn	10	595.14	19.48	566.27	576.67	608.73	610.54	611.08
	utiml.R	10	2701.30	102.56	2536.77	2606.49	2704.90	2800.31	2807.30
medical	MEKA	10	6.66	0.15	6.43	6.54	6.65	6.78	6.87
	MLC-Toolbox	10	5.06	0.14	4.83	4.99	5.08	5.14	5.24
	MULAN	10	7.17	0.32	6.90	7.01	7.08	7.14	8.04
	scikit-multilearn	10	3.65	0.07	3.58	3.58	3.64	3.71	3.76
	utiml.R	10	69.57	0.19	69.35	69.44	69.59	69.63	69.99
scene	MEKA	10	4.86	0.16	4.72	4.73	4.83	4.89	5.23
	MLC-Toolbox	10	2.21	0.03	2.16	2.19	2.21	2.23	2.26
	MULAN	10	4.82	0.27	4.50	4.59	4.76	5.06	5.26
	scikit-multilearn	10	3.68	0.09	3.56	3.60	3.70	3.76	3.78
	utiml.R	10	12.47	0.10	12.27	12.43	12.48	12.55	12.59
tmc2007	MEKA	10	154.56	3.17	147.05	153.69	155.76	156.33	158.31
	MLC-Toolbox	10	874.41	37.77	810.27	859.82	884.38	897.00	930.94
	MULAN	10	146.53	5.67	143.46	143.76	143.88	144.15	157.80
	scikit-multilearn	10	47.94	0.98	46.83	47.02	47.95	48.85	48.99
	utiml.R	10	1039.77	2.68	1036.34	1038.47	1038.92	1041.36	1044.98
yeast	MEKA	10	6.73	0.50	6.21	6.47	6.64	6.85	8.00
	MLC-Toolbox	10	5.29	0.05	5.20	5.26	5.30	5.33	5.35
	MULAN	10	6.10	0.14	5.91	6.05	6.09	6.14	6.43
	scikit-multilearn	10	4.35	0.11	4.23	4.25	4.34	4.45	4.48
	utiml.R	10	13.24	0.06	13.15	13.19	13.24	13.28	13.33

Table 2: User time of library running Binary Relevance for each of the datasets.

set	library	count	mean	std	min	25%	50%	75%	max
Corel5k	MEKA	10	841.26	5.66	831.88	837.45	842.15	845.64	849.58
	MLC-Toolbox	10	20062.98	24.13	20032.54	20042.36	20058.43	20085.25	20100.32
	MULAN	10	829.86	4.53	824.72	826.75	827.35	833.10	837.73
	scikit-multilearn	10	168.79	0.05	168.72	168.75	168.78	168.80	168.88
	utiml.R	10	506.28	1.88	502.55	505.85	506.18	507.63	508.95
bibtex	MEKA	10	790.58	5.33	778.96	791.70	793.06	793.50	794.51
	MLC-Toolbox	10	30794.48	79.82	30670.80	30734.07	30794.87	30858.47	30920.78
	MULAN	10	787.87	3.13	785.26	786.10	786.86	788.13	795.67
	scikit-multilearn	10	445.47	0.07	445.40	445.42	445.44	445.53	445.59
	utiml.R	10	992.02	3.12	983.88	992.73	992.82	992.86	994.79
birds	MEKA	10	535.83	1.69	534.09	534.66	535.35	537.34	538.52
	MLC-Toolbox	10	112.99	0.72	111.29	113.08	113.19	113.34	113.63
	MULAN	10	366.19	2.80	361.58	364.39	366.54	368.41	369.66
	scikit-multilearn	10	57.03	0.01	57.01	57.02	57.03	57.04	57.05
	utiml.R	10	108.24	0.00	108.24	108.24	108.24	108.24	108.25
delicious	MEKA	10	2943.83	7.09	2934.48	2937.81	2943.49	2949.36	2954.81
	MLC-Toolbox	10	140445.37	8393.01	119607.80	139070.95	144463.25	145189.90	145481.35
	MULAN	10	2902.95	4.46	2896.79	2899.11	2903.03	2906.44	2909.23
	scikit-multilearn	10	1301.89	1.44	1299.53	1301.04	1301.73	1302.37	1304.74
	utiml.R	10	2401.63	12.06	2380.95	2394.94	2401.16	2408.31	2421.12
emotions	MEKA	10	163.80	2.01	160.37	162.80	164.58	165.30	165.70
	MLC-Toolbox	10	78.75	0.63	78.07	78.41	78.69	78.76	80.36
	MULAN	10	183.30	1.63	180.41	183.02	183.24	184.68	185.20
	scikit-multilearn	10	50.80	0.01	50.78	50.79	50.80	50.80	50.82
	utiml.R	10	100.75	0.88	100.30	100.30	100.30	100.30	102.32
enron	MEKA	10	664.52	3.56	661.21	662.56	663.84	665.20	673.58
	MLC-Toolbox	10	1685.68	18.74	1657.18	1671.94	1687.40	1697.67	1716.76
	MULAN	10	665.95	3.25	662.15	663.86	664.72	667.29	673.14
	scikit-multilearn	10	102.01	0.01	102.00	102.00	102.01	102.02	102.02
	utiml.R	10	219.62	4.54	216.90	216.98	217.09	220.97	229.98
genbase	MEKA	10	662.21	1.73	658.96	661.69	662.61	663.14	664.42
	MLC-Toolbox	10	438.85	6.97	430.62	433.28	436.34	443.71	450.86
	MULAN	10	682.95	3.31	677.60	681.85	682.10	683.88	690.77
	scikit-multilearn	10	125.75	0.02	125.73	125.74	125.75	125.76	125.79
	utiml.R	10	146.65	0.07	146.58	146.58	146.64	146.72	146.72
mediamill	MEKA	10	1122.80	4.01	1119.86	1121.22	1121.39	1122.73	1133.81
	MLC-Toolbox	10	14529.87	85.39	14416.90	14479.99	14506.57	14551.84	14717.54
	MULAN	10	1057.10	29.06	974.44	1065.56	1065.99	1066.99	1067.45
	scikit-multilearn	10	572.13	0.31	571.48	572.05	572.11	572.34	572.60
	utiml.R	10	909.74	3.17	904.12	911.34	911.34	911.34	911.34
medical	MEKA	10	645.37	1.23	642.86	645.07	645.28	646.36	646.94
	MLC-Toolbox	10	593.72	4.43	588.30	590.20	593.54	596.65	602.43
	MULAN	10	642.04	2.46	639.50	639.79	642.54	642.62	647.13
	scikit-multilearn	10	89.29	0.02	89.26	89.28	89.29	89.30	89.31
	utiml.R	10	158.05	2.52	156.46	156.46	156.51	158.93	162.40
scene	MEKA	10	628.31	1.92	624.99	627.08	628.97	629.00	631.96
	MLC-Toolbox	10	154.23	2.95	150.09	152.42	153.38	155.38	159.36
	MULAN	10	637.62	1.87	635.83	636.15	636.59	639.01	640.55
	scikit-multilearn	10	105.90	0.04	105.88	105.88	105.89	105.90	106.02
	utiml.R	10	149.66	0.67	149.44	149.44	149.44	149.44	151.46
tmc2007	MEKA	10	804.08	2.02	801.53	802.19	804.07	805.65	806.88
	MLC-Toolbox	10	11013.90	142.29	10827.76	10946.62	10962.75	11055.96	11321.70
	MULAN	10	823.86	1.89	821.51	822.77	823.19	824.86	827.12
	scikit-multilearn	10	536.68	1.53	534.21	535.62	537.36	537.78	538.13
	utiml.R	10	918.84	13.66	912.77	913.02	914.89	915.48	955.07
yeast	MEKA	10	627.35	3.28	623.79	626.16	626.79	626.86	636.20
	MLC-Toolbox	10	319.40	1.94	316.86	318.17	319.49	320.00	323.11
	MULAN	10	631.91	2.94	627.02	630.77	632.25	633.77	635.93
	scikit-multilearn	10	69.11	0.01	69.09	69.10	69.11	69.11	69.12
	utiml.R	10	132.98	0.63	132.78	132.78	132.78	132.78	134.76

Table 3: Maximum memory usage (in kilobytes) of library running Binary Relevance for each of the datasets.

set	library	count	mean	std	min	25%	50%	75%	max
Corel5k	MEKA	10	47.93	1.06	46.67	47.08	47.60	48.82	49.81
	MLC-Toolbox	10	17.19	0.24	16.86	16.98	17.21	17.34	17.54
	MULAN	10	54.92	1.88	51.15	54.24	55.19	56.07	57.34
	scikit-multilearn	10	8.56	0.33	8.15	8.28	8.48	8.81	9.06
bibtex	MEKA	10	46.60	1.37	43.60	45.98	46.64	47.56	48.48
	MLC-Toolbox	10	28.83	0.47	28.15	28.62	28.84	29.08	29.70
	MULAN	10	53.81	1.66	52.24	52.62	52.81	55.21	56.44
	scikit-multilearn	10	26.36	0.14	26.23	26.25	26.32	26.40	26.66
birds	MEKA	10	2.03	0.17	1.82	1.88	1.98	2.21	2.25
	MLC-Toolbox	10	0.40	0.01	0.38	0.40	0.40	0.40	0.41
	MULAN	10	2.60	0.04	2.53	2.57	2.58	2.63	2.66
	scikit-multilearn	10	0.85	0.05	0.77	0.84	0.85	0.86	0.97
delicious	MEKA	10	1406.20	86.09	1264.17	1357.53	1427.58	1480.17	1486.04
	MLC-Toolbox	10	131.27	4.79	123.27	128.79	130.55	135.06	137.92
	MULAN	10	1606.88	63.40	1478.83	1623.74	1632.84	1642.51	1651.81
	scikit-multilearn	10	59.40	1.15	57.64	58.30	59.73	60.41	60.68
emotions	MEKA	10	1.63	0.11	1.49	1.55	1.58	1.72	1.82
	MLC-Toolbox	10	0.37	0.01	0.36	0.36	0.36	0.37	0.38
	MULAN	10	2.16	0.03	2.13	2.14	2.16	2.18	2.21
	scikit-multilearn	10	0.53	0.06	0.47	0.48	0.52	0.59	0.63
enron	MEKA	10	7.54	0.43	7.07	7.21	7.46	7.69	8.39
	MLC-Toolbox	10	1.91	0.03	1.87	1.89	1.90	1.91	1.98
	MULAN	10	8.39	0.26	7.94	8.22	8.36	8.64	8.71
	scikit-multilearn	10	3.54	0.09	3.42	3.48	3.52	3.63	3.67
genbase	MEKA	10	3.94	0.19	3.67	3.81	3.90	4.10	4.21
	MLC-Toolbox	10	0.38	0.02	0.36	0.36	0.38	0.39	0.42
	MULAN	10	4.32	0.19	3.81	4.32	4.36	4.42	4.47
	scikit-multilearn	10	2.92	0.04	2.86	2.90	2.91	2.95	2.98
mediamill	MEKA	10	646.99	7.33	637.66	640.00	648.32	652.74	657.56
	MLC-Toolbox	10	210.37	4.56	205.49	207.03	207.94	214.65	218.22
	MULAN	10	647.64	26.74	573.99	650.70	652.64	658.56	670.37
	scikit-multilearn	10	479.39	15.57	445.40	475.34	486.05	490.02	491.12
medical	MEKA	10	2.74	0.19	2.36	2.67	2.74	2.78	3.05
	MLC-Toolbox	10	0.55	0.02	0.54	0.54	0.55	0.56	0.59
	MULAN	10	3.68	0.18	3.21	3.70	3.72	3.76	3.84
	scikit-multilearn	10	2.52	0.08	2.30	2.52	2.54	2.55	2.59
scene	MEKA	10	3.36	0.14	3.11	3.31	3.37	3.44	3.58
	MLC-Toolbox	10	0.74	0.02	0.72	0.73	0.74	0.75	0.78
	MULAN	10	3.81	0.18	3.48	3.80	3.88	3.92	3.96
	scikit-multilearn	10	2.96	0.05	2.86	2.92	2.98	2.99	3.01
tmc2007	MEKA	10	85.39	3.73	76.84	86.48	86.92	87.17	88.28
	MLC-Toolbox	10	139.91	0.29	139.52	139.66	139.93	140.10	140.33
	MULAN	10	89.85	3.21	80.90	90.44	90.71	90.83	92.52
	scikit-multilearn	10	32.36	1.50	29.61	31.72	33.18	33.32	33.68
yeast	MEKA	10	3.76	0.14	3.47	3.73	3.78	3.87	3.90
	MLC-Toolbox	10	1.18	0.02	1.14	1.17	1.18	1.19	1.21
	MULAN	10	4.02	0.11	3.79	3.98	4.05	4.10	4.14
	scikit-multilearn	10	2.11	0.10	1.93	2.06	2.08	2.16	2.27

Table 4: User time of library running Label Powerset for each of the datasets.

set	library	count	mean	std	min	25%	50%	75%	max
Corel5k	MEKA	10	1666.84	1.71	1664.12	1665.87	1666.58	1668.02	1669.99
	MLC-Toolbox	10	1400.39	35.06	1336.14	1380.37	1396.89	1417.98	1462.22
	MULAN	10	1695.80	7.11	1687.45	1691.15	1694.73	1697.20	1712.86
	scikit-multilearn	10	1534.89	3.24	1530.85	1532.10	1534.55	1536.74	1540.01
bibtex	MEKA	10	1426.38	2.94	1422.25	1425.46	1426.30	1427.00	1431.46
	MLC-Toolbox	10	2852.94	165.46	2573.12	2791.99	2890.47	2929.04	3097.12
	MULAN	10	1432.34	2.81	1428.26	1430.74	1432.21	1433.66	1437.83
	scikit-multilearn	10	1333.13	2.86	1328.46	1331.33	1332.35	1335.19	1338.06
birds	MEKA	10	259.04	0.71	258.55	258.71	258.85	259.02	260.97
	MLC-Toolbox	10	65.78	0.81	64.73	65.28	65.50	66.16	67.57
	MULAN	10	258.67	1.59	256.11	257.51	258.94	259.77	261.12
	scikit-multilearn	10	58.05	0.05	57.97	58.02	58.04	58.07	58.12
delicious	MEKA	10	14385.92	7.52	14378.90	14381.37	14383.44	14385.24	14399.84
	MLC-Toolbox	10	5791.13	170.72	5492.74	5709.92	5826.34	5919.17	5987.11
	MULAN	10	14661.50	7.65	14645.94	14657.66	14661.54	14666.47	14673.41
	scikit-multilearn	10	14213.03	28.43	14172.85	14192.80	14211.53	14241.15	14246.31
emotions	MEKA	10	139.68	1.61	136.51	139.08	139.32	140.76	142.36
	MLC-Toolbox	10	63.48	0.72	62.96	63.22	63.22	63.42	65.48
	MULAN	10	162.92	1.69	159.86	162.52	163.02	163.83	165.68
	scikit-multilearn	10	51.07	0.05	50.99	51.05	51.08	51.10	51.17
enron	MEKA	10	707.42	2.56	701.80	707.04	707.94	708.44	711.58
	MLC-Toolbox	10	215.96	10.90	200.23	207.90	215.11	225.24	232.22
	MULAN	10	714.13	4.37	710.48	712.62	713.34	713.60	726.18
	scikit-multilearn	10	156.54	0.61	155.61	156.04	156.45	157.09	157.38
genbase	MEKA	10	649.53	0.66	648.36	649.28	649.60	649.76	650.84
	MLC-Toolbox	10	114.91	4.80	109.46	111.19	114.27	117.05	125.15
	MULAN	10	640.55	1.35	639.77	639.84	640.01	640.15	643.92
	scikit-multilearn	10	125.80	0.02	125.78	125.79	125.80	125.81	125.84
mediamill	MEKA	10	13629.82	6.41	13624.72	13625.81	13628.58	13630.40	13646.53
	MLC-Toolbox	10	2938.81	12.77	2922.50	2927.15	2939.71	2949.28	2955.52
	MULAN	10	13582.38	50.91	13547.02	13550.89	13553.20	13621.77	13674.80
	scikit-multilearn	10	12801.70	13.43	12773.04	12797.68	12802.40	12809.80	12821.79
medical	MEKA	10	490.32	4.03	487.45	487.55	489.15	489.94	498.78
	MLC-Toolbox	10	129.75	7.82	121.34	123.38	126.39	137.62	140.70
	MULAN	10	545.42	1.74	543.10	544.33	544.88	546.41	549.15
	scikit-multilearn	10	89.29	0.02	89.26	89.28	89.28	89.30	89.31
scene	MEKA	10	501.84	1.37	500.18	500.84	501.25	503.01	504.34
	MLC-Toolbox	10	100.44	1.48	98.37	99.48	100.06	101.22	103.34
	MULAN	10	619.23	13.63	592.73	621.83	624.34	625.81	630.77
	scikit-multilearn	10	106.02	0.01	106.00	106.01	106.02	106.02	106.04
tmc2007	MEKA	10	2873.22	1.60	2870.86	2871.82	2873.66	2874.12	2875.92
	MLC-Toolbox	10	1989.50	140.75	1767.65	1911.14	1975.03	2041.82	2284.26
	MULAN	10	2886.81	4.44	2882.71	2883.54	2885.84	2888.71	2897.22
	scikit-multilearn	10	2729.93	5.23	2719.09	2728.22	2731.22	2733.61	2735.87
yeast	MEKA	10	621.49	1.36	619.78	620.30	621.75	621.96	624.14
	MLC-Toolbox	10	130.58	1.29	129.18	129.71	130.50	130.96	133.66
	MULAN	10	629.12	1.26	626.62	628.23	629.89	629.94	629.98
	scikit-multilearn	10	84.67	0.16	84.45	84.54	84.67	84.79	84.89

Table 5: Maximum memory usage (in kilobytes) of library running Label Powerset for each of the datasets.

Journal of Machine Learning Research 20 (2019) 1-45 Submitted 5/17; Revised 8/18; Published 2/19

Scalable Approximations for Generalized Linear Problems

Murat A. Erdogdu ERDOGDU@CS.TORONTO.EDU
Department of Computer Science
Department of Statistical Sciences
University of Toronto
Toronto, ON M5S 3G3, Canada

Mohsen Bayati BAYATI@STANFORD.EDU
Graduate School of Business
Stanford University
Stanford, CA 94305, USA

Lee H. Dicker LDICKER@STAT.RUTGERS.EDU
Department of Statistics and Biostatistics
Rutgers University
Piscataway, NJ 08854, USA

Editor: Qiang Liu

Abstract

In stochastic optimization, the population risk is generally approximated by the empirical risk which is in turn minimized by an iterative algorithm. However, in the large-scale setting, empirical risk minimization may be computationally restrictive. In this paper, we design an efficient algorithm to approximate the population risk minimizer in generalized linear problems such as binary classification with surrogate losses and generalized linear regression models. We focus on large-scale problems where the iterative minimization of the empirical risk is computationally intractable, i.e., the number of observations n is much larger than the dimension of the parameter p ($n \gg p \gg 1$). We show that under random sub-Gaussian design, the true minimizer of the population risk is approximately proportional to the corresponding ordinary least squares (OLS) estimator. Using this relation, we design an algorithm that achieves the same accuracy as the empirical risk minimizer through iterations that attain up to a quadratic convergence rate, and that are computationally cheaper than any batch optimization algorithm by at least a factor of $\mathcal{O}(p)$. We provide theoretical guarantees for our algorithm, and analyze the convergence behavior in terms of data dimensions. Finally, we demonstrate the performance of our algorithm on well-known classification and regression problems, through extensive numerical studies on large-scale datasets, and show that it achieves the highest performance compared to several other widely used optimization algorithms.

Keywords: Generalized Linear Problems, Stochastic optimization, Subsampling, Dimension reduction in optimization.

1. Introduction

We consider the following stochastic optimization problem

$$\underset{\beta\in\mathbb{R}^p}{\text{minimize}}\ R(\beta) \coloneqq \mathbb{E}\left[\Psi\left(\langle x,\beta\rangle\right) - y\langle x,\beta\rangle\right]\ , \tag{1}$$

©2019 Murat A. Erdogdu, Mohsen Bayati, Lee H. Dicker.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-279.html.

where $\Psi : \mathbb{R} \to \mathbb{R}$ is a non-linear function, $y \in \mathcal{Y} \subset \mathbb{R}$ denotes the response variable, $x \in \mathcal{X} \subset \mathbb{R}^p$ denotes the predictor (or covariate), and the expectation is over the joint distribution of (y, x). The above minimization is called a generalized linear problem in its canonical representation, and it is commonly encountered in statistical learning. Celebrated examples include binary classification with smooth surrogate losses (Buja et al., 2005; Reid and Williamson, 2010), and generalized linear models (GLMs) such as Poisson regression, logistic regression, ordinary least squares, multinomial regression with many applications involving graphical models (Nelder and Baker, 1972; McCullagh and Nelder, 1989; Wainwright and Jordan, 2008; Koller and Friedman, 2009). These methods play a crucial role in numerous machine learning and statistics problems, and they provide a miscellaneous framework for many regression and classification tasks.

The exact minimization of the stochastic optimization problem (1) requires the knowledge of the underlying distribution of the variables (y, x). In practice, however, the joint distribution of the pair is not available; therefore, after observing n independent data points (y_i, x_i), the standard approach is to minimize the following surrogate of (1), often referred to as empirical risk approximation

$$\underset{\beta\in\mathbb{R}^p}{\text{minimize}}\ \widehat{R}(\beta) \coloneqq \frac{1}{n}\sum_{i=1}^{n} \Psi\left(\langle x_i,\beta\rangle\right) - y_i\langle x_i,\beta\rangle. \tag{2}$$

In the case of GLMs, the empirical risk minimization given in (2) is equivalent to the maximum likelihood estimation (MLE), whereas in the case of binary classification, it is generally referred to as surrogate loss minimization. Due to the non-linear structure of the optimization task given in (2), minimizing the empirical risk requires iterative methods. The first-order approximation of the non-linear risk yields the gradient descent algorithm, which attains a (local) linear convergence rate under certain conditions with $\mathcal{O}(np)$ per-iteration cost. Although its convergence rate is slow compared to that of the second-order methods, its modest per-iteration cost makes it practical for large-scale problems. Regardless of the problem formulation, the most commonly used optimization method for computing the MLE is the Newton-Raphson method, which may be viewed as a reweighted least squares algorithm (McCullagh and Nelder, 1989; Buja et al., 2005). This method uses a second-order approximation to benefit from the curvature of the log-likelihood and achieves locally quadratic convergence. A drawback of this approach is its excessive per-iteration cost of $\mathcal{O}(np^2)$. To remedy this, Hessian-free Krylov sub-space based methods such as conjugate gradient can be used, but the resulting direction is imprecise (Hestenes and Stiefel, 1952; Paige and Saunders, 1975; Martens, 2010). In the regime $n \gg p$, another popular optimization technique is the class of Quasi-Newton methods (Bishop, 1995; Nesterov, 2004), which can attain a per-iteration cost of $\mathcal{O}(np)$, and the convergence rate is locally super-linear; a well-known member of this class of methods is the BFGS algorithm (Broyden, 1970; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970).

In this paper, we take an alternative approach for minimizing (1), based on an identity that is well-known in some areas of statistics, but appears to have received relatively little attention for its computational implications in large-scale problems. Let β^{pop} denote the true minimizer of the population risk given in (1), and let β^{ols} denote the corresponding ordinary least squares (OLS) coefficients defined as $\beta^{\mathsf{ols}} = \mathbb{E}\left[xx^T\right]^{-1}\mathbb{E}\left[xy\right]$. Then, under

certain random predictor (design) models,

$$\beta^{\mathsf{pop}} \propto \beta^{\mathsf{ols}}. \tag{3}$$

For logistic regression with Gaussian design (which is equivalent to Fisher's discriminant analysis), (3) was noted by Fisher in the 1930s (Fisher, 1936); a more general formulation for models with Gaussian design is given in Brillinger (1982). The relationship (3) suggests that if the constant of proportionality is known, then β^{pop} can be estimated by computing the OLS estimator, which may be substantially simpler than minimizing the empirical risk. In fact, in some applications like binary classification, it may not be necessary to find the constant of proportionality in (3). Our work in this paper builds on this idea.

Our contributions can be summarized as follows.

1. We show that β^{pop} is approximately proportional to β^{ols} in the random design setting, regardless of the covariate (predictor) distribution. That is, we prove

$$\left\| \beta^{\mathsf{pop}} - c_\Psi \times \beta^{\mathsf{ols}} \right\|_\infty \lesssim c \frac{\|\beta^{\mathsf{pop}}\|_\infty}{\sqrt{p}},$$

   for some $c_\Psi \in \mathbb{R}$ which depends on the non-linearity Ψ. We note that above rate is the *relative* decay, and it is obtained under the assumption that $\mathbb{E}[\langle x, \beta^{\mathsf{pop}} \rangle^2] \lesssim c^2$. Our generalization uses zero-bias transformations (Goldstein and Reinert, 1997). We also show that the relation (3) still holds under certain types of regularization.

2. We design a computationally efficient estimator for β^{pop} by first estimating the OLS coefficients, and then estimating the proportionality constant c_Ψ via line search. We refer to the resulting estimator as the Scaled Least Squares (SLS) estimator and denote it by $\hat{\beta}^{\mathsf{sls}}$. After estimating the OLS coefficients, the second step of our algorithm involves finding a root of a real valued function; this can be accomplished using iterative methods with up to a quadratic convergence rate and only $\mathcal{O}(n)$ per-iteration cost. This is computationally cheaper than the classical batch methods such as gradient descent by at least a factor of $\mathcal{O}(p)$.

3. For random design with sub-Gaussian predictors and bounded $\Psi^{(2)}$, we show that

$$\left\| \hat{\beta}^{\mathsf{sls}} - \beta^{\mathsf{pop}} \right\|_\infty \lesssim \frac{\|\beta^{\mathsf{pop}}\|_\infty}{\sqrt{p}} + \sqrt{\frac{p}{n}}.$$

   This bound characterizes the performance of the proposed estimator in terms of data dimensions, and justifies the use of the algorithm in the large-scale setting where $n \gg p \gg 1$. We also provide theoretical guarantees when subsampling is utilized in the algorithm for further efficiency.

4. We demonstrate how to transform a binary classification problem with smooth surrogate loss into a generalized linear problem, and how our methods can be applied to obtain a computationally efficient optimization scheme.

5. We propose a scalable optimization method for converting one generalized linear problem to another by exploiting the proportionality relation (3). The proposed algorithm requires only $\mathcal{O}(n)$ computations per each iteration, with no additional one-time cost. We further discuss the canonicalization of the square loss, which may be of independent interest for non-convex optimization.

6. We empirically study the statistical and computational performance of $\hat{\beta}^{\mathsf{sls}}$, and compare it to that of the empirical risk minimizer (using several well-known implementations), on a variety of large-scale datasets.

The rest of the paper is organized as follows: Section 1.1 surveys the related work and Section 2 introduces the required background and the notation. In Section 3, we provide the intuition behind the relationship (3), which are based on exact calculations for the Gaussian design setting. In Section 4, we propose our algorithm and discuss its computational properties. Theoretical results are given in Section 5. In Section 6, we propose an algorithm to convert one GLM type to another. We discuss how a binary classification problem can be cast as a generalized linear problem in Section 7, and in Section 7.1 we propose a method to canonicalize the square loss. Section 8 provides a thorough comparison between the proposed algorithm and other existing methods. Finally, we conclude with a brief discussion in Section 9.

1.1. Related work

As mentioned in Section 1, the relationship (3) is well-known in several forms in statistics literature. Brillinger (1982) derived (3) for models with Gaussian predictors using Stein's lemma. Li and Duan (1989) studied model misspecification problems in statistics and derived (3) when the predictor distribution has linear conditional means (this is a slight generalization of Gaussian predictors). The relation (3) has led to various techniques for dimension reduction (Li, 1991; Li and Dong, 2009), and more recently, it has been studied by Plan and Vershynin (2016); Thrampoulidis et al. (2015) in the context of compressed sensing. It has been shown that the standard lasso estimator may be very effective when used in models where the relationship between the expected response and the signal is nonlinear, and the predictors (i.e. the design or sensing matrix) are Gaussian. A common theme for all of this previous work is that it focuses solely on settings where (3) holds exactly and the predictors (covariates) are Gaussian (or, in the case of Li and Duan (1989), very nearly Gaussian).

The proportionality relation is solely based on Stein's lemma and its variants. There are recent studies using Stein's lemma in various machine learning and optimization tasks such as second-order optimization (Erdogdu, 2015, 2016, 2017), Bayesian inference (Liu and Wang, 2016), measuring sample quality and model evaluation tasks (Gorham and Mackey, 2015; Liu et al., 2016).

Two key novelties of the present paper are (i) our focus on the computational benefits following from (3) for large scale problems with $n \gg p \gg 1$; and (ii) our rigorous finite sample analysis of models with non-Gaussian predictors, where (3) is shown to be approximately valid. To the best of our knowledge, the present paper and its earlier version Erdogdu et al. (2016) are the first to consider the relation (3) in the context of optimization.

2. Preliminaries and Notation

We assume a random design setting, where the observed data consists of n random iid pairs (y_1, x_1), (y_2, x_2), ..., (y_n, x_n); $y_i \in \mathcal{Y} \subset \mathbb{R}$ is the response variable and $x_i = (x_{i1}, \cdots, x_{ip})^T \in \mathcal{X} \subset \mathbb{R}^p$ is the vector of predictors or covariates. We focus on problems where the minimization (1) is desirable, but we allow model misspecification, and do not need to assume that (y_i, x_i) are actually drawn from a particular parametric model. In other words, the joint distribution of (y_i, x_i) can be arbitrary.

$$\beta^{\mathsf{pop}} = \underset{\beta \in \mathbb{R}^p}{\operatorname{argmin}} \ \mathbb{E}\left[\Psi(\langle x_i, \beta\rangle) - y_i \langle x_i, \beta\rangle\right]. \tag{4}$$

While we make no assumptions on Ψ beyond smoothness, note that when the optimization problem is GLM, and Ψ is the cumulant generating function for $y_i \mid x_i$, then the problem reduces to the standard GLM with canonical link and regression parameters β^{pop} (McCullagh and Nelder, 1989). Examples of GLMs in this form include logistic regression with $\Psi(w) = \log\{1 + e^w\}$, Poisson regression with $\Psi(w) = e^w$, and linear regression (least squares) with $\Psi(w) = w^2/2$.

Our objective is to find a computationally efficient estimator for β^{pop}. The alternative estimator for β^{pop} proposed in this paper is related to the OLS coefficient vector, which is defined by $\beta^{\mathsf{ols}} := \mathbb{E}[x_i x_i^T]^{-1}\mathbb{E}[x_i y_i]$; the corresponding OLS estimator is $\hat{\beta}^{\mathsf{ols}} := (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y$, where $\mathbf{X} = (x_1, \ldots, x_n)^T$ is the $n \times p$ design matrix and $y = (y_1, \ldots, y_n)^T \in \mathbb{R}^n$.

Additionally, throughout the text we let $[m] = \{1, 2, ..., m\}$, for positive integers m, and we denote the size of a set S by $|S|$. The m-th derivative of a function $g : \mathbb{R} \to \mathbb{R}$ is denoted by $g^{(m)}$. For a vector $u \in \mathbb{R}^p$ and a $n \times p$ matrix $\mathbf{U}$, we let $\|u\|_q$ and $\|\mathbf{U}\|_q$ denote the ℓ_q-vector and -operator norms, respectively. If $S \subseteq [n]$, let $\mathbf{U}_S$ denote the $|S| \times p$ matrix obtained from $\mathbf{U}$ by extracting the rows that are indexed by S. For a symmetric matrix $\mathbf{M} \in \mathbb{R}^{p\times p}$, $\lambda_{\max}(\mathbf{M})$ and $\lambda_{\min}(\mathbf{M})$ denote the maximum and minimum eigenvalues, respectively, and $\rho_q(\mathbf{M})$ for $q \in \{1, 2, \infty\}$ denotes the condition number of $\mathbf{M}$ with respect to ℓ_q-norm. We denote by N_p the p-variate normal distribution, and all expectations are over all randomness inside the brackets. Finally, we use $a \lesssim b$ and $a \leq \mathcal{O}(b)$ interchangeably, whichever is convenient (where $\mathcal{O}(\cdot)$ refers to the big O notation).

3. From OLS to the True Minimizer: Gaussian Case

To motivate our methodology, we assume in this section that the covariates are multivariate normal, as in Brillinger (1982). These distributional assumptions will be relaxed to a certain extent in Section 5.

Proposition 1 *Assume that the covariates are multivariate normal with mean 0 and covariance matrix $\mathbf{\Sigma}$, i.e. $x_i \sim \mathsf{N}_p(0, \mathbf{\Sigma})$. Then β^{pop} can be written as*

$$\beta^{\mathsf{pop}} = c_\Psi \times \beta^{\mathsf{ols}}, \tag{5}$$

where $c_\Psi \in \mathbb{R}$ is the fixed point of the mapping

$$z \to \mathbb{E}\left[\Psi^{(2)}(\langle x_i, \beta^{\mathsf{ols}}\rangle z)\right]^{-1}. \tag{6}$$

Proof [of Proposition 1] The stationary point of the minimization problem in (4) satisfies the following normal equations,

$$\mathbb{E}[y_i x_i] = \mathbb{E}\left[x_i \Psi^{(1)}(\langle x_i, \beta\rangle)\right] \quad \text{for} \quad i = 1, 2, ..., p. \tag{7}$$

Now, denote by $\phi(x \mid \mathbf{\Sigma})$ the multivariate normal density with mean 0 and covariance matrix $\mathbf{\Sigma}$. We recall the well-known property of Gaussian density $\mathrm{d}\phi(x \mid \mathbf{\Sigma})/\mathrm{d}x = -\mathbf{\Sigma}^{-1}x\phi(x \mid \mathbf{\Sigma})$. Using this and the integration by parts on the right hand side of the above equation, we obtain for $i = 1, 2, ..., p$

$$\begin{aligned}\mathbb{E}\left[x_i \Psi^{(1)}(\langle x_i, \beta\rangle)\right] &= \int x\Psi^{(1)}(\langle x, \beta\rangle)\phi(x \mid \mathbf{\Sigma})\, \mathrm{d}x, \\ &= \mathbf{\Sigma}\beta \ \underbrace{\mathbb{E}\left[\Psi^{(2)}(\langle x_i, \beta\rangle)\right]}_{\in \mathbb{R}},\end{aligned} \tag{8}$$

which is basically the Stein's lemma. Combining this with the normal equations (7) and multiplying both side with $\mathbf{\Sigma}^{-1}$, we obtain the desired result. ■

Proposition 1 and its proof provide the main intuition behind our proposed method. Observe that in our derivation, we only worked with the right hand side of the normal equations (7) which does not depend on the response variable y_i. Therefore, the equivalence will hold regardless of the joint distribution of (y_i, x_i). This is the main difference from the proof of Brillinger (1982) where y_i is assumed to follow a single index model. In Section 5, where we extend the method to non-Gaussian predictors, the identity (8) is generalized via the zero-bias transformations from Goldstein and Reinert (1997).

3.1. Regularization

A version of Proposition 1 incorporating regularization — an important tool for datasets where p is large relative to n or the predictors are highly collinear — is also possible, as outlined briefly in this section. We focus on ℓ^2-regularization (ridge regression) in this section; some connections with lasso (ℓ^1-regularization) are discussed in Section 5 and Corollary 6.

For $\lambda \geq 0$, define the ℓ_2-regularized empirical risk minimizer,

$$\beta_\lambda^{\mathsf{pop}} = \underset{\beta \in \mathbb{R}^p}{\operatorname{argmin}} \ \mathbb{E}\left[\Psi(\langle x_i, \beta\rangle) - y_i\langle x_i, \beta\rangle\right] + \frac{\lambda}{2}\|\beta\|_2^2 \tag{9}$$

and the corresponding ℓ^2-regularized OLS coefficients $\beta_\lambda^{\mathsf{ols}} = \left(\mathbb{E}\left[x_i x_i^T\right] + \lambda\mathbf{I}\right)^{-1}\mathbb{E}\left[x_i y_i\right]$ (so $\beta^{\mathsf{pop}} = \beta_0^{\mathsf{pop}}$ and $\beta^{\mathsf{ols}} = \beta_0^{\mathsf{ols}}$). The same argument as above implies that

$$\beta_\lambda^{\mathsf{pop}} = c_\Psi \times \beta_\gamma^{\mathsf{ols}}, \quad \text{where} \quad \gamma = \lambda c_\Psi. \tag{10}$$

This suggests that the ordinary ridge regression for the linear model can be used to estimate the ℓ^2-regularized empirical risk minimizer $\beta_\lambda^{\mathsf{pop}}$. Further pursuing these ideas for problems where regularization is a critical issue may be an interesting area for future research.

Algorithm 1 SLS: Scaled Least Squares Estimator

Input: Data $(y_i, x_i)_{i=1}^n$

Step 1. Compute the least squares estimator: $\hat{\beta}^{\mathsf{ols}}$ **and** $\hat{y} = \mathbf{X}\hat{\beta}^{\mathsf{ols}}$**.**

For a sub-sampling based OLS estimator, let $S \subset [n]$ be a random subset and take $\hat{\beta}^{\mathsf{ols}} = \frac{|S|}{n}(\mathbf{X}_S^T\mathbf{X}_S)^{-1}\mathbf{X}^T y$.

Step 2. Solve the following equation for $c \in \mathbb{R}$**:** $1 = \frac{c}{n}\sum_{i=1}^n \Psi^{(2)}(c\,\hat{y}_i)$**.**

Use Newton's root-finding method:

Initialize c;

Repeat until convergence:

$$c \leftarrow c - \frac{c\,\frac{1}{n}\sum_{i=1}^n \Psi^{(2)}(c\,\hat{y}_i) - 1}{\frac{1}{n}\sum_{i=1}^n \left\{\Psi^{(2)}(c\,\hat{y}_i) + c\,\hat{y}_i\Psi^{(3)}(c\,\hat{y}_i)\right\}}.$$

Output: $\hat{\beta}^{\mathsf{sls}} = c \times \hat{\beta}^{\mathsf{ols}}$.

4. SLS: Scaled Least Squares Estimator

Motivated by the results in the previous section, we design a computationally efficient algorithm that approximates the stochastic optimization problem (1) that is as simple as solving the least squares problem; it is described in Algorithm 1. The algorithm has two basic steps. First, we estimate the OLS coefficients, and then in the second step we estimate the proportionality constant via a simple root-finding algorithm.

There are numerous fast optimization methods to solve the least squares problem, and even a superficial review of these could go beyond the page limits of this paper. We emphasize that this step (finding the OLS estimator) does not have to be iterative and it is the main computational cost of the proposed algorithm. We suggest using a sub-sampling based estimator for β^{ols}, where we only use a subset of the observations to estimate the covariance matrix. Let $S \subset [n]$ be a random sub-sample and denote by $\mathbf{X}_S$ the sub-matrix formed by the rows of $\mathbf{X}$ in S. Then the sub-sampled OLS estimator is given as $\hat{\beta}^{\mathsf{ols}} = \left(\frac{1}{|S|}\mathbf{X}_S^T\mathbf{X}_S\right)^{-1}\frac{1}{n}\mathbf{X}^T y$. Properties of sub-sampling and sketching based estimators have been well-studied (Vershynin, 2010; Dhillon et al., 2013; Erdogdu and Montanari, 2015; Pilanci and Wainwright, 2015; Roosta-Khorasani and Mahoney, 2016a,b). For sub-Gaussian covariates, it suffices to use a sub-sample size of $\mathcal{O}(p\log(p))$ (Vershynin, 2010). Hence, this step requires a single time computational cost of $\mathcal{O}\left(|S|p^2 + p^3 + np\right) \approx \mathcal{O}\left(p\max\{p^2\log(p), n\}\right)$. For other approaches, we refer reader to Rokhlin and Tygert (2008); Drineas et al. (2011); Dhillon et al. (2013); Erdogdu and Montanari (2015) and the references therein.

The second step of Algorithm 1 involves solving a simple root-finding problem. As with the first step, there are numerous methods available for completing this task. Newton's root-finding method with quadratic convergence or Halley's method with cubic convergence may be appropriate choices. We highlight that this step costs only $\mathcal{O}(n)$ per-iteration and that we can attain fast convergence rates of higher order algorithms. The resulting per-iteration cost is cheaper than commonly used batch algorithms by at least a factor of $\mathcal{O}(p)$ — indeed, the cost of computing the gradient is $\mathcal{O}(np)$. For simplicity, we use Newton's root-finding method. We also note that in certain classification problems, only the direction of β^{pop} is needed to make a prediction, so there is no need to compute the scaling constant.

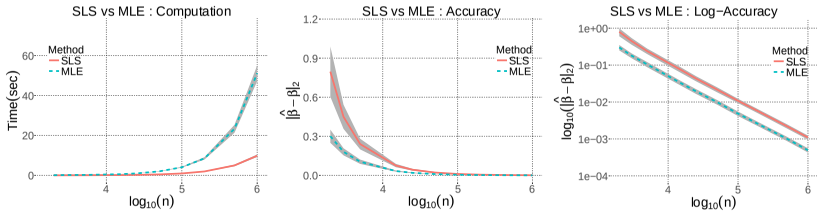


Figure 1: Logistic regression with iid standard Gaussian design. The left plot shows the computational cost (time) for finding the MLE and SLS as n grows and $p = 200$. The middle and the right plots depict the accuracy of the estimators in standard and log scales, respectively. In the regime where the MLE is expensive to compute, the SLS is found much more rapidly and has the same accuracy. R's built-in functions are used to find the MLE.

Correct initialization of the scaling constant c depends on the optimization problem. For example, in the case of GLM problems, assuming that the GLM is a good approximation to the true conditional distribution, by the law of total variance and basic properties of GLMs, we have

$$\mathrm{Var}(y_i) = \mathbb{E}\left[\mathrm{Var}(y_i \mid x_i)\right] + \mathrm{Var}\left(\mathbb{E}[y_i \mid x_i]\right) \approx c_\Psi^{-1} + \mathrm{Var}\big(\Psi^{(1)}(\langle x_i, \beta\rangle)\big). \tag{11}$$

It follows that the initialization $c = 2/\mathrm{Var}(y_i)$ is reasonable as long as we have $c_\Psi^{-1} \approx \mathbb{E}\left[\mathrm{Var}(y_i \mid x_i)\right]$ not much smaller than $\mathrm{Var}\big(\Psi^{(1)}(\langle x_i, \beta\rangle)\big)$. Our experiments show that SLS is very robust to initialization.

In Figure 1, we compare the performance of our SLS estimator to that of the MLE in a GLM optimization problem, when both are used to analyze synthetic data generated from a logistic regression model under general Gaussian design with randomly generated covariance matrix. The left plot shows the computational cost of obtaining both estimators as n increases for fixed $p = 200$. The middle and the right plots show the accuracy of the estimators where the latter is in log-scale. In the regime $n \gg p \gg 1$ — where the MLE is hard to compute — the MLE and the SLS achieve the same accuracy, yet SLS has significantly smaller computation time. We refer the reader to Section 5 for theoretical results characterizing the finite sample behavior of the SLS.

5. Theoretical Results

In this section, we use the zero-bias transformations (Goldstein and Reinert, 1997) to generalize the equivalence relation given in the previous section to the settings where the covariates are non-Gaussian.

Definition 2 *Let z be a random variable with mean 0 and variance σ^2. Then, there exists a random variable z^* that satisfies $\mathbb{E}[zf(z)] = \sigma^2\mathbb{E}[f^{(1)}(z^*)]$, for all differentiable functions f. The distribution of z^* is said to be the z-zero-bias distribution.*

The existence of z^* in Definition 2 is a consequence of Riesz representation theorem (Goldstein and Reinert, 1997). The normal distribution is the unique distribution whose zero-bias

transformation is itself (i.e. the normal distribution is a fixed point of the operation mapping the distribution of z to that of z^* – which is basically Stein's lemma).

To provide some intuition behind the usefulness of the zero-bias transformation, we refer back to the proof of Proposition 1. For simplicity, assume that the covariate vector x_i has iid entries with mean 0, and variance 1. Then the zero-bias transformation applied to the j-th normal equation in (7) yields

$$\underbrace{\mathbb{E}\left[y_i x_{ij}\right] = \mathbb{E}\left[x_{ij}\Psi^{(1)}\big(x_{ij}\beta_j + \Sigma_{k\neq j} x_{ik}\beta_k\big)\right]}_{j\text{-th normal equation}} = \underbrace{\beta_j \mathbb{E}\left[\Psi^{(2)}\left(x^*_{ij}\beta_j + \Sigma_{k\neq j} x_{ik}\beta_{ik}\right)\right]}_{\text{Zero-bias transformation}}. \tag{12}$$

The distribution of x^*_{ij} is the x_{ij}-zero-bias distribution and is entirely determined by the distribution of x_{ij}; general properties of x^*_{ij} can be found, for example, in Chen et al. (2010). If β is well spread, it turns out that taken together, with $j = 1, \ldots, p$, the far right-hand side in (12) behaves similar to the right side of (8), with $\mathbf{\Sigma} = \mathbf{I}$; that is, the behavior is similar to the Gaussian case, where the proportionality relationship given in Proposition 1 holds. This argument leads to an approximate proportionality relationship for problems with non-Gaussian predictors, which, when carried out rigorously, yields the following result.

Theorem 3 *Suppose that the whitened covariates $w_i = \mathbf{\Sigma}^{-1/2}x_i$ are independent random vectors with mean 0, covariance $\mathbf{I}$, and sub-Gaussian norm κ_x. Furthermore, w_i's have constant first and second conditional moments, i.e., $\forall j \in [p]$ and $\tilde{\beta} = \mathbf{\Sigma}^{1/2}\beta^{\mathsf{pop}}$, $\mathbb{E}[w_{ij}|\Sigma_{k\neq j}\tilde{\beta}_k w_{ik}]$ and $\mathbb{E}[w^2_{ij}|\Sigma_{k\neq j}\tilde{\beta}_k w_{ik}]$ are deterministic, and the function $\Psi^{(2)}$ is Lipschitz continuous with constant k. Then, for $c_\Psi = 1/\mathbb{E}\left[\Psi^{(2)}(\langle x_i, \beta^{\mathsf{pop}}\rangle)\right]$, we have*

$$\left\| \frac{1}{c_\Psi} \times \beta^{\mathsf{pop}} - \beta^{\mathsf{ols}} \right\|_\infty \leq \eta \|\mathbf{\Sigma}^{1/2}\beta^{\mathsf{pop}}\|_\infty \|\beta^{\mathsf{pop}}\|_\infty, \quad \textit{where} \quad \eta = 8k\kappa_x^3\rho_\infty. \tag{13}$$

Theorem 3 is proved in Section 10 of the Appendix. It implies that the population parameters β^{ols} and β^{pop} are approximately equivalent up to a scaling factor, with a *relative* error bound of $\mathcal{O}(\|\mathbf{\Sigma}^{1/2}\beta^{\mathsf{pop}}\|_\infty)$. For the above analysis to provide a decreasing bound in the dimension p, one needs a tractability assumption on either one of the terms in the error bound. The most common way to achieve this is to introduce a structure on the inner product between x and β^{pop}, e.g. $\mathbb{E}[\langle x, \beta^{\mathsf{pop}}\rangle^2]^{1/2} = o\left(\sqrt{p}\right)$ suffices to get a decay in (13). We emphasize that it is common practice to standardize the response and the features before training any model; therefore this preliminary procedure can provide the required order and consequently the desired decay in p.

Below, we discuss two commonly encountered settings in the literature as two corollaries.

Corollary 4 *Let the assumptions of Theorem 3 hold. If we further assume that the covariates x_i are supported on a ball of radius r, and $\Sigma^{1/2}$ is diagonally dominant, we have*

$$\left\| \frac{1}{c_\Psi} \times \beta^{\mathsf{pop}} - \beta^{\mathsf{ols}} \right\|_\infty \leq \eta \frac{\|\beta^{\mathsf{pop}}\|^2_\infty}{\sqrt{p}}, \quad \textit{where} \quad \eta = 16kr\kappa_x^3\sqrt{\rho_2}\rho_\infty. \tag{14}$$

Note that we do not require an underlying model assumption between y and x. Therefore, one can always perform under the finite support assumption by first scaling the design

matrix appropriately, and then using the proposed estimator. This assumption is common in many fields, yet can appear in different forms. For example in the machine learning literature, the standard convention (the same setting in Corollary 4) is to assume that $\|x_i\|_2 \leq r$, and that $\|\beta^{\mathsf{pop}}\|_2 = \mathcal{O}\left(\sqrt{p}\right)$ (Kalai and Sastry, 2009; Shalev-Shwartz et al., 2009; Kakade et al., 2011), or the inner product is supported on an interval $|\langle x, \beta^{\mathsf{pop}}\rangle| \leq b$ almost surely (Alquier and Biau, 2013), where the latter is a more strict assumption. In the compressed sensing and other areas in statistics, a similar assumption is imposed on the covariance matrix $\mathbb{E}\left[xx^T\right] = \tilde{\mathbf{\Sigma}}/p$ which provides us with $\mathbb{E}[\|x\|_2^2] = \mathcal{O}(1)$ (Donoho et al., 2009; Bayati and Montanari, 2012; Bayati et al., 2013; Su and Candes, 2016).

On the other hand, the standard convention in the dimension reduction techniques is to assume that the covariates have norm of order $\sqrt{p}$, i.e. $\mathbb{E}\left[\|x\|_2\right] = \mathcal{O}\left(\sqrt{p}\right)$, and the coefficients have norm of order 1, i.e. $\|\beta\|_2 = \tau = \mathcal{O}(1)$ (Duan and Li, 1991; Hall and Li, 1993; Hristache et al., 2001). This setting again yields that the inner product between those two vectors are tractable and provides us with a decay in the latter term, i.e. $\|\beta^{\mathsf{pop}}\|_\infty = \mathcal{O}\left(1/\sqrt{p}\right)$. In this setting, Theorem 3 yields the following corollary.

Corollary 5 *Let the assumptions of Theorem 3 hold. If we further assume that β^{pop} is r-well-spread in the sense that for $\|\beta^{\mathsf{pop}}\|_2 = \tau$, we have $\tau/\|\beta^{\mathsf{pop}}\|_\infty = r\sqrt{p}$ for some $r \leq 1$,*

$$\left\| \frac{1}{c_\Psi} \times \beta^{\mathsf{pop}} - \beta^{\mathsf{ols}} \right\|_\infty \leq \eta \frac{\|\beta^{\mathsf{pop}}\|_\infty}{\sqrt{p}}, \quad \textit{where} \quad \eta = 8k\kappa_x^3\rho_\infty(\tau/r)\|\mathbf{\Sigma}^{1/2}\|_\infty. \tag{15}$$

In the above bound, the decay is in fact $1/p$, but we chose to state the *relative decay*. We emphasize that both settings in Corollaries 4 and 5 are equivalent and both assumptions lead a tractable inner product $\langle x, \beta^{\mathsf{pop}}\rangle$. The constants κ_x, ρ_k are invariant to the scalings considered in both settings.

The assumption that β^{pop} is well-spread can be relaxed with minor modifications. For example, if we have a sparse coefficient vector, where $\mathrm{supp}(\beta^{\mathsf{pop}}) = \{j;\ \beta^{\mathsf{pop}}_j \neq 0\}$ is the support set of β^{pop}, then Corollary 5 holds with p replaced by the size of the support set.

The assumptions on the conditional moments are the relaxed versions of assumptions that are commonly encountered in dimension reduction techniques. For example, sliced inverse regression methods assume that the first conditional moment $\mathbb{E}\left[x|\langle x, \beta\rangle\right]$ is linear in x for all β (Li and Duan, 1989; Li, 1991), which is satisfied by elliptically distributed random vectors. An important case that is not covered by these methods is the independent coordinate case, i.e., when the whitened covariates have independent, but not necessarily identical entries. This is made possible by introducing additional approximation error (i.e. suboptimality due to non-Gaussian design). We refer reader to Li and Dong (2009), for a good review of dimension reduction techniques and their corresponding assumptions. We also highlight that our moment assumptions can be relaxed further, at the expense of introducing some additional complexity into the results.

An interesting consequence of Theorem 3 and the corollaries following the theorem is that whenever an entry of β^{pop} is zero, the corresponding entry of β^{ols} has to be small, and conversely. For $\lambda \geq 0$, define the lasso coefficients

$$\beta^{\mathsf{lasso}}_\lambda = \underset{\beta\in\mathbb{R}^p}{\mathrm{argmin}} \ \frac{1}{2}\mathbb{E}\left[(y_i - \langle x_i, \beta\rangle)^2\right] + \lambda\|\beta\|_1. \tag{16}$$

Corollary 6 *Let* $\text{supp}(\beta)$ *denote the support of* β*, i.e., the set* $\{i \in [p] : \beta_i \neq 0\}$*. Given* $\|\frac{1}{c_\Psi}\beta^{\mathsf{pop}} - \beta^{\mathsf{ols}}\|_\infty \leq \delta$*, and for any* $\lambda \geq \delta$*, if* $\mathbb{E}[x_i] = 0$ *and* $\mathbb{E}\left[x_i x_i^T\right] = \mathbf{I}$*, we have*

$$\text{supp}(\beta^{\mathsf{lasso}}) \subset \text{supp}(\beta^{\mathsf{pop}}).$$

Further, if λ *and* β^{pop} *also satisfy that* $\forall j \in \text{supp}(\beta^{\mathsf{pop}})$*,* $|\beta_j^{\mathsf{pop}}| > c_\Psi(\lambda + \delta)$*, then we have* $\text{supp}(\beta^{\mathsf{lasso}}) = \text{supp}(\beta^{\mathsf{pop}})$.

We note that δ above can be obtained using Theorem 3 or its corollaries. So far in this section, we have only discussed properties of the population parameters, such as β^{pop} and β^{ols}. In the remainder of this section, we turn our attention to the results on the estimators that are the main focus of this paper; these results ultimately build on our earlier results, i.e. Theorem 3. We will be considering the setting in Corollary 5 in the rest of this section, i.e., $\|x\|_2 = \mathcal{O}\left(\sqrt{p}\right)$ and $\|\beta^{\mathsf{pop}}\|_2 = \mathcal{O}(1)$. Results presented below can be easily generalized to the setting in Corollary 4 as well.

In order to precisely describe the performance of $\hat{\beta}^{\mathsf{sls}}$, we first need bounds on the OLS estimator. The OLS estimator has been studied extensively in the literature; however, for our purposes, we find it convenient to derive a new bound on its accuracy. While we have not seen this exact bound elsewhere, it is very similar to Theorem 5 of Dhillon et al. (2013).

Proposition 7 *Assume that* $\mathbb{E}[x_i] = 0$*,* $\mathbb{E}\left[x_i x_i^T\right] = \mathbf{\Sigma}$*, and that* $\mathbf{\Sigma}^{-1/2}x_i$ *and* y_i *are sub-Gaussian with norms* κ *and* γ*, respectively. For* λ_{min} *denoting the smallest eigenvalue of* $\mathbf{\Sigma}$*, and* $|S| > \eta p$*,*

$$\left\|\hat{\beta}^{\mathsf{ols}} - \beta^{\mathsf{ols}}\right\|_2 \leq \eta \lambda_{min}^{-1/2}\sqrt{\frac{p}{|S|}}, \tag{17}$$

with probability at least $1 - 3e^{-p}$*, where* η *depends only on* γ *and* κ*.*

Proposition 7 is proved in Section 10. Our main result on the performance of $\hat{\beta}^{\mathsf{sls}}$ is given next.

Theorem 8 *Let the assumptions of Theorem 3, Corollary 5, and Proposition 7 hold. Further assume that* $\left|\Psi^{(2)}\right| \leq b$ *and for some* $\bar{c}$ *and* ϵ *the function* $f(z) = z\mathbb{E}\left[\Psi^{(2)}(\langle x, \beta^{\mathsf{ols}}\rangle z)\right]$ *satisfies* $f(\bar{c}) > 1 + \epsilon$*, and the derivative of* f *in the interval* $[0, \bar{c}]$ *does not change sign, i.e., its absolute value is lower bounded by* $\upsilon > 0$*. Then, for* n *and* $|S|$ *sufficiently large, with probability at least* $1 - 4e^{-p}$*, we have*

$$\left\|\hat{\beta}^{\mathsf{sls}} - \beta^{\mathsf{pop}}\right\|_\infty \leq \eta_1 \frac{\|\beta^{\mathsf{pop}}\|_\infty}{\sqrt{p}} + \eta_2\sqrt{\frac{p}{|S|}}, \tag{18}$$

where the constants η_1 *and* η_2 *are defined by*

$$\eta_1 = \eta k\bar{c}\rho_\infty(\tau/r)\|\mathbf{\Sigma}^{1/2}\|_\infty \tag{19}$$

$$\eta_2 = \eta\bar{c}\left(\lambda_{min}^{-1/2} + k\upsilon^{-1}\max\left\{(\|\Sigma^{1/2}\beta^{\mathsf{ols}}\|_2 + 1)\|\Sigma\|_2 + b/k, \bar{c}\right\}\right), \tag{20}$$

and $\eta > 0$ *is a constant depending on* κ *and* γ*.*

Note that the convergence rate of the upper bound in (18) depends on the sum of the two terms, both of which are functions of the data dimensions n and p. The first term on the right in (18) comes from Theorem 3 and Corollary 5, which bounds the discrepancy between $c_\Psi \times \beta^{\mathsf{ols}}$ and β^{pop}. This term is small when p is large, and it does not depend on the number of observations n. If n and p grow together, then p has to grow at a smaller rate to achieve convergence. We note that the actual decay of this term is $\mathcal{O}(1/p)$ (since $\|\beta^{\mathsf{pop}}\|_\infty = \mathcal{O}\left(1/\sqrt{p}\right)$), but we state the relative decay which is $\mathcal{O}\left(1/\sqrt{p}\right)$.

The second term in the upper bound (18) comes from estimating β^{ols} and c_Ψ. This term is increasing in p, which reflects the fact that estimating β^{pop} is more challenging when p is large. As expected, this term is decreasing in n and $|S|$, i.e. larger sample size yields better estimates. When the full OLS solution is used ($|S| = n$), the second term becomes $\mathcal{O}(\sqrt{p/n})$, which suggests that the sample size should be at least of order p for good performance. We note that in the case of Gaussian covariates, $\sqrt{n}$-consistency follows since we don not have a bias term.

In certain loss functions, the uniform boundedness assumption on $\Psi^{(2)}$ may be too strict. However, this assumption can be removed by making an additional assumption that the covariates have bounded support. This yields the same decay rate $\mathcal{O}\left(\sqrt{p/n}\right)$ in the second term. However, if one needs to preserve the sub-Gaussian assumption on the covariates, then the above analysis – when carried out without the uniform bound assumption on $\Psi^{(2)}$ – yields the following result. We note that we still require the uniform Lipschitz assumption but one can always utilize the local Lipschitz properties of the function $\Psi^{(2)}$.

Theorem 9 *Let the assumptions of Theorem 3, Corollary 5, and Proposition 7 hold with* $\mathbb{E}[\|\mathbf{\Sigma}^{-1/2}x\|_2] = \tilde{\mu}\sqrt{p}$*. Further assume that* $\sup_{\beta:\|\beta - \Sigma^{1/2}\beta^{\mathsf{ols}}\|_2 \leq 1}\|\Psi^{(2)}(\langle x_i, \beta\rangle)\|_{\psi_2} \leq \kappa_g$ *and for some* $\bar{c}$ *and* ϵ *the function* $f(z) = z\mathbb{E}\left[\Psi^{(2)}(\langle x, \beta^{\mathsf{ols}}\rangle z)\right]$ *satisfies* $f(\bar{c}) > 1 + \epsilon$*, and the derivative of* f *in the interval* $[0, \bar{c}]$ *does not change sign, i.e., its absolute value is lower bounded by* $\upsilon > 0$*. Then, for* n *and* $|S|$ *sufficiently large, with probability at least* $1 - 5e^{-p}$*, we have*

$$\left\|\hat{\beta}^{\mathsf{sls}} - \beta^{\mathsf{pop}}\right\|_\infty \leq \eta_1 \frac{\|\beta^{\mathsf{pop}}\|_\infty}{\sqrt{p}} + \eta_2\sqrt{\frac{p}{\min\{n/\log(n), |S|\}}}, \tag{21}$$

where the constants η_1 *and* η_2 *are defined by*

$$\eta_1 = \eta k\bar{c}\rho_\infty(\tau/r)\|\mathbf{\Sigma}^{1/2}\|_\infty \tag{22}$$

$$\eta_2 = \eta\bar{c}\left(\lambda_{min}^{1/2} + \upsilon^{-1}\|\beta^{\mathsf{ols}}\|_\infty\max\{(\kappa_g + k/\tilde{\mu}), k\bar{c}\kappa_x\}\right), \tag{23}$$

and $\eta > 0$ *is a constant depending on* κ_x *and* γ*.*

We observe that this time when the full OLS solution is used ($|S| = n$), the second term becomes $\mathcal{O}(\sqrt{p\log(n)/n})$, which is worse than the rate given in Theorem 8 by a logarithmic factor. In this setting, we require that $n/\log(n)$ should be at least of order p for good performance. Also, note that there is a theoretical threshold for the sub-sampling size $|S|$, namely $\mathcal{O}(n/\log(n))$, beyond which further sub-sampling provides no improvement.

When the covariates are Gaussian, the Lipschitz assumption on the function $\Psi^{(2)}$ is enough the guarantee the sub-Gaussianity of $\Psi^{(2)}(\langle x_i, \beta\rangle)$. However, we note that the condition given in the Theorem is more general.

5.1. Convergence in ℓ_2-norm

In the sequel, we assume that the full OLS solution is used ($|S| = n$). Using Proposition 7 and Lemma 16, we can show that the ℓ_2-distance between the estimator $\hat{\beta}^{\mathsf{sls}}$ and the population parameter $c_\Psi \beta^{\mathsf{ols}}$ converges to 0 with rate $\sqrt{n}$ when p is fixed. It is straightforward to see that under Gaussian design, this leads to consistency in ℓ_2-norm with convergence rate $\sqrt{n}$. However, the quantity $c_\Psi \beta^{\mathsf{ols}}$ is in general different from β^{pop} under general sub-Gaussian design; thus, in such cases consistency cannot be achieved in the classical setting where p is fixed (see i.e. Claim 1 in Chen and Banerjee (2017)). In such a scenario, Theorem 3 can be used to obtain an upper bound on the suboptimality resulting from general sub-Gaussian design.

In order to talk about consistency when $n, p \to \infty$ under ℓ_2-norm, a normalization by one of the growing dimensions is required, see e.g. Bayati and Montanari (2012); Bayati et al. (2013). Hence, in this regime, we focus on the quantity $\frac{1}{\sqrt{p}}\|\hat{\beta}^{\mathsf{sls}} - \beta^{\mathsf{pop}}\|_2$ under the assumptions of Corollary 4 with $\mathbf{\Sigma} = \mathbf{I}$. Since we have $\sqrt{n}$-convergence of our estimator to the population parameter $c_\Psi \beta^{\mathsf{ols}}$, consistency reduces to showing that the suboptimality term $\frac{1}{\sqrt{p}}\left\|c_\Psi \beta^{\mathsf{ols}} - \beta^{\mathsf{pop}}\right\|_2$ converges to 0. This term can be controlled with additional sparsity-type structural assumptions on the problem. Below, we discuss two scenarios that may lead to such a setting.

Sparsity. Assume that β^{pop} has s non-zero terms, i.e., $\|\beta^{\mathsf{pop}}\|_0 = s$. Following the same proof technique of Theorem 3, we can show that

$$\frac{1}{\sqrt{p}}\left\|c_\Psi \beta^{\mathsf{ols}} - \beta^{\mathsf{pop}}\right\|_2 \le C\sqrt{\frac{s}{p}}\|\beta^{\mathsf{pop}}\|_\infty, \tag{24}$$

for some constant $C > 0$. As long as the non-sparsity level satisfies $s = o(p)$, we can attain consistency in ℓ_2-norm. The rate of convergence will be determined by the growing rates of $n, p(n)$ and $s(n)$. For example, to attain $\sqrt{n}$-convergence, one needs $s = \mathcal{O}(1)$ and $p = \mathcal{O}(n)$, but milder growth conditions can still lead to consistency at the expense of slower convergence.

Normality. Similarly, one can assume that a subset of features with size s follow a non-Gaussian distribution, and the remaining $p - s$ features are normally distributed. The same argument above can also provides consistency in ℓ_2-norm – again in this case and the convergence rate will be determined by the ratio $\sqrt{s/p}$.

5.2. Verifying the key assumptions

The assumption on $f(z) = z\mathbb{E}\left[\Psi^{(2)}(\langle x, \beta^{\mathsf{ols}}\rangle, z)\right]$ appears in both theorems and depends both on the function Ψ and the underlying covariate distribution. Therefore, in order to verify this assumption rigorously, one needs to know the distribution of the covariates which is not always available. However, a straightforward way to verify this assumption is to use the sample mean estimate $\hat{f}(z) = z\frac{1}{n}\sum_{i=1}^n \Psi^{(2)}(\langle x_i, \hat{\beta}^{\mathsf{ols}}\rangle z)$ using the dataset at hand to estimate $f(z)$. In Figure 2, we use this technique to verify that our assumptions are valid for two commonly used loss functions, namely, boosting and logistic regression when the covariates are normally distributed $x \sim \mathsf{N}(0, \frac{1}{p}\mathbf{I})$ and $\|\beta^{\mathsf{ols}}\|_2 = \sqrt{p}/4$. We observe through the figure that for both loss functions, one can easily verify that the conditions are satisfied.

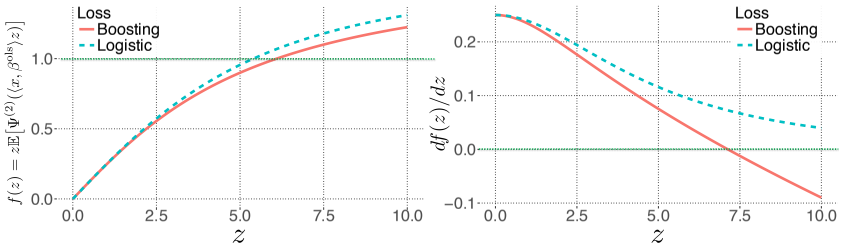


Figure 2: We demonstrate how to validate our assumption on $f(z) = z\mathbb{E}\left[\Psi^{(2)}(\langle x, \beta^{\mathsf{ols}}\rangle z)\right]$ on two commonly used loss functions. Left plot shows $f(z)$, whereas right plots shows its derivative. In these plots, covariates are generated from $x \sim \mathsf{N}(0, \mathbf{I}/p)$, and $\|\beta^{\mathsf{ols}}\|_2 = \sqrt{p}/4$.

The assumption on $f(z)$ can be verified by a more rigorous treatment. For simplicity, we again consider the compressed sensing setting where we assume $x \sim \mathsf{N}(0, \frac{1}{p}\mathbf{I})$, and this time we let $\|\beta^{\mathsf{pop}}\|_2 = c_\Psi \left\|\beta^{\mathsf{ols}}\right\|_2 \approx \sqrt{p}$. In order to obtain a scaling constant $c_\Psi \le 20$, assume that we have $\left\|\beta^{\mathsf{ols}}\right\|_2 = \sqrt{p}/20$.

Boosting loss: In this case, we have $\Psi(z) = z/2 + \sqrt{1 + z^2/4}$ which yields

$$\Psi^{(2)}(z) = \frac{1}{4}(1 + z^2/4)^{-3/2} \quad \text{and} \quad \Psi^{(4)}(z) = \frac{3}{64}\frac{5z^2(1 + z^2/4)^{-2} - 4}{(1 + z^2/4)^{5/2}}. \tag{25}$$

We notice that both of these functions are even, i.e., $\Psi^{(2)}(z) = \Psi^{(2)}(-z)$, and $\Psi^{(4)}(z) = \Psi^{(4)}(-z)$. We use the local convexity for $z \ge 0$ around $z = 2$, and obtain $\Psi^{(2)}(z) \ge a - bz$ where $a = \Psi^{(2)}(2) - 2\Psi^{(3)}(2)$ and $b = -\Psi^{(3)}(2)$. For $W \sim \mathsf{N}(0, 1)$ and ϕ denoting the standard normal density, we write

$$\begin{aligned} f(z) =& z\mathbb{E}\left[\Psi^{(2)}(\langle x, \beta^{\mathsf{ols}}\rangle z)\right] = z\mathbb{E}\left[\Psi^{(2)}(Wz/20)\right], \\ =& 2z\int_0^\infty \Psi^{(2)}(wz/20)\phi(w)dw \ge 2z\int_0^{20a/bz}(a - bwz/20)\phi(w)dw, \\ &= 2z\left\{a\Phi(20a/bz) - \frac{a}{2} - \frac{bz}{20\sqrt{2\pi}}(1 - e^{-200a^2/b^2z^2})\right\}. \end{aligned} \tag{26}$$

Using the above bound, we observe that for $\bar{c} = 6$ and $\delta = .23$, we have $f(\bar{c}) > 1 + \delta$.

Next, we verify the derivative condition on f, that is, $f^{(1)}$ does not change sign in the interval $[0, \bar{c}]$. By the Stein's lemma, we write

$$f^{(1)}(z) = \mathbb{E}\left[\Psi^{(2)}(Wz/20)\right] + (z/20)^2\mathbb{E}\left[\Psi^{(4)}(Wz/20)\right]. \tag{27}$$

Using the previous inequalities together with the above expression, we have

$$\begin{aligned} f^{(1)}(z) \ge& \mathbb{E}\left[\Psi^{(2)}(Wz/20)\right] - \frac{9}{100}\left|\Psi^{(4)}\right| \\ \ge& 2\left\{a\Phi(20a/bz) - \frac{a}{2} - \frac{bz}{20\sqrt{2\pi}}(1 - e^{-200a^2/b^2z^2})\right\} - \frac{27}{1600} \ge 0.188, \end{aligned} \tag{28}$$

which verifies the derivative assumption on $f(z)$.

Logistic regression: In this case, we have $\Psi(z) = \log(1 + e^z)$ which yields

$$\Psi^{(2)}(z) = \frac{e^z}{(1+e^z)^2} \quad \text{and} \quad \Psi^{(4)}(z) = \frac{e^z(1 - 4e^z + e^{2z})}{(1+e^z)^4}. \tag{29}$$

As before, both $\Psi^{(2)}$ and $\Psi^{(4)}$ are even functions. Again, appealing to the local convexity properties, for $z \geq 0$ around $z = 2.5$, we obtain $\Psi^{(2)}(z) \geq a - bz$ where $a = \Psi^{(2)}(2.5) - 2.5\Psi^{(3)}(2.5)$ and $b = -\Psi^{(3)}(2.5)$. Using (26), we obtain

$$f(z) \geq 2z \left\{ a\Phi\big(20a/bz\big) - \frac{a}{2} - \frac{bz}{20\sqrt{2\pi}}(1 - e^{-200a^2/b^2z^2}) \right\}. \tag{30}$$

Using the above bound, we see that for $\bar{c} = 6$, and $\delta = .22$ we have $f(\bar{c}) > 1 + \delta$.

For the derivative bound, we again use (27), and obtain that for $z \in [0, \bar{c}]$,

$$f^{(1)}(z) \geq \mathbb{E}\left[\Psi^{(2)}(Wz/20)\right] - \frac{9}{100}\left|\Psi^{(4)}\right| \geq \mathbb{E}\left[\Psi^{(2)}(Wz/20)\right] - \frac{9}{800} \geq 0.19, \tag{31}$$

which verifies the assumption.

6. Converting One GLM to Another

In this section, we assume that one has already obtained a set of coefficients by minimizing a certain GLM loss, but wants to minimize another GLM loss using the same dataset. It is often the case that a practitioner would like to change the loss function (or equivalently the model) based on its performance. However, when the dataset is large, training a new model from scratch is extremely time consuming. In the following, we will use the proportionality relation to transition between different loss functions.

Assume that a practitioner fitted a GLM using the loss function (or cumulant generating function) Ψ_1, but they would like to train a new model based on the second loss function Ψ_2. Instead of minimizing the new loss based on Ψ_2, one can exploit the proportionality relation and obtain the coefficients for the new GLM problem. Denote by β_1^{pop} and β_2^{pop} the GLM coefficients corresponding to the loss functions Ψ_1 and Ψ_2, respectively. Using the results in the previous sections, we have

$$\frac{1}{c_{\Psi_1}}\beta_1^{\mathsf{pop}} = \frac{1}{c_{\Psi_2}}\beta_2^{\mathsf{pop}} = \beta^{\mathsf{ols}},$$

for the normal case, that is, both coefficients are proportional to the OLS coefficients which does not depend on the loss function. Therefore, these coefficients β_1^{pop} and β_2^{pop} are also proportional to each other and we can write

$$\beta_2^{\mathsf{pop}} = \frac{c_{\Psi_2}}{c_{\Psi_1}}\,\beta_1^{\mathsf{pop}} \coloneqq \rho\,\beta_1^{\mathsf{pop}}, \tag{32}$$

where the proportionality constant between two different GLM types turns out to be the ratio between c_{Ψ_1} and c_{Ψ_2}, i.e. $\rho = c_{\Psi_2}/c_{\Psi_1}$. Using the definition of c_{Ψ_2}, we write

$$\begin{aligned} 1 &= c_{\Psi_2}\,\mathbb{E}\left[\Psi_2^{(2)}(\langle x, \beta_2^{\mathsf{pop}}\rangle)\right], \\ &= c_{\Psi_1}\rho\,\mathbb{E}\left[\Psi_2^{(2)}(\langle x, \beta_1^{\mathsf{pop}}\rangle\rho)\right]. \end{aligned}$$

Algorithm 2 Conversion from one GLM to another

Input: Data $(y_i, x_i)_{i=1}^n$, and $\hat{\beta}_1^{\mathrm{glm}}$

Step 1. Compute $\hat{y} = \mathbf{X}\hat{\beta}_1^{\mathrm{glm}}$, and $\kappa = \frac{1}{n}\sum_{i=1}^n \Psi_1^{(2)}(\hat{y}_i)$.

Step 2. Solve the following equation for $\rho \in \mathbb{R}$: $\kappa = \frac{\rho}{n}\sum_{i=1}^n \Psi_2^{(2)}(\hat{y}_i\rho)$

Use Newton's root-finding method:

Initialize $\rho = 1$;

Repeat until convergence:

$$\rho \leftarrow \rho - \frac{\rho\,\frac{1}{n}\sum_{i=1}^n \Psi_2^{(2)}(\rho\,\hat{y}_i) - \kappa}{\frac{1}{n}\sum_{i=1}^n \left\{\Psi_2^{(2)}(\rho\,\hat{y}_i) + \rho\,\hat{y}_i\Psi_2^{(3)}(\rho\,\hat{y}_i)\right\}}.$$

Output: $\hat{\beta}_2^{\mathrm{glm}} = \rho \times \hat{\beta}_1^{\mathrm{glm}}$.

Dividing the both sides by c_{Ψ_1} and using the equality $c_{\Psi_1}^{-1} = \mathbb{E}\left[\Psi_1^{(2)}(\langle x, \beta_1^{\mathsf{pop}}\rangle)\right]$, we obtain

$$\mathbb{E}\left[\Psi_1^{(2)}(\langle x, \beta_1^{\mathsf{pop}}\rangle)\right] = \rho\,\mathbb{E}\left[\Psi_2^{(2)}(\langle x, \beta_1^{\mathsf{pop}}\rangle\rho)\right].$$

The above equation only involves β_1^{pop} as the coefficients (which is assumed to be already known/obtained by the practitioner). Therefore, if we solve it for the ratio ρ, we can estimate β_2^{pop} by simply using the proportionality relation given in (32).

The procedure described above is summarized as Algorithm 2. We emphasize that this procedure does not require the computation of the OLS estimator which was the main cost of SLS, and it only requires a per-iteration cost of $\mathcal{O}(n)$. In other words, conversion from one GLM type to another is easier compared to obtaining the GLM coefficients from scratch.

7. Binary Classification with Proper Scoring Rules

In this section, we assume that for $i \in [n]$, the response is binary $y_i \in \{0, 1\}$. The binary classification problem can be described by the following minimization of an empirical risk

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}}\ \frac{1}{n}\sum_{i=1}^n \ell(y_i; q(\langle x_i, \beta\rangle)), \tag{33}$$

where ℓ and q are referred to as the loss and the link functions, respectively. There are various loss functions that are used in practice. Examples include log-loss, boosting loss, square loss etc (See Table 1). As before, we constrain our analysis on the canonical links. The concept of canonical links for binary classification is introduced by Buja et al. (2005), and it is quite similar to the generalized linear problems.

For any given loss function, we define the partial losses $\ell_k(\cdot) = \ell(y = k; \cdot)$ for $k \in \{0, 1\}$. Since we have a binary response variable, we can write any loss in the following format

$$\begin{aligned} \ell(y; q) =& y\ell_1(q) + (1 - y)\ell_0(q), \\ =& y\,(\ell_1(q) - \ell_0(q)) + \ell_0(q). \end{aligned}$$

Table 1: Common loss functions and their inverse canonical links

Name	Loss function: $\ell(y;q)$	Weight: $w(q)$	Inverse Canonical link: q
Log-loss	$-y\log(q)-(1-y)\log(1-q)$	$\frac{1}{q(1-q)}$	$\frac{1}{1+\exp(-z)}$
Boosting loss	$y(q^{-1}-1)^{1/2}+(1-y)(q^{-1}-1)^{-1/2}$	$\frac{1}{[q(1-q)]^{3/2}}$	$\frac{1}{2}+\frac{z/4}{2(z^2/4+1)^{1/2}}$
Square loss	$y(1-q)^2+(1-y)q^2$	1	$\frac{1+z}{2}$

The above formulation is of the form of a generalized linear problem. Before moving forward, we recall the concept of proper scoring in binary classification, which is sometimes referred to as Fisher consistency.

Definition 10 (Proper scoring rules) *Assume that $y \sim$ Bernoulli(η). If the expected loss $\mathbb{E}[\ell(y,q)]$ is minimized by $q=\eta$ for all $\eta\in(0,1)$, we call the loss function a proper scoring rule.*

The following theorem by Schervish (1989) provides a methodology for constructing a loss function for the proper scoring rules.

Theorem 11 (Schervish (1989)) *Let $w(dt)$ be a positive measure on $(0,1)$ that is finite on interval $(\epsilon,1-\epsilon)$ $\forall\epsilon>0$. Then the following defines a proper scoring rule*

$$\ell_1(q)=\int_q^1(1-t)w(dt),\ \text{and}\ \ell_0(q)=\int_0^q tw(dt).$$

The measure $w(dt)$ uniquely defines the loss function (generally referred to as the weight function, since all losses can be written as weighted average of cost weighted misclassification error (Buja et al., 2005; Reid and Williamson, 2010)). Examples of weight functions is given in Table 1. The above theorem has many interesting interpretations; one that is most useful to us is that $\ell_0^{(1)}(q)=qw(q)$.

The notion of canonical links for proper scoring rules are introduced by Buja et al. (2005), which corresponds to the notion of matching loss (Helmbold et al., 1999; Reid and Williamson, 2010). The derivation of canonical links stems from the Hessian of the above minimization, which remedies two potential problems: non-convexity and asymptotic variance inflation. It turns out that by setting $w(q)q^{(1)}$ as constant, one can remedy both problems (Buja et al., 2005). We will skip the derivation and, without loss of generality, assume that the canonical link-loss pair satisfies $w(q)q^{(1)}=1$. Note that any loss function has a natural canonical link. The following Theorem summarizes this concept.

Theorem 12 (Buja et al. (2005)) *For proper scoring rules with $w>0$, there exists a canonical link function which is unique up to addition and multiplication by constants. Conversely, any link function is canonical for a unique proper scoring rule.*

The canonical link for a given loss function can be explicitly derived from the equation $w(q)q^{(1)}=1$. We have provided some examples in Table 1. Using the definition of canonical

link for proper scoring rules, we write the normal equations $\frac{\mathrm{d}}{\mathrm{d}\beta}\mathbb{E}\left[\ell(y,q(\langle x,\beta\rangle))\right]=0$ as

$$\begin{aligned}
&\mathbb{E}\left[xq^{(1)}(\langle x,\beta\rangle)\ell_0^{(1)}(q(\langle x,\beta\rangle))\right]=\mathbb{E}\left[yxq^{(1)}(\langle x,\beta\rangle)\left(\ell_0^{(1)}(q(\langle x,\beta\rangle))-\ell_1^{(1)}(q(\langle x,\beta\rangle))\right)\right],\\
&\implies \mathbb{E}\left[xq^{(1)}(\langle x,\beta\rangle)q(\langle x,\beta\rangle)w(q(\langle x,\beta\rangle))\right]=\mathbb{E}\left[yxq^{(1)}(\langle x,\beta\rangle)w(q(\langle x,\beta\rangle))\right],\\
&\implies \mathbb{E}\left[xq(\langle x,\beta\rangle)\right]=\mathbb{E}\left[yx\right],\\
&\implies \boldsymbol{\Sigma}\beta\mathbb{E}\left[q^{(1)}(\langle x,\beta\rangle)\right]=\mathbb{E}\left[yx\right].
\end{aligned}$$

The last equation provides us with the analog of the proportionality relation we observed in generalized linear problems. In this case, we observe that the proportionality constant becomes $1/\mathbb{E}\left[q^{(1)}(\langle x,\beta\rangle)\right]$. Therefore, our algorithm can be used to obtain a fast training procedure for the binary classification problems under canonical links as well.

7.1. Canonicalization of the Square Loss

Consider a minimization problem of the following form

$$\underset{\beta}{\text{minimize}}\ \frac{1}{n}\sum_{i=1}^n[y_i-f(\langle x_i,\beta\rangle)]^2. \tag{34}$$

The above problem is commonly encountered in many machine learning tasks – specifically, in the context of neural networks, the function f is called the activation function. Here, we consider a toy example to demonstrate how our methodology can be useful in a minimization problem of the above form.

We first use Taylor series expansion around a point θ (which should be close to $\langle x,\beta\rangle$), in order to approximate the function $f(z)$ with a linear function around $f(\theta)$. We write

$$\min_\beta\mathbb{E}\left[(y-f(\langle x,\beta\rangle))^2\right]\approx\min_\beta\mathbb{E}\left[f(\langle x,\beta\rangle)^2-2y\langle x,\beta\rangle f'(\theta)\right]. \tag{35}$$

The above approximation yields

$$\Psi(z)=\frac{f(z)^2}{2f'(\theta)}, \tag{36}$$

and the proportionality relation given in previous sections would hold for the approximation, which will be accurate when the activation function is smooth around the user-specified point θ. This method can be used to derive proportionality relations for GLMs with non-canonical links (conditional on the link being nice), and also may be of interest in non-convex optimization, e.g., reliable initialization.

8. Experiments

8.1. Numerical Analysis of Convergence

This section contains the results of numerical studies that demonstrate the convergence results presented in Section 5. We first demonstrate the convergence of the population

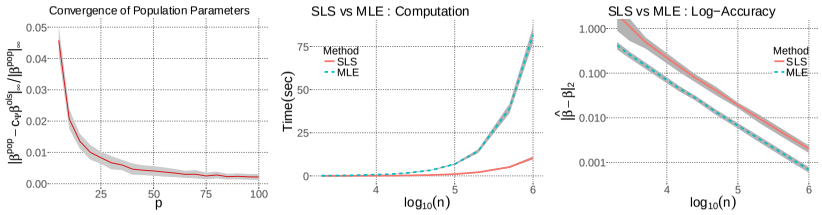


Figure 3: Logistic regression with iid Bernoulli ± 1 design. The left plot shows the relative error between the population parameters. The middle plot shows the computational cost (time) for finding the MLE and SLS as n grows and $p = 200$. The right plot depict the accuracy of the estimators in log scale. In the regime where the MLE is expensive to compute, the SLS is found much more rapidly and has the same accuracy. `R`'s built-in functions are used to find the MLE.

parameters with respect to the dimension. In this experiment, we estimate the relative error $\left\|\beta^{\mathsf{pop}} - c_\Psi \beta^{\mathsf{ols}}\right\|_\infty / \left\|\beta^{\mathsf{pop}}\right\|_\infty$ using a Monte-Carlo simulation. The dimension takes values from $p \in \{5, 10, 15, ..., 100\}$, and for each fixed p, we choose the sample size $n(p) = Kp$ where $K = 10{,}000$. $\beta^{\mathsf{pop}} = (1, 1, ..., 1)/\sqrt{p}$ is fixed throughout this experiment. Covariates are generated from a Bernoulli distribution $\mathbb{P}(x_{ij} = \pm 1) = 0.5$, and the response is generated from the logistic model. We estimate β^{ols} using coordinatewise average of the ordinary least squares estimator over the whole dataset. The scaling constant is estimated by the inverse of $\frac{1}{n}\sum_{i=1}^n \Psi^{(2)}(\langle x_i, \beta^{\mathsf{pop}}\rangle)$. We repeat this experiment 100 times for each p, and the results are reported in left plot in Figure 3. We observe that the relative error decreases with increasing dimension.

The second experiment demonstrates the effect of sample size. In this experiment, we fix $p = 200$, and the sample size varies $n \in 10^3 \times \{2, 3, 5, 15, 25, 50, 100, 200, 500, 1000\}$. Covariates are again generated from Bernoulli distribution $\mathbb{P}(x_{ij} = \pm 1) = 0.5$, and the response is generated from the logistic model. Maximum likelihood estimator is obtained through `R`'s built-in function `glm`. We report the computation time of these estimators as well as their estimation error in the middle and the right plots in Figure 3, respectively. We observe that MLE performs consistently better in terms of test error, yet it is significantly expensive to compute when n gets larger. In the large-scale regime, the difference between the achieved test errors becomes negligible as seen in the right plot. This is the regime that SLS provides significant advantage. We also emphasize that MLE and SLS has the same convergence rate as can be observed in the right plot.

8.2. Comparisons With Other Optimization Methods

This section contains the results of a variety of numerical studies, which show that the Scaled Least Squares estimator reaches its minimum achievable test error substantially faster than commonly used batch algorithms for finding the MLE. Both logistic and Poisson regression models (two types of GLMs) are utilized in our analyses, which are based on several synthetic and real datasets.

Below, we briefly describe the optimization algorithms for the MLE that were used in the experiments.

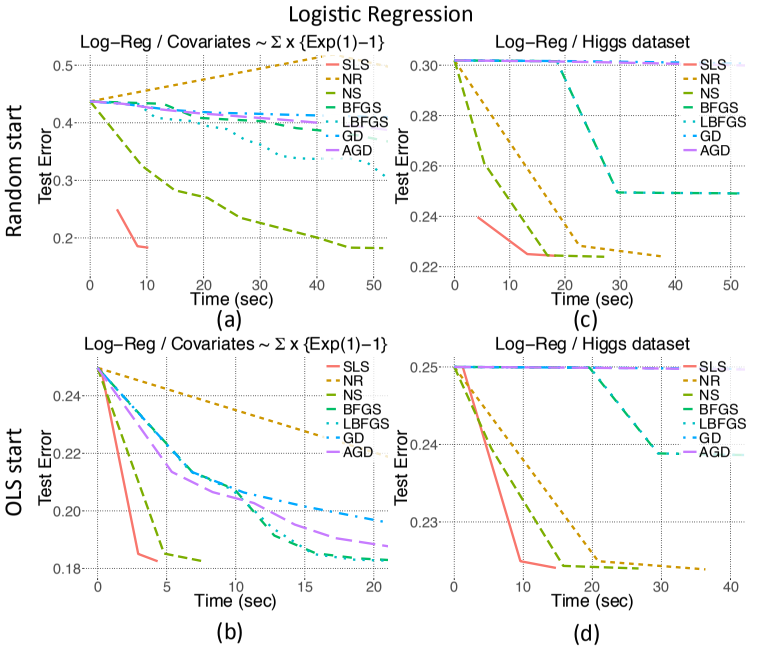


Figure 4: We compared the performance of SLS to that of MLE for the logistic regression problem on several datasets. MLE optimization is solved by various optimization algorithms. SLS is represented with red straight line. The details are provided in Table 2.

1. **Newton-Raphson (NR)** achieves locally quadratic convergence by scaling the gradient by the inverse of the Hessian evaluated at the current iterate. Computing the Hessian has a per-iteration cost of $\mathcal{O}\left(np^2\right)$, which makes it impractical for large-scale datasets.

2. **Newton-Stein (NS)** is a recently proposed second-order batch algorithm specifically designed for GLMs (Erdogdu, 2015, 2016). The algorithm uses Stein's lemma and sub-sampling to efficiently estimate the Hessian with a cost of $\mathcal{O}\left(np\right)$ per-iteration, achieving near quadratic rates.

3. **Broyden-Fletcher-Goldfarb-Shanno (BFGS)** is the most popular and stable quasi-Newton method (Nesterov, 2004). At each iteration, the gradient is scaled by a matrix that is formed by accumulating information from previous iterations and gradient computations. The convergence is locally super-linear with a per-iteration cost of $\mathcal{O}\left(np\right)$.

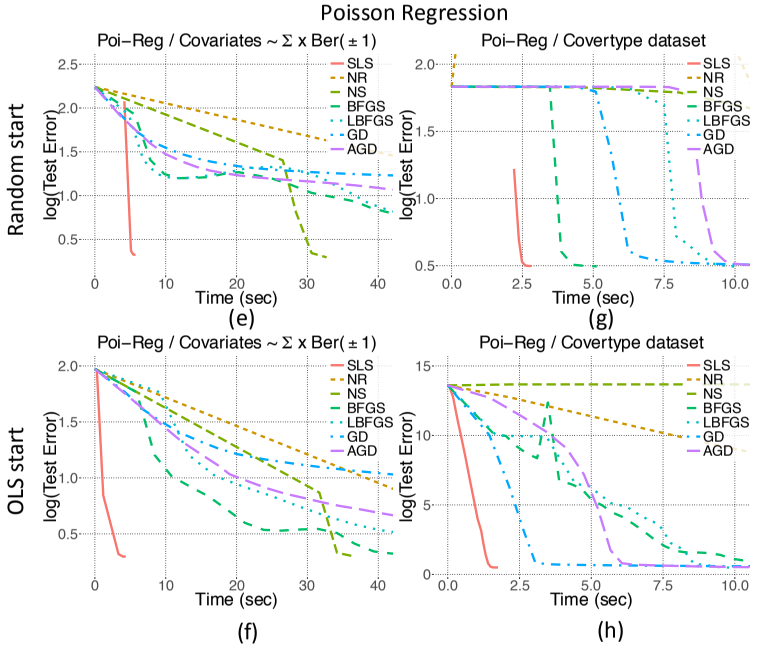


Figure 5: We compared the performance of SLS to that of MLE for the Poisson regression problem on several datasets. MLE optimization is solved by various optimization algorithms. SLS is represented with red straight line. Details are given in Table 2.

4. **Limited memory BFGS (LBFGS)** is a variant of BFGS, which uses only the recent iterates and gradients to approximate the Hessian, providing significant improvement in terms of memory usage. LBFGS has many variants; we use the formulation given in Bishop (1995).

5. **Gradient descent (GD)** takes a step in the opposite direction of the gradient, evaluated at the current iterate. Its performance strongly depends on the condition number of the design matrix. Under certain assumptions, the convergence is linear with $\mathcal{O}(np)$ per-iteration cost.

6. **Accelerated gradient descent (AGD)** is a modified version of gradient descent with an additional "momentum" term (Nesterov, 1983). Its per iteration cost is $\mathcal{O}(np)$ and its performance strongly depends on the smoothness of the objective function.

For all the algorithms for computing the MLE, the step size at each iteration is chosen via the backtracking line search (Boyd and Vandenberghe, 2004).

Table 2: Details of the experiments shown in Figures 4 and 5 with simulated and real datasets HIGGS (Baldi et al., 2014), and Covertype (Blackard and Dean, 1999)

Model	Logistic regression				Poisson regression			
Dataset	$\mathbf{\Sigma}\times$Exp(1)-1		Higgs		$\mathbf{\Sigma}\times$Ber($\pm$1)		Covertype	
Size	$n=6.0\times10^5$, $p=300$		$n=1.1\times10^7$, $p=29$		$n=6.0\times10^5$, $p=300$		$n=5.8\times10^5$, $p=53$	
Initial	Rnd	Ols	Rnd	Ols	Rnd	Ols	Rnd	Ols
Plot	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
Method↓	Time in seconds / number of iterations (to reach min test error)							
Sls	8.34/4	2.94/3	13.18/3	9.57/3	5.42/5	3.96/5	2.71/6	1.66/20
Nr	301.06/6	82.57/3	37.77/3	36.37/3	170.28/5	130.1/4	16.7/8	32.48/18
Ns	51.69/8	7.8/3	27.11/4	26.69/4	32.71/5	36.82/4	21.17/10	282.1/216
Bfgs	148.43/31	24.79/8	660.92/68	701.9/68	67.24/29	72.42/26	5.12/7	22.74/59
Lbfgs	125.33/39	24.61/8	6368.1/651	6946.1/670	224.6/106	357.1/88	10.01/14	10.05/17
Gd	669/138	134.91/25	100871/10101	141736/13808	1711/513	1364/374	14.35/25	33.58/87
Agd	218.1/61	35.97/12	2405.5/251	2879.69/277	103.3/51	102.74/40	11.28/15	11.95/25

Recall that the proposed Algorithm 1 is composed of two steps; the first finds an estimate of the OLS coefficients. This up-front computation is not needed for any of the MLE algorithms described above. On the other hand, each of the MLE algorithms requires some initial value for β, but no such initialization is needed to find the OLS estimator in Algorithm 1. This raises the question of how the MLE algorithms should be initialized, in order to compare them fairly with the proposed method. We consider two scenarios in our experiments: first, we use the OLS estimator computed for Algorithm 1 to initialize the MLE algorithms; second, we use a random initial value.

On each dataset, the main criterion for assessing the performance of the estimators is how rapidly the minimum test error is achieved. The test error is measured as the mean squared error of the estimated mean using the current parameters at each iteration on a test dataset, which is a randomly selected (and set-aside) 10% portion of the entire dataset. As noted previously, the MLE is more accurate for small n (see Figure 1). However, in the regime considered here ($n \gg p \gg 1$), the MLE and the SLS perform very similarly in terms of their error rates; for instance, on the Higgs dataset, the SLS and MLE have test error rates of 22.40% and 22.38%, respectively. For each dataset, the minimum achievable test error is set to be the maximum of the final test errors, where the maximum is taken over all of the estimation methods. Let $\mathbf{\Sigma}^{(1)}$ and $\mathbf{\Sigma}^{(2)}$ be two randomly generated covariance matrices. The datasets we analyzed were: (i) a synthetic dataset generated from a logistic regression model with iid {exponential(1)-1} predictors scaled by $\mathbf{\Sigma}^{(1)}$; (ii) the Higgs dataset (logistic regression) Baldi et al. (2014); (iii) a synthetic dataset generated from a Poisson regression model with iid binary(±1) predictors scaled by $\mathbf{\Sigma}^{(2)}$; (iv) the Covertype dataset (Poisson regression) Blackard and Dean (1999).

In all cases, the SLS outperformed the alternative algorithms for finding the MLE by a large margin, in terms of computation. Detailed results may be found in Figures 4 and 5, and Table 2. We provide additional experiments with different datasets in Section 12.

9. Discussion

In this paper, we showed that the true minimizer of a generalized linear problem and the OLS estimator are approximately proportional under the general random design setting. Using this relation, we proposed a computationally efficient algorithm for large-scale problems that achieves the same accuracy as the empirical risk minimizer by first estimating the OLS coefficients and then estimating the proportionality constant through iterations that can attain quadratic or cubic convergence rate, with only $\mathcal{O}(n)$ per-iteration cost.

We briefly mentioned that the proportionality between the coefficients holds even when there is regularization in Section 3.1. Further pursuing this idea may be interesting for large-scale problems where regularization is crucial. Another interesting line of research is to find similar proportionality relations between the parameters in other large-scale optimization problems such as support vector machines. Such relations may reduce the problem complexity significantly.

10. Proof of Main Results

In this section, we provide the details and the proofs of our technical results. For convenience, we briefly state the following definitions.

Definition 13 (Sub-Gaussian) *For a given constant κ, a random variable $x \in \mathbb{R}$ is said to be* sub-Gaussian *if it satisfies $\sup_{m\geq 1} m^{-1/2}\mathbb{E}\left[|x|^m\right]^{1/m} \leq \kappa$. Smallest such κ is the sub-Gaussian norm of x and it is denoted by $\|x\|_{\psi_2}$. Similarly, a random vector $y \in \mathbb{R}^p$ is a* sub-Gaussian vector *if there exists a constant κ' such that $\sup_{v\in S^{p-1}} \|\langle y, v\rangle\|_{\psi_2} \leq \kappa'$.*

Definition 14 (Sub-exponential) *For a given constant κ, a random variable $x \in \mathbb{R}$ is called* sub-exponential *if it satisfies $\sup_{m\geq 1} m^{-1}\mathbb{E}\left[|x|^m\right]^{1/m} \leq \kappa$. Smallest such κ is the sub-exponential norm of x and it is denoted by $\|x\|_{\psi_1}$. Similarly, a random vector $y \in \mathbb{R}^p$ is a* sub-exponential vector *if there exists a constant κ' such that $\sup_{v\in S^{p-1}} \|\langle y, v\rangle\|_{\psi_1} \leq \kappa'$.*

The proof of Theorem 3 is given next.

Proof [of Theorem 3] For simplicity, we denote the whitened covariate by $w = \mathbf{\Sigma}^{-1/2}x$. Since w is sub-Gaussian with norm κ, its j-th entry w_j has bounded third moment. That is,

$$\kappa = \sup_{\|u\|_2=1} \|\langle u, w\rangle\|_{\psi_2} \geq \|w_j\|_{\psi_2} = \sup_{m\geq 1} m^{-1/2}\mathbb{E}\left[|w_j|^m\right]^{1/m} \geq \frac{1}{\sqrt{3}}\mathbb{E}\left[|w_j|^3\right]^{1/3}, \tag{37}$$

where in the first step, we used $u = e_j$, the j-th standard basis vector. Hence, we obtain a bound on the third moment, i.e,

$$\max_j \mathbb{E}\left[|w_j|^3\right] \leq 3^{3/2}\kappa^3. \tag{38}$$

Using the normal equations, we write

$$\begin{aligned}\mathbb{E}\left[yx\right] = \mathbb{E}\left[x\Psi^{(1)}(\langle x, \beta\rangle)\right] =& \mathbf{\Sigma}^{1/2}\mathbb{E}\left[w\Psi^{(1)}(\langle w, \mathbf{\Sigma}^{1/2}\beta\rangle)\right], \\ =& \mathbf{\Sigma}^{1/2}\mathbb{E}\left[w\Psi^{(1)}(\langle w, \tilde{\beta}\rangle)\right],\end{aligned} \tag{39}$$

where we defined $\tilde{\beta} = \mathbf{\Sigma}^{1/2}\beta$. By multiplying both sides with $\mathbf{\Sigma}^{-1}$, we obtain

$$\beta^{\mathsf{ols}} = \mathbf{\Sigma}^{-1/2}\mathbb{E}\left[w\Psi^{(1)}(\langle w, \tilde{\beta}\rangle)\right]. \tag{40}$$

Now we define the partial sums $W_{-i} = \sum_{j\neq i}\tilde{\beta}_j w_j = \langle\tilde{\beta}, w\rangle - \tilde{\beta}_i w_i$. We will focus on the i-th entry of the above expectation given in (40). Denoting the zero biased transformation of w_i conditioned on W_{-i} by w_i^*, we have

$$\begin{aligned}\mathbb{E}\left[w_i\Psi^{(1)}(\langle w, \tilde{\beta}\rangle)\right] =& \mathbb{E}\left[\mathbb{E}\left[w_i\Psi^{(1)}\left(\tilde{\beta}_i w_i + W_{-i}\right)\big|W_{-i}\right]\right], \\ =& \tilde{\beta}_i\mathbb{E}\left[\Psi^{(2)}(\tilde{\beta}_i w_i^* + W_{-i})\right], \\ =& \tilde{\beta}_i\mathbb{E}\left[\Psi^{(2)}(\tilde{\beta}_i(w_i^* - w_i) + \langle w, \tilde{\beta}\rangle)\right],\end{aligned} \tag{41}$$

where in the second step, we used the assumption on conditional moments. Let $\mathbf{D}$ be a diagonal matrix with diagonal entries $\mathbf{D}_{ii} = \mathbb{E}\left[\Psi^{(2)}(\tilde{\beta}_i(w_i^* - w_i) + \langle w, \tilde{\beta}\rangle)\right]$. Using (40) together with (41), we obtain the equality

$$\beta^{\mathsf{ols}} = \mathbf{\Sigma}^{-1/2}\mathbf{D}\tilde{\beta} = \mathbf{\Sigma}^{-1/2}\mathbf{D}\mathbf{\Sigma}^{1/2}\beta. \tag{42}$$

Now, using the Lipschitz continuity assumption of the variance function, we have

$$\left|\mathbb{E}\left[\Psi^{(2)}(\tilde{\beta}_i(w_i^* - w_i) + \langle w, \tilde{\beta}\rangle)\right] - \mathbb{E}\left[\Psi^{(2)}(\langle w, \tilde{\beta}\rangle)\right]\right| \leq k|\tilde{\beta}_i|\mathbb{E}\left[|w_i^* - w_i|\right]. \tag{43}$$

In the following, we will use the properties of zero-biased transformations. Consider the quantity

$$r = \sup \frac{\mathbb{E}\left[|w_i^* - w_i|\big|W_{-i}\right]}{\mathbb{E}\left[|w_i|^3\big|W_{-i}\right]} \tag{44}$$

where w_i^* has w_i-zero biased distribution (conditioned on W_{-i}) and the supremum is taken with respect to all random variables with mean 0, standard deviation 1 and finite third moment, and w_i^* is achieving the minimal ℓ_1 coupling to w_i conditioned on W_{-i}. It is shown in Goldstein (2007) that the above bound holds for $r = 1.5$ for the unconditional zero-bias transformations. Here, we take a similar approach to show that the same bound holds for the conditional case as well. By using the triangle inequality, we have

$$\begin{aligned}\mathbb{E}\left[|w_i^* - w_i|\big|W_{-i}\right] \leq& \mathbb{E}\left[|w_i^*|\big|W_{-i}\right] + \mathbb{E}\left[|w_i|\big|W_{-i}\right] \\ \leq& \frac{1}{2}\mathbb{E}\left[|w_i|^3\big|W_{-i}\right] + \mathbb{E}\left[|w_i|^3\big|W_{-i}\right]^{1/3}.\end{aligned} \tag{45}$$

Since $\mathbb{E}\left[|w_i|^2\big|W_{-i}\right]$ is constant, it is equal to $\mathbb{E}\left[|w_i|^2\right] = 1$. This yields that the second term in the last line is upper bounded by $\mathbb{E}\left[|w_i|^3\big|W_{-i}\right]$. Consequently, by taking expectations over both hand sides we obtain that

$$\mathbb{E}\left[|w_i^* - w_i|\right] \leq 1.5\ \mathbb{E}\left[|w_i|^3\right]. \tag{46}$$

Then the right hand side of (43) can be upper bounded by

$$k|\tilde{\beta}_i|\mathbb{E}\left[|w_i^* - w_i|\right] \leq rk \max_i \left\{|\tilde{\beta}_i|\mathbb{E}\left[|w_i|^3\right]\right\} \leq 8k\kappa^3 \|\mathbf{\Sigma}^{1/2}\beta\|_\infty, \tag{47}$$

where in the second step we used the bound on the third moment given in (38). The last inequality provides us with the following result,

$$\max_i \left|\mathbf{D}_{ii} - \frac{1}{c_\Psi}\right| \leq 8k\kappa^3 \|\mathbf{\Sigma}^{1/2}\beta\|_\infty. \tag{48}$$

Finally, combining this with (40) and (42), we obtain

$$\begin{aligned}
\left\|\beta^{\mathsf{ols}} - \frac{1}{c_\Psi}\beta\right\|_\infty &= \left\|\mathbf{\Sigma}^{-1/2}\mathbf{D}\mathbf{\Sigma}^{1/2}\beta - \frac{1}{c_\Psi}\beta\right\|_\infty \\
&= \left\|\mathbf{\Sigma}^{-1/2}\left(\mathbf{D} - \frac{1}{c_\Psi}\mathbf{I}\right)\mathbf{\Sigma}^{1/2}\beta\right\|_\infty, \\
&\leq \max_i \left|\mathbf{D}_{ii} - \frac{1}{c_\Psi}\right| \left\|\mathbf{\Sigma}^{1/2}\right\|_\infty \left\|\mathbf{\Sigma}^{-1/2}\right\|_\infty \|\beta\|_\infty, \\
&\leq 8k\kappa^3 \rho_\infty \|\mathbf{\Sigma}^{1/2}\|_\infty \|\beta^{\mathsf{pop}}\|_\infty^2,
\end{aligned} \tag{49}$$

which completes the proof. ■

Proof [of Corollary 4] Due to diagonal dominance property, we have

$$\|\Sigma^{1/2}\|_\infty = \max_i \sum_{j=1}^p \left|\Sigma_{ij}^{1/2}\right| \leq 2 \max_i \Sigma_{ii}^{1/2} \leq 2\|\Sigma\|_2^{1/2}. \tag{50}$$

Since we have $\|x\|_2 \leq r$, we write

$$r^2 \geq \mathbb{E}\left[\|x\|_2^2\right] = \mathrm{Tr}\left(\Sigma\right) \geq p\|\Sigma\|_2 / \rho_2. \tag{51}$$

Combining this with the previous inequality we obtain

$$\|\Sigma^{1/2}\|_\infty \leq 2r\sqrt{\rho_2/p}. \tag{52}$$

Finally, the result follows from using the last bound in Theorem 3. ■

Proof [of Corollary 5] The result directly follows from using the r-well-spreadness assumption in Theorem 3. ■

Proof [of Proposition 7] For convenience, we denote the whitened covariates by $w_i = \mathbf{\Sigma}^{-1/2}x_i$. We have $\mathbb{E}\left[w_i\right] = 0$, $\mathbb{E}\left[w_i w_i^T\right] = \mathbf{I}$, and $\|w_i\|_{\psi_2} \leq \kappa$. Also denote the sub-sampled covariance matrix with $\widehat{\mathbf{\Sigma}} = \frac{1}{|S|}\sum_{i\in S} x_i x_i^T$, and its whitened version as $\widetilde{\mathbf{\Sigma}} = \frac{1}{|S|}\sum_{i\in S} w_i w_i^T$. Further, define $\hat{\zeta} = \frac{1}{n}\sum_{i=1}^n w_i y_i$ and $\zeta = \mathbb{E}\left[wy\right]$. Then, we have

$$\hat{\beta}^{\mathsf{ols}} = \widehat{\mathbf{\Sigma}}^{-1}\mathbf{\Sigma}^{1/2}\hat{\zeta} \quad \text{and} \quad \beta^{\mathsf{ols}} = \mathbf{\Sigma}^{-1/2}\zeta.$$

For now, we work on the event that $\widehat{\mathbf{\Sigma}}$ is invertible. We will see that this event holds with very high probability. We write

$$\begin{aligned}
\left\|\mathbf{\Sigma}^{1/2}(\hat{\beta}^{\mathsf{ols}} - \beta^{\mathsf{ols}})\right\|_2 &= \left\|\mathbf{\Sigma}^{1/2}\widehat{\mathbf{\Sigma}}^{-1}\mathbf{\Sigma}^{1/2}\hat{\zeta} - \mathbf{\Sigma}^{-1/2}\zeta\right\|_2, \\
&= \left\|\widetilde{\mathbf{\Sigma}}^{-1}\left\{\hat{\zeta} - \zeta + \left(\mathbf{I} - \mathbf{\Sigma}^{-1/2}\widehat{\mathbf{\Sigma}}\mathbf{\Sigma}^{-1/2}\right)\zeta\right\}\right\|_2, \\
&\leq \left\|\widetilde{\mathbf{\Sigma}}^{-1}\right\|_2 \left\{\left\|\hat{\zeta} - \zeta\right\|_2 + \left\|\mathbf{I} - \widetilde{\mathbf{\Sigma}}\right\|_2 \|\zeta\|_2\right\},
\end{aligned} \tag{53}$$

where we used the triangle inequality and the properties of the operator norm.

For the first term on the right hand side of (53), we write

$$\left\|\widetilde{\mathbf{\Sigma}}^{-1}\right\|_2 = \frac{1}{\lambda_{\min}(\widetilde{\mathbf{\Sigma}})} \leq \frac{1}{1-\delta},$$

where we assumed that such a $\delta > 0$ exists. In fact, when $\delta < 0.5$, we obtain a bound of 2 on the right hand side, which also justifies the invertibility assumption of $\widehat{\mathbf{\Sigma}}$. By Corollary 5.50 of Vershynin (2010) and the following remark, we have with probability at least $1 - 2\exp\{-p\}$,

$$\left\|\widetilde{\mathbf{\Sigma}} - \mathbf{I}\right\|_2 \leq c\sqrt{\frac{p}{|S|}},$$

where c is a constant depending only on κ. When $|S| > 4c^2 p$, we obtain

$$\left|\lambda_{\min}(\widetilde{\mathbf{\Sigma}}) - 1\right| \leq \left\|\widetilde{\mathbf{\Sigma}} - \mathbf{I}\right\|_2 \leq 0.5,$$

where the first inequality follows from the Lipschitz property of the eigenvalues.

Next, we bound the difference between $\hat{\zeta}$ and its expectation ζ. We write the bounds on the sub-exponential norm

$$\begin{aligned}
\|wy\|_{\psi_1} &= \sup_{\|v\|_2=1} \sup_{m\geq 1} m^{-1}\mathbb{E}\left[|\langle v, w\rangle y|^m\right]^{1/m}, \\
&\leq \sup_{\|v\|_2=1} \sup_{m\geq 1} m^{-1}\mathbb{E}\left[|\langle v, w\rangle|^{2m}\right]^{1/2m} \mathbb{E}\left[|y|^{2m}\right]^{1/2m}, \\
&\leq \sup_{\|v\|_2=1} \sup_{m\geq 1} m^{-1/2}\mathbb{E}\left[|\langle v, w\rangle|^{2m}\right]^{1/2m} \sup_{m\geq 1} m^{-1/2}\mathbb{E}\left[|y|^{2m}\right]^{1/2m}, \\
&\leq 2\|w\|_{\psi_2}\|y\|_{\psi_2} = 2\gamma\kappa.
\end{aligned} \tag{54}$$

Hence, we have $\max_i \|w_i y_i - \mathbb{E}\left[w_i y_i\right]\|_{\psi_1} \leq 4\gamma\kappa$. Further, let e_j denote the j-th standard basis, and notice that each entry of w is also sub-Gaussian with norm upper bounded by κ, i.e.,

$$\kappa = \|w\|_{\psi_2} = \sup_{\|u\|_2=1} \|\langle u, w\rangle\|_{\psi_2} \geq \|\langle e_j, w\rangle\|_{\psi_2} = \|w_j\|_{\psi_2}.$$

Also, we can write

$$\begin{aligned} 2\gamma\kappa \geq \|wy\|_{\psi_1} &= \sup_{\|u\|_2=1} \sup_{m\geq 1} m^{-1}\mathbb{E}\left[|\langle u, w\rangle y|^m\right]^{1/m}, \\ &\geq \sup_{\|u\|_2=1} \mathbb{E}\left[|\langle u, w\rangle y|\right], \\ &\geq \sup_{\|u\|_2=1} \mathbb{E}\left[\langle u, w\rangle y\right] = \sup_{\|u\|_2=1} \langle u, \zeta\rangle = \|\zeta\|_2, \end{aligned} \tag{55}$$

where in the last step, we used the fact that dual norm of ℓ_2 norm is itself.

Next, we apply Lemma 17 to $\hat{\zeta} - \zeta$, and obtain with probability at least $1 - \exp\{-p\}$

$$\left\|\hat{\zeta} - \zeta\right\|_2 \leq c\gamma\kappa\sqrt{\frac{p}{n}},$$

whenever $n > c^2 p$ for an absolute constant c.

Combining the above results in (53), we obtain with probability at least $1 - 3\exp\{-p\}$

$$\left\|\mathbf{\Sigma}^{1/2}(\hat{\beta}^{\mathsf{ols}} - \beta^{\mathsf{ols}})\right\|_2 \leq 2\left\{c_1\gamma\kappa\sqrt{\frac{p}{n}} + c_2\gamma\kappa\sqrt{\frac{p}{|S|}}\right\} \leq \eta\sqrt{\frac{p}{|S|}} \tag{56}$$

where η depends only on κ and γ, and $|S| > \eta p$. Finally, we write

$$\begin{aligned} \left\|\hat{\beta}^{\mathsf{ols}} - \beta^{\mathsf{ols}}\right\|_2 &\leq \lambda_{\min}^{-1/2}\left\|\mathbf{\Sigma}^{1/2}(\hat{\beta}^{\mathsf{ols}} - \beta^{\mathsf{ols}})\right\|_2, \\ &\leq \eta\lambda_{\min}^{-1/2}\sqrt{\frac{p}{|S|}}, \end{aligned}$$

with probability at least $1 - 3\exp\{-p\}$, whenever $|S| > \eta p$. ■

The following lemma – combined with the Proposition 7 – provides the necessary tools to prove Theorem 9.

Lemma 15 *For a given function $\Psi^{(2)}$ that is Lipschitz continuous with constant k, we define the function $f : \mathbb{R} \times \mathbb{R}^p \to \mathbb{R}$ as $f(c, \beta) = c\ \mathbb{E}\left[\Psi^{(2)}(\langle x, \beta\rangle c)\right]$, and its empirical counterpart as*

$$\hat{f}(c, \beta) = c\ \frac{1}{n}\sum_{i=1}^{n} \Psi^{(2)}(\langle x_i, \beta\rangle c).$$

For the ball centered around β^{ols} with radius δ,

$$\mathcal{B}^{\delta}(\tilde{\beta}^{\mathsf{ols}}) = \left\{\beta : \left\|\beta - \tilde{\beta}^{\mathsf{ols}}\right\|_2 \leq \delta\right\}, \quad \textit{with} \quad \tilde{\beta}^{\mathsf{ols}} = \Sigma^{1/2}\beta^{\mathsf{ols}},$$

assume that $\sup_{\beta\in\mathcal{B}^{\delta}(\tilde{\beta}^{\mathsf{ols}})}\left\|\Psi^{(2)}(\langle x, \beta\rangle)\right\|_{\psi_2} \leq \kappa_g$, and $\left\|\Sigma^{-1/2}x_i\right\|_{\psi_2} \leq \kappa_x$. Further, for some $\epsilon, \bar{c} > 0$, assume that $f(\bar{c}, \beta^{\mathsf{ols}}) \geq 1+\epsilon$. Then, $\exists c_\Psi > 0$ satisfying the equation $1 = f(c_\Psi, \beta^{\mathsf{ols}})$.

Further, assume that for some $\tilde{\epsilon} > 0$, we have $\epsilon = \tilde{\epsilon}\sqrt{p}$, and n and $|S|$ sufficiently large, i.e.,

$$\min\left\{\frac{n}{\log(n)}, |S|\right\} > K^2/\tilde{\epsilon}^2$$

for $K = \eta\bar{c}\max\{b + \kappa/\tilde{\mu}, k\bar{c}\kappa\}$. Then, with probability $1-5\exp\{-p\}$, there exists a constant $\hat{c}_\Psi \in (0, \bar{c})$ satisfying the equation

$$1 = \hat{c}_\Psi\ \frac{1}{n}\sum_{i=1}^{n} \Psi^{(2)}(\langle x_i, \hat{\beta}^{\mathsf{ols}}\rangle\hat{c}_\Psi).$$

Moreover, if the derivative of $z \to f(z, \beta^{\mathsf{ols}})$ is bounded below in absolute value (i.e. does not change sign) by $\upsilon > 0$ in the interval $z \in [0, \bar{c}]$, then with probability $1 - 5\exp\{-p\}$, we have

$$|\hat{c}_\Psi - c_\Psi| \leq C\sqrt{\frac{p}{\min\{n/\log(n), |S|\}}},$$

where $C = K/\upsilon$.

Proof [of Lemma 15] First statement is obvious. We notice that $f(c, \beta^{\mathsf{ols}})$ is a continuous function in its first argument with $f(0, \beta^{\mathsf{ols}}) = 0$ and $f(\bar{c}, \beta^{\mathsf{ols}}) \geq 1 + \delta$. Hence, there exists $c_\Psi > 0$ such that $f(c_\Psi, \beta^{\mathsf{ols}}) = 1$. If there are many solutions to the above equation, we choose the one that is closest to zero. The condition on the derivative will guarantee the uniqueness of the solution.

Next, we will show the existence of $\hat{c}_\Psi$ using a uniform concentration given by Lemma 18. Define the event $\mathcal{E}$ that $\hat{\beta}^{\mathsf{ols}}$ falls into $\mathcal{B}^{\delta}_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})$, i.e., $\mathcal{E} = \left\{\hat{\beta}^{\mathsf{ols}} \in \mathcal{B}^{\delta}_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})\right\}$. By Proposition 7 and the inequality given in (56), whenever $|S| > \eta p$, we obtain

$$\mathbb{P}\left(\mathcal{E}^C\right) \leq 3\exp\{-p\},$$

where $\mathcal{E}^C$ denotes the complement of the event $\mathcal{E}$, and η is a constant depending only on κ_x and γ, and $\delta = \eta\sqrt{p/|S|}$. For any $c \in [0, \bar{c}]$, on the event $\mathcal{E}$, we have

$$\left|\hat{f}(c, \hat{\beta}^{\mathsf{ols}}) - f(c, \hat{\beta}^{\mathsf{ols}})\right| \leq \sup_{\beta\in\mathcal{B}^{\delta}_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})}\left|\hat{f}(c, \beta) - f(c, \beta)\right|.$$

Hence, we obtain the following inequality

$$\begin{aligned} &\mathbb{P}\left(\sup_{c\in[0,\bar{c}]}\left|\hat{f}(c, \hat{\beta}^{\mathsf{ols}}) - f(c, \hat{\beta}^{\mathsf{ols}})\right| > \epsilon\right) \\ &\leq \mathbb{P}\left(\sup_{c\in[0,\bar{c}]}\left|\hat{f}(c, \hat{\beta}^{\mathsf{ols}}) - f(c, \hat{\beta}^{\mathsf{ols}})\right| > \epsilon; \mathcal{E}\right) + \mathbb{P}\left(\mathcal{E}^C\right), \\ &\leq \mathbb{P}\left(\sup_{c\in[0,\bar{c}]}\sup_{\beta\in\mathcal{B}^{\delta}_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})}\left|\hat{f}(c, \beta) - f(c, \beta)\right| > \epsilon\right) + 3\exp\{-p\}. \end{aligned}$$

In the following, we will use Lemma 18 for the first term in the last line above. Denoting by w, the whitened covariates, we have $\langle x, \beta\rangle = \langle w, \mathbf{\Sigma}^{1/2}\beta\rangle$. Therefore,

$$\begin{aligned}&\sup_{c\in[0,\bar{c}]}\sup_{\beta\in\mathcal{B}^{\delta}_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})}\left|\hat{f}(c,\beta)-f(c,\beta)\right|\\&\leq \bar{c}\sup_{c\in[0,\bar{c}]}\sup_{\beta\in\mathcal{B}^{\delta}_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})}\left|\frac{1}{n}\sum_{i=1}^{n}\Psi^{(2)}(\langle w_i,\mathbf{\Sigma}^{1/2}\beta\rangle c)-\mathbb{E}\left[\Psi^{(2)}(\langle w,\mathbf{\Sigma}^{1/2}\beta\rangle c)\right]\right|.\end{aligned}$$

Next, define the ball centered around $\tilde{\beta}^{\mathsf{ols}} = \mathbf{\Sigma}^{1/2}\beta^{\mathsf{ols}}$, with radius δ as $\mathcal{B}_\delta(\tilde{\beta}^{\mathsf{ols}}) = \mathbf{\Sigma}^{1/2}\mathcal{B}^{\delta}_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})$. We have $\beta \in \mathcal{B}^{\delta}_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})$ if and only if $\mathbf{\Sigma}^{1/2}\beta \in \mathcal{B}_\delta(\tilde{\beta}^{\mathsf{ols}})$. Then, the right hand side of the above inequality can be written as

$$\begin{aligned}&\bar{c}\sup_{c\in[0,\bar{c}]}\sup_{\beta\in\mathcal{B}_{\delta}(\tilde{\beta}^{\mathsf{ols}})}\left|\frac{1}{n}\sum_{i=1}^{n}\Psi^{(2)}(\langle w_i,\beta\rangle c)-\mathbb{E}\left[\Psi^{(2)}(\langle w,\beta\rangle c)\right]\right|,\\&=\bar{c}\sup_{\beta\in\mathcal{B}_{\bar{c}\delta}(\tilde{\beta}^{\mathsf{ols}})}\left|\frac{1}{n}\sum_{i=1}^{n}\Psi^{(2)}(\langle w_i,\beta\rangle)-\mathbb{E}\left[\Psi^{(2)}(\langle w,\beta\rangle)\right]\right|.\end{aligned}$$

Then, by Lemma 18, we obtain

$$\mathbb{P}\left(\sup_{c\in[0,\bar{c}]}\left|\hat{f}(c,\hat{\beta}^{\mathsf{ols}})-f(c,\hat{\beta}^{\mathsf{ols}})\right|>c'\bar{c}(\kappa_g+\kappa_x/\tilde{\mu})\sqrt{\frac{p}{n/\log(n)}}\right)\leq 5\exp\{-p\}\tag{57}$$

whenever $np > 51\max\left\{\chi,\chi^{-1}\right\}$ where $\chi = (\kappa_g+\kappa_x/\tilde{\mu})^2/(c'\delta^2k^2\bar{c}^2\tilde{\mu}^2)$.

Also, by the Lipschitz condition for $\Psi^{(2)}$, we have for any $c\in[0,\bar{c}]$, and β_1,β_2,

$$\begin{aligned}|f(c,\beta_1)-f(c,\beta_2)|\leq& kc^2\mathbb{E}\left[\left|\langle w,\mathbf{\Sigma}^{1/2}(\beta_1-\beta_2)\rangle\right|\right]\\\leq& k\bar{c}^2\kappa_x\left\|\mathbf{\Sigma}^{1/2}(\beta_1-\beta_2)\right\|_2.\end{aligned}$$

Applying the above bound for $\beta_1=\hat{\beta}^{\mathsf{ols}}$ and $\beta_2=\beta^{\mathsf{ols}}$, we obtain with probability $1-3\exp\{-p\}$

$$\left|f(c,\hat{\beta}^{\mathsf{ols}})-f(c,\beta^{\mathsf{ols}})\right|\leq\eta k\bar{c}^2\kappa_x\sqrt{\frac{p}{|S|}},\tag{58}$$

where the last step follows from Proposition 7 and the inequality given in (56).

Combining this with the previous bound, and taking into account that $\mu=\tilde{\mu}\sqrt{p}$, for any $c\in[0,\bar{c}]$, with probability $1-5\exp\{-p\}$, we obtain

$$\begin{aligned}\left|\hat{f}(c,\hat{\beta}^{\mathsf{ols}})-f(c,\beta^{\mathsf{ols}})\right|\leq& c'\bar{c}(\kappa_g+\kappa_x/\tilde{\mu})\sqrt{\frac{p}{n/\log(n)}}+\eta k\bar{c}^2\kappa_x\sqrt{\frac{p}{|S|}}\\\leq& K\sqrt{\frac{p}{\min\{n/\log(n),|S|\}}}\end{aligned}$$

where $K=\eta\bar{c}\max\{\kappa_g+\kappa_x/\tilde{\mu},k\bar{c}\kappa_x\}$. Here, η depends only on κ_x and γ.

In particular, for $c=\bar{c}$ we observe that

$$\begin{aligned}\hat{f}(\bar{c},\hat{\beta}^{\mathsf{ols}})\geq& f(\bar{c},\beta^{\mathsf{ols}})-K\sqrt{\frac{p}{\min\{n/\log(n),|S|\}}}\\\geq& 1+\epsilon-K\sqrt{\frac{p}{\min\{n/\log(n),|S|\}}}.\end{aligned}$$

Therefore, for sufficiently large n and $|S|$ satisfying $\min\left\{\frac{n}{\log(n)},|S|\right\}>K^2/\bar{\epsilon}^2$ we obtain $\hat{f}(\bar{c},\hat{\beta}^{\mathsf{ols}})>1$. Since this function is continuous and $\hat{f}(0,\hat{\beta}^{\mathsf{ols}})=0$, we obtain the existence of $\hat{c}_\Psi\in[0,\bar{c}]$ with probability at least $1-5\exp\{-p\}$.

Now, since $\hat{c}_\Psi$ and c_Ψ satisfy the equations $\hat{f}(\hat{c}_\Psi,\hat{\beta}^{\mathsf{ols}})=f(c_\Psi,\beta^{\mathsf{ols}})=1$ (with high probability), by the inequality given in (57), with probability at least $1-5\exp\{-p\}$, we obtain

$$\left|1-f(\hat{c}_\Psi,\hat{\beta}^{\mathsf{ols}})\right|=\left|\hat{f}(\hat{c}_\Psi,\hat{\beta}^{\mathsf{ols}})-f(\hat{c}_\Psi,\hat{\beta}^{\mathsf{ols}})\right|\leq c'\bar{c}(\kappa_g+\kappa_x/\tilde{\mu})\sqrt{\frac{p}{n/\log(n)}}.$$

Also, by the same argument in (60), and Proposition 7, we get

$$\left|f(\hat{c}_\Psi,\hat{\beta}^{\mathsf{ols}})-f(\hat{c}_\Psi,\beta^{\mathsf{ols}})\right|\leq k\bar{c}^2\kappa\left\|\mathbf{\Sigma}(\hat{\beta}^{\mathsf{ols}}-\beta^{\mathsf{ols}})\right\|_2\leq\eta k\bar{c}^2\kappa_x\sqrt{\frac{p}{|S|}}.$$

Now, using the Taylor's series expansion of $c\to f(c,\beta^{\mathsf{ols}})$ around c_Ψ, and the assumption on the derivative of f with respect to its first argument, we obtain

$$\begin{aligned}\upsilon\left|\hat{c}_\Psi-c_\Psi\right|\leq&\left|f(\hat{c}_\Psi,\beta^{\mathsf{ols}})-f(c_\Psi,\beta^{\mathsf{ols}})\right|\\\leq&\left|f(\hat{c}_\Psi,\beta^{\mathsf{ols}})-f(\hat{c}_\Psi,\hat{\beta}^{\mathsf{ols}})\right|+\left|f(\hat{c}_\Psi,\hat{\beta}^{\mathsf{ols}})-1\right|\\\leq&\eta k\bar{c}^2\kappa_x\sqrt{\frac{p}{|S|}}+c'\bar{c}(\kappa_g+\kappa_x/\tilde{\mu})\sqrt{\frac{p}{n/\log(n)}}\\\leq&K\sqrt{\frac{p}{\min\{n/\log(n),|S|\}}}\end{aligned}$$

with probability at least $1-5\exp\{-p\}$. Here, the constant K is the same as before $K=\eta\bar{c}\max\{\kappa_g+\kappa_x/\tilde{\mu},k\bar{c}\kappa_x\}$.

■

Lemma 16 *For a given function $\Psi^{(2)}$ that is Lipschitz continuous with constant k, and uniformly bounded by b, we define the functions f, $\hat{f}$, and the constants $\epsilon, c_\Psi, \bar{c}$ as in Lemma 15. Assume that x_i's are i.i.d. sub-Gaussian with $\|x_i\|_{\psi_2}\leq\kappa_x$. Then, with probability $1-4\exp\{-p\}$, there exists a constant $\hat{c}_\Psi\in(0,\bar{c})$ satisfying the equation*

$$1=\hat{c}_\Psi\ \frac{1}{n}\sum_{i=1}^{n}\Psi^{(2)}(\langle x_i,\hat{\beta}^{\mathsf{ols}}\rangle\hat{c}_\Psi).$$

Moreover, if the derivative of $z \to f(z, \beta^{\mathsf{ols}})$ is bounded below in absolute value (i.e. does not change sign) by $\upsilon > 0$ in the interval $z \in [0, \bar{c}]$, then with probability $1 - 4\exp\{-p\}$, we have

$$|\hat{c}_\Psi - c_\Psi| \leq K\sqrt{\frac{p}{|S|}},$$

where $K = \eta\upsilon^{-1}\bar{c}k \max\left\{(1 + \|\tilde{\beta}\|_2)\|\Sigma\|_2 + b/k, \bar{c}\right\}$, η depends only on γ and κ_x, and $|S|$ is sufficiently large.

Proof [of Lemma 16] The proof follows from the same steps in Lemma 15. First statement is obvious.

Next, we will show the existence of $\hat{c}_\Psi$ using a uniform concentration given by Lemma 19. Let the ellipsoids $\mathcal{B}^\delta_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})$ and $\mathcal{B}^\delta(\tilde{\beta}^{\mathsf{ols}})$, and the event $\mathcal{E}$ be as in Lemma 15. By Proposition 7 and the inequality given in (56), whenever $|S| > \eta p$, we obtain

$$\mathbb{P}\left(\mathcal{E}^C\right) \leq 3\exp\{-p\},$$

where $\mathcal{E}^C$ denotes the complement of the event $\mathcal{E}$, and η is a constant depending only on κ_x and γ. For any $c \in [0, \bar{c}]$, on the event $\mathcal{E}$, we have

$$\left|\hat{f}(c, \hat{\beta}^{\mathsf{ols}}) - f(c, \hat{\beta}^{\mathsf{ols}})\right| \leq \sup_{\beta \in \mathcal{B}^\delta_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})} \left|\hat{f}(c, \beta) - f(c, \beta)\right|.$$

Using the same arguments as in the previous lemma that led to (57), we obtain the following inequality

$$\begin{aligned}
&\mathbb{P}\left(\sup_{c\in[0,\bar{c}]} \left|\hat{f}(c, \hat{\beta}^{\mathsf{ols}}) - f(c, \hat{\beta}^{\mathsf{ols}})\right| > \epsilon\right) \\
&\leq \mathbb{P}\left(\sup_{c\in[0,\bar{c}]} \sup_{\beta \in \mathcal{B}^\delta_{\mathbf{\Sigma}}(\beta^{\mathsf{ols}})} \left|\hat{f}(c, \beta) - f(c, \beta)\right| > \epsilon\right) + 3\exp\{-p\}, \\
&\leq \mathbb{P}\left(\sup_{\beta \in \mathcal{B}_{\bar{c}\delta}(\tilde{\beta}^{\mathsf{ols}})} \left|\frac{1}{n}\sum_{i=1}^n \Psi^{(2)}(\langle w_i, \beta\rangle) - \mathbb{E}\left[\Psi^{(2)}(\langle w, \beta\rangle)\right]\right| > \epsilon/\bar{c}\right) + 3\exp\{-p\}
\end{aligned}$$

where $\mathcal{B}_{\bar{c}\delta}(\tilde{\beta}^{\mathsf{ols}})$ is the ball centered around $\tilde{\beta}^{\mathsf{ols}} = \mathbf{\Sigma}^{1/2}\beta^{\mathsf{ols}}$ with radius $\delta = \eta\sqrt{p/|S|}$.

Then, using Lemma 19, we obtain

$$\mathbb{P}\left(\sup_{c\in[0,\bar{c}]} \left|\hat{f}(c, \hat{\beta}^{\mathsf{ols}}) - f(c, \hat{\beta}^{\mathsf{ols}})\right| > 2\bar{c}\left(k\left\{\|\tilde{\beta}\|_2 + \eta\sqrt{\frac{p}{|S|}}\right\}\|\Sigma\|_2 + b\right)\sqrt{\frac{p}{n}}\right) \leq 4\exp\{-p\}. \tag{59}$$

By the Lipschitz property of $\Psi^{(2)}$ as well as sub-Gaussian property of x, we have for any $c \in [0, \bar{c}]$, and β_1, β_2,

$$\begin{aligned}
|f(c, \beta_1) - f(c, \beta_2)| \leq& kc^2\mathbb{E}\left[\left|\langle w, \mathbf{\Sigma}^{1/2}(\beta_1 - \beta_2)\rangle\right|\right] \\
\leq& k\bar{c}^2\kappa_x\left\|\mathbf{\Sigma}^{1/2}(\beta_1 - \beta_2)\right\|_2.
\end{aligned}$$

Applying the above bound for $\beta_1 = \hat{\beta}^{\mathsf{ols}}$ and $\beta_2 = \beta^{\mathsf{ols}}$, we obtain with probability $1 - 3\exp\{-p\}$

$$\left|f(c, \hat{\beta}^{\mathsf{ols}}) - f(c, \beta^{\mathsf{ols}})\right| \leq \eta k\bar{c}^2\kappa_x\sqrt{\frac{p}{|S|}}, \tag{60}$$

where the last step follows from Proposition 7 and the inequality given in (56).

Combining this with the previous bound, for any $c \in [0, \bar{c}]$, with probability $1 - 4\exp\{-p\}$, we obtain

$$\begin{aligned}
\left|\hat{f}(c, \hat{\beta}^{\mathsf{ols}}) - f(c, \beta^{\mathsf{ols}})\right| \leq& 2\bar{c}\left(k\|\tilde{\beta}\|_2\|\Sigma\|_2 + \eta k\|\Sigma\|_2\sqrt{\frac{p}{|S|}} + b\right)\sqrt{\frac{p}{n}} + \eta k\bar{c}^2\kappa_x\sqrt{\frac{p}{|S|}} \\
=& 2\bar{c}k\left\{\|\tilde{\beta}\|_2\|\Sigma\|_2\sqrt{\frac{p}{n}} + \eta\|\Sigma\|_2\frac{p}{\sqrt{|S|n}} + b\sqrt{\frac{p}{n}} + \eta\bar{c}\kappa_x\sqrt{\frac{p}{|S|}}\right\} \\
\leq& \eta\bar{c}k\max\left\{(1 + \|\tilde{\beta}\|_2)\|\Sigma\|_2 + b/k, \bar{c}\right\}\sqrt{\frac{p}{|S|}} = K\sqrt{\frac{p}{|S|}}
\end{aligned}$$

where $K = \eta\bar{c}k\max\left\{(1 + \|\tilde{\beta}\|_2)\|\Sigma\|_2 + b/k, \bar{c}\right\}$, and $|S| > \eta p$. Here, η depends only on the constants κ_x and γ. In particular, for $c = \bar{c}$ we observe that

$$\begin{aligned}
\hat{f}(\bar{c}, \hat{\beta}^{\mathsf{ols}}) \geq& f(\bar{c}, \beta^{\mathsf{ols}}) - K\sqrt{\frac{p}{|S|}} \\
\geq& 1 + \epsilon - K\sqrt{\frac{p}{|S|}}.
\end{aligned}$$

Therefore, for sufficiently large $|S|$ satisfying $|S| > K^2 p$, we obtain $\hat{f}(\bar{c}, \hat{\beta}^{\mathsf{ols}}) > 1$. Since this function is continuous and $\hat{f}(0, \hat{\beta}^{\mathsf{ols}}) = 0$, we obtain the existence of $\hat{c}_\Psi \in [0, \bar{c}]$ with probability at least $1 - 4\exp\{-p\}$.

Now, since $\hat{c}_\Psi$ and c_Ψ satisfy the equations $\hat{f}(\hat{c}_\Psi, \hat{\beta}^{\mathsf{ols}}) = f(c_\Psi, \beta^{\mathsf{ols}}) = 1$ (with high probability), by the inequality given in (59), with probability at least $1 - 4\exp\{-p\}$, we obtain

$$\left|1 - f(\hat{c}_\Psi, \hat{\beta}^{\mathsf{ols}})\right| = \left|\hat{f}(\hat{c}_\Psi, \hat{\beta}^{\mathsf{ols}}) - f(\hat{c}_\Psi, \hat{\beta}^{\mathsf{ols}})\right| \leq 2\bar{c}\left(k\left\{\|\tilde{\beta}\|_2 + \eta\sqrt{\frac{p}{|S|}}\right\}\|\Sigma\|_2 + b\right)\sqrt{\frac{p}{n}}.$$

Now, using the Taylor's series expansion of $c \to f(c, \beta^{\mathsf{ols}})$ around c_Ψ, and the assumption on the derivative of f with respect to its first argument, we obtain

$$\begin{aligned}
\upsilon\,|\hat{c}_\Psi - c_\Psi| \leq& \left|f(\hat{c}_\Psi, \beta^{\mathsf{ols}}) - f(c_\Psi, \beta^{\mathsf{ols}})\right| \\
\leq& \left|f(\hat{c}_\Psi, \beta^{\mathsf{ols}}) - f(\hat{c}_\Psi, \hat{\beta}^{\mathsf{ols}})\right| + \left|f(\hat{c}_\Psi, \hat{\beta}^{\mathsf{ols}}) - 1\right| \\
\leq& \eta k\bar{c}^2\kappa_x\sqrt{\frac{p}{|S|}} + 2\bar{c}\left(k\left\{\|\tilde{\beta}\|_2 + \eta\sqrt{\frac{p}{|S|}}\right\}\|\Sigma\|_2 + b\right)\sqrt{\frac{p}{n}} \\
\leq& K\sqrt{\frac{p}{|S|}}
\end{aligned}$$

with probability at least $1-4\exp\{-p\}$. Here, the constant K is the same as before $K = \eta\bar{c}k\max\left\{(1+\|\tilde{\beta}\|_2)\,\|\Sigma\|_2 + b/k, \bar{c}\right\}$. ■

Proof [of Theorem 8] We have

$$\left\|\hat{\beta}^{\mathsf{sls}} - \beta^{\mathsf{pop}}\right\|_\infty \leq \left\|c_\Psi\beta^{\mathsf{ols}} - \beta^{\mathsf{pop}}\right\|_\infty + \left\|\hat{c}_\Psi\hat{\beta}^{\mathsf{ols}} - c_\Psi\beta^{\mathsf{ols}}\right\|_\infty, \tag{61}$$

where we used the triangle inequality for the ℓ_∞ norm. The first term on the right hand side can be bounded using Theorem 3 and Corollary 5. We write

$$\left\|c_\Psi\beta^{\mathsf{ols}} - \beta^{\mathsf{pop}}\right\|_\infty \leq \eta_1\,\frac{\|\beta^{\mathsf{pop}}\|_\infty}{\sqrt{p}}, \tag{62}$$

for $\eta_1 = 8k\bar{c}\kappa^3\rho_\infty\|\mathbf{\Sigma}^{1/2}\|_\infty(\tau/r)$.

For the second term, we write

$$\begin{aligned}\left\|\hat{c}_\Psi\hat{\beta}^{\mathsf{ols}} - c_\Psi\beta^{\mathsf{ols}}\right\|_\infty &= \left\|\hat{c}_\Psi\hat{\beta}^{\mathsf{ols}} \pm \hat{c}_\Psi\beta^{\mathsf{ols}} - c_\Psi\beta^{\mathsf{ols}}\right\|_\infty, \\ &\leq |\hat{c}_\Psi|\left\|\hat{\beta}^{\mathsf{ols}} - \beta^{\mathsf{ols}}\right\|_\infty + |\hat{c}_\Psi - c_\Psi|\left\|\beta^{\mathsf{ols}}\right\|_\infty,\end{aligned} \tag{63}$$

where the first step follows from triangle inequality. By Lemma 16, for sufficiently large n and $|S|$, with probability $1-4\exp\{-p\}$, the constant $\hat{c}_\Psi$ exists and it is in the interval $(0,\bar{c}]$. By the same lemma, with probability $1-4\exp\{-p\}$, we have

$$|\hat{c}_\Psi - c_\Psi| \leq \eta_4\sqrt{\frac{p}{|S|}}, \tag{64}$$

where $\eta_4 = \eta' v^{-1}\bar{c}k\max\left\{(\|\Sigma^{1/2}\beta^{\mathsf{ols}}\|_2+1)\|\Sigma\|_2 + b/k, \bar{c}\right\}$ for some constant η' depending on the sub-Gaussian norms κ and γ.

Also, by the norm equivalence and Proposition 7, we have with probability $1-4\exp\{-p\}$

$$\left\|\hat{\beta}^{\mathsf{ols}} - \beta^{\mathsf{ols}}\right\|_\infty \leq \eta_3\sqrt{\frac{p}{|S|}}, \tag{65}$$

for $\eta_3 = \eta''\lambda_{\min}^{-1/2}$, where η'' is constant depending only on γ and κ.

Finally, combining all these inequalities with the last line of (61), we have with probability $1-4\exp\{-p\}$,

$$\begin{aligned}\left\|\hat{\beta}^{\mathsf{sls}} - \beta^{\mathsf{pop}}\right\|_\infty &\leq \eta_1\frac{\|\beta^{\mathsf{pop}}\|_\infty}{\sqrt{p}} + \eta_3\bar{c}\,\sqrt{\frac{p}{|S|}} + \eta_4\left\|\beta^{\mathsf{ols}}\right\|_\infty\sqrt{\frac{p}{|S|}}, \\ &\leq \eta_1\frac{\|\beta^{\mathsf{pop}}\|_\infty}{\sqrt{p}} + \left(\eta_3\bar{c} + \eta_4\left\|\beta^{\mathsf{ols}}\right\|_\infty\right)\sqrt{\frac{p}{|S|}}, \\ &= \eta_1\frac{\|\beta^{\mathsf{pop}}\|_\infty}{\sqrt{p}} + \eta_2\sqrt{\frac{p}{|S|}},\end{aligned} \tag{66}$$

where

$$\begin{aligned}\eta_1 &= \eta k\bar{c}\rho_\infty(\tau/r)\|\mathbf{\Sigma}^{1/2}\|_\infty \\ \eta_2 &= \eta_3\bar{c} + \eta_4\left\|\beta^{\mathsf{ols}}\right\|_\infty = \eta\bar{c}\left(\lambda_{\min}^{-1/2} + kv^{-1}\max\left\{(\|\Sigma^{1/2}\beta^{\mathsf{ols}}\|_2+1)\|\Sigma\|_2 + b/k, \bar{c}\right\}\right)\end{aligned} \tag{67}$$

■

Proof [of Theorem 9] Proof is essentially the same as that of Theorem 8, where this time we use Lemma 16 instead of Lemma 15. ■

Proof [of Corollary 6] The normal equations for the lasso minimization yields

$$\mathbb{E}\left[xx^T\right]\beta_\lambda^{\mathsf{lasso}} - \beta^{\mathsf{ols}} + \lambda s = 0,$$

where $s \in \partial\left\|\beta_\lambda^{\mathsf{lasso}}\right\|_1$. It is well-known that under the orthogonal design where the covariates have i.i.d. entries, the above equation reduces to $\mathsf{soft}(\beta^{\mathsf{ols}};\lambda) = \beta_\lambda^{\mathsf{lasso}}$, where $\mathsf{soft}(\,\cdot\,;\lambda)$ denotes the soft thresholding operator at level λ. For any $\beta\in\mathbb{R}^p$, let $\mathrm{supp}(\beta)$ denote the support of β, i.e., the set $\{i\in[p]:\beta_i\neq 0\}$. We have

$$\mathrm{supp}(\beta_\lambda^{\mathsf{lasso}}) = \{i\in[p]:\beta_{\lambda,i}^{\mathsf{lasso}}\neq 0\} = \{i\in[p]:|\beta_i^{\mathsf{ols}}|>\lambda\}$$

By Theorem 3, we have $|\beta_i^{\mathsf{ols}}| \leq \frac{1}{c_\Psi}|\beta_i^{\mathsf{pop}}| + \delta$, which implies that

$$\mathrm{supp}(\beta_\lambda^{\mathsf{lasso}}) \subset \left\{i\in[p]:\frac{1}{c_\Psi}|\beta_i^{\mathsf{pop}}| + \delta \geq \lambda\right\}.$$

Hence, whenever $\lambda\geq\delta$, we have $\mathrm{supp}(\beta_\lambda^{\mathsf{lasso}})\subset\mathrm{supp}(\beta^{\mathsf{pop}})$. Further, we have by Theorem 3

$$\frac{1}{c_\Psi}|\beta_i^{\mathsf{pop}}| \leq |\beta_i^{\mathsf{ols}}| + \delta.$$

Hence, whenever $|\beta_i^{\mathsf{pop}}| > c_\Psi\,(\lambda+\delta)$, we get $|\beta_i^{\mathsf{ols}}|>\lambda$. If this condition is satisfied for any entry in the support of β^{pop}, the corresponding lasso coefficient will be non-zero. Therefore, we get $\mathrm{supp}(\beta^{\mathsf{pop}})\subset\mathrm{supp}(\beta_\lambda^{\mathsf{lasso}})$ under this assumption. Combining this with the previous result, we conclude the proof. ■

11. Auxiliary Lemmas

Lemma 17 (Sub-exponential vector concentration) *Let $x_1, x_2, ..., x_n$ be independent centered sub-exponential random vectors with $\max_i\|x_i\|_{\psi_1} = \kappa$. Then we have*

$$\mathbb{P}\left(\left\|\frac{1}{n}\sum_{i=1}^n x_i\right\|_2 > c\kappa\sqrt{\frac{p}{n}}\right) \leq \exp\{-p\}. \tag{68}$$

whenever $n > 4c^2p$ for an absolute constant c.

Proof [of Lemma 17] For a vector $z \in \mathbb{R}^p$, we have $\|z\|_2 = \sup_{\|u\|_2=1} \langle u, z\rangle$ since the dual of ℓ_2 norm is itself. Therefore, we write

$$\mathbb{P}\left(\left\|\frac{1}{n}\sum_{i=1}^n x_i\right\|_2 > t\right) = \mathbb{P}\left(\sup_{\|u\|_2=1} \frac{1}{n}\sum_{i=1}^n \langle u, x_i\rangle > t\right).$$

Now, let $\mathcal{N}_\epsilon$ be an ϵ-net over $\mathcal{S}^{p-1} = \{u \in \mathbb{R}^p : \|u\|_2 = 1\}$, and observe that

$$\max_{u\in\mathcal{N}_\epsilon} \langle u, x\rangle \geq (1-\epsilon) \sup_{\|u\|_2=1} \langle u, x\rangle \quad = (1-\epsilon)\|x\|_2,$$

with $|\mathcal{N}_\epsilon| \leq (1+2/\epsilon)^p$. Hence, we may write

$$\begin{aligned}\mathbb{P}\left(\sup_{\|u\|_2=1} \frac{1}{n}\sum_{i=1}^n \langle u, x_i\rangle > t\right) \leq& \mathbb{P}\left(\max_{u\in\mathcal{N}_\epsilon} \frac{1}{n}\sum_{i=1}^n \langle u, x_i\rangle > t(1-\epsilon)\right),\\ \leq& |\mathcal{N}_\epsilon|\, \mathbb{P}\left(\frac{1}{n}\sum_{i=1}^n \langle u, x_i\rangle > t(1-\epsilon)\right).\end{aligned}$$

For any $u \in \mathcal{S}^{p-1}$, we have $\|\langle u, x_i\rangle\|_{\psi_1} \leq \kappa$. Then, by the Bernstein-type inequality for sub-exponential random variables (Vershynin, 2010), we have

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^n \langle u, x_i\rangle > t(1-\epsilon)\right) \leq \exp\left\{-cn\min\left\{\frac{t^2(1-\epsilon)^2}{\kappa^2}, \frac{t(1-\epsilon)}{\kappa}\right\}\right\},$$

for an absolute constant c. Therefore, the probability on the left hand side of (68) can be bounded by

$$\left(1+\frac{2}{\epsilon}\right)^p \exp\left\{-cn\frac{t^2(1-\epsilon)^2}{\kappa^2}\right\} = \exp\left\{-cn\frac{t^2(1-\epsilon)^2}{\kappa^2} + p\log\left(1+\frac{2}{\epsilon}\right)\right\},$$

whenever $t < \kappa/(1-\epsilon)$. Choosing $\epsilon = 0.5$ and for an absolute constant $c' > 3.24/c$ and letting $t = c'\kappa\sqrt{\frac{p}{n}}$, we conclude the proof. ■

Lemma 18 *Let $B(\tilde\beta)$ denote the ball centered around $\tilde\beta$ with radius δ, i.e.,*

$$B(\tilde\beta) = \left\{\beta : \left\|\beta - \tilde\beta\right\|_2 \leq \delta\right\}.$$

For $i = 1, ..., n$, let $x_i \in \mathbb{R}^p$ be i.i.d. sub-Gaussian random vectors with $\|x_i\|_{\psi_2} \leq \kappa_x$. Given a function $g : \mathbb{R} \to \mathbb{R}$ that is Lipschitz continuous with k, and satisfying $\sup_{\beta\in B(\tilde\beta)} \|g(\langle x_i, \beta\rangle)\|_{\psi_2} \leq \kappa_g$, we have

$$\mathbb{P}\left(\sup_{\beta\in B}\left|\frac{1}{n}\sum_{i=1}^n g(\langle x_i, \beta\rangle) - \mathbb{E}\left[g(\langle x, \beta\rangle)\right]\right| > c(\kappa_g + \kappa_x/\tilde\mu)\sqrt{\frac{p}{n/\log(n)}}\right) \leq 2\exp\{-p\},$$

whenever $np > 51\max\{\chi, \chi^{-1}\}$ for $\chi = (\kappa_g + \kappa_x/\tilde\mu)^2/(c\delta^2k^2\tilde\mu^2)$. Above, c is an absolute constant.

Proof [of Lemma 18] Let $\mathbb{E}[\|x\|_2] = \mu = \tilde\mu\sqrt{p}$ and for $\epsilon > 0$, $\beta \in B(\tilde\beta)$ and $w \in \mathbb{R}^p$ define the bounding functions

$$l_\beta(w) = g(\langle w, \beta\rangle) - \epsilon\|w\|_2/4\mu, \quad u_\beta(w) = g(\langle w, \beta\rangle) + \epsilon\|w\|_2/4\mu.$$

Let $\mathcal{N}_\Delta$ be a net over $B(\tilde\beta)$ in the sense that for any $\beta_1 \in B(\tilde\beta)$, $\exists\beta_2 \in \mathcal{N}_\Delta$ such that $\|\beta_1 - \beta_2\|_2 \leq \Delta$. We fix $\Delta_* = \epsilon/(4k\mu)$ and write $\forall\beta_1 \in B$, $\exists\beta_2 \in \mathcal{N}_{\Delta_*}$,

1. an upper bound of the form:

$$\begin{aligned}g(\langle w, \beta_1\rangle) \leq& g(\langle w, \beta_2\rangle) + k\left|\langle w, \beta_1 - \beta_2\rangle\right|,\\ \leq& g(\langle w, \beta_2\rangle) + k\left\|w\right\|_2\Delta_* = u_{\beta_2}(w),\end{aligned}$$

2. and a lower bound of the form:

$$\begin{aligned}g(\langle w, \beta_1\rangle) \geq& g(\langle w, \beta_2\rangle) - k\left|\langle w, \beta_1 - \beta_2\rangle\right|,\\ \geq& g(\langle w, \beta_2\rangle) - k\left\|w\right\|_2\Delta_* = l_{\beta_2}(w),\end{aligned}$$

where the second steps in the above inequalities follow from the Cauchy-Schwarz inequality. These functions are called *bracketing functions* in the context of empirical process theory.

Hence, we can write that $\forall\beta_1 \in B(\tilde\beta)$, $\exists\beta_2 \in \mathcal{N}_{\Delta_*}$ such that

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^n l_{\beta_2}(x_i) - \mathbb{E}\left[l_{\beta_2}(x)\right] - \epsilon/2 \leq& \frac{1}{n}\sum_{i=1}^n g(\langle x_i, \beta_1\rangle) - \mathbb{E}\left[g(\langle x, \beta_1\rangle)\right],\\ \leq& \frac{1}{n}\sum_{i=1}^n u_{\beta_2}(x_i) - \mathbb{E}\left[u_{\beta_2}(x)\right] + \epsilon/2.\end{aligned}$$

The above inequalities translate to the following conclusion: Whenever the following event happens,

$$\left\{\left|\frac{1}{n}\sum_{i=1}^n g(\langle x_i, \beta_1\rangle) - \mathbb{E}\left[g(\langle x, \beta_1\rangle)\right]\right| > \epsilon\right\},$$

at least one of the following events happens

$$\left\{\frac{1}{n}\sum_{i=1}^n u_{\beta_2}(x_i) - \mathbb{E}\left[u_{\beta_2}(x)\right] > \epsilon/2\right\} \quad \text{or} \quad \left\{\frac{1}{n}\sum_{i=1}^n l_{\beta_2}(x_i) - \mathbb{E}\left[l_{\beta_2}(x)\right] < -\epsilon/2\right\}.$$

Therefore, using the union bound on the above events, we may obtain

$$\begin{aligned}&\mathbb{P}\left(\sup_{\beta\in B(\tilde\beta)}\left|\frac{1}{n}\sum_{i=1}^n g(\langle x_i, \beta\rangle) - \mathbb{E}\left[g(\langle x, \beta\rangle)\right]\right| > \epsilon\right) \qquad (69)\\ &\leq \mathbb{P}\left(\max_{\beta\in\mathcal{N}_{\Delta_*}} \frac{1}{n}\sum_{i=1}^n u_\beta(x_i) - \mathbb{E}\left[u_\beta(x)\right] > \epsilon/2\right)\\ &\quad + \mathbb{P}\left(\max_{\beta\in\mathcal{N}_{\Delta_*}} \frac{1}{n}\sum_{i=1}^n l_\beta(x_i) - \mathbb{E}\left[l_\beta(x)\right] < -\epsilon/2\right).\end{aligned}$$

Note that the right hand side of the above inequality has two terms both of which are of the same form. For simplicity, we bound only the first one. The bound for the second one follows from the exact same steps.

The relation between sub-Gaussian and sub-exponential norms (Vershynin, 2010) allows us to write

$$\|\|x\|_2\|_{\psi_2}^2 \leq \|\|x\|_2^2\|_{\psi_1} \leq \sum_{i=1}^{p} \|x_i^2\|_{\psi_1} \leq 2\sum_{i=1}^{p} \|x_i\|_{\psi_2}^2 \leq 2\kappa_x^2 p, \tag{70}$$

where the second step follows from the triangle inequality. Hence, we conclude that $\|x\|_2 - \mathbb{E}[\|x\|_2]$ is a centered sub-Gaussian random variable with norm upper bounded by $3\kappa_x\sqrt{p}$.

For $\epsilon < 4/3$, we notice that the random variable $u_\beta(x) = g(\langle x, \beta\rangle) + \epsilon\|x\|_2/4\mu$ is also sub-Gaussian with norm

$$\|u_\beta(x)\|_{\psi_2} \leq \kappa_g + \frac{\epsilon}{4\tilde{\mu}} 3\kappa_x \leq \kappa_g + \kappa_x/\tilde{\mu},$$

and consequently, the centered random variable $u_\beta(x) - \mathbb{E}[u_\beta(x)]$ has the sub-Gaussian norm upper bounded by $2\kappa_g + 2\kappa_x/\tilde{\mu}$.

Then, by the Hoeffding-type inequality for the sub-Gaussian random variables, we obtain

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n} u_\beta(x_i) - \mathbb{E}[u_\beta(x)] > \epsilon/2\right) \leq \exp\left\{-cn\frac{\epsilon^2}{(\kappa_g + \kappa_x/\tilde{\mu})^2}\right\}$$

for an absolute constant $c > 0$.

By the same argument above, one can obtain the same result for the function $l_\beta(x)$. Using Hoeffding bounds in (69) along with the union bound over the net, we immediately obtain

$$\mathbb{P}\left(\sup_{\beta\in B(\tilde{\beta})} \left|\frac{1}{n}\sum_{i=1}^{n} g(\langle x_i, \beta\rangle) - \mathbb{E}[g(\langle x, \beta\rangle)]\right| > \epsilon\right) \leq 2|\mathcal{N}_{\Delta_*}| \exp\left\{-cn\frac{\epsilon^2}{(\kappa_g + \kappa_x/\tilde{\mu})^2}\right\}$$

for some absolute constant c.

Using a standard covering argument over the net $\mathcal{N}_{\Delta_*}$ as given in Erdogdu (2015), we have

$$|\mathcal{N}_{\Delta_*}| \leq \left(\frac{\delta\sqrt{p}}{\Delta_*}\right)^p = \left(\frac{4\delta k\tilde{\mu}p}{\epsilon}\right)^p.$$

Combining this with the previous bound, and choosing

$$\epsilon^2 = \frac{p}{n}\frac{(\kappa_g + \kappa_x/\tilde{\mu})^2}{2c}\log\left(\frac{32c\delta^2k^2\tilde{\mu}^2pn}{(\kappa_g + \kappa_x/\tilde{\mu})^2}\right)$$

we get

$$\begin{aligned}
&2\left(\frac{4\delta k\tilde{\mu}p}{\epsilon}\right)^p \exp\left\{-cn\frac{\epsilon^2}{(\kappa_g + \kappa_x/\tilde{\mu})^2}\right\} \\
&= 2\exp\left\{-\frac{p}{2}\log\log\left(\frac{32c\delta^2k^2\tilde{\mu}^2pn}{(\kappa_g + \kappa_x/\tilde{\mu})^2}\right)\right\} \leq 2\exp\{-p\},
\end{aligned}$$

whenever $np > 51\max\{\chi, \chi^{-1}\}$ for $\chi = (\kappa_g + \kappa_x/\tilde{\mu})^2/(c\delta^2k^2\tilde{\mu}^2)$. ■

Lemma 19 *Let $B(\tilde{\beta})$ denote the ball centered around $\tilde{\beta}$ with radius δ, i.e.,*

$$B(\tilde{\beta}) = \left\{\beta : \left\|\beta - \tilde{\beta}\right\|_2 \leq \delta\right\}.$$

For $i = 1, ..., n$, let $x_i \in \mathbb{R}^p$ be i.i.d. random vectors with a covariance matrix Σ. Given a function $g : \mathbb{R} \to \mathbb{R}$ that is uniformly bounded by $b > 0$, and Lipschitz continuous with k,

$$\mathbb{P}\left(\sup_{\beta\in B(\tilde{\beta})} \left|\frac{1}{n}\sum_{i=1}^{n} \Psi^{(2)}(\langle x_i, \beta\rangle) - \mathbb{E}\left[\Psi^{(2)}(\langle x, \beta\rangle)\right]\right| \geq 2\left\{k(\|\tilde{\beta}\|_2 + \delta)\|\Sigma\|_2 + b\right\}\sqrt{\frac{p}{n}}\right) \leq e^{-p}. \tag{71}$$

Proof [of Lemma 19] This is a standard result in the theory of Rademacher complexity – see for example Bartlett and Mendelson (2002). Define the following empirical process

$$G(x_1, ..., x_n) = \sup_{\beta\in B(\tilde{\beta})} \left|\frac{1}{n}\sum_{i=1}^{n} g(\langle x_i, \beta\rangle) - \mathbb{E}[g(\langle x, \beta\rangle)]\right|. \tag{72}$$

We notice that for any i, we have

$$\left|G(.., x_i, ..) - G(.., x_i', ..)\right| \leq \frac{2b}{n}, \tag{73}$$

due to boundedness of the function g. By the McDiarmid's inequality (Azuma), we have

$$\mathbb{P}(G > \epsilon) \leq \exp\left\{-\frac{(\epsilon - \mathbb{E}[G])^2 n}{2b^2}\right\}. \tag{74}$$

The above inequality – when stated differently – reads that with probability at least $1 - \exp\{-p\}$, we have

$$G < \mathbb{E}[G] + b\sqrt{\frac{2p}{n}}. \tag{75}$$

For $\sigma_1, ..., \sigma_n$ i.i.d. random variables with $\mathbb{P}(\sigma_i = \pm 1) = 0.5$, we have

$$\mathbb{E}[G] \leq \frac{2}{n}\mathbb{E}\left[\sup_{\beta\in B(\tilde{\beta})} \sum_{i=1}^{n} \sigma_i g(\langle x_i, \beta\rangle)\right] = \mathcal{R}(g \circ \mathcal{B}(\tilde{\beta})), \tag{76}$$

where $\mathcal{R}$ denoting the Rademacher complexity. Since g is Lipschitz continuous with constant k, we have $\mathcal{R}(g \circ B(\tilde{\beta})) \leq k\mathcal{R}(B(\tilde{\beta}))$ by the Ledoux-Talagrand contraction inequality. We

write

$$\begin{aligned}\mathcal{R}(B(\tilde{\beta})) =& \frac{2}{n}\mathbb{E}\left[\sup_{\beta\in B(\tilde{\beta})}\sum_{i=1}^{n}\sigma_i\langle x_i,\beta\rangle\right] \leq \frac{2}{n}\mathbb{E}\left[\sup_{\beta\in B(\tilde{\beta})}\left\|\sum_{i=1}^{n}\sigma_i x_i\right\|_2\|\beta\|_2\right], \\ \leq& (\|\tilde{\beta}\|_2+\delta)\frac{2}{n}\mathbb{E}\left[\left\|\sum_{i=1}^{n}\sigma_i x_i\right\|_2^2\right]^{1/2} = (\|\tilde{\beta}\|_2+\delta)\frac{2}{n}\left(\sum_{i=1}^{n}\mathbb{E}\left[\|x_i\|_2^2\right]\right)^{1/2}, \\ \leq& 2(\|\tilde{\beta}\|_2+\delta)\|\Sigma\|_2\sqrt{\frac{p}{n}}.\end{aligned} \tag{77}$$

Combining this with the previous results, we obtain

$$\mathbb{P}\left(\sup_{\beta\in B(\tilde{\beta})}\left|\frac{1}{n}\sum_{i=1}^{n}\Psi^{(2)}(\langle x_i,\beta\rangle)-\mathbb{E}\left[\Psi^{(2)}(\langle x,\beta\rangle)\right]\right| \geq 2\left\{k(\|\tilde{\beta}\|_2+\delta)\|\Sigma\|_2+b\right\}\sqrt{\frac{p}{n}}\right) \leq e^{-p}.$$

■

12. Additional Experiments

In this section, we provide additional experiments. The overall setting is the same as Section 8. The only difference is that we change the sampling distribution of the datasets, which are stated in the title of each plot. As in Section 8, SLS estimator outperforms its competitors by a large margin in terms of the computation time.

The results are provided in Figures 7 and 6, and Table 3.

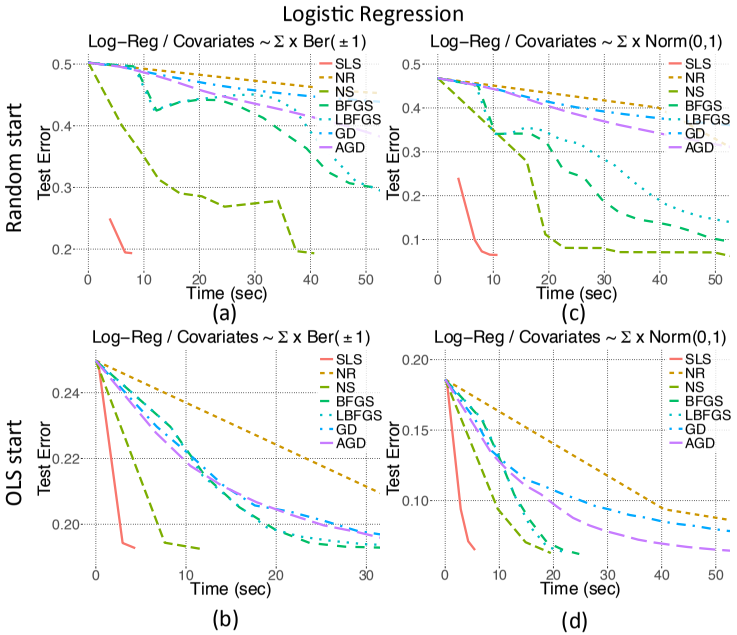


Figure 6: Additional experiments comparing the performance of SLS to that of MLE obtained with various optimization algorithms on several datasets. SLS is represented with red straight line. The details are provided in Table 3

Table 3: Details of the experiments shown in Figures 7 and 6.

Model	Logistic Regression				Poisson Regression			
Dataset	$\mathbf{\Sigma}\times$Ber($\pm$1)		$\mathbf{\Sigma}\times$Norm(0,1)		$\mathbf{\Sigma}\times${Exp(1)-1}		$\mathbf{\Sigma}\times$Norm(0,1)	
Size	$n=6.0\times10^5$, $p=300$		$n=6.0\times10^5$, $p=300$		$n=6.0\times10^5$, $p=300$		$n=6.0\times10^5$, $p=300$	
Init	Rnd	Ols	Rnd	Ols	Rnd	Ols	Rnd	Ols
Plot	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
Method↓	Time in Seconds / Number of Iterations (to reach min test error)							
Sls	6.61/3	2.97/3	9.38/5	4.25/4	14.68/4	2.99/4	6.66/10	4.13/10
Nr	222.21/6	84.08/3	186.33/6	115.76/4	218.1/6	218.9/4	364.63/9	363.4/9
Ns	40.68/10	11.57/3	53.06/9	19.52/4	39.22/6	59.61/4	51.48/10	39.8/10
Bfgs	125.83/33	35.41/9	155.3/48	24.78/8	46.61/20	48.71/12	92.84/36	74.22/38
LBfgs	142.09/38	44.41/12	444.62/143	21.79/7	96.53/39	50.56/12	296.4/111	228.1/117
Gd	409.9/134	79.45/22	1773.1/509	135.62/44	569.1/211	124.31/48	792.3/344	1041.1/366
Agd	177.3/159	43.76/12	359.56/95	53.73/18	157.9/57	63.16/16	74.74/32	62.21/32

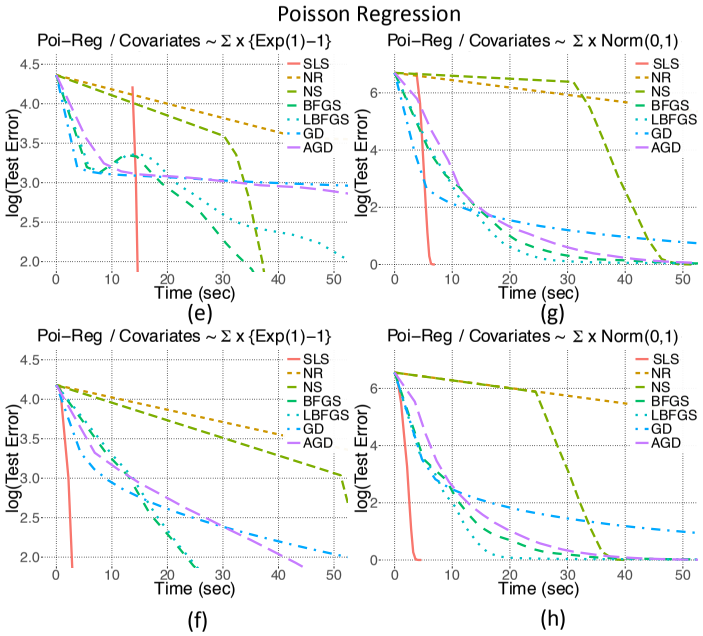


Figure 7: Additional experiments comparing the performance of SLS to that of MLE obtained with various optimization algorithms on several datasets. SLS is represented with red straight line. The details are provided in Table 3

References

Pierre Alquier and Gérard Biau. Sparse single-index model. *Journal of Machine Learning Research*, 14(Jan):243–280, 2013.

Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5, 2014.

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Mohsen Bayati and Andrea Montanari. The lasso risk for gaussian matrices. *IEEE Transactions on Information Theory*, 58(4):1997–2017, 2012.

Mohsen Bayati, Murat A Erdogdu, and Andrea Montanari. Estimating lasso risk and noise level. In *Advances in Neural Information Processing Systems*, pages 944–952, 2013.

Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

Jock A. Blackard and Denis J. Dean. Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. *Comput. Electron. Agr.*, 24:131–151, 1999.

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

David R Brillinger. A generalized linear model with "Gaussian" regressor variables. In *A Festschrift For Erich L. Lehmann*, pages 97–114. CRC Press, 1982.

Charles G Broyden. The convergence of a class of double-rank minimization algorithms 2. the new algorithm. *IMA Journal of Applied Mathematics*, 6(3):222–231, 1970.

Andreas Buja, Werner Stuetzle, and Yi Shen. Loss functions for binary class probability estimation and classification: Structure and applications. 2005.

Louis H. Y. Chen, Larry Goldstein, and Qi-Man Shao. *Normal approximation by Stein's method*. Springer, 2010.

Sheng Chen and Arindam Banerjee. Robust structured estimation with single-index models. In *International Conference on Machine Learning*, pages 712–721, 2017.

Paramveer Dhillon, Yichao Lu, Dean P Foster, and Lyle Ungar. New subsampling algorithms for fast least squares regression. In *Advances in Neural Information Processing Systems*, pages 360–368, 2013.

David L Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009.

Petros Drineas, Michael W Mahoney, S Muthukrishnan, and Tamás Sarlós. Faster least squares approximation. *Numerische Mathematik*, 117(2):219–249, 2011.

Naihua Duan and Ker-Chau Li. Slicing regression: a link-free regression method. *The Annals of Statistics*, pages 505–530, 1991.

Murat A Erdogdu. Newton-stein method: A second order method for glms via stein's lemma. In *Advances in Neural Information Processing Systems*, pages 1216–1224, 2015.

Murat A. Erdogdu. Newton-stein method: an optimization method for glms via stein's lemma. *The Journal of Machine Learning Research*, 17(1):7565–7616, 2016.

Murat A Erdogdu. *Stein's lemma and subsampling in large-scale optimization*. PhD thesis, Stanford University, 2017.

Murat A Erdogdu and Andrea Montanari. Convergence rates of sub-sampled newton methods. In *Advances in Neural Information Processing Systems*, pages 3052–3060, 2015.

Murat A Erdogdu, Mohsen Bayati, and Lee H Dicker. Scaled least squares estimator for glms in large-scale problems. In *Advances in Neural Information Processing Systems*, 2016.

Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals Eugenic*, 7:179–188, 1936.

Roger Fletcher. A new approach to variable metric algorithms. *The computer journal*, 13 (3):317–322, 1970.

Donald Goldfarb. A family of variable-metric methods derived by variational means. *Mathematics of computation*, 24(109):23–26, 1970.

Larry Goldstein. l^1 bounds in normal approximation. *Annals Probability*, 35:1888–1930, 2007.

Larry Goldstein and Gesine Reinert. Stein's method and the zero bias transformation with application to simple random sampling. *Annals of Applied Probability*, 7:935–952, 1997.

Jackson Gorham and Lester Mackey. Measuring sample quality with stein's method. In *Advances in Neural Information Processing Systems*, pages 226–234, 2015.

Peter Hall and Ker-Chau Li. On almost linearity of low dimensional projections from high dimensional data. *The annals of Statistics*, pages 867–889, 1993.

David P Helmbold, Jyrki Kivinen, and Manfred K Warmuth. Relative loss bounds for single neurons. *IEEE Transactions on Neural Networks*, 10(6):1291–1304, 1999.

Magnus R Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems'. *Journal of Research of the National Bureau of Standards*, 49(6), 1952.

Marian Hristache, Anatoli Juditsky, and Vladimir Spokoiny. Direct estimation of the index coefficient in a single-index model. *Annals of Statistics*, pages 595–623, 2001.

Sham M Kakade, Varun Kanade, Ohad Shamir, and Adam Kalai. Efficient learning of generalized linear and single index models with isotonic regression. In *Advances in Neural Information Processing Systems*, pages 927–935, 2011.

Adam Tauman Kalai and Ravi Sastry. The isotron algorithm: High-dimensional isotonic regression. In *Proceedings of the 22nd Annual Conference on Learning Theory*, 2009.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT press, 2009.

Bing Li and Yuexiao Dong. Dimension reduction for nonelliptically distributed predictors. *The Annals of Statistics*, pages 1272–1298, 2009.

Ker-Chau Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316–327, 1991.

Ker-Chau Li and Naihua Duan. Regression analysis under link violation. *Annals of Statistics*, 17:1009–1052, 1989.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances In Neural Information Processing Systems*, pages 2370–2378, 2016.

Qiang Liu, Jason D Lee, and Michael I Jordan. A kernelized stein discrepancy for goodness-of-fit tests and model evaluation. *arXiv preprint arXiv:1602.03253*, 2016.

James Martens. Deep learning via hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 735–742, 2010.

Peter McCullagh and John A. Nelder. *Generalized Linear Models*. Chapman and Hall, 2nd edition, 1989.

John A Nelder and R. Jacob Baker. *Generalized linear models*. Wiley Online Library, 1972.

Yurii Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Math. Dokl.*, 27:372–376, 1983.

Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2004.

Christopher C Paige and Michael A Saunders. Solution of sparse indefinite systems of linear equations. *SIAM journal on numerical analysis*, 12(4):617–629, 1975.

Mert Pilanci and Martin J Wainwright. Newton sketch: A linear-time optimization algorithm with linear-quadratic convergence. *arXiv preprint arXiv:1505.02250*, 2015.

Yaniv Plan and Roman Vershynin. The generalized lasso with non-linear observations. *IEEE Transactions on information theory*, 62(3):1528–1537, 2016.

Mark D Reid and Robert C Williamson. Composite binary losses. *Journal of Machine Learning Research*, 11(Sep):2387–2422, 2010.

Vladimir Rokhlin and Mark Tygert. A fast randomized algorithm for overdetermined linear least-squares regression. *Proceedings of the National Academy of Sciences*, 105(36):13212–13217, 2008.

Farbod Roosta-Khorasani and Michael W Mahoney. Sub-sampled newton methods i: globally convergent algorithms. *arXiv preprint arXiv:1601.04737*, 2016a.

Farbod Roosta-Khorasani and Michael W Mahoney. Sub-sampled newton methods ii: Local convergence rates. *arXiv preprint arXiv:1601.04738*, 2016b.

Mark J Schervish. A general method for comparing probability assessors. *The Annals of Statistics*, pages 1856–1879, 1989.

Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Stochastic convex optimization. In *Conference on Learning Theory (COLT)*, 2009.

David F Shanno. Conditioning of quasi-newton methods for function minimization. *Mathematics of computation*, 24(111):647–656, 1970.

Weijie Su and Emmanuel Candes. Slope is adaptive to unknown sparsity and asymptotically minimax. *The Annals of Statistics*, 44(3):1038–1068, 2016.

Christos Thrampoulidis, Ehsan Abbasi, and Babak Hassibi. Lasso with non-linear measurements is equivalent to one with linear measurements. In *Advances in Neural Information Processing Systems*, pages 3420–3428, 2015.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices, 2010. arXiv:1011.3027.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 2008.

Journal of Machine Learning Research 20 (2019) 1-39 Submitted 6/17; Revised 10/18; Published 1/19

Forward-Backward Selection with Early Dropping

Giorgos Borboudakis borbudak@gmail.com
Ioannis Tsamardinos tsamard.it@gmail.com
Computer Science Department, University of Crete
Gnosis Data Analysis

Editor: Isabelle Guyon

Abstract

Forward-backward selection is one of the most basic and commonly-used feature selection algorithms available. It is also general and conceptually applicable to many different types of data. In this paper, we propose a heuristic that significantly improves its running time, while preserving predictive performance. The idea is to temporarily discard the variables that are conditionally independent with the outcome given the selected variable set. Depending on how those variables are reconsidered and reintroduced, this heuristic gives rise to a family of algorithms with increasingly stronger theoretical guarantees. In distributions that can be faithfully represented by Bayesian networks or maximal ancestral graphs, members of this algorithmic family are able to correctly identify the Markov blanket in the sample limit. In experiments we show that the proposed heuristic increases computational efficiency by about 1-2 orders of magnitude, while selecting fewer or the same number of variables and retaining predictive performance. Furthermore, we show that the proposed algorithm and feature selection with LASSO perform similarly when restricted to select the same number of variables, making the proposed algorithm an attractive alternative for problems where no (efficient) algorithm for LASSO exists.

Keywords: Feature Selection, Forward Selection, Markov Blanket Discovery, Bayesian Networks, Maximal Ancestral Graphs

1. Introduction

The problem of feature selection (a.k.a. variable selection) in supervised learning tasks can be defined as the problem of selecting a minimal-size subset of the variables that leads to an optimal, multivariate predictive model for a target variable (outcome) of interest (Tsamardinos and Aliferis, 2003). Thus, the feature selection's task is to filter out irrelevant variables and variables that are superfluous given the selected ones (that is, weakly relevant variables, see (John et al., 1994; Tsamardinos and Aliferis, 2003)).

Solving the feature selection problem has several advantages. Arguably, the most important one is knowledge discovery: by removing superfluous variables it improves intuition and understanding about the data-generating mechanisms. This is no accident as solving the feature selection problem has been linked to the data-generating causal network (Tsamardinos and Aliferis, 2003). In fact, *it is often the case that the primary goal of data analysis is feature selection* and not the actual resulting predictive model. This is particularly true in medicine and biology where the features selected may direct future experiments and studies. Feature selection is also employed to reduce the cost of measuring the features to make operational a predictive model; for example, it can reduce the monetary cost or inconvenience to a patient of applying a diagnostic model by reducing the number of medical tests and measurements required to perform on a subject for providing a diagnosis. Feature selection also often improves the predictive performance of the resulting model in practice, especially in high-dimensional settings. This is because a good-quality selection of features facilitates modeling, particularly for algorithms susceptible to the curse of dimensionality. There has been a lot of research on feature selection methods in the statistical and machine learning literature. An introduction to the topic, as well as a review of many, prominent methods can be found in (Guyon and Elisseeff, 2003), while the connections between feature selection, the concept of relevancy, and probabilistic graphical models is in (John et al., 1994; Tsamardinos and Aliferis, 2003).

We will focus on forward and backward selection algorithms, which are specific instances of stepwise methods (Kutner et al., 2004; Weisberg, 2005). These methods are some of the oldest, simplest and most commonly employed feature selection methods. An attractive property of stepwise methods is that they are very general, and are applicable to different types of data. For instance, stepwise methods using conditional independence tests can be directly applied to (a) mixed continuous and categorical predictors, (b) cross-sectional or time course data, (c) continuous, nominal, ordinal or time-to-event outcomes, among others, (d) with non-linear tests, such as kernel-based methods (Zhang et al., 2011), and (e) to heteroscedastic data using robust tests; many of the aforementioned tests, along with others have been implemented in the MXM R package (Lagani et al., 2017). The main drawback of forward selection is its computational cost. In order to select k variables, it performs $O(pk)$ tests for variable inclusion, where p is the total number of variables in the input data. This is acceptable for low-dimensional datasets, but becomes unmanageable with increasing dimensionality. Another issue is that forward selection suffers from multiple testing problems and thus may select a large number of irrelevant variables (Flom and Cassell, 2007).

In computer science, forward-backward selection has re-appeared in the context of Markov blanket discovery and Bayesian network learning (Margaritis and Thrun, 2000; Tsamardinos et al., 2003b; Margaritis, 2009), and has been shown to be optimal for distributions that can be faithfully represented by causal graphs. In the signal processing community forward selection is known as orthogonal least squares (Chen et al., 1989). Other algorithms, such as LASSO (Tibshirani, 1996), least-angle regression (LARS) (Efron et al., 2004), forward stagewise regression (FSR) (Efron et al., 2004) and orthogonal matching pursuit (OMP) (Pati et al., 1993; Davis et al., 1994) are all variations of the basic stepwise selection algorithm, while information-theoretic feature selection methods are all approximations of the forward phase using discrete data (Brown et al., 2012). A detailed comparison between LASSO, LARS and FSR is given in (Efron et al., 2004), a comparison between LARS and OMP can be found in (Hameed, 2012), while a comparison between OMP and forward selection (called orthogonal least squares) can be found in (Blumensath and Davies, 2007). We proceed with a brief high-level comparison of the above with the forward selection algorithm. All of the above methods select the next feature using some selection criterion and are equipped with a stopping criterion. Intuitively, they all select the feature that provides the most information for the errors (residuals) of the current model. Forward selection on the other hand, selects the feature that leads to a model providing

©2019 Giorgos Borboudakis and Ioannis Tsamardinos.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-334.html.

the most additional information, given all selected variables. In LASSO, both forward and backward steps can be performed at each iteration. After a feature is selected, forward selection and OMP create a new unrestricted model that also contains the newly selected feature. LASSO, LARS and FSR create a new model by updating the previous one, constraining the coefficients of the new model. LASSO has a stopping criterion based on the L1-norm of the coefficients of the current variables. *Given this synthetic view and connections between the algorithms, we would like to note that any extension to stepwise methods, such as the one proposed in this work, can be translated and directly applied with any of the above feature selection algorithms.*

In this work we extend the forward selection algorithm to deal with the problems above. In Section 3 we propose **early dropping**, a simple heuristic to speed-up forward selection, without sacrificing its quality and maintaining its theoretical guarantees. The idea is, in each iteration of the forward search, to filter out variables that are deemed conditionally independent of the target given the current set of selected variables. After termination, the algorithm is allowed to run up to K additional times, every time initializing the set of selected variables to the ones selected in the previous run. Finally, backward selection is applied on the result of the forward phase. We call this algorithm **Forward-Backward selection with Early Dropping** (FBED^K)[1]. This heuristic is inspired by the theory of Bayesian networks and maximal ancestral graphs (Spirtes et al., 2000; Richardson and Spirtes, 2002), and similar ideas have been successfully applied for feature selection (Aliferis et al., 2010). In Section 3.2 we show that (a) FBED^0 returns a superset of the adjacent nodes in any Bayesian network or maximal ancestral graph that faithfully represents the data distribution (if there exists one and assuming perfect statistical independence tests), (b) FBED^1 returns the Markov blanket of the data distribution, provided the distribution is faithful to a Bayesian network, and (c) FBED$^\infty$ returns the Markov blanket of the data distribution provided the distribution is faithful to a maximal ancestral graph, or equivalently, it is faithful to a Bayesian network where some variables are unobserved (latent). In the experimental evaluation presented in Section 4, we show that FBED^K is 1-2 orders of magnitude faster than FBS, while at the same time selecting a similar number of features and having similar predictive performance. In a comparison between different members of the FBED^K family and FBS we show that FBED^0 and FBED^1 also reduce the number of false variable selections, when the data consist of irrelevant variables only. We also investigated the behavior of FBED^K with increasing number of runs K, showing that a relatively small K is sufficient in most cases to reach optimal predictive performance. Afterwards, we compare FBED^K to FBS, feature selection with LASSO (Tibshirani, 1996) and to the Max-Min Parents and Children algorithm (MMPC) (Tsamardinos et al., 2003a) and show that it often has comparable predictive performance while selecting the fewest variables overall. Finally, we compare FBED^K to feature selection with LASSO (Tibshirani, 1996) when both algorithms are limited to select the same number of variables, showing that both algorithms perform similarly. This, along with the generality of FBED^K makes it an interesting alternative to LASSO, especially for problems where LASSO requires specialized algorithms (like the group LASSO algorithm for logistic regression (Meier et al., 2008), LASSO for mixed-effects linear models (Schelldorfer et al., 2011), and the LASSO and group LASSO methods for functional Poisson regression (Ivanoff et al., 2016)), which may be non-convex (as is the case for mixed-effects linear models (Schelldorfer et al., 2011) or for temporal-longitudinal data (Groll and Tutz, 2014; Tsagris et al., 2018)) and computationally demanding (taking several days to terminate on datasets with just 1000 predictors using Cox's proportional hazards model (Fan et al., 2010)).

1. The early dropping heuristic has also used by an extension of FBED for Big Data settings and map-reduce architectures (Tsamardinos et al., 2018a). However, the main idea and motivation behind it, its theoretical properties and a thorough experimental evaluation are presented in this work.

2. Notation and Preliminaries

We start by introducing the notation and terminology used throughout the paper. We use upper-case letters to denote single variables (for example, X), and bold upper-case letters to denote sets of variables (for example, $\mathbf{Z}$). The terms variable, feature or predictor will be used interchangeably. We will use p and n to refer to the number of variables and samples in a dataset $\mathcal{D}$, respectively. The set of variables in $\mathcal{D}$ will be denoted as $\mathbf{V}$. The target variable (also called outcome) will be referred to as T. Next, we proceed with the basics about stepwise feature selection methods (Kutner et al., 2004; Weisberg, 2005).

2.1. Stepwise Feature Selection

Stepwise methods start with some set of selected variables and try to improve it in a greedy fashion, by either including or excluding a single variable at each step. There are various ways to combine those operations, leading to different members of the stepwise algorithmic family. Two popular members of the stepwise family are the **forward selection** and **backward selection** (also known as backward elimination) algorithms. Forward selection starts with a (usually empty) set of variables and adds variables to it, until some stopping criterion is met. Similarly, backward selection starts with a (usually complete) set of variables and then excludes variables from that set, again, until some stopping criterion is met. Typically, both methods try to include or exclude the variable that offers the highest performance increase. We will call each step of selecting (removing) a variable a forward (backward) **iteration**. Executing forward (backward) iterations until termination will be called a forward (backward) **phase** respectively. An instance of the stepwise family, which we focus on hereafter, is the **Forward-Backward Selection** algorithm (FBS), which first performs a forward phase and then a backward phase on the selected variables. This algorithm is not new; similar algorithms have appeared in the literature before (Margaritis and Thrun, 2000; Tsamardinos et al., 2003b; Margaritis, 2009).

FBS is shown in Algorithm 1. The function Perf evaluates a set of variables and returns their performance relative to some statistical model. Examples are the log-likelihood for logistic regression, the partial log-likelihood for Cox regression and the F-score for linear regression, or their AIC (Akaike, 1973) or BIC (Schwarz et al., 1978) penalized variants. The **selection criterion** C compares the performance of two sets of variables as computed by Perf. For instance, in the previous example C could perform a likelihood ratio test and use a predetermined significance level α to make a decision [2]; we will describe such selection criteria in the next subsection. We will use the predicates $\underset{\text{C}}{>}$, $\underset{\text{C}}{\geq}$ and $\underset{\text{C}}{=}$ to compare two sets

2. In general, the type of criteria used in practice are not limited to that. For example, one may also stop after a fixed number of variables have been selected.

Algorithm 1 Forward-Backward Selection (FBS)

Input: Dataset $\mathcal{D}$, Target T
Output: Selected Variables $\mathbf{S}$

$\mathbf{S} \leftarrow \emptyset$ *//Set of selected variables*
$\mathbf{R} \leftarrow \mathbf{V}$ *//Set of remaining candidate variables*

//Forward phase: iterate until $\mathbf{S}$ *does not change*
while $\mathbf{S}$ changes **do**
  //Identify the best variable V_{best} *out of all remaining variables* $\mathbf{R}$*, according to* Perf
  $V_{best} \leftarrow \underset{V \in \mathbf{R}}{\text{argmax}}\ \textsc{Perf}(\mathbf{S} \cup V)$
  //Select V_{best} *if it increases performance according to criterion* C
  if $\textsc{Perf}(\mathbf{S} \cup V_{best}) \underset{\text{C}}{>} \textsc{Perf}(\mathbf{S})$ **then**
    $\mathbf{S} \leftarrow \mathbf{S} \cup V_{best}$
    $\mathbf{R} \leftarrow \mathbf{R} \setminus V_{best}$
  end if
end while

//Backward phase: iterate until $\mathbf{S}$ *does not change*
while $\mathbf{S}$ changes **do**
  //Identify the worst variable V_{worst} *out of all selected variables* $\mathbf{S}$*, according to* Perf
  $V_{worst} \leftarrow \underset{V \in \mathbf{S}}{\text{argmax}}\ \textsc{Perf}(\mathbf{S} \setminus V)$
  //Remove V_{worst} *if it does not decrease performance according to criterion* C
  if $\textsc{Perf}(\mathbf{S} \setminus V_{worst}) \underset{\text{C}}{\geq} \textsc{Perf}(\mathbf{S})$ **then**
    $\mathbf{S} \leftarrow \mathbf{S} \setminus V_{worst}$
  end if
end while
return $\mathbf{S}$

of variables; they are true if the left-hand-side value is greater, greater or equal, or equal than the right-hand-side value respectively, according to the criterion C.

2.2. Criteria for Variable Selection

Next we will briefly describe some performance functions and selection criteria that are employed in practice; for more details see (Kutner et al., 2004; Weisberg, 2005). The most common choices are statistical tests, information criteria and cross-validation. We will focus on statistical tests, as we use them in the remainder of the paper. We will also describe information criteria and contrast them to statistical tests, but will not further consider cross-validation, mainly because of its high computational cost.

2.2.1. Statistical Tests

Since the models tested at each iteration are nested, one can employ a likelihood-ratio (LR) test (or asymptotically equivalent approximations thereof such as score tests and Wald

tests) for nested models as a selection criterion. We next describe the likelihood-ratio test in more depth. For the LR test, the performance Perf is related to the log-likelihood (LL) and the criterion C tests the hypothesis that both models are equivalent with respect to some pre-specified significance level α. Let $\textsc{Dev}(T|\mathbf{X}) \equiv -2 \cdot \text{LL}(T|\mathbf{X})$ and $\textsc{Par}(T|\mathbf{X})$ be the deviance and number of parameters respectively of a model for target T using variables $\mathbf{X}$. Then, the test statistic of a nested test for models with variables $\mathbf{X}$ (null model) and $\mathbf{X} \cup \mathbf{Y}$ (alternative model) is computed as the difference in deviance of both models, that is, $\textsc{Dev}(T|\mathbf{X}) - \textsc{Dev}(T|\mathbf{X} \cup \mathbf{Y})$, and follows asymptotically a χ^2 distribution with $\textsc{Par}(T|\mathbf{X} \cup \mathbf{Y}) - \textsc{Par}(T|\mathbf{X})$ degrees of freedom (Wilks, 1938) [3].

Tests for nested models are essentially **conditional independence tests**, relative to some statistical model (for example, using linear regression without interaction terms tests for linear dependence), and assuming that the model is correctly specified. If the null model contains the predictors $\mathbf{X}$ and the alternative model contains $\mathbf{X} \cup \mathbf{Y}$, the nested test tests the hypothesis that the coefficients of $\mathbf{Y}$ are zero, or equivalently, that $\mathbf{Y}$ is conditionally independent of the target T given $\mathbf{X}$. We denote **conditional independence** of two non-empty sets $\mathbf{X}$ and $\mathbf{Y}$ given a (possibly empty) set $\mathbf{Z}$ as $(\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z})$. Finally, we note that one is not limited to likelihood-ratio based conditional independence tests, but can use any appropriate conditional independence test, such as a kernel-based test (Zhang et al., 2011).

A problem when using statistical tests for feature selection is that, due to multiple testing, the test statistics do not have the claimed distribution (Hastie et al., 2009) and the resulting p-values are too small (Harrell, 2001; Flom and Cassell, 2007), leading to a high false discovery rate. Approaches to deal with problem include methods that dynamically adjusting significance levels (Hwang and Hu, 2015), or methods that directly deal with the problem of sequential testing of stepwise procedures (G'Sell et al., 2016; Tibshirani et al., 2016). In order to perform tests on the model returned by stepwise selection, one can use resampling-based procedures to correct the p-values (Finos et al., 2010). In addition to the above problems, we note that the model returned by stepwise selection is sub-optimal, as it will have inflated coefficients (Flom and Cassell, 2007), reducing its predictive ability. If the main focus is to obtain a predictive model, methods performing regularization (like L1, L2 or elastic net) are more appropriate. In any case, procedures like cross-validation should be used to estimate out-of-sample predictive performance of the final model. We will not consider the above in this paper; we note however that our proposed algorithm is orthogonal to those methods and could be used in conjunction with them.

2.2.2. Information Criteria

Another way to compare two (or more) competing models is to use information criteria, such as the Akaike information criterion (AIC) (Akaike, 1973) or the Bayesian information criterion (BIC) (Schwarz et al., 1978). Information criteria are based on the fit of a model but additionally penalize the model by its complexity (that is, the number of parameters).

3. This result assumes that the larger hypothesis is correctly specified. In case of model misspecification, the statistic follows a different distribution (Foutz and Srivastava, 1977). Methods to handle model misspecification have been proposed by White (1982) and Vuong (1989). A method for dealing with model misspecification in model selection with information criterion is presented in (Lv and Liu, 2014). As this problem is out of this paper's scope, we did not further consider it.

The AIC and BIC scores of a model for T based on $\mathbf{X}$ are defined as follows:

$$\text{AIC}(T|\mathbf{X}) \equiv \textsc{Dev}(T|\mathbf{X}) + 2 \cdot \textsc{Par}(T|\mathbf{X})$$
$$\text{BIC}(T|\mathbf{X}) \equiv \textsc{Dev}(T|\mathbf{X}) + \log(n) \cdot \textsc{Par}(T|\mathbf{X})$$

where n is the number of samples. In the framework described above, information criteria can be applied by using the information criterion value as the performance function Perf, and a selection criterion C that simply compares the performance of two models, giving preference to the one with the lowest value. Alternatively, one could check that the difference in scores is larger than some threshold.

Neither AIC nor BIC are designed for cases where the number of predictors p is larger than the number of samples n (Chen and Chen, 2008), and thus also suffer from a high false discovery rate, similar to statistical tests. There have been several extensions to handle this problem, like the extended Bayesian information criterion (EBIC) (Chen and Chen, 2008), the generalized information criterion (GIC) (Fan and Tang, 2013; Kim et al., 2012), and the corrected risk information criterion (RIC_c) (Zhang and Shen, 2010), to name a few.

Compared to statistical tests, information criteria are somewhat limited as they can only be computed for models where the model complexity is known (like generalized linear models). An example where information criteria are not applicable are kernel-based tests (Zhang et al., 2011). Thus, statistical tests are inherently more general than information criteria. We will show next how, in case of nested models, using BIC directly corresponds to a likelihood-ratio test for some significance level α; the same reasoning can be applied to AIC and all information criteria that are computed based on the model likelihood and a penalty term. Let $\mathbf{X}$ and $\mathbf{X} \cup \mathbf{Y}$ be two candidate variables sets. $\mathbf{X} \cup \mathbf{Y}$ is selected (that is, the null hypothesis is rejected) if $\text{BIC}(T|\mathbf{X} \cup \mathbf{Y}) < \text{BIC}(T|\mathbf{X})$, or equivalently if $\textsc{Dev}(T|\mathbf{X}) - \textsc{Dev}(T|\mathbf{X} \cup \mathbf{Y}) > \log(n) \cdot (\textsc{Par}(T|\mathbf{X} \cup \mathbf{Y}) - \textsc{Par}(T|\mathbf{X}))$. Note that the left-hand side term equals the statistic of a likelihood-ratio test, whereas the right-hand size corresponds to the critical value. The statistic follows a χ^2 distribution with $k = \textsc{Par}(T|\mathbf{X} \cup \mathbf{Y}) - \textsc{Par}(T|\mathbf{X})$ degrees of freedom, and thus, the significance level equals $\alpha = 1 - F(\log(n) \cdot k; k)$, where $F(v; k)$ is the χ^2 cdf with k degrees of freedom at value v.

2.3. Markov Blankets in Bayesian Networks and Maximal Ancestral Graphs

The proposed algorithm is inspired by the theory of Markov blankets in Bayesian networks and maximal ancestral graphs. Next, we will provide a brief introduction of them; more details can be found in Appendix A. For a comprehensive introduction to Bayesian networks and maximal ancestral graphs we refer the reader to (Pearl, 1988; Spirtes et al., 2000; Richardson and Spirtes, 2002).

A **directed acyclic graph** (DAG) is a graph that only contains directed edges ($\rightarrow$) and has no directed cycles. A **directed mixed graph** is a graph that, in addition to directed edges also contains bi-directed edges ($\leftrightarrow$). A triplet of vertices $\langle X, Y, Z \rangle$ is called a **collider** if there are directed or bi-directed edges from X and Z to Y. A path p is called a **collider path** if every non-endpoint vertex is a collider on p.

Bayesian networks (BNs) consist of a DAG $\mathcal{G}$ and a probability distribution $\mathcal{P}$ over a set of variables $\mathbf{V}$. A DAG $\mathcal{G}$ is Markov and faithful to $\mathcal{P}$ if (a) each variable is conditionally independent of its non-descendants given its parents, and (b) all and only those conditional

independencies in $\mathcal{P}$ are entailed by $\mathcal{G}$. BNs are not closed under marginalization, that is, they are not able to encode latent confounders. BNs have been extended to represent such marginal distributions, and are called **directed maximal ancestral graphs** (DMAGs) (Richardson and Spirtes, 2002). The graphical structure of a DMAG is a directed mixed graph with the following restrictions: (i) it contains no directed cycles, (ii) it contains no almost directed cycles, that is, if $X \leftrightarrow Y$ then neither X nor Y is an ancestor of the other, and (iii) there is no path p such that each non-endpoint on p is a collider and every collider is an ancestor of an endpoint vertex of p.

A **Markov blanket** of a variable T is a **minimal** set of variables $\mathbf{MB}(T)$ that renders T conditionally independent of all remaining variables $\mathbf{V} \setminus \mathbf{MB}(T)$. In case the distribution can be faithfully represented by a BN or DMAG, then the Markov blanket of T is **unique**, and is defined as follows (see Appendix A for a proof sketch).

Definition 1 (Markov blanket) *The Markov blanket of T in a BN or DMAG consists of all vertices adjacent to T, as well as all vertices that are reachable from T through a collider path.*

In case of Bayesian networks, this simplifies to the set of parents, children, and parents of children of T.

3. Speeding-up Forward-Backward Selection

The standard FBS algorithm has two main issues. The first is that it is slow: at each forward iteration, all remaining variables are reconsidered to find the best next candidate. If k is the total number of selected variables and p is the number of input variables, the number of model evaluations (or in our case, independence tests) FBS performs is of the order of $O(kp)$. Although relatively low-dimensional datasets are manageable, it can be very slow for modern datasets which often contain thousands of variables. The second problem is that it suffers from multiple testing issues, resulting in overfitting and a high false discovery rate. This happens because it reconsiders all remaining variables at each iteration; variables will often happen to seem important simply by chance, if they are given enough opportunities to be selected. As a result, it will often select a significant number of false positive variables (Flom and Cassell, 2007). This behavior is further magnified in high-dimensional settings and with larger significance levels α. Next, we describe a simple modification of FBS, improving its running time while reducing the problem of multiple testing.

3.1. The Early Dropping Heuristic

We propose the following modification: after each forward iteration, remove all variables that do not satisfy the criterion C for the current set of selected variables $\mathbf{S}$ from the remaining variables $\mathbf{R}$. In our case, those variables are the ones that are conditionally independent of T given $\mathbf{S}$. The idea is to quickly reduce the number of candidate variables $\mathbf{R}$, while keeping many (possibly) relevant variables in it. The forward phase terminates if no more variables can be selected, either because there is no informative variable or because $\mathbf{R}$ is empty; to distinguish between forward and backward phases, we will call a forward phase

Algorithm 2 Forward-Backward Selection with Early Dropping (FBED^K)

Input: Dataset $\mathcal{D}$, Target T, Maximum Number of Runs K
Output: Selected Variables $\mathbf{S}$

$\mathbf{S} \leftarrow \emptyset$ *//Set of selected variables*
$K_{cur} \leftarrow 0$ *//Initializing current number of runs to 0*

//Forward phase: iterate until (a) run limit reached, or (b) $\mathbf{S}$ *does not change*
while $K_{cur} \leq K \wedge \mathbf{S}$ changes **do**
    $\mathbf{S} \leftarrow$ OneRun$(\mathcal{D}, T, \mathbf{S})$
    $K_{cur} \leftarrow K_{cur} + 1$
end while

//Perform backward selection and return result
return BackwardSelection$(\mathcal{D}, \mathbf{T}, \mathbf{S})$

function OneRun($\mathcal{D}$, T, $\mathbf{S}$)
    $\mathbf{R} \leftarrow \mathbf{V} \setminus \mathbf{S}$ *//Set of remaining candidate variables*

    //Forward phase: iterate until $\mathbf{R}$ *is empty*
    while $|\mathbf{R}| > 0$ **do**
        //Identify best variable V_{best} *out of* $\mathbf{R}$*, according to* Perf
        $V_{best} \leftarrow \underset{V \in \mathbf{R}}{\text{argmax}}$ Perf$(\mathbf{S} \cup V)$
        //Select V_{best} *if it increases performance according to criterion* C
        if Perf$(\mathbf{S} \cup V_{best}) \underset{C}{>}$ Perf$(\mathbf{S})$ **then**
            $\mathbf{S} \leftarrow \mathbf{S} \cup V_{best}$
        end if
        //Drop all variables from $\mathbf{R}$ *not satisfying* C
        $\mathbf{R} \leftarrow \{V : V \in \mathbf{R} \wedge V \neq V_{best} \wedge$ Perf$(\mathbf{S} \cup V) \underset{C}{>}$ Perf$(\mathbf{S})\}$
    end while
    return S
end function

with early dropping a **run**. Extra runs can be performed to reconsider variables dropped previously. This is done by retaining the previously selected variables $\mathbf{S}$ and initializing the set of remaining variables to all variables which have not been selected yet, that is $\mathbf{R} = \mathbf{V} \setminus \mathbf{S}$. The backward phase employed afterwards is identical to the standard backward-selection algorithm (see Algorithm 1). Depending on the number of additional runs K, this defines a family of algorithms, which we call **Forward Backward Selection with Early Dropping** (FBED^K), shown in Algorithm 2. The function OneRun shown in the bottom of Algorithm 2, performs one run until no variables remain in $\mathbf{R}$. Three interesting members of this family are the FBED^0, FBED^1 and FBED$^\infty$ algorithms. FBED^0 performs the first run until termination, FBED^1 performs one additional run and FBED$^\infty$ performs runs until no more variables can be selected. We will focus on those three algorithms hereafter.

The heuristic is inspired by the theory of Bayesian networks and maximal ancestral graphs (Spirtes et al., 2000; Richardson and Spirtes, 2002). Similar heuristics have been applied by Markov blanket based algorithms such as the Max-Min Parents and Children (MMPC) algorithm (Tsamardinos et al., 2003a) and HITON (Aliferis et al., 2003) successfully in practice and in extensive comparative evaluations (Aliferis et al., 2010). These algorithms also remove variables from consideration, and specifically the ones that are conditionally independent given some **subset** of the selected variables. In contrast, FBED^K reconsiders variables dropped during previous runs, while existing methods do not. Thus, *FBED^K bridges two types of algorithms to combine their advantages*: those that condition on all currently selected variables (such as FBS, grow-shrink (Margaritis and Thrun, 2000) and incremental association Markov blanket (Tsamardinos et al., 2003b)), and those that condition on subsets of variables to drop some of them (like MMPC (Tsamardinos et al., 2003a) and HITON (Aliferis et al., 2003)). Doing so, FBED^K manages to have the theoretical properties of the former (as shown in the next subsection), while obtaining speed-ups similar to the latter.

3.2. Comparing the Theoretical Properties of FBED^K to FBS

Due to early dropping of variables, the distributions under which FBED^K and FBS perform optimally are not the same. For all versions of FBED^K, with the exception of FBED$^\infty$, it is relatively straightforward to construct examples where FBS is able to identify variables that can not be identified by FBED^K. We give an example for FBED^0. FBED^0 may remove variables that seem uninformative at first, but become relevant if considered in conjunction with other variables. For example, let $X = T + Y$, with T and Y being independent Gaussian random variables, and assume that T is the outcome for which variable selection is performed. When no variables have been selected (first iteration), X will be dependent with T, while Y will be independent of T as it does not give any information about T by itself, and thus will be dropped. However, after selecting X, Y becomes conditionally dependent again (as $T = X - Y$), but FBED^0 will not select it as it was dropped in the first iteration. Surprisingly, in practice this does not seem to significantly affect the quality of FBED^0. In contrast, FBED^0 often gives better results, while also selecting fewer variables than FBS (see Section 4.4).

As mentioned above, it is not clear how FBS and FBED$^\infty$ are related in the general case. What can be shown is that both identify a minimal set of variables, although the identified solutions may not necessarily be the same.

Definition 2 (Minimal Variable Set) *Let* $\mathbf{V}$ *be the set of all variables and* $\mathbf{S}$ *a set of selected variables. We call a set of variables* $\mathbf{S}$ *minimal with respect to some outcome* T*, if:*

1. *No variable can be removed from* $\mathbf{S}$ *given the rest of the selected variables, that is,* $\forall V_i \in \mathbf{S}, (T \not\perp V_i \mid \mathbf{S} \setminus V_i)$ *holds.*

2. *Let* $\mathbf{R} = \mathbf{V} \setminus \mathbf{S}$*. No variable from* $\mathbf{R}$ *can be included in* $\mathbf{S}$*, that is,* $\forall V_i \in \mathbf{R}, (T \perp V_i \mid \mathbf{S})$ *holds.*

Corollary 3 *Any set of variables* $\mathbf{S}$ *selected by FBS is minimal.*

Proof See Appendix B. ■

Corollary 4 *Any set of variables* $\mathbf{S}$ *selected by FBED*$^{\infty}$ *is minimal.*

Proof See Appendix B. ■

In words, a minimal set is a set such that no single variable can be included to or removed from using forward and backward iterations respectively, that is, it is a local optimum for stepwise algorithms. Note that, although no single variable is informative for T if looked at separately, there may be sets of variables that are informative if considered jointly. A simple example is if all variables are binary and $T = X \oplus Y$, where $\oplus$ is the logical XOR operator. In this case $\mathbf{S} = \emptyset$ is minimal, as neither X nor Y are dependent with T, even though the set $\{X, Y\}$ fully determines T. Thus, none of the algorithms gives a globally optimal solution in all distributions.

We next consider the special case in which distributions can be represented by Bayesian networks or maximal ancestral graphs. We show that FBED^1 and FBED$^{\infty}$ identify the Markov blanket of a BN and DMAG respectively, assuming (a) that the distribution can be faithfully represented by the respective graph, and (b) that the algorithms have access to an **independence oracle**[4], which correctly determines whether a given conditional (in)dependence holds. This also holds for FBS but will not be shown here; proofs for similar algorithms exist (Margaritis and Thrun, 2000; Tsamardinos et al., 2003b) and can be easily adapted to FBS. For FBED^0 it can be shown that it selects a superset of the variables that are adjacent to T in the graph; this can be shown using the fact that, under the Markov and faithfulness assumptions, adjacent variables are dependent with T given any subset of the remaining variables.

Theorem 5 *If the distribution can be faithfully represented by a Bayesian network, then FBED*^{1} *identifies the Markov blanket of the target* T*.*

Proof See Appendix B. ■

Theorem 6 *If the distribution can be faithfully represented by a directed maximal ancestral graph, then FBED*$^{\infty}$ *identifies the Markov blanket of the target* T*.*

4. Assuming access to an independence oracle allows one to analyze whether the strategy used by FBED^K for identifying a Markov blanket is correct; thus, in practice, any errors in the output are due to statistical errors of the tests and not due to the heuristics or strategy used by FBED^K. Furthermore, it allows one to analyze the asymptotic behavior of algorithms without parametric distributional assumptions (for example, multivariate normality), but structural assumptions (for example, faithfulness). Assuming a conditional independence oracle is a standard assumption for the theoretical analysis of Markov blanket and causal discovery algorithms (for example, see (Spirtes et al., 2000; Aliferis et al., 2010)). In practice, FBED^K will not have access to an oracle, but will perform conditional independence tests to decide (in)dependence. There exist tests that, in the sample limit, will correctly identify (in)dependence. Examples include the partial correlation test for multivariate Gaussian data, and the G-test (Agresti, 2002) for multinomial data.

Proof See Appendix B. ■

3.3. Limitations and Practical Considerations

We have shown that FBED^K is able to solve the feature selection problem (that is, identify the Markov blanket of T) for distributions that are faithful to causal graphs. In practice, FBED^K may fail to identify the Markov blanket for several reasons. Naturally, in case the distribution can't be faithfully modeled with causal graphs, there is no guarantee of how close the solution will be to the optimal solution. However, previous comparisons show that forward selection performs as well as best subset selection, and is competitive with lasso (Hastie et al., 2017), indicating that its solutions are reasonably good approximations to the best subset solution, which we also confirm in the experimental section. Another, more subtle issue is if the conditional independence tests used are not appropriate to capture the dependencies present in the distribution. For instance, if all relations are non-linear and linear tests are used, there is no guarantee that any of the important variables will be selected. However, this is an issue with all feature selection algorithms (and predictive algorithms in general) and is not specific to FBED^K. Finally, if sample size is too low, or if the significance level is not set appropriately, dependencies may be incorrectly labeled as independencies and vice versa. Again, this is a general problem with all algorithms and can be handled by increasing sample size (if possible) and by appropriately setting or tuning the significance level. For example, for the task of learning Bayesian networks from Gaussian data using the PC algorithm (Spirtes et al., 2000), Kalisch and Bühlmann (2007) have shown that (under mild conditions) the significance level can be set in a way to ensure consistency asymptotically (Kalisch and Bühlmann, 2007, Theorem 1). The problem of learning Bayesian networks and Markov blanket discovery are closely related, and such results can possibly be translated and used by algorithms such as FBED^K, but it is out of the scope of the current paper.

We proceed with additional considerations regarding the sample size required to use FBED^K. FBED^K identifies the next variable to select conditional on all currently selected variables. Because of this, it can in principle take complex multivariate dependencies into consideration when selecting a variable. The complexity depends on the conditional independence test used (for example, non-linear tests can model more complex relations than linear tests). However, there is a clear trade-off between the complexity of dependencies that can be identified, and the sample size required to do so. For instance, if all variables are binary, the G-test of conditional independence (Agresti, 2002) can be used, which can identify any type of interaction between variables. In this case, the number of parameters increases exponentially with the number of selected variables: for k selected variables, the number of parameters is in the order of $O(2^k)$, and consequently, the number of samples required to have sufficient power also increases exponentially. Using linear models (for example, linear, logistic or Cox regression for continuous, categorical or time-to-event outcomes respectively), simpler, linear dependencies can be identified, and the number of samples required increases only linearly with the number of parameters. Rules of thumb for setting the minimum sample size for linear models are given in (Peduzzi et al., 1996; Harrell, 2001; Vittinghoff and McCulloch, 2007). For binary logistic regression, one recommendation is to

use at least $s = c/\min(p_0, p_1) \cdot k$ samples (Peduzzi et al., 1996), where p_0 and p_1 are the proportion of negative and positive classes of T respectively, k is the number of parameters in the model and c is a user-set parameter, which is usually recommended to be between 5 and 20, with larger values leading to more accurate results. Thus, multivariable methods like FBED^K should only be used when sufficient sample size is available; alternatively, one can use rules of thumb as stopping criteria (that is, to determine when to stop selecting variables).

Next, we make a few recommendations based on the above considerations; exact rules are hard to devise, as they depend on the specific problem at hand. In case sample size is very low (a few tens or hundreds of samples), sample-efficient methods like the max-min parents and children algorithm (Tsamardinos et al., 2003a) (which condition only on small subsets of variables), information-theoretic feature selection methods (Brown et al., 2012) (which only condition on up to 1 variable), or univariate feature selection methods are more preferable than methods like FBED^K. Otherwise, we recommend using linear multivariable methods like OMP (Pati et al., 1993; Davis et al., 1994), LASSO (Tibshirani, 1996) or FBED^K with linear tests, and if sample size allows to also consider FBED^K using non-linear tests. Finally, we believe it is also worth considering robust tests (Lagani et al., 2017) for FBED^K, as outliers often exist in practice and may negatively impact tests which do not take them into account.

4. Experimental Evaluation

In this section we evaluate FBED^K, and compare it to the standard FBS algorithm[5], feature selection with LASSO (called LASSO-FS hereafter) (Tibshirani, 1996), the Max-Min Parents and Children algorithm (MMPC) (Tsamardinos et al., 2003a), and no feature selection (NO-FS), which was used as the baseline method. We note that MMPC is designed specifically for low-sample size and high-dimensional settings, and thus may not perform optimally in the datasets considered here. The reason we compare against it is because, it belongs in the same category of algorithms as FBED^K (that is, is also inspired by causal graphs).

We implemented all algorithms in Matlab except for LASSO, for which we used the glmnet implementation (Qian et al., 2013). We used 12 binary classification datasets, with sample sizes ranging from 200 to 16772 and number of variables between 166 and 100000. The datasets were selected from various competitions (Guyon et al., 2004, 2006a) and the UCI repository (Dietterich et al., 1994), and were selected to cover a wide range of variable and sample sizes. A summary of the datasets is shown in Table 1. All experiments were performed in Matlab, running on a desktop computer with an Intel i7-7700K processor and 32GB of RAM.

5. We want to point out that, although the early dropping heuristic only affects the forward phase of the algorithm, we chose evaluate FBED^K and FBS with the backward phase. This is done mainly as FBED^K and FBS require the backward phase to have provable theoretical guarantees, and because that is how the algorithms are presented throughout the paper. To ensure that this will not significantly affect the results, we performed a few preliminary anecdotal experiments, and observed that (a) the number of features removed by backward selection is typically small, and (b) the predictive performance is very similar when backward selection is used.

Table 1: Binary classification datasets used in the experimental evaluation. n is the number of samples, p is the number of predictors and $P(T = 1)$ is the proportion of instances where $T = 1$.

Dataset	n	p	P(T = 1)	Type	Domain	Source
musk (v2)	6598	166	0.15	Real	Musk Activity Prediction	UCI ML Repository (Dietterich et al., 1994)
sylva	14394	216	0.94	Mixed	Forest Cover Types	WCCI 2006 Challenge (Guyon et al., 2006a)
madelon	2600	500	0.5	Integer	Artificial	NIPS 2003 Challenge (Guyon et al., 2004)
secom	1567	590	0.93	Real	Semi-Conductor Manufacturing	UCI ML Repository M. McCann, A. Johnston
gina	3568	970	0.51	Real	Handwritten Digit Recognition	WCCI 2006 Challenge (Guyon et al., 2006a)
hiva	4229	1617	0.96	Binary	Drug discovery	WCCI 2006 Challenge (Guyon et al., 2006a)
gisette	7000	5000	0.5	Integer	Handwritten Digit Recognition	NIPS 2003 Challenge (Guyon et al., 2004)
p53 Mutants	16772	5408	0.01	Real	Protein Transcriptional Activity	UCI ML Repository (Danziger et al., 2006)
arcene	200	10000	0.56	Binary	Mass Spectrometry	NIPS 2003 Challenge (Guyon et al., 2004)
nova	1929	16969	0.72	Binary	Text classification	WCCI 2006 Challenge (Guyon et al., 2006a)
dexter	600	20000	0.5	Integer	Text classification	NIPS 2003 Challenge (Guyon et al., 2004)
dorothea	1150	100000	0.9	Binary	Drug discovery	NIPS 2003 Challenge (Guyon et al., 2004)

The remainder of this section is organized as follows. First, we describe in detail the experimental setup, that is, all algorithms used, their hyper-parameters, and how we performed model selection and performance estimation. We proceed by evaluating how the running time, number of selected variables and predictive performance of FBED^K is affected by the number of runs K. Afterwards, we compare FBED^K to FBS, to show the effects of the early dropping heuristic. Next, we evaluate FBED^K in a realistic scenario with other feature selection methods, where hyper-parameters of the feature selection and classification algorithms are optimized. Then, we compare FBED^K and LASSO in terms of predictive ability when the number of variables to select is fixed so that all algorithms produce solutions of equal size. This is done for two reasons: (a) because LASSO tends to select many variables otherwise, giving it an advantage over FBED^K in terms of predictive performance, and (b) because this allows us to evaluate how well FBED^K orders the variables in comparison to LASSO. Finally, we compare FBED^0, FBED^1, FBED$^\infty$ and FBS on simulated data containing only irrelevant variables, investigating how is each algorithm is affected by multiple testing in terms of the falsely selected variables.

4.1. Experimental Setup

We present an overview of the experimental setup next. Additional details for each specific experiment are described in the respective section.

4.1.1. Feature selection algorithms

First of all we note that, although the early dropping heuristic only affects the forward phase of the algorithm, we will use FBED^K and FBS with their backward phase in the experiments. This is done mainly as FBED^K and FBS require the backward phase to have provable theoretical guarantees, and because that is how they are presented throughout the paper.

As selection criteria for FBED^K, FBS and MMPC we used a nested likelihood-ratio independence test based on logistic regression. For FBED^K and MMPC, the significance level α of the conditional independence test was set to $\{0.001, 0.005, 0.01, 0.05, 0.1\}$, covering a range of commonly used values, while for FBS we explored a total of 100 values, uniformly spaced in $[0.001, 0.01]$. For the K value of FBED^K we used $\{0, 1, \ldots, \infty\}$, while the maximum conditioning size $maxK$ of MMPC was set to $\{1, 2, 3, 4\}$. For LASSO-FS we set all parameters to their default values and set the maximum number of λ values, λ_{max}, to 100. Thus, we used 5 hyper-parameter combinations for each value K of FBED^K, 100 for FBS and LASSO-FS, and 20 for MMPC.

Unfortunately, for MMPC there were 2 datasets where not all hyper-parameter combinations were executed, as they were taking too long to terminate (see results about running time in Appendix D); we stopped execution if a time limit of 2 days was exceeded. Specifically, for the gisette and nova datasets MMPC was only executed with $maxK \leq 2$

We would like to point out that for a given value K for FBED^K, all solutions with fewer runs (smaller K) can be computed with minimal computational overhead, as the forward phases have already been computed and only the backward phases need to be performed separately. As the number of variables selected is relatively small, the computational cost of the backward phases is usually negligible. Thus, FBED^K required a single execution with $K = \infty$ for a given α. Unfortunately, something similar can not be done with MMPC, and thus it has to be executed for each hyper-parameter value separately [6].

4.1.2. Predictive models

We used both, linear and non-linear predictive models. As linear models we used elastic net regularized logistic regression (Zou and Hastie, 2005), using $\lambda_{max} = 100$ and the mixture parameter α set to $\{0, 0.25, 0.5, 0.75, 1\}$ ($\alpha = 0$ corresponds to L2 regularization and $\alpha = 1$ to L1 regularization), leading to a total of 500 hyper-parameter combinations. We remind the reader that regularization is important, especially after feature selection has been performed, in order to improve predictive performance due to inflated coefficients (Flom and Cassell, 2007) (see also Section 2.2). As non-linear models we used Gaussian support vector machines (SVM) (Cortes and Vapnik, 1995) and random forests (RF) (Breiman, 2001). For SVMs we used the LIBSVM (Chang and Lin, 2011) implementation, while for RFs we used the TreeBagger implementation in Matlab. The cost hyper-parameter C of SVMs was set to $\{2^{-10}, 2^{-9}, \ldots, 2^9\}$ (a total of 20 values), while the remaining hyper-parameters were set to their default values. For RFs the number of trees was set to 500, the minimum leaf node size was set to $\{1, 5, 9\}$ and the number of variables to split at each node was set to $\{0.5, 1, 1.5\} \cdot \sqrt{p}$ (9 combinations in total).

6. This is only partially true, as for FBED^K, FBS and MMPC we implemented a caching mechanism to avoid fitting the same logistic regression model more than once. This mechanism was not used however for experiments measuring the running time of the algorithms.

4.1.3. Linear vs non-linear models

Throughout the section, we will report results obtained by using only linear models or a combination of linear and non-linear models. The former is done to evaluate the ability of the feature selection methods of identifying features that are linearly (or possibly monotonically) related to the outcome. The reason for that is that all evaluated methods can only identify such types of dependencies; we note that all algorithms (except for LASSO) can be trivially adapted to also handle non-linear dependencies by using an appropriate conditional independence test. Non-linear models were also considered to better simulate a realistic scenario, as such methods would be used in a typical analysis. Furthermore, it is interesting to see whether there are any significant differences between linear and non-linear modeling for any of the considered feature selection algorithms.

4.1.4. Model selection and performance estimation protocols

Ideally, we would like to evaluate each feature selection algorithm using an optimal predictive model, in order to measure how informative the selected features are. As an optimal model is not available in practice, we approximate this by using a variety of predictive algorithms as well as multiple hyper-parameter value combinations for each (see above), and perform hyper-parameter optimization (also called tuning or model selection) to find a good approximate model; interested readers may refer to (Feurer and Hutter, 2018; Tsamardinos et al., 2018b) for more details. We proceed with a description of the model selection and performance estimation protocols we used.

As the performance metric we optimize and report the area under the ROC curve (AUC). For model selection and performance estimation we used a 60/20/20 stratified split of the data, using 60% as a training set, 20% as a validation set and the remaining 20% as a test set. A hyper-parameter configuration is defined as a combination of a feature selection algorithm and its hyper-parameters, as well as a modeling algorithm and its hyper-parameters. Given a set of configurations, the best one is chosen by training models for all of them on the training set and selecting the configuration of the model with the highest performance on the validation set. Finally, the predictive performance of that configuration is obtained by training a final model on the pooled training and validation sets, and evaluating it on the test set. To account for the variation due to the data splitting, we repeated this procedure multiple times for different splits and report averages over repetitions. For datasets with more than 1000 samples the number of repetitions was set to 10, and to 50 for the rest.

4.2. Effect of the Number of Runs K

We performed an experiment to measure the effect of K on the running time, number of selected variables and predictive performance of FBED^K. For the running time and number of selected variables we executed FBED^K once for each hyper-parameter value on

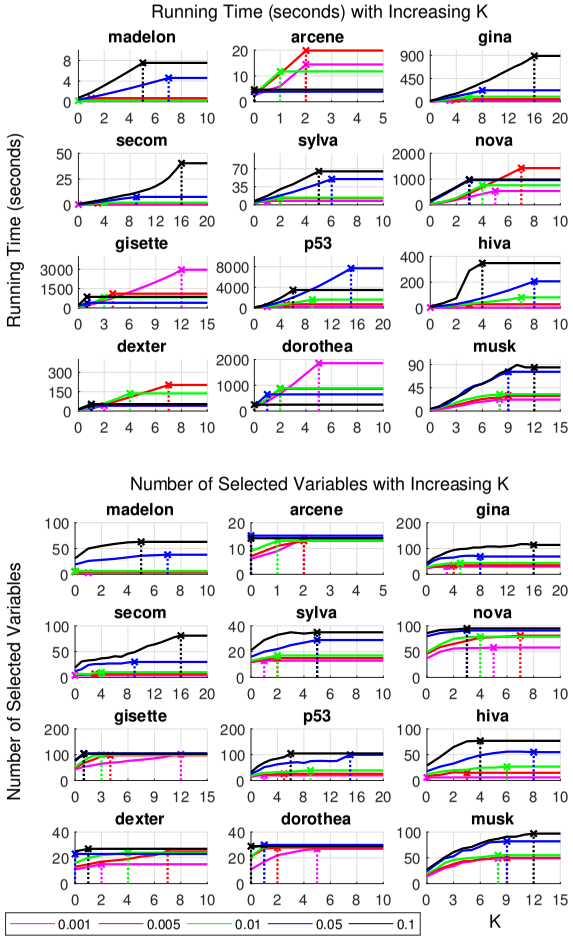


Figure 1: The figure shows how the running time (top) and the number of selected variables (bottom) vary with an increasing number of runs K for different values of the threshold parameter α. The vertical lines indicate the value of K for which FBED^K has converged. We can see that running time increases almost linearly with increasing K. Also, most progress is made in the first few runs, and any additional runs increase running time while only selecting a few more variables.

the complete datasets, while for the predictive performance we used the model selection and performance estimation protocols described previously and report averages over multiple repetitions of the experiment.

Figure 1 shows how the number of runs K affects the running time and the number of selected variables. Vertical lines show the value of K for which the algorithm has converged (that is, after that point more runs do not select any more variables). We can see that running time increases almost linearly with an increasing number of runs K, meaning that any additional run has a roughly linear computational cost with respect to the size of the dataset. Furthermore, convergence is typically achieved in less than 10 runs, although for a few cases up to 16 runs are required. As expected, the number of selected variables increases with K, as well as with the threshold α. In the majority of cases however, most progress is made in the first few runs, and further runs increase the number of selected variables only marginally. *Based on those results, we recommend considering relatively small values of K, up to $K < 10$.*

Figure 2 shows how the area under the ROC curve (AUC) varies with an increasing number of runs K, for 5 different values of the threshold parameter α of FBED^K. We observe that, although AUC often tends to increase with K, this is not always the case. For instance, for the nova and dexter datasets, AUC actually decreases with K for some values of α. The maximum AUC is typically achieved with relatively small values of K, further suggesting that considering higher values for K is not necessary. Also, there are no clear relationships between AUC, the value of α and the type of predictive models used. Depending on the dataset different values of α or predictive models may be optimal. *Thus, in practice we recommend considering several combinations of α and K.* Optimizing over both α and K will be considered in Section 4.4.

4.3. FBED^K vs FBS

In this section we compare FBED^K to the standard FBS algorithm in terms of predictive performance, number of selected variables and running time. The algorithms were compared on the same hyper-parameters (for example, FBS vs FBED^0 with $\alpha = 0.01$); results when also optimizing over hyper-parameters are shown in Section 4.4. Model selection and performance estimation was performed for each feature selection algorithm and each hyper-parameter value separately, following the procedure described in the experimental setup. *The main goal of this comparison is to show that FBED^K and FBS perform similarly for the same hyper-parameters, with the former being faster.* A summary of the results is presented next.

Figure 3 shows how the algorithms compare in terms of predictive performance, number of selected variables and running time. Each column shows the distribution of the respective metric across all thresholds and datasets, as well as the mean and median values. The difference in AUC is computed is AUC(FBED^K) - AUC(FBS), the relative number of selected variables is computed as the ratio of variables selected by FBED^K compared to FBS, and the speed-up is computed as Time(FBS) / Time(FBED^K). The y-axis corresponds to different values of K used by FBED^K. Only the first few values ($K \leq 9$), as well as the last one ($K = \infty$) are shown, as the left-out ones were almost identical to $K = \infty$.

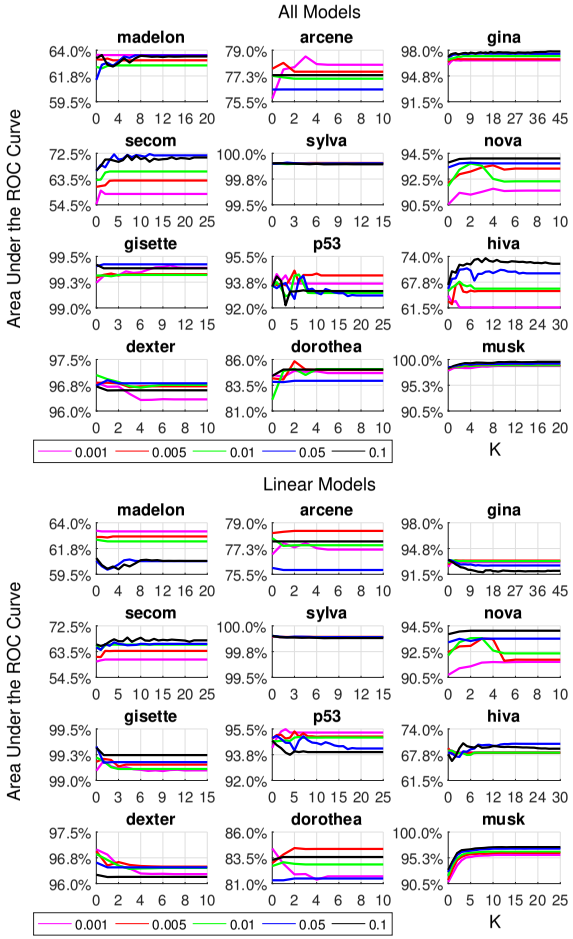


Figure 2: The figure shows how the AUC varies with an increasing number of runs K for different values of the threshold parameter α, using non-linear and linear models (top) or linear models only (bottom). There is no clear pattern for which thresholds or values of K to prefer, but the optimal values depend on the specific dataset, as well as on the predictive models used. However, in most cases only a few runs are required to achieve maximal AUC.

Regarding predictive performance, FBED^K performs as good as FBS on average, irrespective of the type of predictive models used. For FBED^K with $K < 3$, the average

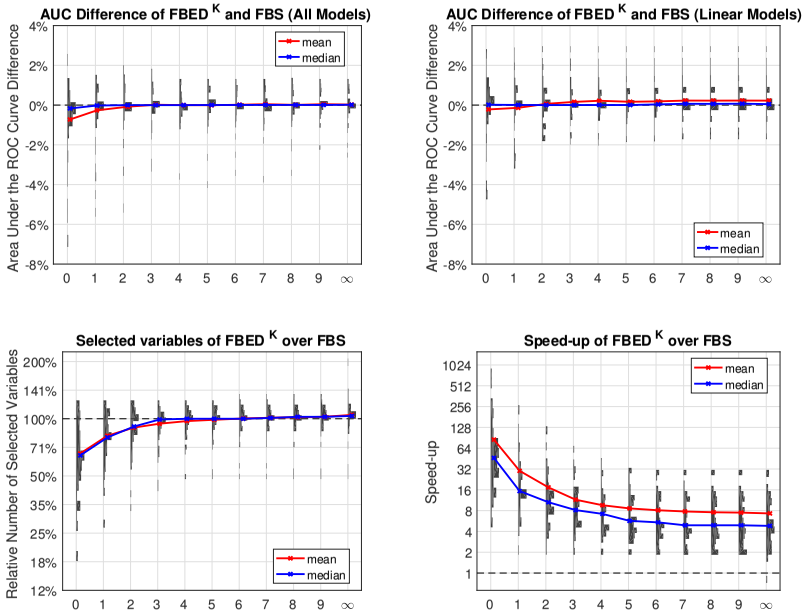


Figure 3: The x-axis of the figures on the top row shows the difference in AUC between FBED^K and FBS, with positive values indicating that FBED^K performs better than FBS. The AUC of the top left figure is computed by optimizing over all models, while for the one of the top right figure only linear models were considered. The relative number of selected variables (bottom left) shows the number of variables selected by FBED^K compared to the ones selected by FBS. The speed-up (bottom right) is computed as the one obtained by FBED^K over FBS. For all cases, the distribution over all thresholds and datasets is shown, as well as the mean and median values. The y-axis on all figures is the value of K used by FBED^K. Overall, FBED^K has a virtually identical performance with FBS, while being on average between 1 and 2 orders of magnitude faster.

difference in AUC is less than 1% while the median difference is close to 0, and for all other K the performance is almost identical to FBS. We note that all those lower-performing cases are also the ones where FBED^K selected much fewer variables than FBS. In terms of the number of selected variables, FBED^K produces smaller solutions for $K < 3$, and tends to select the same number as FBS with increasing K. Finally, in terms of running time, FBED^K is significantly faster than FBS, being about 1-2 orders of magnitude faster on average in all cases.

An interesting case is for FBED^3, where the number of selected variables and AUC between both algorithms is almost identical, while being around 10 times faster than FBS. If

speed and small solutions are important, FBED^0 and FBED^1 are good choices, as they are ∼30-100 times faster than FBS, selecting only ∼70%-80% of the variables with a minimal drop in AUC. Therefore, if the number of variables is high, FBED^0 and FBED^1 are preferable due to their low computational lost. Furthermore, in low sample settings the smaller solutions of FBED^0 and FBED^1 are important, as selecting many variables leads to loss of power and overfitting. If on the other hand the sample size is large and the number of variables is relatively small, both FBS and FBED^K with higher values of K are reasonable choices, with the latter being more attractive, as it is around 1 order of magnitude faster and thus can scale to higher variable sizes.

4.4. Comparison of FBED^K with other Feature Selection Methods

We performed an experiment where we also optimize over the hyper-parameter values of feature selection algorithms. *The main objective of this comparison is to compare FBED^K to other feature selection algorithms in a realistic scenario, where hyper-parameter values are optimized.* For this comparison we focus on the predictive performance and number of selected variables; additional results showing the running time of each algorithm can be found in Appendix D.

4.4.1. Setup

For FBED^K optimization is performed over the threshold α and the number of runs K. We examine four versions of FBED^K: FBED^0, FBED$^{\leq 1}$, FBED$^{\leq 3}$ and FBED$^{\leq \infty}$. FBED$^{\leq K}$ means that optimization was performed for all results up to K runs. Thus, the number of hyper-parameter configurations used were 5, 10, 20 and around 50 (for most cases) for FBED^0, FBED$^{\leq 1}$, FBED$^{\leq 3}$ and FBED$^{\leq \infty}$ respectively. The hyper-parameter values for FBS, MMPC and LASSO-FS are the ones described in Section 4.1.1 (a total of 100, 20 and 100 respectively). We also included results when no feature selection was performed (NO-FS). Finally, we remind the reader that we used two sets of classification algorithms and hyper-parameters, one containing only linear algorithms (elastic net regularized logistic regression) and one also containing non-linear ones in addition to the linear ones (Gaussian support vector machines and random forests). For brevity, we will refer to linear models as LM and to the combination of linear and non-linear models as NLM hereafter.

4.4.2. Results

A summary of the results averaged over repetitions, measuring the AUC and number of selected variables is shown in Tables 2 and 3. For each algorithm, we computed a score which is the average rank of that algorithm over all datasets. The final rank of an algorithm is then computed based on that score. We used a bootstrap-based procedure to compute the probability of an algorithm being significantly better or worse than all competitors, and used a threshold of 95%. The procedure is described in more detail in Appendix C. In the tables, algorithms that are statistically significantly better than all others are shown in bold, whereas algorithms that are worse than the rest are shown in italic.

Overall, performing no feature selection has the highest AUC. Out of all feature selection methods, LASSO-FS offers the best predictive performance, statistically significantly outperforming the rest in 4 datasets. MMPC outperforms the rest in 2 and 3 datasets using

Table 2: Area under the ROC curve and number of selected variables for all feature selection algorithms using linear and non-linear models. The results are obtained after optimizing the hyper-parameters of the feature selection and modeling algorithms. Bold and italic entries denote that the method is significantly better or worse than all other feature selection methods (excluding NO-FS) respectively. The score is the average rank of each method over all datasets and the final rank is computed using those scores. Methods that select more variables tend to also perform better (Spearman correlation between AUC and variable rankings is -0.976).

	Algorithm	musk	sylva	madelon	secom	gina	hiva	gisette	p53	arcene	nova	dexter	dorothea	Score	Rank
AUC (all models)	FBED^0	*98.5*	99.9	63.4	63.7	*97.0*	67.4	99.4	93.3	77.5	93.6	96.7	83.8	6.33	8
	FBED$^{\leq 1}$	98.7	99.9	63.3	63.0	97.3	70.9	99.4	93.5	76.9	94.0	96.8	83.5	6.17	7
	FBED$^{\leq 3}$	99.2	99.9	63.0	68.0	97.3	68.8	99.4	93.2	77.3	93.6	96.8	84.3	5.46	6
	FBED$^{\leq \infty}$	99.5	99.9	63.6	67.6	97.6	72.3	99.4	93.5	77.3	93.6	96.8	84.9	4.71	4
	FBS	99.5	99.9	64.8	69.5	97.5	69.1	99.4	94.1	77.2	92.4	*95.9*	84.3	5.00	5
	MMPC	98.9	99.9	65.1	63.4	98.1	68.3	99.6	93.1	79.6	**96.7**	97.3	**91.7**	4.00	3
	LASSO-FS	**99.9**	99.9	**82.3**	69.8	98.2	74.2	**99.7**	94.2	**82.4**	96.1	97.2	89.0	2.58	2
	NO FS	99.9	99.9	82.8	71.9	98.3	74.3	99.6	94.7	90.0	96.6	98.2	94.6	1.75	1
Selected Variables	FBED^0	**22.1**	13.4	8.4	15.0	**32.9**	**18.8**	72.9	24.2	8.7	66.5	17.4	21.6	1.33	1
	FBED$^{\leq 1}$	35.6	15.2	8.2	18.6	47.7	34.1	80.2	23.5	9.6	71.9	18.0	23.2	2.79	2
	FBED$^{\leq 3}$	47.1	21.1	14.0	24.4	56.5	43.5	80.2	26.4	9.3	72.1	18.5	22.3	3.63	4
	FBED$^{\leq \infty}$	77.6	25.5	14.9	41.1	105.8	75.1	79.4	33.7	9.3	72.3	18.7	22.4	4.79	5
	FBS	76.1	21.1	19.6	28.4	78.7	33.8	**65.0**	32.6	11.0	71.6	16.9	22.0	3.46	3
	MMPC	42.5	20.7	17.0	16.2	155.4	51.7	*388.2*	68.8	*43.9*	*879.7*	*187.2*	*445.1*	5.42	6
	LASSO-FS	*138.2*	*43.0*	*495.4*	*133.3*	*406.4*	*306.6*	187.5	*161.9*	29.6	349.8	97.9	70.1	6.58	7
	NO FS	166.0	213.0	500.0	468.0	970.0	1617.0	4948.0	5408.0	9955.0	11853.0	9988.0	88215.0	8.00	8

Table 3: Area under the ROC curve and number of selected variables for all feature selection algorithms using linear models. The results are obtained after optimizing the hyper-parameters of the feature selection and modeling algorithms. Bold and italic entries denote that the method is significantly better or worse than all other feature selection methods (excluding NO-FS) respectively. The score is the average rank of each method over all datasets and the final rank is computed using those scores. Methods that select more variables tend to also perform better (Spearman correlation between AUC and variable rankings is -0.595), but the effect is not as strong as the one of the previous results (Table 2).

	Algorithm	musk	sylva	madelon	secom	gina	hiva	gisette	p53	arcene	nova	dexter	dorothea	Score	Rank
AUC (linear models)	FBED^0	*93.0*	99.9	62.3	65.1	93.3	69.4	99.3	94.0	78.0	93.3	96.6	81.7	4.58	4
	FBED$^{\leq 1}$	94.6	99.9	62.2	62.6	93.3	68.3	99.3	94.4	78.0	93.7	96.6	83.5	4.83	5
	FBED$^{\leq 3}$	96.5	99.9	62.0	64.0	93.2	68.3	99.2	95.0	77.7	93.4	96.5	83.5	5.38	6
	FBED$^{\leq \infty}$	97.1	99.9	62.0	63.8	92.1	69.0	99.2	95.1	77.7	93.4	96.5	83.3	5.71	7
	FBS	97.1	99.9	62.2	67.0	92.5	67.3	99.2	94.6	75.5	92.5	*95.6*	83.3	5.92	8
	MMPC	93.8	**99.9**	62.3	64.3	93.4	68.3	99.3	95.3	75.6	**96.7**	96.6	**89.3**	3.50	3
	LASSO-FS	**97.5**	99.9	62.1	64.8	92.1	70.7	**99.6**	95.4	**80.6**	95.8	**97.3**	86.9	3.17	2
	NO FS	97.6	99.9	59.5	66.6	91.2	71.8	99.6	95.8	87.3	96.5	98.3	91.7	2.92	1
Selected Variables	FBED^0	**22.7**	**14.1**	8.0	15.6	**34.2**	**20.4**	71.5	23.2	**8.6**	**66.5**	17.5	21.0	1.21	1
	FBED$^{\leq 1}$	35.2	16.8	10.6	18.0	46.9	31.2	79.3	25.5	9.4	71.4	17.9	22.0	2.71	2
	FBED$^{\leq 3}$	51.4	23.3	11.6	26.4	51.9	40.3	83.9	32.8	9.7	71.6	18.3	21.2	3.88	4
	FBED$^{\leq \infty}$	86.7	25.3	11.6	47.0	114.1	70.2	84.0	33.8	9.7	71.6	18.3	21.3	4.71	5
	FBS	79.5	22.8	9.0	34.7	80.1	29.3	**65.0**	34.6	11.0	72.4	16.7	22.0	3.63	3
	MMPC	43.3	34.2	8.0	17.7	137.8	36.1	*389.0*	92.2	*43.0*	*879.7*	*193.4*	*363.2*	5.29	6
	LASSO-FS	*144.5*	*57.8*	47.2	93.5	*369.3*	*285.5*	206.5	*208.0*	25.7	307.8	87.0	74.6	6.58	7
	NO FS	166.0	213.0	500.0	468.0	970.0	1617.0	4948.0	5408.0	9955.0	11853.0	9988.0	88215.0	8.00	8

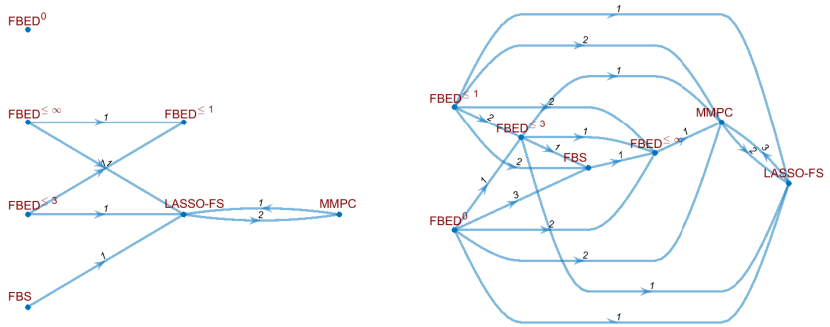
NLM and LM respectively. In two cases FBED^0 is significantly worse than the rest (musk and gina), while FBS is the worst in 1 dataset. However, in terms of the number of selected

variables, LASSO-FS selects statistically significantly more in 7 and 5 cases for NLM and LM, while MMPC selects the most variables in 5 datasets for both NLM and LM. FBED^K and FBS on the other hand tend to select fewer variables. Thus, *there is a clear trade-off between model interpretability (number of selected variables) and predictive performance (AUC)*. Specifically, there is a -0.976 and -0.595 Spearman correlation between the AUC and selected variables ranks for NLM and LM respectively. For that reason, we performed an additional experiment, comparing the AUC between FBED^K, FBS and LASSO-FS by constraining the total number of variables to select, presented in Section 4.5.

A strong outlier in the NLM case is the difference in performance of LASSO-FS compared to the other methods on the madelon dataset, where LASSO-FS reaches an AUC that is $17.3 - 19.5\%$ higher, which is also close to the AUC of NO-FS. The main reason why all methods fail is because the madelon dataset has been constructed in a way that makes it hard for linear methods (FBED^K, FBS and MMPC using the logistic regression test). Specifically, the outcome variable has been artificially constructed to be a XOR-type problem of 5 variables (Guyon et al., 2006b). Further evidence for the hardness of this problem is the fact that using LM and NO-FS achieves an AUC of only 59.5 (even lower than all feature selection methods). LASSO-FS, although also linear, is able to pick up the signal as it basically performs no feature selection, selecting 495.4 out of 500 variables on average. This happens because LASSO-FS explores up to 100 values for λ, some of which correspond to very dense solutions. In contrast, due to the experimental setup, none of the remaining methods selects that many variables in this case.

An interesting observation is the fact that FBED^0 and FBED$^{\leq 1}$, the forward selection methods selecting the fewest variables, are ranked higher in terms of AUC than FBED$^{\leq 3}$, FBED$^{\leq \infty}$ and FBS when using LM. Thus, they are better suited to pick out linear trends in the data, producing solutions that are smaller and more predictive compared to algorithms of the same type. Using NLM gives an even bigger advantage to methods selecting more variables, as this increases the chance to also capture some non-linear signals in the data.

Finally, we performed a statistical test between all pairs of methods to identify cases where a method outperforms others, both in terms of AUC and number of selected variables. Again, we used a bootstrap-based test, computing the joint probability that method A has a higher AUC and selecting fewer variables than method B, using the same procedure as described in Appendix C. A method is considered to dominate another, if that probability is higher than 95%. The results are summarized in Figure 4. Each node corresponds to a feature selection method, and a directed edge from method A to B with weight w denotes that A dominates B in w datasets. We observe that, except for a single case where FBED$^{\leq \infty}$ gets dominated by FBS in 1 dataset for LM, FBED^K is never dominated by any other method, neither for the NLM nor for the LM case. In both NLM and LM cases, FBED^K dominates the competitors in 1-3 cases, while LASSO-FS and MMPC only dominate each other in 1-3 cases. Especially interesting is the fact that for NLM, FBED$^{\leq 3}$, FBED$^{\leq \infty}$ and FBS (that is, the forward-selection algorithms typically selecting the most variables) are the only algorithms that dominate others, while also not getting dominated. On the other hand, using LM only FBED^0 and FBED$^{\leq 1}$ both dominate others and are not getting dominated. This agrees with the observation made before, that FBED^0 and FBED$^{\leq 1}$ are particularly well suited to identify compact and linearly predictive solutions.



(a) Dominating Relationships (All Models) (b) Dominating Relationships (Linear Models)

Figure 4: The figures show how often a feature selection method dominates another (that is, has a higher AUC while selecting fewer variables), using non-linear models (left) and linear models (right). An edge from method A to B with weight w indicates that A dominates B in w datasets. Except for FBED$^{\leq \infty}$ for linear models, which gets dominated by FBS in 1 dataset, methods in the FBED^K family are never dominated by FBS, MMPC or LASSO-FS, while typically dominating them in 1-3 datasets.

Overall, *there is no clear winner, and the choice depends solely on the goal*. If the goal is predictive performance, LASSO-FS or MMPC are clearly preferable. If on the other hand one is interested in interpretability, then methods from the FBED^K family with small values of K are preferable. Regarding FBS, there is no scenario where it is preferable over one of the other algorithms. We must note that those results are somewhat artificial, as the performance of FBED^K and FBS highly depends on the hyper-parameter values chosen for the experiment, while LASSO-FS is not as sensitive to those choices. Furthermore, the fact that hyper-parameters are optimized based on performance naturally tends to favor methods that select more variables, putting LASSO-FS at a disadvantage in terms of interpretability.

4.5. Fixing the Number of Selected Variables

As confirmed by the previous experiment, there are two main trade-offs for feature selection algorithms: (a) the number of selected variables, with fewer variables leading to more interpretable results, and (b) the predictive ability of the selected variables, with more variables typically leading to better results. In general, algorithms that select more variables also tend to perform better in terms of predictive performance. Because of that, *we performed a comparison where algorithms are forced the select the same number of variables. That way, the predictive performance of algorithms can be compared on equal footing.*

We will compare FBED^K and LASSO-FS, when both are limited to select the same number of features. FBS is not included in the comparison, as we have already shown in

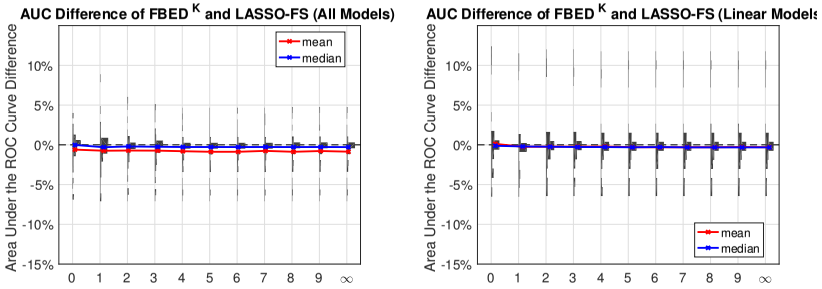


Figure 5: **LASSO-FS with limit on selected variables:** The x-axis shows the distribution of the difference in AUC of FBED^K and LASSO-FS, with positive values indicating that FBED^K performs better. The y-axis corresponds to value of K used by FBED^K. The mean and median values are shown in red and blue respectively. The average difference using non-linear models is 0.78%, and 0.23% when using only linear models. The difference can be explained by the arcene dataset, where LASSO-FS outperforms FBED^K even when selecting the same number of variables. A more detailed explanation is given in the main text.

Section 4.3 that FBS and FBED^K exhibit similar predictive performance with a similar number of selected features, with FBED^K being orders of magnitude faster. MMPC was not included, because neither MMPC nor FBED^K allow to set the number of features to select.

In order to perform the comparison, we executed FBED^K for multiple hyper-parameter values (the ones given in the experimental setup) and then executed LASSO-FS with the constraint to select the same number of variables as FBED^K did. As it was not always possible to select the exact same number of variables, we identified the solution of LASSO-FS with at least as many variables as FBED^K. Except for a few cases where LASSO-FS selected 1 more variable than FBED^K, both methods selected the same number of variables. As a final comment, we note that the above experiment does not favor FBED^K over LASSO-FS, as no optimization over its hyper-parameter values is performed. The reason we did not use a fixed number of variables M to select is because there is no easy way to select exactly M variables using FBED^K.

The comparison was performed similarly to the one between FBED^K and FBS in Section 4.3. Specifically, for a given K and threshold α, we computed the difference in AUC obtained by FBED^K and LASSO-FS. Thus, the only hyper-parameter optimization performed was over predictive models and not over the hyper-parameters of FBED^K. The results are shown in Figure 5. The x-axis corresponds to the difference in AUC between FBED^K and LASSO-FS, while the y-axis indicates the value K of FBED^K. We can see that, overall, both methods perform very similarly, regardless of whether linear models or non-linear models were used. For non-linear models, LASSO-FS outperforms FBED^K on average (over all K and α) by 0.78%, while using linear models the difference drops to

0.23%. In terms of median difference in AUC, a metric which is more stable with the respect to the datasets and hyper-parameter values used, both algorithms perform almost identically. Those results also agree with previous comparisons between forward selection and LASSO-FS (Hastie et al., 2017).

We investigated the difference in performance and found that it can be attributed to the arcene dataset. By removing this dataset, LASSO-FS outperforms FBED^K by 0.32% on average using non-linear models, while for linear models FBED^K performs better by 0.22%. Note that, arcene is the dataset which contains the fewest number of samples, while also containing a large number of variables. It contains 200 samples and 10000 variables, and only 120/160 are used for training and validation respectively. Theoretical results by Ng (2004) show that LASSO performs well in settings with low sample size and many irrelevant variables, as is the case for the arcene dataset. One possible explanation for the lower performance of FBED^K on arcene is that forward selection based procedures use more effective degrees of freedom (Hastie et al., 2017, Figure 1), thus requiring more sample size to have sufficient statistical power to pick up weak signals. It would be interesting to study this effect in more depth, but it is out of the scope of the current paper. In summary, *LASSO-FS and FBED^K perform similarly when the number of variables to select is the same.*

4.6. Simulation Study on the Multiple Testing Problem

The idea of early dropping of variables used by FBED^K does not only reduce the running time, but also reduces the problem of multiple testing, in some sense. Specifically, it reduces the number of variables falsely selected due to type I errors. In general, the number of false selections is related to the total number of variables considered in all forward iterations. Thus, the effect highly depends on the value of K used by FBED^K, with higher values of K leading to more false selections. We show this for FBED^0 by considering a simple scenario, where none of the candidate variables are predictive for the outcome. Then, in the worst case, FBED^0 will select about $\alpha \cdot p$ of the variables on average (where α is the significance level), since all other variables will be dropped in the first iteration. This stems from the fact that, under the null hypothesis of conditional independence, the p-values are uniformly distributed. In practice, the number of selected variables will be even lower, as FBED^0 will keep dropping variables after each variable inclusion. On the other hand, FBS may select a much larger number of variables, since each variable is given the chance to be included in the output at each iteration and will often do so, simply by chance.

We performed a small simulation to investigate the behavior of FBED^0, FBED^1, FBED$^\infty$ and how they compare to FBS. We generated 500 normally distributed datasets with 1000 samples each, a uniformly distributed random binary outcome, and considered different variable sizes $p \in \{100, 200, 300, 400, 500\}$ and 5 significance levels α uniformly spaced in $[0.01, 0.1]$. All variables are generated randomly, and there is no dependency between any of them. Thus, a false positive rate of about α is expected, if no adjustment is done to control the false discovery rate. For each setting, we computed the ratio of false positives with respect to the expected number of false positives.

Figure 6 (top) shows how the ratio varies for sample sizes 100 and 500 with increasing α, and Figure 6 (bottom) shows how the ratio varies for α 0.01 and 0.1 with increasing number

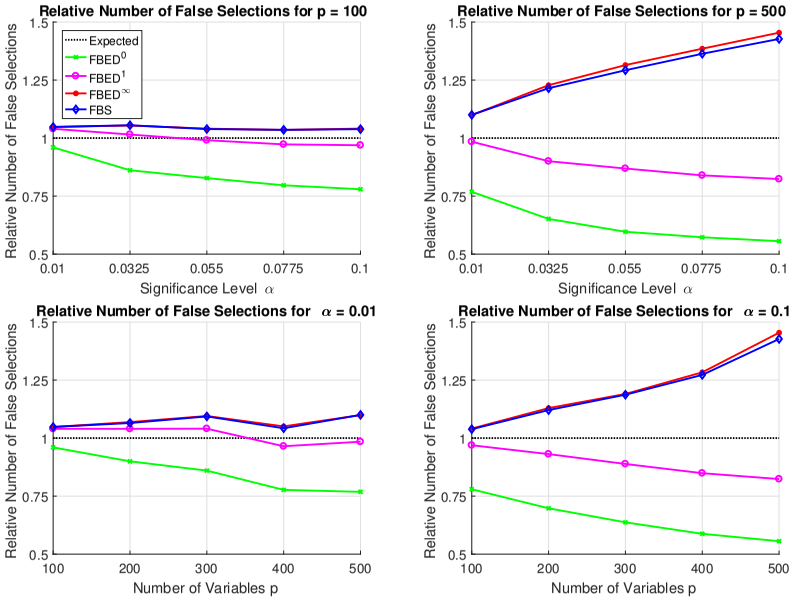


Figure 6: The figures show the relative number of false selections by each algorithm on randomly generated data. The expected number of false selections is $\alpha \cdot p$, where α is the significance level and p the number of variables. The numbers are computed as the ratio between the average number of selected variables to the expected false positives. FBED^0 and FBED^1 typically select fewer variables than expected, and their behavior improves with increasing α and p. FBED$^\infty$ and FBS on the other hand select more false positive variables, getting worse with larger values of α or on datasets with more variables.

of variables. In all cases, FBED^0 and FBED^1 select fewer false positives than expected, and their behavior improves both with increasing α and number of variables. FBED$^\infty$ and FBS perform almost identically, and tend to select more variables. We also observe that the number of false positives increases both with α and with the number of variables. Thus, *in case one is interested to limit the number of false selection, we recommend running FBED^K with a small value of K.*

5. Conclusion

We presented the early dropping heuristic to speed-up the forward-backward feature selection algorithm, which gives rise to a family of algorithms, called forward-backward selection with early dropping (FBED^K). Early dropping is a simple heuristic that leads to orders of magnitude speed-up, especially in high-dimensional datasets, while still maintaining the

theoretical guarantees of forward-backward selection. We prove that FBED^1 and FBED$^\infty$ identify the optimal solution (Markov blanket) if the distribution of the data can be faithfully represented by a Bayesian network or maximal ancestral graph respectively, similar to the standard forward-backward selection (FBS).

A useful property of FBED^K is that it is a general algorithm that can be adapted to handle different variable types (for example, continuous, categorical, ordinal), cross-sectional and time-course data, linear and non-linear dependencies, as well as different analysis tasks (for example, regression, classification, survival analysis) by using an appropriate conditional independence test. In contrast, algorithms like LASSO (Tibshirani, 1996), although being computationally fast and performing well in terms of predictive performance for common problems like regression and classification, are not as general (Meier et al., 2008; Schelldorfer et al., 2011; Ivanoff et al., 2016) and are computationally demanding for some problems (Fan et al., 2010; Groll and Tutz, 2014; Tsagris et al., 2018).

In experiments we demonstrate that FBED^K behaves similarly to FBS in terms of predictive performance and number of selected variables, while being 1-2 orders of magnitude faster. Compared to other feature selection algorithms like LASSO (Tibshirani, 1996) and MMPC (Tsamardinos et al., 2003a), FBED^K has competitive predictive performance, while selecting the fewest variables, which is especially important if feature selection is performed for knowledge discovery. An interesting result is that FBED^K and LASSO perform about equally well, when limited to select the same number of variables. This, combined with the fact that FBED^K is more general, makes it an attractive alternative to LASSO, especially for problems where no efficient solution to the LASSO problem exists.

Acknowledgments

We would like to thank Vincenzo Lagani and Michalis Tsagris for their helpful comments. This work was funded by the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 617393.

Appendix A. Bayesian Networks and Maximal Ancestral Graphs

We will briefly introduce Bayesian networks and maximal ancestral graphs, which we will use to show theoretical properties of FBED^K.

A **directed acyclic graph** (DAG) is a graph that only contains directed edges ($\rightarrow$) and has no directed cycles. A **directed mixed graph** is a graph that, in addition to directed edges also contains bi-directed edges ($\leftrightarrow$). The graphs contain no self-loops, and vertices can be connected only by a single edge. Two vertices are called **adjacent** if they are connected by an edge. An edge between X and Y is called **into** Y if $X \rightarrow Y$ or $X \leftrightarrow Y$. A **path** in a graph is a sequence of unique vertices $\langle V_1, \ldots, V_k \rangle$ such that each consecutive pair of vertices is adjacent. The first and last vertices in a path are called **endpoints**. A path is called directed if $\forall 1 \leq i \leq k$, $V_i \rightarrow V_{i+1}$. If $X \rightarrow Y$ is in a graph, then X is a **parent** of Y and Y a **child** of X. A vertex W is a **spouse** of X, if both share a common child. A vertex X is an **ancestor** of Y, and Y is a **descendant** of X, if $X = Y$ or there is a directed path from X to Y. A triplet $\langle X, Y, Z \rangle$ is called a **collider** if Y is adjacent to X and Z, and both, X and Z are into Y. A triplet $\langle X, Y, Z \rangle$ is called **unshielded** if Y is

adjacent to X and Z, but X and Z are not adjacent. A path p is called a **collider path** if every non-endpoint vertex is a collider on p.

Bayesian networks (BNs) consist of a DAG $\mathcal{G}$ and a probability distribution $\mathcal{P}$ over a set of random variables $\mathbf{V}$. Each such variable is represent by a vertex in $\mathcal{G}$, and thus, the terms variable and vertex will be used interchangeably. The DAG represents dependency relations between variables in $\mathbf{V}$ and is linked with $\mathcal{P}$ through the **Markov condition**, which states that each variable is conditionally independent of its non-descendants given its parents. Those are not the only independencies encoded in the DAG; the Markov condition entails additional independencies, which can be read from the DAG using a graphical criterion called **d-separation** (Verma and Pearl, 1988; Pearl, 1988). In order to present the d-separation criterion we first introduce the notion of blocked paths. A (not necessarily directed) path p between two nodes X and Y is called **blocked** by a set of nodes $\mathbf{Z}$ if there is a node V on p that is a collider and, neither V nor any of its descendants are in $\mathbf{Z}$, or if V is not a collider and it is in $\mathbf{Z}$. If all paths between X and Y are blocked by $\mathbf{Z}$, then X and Y are **d-separated** given $\mathbf{Z}$; otherwise X and Y are **d-connected** given $\mathbf{Z}$. The **faithfulness condition** states that all and only those conditional independencies in $\mathcal{P}$ are entailed by the Markov condition applied to $\mathcal{G}$. In other words, the faithfulness condition requires that two variables X and Y are d-separated given a set of variables $\mathbf{Z}$ if and only if they are conditionally independent given $\mathbf{Z}$.

Bayesian networks are not closed under marginalization: a marginalized DAG, containing only a subset of the variables of the original DAG, may not be able to exactly represent the conditional independencies of the marginal distribution (Richardson and Spirtes, 2002). **Directed maximal ancestral graphs** (DMAGs) (Richardson and Spirtes, 2002) are an extension of BNs, which are able to represent such marginal distributions, that is, they admit the presence of latent confounders. The graphical structure of a DMAG is a directed mixed graph with the following restrictions: (i) it contains no directed cycles, (ii) it contains no almost directed cycles, that is, if $X \leftrightarrow Y$ then neither X nor Y is an ancestor of the other, and (iii) there is no primitive inducing path between any two non-adjacent vertices, that is, there is no path p such that each non-endpoint on p is a collider and every collider is an ancestor of an endpoint vertex of p. The d-separation criterion analogue for DMAGs is called the **m-separation criterion**, and follows the same definition.

A **Markov blanket** of a variable T is a **minimal** set of variables $\mathbf{MB}(T)$ that renders T conditionally independent of all remaining variables $\mathbf{V} \setminus \mathbf{MB}(T)$. In case faithfulness holds, and the distribution can be represented by a BN or DMAG, then the Markov blanket is **unique**. For a BN, the Markov blanket of T consists of its parents, children and spouses. For DMAGs it is slightly more complicated: the Markov blanket of T consists of its parents, children and spouses, as well as its district (all vertices that are reachable by bi-directed edges), the districts of its children and the parents of all districts (Richardson, 2003). An alternative definition is given next.

Definition 1 The Markov blanket of T in a BN or DMAG consists of all vertices adjacent to T, as well as all vertices that are reachable from T through a collider path.

A proof sketch follows. Recall that a collider path of length $k-1$ is of the form $X_1 * \rightarrow X_2 \dots X_{k-1} \leftarrow * X_k$, where the path between X_2 and X_{k-1} contains only bi-directed edges. Given this, it is easy to see that Definition 1 includes vertices directly adjacent to T, its spouses (collider path of length 2), and in the case of DMAGs, vertices D in the district of

T ($T \leftrightarrow \dots \leftrightarrow D$), vertices D in the district of any children C of T ($T \rightarrow C \leftrightarrow \dots \leftrightarrow D$), and all parents P of any vertex D in some district ($T* \rightarrow \dots \leftrightarrow D \leftarrow P$). As the previous cases capture exactly all possibilities of nodes reachable from T through a collider path, Definition 1 does not include any additional variables that are not in the Markov blanket of T.

Appendix B. Proofs

We proceed by listing some axioms about conditional independence (Pearl, 2000), called **semi-graphoid** axioms, which will be useful later on. Those axioms are general, as they hold for any probability distribution. For all of the proofs we assume that the algorithms have access to an **independence oracle** that can perfectly determine whether a given conditional dependence or independence holds. Furthermore, in all proofs we will use the terms d-connected/m-connected (d-separated/m-separated) and dependent (independent) interchangeably; this is possible due to the faithfulness assumption.

Symmetry	$(\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z}) \Rightarrow (\mathbf{Y} \perp \mathbf{X} \mid \mathbf{Z})$
Decomposition	$(\mathbf{X} \perp \mathbf{Y} \cup \mathbf{W} \mid \mathbf{Z}) \Rightarrow (\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z}) \wedge (\mathbf{X} \perp \mathbf{W} \mid \mathbf{Z})$
Weak Union	$(\mathbf{X} \perp \mathbf{Y} \cup \mathbf{W} \mid \mathbf{Z}) \Rightarrow (\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z} \cup \mathbf{W})$
Contraction	$(\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z}) \wedge (\mathbf{X} \perp \mathbf{W} \mid \mathbf{Y} \cup \mathbf{Z}) \Rightarrow (\mathbf{X} \perp \mathbf{Y} \cup \mathbf{W} \mid \mathbf{Z})$

Using those axioms we prove the following lemma.

Lemma 7 *Let A, T be variables and $\mathbf{B}$, $\mathbf{C}$ sets of variables. Then $(T \perp A \mid \boldsymbol{B} \cup \mathbf{C}) \wedge (T \perp \boldsymbol{B} \mid \mathbf{C}) \Rightarrow (T \perp A \mid \mathbf{C})$ holds for any such variables.*

Proof

$(T \perp A \mid \mathbf{B} \cup \mathbf{C}) \wedge (T \perp \mathbf{B} \mid \mathbf{C}) \Rightarrow$ (**Contraction**)
$(T \perp A \cup \mathbf{B} \mid \mathbf{C}) \Rightarrow$ (**Decomposition**)
$(T \perp A \mid \mathbf{C}) \wedge (T \perp \mathbf{B} \mid \mathbf{C})$

■

The following lemma will be useful for proving some of the theorems.

Lemma 8 *Let $\mathbf{S}$ be a set of variables selected for some target T and $\mathbf{R} = \mathbf{V} \setminus \mathbf{S}$. Assume that $\forall V_r \in \mathbf{R}$ $(T \perp V_r \mid \mathbf{S})$ holds. Then, if $\exists V_s \in \mathbf{S}$ such that $(T \perp V_s \mid \mathbf{S} \setminus V_s)$ holds, $\forall V_r \in \mathbf{R}$ $(T \perp V_r \mid \mathbf{S} \setminus V_s)$ also holds.*

Proof We are given that $\forall V_r \in \mathbf{R}$ $(T \perp V_r \mid \mathbf{S})$ holds. By applying Lemma 7 to each variable in $V_r \in \mathbf{R}$ with $A = V_r$, $\mathbf{B} = \{V_s\}$ and $\mathbf{C} = \mathbf{S} \setminus V_s$, we get that $(T \perp V_r \mid V_s \cup (\mathbf{S} \setminus V_s)) \wedge (T \perp V_s \mid \mathbf{S} \setminus V_s) \Rightarrow (T \perp V_r \mid \mathbf{S} \setminus V_s)$ holds for any such V_r, which concludes the proof. ■

To put it simple, Lemma 8 states that if we remove any variable V_s from a set of selected variables $\mathbf{S}$ by conditioning on $\mathbf{S} \setminus V_s$, no variable that is not in $\mathbf{S}$ becomes conditionally dependent with T given $\mathbf{S} \setminus V_s$. In practice this means that removing variables using backward selection from a set of variables selected by forward selection will not create any additional conditional dependencies, meaning that we do not have to reconsider them again.

Proof of Corollary 3
Proof To show that $\mathbf{S}$ is minimal, we have to show the following

i $\forall V_s \in \mathbf{S}$ $(T \not\perp V_s \mid \mathbf{S} \setminus V_s)$ (*No variable can be removed*)

ii $\forall V_r \in \mathbf{V_D} \setminus \mathbf{S}, (T \perp V_r \mid \mathbf{S})$ (*No variable can be added*)

Proof of (i): This holds trivially, as backward selection removes any variable $V_s \in \mathbf{S}$ if $(T \perp V_s \mid \mathbf{S} \setminus V_s)$ holds.
Proof of (ii): We know that after the termination of forward selection, no variable can be added, that is, $\forall V_r \in \mathbf{R}$ $(T \perp V_r \mid \mathbf{S})$ holds. Given that, Lemma 8 can be repeatedly applied after each variable removal by backward selection, and thus no variable in $\mathbf{R}$ can be added to $\mathbf{S}$. ■

Proof of Corollary 4
Proof As is the case with FBS, the forward selection phase of FBED$^\infty$ stops if no more variables can be included. Using this fact, the proof is identical to the one of Theorem 3. ■

Proof of Theorem 5
Proof In the first run of FBED^1, all variables that are adjacent to T (that is, its parents and children) will be selected, as none of them can be d-separated from T by any set of variables. In the next run, all variables connected through a collider path of length 2 (that is, the spouses of T) will become d-connected with T, since the algorithm conditions on all selected variables (including its children), and thus will be selected. The resulting set of variables includes the Markov blanket of T, but may also include additional variables. Next we show that all additional variables will be removed by the backward selection phase. Let MB(T) be the Markov blanket of T and $\mathbf{S_{ind}} = \mathbf{S} \setminus$MB($T$) be all selected variables not in the Markov blanket of T. By definition, $(T \perp \mathbf{X} \mid MB(T))$ holds for any set of variables $\mathbf{X}$ not in MB(T), and thus also for variables $\mathbf{S_{ind}}$. By applying the weak union graphoid axiom, one can infer that $\forall S_i \in \mathbf{S_{ind}}, (T \perp S_i \mid MB(T) \cup \mathbf{S_{ind}} \setminus S_i)$ holds, and thus some variable S_j will be removed in the first iteration. Using the same reasoning and the definition of a Markov blanket, it can be shown that all variables in $\mathbf{S_{ind}}$ will be removed from MB(T) at some iteration. To conclude, it suffices to use the fact that variables in MB(T) will not be removed by the backward selection, as they are not conditionally independent of T given the remaining variables in MB(T). ■

Proof of Theorem 6
Proof In the first run of FBED$^\infty$, all variables that are adjacent to T (that is, its parents, children and variables connected with T by a bi-directed edge) will be selected, as none

of them can be m-separated from T by any set of variables. After each run additional variables may become admissible for selection. Specifically, after k runs all variables that are connected with T by a collider path of length k will become m-connected with T, and thus will be selected; we prove this next. Assume that after k runs all variables connected with T by a collider path of length at most $k-1$ have been selected. By conditioning on all selected variables, all variables that are into some selected variable connected with T by a collider path will become m-connected with T. This is true because conditioning on a variable Y in a collider $\langle X, Y, Z \rangle$ m-connects X and Z. By applying this on each variable on some collider path, it is easy to see that its end-points become m-connected. Finally, after applying the backward selection phase, all variables that are not in the Markov blanket of T will be removed; the proof is identical to the one used in the proof of Theorem 5 and thus will be omitted. ■

Appendix C. Bootstrap Test For Comparing Algorithms

We used bootstrapping to compute the probability that algorithm A_i is better/worse or equal than all others in terms of some measure of interest f (for example, AUC), that is $P(\wedge_{j \neq i} f(A_i) \geq f(A_j))$. The procedure is described next. Let $f_{i,k}$ denote the measure of interest of algorithm i on test set k. We resample with replacement $B = 100000$ times the test sets and compute f, denoted as $f_{i,k,b}$ for the b-th sample of algorithm i and test set k. Then, $f_{i,k,b}$ are averaged over test sets, obtaining $\hat{f}_{i,b}$. The probability $P(\wedge_{j \neq i} f(A_i) \geq f(A_j))$ is then computed as $1/B \sum_b I(\hat{f}_{i,b} \geq \max_j \hat{f}_{j,b})$, where I is the indicator function.

Appendix D. Running Times of FBED^K, FBS, MMPC and LASSO

Table 4 shows the running time of each feature selection algorithm and configuration, on all datasets. The values correspond to a single run on the complete dataset. All runs were performed on a single machine, and no runs were performed simultaneously. For FBED^K we only show running times for $K \in \{0, 1, 3, \infty\}$, and for LASSO-FS we show results for $\lambda_{max} \in \{25, 100, 500\}$. MMPC for a given value of $maxK$ is denoted as MMPC^{maxK}.

It can clearly be seen that LASSO-FS is the fastest in large datasets, irrespective of the number of λ values used. For smaller datasets (musk, sylva, madelon, secom, gina and hiva), FBED^0 and FBED^1 are often at least as fast as LASSO-FS. FBS and MMPC with $maxK \geq 3$ are the slowest among all algorithms. For the gisette and nova datasets, MMPC with $maxK \geq 3$ fails to terminate after a timeout limit of 2 days. For $maxK \leq 2$ MMPC often has competitive performance with the other algorithms. We note that MMPC was designed specifically for low sample sizes and high dimensional data, such as data from biological domains which contain a few tens or hundreds of samples and tens of thousands of variables, explaining the high running times on the datasets considered in our experiments, most of which contain thousands of samples.

The large difference between the running time of LASSO-FS and the other algorithms can largely be attributed to their implementations. For LASSO-FS the glmnet implementation was used, which is highly optimized and written in FORTRAN. In contrast, for FBED^K, FBS and MMPC we used a custom logistic regression implementation written in Matlab.

A difference of 1-2 orders of magnitude can be expected between the same implementation in a low-level language such as FORTRAN, C or C++ and higher-level languages such as Matlab. Therefore, we would expect that an implementation in a lower-level language would perform similarly to LASSO-FS. Of course, LASSO-FS has the advantage that it returns the whole solution path, and thus would still be faster in practice if hyper-parameter optimization is also performed.

Table 4: Running times in seconds on the full datasets.

	Algorithm	musk	sylva	madelon	secom	gina	hiva	gisette	p53	arcene	nova	dexter	dorothea
$\alpha = 0.001$	FBED^0	1.5	3.3	0.3	0.3	5.0	2.8	56.7	48.6	2.6	21.8	6.2	133.9
	FBED^1	3.2	7.6	0.7	0.3	11.3	2.8	186.5	137.8	6.0	90.6	20.1	377.9
	FBED^3	9.1	7.6	0.7	0.3	24.8	2.8	546.3	242.3	14.5	308.0	42.8	1033.0
	FBED$^\infty$	22.1	7.6	0.7	0.3	24.8	2.8	2962.7	242.3	14.5	536.1	42.8	1855.3
	FBS	46.0	36.3	1.3	1.4	140.4	22.4	5773.0	1186.4	66.8	3486.3	309.2	7233.3
	MMPC^1	3.9	9.5	0.3	0.5	21.2	5.0	309.2	115.0	4.1	112.9	14.9	91.8
	MMPC^2	7.9	17.9	0.3	0.5	84.0	5.4	1885.4	163.4	4.1	1524.9	17.9	92.0
	MMPC^3	7.9	20.9	0.3	0.6	171.2	5.6	N/A	175.9	4.1	N/A	19.9	92.0
	MMPC^4	9.8	21.8	0.3	0.5	292.4	5.6	N/A	210.7	4.1	N/A	21.3	93.1
$\alpha = 0.005$	FBED^0	1.8	3.9	0.2	0.3	6.7	3.3	79.2	60.4	2.8	31.3	6.5	139.9
	FBED^1	5.4	8.3	0.6	0.8	14.2	10.7	244.2	158.5	10.8	114.1	22.3	457.0
	FBED^3	11.7	12.9	0.6	1.6	36.2	28.1	756.1	418.0	19.9	435.7	77.9	865.6
	FBED$^\infty$	29.7	12.9	0.6	1.6	49.2	28.1	1105.3	700.5	19.9	1426.2	202.2	865.6
	FBS	82.2	50.4	2.1	3.1	192.7	82.7	10932.7	3864.6	66.8	7015.8	503.0	7233.3
	MMPC^1	4.6	10.4	0.3	0.6	25.8	6.0	384.6	142.2	4.7	238.3	17.4	106.3
	MMPC^2	10.5	24.5	0.3	0.6	126.0	6.4	2869.2	230.4	4.7	3993.7	30.2	107.1
	MMPC^3	10.2	43.6	0.3	0.6	307.1	6.9	N/A	330.3	4.7	N/A	36.6	115.0
	MMPC^4	11.5	45.0	0.3	0.6	470.4	6.9	N/A	469.6	4.7	N/A	54.2	126.6
$\alpha = 0.01$	FBED^0	2.0	4.3	0.2	0.3	8.3	3.6	88.0	64.2	3.0	39.0	6.8	144.7
	FBED^1	5.3	8.9	0.2	0.8	16.8	11.9	261.1	164.3	11.8	123.3	25.2	463.0
	FBED^3	16.5	14.1	0.2	1.7	46.0	32.3	824.3	450.7	11.8	505.8	95.4	884.6
	FBED$^\infty$	32.5	14.1	0.2	2.1	97.6	81.8	824.3	1615.0	11.8	760.0	136.7	884.6
	FBS	94.3	61.0	2.5	4.4	234.8	150.2	10932.7	5160.2	66.8	8811.8	503.0	7233.3
	MMPC^1	5.0	12.8	0.3	0.6	28.0	7.0	426.9	161.3	5.1	347.0	19.6	119.1
	MMPC^2	12.4	28.7	0.3	0.6	141.0	7.8	3570.4	269.8	5.3	8262.0	41.6	128.8
	MMPC^3	12.7	45.1	0.4	0.6	426.0	9.1	N/A	460.1	5.3	N/A	63.5	146.0
	MMPC^4	13.3	62.0	0.4	0.7	573.4	9.8	N/A	728.7	5.3	N/A	104.6	200.6
$\alpha = 0.05$	FBED^0	3.3	5.6	0.4	0.5	14.2	6.4	140.2	95.2	3.9	111.9	9.5	201.0
	FBED^1	6.4	11.4	0.9	1.1	34.9	17.0	398.6	242.8	3.9	380.7	42.0	644.6
	FBED^3	25.6	24.8	2.0	2.7	82.8	50.5	398.6	779.5	3.9	957.2	42.0	644.6
	FBED$^\infty$	76.1	49.9	4.6	7.7	221.6	206.5	398.6	7688.4	3.9	957.2	42.0	644.6
	FBS	191.9	108.6	18.6	15.4	1243.4	3490.9	10932.7	31681.6	66.8	10119.3	621.0	7233.3
	MMPC^1	6.4	15.0	0.6	0.8	38.3	16.6	604.2	266.6	7.9	1050.6	48.9	406.0
	MMPC^2	15.8	48.1	1.5	1.1	222.2	22.4	6279.9	566.7	10.2	45093.2	204.1	750.2
	MMPC^3	22.0	152.2	3.3	1.1	921.7	46.3	N/A	977.6	12.7	N/A	525.3	1499.5
	MMPC^4	23.3	209.4	5.6	1.3	1499.4	68.8	N/A	2103.9	12.2	N/A	624.8	3818.6
$\alpha = 0.1$	FBED^0	4.1	7.6	0.8	0.7	20.1	11.0	213.0	128.4	4.7	162.3	14.1	246.7
	FBED^1	8.9	17.3	1.8	1.9	49.0	37.2	858.6	359.8	4.7	428.5	53.7	246.7
	FBED^3	26.8	40.0	4.5	4.0	127.5	289.0	858.6	1280.7	4.7	980.2	53.7	246.7
	FBED$^\infty$	84.6	64.9	7.5	40.3	889.1	347.7	858.6	3476.2	4.7	980.2	53.7	246.7
	FBS	253.7	246.7	48.7	29.0	3688.9	9997.9	10932.7	31681.6	66.8	10784.8	621.0	7669.2
	MMPC^1	7.0	16.9	1.2	1.3	44.2	28.1	733.7	388.2	13.3	2063.8	117.0	1033.3
	MMPC^2	17.7	96.9	7.3	2.6	278.0	68.4	8313.7	955.9	24.5	122716.3	715.0	3873.2
	MMPC^3	25.2	187.9	52.4	4.2	1315.8	160.0	N/A	1675.6	29.5	N/A	2065.3	10196.9
	MMPC^4	33.6	371.5	236.8	6.3	2959.9	405.1	N/A	4359.6	39.5	N/A	3761.7	24259.0
	LASSO-FS^{25}	12.2	7.0	2.7	6.0	12.3	13.7	6.9	118.6	0.2	1.6	0.4	4.0
	LASSO-FS^{100}	12.2	11.3	4.1	7.3	16.2	16.3	8.9	65.0	0.3	3.3	0.8	11.0
	LASSO-FS^{500}	10.5	19.7	4.9	11.2	22.5	34.1	24.5	133.4	1.0	14.7	3.5	51.1

References

Alan Agresti. *Categorical Data Analysis*. Wiley Series in Probability and Statistics. Wiley-Interscience, 2nd edition, 2002.

Hirotogu Akaike. Information theory and an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory*, pages 267–281, 1973.

Constantin F. Aliferis, Ioannis Tsamardinos, and Alexander Statnikov. HITON: a novel Markov blanket algorithm for optimal variable selection. In *AMIA Annual Symposium Proceedings*, volume 2003, page 21. American Medical Informatics Association, 2003.

Constantin F. Aliferis, Alexander Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D. Koutsoukos. Local causal and Markov blanket induction for causal discovery and feature selection for classification part i: Algorithms and empirical evaluation. *Journal of Machine Learning Research*, 11(Jan):171–234, 2010.

Thomas Blumensath and Mike E. Davies. On the difference between Orthogonal Matching Pursuit and Orthogonal Least Squares. Technical report, University of Edinburgh, 2007.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Gavin Brown, Adam Pocock, Ming-Jie Zhao, and Mikel Luján. Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. *Journal of Machine Learning Research*, 13:27–66, January 2012.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology*, 2(3):27, 2011.

Jiahua Chen and Zehua Chen. Extended Bayesian information criteria for model selection with large model spaces. *Biometrika*, 95(3):759–771, 2008.

Sheng Chen, Stephen A Billings, and Wan Luo. Orthogonal least squares methods and their application to non-linear system identification. *International Journal of control*, 50 (5):1873–1896, 1989.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995.

Samuel A. Danziger, S. Joshua Swamidass, Jue Zeng, Lawrence R. Dearth, Qiang Lu, Jonathan H. Chen, Jianlin Cheng, Vinh P. Hoang, Hiroto Saigo, Ray Luo, et al. Functional census of mutation sequence spaces: the example of p53 cancer rescue mutants. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(2):114–125, 2006.

Geoffrey M. Davis, Stephane G. Mallat, and Zhifeng Zhang. Adaptive time-frequency decompositions. *Optical engineering*, 33(7):2183–2192, 1994.

Thomas G. Dietterich, Ajay N. Jain, Richard H. Lathrop, and Tomas Lozano-Perez. A comparison of dynamic reposing and tangent distance for drug activity prediction. *Advances in Neural Information Processing Systems*, pages 216–216, 1994.

Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

Jianqing Fan, Yang Feng, Yichao Wu, et al. High-dimensional variable selection for Cox's proportional hazards model. In *Borrowing Strength: Theory Powering Applications–A Festschrift for Lawrence D. Brown*, pages 70–86. Institute of Mathematical Statistics, 2010.

Yingying Fan and Cheng Yong Tang. Tuning parameter selection in high dimensional penalized likelihood. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(3):531–552, 2013.

Matthias Feurer and Frank Hutter. Hyperparameter optimization. In Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors, *AutoML: Methods, Sytems, Challenges*, chapter 1, pages 3–37. Springer, December 2018. To appear.

Livio Finos, Chiara Brombin, and Luigi Salmaso. Adjusting stepwise p-values in generalized linear models. *Communications in Statistics-Theory and Methods*, 39(10):1832–1846, 2010.

Peter L. Flom and David L. Cassell. Stopping stepwise: Why stepwise and similar selection methods are bad, and what you should use. In *NorthEast SAS Users Group Inc 20th Annual Conference*, 2007.

Robert V. Foutz and R. C. Srivastava. The performance of the likelihood ratio test when the model is incorrect. *The Annals of Statistics*, 5(6):1183–1194, 1977.

Andreas Groll and Gerhard Tutz. Variable selection for generalized linear mixed models by L1-penalized estimation. *Statistics and Computing*, 24(2):137–154, 2014.

Max Grazier G'Sell, Stefan Wager, Alexandra Chouldechova, and Robert Tibshirani. Sequential selection procedures and false discovery rate control. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(2):423–444, 2016.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

Isabelle Guyon, Steve Gunn, Asa Ben-Hur, and Gideon Dror. Result analysis of the NIPS 2003 feature selection challenge. In *Advances in neural information processing systems*, pages 545–552, 2004.

Isabelle Guyon, Amir Reza Saffari Azar Alamdari, Gideon Dror, and Joachim M. Buhmann. Performance prediction challenge. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 1649–1656. IEEE, 2006a.

Isabelle Guyon, Jiwen Li, Theodor Mader, Patrick A Pletscher, Georg Schneider, and Markus Uhr. Feature selection with the CLOP package. Technical report, 2006b.

Mazin Abdulrasool Hameed. Comparative analysis of orthogonal matching pursuit and least angle regression. Master's thesis, Michigan State University, Electrical Engineering, 2012.

Frank Harrell. *Regression Modeling Strategies*. Springer, corrected edition, January 2001.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2nd edition, 2009.

Trevor Hastie, Robert Tibshirani, and Ryan J Tibshirani. Extended comparisons of best subset selection, forward stepwise selection, and the lasso. *arXiv preprint arXiv:1707.08692*, 2017.

Jing-Shiang Hwang and Tsuey-Hwa Hu. A stepwise regression algorithm for high-dimensional variable selection. *Journal of Statistical Computation and Simulation*, 85 (9):1793–1806, 2015.

Stéphane Ivanoff, Franck Picard, and Vincent Rivoirard. Adaptive lasso and group-lasso for functional Poisson regression. *Journal of Machine Learning Research*, 17(Jan):1903–1948, 2016.

George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. In *Machine learning: Proceedings of the Eleventh International Conference*, pages 121–129. Morgan Kaufmann, 1994.

Markus Kalisch and Peter Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8(Mar):613–636, 2007.

Yongdai Kim, Sunghoon Kwon, and Hosik Choi. Consistent model selection criteria on high dimensions. *Journal of Machine Learning Research*, 13(Apr):1037–1057, 2012.

Michael H. Kutner, Christopher J. Nachtsheim, John Neter, and William Li. *Applied Linear Statistical Models*. McGraw-Hill/Irwin, 5 edition, August 2004.

Vincenzo Lagani, Giorgos Athineou, Alessio Farcomeni, Michail Tsagris, and Ioannis Tsamardinos. Feature selection with the R package MXM: Discovering statistically equivalent feature subsets. *Journal of Statistical Software*, 80(7), 2017.

Jinchi Lv and Jun S. Liu. Model selection principles in misspecified models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1):141–167, 2014.

Dimitris Margaritis. Toward provably correct feature selection in arbitrary domains. In *Advances in Neural Information Processing Systems*, pages 1240–1248, 2009.

Dimitris Margaritis and Sebastian Thrun. Bayesian network induction via local neighborhoods. In S. A. Solla, T. K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 505–511. MIT Press, 2000.

Lukas Meier, Sara Van De Geer, and Peter Bühlmann. The group Lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2008.

Andrew Y. Ng. Feature selection, L1 vs. L2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM, 2004.

Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Conference Record of The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers*, pages 40–44. IEEE, 1993.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

Judea Pearl. *Causality, Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, U.K., 2000.

Peter Peduzzi, John Concato, Elizabeth Kemper, Theodore R. Holford, and Alvan R. Feinstein. A simulation study of the number of events per variable in logistic regression analysis. *Journal of clinical epidemiology*, 49(12):1373–1379, 1996.

Junyang Qian, Ttrevor Hastie, Jerome Friedman, Rob Tibshirani, and Noah Simon. Glmnet for Matlab, 2013.

Thomas Richardson. Markov properties for acyclic directed mixed graphs. *Scandinavian Journal of Statistics*, 30(1):145–157, 2003.

Thomas Richardson and Peter Spirtes. Ancestral graph Markov models. *Annals of Statistics*, pages 962–1030, 2002.

Jürg Schelldorfer, Peter Bühlmann, and Sara Van De Geer. Estimation for high-dimensional linear mixed-effects models using L1-penalization. *Scandinavian Journal of Statistics*, 38 (2):197–214, 2011.

Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2): 461–464, 1978.

Peter Spirtes, Clark N. Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2 edition, 2000.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, pages 267–288, 1996.

Ryan J. Tibshirani, Jonathan Taylor, Richard Lockhart, and Robert Tibshirani. Exact post-selection inference for sequential regression procedures. *Journal of the American Statistical Association*, 111(514):600–620, 2016.

Michail Tsagris, Vincenzo Lagani, and Ioannis Tsamardinos. Feature selection for high-dimensional temporal data. *BMC bioinformatics*, 19(1):17, 2018.

Ioannis Tsamardinos and Constantin F. Aliferis. Towards principled feature selection: relevancy, filters and wrappers. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

Ioannis Tsamardinos, Constantin F. Aliferis, and Alexander Statnikov. Time and sample efficient discovery of Markov blankets and direct causal relations. In *Proceedings of the Ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 673–678. ACM, 2003a.

Ioannis Tsamardinos, Constantin F. Aliferis, and Alexander R. Statnikov. Algorithms for large scale Markov blanket discovery. In *FLAIRS conference*, volume 2, 2003b.

Ioannis Tsamardinos, Giorgos Borboudakis, Pavlos Katsogridakis, Polyvios Pratikakis, and Vassilis Christophides. A greedy feature selection algorithm for Big Data of high dimensionality. *Machine Learning*, Aug 2018a.

Ioannis Tsamardinos, Elissavet Greasidou, and Giorgos Borboudakis. Bootstrapping the out-of-sample predictions for efficient and accurate cross-validation. *Machine Learning*, May 2018b.

Thomas S. Verma and Judea Pearl. Causal Networks: Semantics and Expressiveness. In *Proceedings, 4th Workshop on Uncertainty in Artificial Intelligence*, pages 352–359, August 1988.

Eric Vittinghoff and Charles E. McCulloch. Relaxing the rule of ten events per variable in logistic and Cox regression. *American journal of epidemiology*, 165(6):710–718, 2007.

Quang H. Vuong. Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica: Journal of the Econometric Society*, pages 307–333, 1989.

Sanford Weisberg. *Applied linear regression*, volume 528. John Wiley & Sons, 2005.

Halbert White. Maximum likelihood estimation of misspecified models. *Econometrica*, 50 (1):1–25, 1982.

Samuel S. Wilks. The large-sample distribution of the likelihood ratio for testing composite hypotheses. *The Annals of Mathematical Statistics*, 9(1):60–62, March 1938.

Kun Zhang, Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. Kernel-based conditional independence test and application in causal discovery. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 804–813, 2011.

Yongli Zhang and Xiaotong Shen. Model selection procedure for high-dimensional data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 3(5):350–358, 2010.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67:301–320, 2005.

Journal of Machine Learning Research 20 (2019) 1-49 Submitted 6/17; Revised 8/18; Published 2/19

Dynamic Pricing in High-dimensions

Adel Javanmard AJAVANMA@USC.EDU
Department of Data Sciences and Operations
Marshall School of Business
University of Southern California
Los Angeles, CA 90089 , USA

Hamid Nazerzadeh NAZERZAD@USC.EDU
Department of Data Sciences and Operations
Marshall School of Business
University of Southern California
Los Angeles, CA 90089 , USA

Editor: Peter Auer

Abstract

We study the pricing problem faced by a firm that sells a large number of products, described via a wide range of features, to customers that arrive over time. Customers independently make purchasing decisions according to a general choice model that includes products features and customers' characteristics, encoded as d-dimensional numerical vectors, as well as the price offered. The parameters of the choice model are a priori unknown to the firm, but can be learned as the (binary-valued) sales data accrues over time. The firm's objective is to maximize its revenue. We benchmark the performance using the classic regret minimization framework where the regret is defined as the expected revenue loss against a clairvoyant policy that knows the parameters of the choice model in advance, and always offers the revenue-maximizing price. This setting is motivated in part by the prevalence of online marketplaces that allow for real-time pricing.

We assume a structured choice model, parameters of which depend on s_0 out of the d product features. Assuming that the market noise distribution is known, we propose a dynamic policy, called Regularized Maximum Likelihood Pricing (RMLP) that leverages the (sparsity) structure of the high-dimensional model and obtains a logarithmic regret in T. More specifically, the regret of our algorithm is of $O(s_0 \log d \cdot \log T)$. Furthermore, we show that no policy can obtain regret better than $O(s_0(\log d + \log T))$. In addition, we propose a generalization of our policy to a setting that the market noise distribution is unknown but belongs to a parametrized family of distributions. This policy obtains regret of $O(\sqrt{(\log d)T})$. We further show that no policy can obtain regret better than $\Omega(\sqrt{T})$ in such environments.

Keywords: Revenue Management, Dynamic Pricing, High-dimensional Regression, Maximum Likelihood, Regret, Sparsity

1. Introduction

A central challenge in revenue management is determining the optimal pricing policy when there is uncertainty about customers' willingness to pay. Due to its importance, this problem has been studied extensively (Kleinberg and Leighton, 2003; Besbes and Zeevi, 2009; Badanidiyuru et al., 2013; Wang et al., 2014; Broder and Rusmevichientong, 2012; Keskin and Zeevi, 2014; den Boer and Zwart, 2014; Cohen et al., 2016). Most of these models are built around the following classic setting: customers arrive over time; the seller posts a price for each customer; if the customer's valuation is above the posted price, a sale occurs and the seller collects a revenue in the amount of the posted price; otherwise, no sale occurs and no revenue is generated. Based on this and the previous feedbacks, the seller updates the posted price. Therefore, the seller is involved in the realm of exploration-exploitation as he needs to choose between learning about the valuations and exploiting what has been learned so far to collect revenue.

In this work, we consider a setting with a large number of products which are defined via a wide range of features. The valuations are given by $v(\theta, x)$ with x being the (observable) feature vectors of products and θ_0 representing the customer's characteristics true parameters of the choice model, which is initially unknown to the seller, cf. (Amin et al., 2014; Cohen et al., 2016). An important special case of this setting is the linear model in which

$$v(\theta, x) = \theta_0 \cdot x + z\,, \tag{1}$$

where z captures the idiosyncratic noise in valuations.

Our setting is motivated in part by applications in online marketplaces. For instance, a company such as Airbnb recommends prices to hosts based on many features including the space (number of rooms, beds, bathrooms, etc.), amenities (AC, WiFi, washer, parking, etc.), the location (accessibility to public transportation, walk score of the neighborhood, etc.), house rules (pet-friendly, non-smoking, etc.), as well as the prediction of the demand which itself depends on many factors including the date, events in the area, availability and prices of near-by hotels, etc. (Airbnb Documentation, 2015). Therefore, the vector describing each property can have hundreds of features. Another important application comes from online advertising. Online publishers set the (reserve) price of ads based on many features including user's demographic, browsing history, the context of the webpage, the size and location of the ad on the page, etc.

In this work, we propose *Regularized Maximum Likelihood Pricing* (RMLP) policy for dynamic pricing in high-dimensional environments. As suggested by its name, the policy uses maximum likelihood method to estimate the true parameters of the choice model. In addition, using an (ℓ_1-norm) regularizer, our policy exploits the structure of the optimal solution; namely, the performance of the RMLP policy significantly improves if the valuations are essentially determined by a small subset of features. More formally, the difference between the revenue obtained by our policy and the benchmark policy that knows in advance the true parameters of the choice model, θ_0, is bounded by $O\big(s_0 \log d \cdot \log T\big)$, where T, d, and s_0 respectively denote the length of the horizon, number of the features, and sparsity (i.e., number of non-zero elements of θ_0). We show that our results are nearly optimal and no policy can obtain regret better than $O\big(s_0(\log d + \log T)\big)$.

In the RMLP policy it is assumed that the noise distribution (distribution of term z in valuation model (1)) is known to the seller. We also consider the setting that the noise distribution is unknown but belongs to a parametrized family of distributions. We propose RMLP-2 policy for such setting and prove that it achieves a regret of $O(\sqrt{(\log d)T})$. We further how that no policy can get regret better than $\Omega(\sqrt{T})$ in this setting.

©2019 Adel Javanmard and Hamid Nazerzadeh.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-357.html.

We point out that our results can be applied to applications where the features dimension is larger than the time horizon of interest. A powerful pricing policy for these applications should obtain regret that scales gracefully with the dimension. Note that in general, little can be learned about the model parameters θ_0 if $T < d$, because the number of degrees of freedom d exceeds the number of observations T, and therefore, any estimator can be arbitrary erroneous. However, when there is prior knowledge about the structure of unknown parameter θ_0, (e.g., sparsity), then accurate estimations are attainable even when $T < d$.

1.1. Organization and Contributions

In the remaining part of the introduction, we discuss how our work is positioned with respect to the literature on dynamic pricing and high-dimensional statistics. The rest of the paper presents the following contributions:

- Section 2. We formally present our model and discuss the technical assumptions and the benchmark policy.
- Section 3. We present the RMLP policy that uses a regularized maximum likelihood function to estimate the parameters of the choice model and subsequently sets the best prices based on the obtained estimates. The RMLP policy assumes that the market noise distribution is known and forms the log-likelihood function based on this knowledge.
- Section 4. We analyze the regret of the RMLP policy with respect to a clairvoyant who knows the parameters of the choice model in advance. We show that it achieves a regret of $O(s_0 \log d \cdot \log T)$.
- Section 5. We provide a lower bound on the regret of any policy that does not know the parameters of the choice model in advance. Namely, we show that no policy can obtain regret better than $O(s_0(\log d + \log T))$.
- Section 6. We generalize the RMLP policy to non-linear valuations functions and show that, similar to the case of linear valuations, it achieves a regret of $O(s_0 \log d \cdot \log T)$.
- Section 7. We consider a setting where the exact underlying noise distribution is not known but it is assumed to belong to a known class of log-concave distributions. We propose the RMLP-2 policy, so-named, for this setting whose regret is $O(\sqrt{(\log d)T})$. We also argue that no policy can get better than $O(\sqrt{T})$ regret in this setting.
- Section 8. We provide the proofs of our main theorems and results.

We conclude our work along with some future directions in Section 9. Proof of technical lemmas are deferred to the appendices.

1.2. Related Work

Our work contributes to the literature on dynamic pricing as well as high dimensional statistics. In the following, we briefly overview the work closest to ours in these contexts.

Dynamic pricing and learning. The literature on dynamic pricing and learning has been growing over the past few years, motivated in part by the advances in big data technology that allow firms to easily collect and utilize information. We briefly discuss some of the recent lines of research in this literature. We refer to den Boer (2015) for an excellent survey on this topic.

Parametric Approach. A natural approach to capture uncertainty about the customers' valuations is to model the uncertainty using a small number of parameters, and then estimate those parameters using classical statistical methods such as maximum likelihood (Broder and Rusmevichientong, 2012; den Boer and Zwart, 2013, 2014; Chen et al., 2015) or least square estimation (Keskin, 2014; Bastani and Bayati, 2016). Our work is similar to this line of work, in that we assume a parametric model for customer's valuations and apply the maximum likelihood method using the randomness of the idiosyncratic noise in valuations. However, the parameter vector θ is high-dimensional, whose dimension d (that can even exceed the time horizon of interest T). We use *regularized* maximum-likelihood in order to promote sparsity structure in the estimated parameter. Further, our pricing policy has an episodic theme which makes the posted prices p_t in each episode independent of the idiosyncratic noise in valuations, z_t, in that episode. This is in contrast to other policies based on maximum-likelihood, such as MLE-GREEDY (Broder and Rusmevichientong, 2012), or greedy iterative least square (GILS) (Keskin, 2014; den Boer and Zwart, 2014; Qiang and Bayati, 2016) that use the entire history of observations to update the estimate for the model parameters at each step.

Bayesian Approach. One of the earliest work on Bayesian parametric approach in this context is by (Rothschild, 1974) who consider a Bayesian framework where the firm can choose from two prices with unknown demand and show that (myopic) Bayesian policies may lead to "incomplete learning." However, carefully designed variations of the myopic policies can (optimally) learn the optimal price (Harrison et al., 2012); see also Keller and Rady (1999); Araman and Caldentey (2009).

Non-Parametric models. An early work in non-parametric setting is by Kleinberg and Leighton (2003). They model the dynamic pricing problem as a multi-armed bandit (MAB) where each arm corresponds to a (discretized) posted price. They propose an $O(\sqrt{T})$-algorithm where T is the length of the horizon. Similar results have been obtained in more general settings (Badanidiyuru et al., 2013; Agrawal and Devanur, 2014) including setting with inventory constraints (Besbes and Zeevi, 2009; Babaioff et al., 2012; Wang et al., 2014).

Feature-based Models. Recent papers on dynamic pricing consider models with features/covariates. Amin et al. (2014), in a model similar to ours, present an algorithm that obtains regret $O(T^{2/3})$; they also study dynamic incentive compatibility in repeated auctions. Another closely related work to ours is by Cohen et al. (2016). Their model differs from ours in two main aspects: *i*) their model is deterministic (no idiosyncratic noise) *ii*) the arrivals (of features vectors) is modeled as adversarial. They propose a clever binary-search approach using the Ellipsoid method which obtains regret of $O(d^2 \log(T/d))$. Recently, Golrezaei et al. (2018) studied the problem of dynamic pricing when a group of strategic buyers competes with each other in repeated contextual auctions. To the extent of our knowledge, ours is the first work that highlights the role of structure/sparsity in dynamic pricing.

Qiang and Bayati (2016) study a model where the seller can observe the demand itself, not a binary signal as in our setting. They show that a myopic policy based on least-square estimations can obtain a logarithmic regret. Bastani and Bayati (2016) study a multi-armed bandit setting, with discrete arms, and high-dimensional covariates, generalizing results of Goldenshluger and Zeevi (2013). More precisely, the decision-maker has access to K arms (decisions), where each arm i has an unknown sparse parameter vector $\beta_i \in \mathbb{R}^d$. Upon pulling arm i at time t, the decision-maker gets reward $x_t \cdot \beta_i + z_t$. The authors present an algorithm, using a LASSO estimator, that obtains regret $O\left(K(\log T + \log d)^2\right)$ where K denotes the number of arms. By contrast, in our setting decisions (prices) can take continuous values and also the resulting rewards do not follow a similar relationship with the covariates and the decisions.

High-dimensional statistics. There has been a great deal of work on regularized estimator under the high-dimensional scaling; see e.g., Van de Geer (2008). Closer to the spirit of our work is the problem of 1-bit compressed sensing (Plan and Vershynin, 2013; Bhaskar and Javanmard, 2015). In this problem, linear measurements are observed for an unknown parameter of interest but only the sign of these measurements are observed. Note that in our problem, the seller is involved in both the learning task and also the policy design. Specifically, he should decide on the prices, which directly affect collected revenue and also indirectly influence the difficulty of the learning task. The market values are then compared with the posted prices, in contrast to 1-bit compressed sensing where the measurements are compared with zero (sign information). In addition, the pricing problem has an online nature while the 1-bit compressed sensing is mostly studied for offline setting. Finally, note that prices are set based on customer's purchase behavior, and hence introduce dependency among the collected information about the model parameters.

1.3. Notations

For a vector v, $\mathrm{supp}(v)$ represents the positions of nonzero entries of v. Further, for a vector v and a subset J, v_J is the restriction of v to indices in J. We write $\|v\|_p$ for the standard ℓ_p norm of a vector v, i.e., $\|v\|_p = (\sum_i |v_i|^p)^{1/p}$ and $\|v\|_0$ for the umber of nonzero entries of v. If the subscript p is omitted, it should be deemed as ℓ_2 norm. For two vectors $a, b \in \mathbb{R}^d$, the notation $a \cdot b = \sum_{i=1}^d a_i b_i$ represents the standard inner product. For two functions $f(n)$ and $g(n)$, the notation $f(n) = O(g(n))$ means that f is bounded above by g asymptotically, namely, $f(n) \le Cg(n)$ for some fixed positive constant $C > 0$. Throughout, $\phi(x) = e^{-x^2/2}/\sqrt{2\pi}$ is the Gaussian density and $\Phi(x) \equiv \int_{-\infty}^{x} \phi(u)\mathrm{d}u$ is the Gaussian distribution.

2. Choice Model

We consider a seller, who has a product for sale in each period $t = 1, 2, \cdots, T$, where T denotes the length of the horizon and may be unknown the to the seller. Each product is represented by an *observable* vector of features (covariates) $x_t \in \mathcal{X} \subseteq \mathbb{R}^d$. Products may vary across periods and we assume that feature vectors x_t are sampled independently from a fixed, but a priori *unknown*, distribution $\mathbb{P}_X$, supported on a bounded set $\mathcal{X}$.

The product at time t has a market value $v_t = v(x_t)$, which is *not observed* by the seller and function v is (a priori) unknown. At each period t, the seller posts a price p_t. If $p_t \le v_t$, a sale occurs, and the seller collects revenue p_t. If the price is set higher than the market value, $p_t > v_t$, no sale occurs and no revenue is obtained. The goal of the seller is to design a pricing policy that maximizes the collected revenue.

We first assume that the market value of a product is a linear function of its covariates, namely

$$v(x_t) = \theta_0 \cdot x_t + z_t\,, \tag{2}$$

where $a \cdot b$ denotes the inner product of vectors a and b. Here, $\{z_t\}_{t\ge 1}$ are idiosyncratic shocks, referred to as noise, which are drawn independently and identically from a distribution with mean zero and cumulative function F, with density $f(x) = F'(x)$, cf. (Keskin and Zeevi, 2014). The noise can account for the features that are not measured. We generalize our model to non-linear valuation functions in Section 6. Parameter θ_0 is a priori unknown to the seller. Therefore, the seller is involved in the realm of exploration-exploitation as he needs to choose between learning θ_0 and exploiting what has been learned so far to collect revenue.

Let y_t be the response variable that indicates whether a sale has occurred at period t:

$$y_t = \begin{cases} +1 & \text{if } v_t \ge p_t\,, \\ -1 & \text{if } v_t < p_t\,. \end{cases} \tag{3}$$

Note that the above model can be represented as the following probabilistic model:

$$y_t = \begin{cases} +1 & \text{with probability } \ 1 - F\left(p_t - \theta_0 \cdot x_t\right)\,, \\ -1 & \text{with probability } \ F\left(p_t - \theta_0 \cdot x_t\right)\,. \end{cases} \tag{4}$$

Our proposed algorithm exploits the structure (sparsity) of the feature space to improve its performance. To this aim, let s_0 denote the number of nonzero coordinates of θ_0, i.e., $s_0 = \|\theta_0\|_0 = \sum_{j=1}^d \mathbb{I}(\theta_{0j} \neq 0)$. We remark that s_0 is a priori unknown to the seller.

2.1. Technical Assumptions

To simplify the presentation, we assume that $\|x_t\|_\infty \le 1$, for all $x_t \in \mathcal{X}$, and $\|\theta_0\|_1 \le W$ for a known constant $W \ge 1$, where for a vector $u = (u_1, \ldots, u_d)$, $\|u\|_\infty = \max_{i\in[d]} |u_i|$ denotes the maximum absolute value of its entries and $\|u\|_1 = \sum_{i=1}^d |u_i|$. We denote by Ω the set of feasible parameters, i.e.,

$$\Omega = \left\{\theta \in \mathbb{R}^{d+1} : \|\theta\|_0 \le s_0\,,\ \|\theta\|_1 \le W\right\}. \tag{5}$$

We also make the following assumption on the distribution of noise F.

Assumption 1 *The function $F(v)$ is strictly increasing. Further, $F(v)$ and $1 - F(v)$ are log-concave in v.*

Log-concavity is a widely-used assumption in the economics literature (Bagnoli and Bergstrom, 2005). Note that if the density f is symmetric and the distribution F is log-concave, then $1-F$ is also log-concave. Assumption 1 is satisfied by several common probability distributions including normal, uniform, Laplace, exponential, and logistic. Note that the cumulative distribution function of all log-concave densities is also log-concave (Boyd and Vandenberghe, 2004).

Our second assumption is on the product feature vectors.

Assumption 2 *Product feature vectors are generated independently from a fixed unknown distribution $\mathbb{P}_X$ with a bounded support $\mathcal{X} \in \mathbb{R}^d$. We denote by $\Sigma = \mathbb{E}(x_t x_t^\intercal)$ the second moment matrix of distribution $\mathbb{P}_X$, and assume that Σ is a positive definite matrix. Namely, all of its eigenvalues are bounded from below by a constant $C_{\min} > 0$. We also denote the maximum eigenvalue of Σ by $C_{\max}$.*

The above assumption holds for many common probability distributions, such as uniform, truncated normal, and in general truncated version of many more distributions. Generally, if $\mathbb{P}_X$ is bounded below from zero on an open set around the origin, then it has a positive definite second moment matrix. Let us stress that the seller knows neither the distribution $\mathbb{P}_X$, nor its second moment Σ.

2.2. Clairvoyant Policy and Performance Metric

We evaluate the performance of our algorithm using the common notion of regret: the expected revenue loss compared with the optimal pricing policy that knows θ_0 in advance (but not the realizations of $\{z_t\}_{t\geq 1}$). Let us first characterize this benchmark policy.

Using Eq. (2), the expected revenue from a posted price p is equal to

$$p \times \mathbb{P}(v_t \geq p) = p(1 - F(p - \theta_0 \cdot x_t))\,.$$

Therefore, using first order conditions, for the optimal posted price, denoted by p^*, we have

$$p^*(x_t) = \frac{1 - F(p^* - \theta_0 \cdot x_t)}{f(p^* - \theta_0 \cdot x_t)}\,. \tag{6}$$

To simplify the presentation, let $p_t^* = p^*(x_t)$ denote the optimal price at time t.

We now define $\varphi(v) \equiv v - (1-F(v))/f(v)$ corresponding to the *virtual valuation* function commonly used in mechanism design (Myerson, 1981). By Assumption 1, φ is injective and hence we can define function g as follows

$$g(v) \equiv v + \varphi^{-1}(-v)\,. \tag{7}$$

It is easy to verify that g is non-negative. Note that by Eq. (6), for the optimal price we have

$$\theta_0 \cdot x_t + \varphi(p^* - \theta_0 \cdot x_t) = 0\,.$$

Therefore, by rearranging the terms for the optimal price at time t we have

$$p_t^* = g(\theta_0 \cdot x_t)\,. \tag{8}$$

We can now formally define the regret of a policy. Let π be the seller's policy that sets price p_t at period t, and p_t can depend on the history of events up to time t. The worst-case regret is defined as:

$$\mathsf{Regret}_\pi(T) \equiv \max_{\substack{\theta_0 \in \Omega \\ \mathbb{P}_X \in Q(\mathcal{X})}} \mathbb{E}\left[\sum_{t=1}^{T}\Big(p_t^* \mathbb{I}(v_t \geq p_t^*) - p_t \mathbb{I}(v_t \geq p_t)\Big)\right], \tag{9}$$

where the expectation is with respect to the distributions of idiosyncratic noise, z_t, and $\mathbb{P}_X$, the distribution of feature vectors. Also, $Q(\mathcal{X})$ represents the set of probability distributions supported on a bounded set $\mathcal{X}$.

Our algorithm uses the sparsity structure of θ_0 and learns the model with order of magnitude less data compared to a structure-ignorant algorithm. In Section 4, we show that our pricing scheme achieves a regret bound of $O\big(s_0 \log d \cdot \log T\big)$.

Note that the setting with known market noise distribution is equivalent to positing a parametric choice model on the users' behavior. Several parametric choice models have been widely used by practitioners and have been analyzed by researchers. Perhaps the most well-known example is the multinomial logit (MNL) model, which has quite a long history in economic and transportation research (McFadden, 2001), as well as marketing (Hauser and Koppelman, 1979; Gensch and Recker, 1979; Louviere and Woodworth, 1983; Guadagni and Little, 2008). Note that the MNL model corresponds to the case where customers are offered multiple products at each round and the market noise is coming from the Gumbel distribution (Talluri and Van Ryzin, 2006). In the case of single product offered at each round (similar to our setting), this reduces to noise coming from the logistic distribution. Other prominent MNL models include Nested MNL, latent-class MNL, random-parameter mixed MNL, and hierarchical Bayesian MNL (Elshiewy et al., 2017).

In Section 7 we relax the assumption of knowing the market noise distribution.

3. A Regularized Maximum Likelihood Pricing (RMLP) Policy

In this section, we present our dynamic pricing policy. Our policy runs in an episodic fashion. Episodes are indexed by k and time periods are indexed by t. The length of episode k is denoted by τ_k. Throughout episode k, we set the prices equal to $p_t = g(x_t \cdot \widehat{\theta}^k)$ where $\widehat{\theta}^k$ denotes the estimate of θ_0 which is obtained from the observations $\{(x_t, y_t, p_t)\}$ in the *previous* episode. Note that by Eq. (7), p_t is the optimal posted price if $\widehat{\theta}^k$ was the true underlying parameter of the model.

We estimate θ_0 using a regularized maximum-likelihood estimator; see Eq. (10) where the (normalized) negative log-likelihood function for θ is given by Eq. (11). We note that as a consequence of the log concavity assumption on F and $1-F$, the optimization problem (10) is a convex problem. There is a large toolkit of various optimization methods (e.g., alternating direction method of multipliers (ADMM), fast iterative shrinkage-thresholding algorithm (FISTA), accelerated projected gradient descent, among many others) that can be used to solve this optimization problem. There are also recent developments on distributed solvers for ℓ_1 regularized cost function (Boyd et al., 2011).

Observe that by design, prices posted in the k-th episode are independent from the market value noises in this period, i.e., $\{z_t\}_{t=\tau_k}^{\tau_{k+1}-1}$. This allows us to estimate θ_0 for each

Input: (at time 0) function g, regularizations λ_k, W (bound on $\|\theta_0\|_1$),
Input: (arrives over time) covariate vectors $\{x_t\}_{t\in\mathbb{N}}$
Output: prices $\{p_t\}_{t\in\mathbb{N}}$

1: $\tau_1 \leftarrow 1$, $p_1 \leftarrow 0$, $\widehat{\theta}^1 \leftarrow 0$
2: **for** each episode $k = 2, 3, \ldots$ **do**
3: Set the length of k-th episode: $\tau_k \leftarrow 2^{k-1}$.
4: Update the model parameter estimate $\widehat{\theta}^k$ using the regularized ML estimator obtained
from observations in the previous episode:

$$\widehat{\theta}^k = \underset{\|\theta\|_1 \le W}{\arg\min} \left\{ \mathcal{L}(\theta) + \lambda_k \|\theta\|_1 \right\}, \tag{10}$$

with

$$\mathcal{L}(\theta) = -\frac{1}{\tau_{k-1}} \sum_{t=\tau_{k-1}}^{\tau_k - 1} \Big\{ \mathbb{I}(y_t = 1) \log(1 - F(p_t - \theta \cdot x_t)) + \mathbb{I}(y_t = -1) \log(F(p_t - \theta \cdot x_t)) \Big\}. \tag{11}$$

5: For each period t during the k-th episode, set

$$p_t \leftarrow g(\widehat{\theta}^k \cdot x_t). \tag{12}$$

Algorithm 1: RMLP policy for dynamic pricing

episode separately; see Proposition 10 in Section 8.1. Comparing to policies that use the entire data sale history in making decisions, some remarks are in order:

- *Perishability of data:* In practical applications, the unknown demand parameters will change over time, raising the concern of perishability of data. Namely, collected data becomes obsolete after a while and cannot be relied on for estimating the model parameters (Keskin and Zeevi, 2016; Javanmard, 2017). Common policies to mitigate this problem (discussed in (Keskin and Zeevi, 2016)) include moving windows and decaying weights which use only recent data to learn the model parameters. By contrast, methods that use the entire historical data, with equal weight, suffers from this problem.

- *Simplicity and efficiency:* In RMLP policy, estimates of the model parameters are updated only at the first period of each episode ($\log T$ updates). Further, at each update, the policy uses only the historical data from the previous episode. These two ideas together, not only allow for a neat analysis of the statistical dependency among samples but also decrease the computational cost. Scalability of the pricing policy is indispensable in practical applications as the sales data is collected at an unprecedented rate.

- *The impact on regret:* By using half of the historical data at each update, our policy loses at most a factor 2 in the total regret. (This becomes clear shortly when we discuss the estimation error rate in terms of number of samples.)

The lengths of episodes in our algorithm increase geometrically ($\tau_k = 2^{k-1}$), allowing for more accurate estimate of θ_0 as the episode index grows. The algorithm terminates at the end of the horizon (period T), but note that it does not need to know the length of the horizon in advance.

The regularization parameter λ_k constrains the ℓ_1 norm of the estimator $\widehat{\theta}^k$. Selecting the value of λ_k is of crucial importance as it affects the estimator error. We set it as $\lambda_k = O\left(\sqrt{(\log d)/\tau_{k-1}}\right)$. More precisely, define

$$u_F \equiv \sup_{|x|\le 3W} \Big\{ \max\Big\{ \log' F(x), -\log'(1 - F(x)) \Big\} \Big\},$$

where the derivatives are w.r.t. x. By the log-concavity property of F and $1 - F$, we have

$$u_F = \max\Big\{ \log' F(-2W), -\log'(1 - F(2W)) \Big\}. \tag{13}$$

Hence, u_F captures the steepness of $\log F$. In order to minimize the regret, we run the RMLP policy with

$$\lambda_k = 4u_F \sqrt{\frac{\log d}{\tau_{k-1}}}. \tag{14}$$

Note that exploration and exploitation tasks are mixed in our algorithm. In the beginning of each episode, we use what is learned from previous episode to improve the estimation of θ_0 and then we exploit this estimate throughout the current episode to incur little regret. Meanwhile, the observations gathered in the current episode are used to update our estimate of θ_0 for the next episode.

The knowledge of the market noise distribution is used by the RMLP policy, in estimating the model parameters (constructing the log-likelihood function and choosing the regularization λ_k) as well as in setting the prices. (The pricing function g is defined based on the noise distribution F.)

In the next section we analyze the regret of the RMLP policy.

4. Regret Analysis

The following theorem bounds the regret of our dynamics pricing policy.

Theorem 3 (Regret Upper Bound) *Suppose Assumptions 1 and 2 hold and the . Then, the regret of the RMLP policy is of* $O\left(\frac{W}{C_{\min}^2} s_0 \log d \cdot \log T\right)$.

Let us stress again that the RMLP policy does not need to know the sparsity s_0, although its regret scales linearly with s_0. Further, the RMLP is assumed to know the market noise distribution F. [1] Below we provide an outline for the proof of Theorem 3 and defer its complete proof to Section 8.1.

1. It is useful to compare Theorem 3 with the result of Chen et al. (2015), [Theorem 5.4], which also uses a maximum likelihood estimator without exploiting sparsity structure and achieves regret $O\left(\frac{d}{C_{\min}} \sqrt{\log(Td)T}\right)$.

1. In RMLP, the updates in the model parameter estimation only occurs at the beginning of each episode, with using only the samples collected in the previous episode. Therefore, the prices posted in each episode are independent from the market value noises in that episode. This observation also verifies that $\mathcal{L}(\theta)$ given by (11), is indeed the negative log-likelihood of the samples collected in k-th episode. Note that this independence is not a mere serendipity, rather it holds because of the specific design of RMLP policy. Using this property, we use tools from high-dimensional statistics to bound the estimation error. To bound the error term $\|\theta^k - \theta_0\|_2$, we compare the function values $\mathcal{L}(\theta^k)$ and $\mathcal{L}(\theta_0)$. The main challenge here is that $\mathcal{L}(\theta)$ is not strictly convex in θ.[2] Hence, there can be, in principle, parameter vectors θ_1 and θ_2 that are close to each other and nevertheless the values of function $\mathcal{L}$ at these points are far from each other.

   To cope with this challenge, we show that a so-called *restricted eigenvalue condition* holds for the feature products. This notion implies that $\mathcal{L}(\theta)$ is strictly convex on the set of sparse vectors.[3] Using the restricted eigenvalue condition, we show the following ℓ_2 error for the regularized log-likelihood estimate in the k-th episode, $\widehat{\theta}^k$, holds true

$$\|\widehat{\theta}^k - \theta_0\|_2 = O(\sqrt{s_0}\lambda_k) = O\left(\sqrt{\frac{s_0 \log d}{\tau_{k-1}}}\right).$$

   As expected, the estimate gets more accurate as the episode's length increases; see Section 8.1 for more details.

2. For any $p \geq 0$, denote by $\mathsf{rev}_t(p) = p(1-F(p-x_t\cdot\theta_0))$, the expected revenue under price p. We bound reg_t in terms of $\mathsf{rev}_t(p_t^*) - \mathsf{rev}_t(p_t)$. Since $p_t^* \in \arg\max\{\mathsf{rev}_t(p)\}$, we have $\mathsf{rev}_t'(p_t^*) = 0$, and by Taylor expansion of rev_t around p_t^*, we obtain $\mathsf{rev}_t(p_t^*) - \mathsf{rev}_t(p_t) = O((p_t^* - p_t)^2)$.

3. For t in the k-th episode, namely $\tau_{k-1} \leq t \leq \tau_k - 1$, we have

$$p_t^* - p_t = g(\theta_0 \cdot x_t) - g(\widehat{\theta}^k \cdot x_t) \leq |(\theta_0 - \widehat{\theta}^k) \cdot x_t|,$$

   which follows by showing that g is 1-Lipschitz. Further, by Assumption 2 (without loss of generality assume $C_{\max} > 1$), we have

$$\mathbb{E}[((\theta_0 - \widehat{\theta}^k) \cdot x_t)^2] = \mathbb{E}[(\theta_0 - \widehat{\theta}^k)^\mathsf{T} \Sigma (\theta_0 - \widehat{\theta}^k)] \leq C_{\max} \mathbb{E}[\|\widehat{\theta}^k - \theta_0\|_2^2],$$

   where the equality holds because x_t is independent of $\widehat{\theta}^k$, and the last step follows since the maximum eigenvalue of the second moment matrix $\Sigma = \mathbb{E}(x_t x_t^\mathsf{T})$ is at most $C_{\max} > 0$.

2. Note that $\nabla^2_\theta \mathcal{L} = (-1/\tau_{k-1}) \sum_{t=\tau_{k-1}}^{\tau_k - 1} (\partial^2/\partial_{u_t}^2 \mathcal{L}) x_t x_t^\mathsf{T}$, where $u_t = p_t - \theta \cdot x_t - \alpha_0$. Therefore, $\nabla^2_\theta \mathcal{L}$ is a $d \times d$ matrix of rank at most $\tau_k - \tau_{k-1}$. Hence, $\mathcal{L}(\theta)$ is strictly convex in θ only if $\tau_k - \tau_{k-1} \geq d$. However, since we are not updating our estimates in the middle of an episode, episodes of length d yield the regret to scale linearly in d, which is not desired.

3. It is strictly convex over the set of s_0 sparse vectors in d-dimension if the number of samples is above $cs_0 \log d$ for a suitable constant $c > 0$.

   Let reg_t be the regret occurred at step t. Combining the above bounds (step 2 and 3), we arrive at $\mathbb{E}[\mathsf{reg}_t] = O(s_0(\log d)/\tau_{k-1})$. Therefore, the cumulative expected regret in episode k works out at $O(s_0 \log d)$. Since the length of episodes increase geometrically, there are $O(\log T)$ episodes by time T. This implies that the total expected regret by time T is $O(s_0 \log d \log T)$.

4.1. Comparison with the "Common" Regret of Bound $\Omega(\sqrt{T})$

There is an often-seen regret bound $\Omega(\sqrt{T})$ in the literature of online decision making, which can be improved to a logarithmic regret bound if some type of "separability assumption" holds true (Dani et al., 2008; Abbasi-Yadkori et al., 2012). Separability assumption posits that there is a positive constant gap between the rewards of the best and the second best actions. In our framework, the parameter θ belongs to a continuous set in $\mathbb{R}^{d+1}$ and therefore the separability assumption cannot be enforced because by choosing θ arbitrary close to θ_0, one can obtain suboptimal (but arbitrary close to optimal) reward. However, the regret of our policy scales only logarithmically with T. Here, we contrast our logarithmic regret bound with the folklore lower bound $\Omega(\sqrt{T})$, to build further insight on our results.

Uninformative prices and $\Omega(\sqrt{T})$ lower-bound. We focus on (Broder and Rusmevichientong, 2012) which has a close framework to ours in that it considers a dynamic pricing policy from purchasing decisions and presents a pricing policy based on maximum likelihood estimation with regret $O(\sqrt{T})$. Adopting their notation, it is assumed that market values v_t are independent and identically distributed random variables coming from a distribution function that belongs to some family parametrized by $\mathbf{z}$. Denote by $d(p;\mathbf{z})$ the demand curve. This curve determines the probability of a purchase at a given price, i.e., $d(p;\mathbf{z}) = \mathbb{P}_\mathbf{z}(v_t \geq p)$. (Broder and Rusmevichientong, 2012) show that the worst-case regret of any pricing policy must be at least $\Omega(\sqrt{T})$ (see Theorem 3.1 therein). The bound is proved by considering a specific family of demand curves $d(p;\mathbf{z})$, such that all demand curves in this family intersect at a common price. Further, the common price is the optimal price for a specific choice of parameter $\mathbf{z}_0$, i.e, $p^*(\mathbf{z}_0)$.[4] Therefore, the price $p^*(\mathbf{z}_0)$ is "uninformative" since no policy can gain information about the demand parameter $\mathbf{z}$, while pricing $p^*(\mathbf{z}_0)$. The idea behind the derived lower bound for the worst-case regret is that for a policy to learn the underlying demand curve fast enough, it must necessarily choose prices that are away from (the uninformative) price $p^*(\mathbf{z}_0)$ and this leads to a large regret when the true demand curve is indeed $\mathbf{z}_0$.

Intuition behind our results. In contrast to the previous case, for our framework there is no such uninformative price. First, note that the for a choice model with parameters $\theta_0 = (\theta_0, \alpha_0)$, the demand curve at time t is given by

$$d_t(p;\theta_0) = 1 - F(p - \theta_0 \cdot x_t),$$

For $n \geq 1$, we define the aggregate demand function up to time n as $d_1^n = (d_1, d_2, \ldots, d_n)$. In the following, we argue that under our setting, there is no uninformative price. For any

4. Specifically, they consider $d(p;\mathbf{z}) = 0.5 + \mathbf{z} - \mathbf{z}p$. Hence $d(1;\mathbf{z}) = 1$, for all $\mathbf{z}$ and it is shown that $p^*(\mathbf{z}_0) = 1$ for $\mathbf{z}_0 = 0.5$.

price p and any θ_1, θ_2, we have

$$\frac{1}{n}\|d_1^n(p,\theta_1) - d_1^n(p,\theta_2)\|_2^2 = \frac{c^2}{n}\sum_{\ell=1}^{n}((\theta_1-\theta_2)\cdot x_t)^2$$
$$= \frac{c^2}{n}\|X(\theta_1-\theta_2)\|_2^2\,,$$

where X is the matrix with rows x_ℓ, for $1 \le \ell \le t$. We also used the fact that $f(z) \ge c > 0$ for some constant c because F is strictly increasing by Assumption 1. As we show in Appendix A, for $n \ge c_0 s_0 \log d$ (with c_0 a proper constant), X satisfy a so-called "restricted eigenvalue", by which we have

$$\frac{1}{n}\|X(\theta_1-\theta_2)\|_2^2 \ge \frac{C_{\min}}{2}\|\theta_1-\theta_2\|_2^2\,. \tag{15}$$

Therefore, for any fixed price p, if we vary the demand parameters θ_1 to some other value θ_2, then the aggregate demand at price p also changes by an amount proportional to $\|\theta_1-\theta_2\|_2$. Hence, any price in this setting is informative about the model parameters.

To build further insight, let us consider a more general choice model, where the utility of the customer from buying a product with feature vectors x_t at price p is given by

$$u(x_t) = \theta_0 \cdot x_t - \beta_0 p + z_t\,, \tag{16}$$

where θ_0, β_0 are unknown model parameters and z_t is the noise term. The customer buys the product iff $u(x_t) \ge 0$. Note that the model we studied in this paper (see Equation (3)) is a special case when the price sensitivity β_0 is known and hence can be normalized to 1. We next argue that in case of unknown β_0, the uninformative prices do exist and hence the $\Omega(\sqrt{T})$ is still in place.

To see this, let $\theta_{0,i} = 0$ for $2 \le i \le d$. Fix arbitrary a, and set $\beta_0 = g(a) - a + \theta_{0,1}$. Also, let $x_{t,1} = 1$, for all t. Then, the demand curves will be unaltered over time and are given by

$$d_t(p,\theta) = 1 - F(\beta_0 p - \theta_{0,1}) = 1 - F\left((g(a) - a + \theta_{0,1})p - \theta_{0,1}\right)\,.$$

We next verify that $p^* = 1$ is the optimal price for the specific choice of $\theta_{0,1} = a$. In this case, the expected revenue form a posted price p is given by

$$p \times d_t(p,\theta) = p\left(1 - F\left(g(a)p - a\right)\right)\,. \tag{17}$$

By definition of the pricing function g, given by (7), it is easy to see that $p^* = 1$ sets the gradient of the expected revenue to zero.

Next note that all the demand curves, regardless of value $\theta_{0,1}$, intersect at $p^* = 1$ (they all have the value $1 - F(g(a) - a)$ at this price). Therefore, p^* is an uninformative price and no policy can gain information about $\theta_{0,1}$ by pricing at p^*. However, when $\theta_{0,1} = a$, choosing prices that are away from this informative price leads to a large regret. Prices that are close to p^* does not have any information gain, and contrasting these two points, it can be shown that the worst case regret is of order $\Omega(\sqrt{T})$. A formal proof follows the same lines of the proof of (Broder and Rusmevichientong, 2012, Theorem 3.1) and is omitted.

Finally, it is worth noting that the rate of learning demand parameter θ_0 is chiefly derived by three factors:

- Non-smoothness of distribution function F, as it controls the amount of information obtained about $x_t \cdot \theta_0$ at each t.
- The rate by which the feature vectors x_t span the parameter space. This is controlled through the minimum eigenvalue of Σ, i.e., $C_{\min}$. If $C_{\min}$ is small, the randomly generated features are relatively aligned and one requires larger sample size to estimate θ_0 within specified accuracy.
- Complexity of θ_0. This is captured through the sparsity measure s_0.

Later in Proposition 10, we see how these factors contribute to the learning rate of θ_0.

4.2. Role of $C_{\min}$

In establishing our results, we relied on Assumption 2 which requires the population covariance of features to be positive definite. The lower bound on its eigenvalues, denoted by $C_{\min}$, appears in our regret bound as a factor $1/C_{\min}^2$.

As evident from the proof of Proposition 10, Assumption 1 can be replaced by the weaker *restricted eigenvalue condition* (Bühlmann and van de Geer, 2011; Candes and Tao, 2007), which is a common assumption in high-dimensional statistical learning. While assumption $C_{\min} > 0$ allows for a fast learning rate of model parameters and a regret bound $O(\log T)$, RMLP policy can still provably achieve regret $O(\sqrt{T})$, even when $C_{\min} = 0$.

Theorem 4 *Suppose that product feature vectors are generated independently from a probability distribution $\mathbb{P}_X$ with a bounded support $\mathcal{X} \in \mathbb{R}^d$. Also, assume that the choice model parameters $\theta_0 \in \Omega$, where the set Ω is given by* (5)*. Under Assumption 1, the regret of RMLP policy is of $O(W^3\sqrt{(\log d)T})$.*

Proof of Theorem 4 is given in Section 8.2. We remark that the regret bound in Theorem 4 is independent of the sparsity s_0 and scales logarithmically with the feature dimension d. From the technical point of view, this is due to the fact that in our analysis instead of bounding the estimation errors $\|\theta_0 - \widehat{\theta}^k\|_2$, we directly bound the prediction errors $\|X^{(k-1)}(\theta_0 - \widehat{\theta}^k)\|_2$, where $X^{(k-1)}$ is the matrix obtained by putting the feature vectors in episode $(k-1)$ as its rows. We then connect the total regret in each episode to the corresponding prediction error. By this analysis, we are able to remove the requirement $C_{\min} > 0$ and obtain a regret that is independent of the sparsity s_0, at the cost of getting $O(\sqrt{T})$ regret bound.

5. Lower Bound on Regret

As discussed in Section 2.2, if the true parameter θ_0 is known, the optimal policy (in terms of expected revenue) is the one that sets prices as $p_t = g(x_t \cdot \theta_0)$. Let $\mathcal{H}_t = \{x_1, x_2, \ldots, x_t, z_1, z_2, \ldots, z_t\}$ denote the history set up to time t, and recall that Ω denotes the set of feasible parameters, i.e., $\Omega = \{\theta \in \mathbb{R}^d : \|\theta\|_0 \le s_0\,,\ \|\theta\|_1 \le W\}$. We consider the following set of policies, Π:

$$\Pi = \Big\{\pi : \pi(p_t) = g(x_t \cdot \theta_t),\ \text{for some } \theta_t \in \Omega,\ \text{such that } \theta_t \text{ is } \mathcal{H}_{t-1}\text{-measurable}\Big\}\,. \tag{18}$$

Here $\pi(p_t)$ denotes the price posted by policy π at time t.

We provide a lower bound on the achievable regret by any policy in set Π. Indeed this lower bound applies to an oracle who fully observes the market values after the price is either accepted or rejected. Compared to our setting, where the seller observes only the binary feedbacks (purchase/no purchase), this oracle appears exceedingly powerful at first sight but surprisingly, the derived lower bound matches the regret of our dynamic policy, up to a logarithmic factor.

Theorem 5 *Consider linear model* (2) *where the market values $v(x_t)$, $1 \le t \le T$, are fully observed. We further assume that market value noises are generated as $z_t \sim \mathsf{N}(0, \sigma^2)$. Let Π be the set of policies given by* (18)*. Then, there exists constant $C' > 0$ (depending on W and σ), such that the following holds true for all $T \in \mathbb{N}$.*

$$\min_{\pi \in \Pi} \mathsf{Regret}_\pi(T) \ge C' \left\{ s_0 \log\left(\frac{T}{s_0}\right) + \min\left[\frac{T}{s_0}, s_0 \log\left(\frac{d}{s_0}\right)\right]\right\}. \tag{19}$$

In the following we give an outline for the proof of Theorem 5, summarizing its main steps and defer the complete proof to Section 8.3.

1. We derive a lower bound for regret in terms of the minimax estimation error. Specifically, for $t \in \mathbb{N}$, let

$$\mathsf{reg}_t \equiv p_t^* \mathbb{I}(v_t \ge p_t^*) - p_t \mathbb{I}(v_t \ge p_t) \tag{20}$$

be the regret at period t. We show that

$$\max_{\theta_0 \in \Omega} \mathbb{E}(\mathsf{reg}_t) \ge c \max_{\theta_0 \in \Omega} \mathbb{E}\{\min(\|\theta_t - \theta_0\|_2^2, C)\}, \tag{21}$$

for some constants $c, C > 0$.

2. Let $\theta_1^T = (\theta_t)_{t=1}^T$ and define $\mathsf{d}(\theta_1^T, \theta) \equiv \sum_{t=1}^T \min(\|\theta_t - \theta\|_2^2, C)$. We use a standard argument (Le Cam's method) that relates the minimax ℓ_2 risk, $\min_{\theta_1^T} \max_{\theta_0 \in \Omega} \mathbb{E}[\mathsf{d}(\theta_1^T, \theta_0)]$, in terms of the error in multi-way hypothesis problem (Tsybakov, 2008). We first construct a maximal set of points in Ω, such that minimum pairwise distances among them is at least δ. (Such set is usually referred to as a δ-packing in the literature). Here δ is a free parameter to be determined in the proof. We then use a standard reduction to show that any estimator with small minimax risk should necessarily solve a hypothesis testing problem over the packing set, with small error probability. More specifically, suppose that nature chooses one point from the packing set uniformly at random and conditional on nature's choice of the parameter vector, say θ_0, the market value are generated according to $\langle x_t, \theta_0 \rangle + z_t$ with $z_t \sim \mathsf{N}(0, \sigma^2)$. The problem is reduced to lower bounding the error probability in distinguishing θ_0 among the candidates in the packing set using the observed market values.

3. We apply Fano's inequality from information theory to lower bound the probability of error (Tsybakov, 2008). The Fano's bound involves the logarithm of the cardinality of the δ-packing set as well as the mutual information between the observations (market values) and the random parameter vector θ_0 chosen uniformly at random from the packing set. Le Cam's method is used to derive minimal risk lower bound for an estimator $\widehat{\theta}$, while here we have a sequence of estimators and need to adjust the Le Cam's method to get the lower bound for $\mathsf{d}(\theta_1^T, \theta_0)$.

6. Nonlinear Valuation Function

In previous sections, we focused exclusively on linear valuation function given by Eq (2). Here, we extend our results and assume that the market valuations are modeled by a *nonlinear* function that depends on products' features and an independent noise term. Specifically, the market value of a product with feature vector x_t is given by

$$v(x_t) = \psi(\theta_0 \cdot \phi(x_t) + z_t), \tag{22}$$

where the original features x_t are transformed by a feature mapping $\phi : \mathbb{R}^d \mapsto \mathbb{R}^d$, and function $\psi : \mathbb{R} \mapsto \mathbb{R}$ is a general function that is log-concave and strictly increasing. Important examples of this model include log-log model ($\psi(x) = e^x$, $\phi(x) = \ln(x)$), semi-log model ($\psi(x) = e^x$, $\phi(x) = x$), and logistic model ($\psi(x) = e^x/(1+e^x)$, $\psi(x) = x$).

Model (22) allows us to capture correlations and non-linear dependencies on the features. We next state our assumption on the feature mapping ϕ and then discuss our dynamic pricing policy and its regret bound for the general setting (22).

Assumption 6 *Let p_X be an (unknown) distribution from which the original features x_t are sampled independently. Suppose that the feature mapping ϕ has continuous derivative and denote by $\Sigma_\phi \equiv \mathbb{E}(\phi(x) \cdot \phi(x)^\mathsf{T})$ the second moment matrix of $\phi(x)$ under $\mathbb{P}_X$. We assume that there exist constants $C_{\min}$ and $C_{\max}$ such that for every eigenvalue σ of Σ_ϕ, we have $0 < C_{\min} \le \sigma < C_{\max} < \infty$.*

Invoking Assumption 1, $\mathbb{P}_X$ has a bounded support $\mathcal{X}$ and since ϕ has continuous derivative, it is Lipschitz on $\mathcal{X}$ and hence the image of $\mathcal{X}$ under ϕ remains bounded. Therefore, the new features $\phi(x_t)$ are also sampled independently from a bounded set. The condition on Σ_ϕ is analogous to that on Σ, as required by Assumption 2 for the linear setting.

Based on feature mapping ϕ, validity of Assumption 6 may depend on all moments of distribution $\mathbb{P}_X$. We provide an alternative to this assumption, which only depends on feature mapping ϕ and the second moment of $\mathbb{P}_X$. In stating the assumption, we use the notation $\mathcal{D}_\phi$ to denote the derivative matrix of a feature mapping ϕ. Precisely, for $\phi = (\phi_1, \ldots, \phi_d)$, with ϕ_i real-valued function defined on $\mathbb{R}^d$, we write $\mathcal{D}_\phi = (\partial \phi_i / \partial x_j)_{1 \le i \le j \le d}$.

Assumption 7 ***(alternative to Assumption 6)*** *Suppose that feature mapping ϕ has continuous derivative and its derivative $\mathcal{D}_\phi(x)$ is full-rank for almost all x. In addition, there exist constants $C_{\min}$ and $C_{\max}$ such that for every eigenvalue σ of covariance Σ, we have $0 < C_{\min} \le \sigma < C_{\max} < \infty$.*

Recall that the noise terms $\{z_t\}_{t \ge 1}$ are drawn independently and identically from a distribution with cumulative function F and density $f(x)$. Let $\lambda(v) = f(v)/(1 - F(v))$ be the hazard rate function for distribution F. For a log-concave function ψ, we define

$$g_\psi^{-1}(v) \equiv v - \lambda^{-1}\Big(\frac{\psi'(v)}{\psi(v)}\Big). \tag{23}$$

Input: (at time 0) function g, regularizations λ_k, W (bound on $\|\theta_0\|_1$)
Input: (arrives over time) covariate vectors $\{\tilde{x}_t = \phi(x_t)\}_{t\in\mathbb{N}}$
Output: prices $\{p_t\}_{t\in\mathbb{N}}$

1: $\tau_1 \leftarrow 1$, $p_1 \leftarrow 0$, $\widehat{\theta}^1 \leftarrow 0$
2: **for** each episode $k = 2, 3, \ldots$ **do**
3: Set the length of k-th episode: $\tau_k \leftarrow 2^{k-1}$.
4: Update the model parameter estimate $\widehat{\theta}^k$ using the regularized ML estimator obtained
from observations in the previous episode:

$$\widehat{\theta}^k = \underset{\|\theta\|_1 \le W}{\arg\min} \ \{\mathcal{L}(\theta) + \lambda_k \|\theta\|_1\} \ , \tag{24}$$

where $\mathcal{L}(\theta)$ is given by:

$$\mathcal{L}(\theta) = -\frac{1}{\tau_{k-1}} \sum_{t=\tau_{k-1}}^{\tau_k - 1} \Big\{ \mathbb{I}(y_t = 1) \log(1 - F(\psi^{-1}(p_t) - \theta \cdot \tilde{x}_t)) + \mathbb{I}(y_t = -1) \log(F(\psi^{-1}(p_t) - \theta \cdot \tilde{x}_t)) \Big\} . \tag{25}$$

5: For each period t during the k-th episode, set

$$p_t \leftarrow \psi(g_\psi(\widehat{\theta}^k \cdot \tilde{x}_t)) . \tag{26}$$

Algorithm 2: RMLP Policy for dynamic pricing under the nonlinear setting

Note that $\psi'(v)/\psi(v) = \log' \psi(v)$ and since ψ is log-concave, this term is decreasing. Further, since $1 - F$ is log-concave then its hazard rate λ is increasing (See proof of Lemma C.1.) Combining these observations, we have that $-\lambda^{-1}(\psi'(v)/\psi(v))$ is increasing. Consequently,

- Right-hand side of (23) is strictly increasing and hence, g_ψ^{-1} is well-defined.
- We have $(g_\psi^{-1})'(v) \ge 1$, for all v. This implies that $0 < g'_\psi(v) \le 1$, for all v.

It is worth noting that for $\psi(v) = v$ (linear model), we have $g_\psi = g$, where g is defined by (7). Our pricing policy for the nonlinear model is conceptually similar to the linear setting: The policy runs in an episodic manner. During episode k, the prices are set as $p_t = \psi(g_\psi(\widehat{\theta}^k \cdot x_t))$, where $\widehat{\theta}^k$ denotes the estimate of the true parameter vector θ_0, using a regularized maximum-likelihood estimator applied to observations in the *previous* episode, and $\tilde{x}_t = \phi(x_t)$.

We describe our (modified) RMLP policy in Algorithm 2. There a few differences between Algorithm 2 and Algorithm 1: Firstly, the features x_t are replaced by $\tilde{x}_t = \phi(x_t)$. Secondly, in the regularized estimator, prices p_t are replaced by $\psi^{-1}(p_t)$. Thirdly, in the last step of algorithm prices are set as $\psi(g_\psi(\widehat{\theta}^k \cdot x_t))$, with g_ψ defined by Equation (23).

Our next theorem bounds the regret of our pricing policy (Algorithm 2).

Theorem 8 *Let ψ be log-concave and strictly increasing. Suppose that Assumptions 1 and 6 (or its alternative, Assumption 7) hold. Then, regret of the RMLP policy described as Algorithm 2 is of $O(s_0 \log d \cdot \log T)$.*

Proof of Theorem 8 is given in Appendix 8.4. Here, we summarize its key ingredients.

1. By increasing property of ψ, a sale occurs at period t when $z_t \ge \psi^{-1}(p_t) - \theta_0 \cdot \tilde{x}_t$. Hence, the log-likelihood estimator for this setting reads as (25). By virtue of Assumption 6 (or its alternative, Assumption 7) we get a similar estimation error for the regularized estimator to the one in Proposition 10.

2. Similar to our derivation for linear setting, we show that the optimal pricing policy that knows θ_0 in advance is given by $p_t^* = \psi(g_\psi(\theta_0 \cdot \tilde{x}_t))$, where g_ψ is defined based on Equation (23).

3. The difference between the posted price and the optimal price can be bounded as $p_t - p_t^* = \psi(g_\psi(\widehat{\theta}^k \cdot \tilde{x}_t)) - \psi(g_\psi(\theta_0 \cdot \tilde{x}_t)) \le L|\tilde{x}_t \cdot (\widehat{\theta}^k - \theta_0)|$, for a constant $L > 0$. This bound is similar to the corresponding bound for the linear setting, and following the same lines of our regret analysis for that case, we get $R(T) = O(s_0 \log d \cdot \log T)$.

7. Knowledge of Market Noise Distribution

The proposed RMLP policy has assumed that the market noise distribution F is known to the seller. Knowledge of F has been used both in estimating the model parameters θ_0 and in setting the prices p_t. On the other hand, the benchmark policy is also assumed to have access to model parameters and the distribution F. Therefore, the regret bound established in Theorem 3 essentially measures how much the seller loses in revenue due to lack of knowledge of the underlying model parameters. In practice, however, the underlying distribution of valuations is not given and this raises the question of *distribution-independent* pricing policy.

It is worth mentioning that in some applications, although the underlying distribution of valuations is unknown, it belongs to a known class of distributions. For example, lognormal distributions have proved to be a good fit for the distribution of valuations of advertisers in online advertising markets (Edelman et al., 2007; Lahaie and Pennock, 2007; Xiao et al., 2009; Balseiro et al., 2014). In Section 7.1, we consider a setting where the underlying distribution belongs to a known class of log-concave distributions and propose a policy whose regret is $O(\sqrt{T})$. We also argue that no policy can get a better regret bound.

7.1. Unknown Distribution from a Known Class

Suppose that the market noises are generated from a log-concave distribution $F_{m,\sigma}$ (e.g., Log-normal), with unknown mean m and unknown variance σ^2. Without loss of generality, we can assume that $m = 0$; otherwise, in the valuation model (2), m can be absorbed in the features as the intercept term. We next explain how the RMLP policy can be adapted to this case.

Input: Pricing function g (corresponding to $F_{0,1}$), regularizations λ_k, W
Input: (arrives over time) covariate vectors $\{x_t\}_{t\in\mathbb{N}}$
Output: prices $\{p_t\}_{t\in\mathbb{N}}$

1: $\tau_1 \leftarrow 1$, $p_1 \leftarrow 0$, $\widehat{\theta}^1 \leftarrow 0$
2: **for** each episode $k = 2, 3, \dots$ **do**
3: Set the length of k-th episode: $\tau_k \leftarrow 2^{k-1}$.
4: Update the model parameter estimate $\widehat{\mu}^k$ using the regularized ML estimator:

$$(\widehat{\mu}^k, \widehat{\beta}^k) = \underset{\|(\mu/\beta,\beta)\|_1 \le W}{\arg\min} \{\mathcal{L}(\mu,\beta) + \lambda_k \|\mu\|_1\}\,, \tag{28}$$

where $\mathcal{L}(\mu,\beta)$ is given by:

$$\mathcal{L}(\mu,\beta) = -\frac{1}{\tau_{k-1}} \sum_{t=\tau_{k-1}}^{\tau_k-1} \Big\{ \mathbb{I}(y_t = 1)\log\left(1 - F(\beta p_t - \mu\cdot x_t)\right) + \mathbb{I}(y_t = -1)\log\left(F(\beta p_t - \mu\cdot x_t)\right) \Big\}. \tag{29}$$

5: For each period t during the k-th episode, set

$$p_t \leftarrow \frac{1}{\widehat{\beta}^k} g(\widehat{\mu}^k \cdot x_t)\,. \tag{30}$$

Algorithm 3: RMLP-2 policy for dynamic pricing

Define $\beta_0 = 1/\sigma$ and consider the transformation $\tilde{v}_t = \beta_0 v_t$, $\mu_0 = \beta_0\theta_0$, $\tilde{z}_t = \beta_0 z_t$. Then, the valuation model (2) can be written as

$$\tilde{v}_t = x_t \cdot \mu_0 + \tilde{z}_t\,, \tag{27}$$

where $\tilde{z}_t$ are drawn from $F_{0,1}$. To lighten the notation, we use the shorthand $F \equiv F_{0,1}$. The response variables y_t are then given by $y_t = \mathbb{I}(\tilde{v}_t \ge \beta_0 p_t)$.

We propose a variant of RMLP policy, called RMLP-2 for this case. The RMLP-2 policy is very similar to RMLP except that the maximum log-likelihood is formed to estimate the parameter vector (μ_0, β_0) jointly. Likewise, the regularizations are set as $\lambda_k = 4u_F\sqrt{(\log d)/\tau_{k-1}}$, where u_F is defined based on the distribution $F \equiv F_{0,1}$ as in Equation (13). The policy then offers the optimal prices based on the current parameter estimates at the time. Specifically, for episode k, we set $p_t = (1/\widehat{\beta}^k)g(\widehat{\mu}^k \cdot x_t)$, where the pricing function g, given by (7), is defined based on the distribution $F \equiv F_{0,1}$. A formal description of RMLP-2 is given in Algorithm 3.

Our next result bounds the regret of RMLP-2.

Theorem 9 *Consider the valuation model* (2)*, where noises z_t are generated from a distribution $F_{m,\sigma}$, with unknown mean m and variance σ^2. Assuming that distribution $F_{m,\sigma}$ satisfies Assumption 1, the regret of RMLP-2 policy is of $O(\sqrt{(\log d)T})$. Further, regret of any pricing policy in this case is $\Omega(\sqrt{T})$.*

We refer to Section 8.5 for the proof of Theorem 9. As discussed in the proof, the lower bound $\Omega(\sqrt{T})$ applies to this case due to the existence of non-informative prices; See also Section 4.1.

8. Proof of Theorems

8.1. Proof of Theorem 3

Following step 1 of the proof outline mentioned in Section 4, we consider the problem of estimating θ_0 based on observations from previous episode. Before we proceed, let us emphasize once again that the way RMLP is designed, posted prices at each episode are statistically independent from the market noises in that episode. This can be easily observed because $p_t = g(x_t \cdot \widehat{\theta}^k)$ for t belonging in the k-th episode, and $\widehat{\theta}^k$ is estimated based on the samples in the $(k-1)$-th episode.

We fix $k \ge 1$ and to lighten the notation, we use the indices $1, 2, \dots, n$ to correspond to periods in the k-the episode, i.e., $t = \tau_k, \tau_k + 1, \dots, \tau_{k+1} - 1$.

Using probabilistic model (4), θ_0 is estimated by solving a regularized maximum likelihood (ML) optimization problem. The (normalized) negative log-likelihood function for θ reads as

$$\mathcal{L}(\theta) = -\frac{1}{n}\sum_{t=1}^{n} \Big\{ \mathbb{I}(y_t = 1)\log(1 - F(p_t - \theta\cdot x_t)) + \mathbb{I}(y_t = -1)\log(F(p_t - \theta\cdot x_t)) \Big\}. \tag{31}$$

Parameter θ is estimated as the solution of the following program:

$$\widehat{\theta} = \underset{\|\theta\|_1 \le W}{\arg\min}\ \mathcal{L}(\theta) + \lambda\|\theta\|_1\,. \tag{32}$$

Define ℓ_F as follows which corresponds to "flatness" of function $\log F$:

$$\ell_F \equiv \inf_{|x|\le 3W} \Big\{ \min\Big\{ -\log'' F(x), -\log''(1 - F(x)) \Big\} \Big\}\,. \tag{33}$$

By Assumption 1, the log-concavity property of F and $1 - F$, we have $\ell_F > 0$.

The next theorem upper bounds the estimation error of the proposed regularized estimator.

Proposition 10 (Estimation Error) *Consider linear model* (2) *with $\theta_0 \in \Omega$, under Assumptions 1 and 2. Let $\widehat{\theta}$ be the solution of optimization problem* (32) *with $\lambda \ge 4u_F\sqrt{(\log d)/n}$. Then, there exist positive constants c_0 and C such that, for $n \ge c_0 s_0 \log(d)$, the following inequality holds with probability at least $1 - 1/d - 2e^{-n/(c_0 s_0)}$:*

$$\|\widehat{\theta} - \theta_0\|_2^2 \le \frac{16 s_0 \lambda^2}{\ell_F^2 C_{\min}^2}\,. \tag{34}$$

We refer to Appendix A for the proof of Proposition 10.

As we see the ℓ_2 estimation error scales linearly with the sparsity level s_0. As s_0 increases, the number of parameters to be estimated becomes larger and this makes the estimation problem harder, leading to worse ℓ_2 bound for a fixed number of samples, n. Further,

choosing $\lambda \sim \sqrt{(\log d)/n}$ (where $\sim$ indicates equality up to a constant factor), our ℓ_2 bound scales logarithmically in the dimension of the demand space, d. This allows to deal with high-dimensional applications and obtain a regret that scales logarithmically in d. Further, the estimation error shrinks as $\sim 1/n$; getting more samples with fixed value of s_0 and d leads to better estimation accuracy. Finally, note that for small values of ℓ_F, the log-likelihood function is very flat and there can be, in principle, vectors θ of log-likelihood value very close to the optimum and nevertheless far from the optimum. In other words, estimation task becomes harder as ℓ_F gets smaller and this is clearly reflected in the derived estimation bound.

We next use Proposition 10 to bound the expected estimation error.

Corollary 11 *Under assumptions of Proposition 10, the following holds true:*

$$\mathbb{E}(\|\widehat{\theta}-\theta_0\|_2^2) \le \frac{16 s_0 \lambda^2}{\ell_F^2 C_{\min}^2} + 4W^2\left(\frac{1}{d} + 2e^{-n/(c_0 s_0)}\right). \tag{35}$$

Proof of Corollary 11 is straightforward and is omitted.

In the next proposition, we improve bound (35) for $n \ge c_1 d$, for a constant $c_1 > 0$. As we will see, the following result is useful to develop sharper upper bound for regret of RMLP policy.

Proposition 12 *Under assumptions of Proposition 10, there exist constants $c, c_1 > 0$, such that for $n \ge c_1 d$, the following holds true:*

$$\mathbb{E}(\|\widehat{\theta}-\theta_0\|_2^2) \le \frac{16(s_0+1)\lambda^2}{\ell_F^2 C_{\min}^2} + 4W^2 e^{-cn^2}. \tag{36}$$

Proposition 12 is proved in Appendix B.

We next establish some useful properties of the virtual valuation function φ and the price function g.

Lemma 13 *If $1-F$ is log-concave, then the virtual valuation function φ is strictly monotone increasing.*

Lemma 14 *If $1-F$ is log-concave, then the price function g satisfies $0 < g'(v) < 1$, for all values of $v \in \mathbb{R}$.*

Proofs of Lemma 13 and 14 are given in Appendix C.1 and C.2, respectively.

Given that $\|\widehat{\theta}^k\|_1 \le W$ and $|x_t \cdot \widehat{\theta}^k| \le W$ for all t, k,

$$p_t = g(x_t \cdot \widehat{\theta}^k) \le 2|x_t \cdot \widehat{\theta}^k| \le 2W, \tag{37}$$

where in the first inequality we used the fact that $\varphi(v)$ is increasing as per Lemma 13 and hence $g(v) = v + \varphi^{-1}(-v) \le v + |v| \le 2|v|$. Similarly, we have $p_t^* \le 2W$ for all t.

We are now ready to bound the regret of our policy. For $t \ge 1$, let

$$\mathsf{reg}_t \equiv p_t^* \mathbb{I}(v_t \ge p_t^*) - p_t \mathbb{I}(v_t \ge p_t) \tag{38}$$

be the regret at period t. Further, let $\mathcal{H}_t = \{x_1, x_2, \dots, x_t, z_1, z_2, \dots, z_t\}$ be the history set, up to time t (more precisely, $\mathcal{H}_t$ is the filtration generated by $\{x_1, x_2, \dots, x_t, z_1, z_2, \dots, z_t\}$). We also define $\bar{\mathcal{H}}_t = \mathcal{H}_t \cup \{x_{t+1}\}$ as the filtration obtained after augmenting by the new feature x_{t+1}.

We write

$$\begin{aligned}\mathbb{E}(\mathsf{reg}_t|\bar{\mathcal{H}}_{t-1}) &= \mathbb{E}(p_t^*\mathbb{I}(v_t \ge p_t^*)|\bar{\mathcal{H}}_{t-1}) - \mathbb{E}(p_t\mathbb{I}(v_t \ge p_t)|\bar{\mathcal{H}}_{t-1}) \\ &= p_t^*(1-F(p_t^* - x_t\cdot\theta_0)) - p_t(1-F(p_t - x_t\cdot\theta_0)).\end{aligned} \tag{39}$$

Define $\mathsf{rev}_t(p) \equiv p(1-F(p - x_t\cdot\theta_0))$ as the expected revenue under price p. Note that $p_t^* \in \arg\max \mathsf{rev}_t(p)$ and thus $r_t'(p_t^*) = 0$. By Taylor expansion,

$$\mathsf{rev}_t(p_t) = \mathsf{rev}_t(p_t^*) + \frac{1}{2} r_t''(p)(p_t - p_t^*)^2, \tag{40}$$

for some p between p_t and p_t^*.

We next show that $|r_t''(p)| \le 2(B + WB')$, with $B = \max_v f(v)$, and $B' = \max_v f'(v)$. To see this, we write

$$|r_t''(p)| = |2f(p - x_t\cdot\theta_0) + pf'(p - x_t\cdot\theta_0)| \le 2B + 2WB', \tag{41}$$

where we use the fact that $p_t, p_t^* \le 2W$ and consequently $p \le 2W$. We let $C = 2\max(B, B')$. Then, combining Equations (39), (40), (41), along with 1-Lipschitz property of g gives

$$\begin{aligned}\mathbb{E}(\mathsf{reg}_t|\bar{\mathcal{H}}_{t-1}) &\le (B+WB')(p_t^* - p_t)^2 \\ &\le CW(p_t^* - p_t)^2 \\ &= CW(g(\theta_0\cdot x_t) - g(\widehat{\theta}^k\cdot x_t))^2 \\ &\le CW\,|x_t\cdot(\theta_0 - \widehat{\theta}^k)|^2.\end{aligned} \tag{42}$$

Here, in the second inequality we used the fact that $W \ge 1$. Given that x_t is independent of $\mathcal{H}_{t-1}$, we have

$$\mathbb{E}(\mathsf{reg}_t|\mathcal{H}_{t-1}) \le CW\langle\widehat{\theta}^k - \theta_0, \Sigma(\widehat{\theta}^k - \theta_0)\rangle, \tag{43}$$

where $\Sigma = \mathbb{E}(x_t x_t^{\mathsf{T}})$. Since the maximum eigenvalue of Σ is bounded by $C_{\max}$, we obtain

$$\mathbb{E}(\mathsf{reg}_t) = \mathbb{E}\left(\mathbb{E}(\mathsf{reg}_t|\mathcal{H}_{t-1})\right) \le CC_{\max}W\mathbb{E}(\|\widehat{\theta}^k - \theta_0\|_2^2). \tag{44}$$

Now, since the length of episodes grows exponentially, the number of episodes by period T is logarithmic in T. Specifically, T belongs to episode $K = \lfloor \log T \rfloor + 1$. Hence,

$$\mathsf{Regret}(T) = \sum_{k=1}^{K} \mathsf{Regret}(k\text{th Episode}). \tag{45}$$

We bound the total regret over each episode by considering three separate cases:

- $2^{k-2} \le c_0 s_0 \log d$: Here, c_0 is the constant in the statement of Proposition 10. In this case, episodes are not large enough to estimate θ_0 accurately enough, and thus we use a naive bound on regret. Clearly, by (37), we have $\mathbb{E}(\mathsf{reg}_t) \le p_t^* \le 2W$. Since the length of k th episode is $2^{k-1} \le 2c_0 s_0 \log d$, the total regret incurred during episode k is at most $4c_0 W s_0 \log d$.

- $c_0 s_0 \log d \le 2^{k-2} \le c_1 d$: Here, c_1 is the constant in the statement of Proposition 12. Continuing from Equation (44) and applying Corollary 11 to episode k, we obtain

$$\begin{aligned}
\mathsf{Regret}(k\text{th Episode}) &= \sum_{t=\tau_k}^{\tau_{k+1}-1} \mathbb{E}(\mathsf{reg}_t) \\
&\le C C_{\max} W \sum_{t=\tau_k}^{\tau_{k+1}-1} \mathbb{E}(\|\widehat{\theta}^k - \theta_0\|_2^2) \\
&\le C C_{\max} W \left\{ \frac{16 s_0 \lambda_k^2}{\ell_F^2 C_{\min}^2} \tau_k + 4W^2 \left(\frac{\tau_k}{d} + 2\tau_k e^{-\tau_{k-1}/(c_0 s_0)} \right) \right\} \\
&\le C C_{\max} W \left\{ \left(\frac{16 u_F}{\ell_F C_{\min}} \right)^2 2 s_0 \log d + 8W^2 \left(2c_1 + 2\tau_{k-1} e^{-\frac{\tau_{k-1}}{c_0 s_0}} \right) \right\},
\end{aligned} \tag{46}$$

  where in the last step we used $\tau_k = 2\tau_{k-1}$ and $\tau_k = 2^{k-1} \le 2c_1 d$. Therefore, in this case

$$\mathsf{Regret}(k\text{th Episode}) \le \frac{C' W}{C_{\min}^2} s_0 \log d\,, \tag{47}$$

  where C' hides various constants in the right-hand side of (46).

- $c_1 d < 2^{k-2}$: Continuing from Equation (44) and applying Proposition 12 to episode k, we obtain

$$\begin{aligned}
\mathsf{Regret}(k\text{th Episode}) &= \sum_{t=\tau_k}^{\tau_{k+1}-1} \mathbb{E}(\mathsf{reg}_t) \\
&\le C C_{\max} W \sum_{t=\tau_k}^{\tau_{k+1}-1} \mathbb{E}(\|\widehat{\theta}^k - \theta_0\|_2^2) \\
&\le C C_{\max} W \left\{ \frac{16(s_0+1)\lambda_k^2}{\ell_F^2 C_{\min}^2} \tau_k + 4\tau_k W^2 e^{-c\tau_{k-1}^2} \right\} \\
&\le C C_{\max} W \left\{ \left(\frac{16 u_F}{\ell_F C_{\min}} \right)^2 2(s_0+1) \log d + 8W^2 \tau_{k-1} e^{-c\tau_{k-1}^2} \right\},
\end{aligned} \tag{48}$$

  Therefore, in this case

$$\mathsf{Regret}(k\text{th Episode}) \le \frac{C' W}{C_{\min}^2} s_0 \log d\,, \tag{49}$$

where C' hides various constants in the right-hand side of (48).

Combining the above three cases into Equation (45), we get

$$\mathsf{Regret}(T) \le K \frac{C' W}{C_{\min}^2} s_0 \log d = O\Big(\frac{W}{C_{\min}^2} s_0 \log d \cdot \log T \Big)\,, \tag{50}$$

which concludes the proof.

8.2. Proof of Theorem 4

By using Equation (43), we have

$$\mathbb{E}(\mathsf{reg}_t) \le \frac{C}{2} \mathbb{E}\left(\left\langle \widehat{\theta}^k - \theta_0, \Sigma(\widehat{\theta}^k - \theta_0) \right\rangle \right)\,, \tag{51}$$

with $\Sigma = \mathbb{E}(x_t x_t^{\mathsf{T}})$ and $C = 2(B + WB')$, where $B = \max_v f(v)$ and $B' = \max_v f'(v)$. Therefore, letting $K = \lfloor \log T \rfloor + 1$, we get

$$\mathsf{Regret}(T) = \sum_{k=1}^{K} \mathsf{Regret}(k\text{th Episode}) \le \frac{C}{2} \sum_{k=1}^{K} \mathbb{E}\left(\left\langle \widehat{\theta}^k - \theta_0, \Sigma(\widehat{\theta}^k - \theta_0) \right\rangle \right) \tau_k\,. \tag{52}$$

We next bound the right-hand side of the above bound. Let $X^{(k)} \in \mathbb{R}^{\tau_k \times d}$ be the matrix obtained by stacking feature vectors in episode k as rows. By applying bound (91) to samples in episode $(k-1)$, we get that with probability at least $1 - 1/d$,

$$\frac{2\ell_F}{\tau_{k-1}} \left\| X^{(k-1)}(\theta_0 - \widehat{\theta}^k) \right\|^2 + 2\lambda_k \|\widehat{\theta}^k\|_1 \le \lambda_k \|\widehat{\theta}^k - \theta_0\|_1 + 2\lambda_k \|\theta_0\|_1\,. \tag{53}$$

Hence,

$$\frac{2\ell_F}{\tau_{k-1}} \left\| X^{(k-1)}(\theta_0 - \widehat{\theta}^k) \right\|^2 \le \lambda_k \|\widehat{\theta}^k - \theta_0\|_1 + 2\lambda_k \|\theta_0\|_1 - 2\lambda_k \|\widehat{\theta}^k\|_1 \le 3\lambda_k \|\widehat{\theta}^k - \theta_0\|_1\,. \tag{54}$$

For $k \ge 1$, let $S^{(k)} \in \mathbb{R}^{d \times d}$ be the empirical second moment of $X^{(k)}$, and define $E^{(k)} = \Sigma - S^{(k)}$. Then,

$$\left\langle \widehat{\theta}^k - \theta_0, \Sigma(\widehat{\theta}^k - \theta_0) \right\rangle = \left\langle \widehat{\theta}^k - \theta_0, S^{(k-1)}(\widehat{\theta}^k - \theta_0) \right\rangle + \left\langle \widehat{\theta}^k - \theta_0, E^{(k-1)}(\widehat{\theta}^k - \theta_0) \right\rangle\,. \tag{55}$$

The first term is bounded using Equation (54) as follows:

$$\left\langle \widehat{\theta}^k - \theta_0, S^{(k-1)}(\widehat{\theta}^k - \theta_0) \right\rangle = \frac{1}{\tau_{k-1}} \left\| X^{(k-1)}(\theta_0 - \widehat{\theta}^k) \right\|^2 \le \frac{3\lambda_k}{2\ell_F} \|\widehat{\theta}^k - \theta_0\|_1 \le \frac{3W}{\ell_F} \lambda_k\,, \tag{56}$$

with probability at least $1 - 1/d$.

The second term can be bounded by virtue of the following lemma, whose proof is deferred to Appendix C.8

Lemma 15 *For any $k \geq 1$ and any vector $v \in \mathbb{R}^d$, we have*

$$\langle v, E^{(k)} v \rangle \leq 3\sqrt{\frac{\log d}{\tau_k}}\, \|v\|_1^2\,,$$

with probability at least $1 - 8/d^2$.

By Lemma 15, we have

$$\left\langle \widehat{\theta}^k - \theta_0, E^{(k-1)}(\widehat{\theta}^k - \theta_0) \right\rangle \leq 12\sqrt{\frac{\log d}{\tau_{k-1}}} W^2\,, \tag{57}$$

with probability at least $1 - 8/d^2$.

Combining Equations (56) and (57), with probability at least $1 - 9/d$ we have

$$\left\langle \widehat{\theta}^k - \theta_0, \Sigma(\widehat{\theta}^k - \theta_0) \right\rangle \leq \frac{3W}{\ell_F}\lambda_k + 12\sqrt{\frac{\log d}{\tau_{k-1}}} W^2 = 12\left(\frac{W u_F}{\ell_F} + W^2\right)\sqrt{\frac{\log d}{\tau_{k-1}}}\,. \tag{58}$$

Following a similar argument as in Section 8.1 (see Equation (45) and onwards) we have that the following holds for a suitable constant $C > 0$:

$$\begin{aligned}\mathsf{Regret}(T) &\leq CW^3 \sum_{k=2}^{K} \sqrt{\frac{\log d}{\tau_{k-1}}} \cdot \tau_k \\ &\leq \sqrt{2} CW^3 \sum_{k=2}^{K} \sqrt{(\log d)\tau_{k-1}} = O(W^3\sqrt{(\log d)T})\,,\end{aligned}$$

which completes the proof.

8.3. Proof of Theorem 5

The regret benchmark (9) is defined as the maximum gap between a policy and the oracle policy over different $\theta_0 \in \Omega$ and $p_X \in Q(\mathcal{X})$. Without loss of generality, we assume $\mathcal{X} = [-1, 1]^d$. In order to obtain a lower bound on the regret, it suffices to consider a specific distribution in $Q(\mathcal{X})$. We consider a distribution p_X that selects coordinates x_i, $1 \leq i \leq d$, uniformly at random from $\{-1, 1\}$ and independent of each other. We further assume $\theta_0 \in \Omega$, with Ω given by (5).

Fix an arbitrary policy π in family Π. By definition of policy set Π, we have $\pi(p_t) = g(x_t \cdot \theta_t)$, for some $\theta_t \in \Omega$, which is $\mathcal{H}_{t-1}$-measurable . Recalling our notation in the proof of Theorem 3, reg_t denotes the regret occurred at step t and by Equations (39), (40), we have

$$\mathbb{E}(\mathsf{reg}_t | \bar{\mathcal{H}}_{t-1}) = \mathsf{rev}_t(p_t^*) - \mathsf{rev}_t(p_t) = -\frac{1}{2}\mathsf{rev}_t''(p)(p_t - p_t^*)^2\,, \tag{59}$$

for some p between p_t and p_t^*.

Our first lemma will be used in lower bounding $\mathbb{E}(\mathsf{reg}_t | \bar{\mathcal{H}}_{t-1})$.

Lemma 16 *There exists a constant $c_1 > 0$ (depending on W and σ) such that, with probability one*[5]*, $r_t''(p_t^*) \leq -c_1$, for all $t \geq 1$. Further, there exists constant $\delta > 0$ (depending on W and σ) such that $r_t''(p) \leq -c_1/4$ for $p \in [p_t^* - \delta, p_t^* + \delta]$, with probability one.*

Proof of Lemma 16 is given in Appendix C.3.

Continuing from Equation (59), we consider two separate cases:

- $|p_t - p_t^*| \leq \delta$: We have $p \in [p_t^* - \delta, p_t^* + \delta]$ and therefore by applying Lemma 16 we obtain

$$\mathbb{E}(\mathsf{reg}_t | \bar{\mathcal{H}}_{t-1}) = \mathsf{rev}_t(p_t^*) - \mathsf{rev}_t(p_t) \geq \frac{c}{8}(p_t - p_t^*)^2\,. \tag{60}$$

- $|p_t - p_t^*| > \delta$: Since function rev_t has only one local maximum, namely p_t^*, the function is increasing before p_t^* and decreasing afterward. Therefore, if $p_t \leq p_t^* - \delta$ then

$$\mathsf{rev}_t(p_t) \leq \mathsf{rev}_t(p_t^* - \delta) = \mathsf{rev}_t(p_t^*) + \frac{1}{2}\mathsf{rev}_t''(p)\delta^2 \leq \mathsf{rev}_t(p_t^*) - \frac{c_1}{8}\delta^2\,, \tag{61}$$

where p is some point in $[p_t^* - \delta, p_t^*]$ and we applied Lemma 16 in the last step.

Similarly, for $p_t \geq p_t^* + \delta$ we obtain

$$\mathsf{rev}_t(p_t) \leq \mathsf{rev}_t(p_t^* + \delta) = \mathsf{rev}_t(p_t^*) + \frac{1}{2}\mathsf{rev}_t''(p)\delta^2 \leq \mathsf{rev}_t(p_t^*) - \frac{c_1}{8}\delta^2\,, \tag{62}$$

where $p \in [p_t^* - \delta, p_t^*]$ this time. Combining these two inequalities, we get that $\mathsf{rev}_t(p_t^*) - \mathsf{rev}_t(p_t) \geq c_1\delta^2/8$, if $|p_t^* - p_t| \geq \delta$.

Writing the bounds in the two cases together, we get

$$\mathbb{E}(\mathsf{reg}_t | \bar{\mathcal{H}}_{t-1}) \geq \mathsf{rev}_t(p_t^*) - \mathsf{rev}_t(p_t) \geq \begin{cases} \frac{c_1}{8}(p_t - p_t^*)^2\,, & \text{if } |p_t - p_t^*| \leq \delta\,, \\ \frac{c_1}{8}\delta^2\,, & \text{if } |p_t - p_t^*| > \delta\,. \end{cases} \tag{63}$$

We proceed by relating the lower bound to the error in estimation θ_0.

$$\begin{aligned}\mathbb{E}(\mathsf{reg}_t | \bar{\mathcal{H}}_{t-1}) &\geq \frac{c_1}{8}\min\left((p_t - p_t^*)^2, \delta^2\right) = \frac{c_1}{8}\min\left((g(x_t \cdot \theta_t) - g(x_t \cdot \theta_0))^2, \delta^2\right) \\ &\geq \frac{c_1}{8}\min\left(c_2^2\,|x_t \cdot (\theta_t - \theta_0)|^2, \delta^2\right),\end{aligned} \tag{64}$$

where we used the fact that by Lemma 14, $g'(v) > c_2$ over the bounded interval $[-W, W]$, for some constant $c_2 > 0$. We recall the definition of history set $\mathcal{H}_t \equiv \bar{\mathcal{H}}_t \backslash \{x_{t+1}\} = \{x_1, x_2, \ldots, x_t, z_1, z_2, \ldots, z_t\}$. Since $\mathcal{H}_t \subseteq \bar{\mathcal{H}}_t$, by iterated law of expectation, we get

$$\mathbb{E}(\mathsf{reg}_t | \mathcal{H}_{t-1}) = \mathbb{E}(\mathbb{E}(\mathsf{reg}_t | \bar{\mathcal{H}}_{t-1}) | \mathcal{H}_{t-1}) \geq \frac{c_1}{8}\mathbb{E}\Big(\min\left(c_2^2 |x_t \cdot (\theta_t - \theta_0)|_2^2, \delta^2\right)\Big| \mathcal{H}_{t-1}\Big)\,. \tag{65}$$

Note that x_t is independent of $\mathcal{H}_{t-1}$ and $\theta_t - \theta_0$ is $\mathcal{H}_{t-1}$-measurable.

We use the following lemma to lower bound the right-hand side of (65).

5. The randomness comes from randomness in prices which in turn comes from randomness in features x_t.

Lemma 17 *Let $x \in \mathbb{R}^d$ be a random vector such that its coordinates are chosen independently and uniformly at random from $\{-1, 1\}$. Further, suppose that $v \in \mathbb{R}^d$ and $\delta > 0$ are deterministic. Then,*

$$\mathbb{E}\Big(\min\big((x \cdot v)^2, \delta^2\big)\Big) \geq 0.1 \min(\|v\|_2^2, \delta^2)\,. \tag{66}$$

Proof of Lemma 17 is given in Appendix C.4.

Applying Lemma 17 to bound (65), we obtain

$$\mathbb{E}(\mathsf{reg}_t|\mathcal{H}_{t-1}) \geq \frac{c_1 c_2^2}{80}\mathbb{E}\Big(\min\Big(\|\theta_t - \theta_0\|_2^2, \delta^2/c_2^2\Big)\Big|\mathcal{H}_{t-1}\Big)\,. \tag{67}$$

Now, taking expectation from both sides with respect to $\mathcal{H}_{t-1}$, we arrive at

$$\mathbb{E}(\mathsf{reg}_t) \geq \frac{c_1 c_2^2}{80}\mathbb{E}\Big(\min\Big(\|\theta_t - \theta_0\|_2^2, \delta^2/c_2^2\Big)\Big)\,. \tag{68}$$

Equation (68) lower bounds the expected regret at each step to the ℓ_2 estimation error.

We continue by establishing a minimax lower bound on ℓ_2-risk of estimation.

Lemma 18 *Consider linear model (2), with $\alpha_0 = 0$, and assume that the market values $v(x_t)$, $1 \leq t \leq T$, are fully observed and the feature vectors are generated according to p_X, described above. We further assume that the noise in market value is generated as $z_t \sim \mathsf{N}(0, \sigma^2)$. For a sequence of estimators θ_t, we let $\theta_1^t = (\theta_1, \theta_2, \ldots, \theta_t)$. Then, for any fixed value $C > 0$, there exists a nonnegative constant $\widetilde{C}$, depending on C, σ, W, such that*

$$\min_{\theta_1^T} \max_{\theta_0 \in \Omega} \sum_{t=1}^T \mathbb{E}\left(\min\left(\|\theta_t - \theta_0\|_2^2, C\right)\Big| x_1, \ldots, x_T\right) \geq \widetilde{C}\Big\{ s_0 \log\Big(\frac{T}{s_0}\Big) + \min\Big[\frac{T}{s_0}, s_0 \log\Big(\frac{d}{s_0}\Big)\Big]\Big\}\,. \tag{69}$$

Proof of Lemma 18 is given in Appendix C.5.

We are now ready to lower bound the regret of any policy in Π.

$$\begin{aligned}\mathsf{Regret}(T) &\geq \max_{\theta_0 \in \Omega} \sum_{t=1}^T \mathbb{E}(\mathsf{reg}_t) \geq \frac{c_1 c_2^2}{80} \sum_{t=1}^T \mathbb{E}\Big(\min\Big(\|\theta_t - \theta_0\|_2^2, \delta^2/c_2^2\Big)\Big) \\ &\geq \widetilde{C}\frac{c_1 c_2^2}{80}\Big\{ s_0 \log\Big(\frac{T}{s_0}\Big) + \min\Big[\frac{T}{s_0}, s_0 \log\Big(\frac{d}{s_0}\Big)\Big]\Big\}\,. \end{aligned} \tag{70}$$

where the last step follows from Lemma 18.

8.4. Proof of Theorem 8

Let $\tilde{x}_t = \phi(x_t)$ denote the transformed features under the feature-map. Also, let $\tilde{p}_t = \psi^{-1}(p_t)$. We first show that Assumption 7 implies Assumption 6, and therefore it suffices to prove the theorem under Assumption 6.

Lemma 19 *Suppose that Assumption 1 hold true. Then, Assumption 7 implies Assumption 6.*

Proof of Lemma 19 is given in Appendix C.6.

By Assumption 1, the support of $\mathbb{P}_X$ is a bounded set $\mathcal{X}$. Given that ϕ has a continuous derivative, it is Lipschitz on the bounded set $\mathcal{X}$ and ergo the image of $\mathcal{X}$ remains bounded under the feature-map ϕ. Putting differently, features $\tilde{x}_t$ are sampled from a bounded set in $\mathbb{R}^d$. Without loss of generality, we assume $\|\tilde{x}_t\|_\infty \leq 1$. Further, as per Assumption 6, the covariance of the underlying distribution Σ_ϕ is positive definite with bounded eigenvalues.

On a different note, since ψ is strictly increasing, a sale occurs at period t when $\theta_0 \cdot \tilde{x}_t + z_t \geq \psi^{-1}(p_t) = \tilde{p}_t$. Therefore the (negative) log-likelihood function for θ reads as

$$\mathcal{L}(\theta) = -\frac{1}{\tau_{k-1}} \sum_{t=\tau_{k-1}}^{\tau_k - 1} \Big\{ \mathbb{I}(y_t = 1)\log(1 - F(\tilde{p}_t - \theta \cdot \tilde{x}_t)) + \mathbb{I}(y_t = -1)\log(F(\tilde{p}_t - \theta \cdot \tilde{x}_t))\Big\}\,.$$

The estimation bound (34) also holds for this setting and the proof goes along the same lines of the proof of Proposition 10, with slight modifications: (i) the features x_t and prices p_t should be replaced by $\tilde{x}_t$ and $\tilde{p}_t$. (ii) Quantity u_F and ℓ_F in the statement of Proposition 10 should be set as $M = (1/3)g_\psi(0) + (2/3)W$. This follows from the bounds below

$$\tilde{p}_t = g_\psi(\tilde{x}_t \cdot \widehat{\theta}^k) \leq g_\psi(0) + |\tilde{x}_t \cdot \widehat{\theta}^k| \leq g_\psi(0) + W\,, \tag{71}$$

where we used the facts that g_ψ is 1-Lipschitz and increasing as explained below Equation (23).

We next characterize the optimal policy when the true parameter θ_0 is known. The expected revenue from a posted price p works out at $p(1 - F(\psi^{-1}(p) - \theta_0 \cdot \tilde{x}_t))$. Writing this in terms of $\tilde{p} = \psi^{-1}(p)$, the first order condition for the optimal price reads as

$$\lambda(\tilde{p}^* - \theta_0 \cdot \tilde{x}_t) \equiv \frac{f(\tilde{p}^* - \theta_0 \cdot \tilde{x}_t)}{1 - F(\tilde{p}^* - \theta_0 \cdot \tilde{x}_t)} = \frac{\psi'(\tilde{p}^*)}{\psi(\tilde{p}^*)}\,, \tag{72}$$

where λ denotes the hazard rate function. Equivalently

$$\theta_0 \cdot \tilde{x}_t = \tilde{p}^* - \lambda^{-1}\Big(\frac{\psi'(\tilde{p}^*)}{\psi(\tilde{p}^*)}\Big)\,. \tag{73}$$

By definition of function g_ψ as per Equation (23), we get $\tilde{p}^* = g_\psi(\theta_0 \cdot \tilde{x}_t)$ and thus $p^* = \psi(g_\psi(\theta_0 \cdot \tilde{x}_t))$.

We are now ready to bound the regret of the algorithm. Similar to Equation (42), we have

$$\begin{aligned}\mathbb{E}(\mathsf{reg}_t|\overline{\mathcal{H}}_{t-1}) \leq \frac{C}{2}(p_t^* - p_t)^2 &= \frac{C}{2}\Big[\psi(g_\psi(\theta_0 \cdot \tilde{x}_t)) - \psi(g_\psi(\widehat{\theta}^k \cdot \tilde{x}_t))\Big]^2 \\ &\leq \frac{C}{2}L(g_\psi(\theta_0 \cdot \tilde{x}_t)) - g_\psi(\widehat{\theta}^k \cdot \tilde{x}_t))^2 \leq \frac{LC}{2}|\tilde{x}_t \cdot (\theta_0 - \widehat{\theta}^k)|^2\,, \end{aligned} \tag{74}$$

where $L \equiv \max_{|v| \leq \psi(M)} |\psi'(v)|$ (since ψ is continuously differentiable, it attains a maximum over a bounded set.) In addition, we used the fact that $g'_\psi(v) \leq 1$ as explained below Equation (23). The inequalities above then follow from the mean-value theorem.

Given that $\tilde{x}_t$ is independent of $\mathcal{H}_{t-1}$, we have

$$\mathbb{E}(\mathsf{reg}_t|\mathcal{H}_{t-1}) \leq \frac{LC}{2}\langle \widehat{\theta}^k - \theta_0, \Sigma_\phi(\widehat{\theta}^k - \theta_0)\rangle\,, \tag{75}$$

where $\Sigma_\phi = \mathbb{E}(x_t x_t^\mathsf{T})$. Using Assumption 6,

$$\mathbb{E}(\mathsf{reg}_t) = \mathbb{E}\left(\mathbb{E}(\mathsf{reg}_t|\mathcal{H}_{t-1})\right) \le \frac{1}{2} LCC_{\max}\mathbb{E}(\|\widehat{\theta}^k - \theta_0\|_2^2)\,. \tag{76}$$

Rest of the proof is similar to proof of Theorem 8.1 (see after Equation (45)).

8.5. Proof of Theorem 9

We consider representation (27) of the valuations. Letting $\tilde{x}_t = (-x_t; p_t)$ and $\eta = (\mu; \beta)$ vectors in $\mathbb{R}^{(d+1)\times 1}$, we can write the log-likelihood loss as:

$$\mathcal{L}(\eta) = -\frac{1}{\tau_{k-1}} \sum_{t=\tau_{k-1}}^{\tau_k - 1} \Big\{ \mathbb{I}(y_t = 1)\log\left(1 - F\left(\tilde{x}_t \cdot \eta\right)\right) + \mathbb{I}(y_t = -1)\log\left(F\left(\tilde{x}_t \cdot \eta\right)\right) \Big\}\,. \tag{77}$$

We let $p^{(k)} = (p_{\tau_{k-1}}, \ldots, p_{\tau_k - 1})^\mathsf{T}$ be the vector of prices posted in episode k. We also construct feature matrix $X^{(k)}$ by putting features x_t that arrive in episode k, as its rows. Matrix $\widetilde{X}^{(k)}$ is constructed likewise by stacking augmented features x_t. Hence, $\widetilde{X}^{(k)} = [-X^{(k)}, p^{(k)}]$.

The next lemma is a technical lemma that will be used later in bounding the regret. We refer the reader to Appendix C.9 for its proof.

Lemma 20 *Assume that the product features $x_t \in \mathbb{R}^d$ are generated independently from an unknown distribution with bounded support in $\mathbb{R}^d$. Then, with the choice of regularization parameters $\lambda_k = 4u_F\sqrt{(\log d)/\tau_{k-1}}$ in the RMLP-2 policy, the following inequalities hold true:*

$$\frac{1}{\tau_{k-1}}\|X^{(k-1)}(\mu_0 - \widehat{\mu}^k)\|^2 \;\le\; \frac{4W}{\ell_F}\lambda_k\,, \tag{78}$$

$$(\beta_0 - \widehat{\beta}^k)^2 \;\le\; \frac{c_W}{\ell_F}\lambda_k\,, \tag{79}$$

where $c_W > 0$ is a constant that depends on W.

We next figure out the clairvoyant policy.

Lemma 21 *Let g be the pricing function corresponding to distribution $F = F_{0,1}$, given by $g(v) = v + \varphi^{-1}(-v)$, where $\varphi(v) = v - (1 - F(v))/f(v)$ is the virtual valuation function. Then, under model (27), the clairvoyant optimal prices are given by*

$$p_t^* = \frac{1}{\beta_0} g(x_t \cdot \mu_0)\,. \tag{80}$$

Proof of Lemma 21 is given in Appendix C.7.

We next bound the regret at other periods of the episode. Let $\bar{\mathcal{H}}_t = \{\tilde{x}_1, \ldots, \tilde{x}_t, \tilde{x}_{t+1}, z_1, \ldots, z_t\}$ be the history set up to time t. Similar to (42), we write

$$\begin{aligned}
\mathbb{E}(\mathsf{reg}_t|\bar{\mathcal{H}}_{t-1}) &\le CW(p_t^* - p_t)^2 \\
&= CW\Big(\frac{1}{\beta_0}g(x_t\cdot\mu_0) - \frac{1}{\widehat{\beta}^k}g(x_t\cdot\widehat{\mu}^k)\Big)^2 \\
&\le \frac{2CW}{\beta_0^2}\Big(g(x_t\cdot\mu_0) - g(x_t\cdot\widehat{\mu}^k)\Big)^2 + 2CW\Big(\frac{1}{\beta_0} - \frac{1}{\widehat{\beta}^k}\Big)^2 g(x_t\cdot\widehat{\mu}^k)^2 \\
&\le \frac{2CW}{\beta_0^2}|x_t\cdot(\mu_0 - \widehat{\mu}^k)|^2 + \frac{8CW^3}{\beta_0^2}(\widehat{\beta}^k - \beta_0)^2\,.
\end{aligned} \tag{81}$$

In the last step, the first term is bounded by using 1-Lipschitz property of g and the second term is bounded by using the following chain of inequalities:

$$\frac{1}{\widehat{\beta}^k}g(x_t\cdot\widehat{\mu}^k) \le \frac{2}{\widehat{\beta}^k}|x_t\cdot\widehat{\mu}^k| \le \frac{2}{\widehat{\beta}^k}\|x_t\|_\infty\|\widehat{\mu}^k\|_1 \le 2W\,.$$

Note that in the first inequality we used the fact that $\varphi(v)$ is increasing for log-concave distributions and hence $g(v) = v + \varphi^{-1}(-v) \le v + |v| \le 2|v|$. The last inequality holds because $\|\widehat{\mu}^k/\widehat{\beta}^k\| \le W$ by the optimization constraint in (28).

Recalling our notation $\mathcal{H}_t = \bar{\mathcal{H}}_t\backslash\{\tilde{x}_{t+1}\}$ and applying the law of iterated expectations, we have

$$\begin{aligned}
\mathbb{E}(\mathsf{reg}_t|\mathcal{H}_{t-1}) &\le \frac{2CW}{\beta_0^2}\Big\langle\widehat{\mu}^k - \mu_0, \Sigma(\widehat{\mu}^k - \mu_0)\Big\rangle + \frac{8CW^3}{\beta_0^2}(\widehat{\beta}^k - \beta_0)^2 \\
&\le C_1W^3\Big(\Big\langle\widehat{\mu}^k - \mu_0, \Sigma(\widehat{\mu}^k - \mu_0)\Big\rangle + (\widehat{\beta}^k - \beta_0)^2\Big)\,,
\end{aligned} \tag{82}$$

with $C_1 = 8CW/\beta_0^2$. Following a similar derivation to (58) we get

$$\Big\langle\widehat{\mu}^k - \mu_0, \Sigma(\widehat{\mu}^k - \mu_0)\Big\rangle \le 16\left(\frac{Wu_F}{\ell_F} + W^2\right)\sqrt{\frac{\log d}{\tau_{k-1}}}\,. \tag{83}$$

Using Equations (83) and (79) in Equation (82), we get

$$\mathbb{E}(\mathsf{reg}_t) \le \underbrace{C_1W^3\left[16\left(\frac{Wu_F}{\ell_F} + W^2\right) + 4\frac{u_Fc_W}{\ell_F}\right]}_{C_2}\sqrt{\frac{\log d}{\tau_{k-1}}}\,. \tag{84}$$

Following a similar argument as in Section 8.1 (see Equation (45) and onwards) we have that the following holds for a suitable constant $C > 0$:

$$\begin{aligned}
\mathsf{Regret}(T) &\le C_2\sum_{k=2}^{K}\sqrt{\frac{\log d}{\tau_{k-1}}}\cdot\tau_k \\
&\le \sqrt{2}C_2\sum_{k=2}^{K}\sqrt{(\log d)\tau_{k-1}} = O(\sqrt{(\log d)T})\,,
\end{aligned}$$

where we used $\tau_k = 2\tau_{k-1}$.

For the lower bound $\Omega(\sqrt{T})$, note that under model (27) we can define the (scaled) customer's utility as

$$\tilde{u}(x_t) = \mu_0 \cdot x_t - \beta_0 p + \tilde{z}_t \,. \tag{85}$$

Then, a purchase occurs if $\tilde{u}(x_t) > 0$. Following our discussion in Section 4.1 (see after Equation (16)), since β_0 is unknown, the uninformative prices do exist and therefore $\Omega(\sqrt{T})$ applies to this case.

9. Conclusion

In this work, we leverage tools from statistical learning to design a dynamic pricing policy for a setting wherein the products are described via high-dimensional features. Our policy is computationally efficient and by exploiting the structure of demand parameters, it obtains a regret that scales gracefully with the features dimension and the time horizon. Namely, the regret of our algorithm scales linearly with the sparsity of the optimal solution and logarithmically with the dimension. We also show an $O(\log T)$ dependence of the regret on the length of the horizon. On the flip side, we provide a lower-bound of $O(\log T)$ on the regret of any algorithm that does not know the true parameters of the model in advance.

A natural next step is providing a tight bound on the regret, closing the gap between the derived upper and lower bounds. Another step would be assuming that θ^* is not exactly sparse, but it can be well approximated by a sparse vector, i.e, $\|\theta_0 - \theta_{s_0}\|_1 \leq \delta$ for some s_0-sparse vector θ_{s_0}. An interesting question is to figure out how the regret scales with δ.

The choice model that proposed in this work assumes one product arrived at each period, and describes the customer's purchase behavior based on the product features and the posted price. A more general choice model would be the one that assumes multiple products at each period. More specifically, each customer has a "consideration" set which includes products left after the customer has narrowed down her choices based on her own personal screening criteria, and then chooses the product from this set which brings maximum utility. We model the no purchase option as an extra product. This generalization is the focus of a future work.

We also believe the ideas and techniques developed in this work can be be applied to other settings such as personalized pricing where information about the buyers can be used for price differentiation or optimizing reserve prices in online ad auctions. Another application would be assortment optimization and learning consumer choice models both in terms of the role of the structure (Farias et al., 2013; Kallus and Udell, 2016) as well as inventory-constrained environments (Golrezaei et al., 2014; Ferreira et al., 2018; Cheung et al., 2018).

Acknowledgments

Authors are thankful to Arnoud den Boer and Paat Rusmevichientong for their suggestions that improved this work. A. J. would also like to acknowledge the financial support of the Office of the Provost at the University of Southern California through the Zumberge Fund Individual Grant Program, and an Outlier Research in Business (iORB) grant from the USC Marshall School of Business (2018). Authors are supported in part by a Google Faculty Research Award (2016).

Appendix A. Proof of Proposition 10

We start by reviewing the notion of *restricted eigenvalue* (RE) which is commonplace in high-dimensional statistical estimation.

Definition 22 *For a given matrix $A \in \mathbb{R}^{d\times d}$ and some integer s such that $1 \le s_0 \le d$ and a positive number c, we say that* Restricted Eigenvalue *(RE) condition is met if*

$$\kappa^2(A, s_0, c) \equiv \min_{\substack{J\subseteq[p]\\ |J|\le s_0}} \min_{\substack{v\neq 0\\ \|v_{J^c}\|_1\le c\|v_J\|_1}} \frac{v^\mathsf{T} A v}{\|v_J\|_2^2} > 0\,.$$

It is shown in (Bühlmann and van de Geer, 2011) and (Rudelson and Zhou, 2013) that when two matrices A_0, A_1 are close to each other (in the maximum element-wise norm) compared to sparsity s_0, the RE condition for A_0 implies the RE condition for A_1. This is particularly useful when A_0 is a population second moment matrix and A_1 is a corresponding empirical second moment matrix. To apply this result to our case, let $X \in \mathbb{R}^{n\times d}$ be the feature matrix with rows x_t, corresponding to n products. Let $\Sigma = \mathbb{E}(x_t x_t^\mathsf{T})$ be the empirical second moment. By Assumption 2, we have $\Sigma \succeq C_{\min}\mathrm{I}$ and therefore, Σ satisfies RE condition with $\kappa^2(\widetilde{\Sigma}, s_0, 3) \ge C_{\min}$. By using the following result, we conclude that $\widehat{\Sigma} = (X^\mathsf{T}X)/n$ also satisfies RE condition with $\kappa^2(\widehat{\Sigma}, s_0, 3) \ge C_{\min}/2$.

Proposition 23 *Let $\widehat{\Sigma} = (X^\mathsf{T}X)/n$ and let $S = \mathrm{supp}(\theta_0)$ be the support of θ_0. Under Assumption 2, $\widehat{\Sigma}$ satisfies the restricted eigenvalue condition with constant $\kappa(\widehat{\Sigma}, s_0, 3) \ge \sqrt{C_{\min}/2}$, with probability $1 - e^{-2n/(c_0 s_0)}$ and $c_0 = 768/C_{\min}^2$, provided that $n \ge c_0 s_0 \log d$,*

Proposition 23 follows from the results established in (Bühlmann and van de Geer, 2011) and (Rudelson and Zhou, 2013). We outline the main steps of its proof in Appendix A.1 for the reader's convenience.

By the second-order Taylor's theorem, expanding around θ_0 we have

$$\mathcal{L}(\theta_0) - \mathcal{L}(\widehat{\theta}) = -\langle \nabla\mathcal{L}(\theta_0), \widehat{\theta} - \theta_0\rangle - \frac{1}{2}\langle \widehat{\theta} - \theta_0, \nabla^2\mathcal{L}(\tilde{\theta})(\widehat{\theta} - \theta_0)\rangle\,, \tag{86}$$

for some $\tilde{\theta}$ on the line segment between θ_0 and $\widehat{\theta}$. Invoking (31), we have

$$\nabla\mathcal{L}(\theta) = \frac{1}{n}\sum_{t=1}^n \xi_t(\theta) x_t\,, \quad \nabla^2\mathcal{L}(\theta) = \frac{1}{n}\sum_{t=1}^n \eta_t(\theta) x_t x_t^\mathsf{T}\,, \tag{87}$$

where ∇ and ∇^2 represents the gradient and the hessian w.r.t θ. Further,

$$\begin{aligned}\xi_t(\theta) &= -\frac{f(u_t(\theta))}{F(u_t(\theta))}\mathbb{I}(y_t = -1) + \frac{f(u_t(\theta))}{1 - F(u_t(\theta))}\mathbb{I}(y_t = +1)\\ &= -\log' F(u_t(\theta))\mathbb{I}(y_t = -1) - \log'(1 - F(u_t(\theta)))\mathbb{I}(y_t = +1)\,.\end{aligned}$$

$$\begin{aligned}\eta_t(\theta) &= \left(\frac{f(u_t(\theta))^2}{F(u_t(\theta))^2} - \frac{f'(u_t(\theta))}{F(u_t(\theta))}\right)\mathbb{I}(y_t = -1) + \left(\frac{f(u_t(\theta))^2}{(1 - F(u_t(\theta)))^2} + \frac{f'(u_t(\theta))}{1 - F(u_t(\theta))}\right)\mathbb{I}(y_t = +1)\\ &= -\log'' F(u_t(\theta))\mathbb{I}(y_t = -1) - \log''(1 - F(u_t(\theta)))\mathbb{I}(y_t = +1)\,,\end{aligned}$$

where $u_t(\theta) = p_t - \langle x_t, \theta\rangle$, and $\log' F(x)$ and $\log'' F(x)$ represent first and second derivative w.r.t x, respectively.

By Equation (37), we have

$$|u_t(\theta_0)| \le |p_t| + \|x_t\|_\infty\|\theta_0\|_1 \le 3W\,.$$

Further, recall that the sequences $\{p_t\}_{t=1}^n$ and $\{x_t\}_{t=1}^n$ are independent of $\{z_t\}_{t=1}^n$. Therefore, $\{u_t(\theta_0)\}_{t=1}^T$ and $\{z_t(\theta_0)\}_{t=1}^T$ are independent and by (4), we have $\mathbb{E}[\xi_t(\theta_0)] = \mathbb{E}[\mathbb{E}[\xi_t(\theta_0)|u_t(\theta_0)]] = 0$. Further, by definition of u_F, cf. Equation (13), we have $|\xi_t(\theta_0)| \le u_F$.

We next introduce the set

$$\mathcal{F} \equiv \left\{\|\nabla\mathcal{L}(\theta_0)\|_\infty \le 2u_F\sqrt{\frac{\log d}{n}}\right\}. \tag{88}$$

By applying Azuma-Hoeffding inequality followed by union bounding over d coordinates of feature vectors, we obtain $\mathbb{P}(\mathcal{F}) \ge 1 - 1/d$.

On the other note, $\|\theta_0\|_1, \|\tilde{\theta}\|_1 \le W$ and hence $\|\tilde{\theta}\|_1 \le W$. This implies that $|u_t(\tilde{\theta})| \le 3W$. Therefore, by definition of ℓ_F, cf. Equation (33), we have $\eta_t(\tilde{\theta}) \ge \ell_F$. Recalling Equation (87), we get $\nabla^2\mathcal{L}(\tilde{\theta}) \succeq \ell_F(\widetilde{X}^\mathsf{T}\widetilde{X}/n)$.

By optimality of $\widehat{\theta}$, we write

$$\mathcal{L}(\widehat{\theta}) + \lambda\|\widehat{\theta}\|_1 \le \mathcal{L}(\theta_0) + \lambda\|\theta_0\|_1\,, \tag{89}$$

and by rearranging the terms and using (86), we arrive at

$$\frac{\ell_F}{n}\|X(\theta_0 - \widehat{\theta})\|^2 + \lambda\|\widehat{\theta}\|_1 \le \|\nabla\mathcal{L}(\theta_0)\|_\infty\|\widehat{\theta} - \theta_0\|_1 + \lambda\|\theta_0\|_1\,. \tag{90}$$

Form now on, the analysis is exactly similar to the oracle inequality for Lasso estimator. We bring the analysis here for the reader's convenience.

Choosing $\lambda \ge 4u_F\sqrt{(\log d)/n}$, we have on $\mathcal{F}$

$$\frac{2\ell_F}{n}\|X(\theta_0 - \widehat{\theta})\|^2 + 2\lambda\|\widehat{\theta}\|_1 \le \lambda\|\widehat{\theta} - \theta_0\|_1 + 2\lambda\|\theta_0\|_1\,. \tag{91}$$

Let $S = \mathrm{supp}(\theta_0)$. On the left-hand side using triangle inequality, we have

$$\|\widehat{\theta}\|_1 = \|\widehat{\theta}_S\|_1 + \|\widehat{\theta}_{S^c}\|_1 \ge \|\widehat{\theta}_S\|_1 - \|\widehat{\theta}_S - \theta_{0,S}\|_1 + \|\widehat{\theta}_{S^c}\|_1\,.$$

On the right-hand side, we have

$$\|\widehat{\theta} - \theta_0\|_1 = \|\widehat{\theta}_S - \theta_{0,S}\|_1 + \|\widehat{\theta}_{S^c}\|_1\,.$$

Using these two inequalities in (91), we get

$$\frac{2\ell_F}{n}\|X(\theta_0 - \widehat{\theta})\|^2 + \lambda\|\widehat{\theta}_{S^c}\|_1 \le 3\lambda\|\widehat{\theta}_S - \theta_{0,S}\|_1\,. \tag{92}$$

We next write

$$\begin{aligned}\frac{2\ell_F}{n}\|X(\theta_0-\widehat{\theta})\|^2+\lambda\|\widehat{\theta}-\theta_0\|_1 &= \frac{2\ell_F}{n}\|X(\theta_0-\widehat{\theta})\|^2+\lambda\|\widehat{\theta}_S-\theta_{0,S}\|_1+\lambda\|\widehat{\theta}_{S^c}\|_1\\ &\overset{(a)}{\le} 4\lambda\|\widehat{\theta}_S-\theta_{0,S}\|_1 \overset{(b)}{\le} 4\lambda\sqrt{s_0}\|\widehat{\theta}_S-\theta_{0,S}\|_2\\ &\overset{(c)}{\le} \frac{4\lambda\sqrt{2s_0}}{\sqrt{nC_{\min}}}\|X(\widehat{\theta}-\theta_0)\|_2\\ &\overset{(d)}{\le} \frac{\ell_F}{n}\|X(\widehat{\theta}-\theta_0)\|_2^2+\frac{8\lambda^2 s_0}{\ell_F C_{\min}}\,,\end{aligned}$$

where (a) follows from Equation (92); (b) holds by Cauchy-Shwarz inequality; (c) follows form the RE condition, which holds for $\widehat{\Sigma}=(X^\mathsf{T}X)/n$ as stated by Proposition 23, with $\kappa(\widehat{\Sigma},s_0,3)\ge\sqrt{C_{\min}/2}$, and recalling the inequality $\|\widehat{\theta}_{S^c}-\theta_{0,S^c}\|_1=\|\widehat{\theta}_{S^c}\|_1\le 3\|\widehat{\theta}_S-\theta_{0,S}\|$ as per Equation (92); Finally (d) follows from the inequality $2\sqrt{ab}\le a^2+b^2$. Rearranging the terms, we obtain

$$\frac{\ell_F}{n}\|X(\theta_0-\widehat{\theta})\|^2+\lambda\|\widehat{\theta}-\theta_0\|_1\le\frac{8\lambda^2 s_0}{\ell_F C_{\min}}\,. \tag{93}$$

Applying the RE condition again to the l.h.s of (93), we get

$$C_{\min}\frac{\ell_F}{2}\|\theta_0-\widehat{\theta}\|_2^2\le\frac{\ell_F}{n}\|X(\theta_0-\widehat{\theta})\|^2\le\frac{8\lambda^2 s_0}{\ell_F C_{\min}}\,, \tag{94}$$

and therefore,

$$\|\theta_0-\widehat{\theta}\|_2^2\le\frac{16 s_0\lambda^2}{\ell_F^2 C_{\min}^2}\,. \tag{95}$$

The result follows.

A.1. Proof of Proposition 23

The proof follows by combining two lemmas from (Bühlmann and van de Geer, 2011).

We show the desired result holds for a more general case, namely for X with subgaussian entries. Before stating the proof, we recall a few definitions and notations.

Definition 24 *A random variable ν is subgaussian if there exist constants L,σ_0 such that*

$$\mathbb{E}(e^{\frac{\nu^2}{L^2}})\le\frac{\sigma_0^2}{L^2}+1\,.$$

Note that bounded random variables are subgaussian. Specifically, if $|\nu|\le\nu_{\max}$, then ν is subgaussian with $L=\nu_{\max}$ and $\sigma_0=\nu_{\max}\sqrt{e-1}$.

For a matrix A, we let $\|A\|_\infty$ denote its (element wise) maximum norm, i.e., $\|A\|_\infty=\max_{i,j}|A_{ij}|$. The next lemma shows that if two matrices are close enough in maximum norm and if the compatibility condition holds for one of them then it would also hold for the other one.

Lemma 25 *Suppose that the restricted eigenvalue (RE) condition holds for Σ_0 with constant $\kappa(\Sigma_0,s_0,3)>0$. If*

$$\|\Sigma_0-\Sigma_1\|_\infty\le\frac{\kappa^2(\Sigma_0,s_0,3)}{32 s_0}\,,$$

then the RE condition holds for Σ_1 with constant $\kappa(\Sigma_1,s_0,3)\ge\kappa(\Sigma_0,s_0,3)/\sqrt{2}$.

Proof Proof of Lemma 25 We refer to Problem 6.10 of (Bühlmann and van de Geer, 2011). ■

Lemma 26 *Consider $X\in\mathbb{R}^{n\times p}$ with i.i.d. rows generated from a distribution with covariance $\Sigma\in\mathbb{R}^{d\times d}$. Let $\widehat{\Sigma}=X^\mathsf{T}X/n$ be the corresponding empirical second moment. Further, suppose that the entries of X are uniformly subgaussian with parameters L,σ_0. If $n\ge c_0 L s_0\log d$ with $c_0=768L/\kappa^2(\Sigma,s_0,3)$, then*

$$\mathbb{P}\left[\|\widehat{\Sigma}-\Sigma\|_\infty\ge\frac{\kappa^2(\Sigma,s_0,3)}{384 s_0}\Big(2+\frac{7\sigma_0}{L}\Big)\right]\le e^{-\frac{2n}{cs_0}}\,. \tag{96}$$

Proof Proof of Lemma 26 The result follows readily from Problem 14.3 on page 535 of (Bühlmann and van de Geer, 2011). ■

Next we note that Σ satisfies the restricted eigenvalue condition with constant $\kappa^2(\Sigma,s_0,3)\ge C_{\min}$ because of Assumption 2. Further, since $\|x_t\|_\infty\le 1$, we can apply the result of Lemma 26 with $L=1$, $\sigma_0=\sqrt{e-1}$. Proposition 23 then follows from Lemma 25.

Appendix B. Proof of Proposition 12

Define the event $\mathcal{B}_n$ as follows:

$$\mathcal{B}_n\equiv\Big\{X\in\mathbb{R}^{n\times d}:\ \sigma_{\min}(X^\mathsf{T}X/n)>C_{\min}/2\Big\}\,. \tag{97}$$

Using concentration bounds on the spectrum of random matrices with subgaussian rows (see (Vershynin, 2010, Equation (5.26))), there exist constants $c,c_1>0$ such that for $n>c_1 d$, we have $\mathbb{P}(\mathcal{B}_n)\ge 1-e^{-cn^2}$.

For $\gamma>0$, we define the event $\mathcal{F}_\gamma=\{\|\nabla\mathcal{L}(\theta_0)\|_\infty\le\gamma\}$. Using characterization (87), and by applying Azuma-Hoeffding inequality (similar to our argument after Equation (88)), we obtain

$$\mathbb{P}(\mathcal{F}_\gamma)\ge 1-d\exp\Big(-\frac{n\gamma^2}{2u_F^2}\Big)\,. \tag{98}$$

We also let $\mathcal{E}_{1,n}\equiv\mathcal{B}_n\cap\mathcal{F}_{\lambda/2}$, $\mathcal{E}_{2,n}\equiv\mathcal{B}_n\cap\mathcal{F}^c_{\lambda/2}$. To lighten the notation, we use the shorthand $D\equiv\|\widehat{\theta}-\theta_0\|_2^2$.

We then have

$$\mathbb{E}(D)=\mathbb{E}(D\cdot\mathbb{I}(\mathcal{B}_n^c))+\mathbb{E}(D\cdot\mathbb{I}(\mathcal{E}_{1,n}))+\mathbb{E}(D\cdot\mathbb{I}(\mathcal{E}_{2,n}))\,. \tag{99}$$

We treat each of the terms on the right-hand side separately.

- **Term 1:** We have

$$\mathbb{E}(D\cdot\mathbb{I}(\mathcal{B}_n^c))\le 4W^2\mathbb{P}(\mathcal{B}_n^c)\le 4W^2e^{-cn^2}\,. \tag{100}$$

- **Term 2:** Similar to proof of Proposition 10, on $\mathcal{E}_{1,n}$, we have $D\le 16s_0\lambda^2/(\ell_F^2C_{\min}^2)$. Hence,

$$\mathbb{E}(D\cdot\mathbb{I}(\mathcal{E}_{1,n}))\le\frac{16s_0\lambda^2}{\ell_F^2C_{\min}^2}\,. \tag{101}$$

- **Term 3:** To bound term 3, we first prove the following lemma.

  Lemma 27 *On event $\mathcal{E}_n(\gamma)\equiv\mathcal{B}_n\cap\mathcal{F}_\gamma$, with $\gamma>\lambda/2$, we have*

$$D\le\Big(\frac{36}{C_{\min}^2\ell_F^2}\Big)\gamma^2 d.$$

  Lemma 27 is proved in Section B.1.

We next bound term 3 as follows. Let $L=9\lambda^2d/(C_{\min}^2\ell_F^2)$.

$$\begin{aligned}\mathbb{E}(D\cdot\mathbb{I}(\mathcal{E}_{2,n}))&=\int_0^\infty\mathbb{P}\Big(D\cdot\mathbb{I}(\mathcal{E}_{2,n})>\alpha\Big)\mathrm{d}\alpha\\&=L\int_0^\infty\mathbb{P}\Big(D\cdot\mathbb{I}(\mathcal{E}_{2,n})>Lc\Big)\mathrm{d}c\\&=L\int_0^1\mathbb{P}\Big(D\cdot\mathbb{I}(\mathcal{E}_{2,n})>Lc\Big)\mathrm{d}c+L\int_1^\infty\mathbb{P}\Big(D\cdot\mathbb{I}(\mathcal{E}_{2,n})>Lc\Big)\mathrm{d}c\,.\end{aligned} \tag{102}$$

For the first term on the right-hand side we write

$$L\int_0^1\mathbb{P}\Big(D\cdot\mathbb{I}(\mathcal{E}_{2,n})>Lc\Big)\mathrm{d}c\le L\int_0^1\mathbb{P}(\mathcal{E}_{2,n})\le L\mathbb{P}(\mathcal{F}_{\lambda/2}^c)\le\frac{L}{d}\,, \tag{103}$$

where the last step holds from Equation (98) with $\gamma=\lambda/2$.

We next upper bound the second term. For arbitrary fixed $c>1$, let $\gamma=\sqrt{Lc/(d\kappa)}$, with $\kappa=36/(\ell_F^2C_{\min}^2)$. It is easy to verify that $\gamma=\sqrt{c}\lambda/2>\lambda/2$. Further, by virtue of Lemma 27, on $\mathcal{E}(\gamma)$ we have $D\le\kappa\gamma^2d=Lc$. Hence,

$$\mathbb{P}\Big(D\cdot\mathbb{I}(\mathcal{E}_{2,n})>Lc\Big)\le\mathbb{P}\Big(\mathcal{E}(\gamma)^c\cap\mathcal{E}_{2,n}\Big)\le\mathbb{P}(\mathcal{F}_\gamma^c\cap\mathcal{B}_n)\le\mathbb{P}(\mathcal{F}_\gamma^c)\,. \tag{104}$$

Further, applying Equation (98) and plugging for γ, we obtain

$$\mathbb{P}(\mathcal{F}_\gamma^c)\le d\exp\Big(-\frac{nLc}{2u_F^2\kappa d}\Big)=d\exp\Big(-\frac{n\lambda^2}{8u_F^2}\Big)\le d^{1-2c}\,. \tag{105}$$

Here, the second step follows from definition of L and the last step holds because $\lambda\ge 4u_F\sqrt{(\log d)/n}$.

Combining Equations (104) and (105), we have

$$L\int_1^\infty\mathbb{P}\Big(D\cdot\mathbb{I}(\mathcal{E}_{2,n})>Lc\Big)\mathrm{d}c\le L\int_1^\infty d^{1-2c}\,\mathrm{d}c\le\frac{L}{2d\log d}\,. \tag{106}$$

Using bounds (103) and (106) in Equation (102), we obtain

$$\mathbb{E}(D\cdot\mathbb{I}(\mathcal{E}_{2,n}))\le\frac{L}{d}\left(1+\frac{1}{2\log d}\right)\le\frac{3L}{2d}<\frac{16\lambda^2}{\ell_F^2C_{\min}^2}\,. \tag{107}$$

The result follows by putting the upper bounds on the three terms together.

B.1. Proof of Lemma 27

We start by rewriting Equation (90), which follows from optimality of $\widehat{\theta}$ and log-concave property of the loss function:

$$\frac{\ell_F}{n}\|X(\theta_0-\widehat{\theta})\|^2+\lambda\|\widehat{\theta}\|_1\le\|\nabla\mathcal{L}(\theta_0)\|_\infty\|\widehat{\theta}-\theta_0\|_1+\lambda\|\theta_0\|_1\,. \tag{108}$$

On event $\mathcal{E}_n(\gamma)$, Equation (108) implies that

$$\frac{1}{2}C_{\min}\ell_F\|\theta_0-\widehat{\theta}\|_2^2+\lambda\|\widehat{\theta}\|_1\le\gamma\|\widehat{\theta}-\theta_0\|_1+\lambda\|\theta_0\|_1\,. \tag{109}$$

Using the assumption $\gamma>\lambda/2$ and our shorthand $D\equiv\|\widehat{\theta}-\theta_0\|_2^2$, we get

$$\begin{aligned}\frac{1}{2}C_{\min}\ell_F\|\widehat{\theta}-\theta_0\|_2^2&\le\gamma\|\widehat{\theta}-\theta_0\|_1+\lambda\|\theta_0\|_1-\lambda\|\widehat{\theta}\|_1\\&\le(\gamma+\lambda)\|\widehat{\theta}-\theta_0\|_1\\&\le 3\gamma\|\widehat{\theta}-\theta_0\|_1\le 3\gamma\sqrt{d}\|\widehat{\theta}-\theta_0\|_2\,.\end{aligned}$$

Writing the above bound in terms of our shorthand $D\equiv\|\widehat{\theta}-\theta_0\|_2^2$, we obtain the desired result.

Appendix C. Proof of Technical Lemmas

C.1. Proof of Lemma 13

We write the virtual valuation function as $\varphi(v)=v-1/\lambda(v)$ where $\lambda(v)=\frac{f(v)}{1-F(v)}=-\log'(1-F(v))$ is the hazard rate function. Since $1-F$ is log-concave, the hazard function $\lambda(v)$ is increasing which implies that φ is strictly increasing. Indeed, by this argument $\varphi'(v)>1$.

C.2. Proof of Lemma 14

Recalling the definition $g(v)=v+\varphi^{-1}(-v)$, we have $g'(v)=1-1/\varphi'(\varphi^{-1}(-v))$. Since φ is strictly increasing by Lemma 13, we have $g'(v)<1$. The claim $g'(v)>0$ follows if we show $\varphi'(\varphi^{-1}(-v))>1$. For this we refer to the proof of Lemma 13, where we showed that $\varphi'(v)>1$ for all v.

C.3. Proof of Lemma 16

Let $\phi(v)$ and $\Phi(v)$ respectively denote the density and the distribution function of standard normal variable. Function h_t and its derivatives read as

$$\mathsf{rev}_t(p) = p\Big(1-\Phi\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)\Big)\,, \tag{110}$$

$$\mathsf{rev}_t'(p) = 1-\Phi\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)-\frac{p}{\sigma}\phi\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)\,, \tag{111}$$

$$\mathsf{rev}_t''(p) = \frac{1}{\sigma}\Big[\frac{p}{\sigma}\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)-2\Big]\phi\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)\,. \tag{112}$$

Define $\xi\equiv p_t^*-x_t\cdot\theta_0=g(x_t\cdot\theta_0)-x_t\cdot\theta_0$. Writing $\mathsf{rev}_t''(p_t^*)$ in term of ξ, we obtain

$$r_t''(p_t^*) = \frac{1}{\sigma}\Big[\frac{\xi}{\sigma}\Big(\frac{1-\Phi(\xi/\sigma)}{\phi(\xi/\sigma)}\Big)-2\Big]\phi(\xi/\sigma)\,. \tag{113}$$

By tail bound inequality for Gaussian distribution $1-\Phi(\xi/\sigma)\le(\sigma/\xi)\phi(\xi/\sigma)$ for $\xi\ge 0$. Therefore,

$$\frac{\xi}{\sigma}\Big(\frac{1-\Phi(\xi/\sigma)}{\phi(\xi/\sigma)}\Big)-2\le -1\,, \tag{114}$$

and the same bound obviously holds for $\xi<0$.

By definition of function g, $|\xi|\le 3W$ with φ being the virtual valuation fusion corresponding to the Gaussian distribution. Hence, $\phi(\xi/\sigma)\ge\phi(3W/\sigma)$. Putting this together with (114), we get $\mathsf{rev}_t''(p_t^*)\le -c_1$ with $c_1=(1/\sigma)\phi(3W/\sigma)$.

For the second part of the Lemma statement, set $\delta=\min\Big\{3W,\sigma^2/(18W),\sigma^2\phi(3W/\sigma)\Big\}$. For $p\in[p_t^*-\delta,p_t^*+\delta]$, we have

$$\Big|\frac{p}{\sigma}\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)-\frac{p_t^*}{\sigma}\Big(\frac{p_t^*-x_t\cdot\theta_0}{\sigma}\Big)\Big|\le\frac{1}{\sigma^2}|p-p_t^*|\cdot|p+p_t^*-x_t\cdot\theta_0|\le\frac{1}{\sigma^2}\delta(5W+\delta)\le\frac{1}{2}\,. \tag{115}$$

Using Equation (114) we get $p(p-x_t\cdot\theta_0)/\sigma^2\le -1/2$. Further,

$$\Big|\phi\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)-\phi\Big(\frac{p_t^*-x_t\cdot\theta_0}{\sigma}\Big)\Big|\le\frac{|p-p_t^*|}{2\sigma^2}\le\frac{\delta}{2\sigma^2}\le\frac{1}{2}\phi(3W/\sigma)\,. \tag{116}$$

Therefore,

$$\phi\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)\ge\phi(\xi/\sigma)-\frac{1}{2}\phi(3W/\sigma)\ge\frac{1}{2}\phi(3W/\sigma)\,. \tag{117}$$

Combining (115), (117) we obtain

$$\mathsf{rev}_t''(p) = \frac{1}{\sigma}\Big[\frac{p}{\sigma}\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)-2\Big]\phi\Big(\frac{p-x_t\cdot\theta_0}{\sigma}\Big)\le-\frac{1}{4\sigma}\phi(3W/\sigma)=-\frac{c_1}{4}\,. \tag{118}$$

The result follows.

C.4. Proof of Lemma 17

Let $Z=x\cdot v$ and $\tilde{Z}=Z/\|v\|_2$. Note that $\mathrm{Var}(\tilde{Z})=1$. Write the expectation in terms of the tail probability

$$\mathbb{E}(\min(Z^2,\delta^2)) = \int_0^{\delta^2}\mathbb{P}(Z^2\ge t)\mathrm{d}t=\int_0^{\delta^2}\mathbb{P}\Big(|\tilde{Z}|\ge\frac{\sqrt{t}}{\|v\|_2}\Big)\mathrm{d}t\,. \tag{119}$$

We consider two cases:

- $\delta\le\|v\|_2$: The right-hand side in (119) can be lower bounded as

$$\int_0^{\delta^2}\mathbb{P}\Big(|\tilde{Z}|\ge\frac{\sqrt{t}}{\|v\|_2}\Big)\mathrm{d}t\ge\int_0^{\delta^2}\mathbb{P}\Big(|\tilde{Z}|\ge\frac{\sqrt{t}}{\delta}\Big)\mathrm{d}t=2\delta^2\int_0^1 t\,\mathbb{P}(|\tilde{Z}|\ge t)\mathrm{d}t\,. \tag{120}$$

  In the sequel, we provide two separate lower bounds for the right-hand side.

  Let $\xi\equiv\mathbb{P}(|\tilde{Z}|\ge 1)$. We have

$$\int_0^1 t\,\mathbb{P}\Big(|\tilde{Z}|\ge t\Big)\mathrm{d}t\ge\int_0^1 t\,\xi\mathrm{d}t\ge\frac{\xi}{2}\,. \tag{121}$$

  We proceed to obtain another bound which utilizes the fact $\mathrm{Var}(\tilde{Z})=1$.

$$\begin{aligned}\int_0^1 t\,\mathbb{P}(|\tilde{Z}|\ge t)\mathrm{d}t&=\int_0^\infty t\,\mathbb{P}(|\tilde{Z}|\ge t)\mathrm{d}t-\int_1^\infty t\,\mathbb{P}(|\tilde{Z}|\ge t)\mathrm{d}t\\&=\frac{1}{2}\mathrm{Var}(\tilde{Z})-\int_1^\infty t\,\mathbb{P}(|\tilde{Z}|\ge t)\mathrm{d}t\,.\end{aligned} \tag{122}$$

  For $t\ge 1$, we have $\mathbb{P}(|\tilde{Z}|\ge t)\le\xi$. Further, by applying Chernoff bound, we get

$$\begin{aligned}\mathbb{P}(|\tilde{Z}|\ge t)&=2\mathbb{P}(\tilde{Z}\ge t)=2\mathbb{P}(e^{\lambda\tilde{Z}}\ge e^{\lambda t})\le e^{-\lambda t}\mathbb{E}(e^{\lambda\tilde{Z}})\\&=2e^{-\lambda t}\prod_{i=1}^d\Big(\frac{e^{\lambda\frac{v_i}{\|v\|_2}}+e^{-\lambda\frac{v_i}{\|v\|_2}}}{2}\Big)\le 2e^{-\lambda t}\prod_{i=1}^d e^{\lambda^2\frac{v_i^2}{2\|v\|_2^2}}=2e^{\frac{\lambda^2}{2}-\lambda t}\,.\end{aligned}$$

  Setting $\lambda=t$ leads to $\mathbb{P}(|\tilde{Z}|\ge t)\le 2e^{-\frac{t^2}{2}}$. Combining these bounds into (121), we obtain

$$\int_0^1 t\,\mathbb{P}(|\tilde{Z}|\ge t)\mathrm{d}t\ge\frac{1}{2}-\int_1^\infty t\,\min(2e^{-\frac{t^2}{2}},\xi)\mathrm{d}t=\frac{1-\xi}{2}-\xi\log\Big(\frac{2}{\xi}\Big)\,. \tag{123}$$

  We summarize bounds (121) and (123) as in

$$\int_0^1 t\,\mathbb{P}\Big(|\tilde{Z}|\ge t\Big)\mathrm{d}t\ge\min_{\xi\in[0,1]}\max\Big(\xi,\frac{1-\xi}{2}-\xi\log\Big(\frac{2}{\xi}\Big)\Big)>0.05\,. \tag{124}$$

  Turning back to Equation (120), in this case we have

$$\int_0^{\delta^2}\mathbb{P}\Big(|\tilde{Z}|\ge\frac{\sqrt{t}}{\|v\|_2}\Big)\mathrm{d}t\ge 0.1\delta^2\,. \tag{125}$$

- $\delta \geq \|v\|_2$: Similar to the previous case, the right-hand side in (119) can be lower bounded as

$$\int_0^{\delta^2} \mathbb{P}\Big(|\tilde{Z}| \geq \frac{\sqrt{t}}{\|v\|_2}\Big)\mathrm{d}t \geq \int_0^{\|v\|_2^2} \mathbb{P}\Big(|\tilde{Z}| \geq \frac{\sqrt{t}}{\|v\|_2}\Big)\mathrm{d}t\,. \geq 0.1\|v\|_2^2 \tag{126}$$

The above two cases can be summarized as $\mathbb{E}(\min(Z^2, \delta^2)) \geq 0.1\min(\|v\|_2^2, \delta^2)$.

C.5. Proof of Lemma 18

We use a standard argument that relates minimax ℓ_2-risk in terms of the error in multi-way hypothesis testing problem; See e.g. (Yang and Barron, 1999; Yu, 1997). Let $\{\tilde{\theta}_1, \ldots, \tilde{\theta}_m\}$ be a δ-packing of set Ω, meaning that their pairwise distances are all at least δ. Parameter δ is free for now and its value will be determined later in the proof. We further let P_j denote the induced probability on market values $(v(x_1), \ldots, v(x_T))$, conditional on $(x_1, \ldots, x_T)$ and for $\theta_0 = \tilde{\theta}_j$. In other words, in defining distributions P_j we treat feature vectors fixed. Let ν be random variable uniformly distributed on the hypothesis set $\{1, 2, \ldots, m\}$ which indicates the index of the true parameter, i.e, $\nu = j$ means $\theta_0 = \tilde{\theta}_j$.

Define $\mathsf{d}(\theta_1^T, \theta) \equiv \sum_{t=1}^T \min(\|\theta_t - \theta\|_2^2, C)$ and let μ be the value of j for which $\mathsf{d}(\theta_1^T, \tilde{\theta}_j)$ is a minimum. Suppose that δ is chosen such that $\delta^2 \leq C$. If $\mathsf{d}(\theta_1^T, \tilde{\theta}_j) < \delta^2 T/4$ then $\mu = j$, because assuming otherwise, we have $\mu = j' \neq j$, and by triangle inequality

$$\begin{aligned}\min(\|\tilde{\theta}_{j'} - \tilde{\theta}_j\|_2^2, C) &\leq \min(2\|\theta_t - \tilde{\theta}_j\|_2^2 + 2\|\theta_t - \tilde{\theta}_{j'}\|_2^2, C) \\ &\leq \min(2\|\theta_t - \tilde{\theta}_j\|_2^2, C) + \min(2\|\theta_t - \tilde{\theta}_{j'}\|_2^2, C)\,,\end{aligned} \tag{127}$$

for all t, where we used the inequality $\min(a+b, c) \leq \min(a, c) + \min(b, c)$ for $a, b, c \geq 0$. Summing over $t = 1, 2, \ldots, T$, we get

$$T\min(\|\tilde{\theta}_{j'} - \tilde{\theta}_j\|_2^2, C) \leq 2\mathsf{d}(\theta_1^T, \tilde{\theta}_j) + 2\mathsf{d}(\theta_1^T, \tilde{\theta}_{j'}) \leq 4\mathsf{d}(\theta_1^T, \tilde{\theta}_j) < \delta^2 T\,,$$

where we used the assumption $\mu = j'$. But this is a contradiction because $\|\tilde{\theta}_{j'} - \tilde{\theta}_j\|_2 \geq \delta$ (they form a δ-packing of Ω) and $\delta^2 \leq C$.

Using Markov inequality, we can write

$$\begin{aligned}\max_j \mathbb{E}_{P_j}\mathsf{d}(\theta_1^T, \tilde{\theta}_j) &\geq \frac{\delta^2 T}{4}\max_j \mathbb{P}\Big(\mathsf{d}(\theta_1^T, \tilde{\theta}_j) \geq \frac{\delta^2 T}{4}\Big|\nu = j\Big) \\ &\geq \frac{\delta^2 T}{4m}\sum_{j=1}^m \mathbb{P}(\mu \neq j|\nu = j) = \frac{\delta^2 T}{4}\mathbb{P}(\mu \neq \nu)\,.\end{aligned} \tag{128}$$

We use Fano's inequality to lower bound the error probability on the right-hand side. We first construct a δ-packing of Ω similar to the one proposed in (Raskutti et al., 2011, proof of Theorem 1).

Let $s = s_0/2 \leq d/2$ and define

$$\mathcal{A} = \{q \in \{-1, 0, 1\}^d : \ \|q\|_0 = s\}\,.$$

As proved in (Raskutti et al., 2011, Lemma 5), there exists a subset $\tilde{\mathcal{A}} \subseteq \mathcal{A}$ of cardinality $|\tilde{\mathcal{A}}| \geq \exp(\frac{s}{2}\log\frac{d-s/2}{s})$ such that the Hamming distance between any two elements in $\tilde{\mathcal{A}}$ is at least $s/2$. Next, consider the set $\sqrt{\frac{2}{s}}\delta\tilde{\mathcal{A}}$ for some $\delta \leq W/\sqrt{2s}$ whose exact value to be determined later. Then, for q in this set, $\|q\|_1 = \sqrt{2s}\delta \leq W$ and hence $\sqrt{\frac{2}{s}}\delta\tilde{\mathcal{A}} \subseteq \Omega$. Further, for $q, q' \in \sqrt{\frac{2}{s}}\delta\tilde{\mathcal{A}}$, we have the following bounds:

$$\|q - q'\|_2^2 \geq \delta^2\,, \tag{129}$$

$$\|q - q'\|_2^2 \leq 8\delta^2\,. \tag{130}$$

By (129), the set $\sqrt{\frac{2}{s}}\delta\tilde{\mathcal{A}}$ forms a δ-packing for Ω with size $|\tilde{\mathcal{A}}|$.

We now turn back to bound (128). Left-hand side can be lower bounded using Fano's inequality. We omit the details here as it is a standard argument and instead we refer to (Raskutti et al., 2011, proof of Theorem 1) for details. Using Fano's inequality and bound (130), we get

$$\mathbb{P}(\mu \neq \nu) = 1 - \frac{\frac{8T}{2\sigma^2}\delta^2 + \log(2)}{\frac{s}{2}\log(\frac{d-s/2}{s})}\,. \tag{131}$$

Choosing $\delta^2 \leq \delta_1^2 \equiv \frac{\sigma^2 s}{32T}\log(\frac{d-s/2}{s})$, we obtain $\mathbb{P}(\mu \neq \nu) \geq 1/4$. Therefore, setting $\delta^2 = \min(\frac{W^2}{2s}, \delta_1^2, C)$ and combining with bound (128), we conclude that

$$\min_{\theta_1^T}\max_{\theta_0 \in \Omega}\mathbb{E}(\mathsf{d}(\theta_1^T, \theta_0)) \geq \frac{\delta^2 T}{16} = \frac{1}{16}\min\left\{\frac{W^2 T}{2s}, \frac{\sigma^2 s}{32}\log\Big(\frac{d-s/2}{s}\Big), CT\right\}. \tag{132}$$

Now since $s = s_0/2 \leq d/2$, we have $\log((d-s/2)/s) \geq c\log(d/s)$ with some constant $c > 0$. Therefore, by using Equation (132) and substituting for $s = s_0/2$, we obtain

$$\min_{\theta_1^T}\max_{\theta_0 \in \Omega}\mathbb{E}(\mathsf{d}(\theta_1^T, \theta_0)) \geq L_1\,, \tag{133}$$

with

$$L_1 \equiv \frac{1}{16}\min\left\{\frac{W^2 T}{s_0}, \frac{c\sigma^2 s_0}{64}\log\Big(\frac{d}{s_0}\Big), CT\right\}. \tag{134}$$

We next derive another separate lower bound for minimax risk, by assuming that an oracle gives us the true support of θ_0. In this case, the least square estimator, applied to the observed features restricted to the true support S, achieves the optimal minimax ℓ_2 rate. This implies that $\|\theta_t - \theta_0\|_2^2 \geq c\sigma^2 s_0/t$, for $t \geq s_0$ and a constant $c > 0$. Therefore,

$$\min_{\theta_1^T}\max_{\theta_0 \in \Omega}\mathbb{E}(\mathsf{d}(\theta_1^T, \theta_0)) \geq \sum_{t=1}^T \min\Big(c\sigma^2\frac{s_0}{t}, C\Big)\,, \tag{135}$$

from which we obtain

$$\min_{\theta_1^T}\max_{\theta_0 \in \Omega}\mathbb{E}(\mathsf{d}(\theta_1^T, \theta_0)) \geq L_2 \equiv c' s_0\,\log(T/s_0)\,, \tag{136}$$

for some constant $c' > 0$, depending on σ and C.

Combining bounds in (134) and (136), we have

$$\min_{\theta_1^T} \max_{\theta_0 \in \Omega} \mathbb{E}(\mathsf{d}(\theta_1^T, \theta_0)) \geq \frac{1}{2}(L_1 + L_2) \geq \widetilde{C}\left\{ s_0 \log\left(\frac{T}{s_0}\right) + \min\left[\frac{T}{s_0}, s_0 \log\left(\frac{d}{s_0}\right)\right]\right\}, \tag{137}$$

for a constant $\widetilde{C}$ that depends on C, σ, W. The proof is complete.

C.6. Proof of Lemma 19

We recall the notion of 0-*property* established by (Ponomarev, 1987).

Definition 28 *A continuous function has the* 0-property*, if the pre-image of any set of probability zero is a set of probability zero.*

As proved in (Ponomarev, 1987, Theorem 1), if a function $\phi : \mathcal{X} \subseteq \mathbb{R}^d \mapsto \mathbb{R}^d$ is continuously differentiable, then it satisfies 0-property if and only if its derivative $\mathcal{D}\phi$ is full rank for almost all $x \in \mathcal{X}$. Therefore, we need to show that under Assumption 1, if ϕ has 0-property, then Assumption 6 holds true.

Supposing otherwise, there exists a nonzero $v \in \mathbb{R}^d$ such that $v^\mathsf{T}\Sigma_\phi v = 0$. Therefore, $\mathbb{E}((z \cdot \phi(x))^2) = 0$ which implies that $z \cdot \phi(x) = 0$, almost surely. Define $S \equiv \{z \in \mathbb{R}^d : z \cdot \phi(x) = 0\}$. Space S is $(d-1)$-dimensional and all the points in $\phi(\mathcal{X})$ belong to S almost surely, i.e., $\mathbb{P}(\phi(\mathcal{X}) \cap S^c) = 0$. However, since Σ is positive definite (with all of its eigenvalues target than $C_{\min}$, by Assumption 1), $\mathbb{P}_X(S) = 0$. Combining these observations, $\mathbb{P}(\phi(\mathcal{X})) \leq \mathbb{P}(S) + \mathbb{P}(\phi(\mathcal{X}) \cap S^c) = 0$. Since ϕ has the 0-property, this implies that $\mathbb{P}_X(\mathcal{X}) = 0$, which is a contradiction because $\mathcal{X}$ is the support of $\mathbb{P}_X$ and thus $\mathbb{P}_X(\mathcal{X}) = 1$. The result follows.

C.7. Proof of Lemma 21

Under model (27), a purchase occurs at time t with the posted price price p if $\tilde{v}_t \geq \beta_0 p$. This is equivalent to $\tilde{z}_t \geq \beta_0 p - x_t \cdot \mu_0$. Therefore, the expected revenue from a posted price p is given by

$$p \times \mathbb{P}(\tilde{v}_t \geq \beta_0 p) = p(1 - F(\beta_0 p - \mu_0 \cdot x_t)). \tag{138}$$

By setting the first order conditions, the optimal price p_t^* is given by the solution of the following equation:

$$\beta_0 p_t^* = \frac{1 - F(\beta_0 p_t^* - \mu_0 \cdot x_t)}{f(\beta_0 p_t^* - \mu_0 \cdot x_t)}. \tag{139}$$

It is straightforward to verify that the solution p_t^* of the above equation is given by $p_t^* = (1/\beta_0) g(\mu_0 \cdot x_t)$.

C.8. Proof of Lemma 15

Recall the notation $\|E^{(k)}\|_\infty = \max_{i,j} |E_{ij}^{(k)}|$. Note that

$$\langle v, E^{(k)} v\rangle = \sum_{i,j=1}^{d+1} |E_{ij}^{(k)}||v_i||v_j| \leq \|E^{(k)}\|_\infty \sum_{i,j=1}^{d+1} |v_i|\,|v_j| \leq \|E^{(k)}\|_\infty \left(\sum_{i=1}^{d} |v_i|\right)^2 = \|E^{(k)}\|_\infty \|v\|_1^2. \tag{140}$$

Therefore, we only need to bound $\|E^{(k)}\|_\infty$. Fix $1 \leq i, j \leq d+1$. We then have

$$E_{ij}^{(k)} = \Sigma_{ij}^{(k)} - S_{ij}^{(k)} = \frac{1}{\tau_k} \sum_{\ell=1}^{\tau_k} \left\{ \mathbb{E}(X_{\ell i} X_{\ell j}) - X_{\ell i} X_{\ell j} \right\}. \tag{141}$$

Let $u_\ell^{(ij)} = X_{\ell i} X_{\ell j}$. Then, $|u_\ell^{(ij)}| \leq 1$ because $\|x_\ell\|_\infty \leq 1$. By applying Hoeffding's inequality,

$$\mathbb{P}\left(|E_{ij}^{(k)}| \geq 3\sqrt{\frac{\log d}{\tau_k}} \right) \leq \frac{2}{d^4}. \tag{142}$$

Therefore, by union bonding over all indices $1 \leq i, j \leq d+1$, we obtain that $\|E^{(k)}\|_\infty \leq 3\sqrt{(\log d)/\tau_k}$, with probability at least $1 - 8/d^2$. The claim follows from this result along with (140).

C.9. Proof of Lemma 20

Define parameter vectors $\eta_{0,0} = (\mu_0; \beta_0)$, $\eta_{0,\wedge} = (\mu_0; \widehat{\beta}^k)$, $\eta_{\wedge,0} = (\widehat{\mu}^k; \beta_0)$, and $\eta_{\wedge,\wedge} = (\widehat{\mu}^k; \widehat{\beta}^k)$ all in $\mathbb{R}^{d+1}$. By optimality of $\eta_{\wedge,\wedge}$, we have $\nabla_\mu \mathcal{L}(\widehat{\mu}^k) + \lambda_k \nu^k = 0$ and $\nabla_\beta \mathcal{L}(\widehat{\beta}^k) + \lambda_k \rho^k = 0$, where ν^k is a subgradient of ℓ_1 norm at $\widehat{\mu}^k$ and ρ^k is a subgradient of the absolute value function at $\widehat{\beta}^k$. Therefore,

$$\begin{aligned} \nabla\mathcal{L}(\eta_{0,\wedge}) &= (\nabla_\mu \mathcal{L}(\mu_0); \nabla_\beta \mathcal{L}(\widehat{\beta}^k)) = (\nabla_\mu \mathcal{L}(\mu_0); -\lambda_k \rho^k), \\ \nabla\mathcal{L}(\eta_{\wedge,0}) &= (\nabla_\mu \mathcal{L}(\widehat{\mu}^k); \nabla_\beta \mathcal{L}(\beta_0)) = (-\lambda_k \nu^k; \nabla_\mu \mathcal{L}(\beta_0)). \end{aligned} \tag{143}$$

By a similar argument following (88), we have $\|\nabla\mathcal{L}(\eta_{0,0})\|_\infty \leq \lambda_k/2$, with probability at least $1 - 1/d$. Hence, invoking relations (143), we get

$$\|\nabla\mathcal{L}(\eta_{0,\wedge})\|_\infty \leq \lambda_k, \quad \|\nabla\mathcal{L}(\eta_{\wedge,0})\|_\infty \leq \lambda_k. \tag{144}$$

By optimality of $\eta_{\wedge,\wedge}$, we have

$$\mathcal{L}(\eta_{\wedge,\wedge}) + \lambda_k \|\eta_{\wedge,\wedge}\|_1 \leq \mathcal{L}(\eta_{0,\wedge}) + \lambda_k \|\eta_{0,\wedge}\|_1.$$

Following a similar argument to (91), we have

$$\frac{2\ell_F}{\tau_{k-1}} \|\widetilde{X}^{(k-1)}(\eta_{0,\wedge} - \eta_{\wedge,\wedge})\|^2 + 2\lambda_k \|\eta_{\wedge,\wedge}\|_1 \leq 2\lambda_k \|\eta_{\wedge,\wedge} - \eta_{0,\wedge}\|_1 + 2\lambda_k \|\eta_{0,\wedge}\|_1.$$

Applying triangle inequality and rearranging the terms, this yields to the following inequality:

$$\frac{2\ell_F}{\tau_{k-1}}\|\widetilde{X}^{(k-1)}(\eta_{0,\wedge}-\eta_{\wedge,\wedge})\|^2 \le 4\lambda_k\|\eta_{\wedge,\wedge}-\eta_{0,\wedge}\|_1 \le 8W\lambda_k\,. \tag{145}$$

By substituting for $\eta_{0,\wedge}$ and $\eta_{\wedge,\wedge}$, we get Equation (79).

Likewise, we can write (145) with $\eta_{\wedge,0}$ instead of $\eta_{0,\wedge}$:

$$\frac{2\ell_F}{\tau_{k-1}}\|\widetilde{X}^{(k-1)}(\eta_{\wedge,0}-\eta_{\wedge,\wedge})\|^2 \le 8W\lambda_k\,. \tag{146}$$

By substituting for $\eta_{\wedge,0}$ and $\eta_{0,0}$ we obtain

$$\frac{2\ell_F}{\tau_{k-1}}(\beta_0-\widehat{\beta}^k)^2\|p^{(k-1)}\|^2 = \frac{2\ell_F}{\tau_{k-1}}\|\widetilde{X}^{(k-1)}(\eta_{\wedge,0}-\eta_{\wedge,\wedge})\|^2 \le 8W\lambda_k\,, \tag{147}$$

where $p^{(k)}=(p_{\tau_{k-1}},\ldots,p_{\tau_k-1})^{\mathsf{T}}$ is the vector of prices offered in episode $(k-1)$. Hence, in order to prove (79), it suffices to show that $p_t \ge \tilde{c}_W$ for all t. To see this, note that since $g(v)>0$ it attains its minimum over any bounded interval. Therefore, there exists a constant $c'_W>0$, such that $g(v)>c'_W$ for $v\in[-W,W]$. Further, by the constraint in the optimization (28) we have $|\widehat{\beta}^{(k-1)}|\le W$. Combining these two facts, we get

$$p_t = \frac{1}{\widehat{\beta}^{(k-1)}}g(x_t\cdot\widehat{\mu}^{(k-1)}) \ge \frac{c'_W}{W} \equiv \tilde{c}_W\,. \tag{148}$$

The result follows from (147) and setting $c_W = 4W/\tilde{c}_W^2$.

References

Yasin Abbasi-Yadkori, David Pal, and Csaba Szepesvari. Online-to-confidence-set conversions and application to sparse stochastic bandits. In *AISTATS*, pages 1–9, 2012.

Shipra Agrawal and Nikhil R. Devanur. Bandits with concave rewards and convex knapsacks. In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, EC '14, pages 989–1006, 2014.

Airbnb Documentation. Smart pricing: Set prices based on demand. `https://www.airbnb.com/help/article/1168/smart-pricing--set-prices-based-on-demand`, 2015.

Kareem Amin, Afshin Rostamizadeh, and Umar Syed. Repeated contextual auctions with strategic buyers. In *Advances in Neural Information Processing Systems*, pages 622–630, 2014.

Victor F Araman and René Caldentey. Dynamic pricing for nonperishable products with demand learning. *Operations research*, 57(5):1169–1188, 2009.

Moshe Babaioff, Shaddin Dughmi, Robert Kleinberg, and Aleksandrs Slivkins. Dynamic pricing with limited supply. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, EC '12, pages 74–91, 2012. ISBN 978-1-4503-1415-2.

Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 207–216. IEEE, 2013.

Mark Bagnoli and Ted Bergstrom. Log-concave probability and its applications. *Economic theory*, 26(2):445–469, 2005.

Santiago R Balseiro, Jon Feldman, Vahab Mirrokni, and S Muthukrishnan. Yield optimization of display advertising with ad exchange. *Management Science*, 60(12):2886–2907, 2014.

Hamsa Bastani and Mohsen Bayati. Online decision-making with high-dimensional covariates. Working Paper, 2016.

Omar Besbes and Assaf Zeevi. Dynamic pricing without knowing the demand function: risk bounds and near-optimal algorithms. *Operations Research*, 57:1407–1420, 2009.

Sonia A Bhaskar and Adel Javanmard. 1-bit matrix completion under exact low-rank constraint. In *Information Sciences and Systems (CISS), 2015 49th Annual Conference on*, pages 1–6. IEEE, 2015.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Josef Broder and Paat Rusmevichientong. Dynamic pricing under a general parametric choice model. *Operations Research*, 60(4):965–980, 2012.

Peter Bühlmann and Sara van de Geer. *Statistics for high-dimensional data*. Springer-Verlag, 2011.

Emmanuel Candes and Terence Tao. The dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, pages 2313–2351, 2007.

Xi Chen, Zachary Owen, Clark Pixton, and David Simchi-Levi. A statistical learning approach to personalization in revenue management. Working Paper, 2015.

Wang Chi Cheung, Will Ma, David Simchi-Levi, and Xinshang Wang. Inventory balancing with online learning. *Working Paper*, 2018.

Maxime C Cohen, Ilan Lobel, and Renato Paes Leme. Feature-based dynamic pricing. *ACM Conference on Economics and Computation*, 2016.

Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. In *COLT*, pages 355–366, 2008.

A. V. den Boer and A. P. Zwart. Mean square convergence rates for maximum(quasi) likelihood estimation. *Stochastic systems*, 4:1 – 29, 2014. ISSN 1946-5238.

Arnoud V den Boer. Dynamic pricing and learning: historical origins, current research, and new directions. *Surveys in operations research and management science*, 20(1):1–18, 2015.

Arnoud V den Boer and Bert Zwart. Simultaneously learning and optimizing using controlled variance pricing. *Management Science*, 60(3):770–783, 2013.

Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *The American economic review*, 97(1):242–259, 2007.

Ossama Elshiewy, Daniel Guhl, and Yasemin Boztug. Multinomial logit models in marketing-from fundamentals to state-of-the-art. *Marketing ZFP*, 39(3):32–49, 2017.

Vivek F Farias, Srikanth Jagabathula, and Devavrat Shah. A nonparametric approach to modeling choice with limited data. *Management Science*, 59(2):305–322, 2013.

Kris Johnson Ferreira, David Simchi-Levi, and He Wang. Online network revenue management using thompson sampling. *Operations research*, 2018.

Dennis H Gensch and Wilfred W Recker. The multinomial, multiattribute logit choice model. *Journal of Marketing Research*, pages 124–132, 1979.

Alexander Goldenshluger and Assaf Zeevi. A linear response bandit problem. *Stochastic Systems*, 3(1):230–261, 2013.

Negin Golrezaei, Hamid Nazerzadeh, and Paat Rusmevichientong. Real-time optimization of personalized assortments. *Management Science*, 60(6):1532–1551, 2014.

Negin Golrezaei, Adel Javanmard, and Vahab Mirrokni. Dynamic incentive-aware learning: Robust pricing in contextual auctions. *http://dx.doi.org/10.2139/ssrn.3144034*, 2018.

Peter M Guadagni and John DC Little. A logit model of brand choice calibrated on scanner data. *Marketing Science*, 27(1):29–48, 2008.

J Michael Harrison, Bora Keskin, and Assaf Zeevi. Bayesian dynamic pricing policies: Learning and earning under a binary prior distribution. *Management Science*, 58(3): 570–586, 2012.

John R Hauser and Frank S Koppelman. Alternative perceptual mapping techniques: Relative accuracy and usefulness. *Journal of marketing Research*, pages 495–506, 1979.

Adel Javanmard. Perishability of data: Dynamic pricing under varying-coefficient models. *Journal of Machine Learning Research*, 18(53):1–31, 2017. URL `http://jmlr.org/papers/v18/17-061.html`.

Nathan Kallus and Madeleine Udell. Dynamic assortment personalization in high dimensions. Working Paper, 2016.

Godfrey Keller and Sven Rady. Optimal experimentation in a changing environment. *The review of economic studies*, 66(3):475–507, 1999.

Bora Keskin. Optimal dynamic pricing with demand model uncertainty: A squared-coefficient-of-variation rule for learning and earning. Working Paper, 2014.

Bora Keskin and Assaf Zeevi. Dynamic pricing with an unknown demand model: Asymptotically optimal semi-myopic policies. *Operations Research*, 62(5):1142–1167, 2014.

N Bora Keskin and Assaf Zeevi. Chasing demand: Learning and earning in a changing environment. *Mathematics of Operations Research*, 2016.

Robert Kleinberg and Tom Leighton. The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In *Proceedings of 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 594–605. IEEE, 2003.

Sébastien Lahaie and David M Pennock. Revenue analysis of a family of ranking rules for keyword auctions. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 50–56. ACM, 2007.

Jordan J Louviere and George Woodworth. Design and analysis of simulated consumer choice or allocation experiments: an approach based on aggregate data. *Journal of marketing research*, pages 350–367, 1983.

Daniel McFadden. Economic choices. *American economic review*, 91(3):351–378, 2001.

Roger B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1): 58–73, 1981.

Yaniv Plan and Roman Vershynin. One-bit compressed sensing by linear programming. *Communications on Pure and Applied Mathematics*, 66(8):1275–1297, 2013.

Stanislav P Ponomarev. Submersions and preimages of sets of measure zero. *Siberian Mathematical Journal*, 28(1):153–163, 1987.

Sheng Qiang and Mohsen Bayati. Dynamic pricing with demand covariates. Working Paper, 2016.

Garvesh Raskutti, Martin J Wainwright, and Bin Yu. Minimax rates of estimation for high-dimensional linear regression over-balls. *IEEE Transactions on Information Theory*, 57 (10):6976–6994, 2011.

Michael Rothschild. A two-armed bandit theory of market pricing. *Journal of Economic Theory*, 9(2):185–202, 1974.

Mark Rudelson and Shuheng Zhou. Reconstruction from anisotropic random measurements. *IEEE Trans. on Inform. Theory*, 59(6):3434–3447, 2013.

Kalyan T Talluri and Garrett J Van Ryzin. *The theory and practice of revenue management*, volume 68. Springer Science & Business Media, 2006.

A.B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Series in Statistics. Springer New York, 2008. ISBN 9780387790527. URL `https://books.google.com/books?id=mwB8rUBsbqoC`.

Sara A Van de Geer. High-dimensional generalized linear models and the lasso. *The Annals of Statistics*, pages 614–645, 2008.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Zizhuo Wang, Shiming Deng, and Yinyu Ye. Close the gaps: A learning-while-doing algorithm for single-product revenue management problems. *Operations Research*, 62(2): 318–331, 2014.

Baichun Xiao, Wei Yang, and Jun Li. Optimal reserve price for the generalized second-price auction in sponsored search advertising. *Journal of Electronic Commerce Research*, 10 (3):114, 2009.

Yuhong Yang and Andrew Barron. Information-theoretic determination of minimax rates of convergence. *Annals of Statistics*, pages 1564–1599, 1999.

Bin Yu. Assouad, fano and le, cam. In *Research Papers in Probability and Statistics: Festschrift in Honor of Lucien Le Cam*, pages 423–435. Springer-Verlag, 1997.

Journal of Machine Learning Research 20 (2019) 1-44 Submitted 8/17; revised 11/18; Published 2/19

Graphical Lasso and Thresholding: Equivalence and Closed-form Solutions

Salar Fattahi, FATTAHI@BERKELEY.EDU
Department of Industrial Engineering and Operations Research
University of California, Berkeley

Somayeh Sojoudi, SOJOUDI@BERKELEY.EDU
Departments of Electrical Engineering and Computer Sciences and Mechanical Engineering
University of California, Berkeley

Editor: Amir Globerson

Abstract

Graphical Lasso (GL) is a popular method for learning the structure of an undirected graphical model, which is based on an l_1 regularization technique. The objective of this paper is to compare the computationally-heavy GL technique with a numerically-cheap heuristic method that is based on simply thresholding the sample covariance matrix. To this end, two notions of sign-consistent and inverse-consistent matrices are developed, and then it is shown that the thresholding and GL methods are equivalent if: (i) the thresholded sample covariance matrix is both sign-consistent and inverse-consistent, and (ii) the gap between the largest thresholded and the smallest un-thresholded entries of the sample covariance matrix is not too small. By building upon this result, it is proved that the GL method—as a conic optimization problem—has an explicit closed-form solution if the thresholded sample covariance matrix has an acyclic structure. This result is then generalized to arbitrary sparse support graphs, where a formula is found to obtain an approximate solution of GL. Furthermore, it is shown that the approximation error of the derived explicit formula decreases exponentially fast with respect to the length of the minimum-length cycle of the sparsity graph. The developed results are demonstrated on synthetic data, functional MRI data, traffic flows for transportation networks, and massive randomly generated data sets. We show that the proposed method can obtain an accurate approximation of the GL for instances with the sizes as large as $80,000 \times 80,000$ (more than 3.2 billion variables) in less than 30 minutes on a standard laptop computer running MATLAB, while other state-of-the-art methods do not converge within 4 hours.

Keywords: Graphical Lasso, Graphical Model, Sparse Graphs, Optimization

1. Introduction

There has been a pressing need in developing new and efficient computational methods to analyze and learn the characteristics of high-dimensional data with a structured or randomized nature. Real-world data sets are often overwhelmingly complex, and therefore it is important to obtain a simple description of the data that can be processed efficiently. In an effort to address this problem, there has been a great deal of interest in sparsity-promoting techniques for large-scale optimization problems (Coleman and Li 1990; Bach et al. 2012; Benson et al. 2000). These techniques have become essential to the tractability of big-data analyses in many applications, including data mining (Garcke et al. 2001; Muthukrishnan 2005; Wu et al. 2014), pattern recognition (Wright et al. 2010; Qiao et al. 2010), human brain functional connectivity (Sojoudi and Doyle 2014), distributed controller design (Fardad et al. 2011; Fattahi and Lavaei 2017), and compressive sensing (Candes and Romberg 2007; Foucart and Rauhut 2013). Similar approaches have been used to arrive at a parsimonious estimation of high-dimensional data. However, most of the existing statistical learning techniques in data analytics are contingent upon the availability of a sufficient number of samples (compared to the number of parameters), which is difficult to satisfy for many applications (Bühlmann and Van De Geer 2011; Fan and Lv 2010). To remedy the aforementioned issues, a special attention has been paid to the augmentation of these problems with sparsity-inducing penalty functions to obtain sparse and easy-to-analyze solutions.

Graphical lasso (GL) is one of the most commonly used techniques for estimating the inverse covariance matrix (Friedman et al. 2008; Banerjee et al. 2008; Yuan and Lin 2007). GL is an optimization problem that shrinks the elements of the inverse covariance matrix towards zero compared to the maximum likelihood estimates, using an l_1 regularization. There is a large body of literature suggesting that the solution of GL is a good estimate for the unknown graphical model, under a suitable choice of the regularization parameter (Friedman et al. 2008; Banerjee et al. 2008; Yuan and Lin 2007; Liu et al. 2010; Krämer et al. 2009; Danaher et al. 2014). It is known that Graphical Lasso is computationally expensive for large-scale problems. An alternative computationally-cheap heuristic method for estimating graphical models is based on thresholding the sample covariance matrix.

In this paper, we develop a mathematical framework to analyze the relationship between the GL and thresholding techniques. The paper Sojoudi (2016) offers a set of conditions for the equivalence of these two methods, and argues the satisfaction of these conditions in the case where the regularization coefficient is large or equivalently a sparse graph is sought. Although the conditions derived in Sojoudi (2016) shed light on the performance of the GL, they depend on the optimal solution of the GL and cannot be verified without solving the problem. Nonetheless, it is highly desirable to find conditions for the equivalence of the GL and thresholding that are directly in terms of the sample covariance matrix. To this end, two notions of *sign-consistent* and *inverse-consistent* matrices are introduced, and their properties are studied for different types of matrices. It is then shown that the GL and thresholding are equivalent if three conditions are satisfied. The first condition requires a certain matrix formed based on the sample covariance matrix to have a positive-definite completion. The second condition requires this matrix to be sign-consistent and inverse-consistent. The third condition needs a separation between the largest thresholded and the smallest un-thresholded entries of the sample covariance matrix. These conditions can be easily verified for acyclic graphs and are expected to hold for sparse graphs. By building upon these results, an explicit closed-form solution is obtained for the GL method in the case where the thresholded sample covariance matrix has an acyclic support graph. Furthermore, this result is generalized to sparse support graphs to derive a closed-form formula that can serve either as an approximate solution of the GL or the optimal solution of the GL with a perturbed sample covariance matrix. The approximation error (together with the corresponding perturbation in the sample covariance matrix) is shown to be related to the lengths of the cycles in the graph.

©2019 Salar Fattahi and Somayeh Sojoudi.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-501.html.

The remainder of this paper is organized as follows. The main results are presented in Section 3, followed by numerical examples and case studies in Section 4. Concluding remarks are drawn in Section 5. Most of the technical proofs are provided in Appendix.

Notations: Lowercase, bold lowercase and uppercase letters are used for scalars, vectors and matrices, respectively (say $x, \mathbf{x}, X$). The symbols $\mathbb{R}^d$, $\mathbb{S}^d$ and $\mathbb{S}^d_+$ are used to denote the sets of $d\times 1$ real vectors, $d\times d$ symmetric matrices and $d\times d$ symmetric positive-semidefinite matrices, respectively. The notations $\mathrm{trace}(M)$ and $\log\det(M)$ refer to the trace and the logarithm of the determinant of a matrix M, respectively. The $(i,j)^{\text{th}}$ entry of the matrix M is denoted by M_{ij}. Moreover, I_d denotes the $d\times d$ identity matrix. The sign of a scalar x is shown as $\mathrm{sign}(x)$. The notations $|x|$, $\|M\|_1$ and $\|M\|_F$ denote the absolute value of the scalar x, the induced norm-1 and Frobenius norm of the matrix M, respectively. The inequalities $M\succeq 0$ and $M\succ 0$ mean that M is positive-semidefinite and positive-definite, respectively. The symbol $\mathrm{sign}(\cdot)$ shows the sign operator. The ceiling function is denoted as $\lceil\cdot\rceil$. The cardinality of a discrete set $\mathcal{D}$ is denoted as $|\mathcal{D}|_0$. Given a matrix $M\in\mathbb{S}^d$, define

$$\|M\|_{1,\text{off}} = \sum_{i=1}^{d}\sum_{j=1}^{d}|M_{ij}| - \sum_{i=1}^{d}|M_{ii}|,$$
$$\|M\|_{\max} = \max_{i\neq j}|M_{ij}|.$$

Definition 1 *Given a symmetric matrix $S\in\mathbb{S}^d$, the* **support graph or sparsity graph** *of S is defined as a graph with the vertex set $\mathcal{V}:=\{1,2,...,d\}$ and the edge set $\mathcal{E}\subseteq\mathcal{V}\times\mathcal{V}$ such that $(i,j)\in\mathcal{V}$ if and only if $S_{ij}\neq 0$, for every two different vertices $i,j\in\mathcal{V}$. The support graph of S captures the sparsity pattern of the matrix S and is denoted as* $\mathrm{supp}(S)$.

Definition 2 *Given a graph $\mathcal{G}$, define $\mathcal{G}^{(c)}$ as the complement of $\mathcal{G}$, which is obtained by removing the existing edges of $\mathcal{G}$ and drawing an edge between every two vertices of $\mathcal{G}$ that were not originally connected.*

Definition 3 *Given two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ with the same vertex set, $\mathcal{G}_1$ is called a subgraph of $\mathcal{G}_2$ if the edge set of $\mathcal{G}_1$ is a subset of the edge set of $\mathcal{G}_2$. The notation $\mathcal{G}_1\subseteq\mathcal{G}_2$ is used to denote this inclusion.*

Finally, a symmetric matrix M is said to have a **positive-definite completion** if there exists a positive-definite $\tilde{M}$ with the same size such that $\tilde{M}_{ij}=M_{ij}$ for every $(i,j)\in\mathrm{supp}(M)$.

2. Problem Formulation

Consider a random vector $\mathbf{x}=(x_1,x_2,...,x_d)$ with a multivariate normal distribution. Let $\Sigma_*\in\mathbb{S}^d_+$ denote the covariance matrix associated with the vector $\mathbf{x}$. The inverse of the covariance matrix can be used to determine the conditional independence between the

random variables $x_1,x_2,...,x_d$. In particular, if the $(i,j)^{\text{th}}$ entry of Σ_*^{-1} is zero for two disparate indices i and j, then x_i and x_j are conditionally independent given the rest of the variables. The graph $\mathrm{supp}\big(\Sigma_*^{-1}\big)$ (i.e., the sparsity graph of Σ_*^{-1}) represents a graphical model capturing the conditional independence between the elements of $\mathbf{x}$. Assume that Σ_* is nonsingular and that $\mathrm{supp}\big(\Sigma_*^{-1}\big)$ is a sparse graph. Finding this graph is cumbersome in practice because the exact covariance matrix Σ_* is rarely known. More precisely, $\mathrm{supp}\big(\Sigma_*^{-1}\big)$ should be constructed from a given sample covariance matrix (constructed from n samples), as opposed to Σ_*. Let Σ denote an arbitrary $d\times d$ positive-semidefinite matrix, which is provided as an estimate of Σ_*. Consider the convex optimization problem

$$\min_{S\in\mathbb{S}^d_+} -\log\det(S)+\mathrm{trace}(\Sigma S). \tag{1}$$

It is easy to verify that the optimal solution of the above problem is equal to $S^{\text{opt}}=\Sigma^{-1}$. However, there are two issues with this solution. First, since the number of samples available in many applications is small or modest compared to the dimension of Σ, the matrix Σ is ill-conditioned or even singular. Under such circumstances, the equation $S^{\text{opt}}=\Sigma^{-1}$ leads to large or undefined entries for the optimal solution of (1). Second, although Σ_*^{-1} is assumed to be sparse, a small random difference between Σ_* and Σ would make S^{opt} highly dense. In order to address the aforementioned issues, consider the problem

$$\min_{S\in\mathbb{S}^d_+} -\log\det(S)+\mathrm{trace}(\Sigma S)+\lambda\|S\|_{1,\text{off}}, \tag{2}$$

where $\lambda\in\mathbb{R}_+$ is a regularization parameter. This problem is referred to as *Graphical Lasso* (GL). Intuitively, the term $\|S\|_{1,\text{off}}$ in the objective function serves as a surrogate for promoting sparsity among the off-diagonal entries of S, while ensuring that the problem is well-defined even with a singular input Σ. Henceforth, the notation S^{opt} will be used to denote a solution of the GL instead of the unregularized optimization problem (1).

Suppose that it is known *a priori* that the true graph $\mathrm{supp}\big(\Sigma_*^{-1}\big)$ has k edges, for some given number k. With no loss of generality, assume that all nonzero off-diagonal entries of Σ have different magnitudes. Two heuristic methods for finding an estimate of $\mathrm{supp}\big(\Sigma_*^{-1}\big)$ are as follows:

- **Graphical Lasso:** We solve the optimization problem (2) repeatedly for different values of λ until a solution S^{opt} with exactly $2k$ nonzero off-diagonal entries are found.
- **Thresholding:** Without solving any optimization problem, we simply identify those $2k$ entries of Σ that have the largest magnitudes among all off-diagonal entries of Σ. We then replace the remaining d^2-d-2k off-diagonal entries of Σ with zero and denote the thresholded sample covariance matrix as Σ_k. Note that Σ and Σ_k have the same diagonal entries. Finally, we consider the sparsity graph of Σ_k, namely $\mathrm{supp}(\Sigma_k)$, as an estimate for $\mathrm{supp}\big(\Sigma_*^{-1}\big)$.

Definition 4 *It is said that the* **sparsity structures of Graphical Lasso and thresholding are equivalent** *if there exists a regularization coefficient λ such that* $\mathrm{supp}(S^{opt})=\mathrm{supp}(\Sigma_k)$.

Recently, we have verified in several simulations that the GL and thresholding are equivalent for electrical circuits and functional MRI data of 20 subjects, provided that k is on the order of n (Sojoudi 2016). This implies that a simple thresholding technique would obtain the same sparsity structure as the computationally-heavy GL technique. In this paper, it is aimed to understand under what conditions the easy-to-find graph $\text{supp}(\Sigma_k)$ is equal to the hard-to-obtain graph $\text{supp}(S^{\text{opt}})$, without having to solve the GL. Furthermore, we will show that the GL problem has a simple closed-form solution that can be easily derived merely based on the thresholded sample covariance matrix, provided that its underlying graph has an acyclic structure. This result will then be generalized to obtain an approximate solution for the GL in the case where the thresholded sample covariance matrix has an arbitrary sparsity structure. This closed-form solution converges to the exact solution of the GL as the length of the minimum-length cycle in the support graph of the thresholded sample covariance matrix grows. The derived closed-form solution can be used for two purposes: (1) as a surrogate to the exact solution of the computationally heavy GL problem, and (2) as an initial point for common numerical algorithms to numerically solve the GL (see Friedman et al. (2008); Hsieh et al. (2014)). The above results unveil fundamental properties of the GL in terms of sparsification and computational complexity. Although conic optimization problems almost never benefit from an exact or inexact explicit formula for their solutions and should be solved numerically, the formula obtained in this paper suggests that sparse GL and related graph-based conic optimization problems may fall into the category of problems with closed-form solutions (similar to least squares problems).

3. Main Results

In this section, we present the main results of the paper. In order to streamline the presentation, most of the technical proofs are postponed to Appendix.

3.1. Equivalence of GL and Thresholding

In this subsection, we derive sufficient conditions to guarantee that the GL and thresholding methods result in the same sparsity graph. These conditions are only dependent on λ and Σ, and are expected to hold whenever λ is large enough or a sparse graph is sought.

Definition 5 *A matrix $M \in \mathbb{S}^d$ is called* **inverse-consistent** *if there exists a matrix $N \in \mathbb{S}^d$ with zero diagonal elements such that*

$$\begin{aligned} &M + N \succ 0, \\ &\text{supp}(N) \subseteq (\text{supp}(M))^{(c)}, \\ &\text{supp}\left((M+N)^{-1}\right) \subseteq \text{supp}(M). \end{aligned}$$

The matrix N is called **inverse-consistent complement** *of M and is denoted as $M^{(c)}$.*

The next Lemma will shed light on the definition of inverse-consistency by introducing an important class of such matrices that satisfy this property, namely the *set of matrices with positive-definite completions.*

Lemma 6 *Any arbitrary matrix with positive-definite completion is inverse-consistent and has a unique inverse-consistent complement.*

Proof: Consider the optimization problem

$$\begin{aligned} \min_{S \in \mathbb{S}^n} \quad & \text{trace}(MS) - \text{logdet}(S) && \text{(4a)} \\ \text{subject to} \quad & S_{ij} = 0, \qquad \forall (i,j) \in (\text{supp}(M))^{(c)} && \text{(4b)} \\ & S \succeq 0, && \text{(4c)} \end{aligned}$$

and its dual

$$\begin{aligned} \max_{\Pi \in \mathbb{S}^n} \quad & \det(M + \Pi) && \text{(5a)} \\ \text{subject to} \quad & M + \Pi \succeq 0 && \text{(5b)} \\ & \text{supp}(\Pi) \subseteq (\text{supp(M)})^{(c)} && \text{(5c)} \\ & \Pi_{ii} = 0, \qquad i = 1, ..., d. && \text{(5d)} \end{aligned}$$

Note that Π_{ij} is equal to the Lagrange multiplier for (4b) and every $(i,j) \in (\text{supp}(M))^{(c)}$, and is zero otherwise. Since the matrix M has a positive-definite completion, the dual problem is strictly feasible. Moreover, $S = I_d$ is a feasible solution of (4). Therefore, strong duality holds and the primal solution is attainable. On the other hand, the objective function (4a) is strictly convex, which makes the solution of the primal problem unique. Let S^{opt} denote the globally optimal solution of (4). It follows from the first-order optimality conditions that

$$S^{\text{opt}} = (M + \Pi^{\text{opt}})^{-1}.$$

This implies that

$$\begin{aligned} &\text{supp}(\Pi^{\text{opt}}) \subseteq (\text{supp}(M))^{(c)} \\ &\text{supp}((M + \Pi^{\text{opt}})^{-1}) \subseteq \text{supp}(M) \\ &M + \Pi^{\text{opt}} \succ 0. \end{aligned}$$

As a result, $M \in \mathbb{S}^d$ is inverse-consistent and Π^{opt} is its complement. To prove the uniqueness of the inverse-consistent complement of M, let Π denote an arbitrary complement of M. It follows from Definition 5 and the first-order optimality conditions that $(M+\Pi)^{-1}$ is a solution of (4). Since S^{opt} is the unique solution of (4), it can be concluded that $\Pi = \Pi^{\text{opt}}$. This implies that M has a unique inverse-consistent complement. ■

Remark 7 *Two observations can be made based on Lemma 6. First, the positive-definiteness of a matrix is sufficient to guarantee that it belongs to the cone of matrices with positive-definite completion. Therefore, positive-definite matrices are inverse-consistent. Second, upon existence, the inverse-consistent complement of a matrix with positive-definite completion is equal to the difference between the matrix and its unique* ***maximum determinant completion****.*

Definition 8 *An inverse-consistent matrix M is called* **sign-consistent** *if the (i,j) entries of M and $(M+M^{(c)})^{-1}$ are nonzero and have opposite signs for every $(i,j) \in \text{supp}(M)$.*

Example 1 (An inverse- and sign-consistent matrix) To illustrate Definitions 5 and 8, consider the matrix

$$M = \begin{bmatrix} 1 & 0.3 & 0 & 0 \\ 0.3 & 1 & -0.4 & 0 \\ 0 & -0.4 & 1 & 0.2 \\ 0 & 0 & 0.2 & 1 \end{bmatrix}.$$

The graph $\text{supp}(M)$ is a path graph with the vertex set $\{1,2,3,4\}$ and the edge set $\{(1,2),(2,3),(3,4)\}$. To show that M is inverse-consistent, let the matrix $M^{(c)}$ be chosen as

$$M^{(c)} = \begin{bmatrix} 0 & 0 & -0.120 & -0.024 \\ 0 & 0 & 0 & -0.080 \\ -0.120 & 0 & 0 & 0 \\ -0.024 & -0.080 & 0 & 0 \end{bmatrix}.$$

The inverse matrix $(M+M^{(c)})^{-1}$ is equal to

$$\begin{bmatrix} \frac{1}{0.91} & \frac{-0.3}{0.91} & 0 & 0 \\ \frac{-0.3}{0.91} & 1+\frac{0.09}{0.91}+\frac{0.16}{0.84} & \frac{0.4}{0.84} & 0 \\ 0 & \frac{0.4}{0.84} & 1+\frac{0.16}{0.84}+\frac{0.04}{0.96} & \frac{-0.2}{0.96} \\ 0 & 0 & \frac{-0.2}{0.96} & \frac{1}{0.96} \end{bmatrix}.$$

Observe that:

- M and $M+M^{(c)}$ are both positive-definite.
- The sparsity graphs of M and $M^{(c)}$ are complements of each other.
- The sparsity graphs of M and $(M+M^{(c)})^{-1}$ are identical.
- The nonzero off-diagonal entries of M and $(M+M^{(c)})^{-1}$ have opposite signs.

The above properties imply that M is both inverse-consistent and sign-consistent, and $M^{(c)}$ is its complement.

Definition 9 *Given a graph $\mathcal{G}$ and a scalar α, define $\beta(\mathcal{G},\alpha)$ as the maximum of $\|M^{(c)}\|_{\max}$ over all matrices M with positive-definite completions and with the diagonal entries all equal to 1 such that $\text{supp}(M) = \mathcal{G}$ and $\|M\|_{\max} \leq \alpha$.*

Consider the dual solution Π^{opt} introduced in the proof of Lemma 6 and note that it is a function of M. Roughly speaking, the function $\beta(\mathcal{G},\alpha)$ in the above definition provides an upper bound on $\|\Pi^{\text{opt}}\|_{\max}$ over all matrices M with positive-definite completions and with the diagonal entries equal to 1 such that $\text{supp}(M) = \mathcal{G}$ and $\|M\|_{\max} \leq \alpha$. As will be shown later, this function will be used as a *certificate* to verify the optimality conditions for the GL.

Since Σ_* is non-singular and we have a finite number of samples, the elements of the upper triangular part of Σ (excluding its diagonal elements) are all nonzero and distinct with probability one. Let $\sigma_1, \sigma_2, ..., \sigma_{d(d-1)/2}$ denote the absolute values of those upper-triangular entries such that

$$\sigma_1 > \sigma_2 > ... > \sigma_{d(d-1)/2} > 0.$$

Definition 10 *Consider an arbitrary positive regularization parameter λ that does not belong to the discrete set $\{\sigma_1, \sigma_2, ..., \sigma_{d(d-1)/2}\}$. Define the index k associated with λ as an integer number satisfying the relation $\lambda \in (\sigma_{k+1}, \sigma_k)$. If λ is greater than σ_1, then k is set to 0.*

Throughout this paper, the index k refers to the number introduced in Definition 10, which depends on λ.

Definition 11 *Define the* **residue of Σ relative to λ** *as a matrix $\Sigma^{\text{res}}(\lambda) \in \mathbb{S}^d$ such that the (i,j) entry of $\Sigma^{\text{res}}(\lambda)$ is equal to $\Sigma_{ij} - \lambda \times \text{sign}(\Sigma_{ij})$ if $i \neq j$ and $|\Sigma_{ij}| > \lambda$, and it is equal to 0 otherwise. Furthermore, define* **normalized residue of Σ relative to λ** *as*

$$\tilde{\Sigma}^{\text{res}}(\lambda) = D^{-1/2} \times \Sigma^{\text{res}}(\lambda) \times D^{-1/2},$$

where D is diagonal matrix with $D_{ii} = \Sigma_{ii}$ for every $i \in \{1, ..., d\}$.

Notice that $\Sigma^{\text{res}}(\lambda)$ is in fact the soft-thresholded sample covariance matrix with the threshold λ. For notational simplicity, we will use Σ^{res} or $\tilde{\Sigma}^{\text{res}}$ instead of $\Sigma^{\text{res}}(\lambda)$ or $\tilde{\Sigma}^{\text{res}}(\lambda)$ whenever the equivalence is implied by the context. One of the main theorems of this paper is presented below.

Theorem 12 *The sparsity structures of the thresholding and GL methods are equivalent if the following conditions are satisfied:*

- **Condition 1-i:** *$I_d + \tilde{\Sigma}^{\text{res}}$ has a positive-definite completion.*
- **Condition 1-ii:** *$I_d + \tilde{\Sigma}^{\text{res}}$ is sign-consistent.*
- **Condition 1-iii:** *The relation*

$$\beta\left(\text{supp}(\Sigma^{\text{res}}), \|\tilde{\Sigma}^{\text{res}}\|_{\max}\right) \leq \min_{\substack{i \neq j \\ |\Sigma_{ij}| \leq \lambda}} \frac{\lambda - |\Sigma_{ij}|}{\sqrt{\Sigma_{ii}\Sigma_{jj}}}$$

holds.

A number of observations can be made based on Theorem 12. First note that, due to Lemma 6, Condition (1-i) guarantees that $I_d + \tilde{\Sigma}^{\text{res}}$ is inverse-consistent; in fact it holds when $I_d + \tilde{\Sigma}^{\text{res}}$ itself is positive-definite. Note that the positive-definiteness of $I_d + \tilde{\Sigma}^{\text{res}}$ is guaranteed to hold if the eigenvalues of the normalized residue of the matrix Σ relative to λ are greater than -1. Recall that $\lambda \in (\sigma_{k+1}, \sigma_k)$ for some integer k and the off-diagonal entries of $I_d + \tilde{\Sigma}^{\text{res}}$ are in the range $[-1, 1]$. In the case where the number k

is significantly smaller than d^2, the residue matrix has many zero entries. Hence, the satisfaction of Condition (1-i) is expected for a large class of residue matrices; this will be verified extensively in our case studies on the real-world and synthetically generated data sets. Specifically, this condition is automatically satisfied if $I_d+\tilde{\Sigma}^{\text{res}}$ is diagonally dominant. Conditions (1-ii) and (1-iii) of Theorem 12 are harder to check. These conditions depend on the support graph of the residue matrix $\tilde{\Sigma}^{\text{res}}$ and/or how small the nonzero entries of $\tilde{\Sigma}^{\text{res}}$ are. The next two lemmas further analyze these conditions to show that they are expected to be satisfied for large λ.

Lemma 13 *Given an arbitrary graph $\mathcal{G}$, there is a strictly positive constant number $\zeta(\mathcal{G})$ such that*

$$\beta(\mathcal{G},\alpha)\leq \zeta(\mathcal{G})\alpha^2, \qquad \forall\ \alpha\in(0,1) \tag{7}$$

and therefore, Condition (1-iii) is reduced to

$$\zeta(\text{supp}(\Sigma^{\text{res}}))\times \max_{\substack{k\neq l\\ |\Sigma_{kl}|>\lambda}}\left(\frac{|\Sigma_{kl}|-\lambda}{\sqrt{\Sigma_{kk}\Sigma_{ll}}}\right)^2\leq \min_{\substack{i\neq j\\ |\Sigma_{ij}|\leq\lambda}}\frac{\lambda-|\Sigma_{ij}|}{\sqrt{\Sigma_{ii}\Sigma_{jj}}}.$$

Lemma 14 *Consider a matrix M with a positive-definite completion and with unit diagonal entries. Define $\alpha=\|M\|_{\max}$ and $\mathcal{G}=\text{supp}(M)$. There exist strictly positive constant numbers $\alpha_0(\mathcal{G})$ and $\gamma(\mathcal{G})$ such that M is sign-consistent if $\alpha\leq\alpha_0(\mathcal{G})$ and the absolute value of the off-diagonal nonzero entries of M is lower bounded by $\gamma(\mathcal{G})\alpha^2$. This implies that Condition (i-ii) is satisfied if $\|\tilde{\Sigma}^{\text{res}}\|_{\max}\leq\alpha_0(\text{supp}(\Sigma^{\text{res}}))$ and*

$$\gamma(\text{supp}(\Sigma^{\text{res}}))\times \max_{\substack{k\neq l\\ |\Sigma_{kl}|>\lambda}}\left(\frac{|\Sigma_{kl}|-\lambda}{\sqrt{\Sigma_{kk}\Sigma_{ll}}}\right)^2\leq \min_{\substack{i\neq j\\ |\Sigma_{ij}|>\lambda}}\frac{|\Sigma_{ij}|-\lambda}{\sqrt{\Sigma_{ii}\Sigma_{jj}}}. \tag{8}$$

For simplicity of notation, define $r=\frac{\max_i\Sigma_{ii}}{\min_j\Sigma_{jj}}$ and $\Sigma_{\max}=\max_i\Sigma_{ii}$. Assuming that $\|\tilde{\Sigma}^{\text{res}}\|_{\max}\leq\alpha_0(\text{supp}(\Sigma^{\text{res}}))$, Conditions (1-ii) and (1-iii) of Theorem 12 are guaranteed to be satisfied if

$$\zeta(\text{supp}(\Sigma^{\text{res}}))\leq\frac{1}{r^2}\cdot\frac{\frac{\lambda-\sigma_{k+1}}{\Sigma_{\max}}}{\left(\frac{\sigma_1-\lambda}{\Sigma_{\max}}\right)^2}, \qquad \gamma(\text{supp}(\Sigma^{\text{res}}))\leq\frac{1}{r^2}\cdot\frac{\frac{\sigma_k-\lambda}{\Sigma_{\max}}}{\left(\frac{\sigma_1-\lambda}{\Sigma_{\max}}\right)^2}, \tag{9}$$

which is equivalent to

$$\max\left\{\gamma(\text{supp}(\Sigma^{\text{res}})),\zeta(\text{supp}(\Sigma^{\text{res}}))\right\}\leq\frac{2}{r^2}\cdot\frac{\frac{\sigma_k-\sigma_{k+1}}{\Sigma_{\max}}}{\left(\frac{2\sigma_1-\sigma_k-\sigma_{k+1}}{\Sigma_{\max}}\right)^2}.$$

for the choice $\lambda=\frac{\sigma_k+\sigma_{k+1}}{2}$. Consider the set

$$\mathcal{T}=\left\{|\Sigma_{ij}|\ \middle|\ i=1,2,...,d-1,\ j=i+1,...,d\right\}.$$

This set has $\frac{d(d-1)}{2}$ elements. The cardinality of $\{\sigma_1,...,\sigma_{d-1}\}$, as a subset of $\mathcal{T}$, is smaller than the cardinality of $\mathcal{T}$ by a factor of $\frac{d}{2}$. Combined with the fact that $|\sigma_i|<\Sigma_{\max}$ for

every $i=1,...,\frac{d(d-1)}{2}$, this implies that the term $\frac{2\sigma_1-\sigma_{d-1}-\sigma_d}{\Sigma_{\max}}$ is expected to be small and its square is likely to be much smaller than 1, provided that the elements of $\mathcal{T}$ are sufficiently spread. If the number $(2\sigma_1-\sigma_{d-1}-\sigma_d)$ is relatively smaller than the gap $\sigma_{d-1}-\sigma_d$ and $k=O(d)$, then (7) and as a result Conditions (1-ii) and (1-iii) would be satisfied. The satisfaction of this condition will be studied for acyclic graphs in the next section.

3.2. Closed-form Solution: Acyclic Sparsity Graphs

In the previous subsection, we provided a set of sufficient conditions for the equivalence of the GL and thresholding methods. Although these conditions are merely based on the known parameters of the problem, i.e., the regularization coefficient and sample covariance matrix, their verification is contingent upon knowing the value of $\beta(\text{supp}(\Sigma^{\text{res}}),\|\tilde{\Sigma}^{\text{res}}\|_{\max})$ and whether $I_d+\tilde{\Sigma}^{\text{res}}$ is sign-consistent and has a positive-definite completion. The objective of this part is to greatly simplify the conditions in the case where the thresholded sample covariance matrix has an acyclic support graph. First, notice that if $I_d+\tilde{\Sigma}^{\text{res}}$ is positive-definite, it has a trivial positive-definite completion. Furthermore, we will prove that $\zeta(\text{supp}(\Sigma^{\text{res}}))$ in Lemma 13 is equal to 1 when $\text{supp}(\Sigma^{\text{res}})$ is acyclic. This reduces Condition (1-iii) to the simple inequality

$$\|\tilde{\Sigma}^{\text{res}}\|_{\max}^2\leq\min_{\substack{i\neq j\\ |\Sigma_{ij}|\leq\lambda}}\frac{\lambda-|\Sigma_{ij}|}{\sqrt{\Sigma_{ii}\Sigma_{jj}}},$$

which can be verified efficiently and is expected to hold in practice (see Section 4). Then, we will show that the sign-consistency of $I_d+\tilde{\Sigma}^{\text{res}}$ is automatically implied by the fact that it has a positive-definite completion if $\text{supp}(\Sigma^{\text{res}})$ is acyclic.

Lemma 15 *Given an arbitrary acyclic graph $\mathcal{G}$, the relation*

$$\beta(\mathcal{G},\alpha)\leq\alpha^2 \tag{10}$$

holds for every $0\leq\alpha<1$. Furthermore, strict equality holds for (10) if $\mathcal{G}$ includes a path of length at least 2.

Sketch of the Proof: In what follows, we will provide a sketch of the main idea behind the proof of Lemma 15. The detailed analysis can be found in the Appendix. Without loss of generality, one can assume that $\mathcal{G}$ is connected. Otherwise, the subsequent argument can be made for every connected component of $\mathcal{G}$. Consider a matrix M that satisfies the conditions delineated in Definition 9, i.e. 1) it has a positive-definite completion and hence, is inverse-consistent (see Lemma 6), 2) it has unit diagonal entries, 3) the absolute value of its off-diagonal elements is upper bounded by α, and 4) $\text{supp}(M)=\mathcal{G}$. The key idea behind the proof of Lemma 15 lies in the fact that, due to the acyclic structure of $\mathcal{G}$, one can explicitly characterize the inverse-consistent complement of M. In particular, it can be shown that the inverse-consistent complement of M has the following explicit formula: for every $(i,j)\notin\mathcal{G}$, $M^{(c)}_{ij}$ is equal to the multiplication of the off-diagonal elements of M corresponding to the edges in the unique path between the nodes i and j in $\mathcal{G}$. This key insight immediately results in the statement of Lemma 15: the length of the path between

nodes i and j is lower bounded by 2 and therefore, $M_{ij}^{(c)} \leq \alpha^2$. Furthermore, it is easy to see that if $\mathcal{G}$ includes a path of length at least 2, M can be chosen such that for some $(i,j) \notin \mathcal{G}$, we have $M_{ij}^{(c)} = \alpha^2$. ■

Lemma 15 is at the core of our subsequent arguments. It shows that the function $\beta(\mathcal{G}, \alpha)$ has a simple and explicit formula since its inverse-consistent complement can be easily obtained. Furthermore, it will be used to derive *approximate* inverse-consistent complement of the matrices with sparse, but not necessarily acyclic support graphs.

Lemma 16 *Condition (1-ii) of Theorem 12 is implied by its Condition (1-i) if the graph* $\text{supp}(\Sigma^{\text{res}})$ *is acyclic.*

Proof: Consider an arbitrary matrix $M \in \mathbb{S}^d$ with a positive-definite completion. It suffices to show that if $\text{supp}(M)$ is acyclic, then M is sign-consistent. To this end, consider the matrix Π^{opt} introduced in the proof of Lemma 6, which is indeed the unique inverse-consistent complement of M. For an arbitrary pair $(i,j) \in \text{supp}(M)$, define a diagonal matrix $\Phi \in \mathbb{S}^n$ as follows:

- Consider the graph $\text{supp}(M)\backslash\{(i,j)\}$, which is obtained from the acyclic graph $\text{supp}(M)$ by removing its edge (i,j). The resulting graph is disconnected because there is no path between nodes i and j.
- Divide the disconnected graph $\text{supp}(M)\backslash\{(i,j)\}$ into two groups 1 and 2 such that group 1 contains node i and group 2 includes node 2.
- For every $l \in \{1,...,n\}$, define Φ_{ll} as 1 if l is in group 1, and as -1 otherwise.

In light of Lemma 6, $(M+\Pi)^{-1}$ is the unique solution of (4). Similarly, $\Phi(M+\Pi)^{-1}\Phi$ is a feasible point for (4). As a result, the following inequality must hold

$$\Big\{\text{trace}(M(M+\Pi^{\text{opt}})^{-1}) - \text{logdet}((M+\Pi^{\text{opt}})^{-1})\Big\}$$
$$-\Big\{\text{trace}(M\Phi(M+\Pi^{\text{opt}})^{-1}\Phi) - \text{logdet}(\Phi(M+\Pi^{\text{opt}})^{-1}\Phi)\Big\} < 0.$$

It is easy to verify that the left side of the above inequality is equal to twice the product of the (i,j) entries of M and $(M+\Pi)^{-1}$. This implies that the (i,j) entries of M and $(M+\Pi)^{-1}$ have opposite signs. As a result, M is sign-consistent. ■

Definition 17 *Define* $T(\lambda)$ *as a* $d \times d$ *symmetric matrix whose* $(i,j)^{th}$ *entry is equal to* $\Sigma_{ij} + \lambda \times sign(S_{ij}^{opt})$ *for every* $(i,j) \in supp(S^{opt})$*, and it is equal to zero otherwise.*

The next result of this paper is a consequence of Lemmas 15 and 16 and Theorem 12.

Theorem 18 *Assume that the graph* $supp(S^{opt})$ *is acyclic and the matrix* $D + T(\lambda)$ *is positive-definite. Then, the relation* $\mathcal{E}^{opt} \subseteq \mathcal{E}^{\text{res}}$ *holds and the optimal solution* S^{opt} *of the GL can be computed via the explicit formula*

$$S_{ij}^{opt} = \begin{cases} \frac{1}{\Sigma_{ii}}\left(1 + \sum_{(i,m)\in\mathcal{E}^{opt}} \frac{(\Sigma_{im}^{\text{res}})^2}{\Sigma_{ii}\Sigma_{mm}-(\Sigma_{im}^{\text{res}})^2}\right) & \text{if} \quad i=j, \\ \frac{-\Sigma_{ij}^{\text{res}}}{\Sigma_{ii}\Sigma_{jj}-(\Sigma_{ij}^{\text{res}})^2} & \text{if} \quad (i,j)\in\mathcal{E}^{opt}, \\ 0 & \text{otherwise}, \end{cases} \tag{11}$$

where $\mathcal{E}^{opt}$ *and* $\mathcal{E}^{\text{res}}$ *denote the edge sets of* $\text{supp}(S^{opt})$ *and* $\text{supp}(\Sigma^{\text{res}})$*, respectively.*

When the regularization parameter λ is large, the graph $\text{supp}(S^{\text{opt}})$ is expected to be sparse and possibly acyclic. In this case, the matrix $T(\lambda)$ is sparse with small nonzero entries. If $D + T(\lambda)$ is positive-definite and $\text{supp}(S^{\text{opt}})$ is acyclic, Theorem 18 reveals two important properties of the solution of the GL: 1) its support graph is contained in the sparsity graph of the thresholded sample covariance matrix, and 2) the entries of this matrix can be found using the explicit formula (11). However, this formula requires to know the locations of the nonzero elements of S^{opt}. In what follows, we will replace the assumptions of the above theorem with easily verifiable rules that are independent from the optimal solution S^{opt} or the locations of its nonzero entries. Furthermore, it will be shown that these conditions are expected to hold when λ is large enough, i.e., if a sparse matrix S^{opt} is sought.

Theorem 19 *Assume that the following conditions are satisfied:*

- **Condition 2-i.** *The graph* $supp(\Sigma^{\text{res}})$ *is acyclic.*
- **Condition 2-ii.** $I_d + \tilde{\Sigma}^{\text{res}}$ *is positive-definite.*
- **Condition 2-iii.** $\|\tilde{\Sigma}^{\text{res}}\|_{\max}^2 \leq \min\limits_{\substack{i\neq j \\ |\Sigma_{ij}|\leq\lambda}} \frac{\lambda - |\Sigma_{ij}|}{\sqrt{\Sigma_{ii}\Sigma_{jj}}}$.

Then, the sparsity pattern of the optimal solution S^{opt} *corresponds to the sparsity pattern of* Σ^{res} *and, in addition,* S^{opt} *can be obtained via the explicit formula* (11)*.*

The above theorem states that if a sparse graph is sought, then as long as some easy-to-verify conditions are met, there is an explicit formula for the optimal solution. It will later be shown that Condition (2-i) is exactly or approximately satisfied if the regularization coefficient is sufficiently large. Condition (2-ii) implies that the eigenvalues of the normalized residue of Σ with respect to λ should be greater than -1. This condition is expected to be automatically satisfied since most of the elements of $\tilde{\Sigma}^{\text{res}}$ are equal to zero and the nonzero elements have small magnitude. In particular, this condition is satisfied if $I_d + \tilde{\Sigma}^{\text{res}}$ is diagonally dominant. Finally, using (8), it can be verified that Condition (2-iii) is satisfied if

$$\frac{\left(\frac{2\sigma_1-\sigma_k-\sigma_{k+1}}{\Sigma_{\max}}\right)^2}{\frac{\sigma_k-\sigma_{k+1}}{\Sigma_{\max}}} \leq \frac{2}{r^2}. \tag{12}$$

Similar to the arguments made in the previous subsection, (12) shows that Condition (2-iii) is satisfied if $\frac{2\sigma_1-\sigma_k-\sigma_{k+1}}{\Sigma_{\max}}$ is small. This is expected to hold in practice since the choice of λ

entails that $2\sigma_1 - \sigma_k - \sigma_{k+1}$ is much smaller than $\Sigma_{\max}$. Under such circumstances, one can use Theorem 19 to obtain the solution of the GL without having to solve (2) numerically.

Having computed the sample covariance matrix, we will next show that checking the conditions in Theorem 19 and finding S^{opt} using (11) can all be carried out efficiently.

Corollary 20 *Given Σ and λ, the total time complexity of checking the conditions in Theorem 19 and finding S^{opt} using* (11) *is $\mathcal{O}(d^2)$.*

Another line of work has been devoted to studying the connectivity structure of the optimal solution of the GL. In particular, Mazumder and Hastie (2012) and Witten et al. (2011) have shown that the connected components induced by thresholding the covariance matrix and those in the support graph of the optimal solution of the GL lead to the same vertex partitioning. Although this result does not require any particular condition, it cannot provide any information about the edge structure of the support graph and one needs to solve (2) for each connected component using an iterative algorithm, which may take up to $\mathcal{O}(d^3)$ per iteration (Friedman et al. 2008; Banerjee et al. 2008; Mazumder and Hastie 2012). Corollary 20 states that this complexity could be reduced significantly for sparse graphs.

Remark 21 *The results introduced in Theorem 12 can indeed be categorized as a set of "safe rules" that correctly determine sparsity pattern of the optimal solution of the GL. These rules are subsequently reduced to a set of easily verifiable conditions in Theorem 19 to safely obtain the correct sparsity pattern of the acyclic components in the optimal solution. On the other hand, there is a large body of literature on simple and cheap safe rules to prescreen and simplify the sparse learning and estimation problems, including Lasso, logistic regression, support vector machine, group Lasso, etc(Ghaoui et al. (2010); Tibshirani et al. (2012); Fercoq et al. (2015); Ndiaye et al. (2015)). Roughly speaking, these methods are based on constructing a sequence of safe regions that encompass the optimal solution for the dual of the problem at hand. These safe regions, together with the Karush—Kuhn—Tucker (KKT) conditions, give rise to a set of rules that facilitate inferring the sparsity pattern of the optimal solution. Our results are similar to these methods since we also analyze the special structure of the KKT conditions and resort to the dual of the GL to obtain the correct sparsity structure of the optimal solution. However, according to the seminal work Ndiaye et al. (2015), most of the developed results on safe screening rules rely on strong Lipschitz assumptions on the objective function; an assumption that is violated in the GL. This calls for a new machinery to derive theoretically correct rules for this problem; a goal that is at the core of Theorems 12 and 19.*

3.3. Approximate Closed-form Solution: Sparse Graphs

In the preceding subsection, it was shown that, under some mild assumptions, the GL has an explicit closed-form solution if the support graph of the thresholded sample covariance matrix is acyclic. In this part, a similar approach will be taken to find approximate solutions of the GL with an arbitrary underlying sparsity graph. In particular, by closely examining the hard-to-check conditions of Theorem 12, a set of simple and easy-to-verify surrogates will be introduced which give rise to an approximate closed-form solution for the general sparse GL. Furthermore, we will derive a strong upper bound on the approximation error

and show that it decreases exponentially fast with respect to the length of the minimum-length cycle in the support graph of the thresholded sample covariance matrix. Indeed, the formula obtained earlier for acyclic graphs could be regarded as a by-product of this generalization since the length of the minimum-length cycle can be considered as infinity for such graphs. The significance of this result is twofold:

- Recall that the support graph corresponding to the optimal solution of the GL is sparse (but not necessarily acyclic) for a large regularization coefficient. In this case, the approximate error is provably small and the derived closed-form solution can serve as a good approximation for the exact solution of the GL. This will later be demonstrated in different simulations.

- The performance and runtime of numerical (iterative) algorithms for solving the GL heavily depend on their initializations. It is known that if the initial point is chosen close enough to the optimal solution, these algorithms converge to the optimal solution in just a few iterations (Friedman et al. 2008; Hsieh et al. 2014; Zhang and Lavaei 2017). The approximate closed-form solution designed in this paper can be used as an initial point for the existing numerical algorithms to significantly improve their runtime.

The proposed approximate solution for the GL with an arbitrary support graph has the following form:

$$A_{ij} = \begin{cases} \frac{1}{\Sigma_{ii}} \left(1 + \sum\limits_{(i,m)\in\mathcal{E}^{\text{opt}}} \frac{(\Sigma^{\text{res}}_{im})^2}{\Sigma_{ii}\Sigma_{mm} - (\Sigma^{\text{res}}_{im})^2} \right) & \text{if} \quad i = j, \\ \frac{-\Sigma^{\text{res}}_{ij}}{\Sigma_{ii}\Sigma_{jj} - (\Sigma^{\text{res}}_{ij})^2} & \text{if} \quad (i,j) \in \mathcal{E}^{\text{res}}, \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

The definition of this matrix does not make any assumption on the structure of the graph $\mathcal{E}^{\text{res}}$. Recall that Σ^{res} in the above formula is the shorthand notation for $\Sigma^{\text{res}}(\lambda)$. As a result, the matrix A is a function of λ. To prove that the above matrix is an approximate solution of the GL, a few steps need to be taken. First, recall that—according to the proof of Lemma 15—it is possible to explicitly build the inverse-consistent complement of the thresholded sample covariance matrix if its sparsity graph is acyclic. This matrix serves as a *certificate* to confirm that the explicit solution (13) indeed satisfies the KKT conditions for the GL. By adopting a similar approach, it will then be proved that if the support graph of the thresholded sample covariance matrix is sparse, but not necessarily acyclic, one can find an approximate inverse-consistent complement of the proposed closed-form solution to approximately satisfy the KKT conditions.

Definition 22 *Given a number $\epsilon \geq 0$, a $d \times d$ matrix B is called an ϵ-**relaxed inverse** of matrix A if $A \times B = I_d + E$ such that $|E_{ij}| \leq \epsilon$ for every $(i,j) \in \{1,2,...,d\}^2$.*

The next lemma offers optimality (KKT) conditions for the unique solution of the GL.

Lemma 23 (Sojoudi (2016)) *A matrix S^{opt} is the optimal solution of the GL if and only if it satisfies the following conditions for every $i, j \in \{1, 2, ..., d\}$*

$$(S^{opt})^{-1}_{ij} = \Sigma_{ij} \qquad \text{if} \quad i = j, \tag{14a}$$
$$(S^{opt})^{-1}_{ij} = \Sigma_{ij} + \lambda \times \text{sign}(S^{opt}_{ij}) \qquad \text{if} \quad S^{opt}_{ij} \neq 0, \tag{14b}$$
$$\Sigma_{ij} - \lambda \leq (S^{opt})^{-1}_{ij} \leq \Sigma_{ij} + \lambda \qquad \text{if} \quad S^{opt}_{ij} = 0, \tag{14c}$$

where $(S^{opt})^{-1}_{ij}$ denotes the $(i,j)^{th}$ entry of $(S^{opt})^{-1}$.

The following definition introduces a relaxed version of the first-order optimality conditions given in (14).

Definition 24 *Given a number $\epsilon \geq 0$, it is said that the $d \times d$ matrix A satisfies the **ϵ-relaxed KKT conditions** for the GL problem if there exists a $d \times d$ matrix B such that*

- *B is an ϵ-relaxed inverse of the matrix A.*
- *The pair (A, B) satisfies the conditions*

$$B_{ij} = \Sigma_{ij} \qquad \text{if} \quad i = j, \tag{15a}$$
$$|B_{ij} - (\Sigma_{ij} + \lambda \times \text{sign}(A_{ij}))\,| \leq \epsilon \qquad \text{if} \quad A_{ij} \neq 0, \tag{15b}$$
$$|B_{ij} - \Sigma_{ij}| \leq \lambda + \epsilon \qquad \text{if} \quad A_{ij} = 0. \tag{15c}$$

By leveraging the above definition, the objective is to prove that the explicit solution introduced in (13) satisfies the ϵ-relaxed KKT conditions for some number ϵ to be defined later.

Definition 25 *Given a graph $\mathcal{G}$, define the function $c(\mathcal{G})$ as the length of the minimum-length cycle of $\mathcal{G}$ (the number $c(\mathcal{G})$ is set to $+\infty$ if $\mathcal{G}$ is acyclic). Let $\deg(\mathcal{G})$ refer to the maximum degree of $\mathcal{G}$. Furthermore, define $\mathcal{P}_{ij}(\mathcal{G})$ as the set of all simple paths between nodes i and j in $\mathcal{G}$, and denote the maximum of $|\mathcal{P}_{ij}(\mathcal{G})|_0$ over all pairs (i,j) as $P_{\max}(\mathcal{G})$.*

Define $\Sigma_{\max}$ and $\Sigma_{\min}$ as the maximum and minimum diagonal elements of Σ, respectively.

Theorem 26 *Under the assumption $\lambda < \sigma_1$, the explicit solution (13) satisfies the ϵ-relaxed KKT conditions for the GL with ϵ chosen as*

$$\epsilon = \max\left\{\Sigma_{\max}, \sqrt{\frac{\Sigma_{\max}}{\Sigma_{\min}}}\right\} \cdot \delta \cdot (P_{\max}(\text{supp}(\Sigma^{\text{res}})) - 1) \cdot \left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\left\lceil \frac{c(\text{supp}(\Sigma^{\text{res}}))}{2} \right\rceil}, \tag{16}$$

where

$$\delta = 1 + \frac{\deg(\text{supp}(\Sigma^{\text{res}})) \cdot \|\tilde{\Sigma}^{\text{res}}\|^2_{\max}}{1 - \|\tilde{\Sigma}^{\text{res}}\|^2_{\max}} + \frac{(\deg(\text{supp}(\Sigma^{res})) - 1)}{1 - \|\tilde{\Sigma}^{\text{res}}\|^2_{\max}}, \tag{17}$$

if the following conditions are satisfied:

- **Condition 3-i.** *$I_d + \tilde{\Sigma}^{\text{res}}$ is positive-definite.*

- **Condition 3-ii.** $\|\tilde{\Sigma}^{\text{res}}\|^2_{\max} \leq \min\limits_{\substack{i \neq j \\ (i,j) \notin \text{supp}(\Sigma^{\text{res}})}} \frac{\lambda - |\Sigma_{ij}|}{\sqrt{\Sigma_{ii}\Sigma_{jj}}}$.

The number ϵ given in Theorem 26 is comprised of different parts:

- $\|\tilde{\Sigma}^{\text{res}}\|_{\max}$: Notice that $\|\tilde{\Sigma}^{\text{res}}\|_{\max}$ is strictly less than 1 and λ is large when a sparse graph is sought. Therefore, $\|\tilde{\Sigma}^{\text{res}}\|_{\max}$ is expected to be small for sparse graphs. Under this assumption, we have $0 \leq \|\tilde{\Sigma}^{\text{res}}\|_{\max} \ll 1$.
- $c(\text{supp}(\Sigma^{\text{res}}))$: It is straightforward to verify that $c(\text{supp}(\Sigma^{\text{res}}))$ is a non-decreasing function of λ. This is due to the fact that as λ increases, $\Sigma^{\text{res}}(\lambda)$ becomes sparser and this results in a support graph with fewer edges. In particular, if $d \geq 3$, then $c(\text{supp}(\Sigma^{\text{res}})) = 3$ for $\lambda = 0$ and $c(\text{supp}(\Sigma^{\text{res}})) = +\infty$ for $\lambda = \sigma_1$ almost surely.
- $P_{\max}(\text{supp}(\Sigma^{\text{res}}))$ and $\deg(\text{supp}(\Sigma^{\text{res}}))$: These two parameters are also non-decreasing functions of λ and likely to be small for large λ. For a small λ, the numbers $P_{\max}(\text{supp}(\Sigma^{\text{res}}))$ and $\deg(\text{supp}(\Sigma^{\text{res}}))$ could be on the order of $\mathcal{O}(d!)$ and $\mathcal{O}(d)$, respectively. However, these values are expected to be small for sparse graphs. In particular, it is easy to verify that for nonempty and acyclic graphs, $P_{\max}(\text{supp}(\Sigma^{\text{res}})) = 1$.

The above observations imply that if λ is large enough and the support graph of Σ^{res} is sparse, (13) serves as a good approximation of the optimal solution of the GL. In other words, it results from (16) that if $\text{supp}(\Sigma^{\text{res}})$ has a structure that is close to an acyclic graph, i.e., it has only a few cycles with moderate lengths, we have $\epsilon \approx 0$. In Section 4, we will present illustrative examples to show the accuracy of the closed-form approximate solution with respect to the size of the cycles in the sparsity graph.

Consider the matrix A given in (13), and let $\mu_{\min}(A)$ and $\mu_{\max}(A)$ denote its minimum and maximum eigenvalues, respectively. If $\lambda = \sigma_1$, then $A = D^{-1}$ (recall that D collects the diagonal entries of Σ) and subsequently $\mu_{\min}(A) > 0$. Since $\mu_{\min}(\cdot)$ is a continuous function of λ, there exists a number λ_0 in the interval $(0, 1)$ such that the matrix A (implicitly defined based on λ) is positive-definite for every $\lambda \geq \lambda_0$. The following theorem further elaborates on the connection between the closed-form formula and the optimal solution of the GL.

Theorem 27 *There exists an strictly positive number λ_0 such that, for every $\lambda \geq \lambda_0$, the matrix A given in (13) is the optimal solution of the GL problem after replacing Σ with some perturbed matrix $\hat{\Sigma}$ that satisfies the inequality*

$$\|\Sigma - \hat{\Sigma}\|_2 \leq d_{\max}(A) \left(\frac{1}{\mu_{\min}(A)} + 1\right) \epsilon, \tag{18}$$

where $d_{\max}(A)$ is the maximum vertex cardinality of the connected components in the graph $\text{supp}(A)$ and ϵ is given in (16). Furthermore, 18 implies that

$$f(A) - f^* \leq \left(\mu_{\max}(A) + \mu_{\max}(S^{\text{opt}})\right) d_{\max}(A) \left(\frac{1}{\mu_{\min}(A)} + 1\right) \epsilon, \tag{19}$$

where $f(A)$ and f^* are the objective functions of the GL evaluated at A and the optimal solution, respectively.

As mentioned before, if a sparse solution is sought for the GL, the regularization coefficient would be large and this helps with the satisfaction of the inequality $\lambda \geq \lambda_0$. In fact, it will be shown through different simulations that λ_0 is small in practice and hence, this condition is not restrictive. Under this circumstance, Theorem 27 states that the easy-to-construct matrix A is 1) the exact optimal solution of the GL problem with a perturbed sample covariance matrix, and 2) it is the approximate solution of the GL with the original sample covariance matrix. The magnitudes of this perturbation and approximation error are a function of $d_{\max}(A)$, $\mu_{\min}(A)$, $\mu_{\max}(A)$, $\mu_{\max}(S^{\text{opt}})$, and ϵ. Furthermore, it should be clear that A and ϵ are functions of λ and Σ (we dropped this dependency for simplicity of notation). Recall that the disjoint components (or the vertex partitions) of $\text{supp}(A)$ satisfy a nested property: given $1 \geq \lambda_1 > \lambda_2 \geq 0$, the components of $\text{supp}(A)$ for $\lambda = \lambda_1$ are nested within the components of $\text{supp}(A)$ for $\lambda = \lambda_2$ (see Mazumder and Hastie (2012) for a simple proof of this statement). This implies that $d_{\max}(A)$ is a decreasing function of λ. In particular, it can be observed that $d_{\max}(A) = d$ if $\lambda = 0$ and $d_{\max}(A) = 1$ if $\lambda = \sigma_1$. Now, consider $\mu_{\min}(A)$, $\mu_{\max}(A)$, and $\mu_{\max}(S^{\text{opt}})$. First, note that if $\lambda = \sigma_1$, then $A = S^{\text{opt}} = D^{-1}$. Furthermore, it is easy to verify that both A and S^{opt} are continuous functions of λ. Therefore, for large values of λ, $\mu_{\min}(A)$, $\mu_{\max}(A)$, and $\mu_{\max}(S^{\text{opt}})$ are expected to be close to $1/\Sigma_{\max}$, $1/\Sigma_{\min}$, and $1/\Sigma_{\min}$, respectively. In addition, as discussed earlier, ϵ is a decreasing function of λ and vanishes when λ is large enough. Based on these observations, it can be concluded that the upper bound presented in (18) is small if λ is chosen to be large.

Notice that although the aforementioned value of ϵ in (16) and the upper bound in (18) were essential in the study of the effect of the sparsity of the support graph on the accuracy of the presented closed-form solution, they are conservative in practice. These numbers may be tightened significantly for specific sample covariance matrices. We will further discuss the approximation error of the closed-form solution in Section 4.

Warm-start algorithm As delineated before, one of the main strengths of the proposed closed-form solution is that it can be used as an initial point (warm-start) for the numerical algorithms specialized for solving the GL. To this goal, the following warm-start procedure is proposed.

Algorithm 1: Warm-start algorithm

Data: data samples ($\mathbf{x}$), and regularization coefficient (λ)
Result: Solution of the GL (S^{opt})

```
Obtain the residue matrix Σ^res based on Definition 11 and the closed-form solution
  A from (13);
for each component i in supp(Σ^res) do
    if Conditions 2-i, 2-ii, 2-iii are satisfied then
        S^opt[i] ← A[i];
    else
        Find S^opt[i] by numerically solving the GL for component i with initial
          point A[i];
    end
end
```


In the above algorithm, $S^{\text{opt}}[i]$ and $A[i]$ are the submatrices of S^{opt} and A corresponding to the i^{th} component of $\text{supp}(\Sigma_{\text{res}})$. The warm-start algorithm is based on the key fact that the GL decomposes over the disjoint components of $\text{supp}(\Sigma_{\text{res}})$ (Mazumder and Hastie 2012; Witten et al. 2011). In particular, in the first step, the warm-start algorithm obtains the residue matrix according to Definition 11. Next, for every disjoint component of the residue matrix, if its support graph is acyclic and the conditions of Theorem 19 are satisfied, then the corresponding component in S^{opt} is found using the closed-form solution (11). Otherwise, this closed-form solution is provided as an initial point to a numerical algorithm, such as GLASSO and QUIC Friedman et al. (2008); Hsieh et al. (2014), in order to boost the runtime of solving the GL for the considered component. The results of the warm-start algorithm will be evaluated in the next section.

Remark 28 *The statistical analysis of the GL entails that λ should converge to zero as the number of samples grows to infinity. It is worthwhile to mention that our results may not be applicable in the high sampling regime, where λ is close to zero and consequently the thresholded sample covariance matrix is dense. However, notice that the main strength of the GL lies in the high dimensional-low sampling regime where n is much smaller than d and is in the order of $\log d$. Under such circumstances, the proposed explicit formula results in highly accurate solutions for the GL. In fact, it will be shown through massive-scale simulations that in practice, the required conditions on λ—such as the positive-definiteness of $I_d + \tilde{\Sigma}^{\text{res}}$—for the validity of the presented results are much more relaxed than the known conditions on λ to guarantee the statistical consistency of the GL.*

4. Numerical Results

In this section, we will demonstrate the effectiveness of the proposed methods on synthetically generated data, as well as on real data collected from the brain networks and transportation systems.

4.1. Case Study on Synthetic Data

Given a nonnegative number ω, consider an arbitrary sample covariance matrix Σ with the following properties:

- Its diagonal elements are normalized to 1.
- The entries corresponding to an arbitrary spanning tree of $\text{supp}(\Sigma)$ belong to the union of the intervals $[-0.85, -0.95]$ and $[0.85, 0.95]$.
- The off-diagonal entries that do not belong to the spanning tree are in the interval $[-0.85 + \omega, 0.85 - \omega]$.

The goal is to find conditions on λ, ω and the size of the covariance matrix such that Theorem 19 can be used to obtain a closed-form solution for the GL problem. One can choose the value of λ to be greater than σ_d to ensure that the graph $\text{supp}(\Sigma^{\text{res}})$ is acyclic. In particular, if we pick λ in the interval (σ_d, σ_{d-1}), the graph $\text{supp}(\Sigma^{\text{res}})$ becomes a spanning tree.

Select λ as $0.85 - \epsilon$ for a sufficiently small number ϵ and consider Condition (2-ii) in Theorem 19. One can easily verify that $I_d + \Sigma^{\text{res}}$ is positive-definite if the inequality

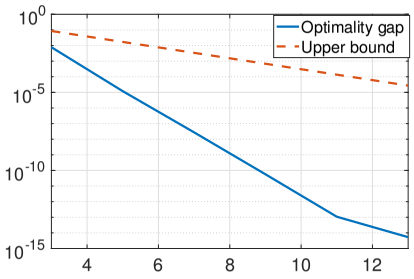


Figure 1: The optimality gap between the closed-form and optimal solutions for the GL

$\frac{1}{\deg(v)} > (\sigma_1 - \lambda)^2$ holds for every node v in $\text{supp}(\Sigma^{\text{res}})$, where $\deg(v)$ is the degree of node v. This condition is guaranteed to be satisfied for all possible acyclic graphs if $(\deg(v))(0.95 - 0.85)^2 < 1$ or equivalently $\deg(v) \leq 100$ for every node v. Regarding Condition (2-iii), it can be observed that the relation $(\sigma_1 - \lambda)^2 \leq \lambda - \sigma_{k+1}$ holds if $(0.95 - 0.85)^2 < 0.85 - (0.85 - \omega)$. This implies that the inequality $\omega > 0.01$ guarantees the satisfaction of Condition (2-iii) for every acyclic graph $\text{supp}(\Sigma^{\text{res}})$. In other words, one can find the optimal solution of the GL problem using the explicit formula in Theorem 19 as long as: 1) a spanning tree structure for the optimal solution of the GL problem is sought, 2) the degree of each node in the spanning tree is not greater than 100, and (3) the difference between σ_{d-1} and σ_d is greater than 0.01. Note that Condition (2) is conservative and can be dropped for certain types of graphs (e.g., path graphs). In practice, the positive-definiteness of $I_d + \Sigma^{\text{res}}$ is not restrictive; we have verified that this matrix is positive-definite for randomly generated instances with the sizes up to $d = 200,000$ even when $\deg(v) > 100$.

Now, consider the following modifications in the experiment:

- The elements of Σ corresponding to a cycle of length d are randomly set to -0.8 or 0.8 with equal probability.
- The off-diagonal entries that do not correspond to the above cycle are in the interval $[-0.7, 0.7]$.

If λ is chosen as 0.75, then the graph $\text{supp}(\Sigma^{\text{res}})$ coincides with a cycle of length d. Furthermore, $I_d + \Sigma^{\text{res}}$ is diagonally dominant and hence positive-definite for every d. Figure 1 shows the optimality gap of the proposed closed-form solution and its derived theoretical upper bound (i.e. the left and right hand sides of (19), respectively) with respect to the length of the cycle d in log-linear scale. (note that $\deg(\text{supp}(\Sigma^{\text{res}}))$ and $P_{\max}(\text{supp}(\Sigma^{\text{res}}))$ in (19) are both equal to 2). Two important observations can be made based on this figure.

- In practice, the performance of the derived closed-form solution is significantly better than its theoretical upper bounds. In fact, this error is less than 10^{-6} when the length of the minimum-length cycle is at least 6. The high accuracy of the closed-form solution will become more evident in the subsequent case studies on large-scale problems.

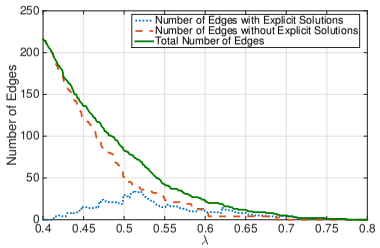
- It can be seen that the logarithm of the optimality gap is approximately a linear function of the cycle length. This matches the behavior of the theoretical bounds introduced in Theorems 26 and 27: the approximation error is exponentially decreasing with respect to the length of the minimum-length cycle.

4.2. Case Study on Brain Networks

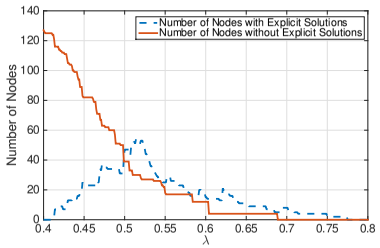
Consider the problem of estimating the brain functional connectivity network based on a set of resting state functional MRI (fMRI) data collected from 20 individual subjects (Vértes et al., 2012). The data for each subject correspond to disjoint brain activities and are correlated due to the underlying functional connectivity structure of the brain. In order to represent these dependencies, each disjoint region of the brain can be considered as a node and the correlation between two different regions can be resembled by an edge between the nodes. The data set for each subject consists of 134 samples of low frequency oscillations taken from 140 different cortical brain regions. We construct a normalized sample covariance matrix by combining the data sets of all 20 subjects (note that the data for each individual is limited and not informative enough, but the combined data provides rich information about the brain network). The goal is to use the GL to estimate the underlying functional connectivity network of different regions of the brain based on the obtained 140×140 sample covariance matrix. We study the thresholded sample covariance matrix and the derived closed-form solution for different values of the regularization coefficient in order to analyze their accuracy.

Figure 2a shows the number of edges in the sparsity graph of the thresholded sample covariance matrix that belong to those connected components satisfying the conditions in Theorem 19. The formula derived in this paper is able to find the optimal values of the entries of the solution corresponding to these edges. It can be observed that if λ is greater than 0.51, then almost half of the edges in the sparsity graph of the optimal solution can be found using the proposed explicit formula. This is due to the fact that the corresponding entries in the residue matrix belong to the acyclic components of its sparsity graph and satisfy the conditions of Theorem 19. Figure 2b depicts the number of nodes that belong to the components (with sizes greater than 1) for which the corresponding submatrices of the solution of the GL have an explicit formula. Note that those entries in the optimal solution that correspond to isolated nodes are trivially equal to 0. Therefore, in order to better reflect the significance of the derived solution, we have only considered the components with at least two nodes. It can be observed that if λ is greater than 0.5, then the number of nodes belonging to the components with explicit formula is greater than the number of those nodes associated with inexact closed-form solutions. Figure 2c demonstrates the number of edges in the sparsity graph of the optimal solution, together with the number of mismatches in the edge sets of the sparsity graphs of the optimal and thresholded solutions. Notice that the number of mismatches is less than 10% when λ is greater than 0.35 and is almost 0 when λ is greater than 0.5.

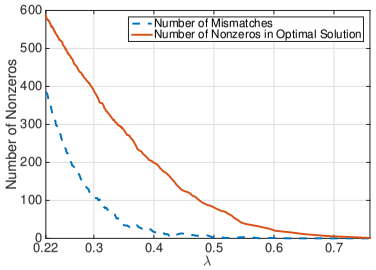
Figure 2d shows the minimum eigenvalues of the optimal and closed-form approximate solutions for different values of λ. The approximate solution is positive-definite when λ is greater than 0.37. This implies that λ_0 in Corollary 27 is equal to 0.37. Figures 2e and 2f depict the 2-norm of the approximation error (the difference between the optimal and closed-



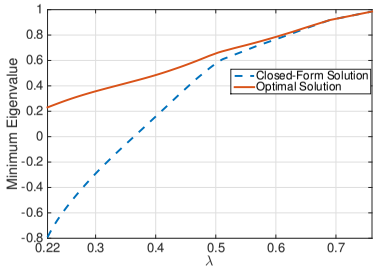
(a) Number of edges



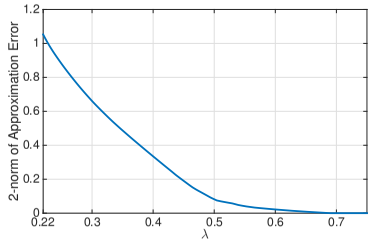
(b) Number of nodes



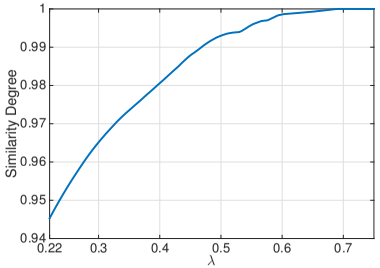
(c) Number of nonzeros and mismatches



(d) Minimum eigenvalue



(e) 2-norm of the approximation error



(f) Similarity degree

Figure 2: a) Number of edges in the sparsity graph of the closed-form approximate solution whose corresponding entries are guaranteed to be equal to those in the sparsity graph of the optimal solution due to Theorem 19. b) Number of nodes that belong to the components for which the corresponding submatrices of the optimal solution have explicit formulas. c) Number of edges in the sparsity graph of the optimal solution, compared to the number of mismatches. d) Minimum eigenvalues of the optimal and closed-form approximate solutions. e) The 2-norm of the difference between the optimal and approximate solutions. f) The similarity degree between the optimal and approximate solutions.

form approximate solutions) and the similarity degree between these two solutions, which is defined as

$$\text{similarity degree} = \frac{\text{trace}(\tilde{S}^{\text{opt}} \times \tilde{A})}{\|\tilde{S}^{\text{opt}}\|_F \times \|\tilde{A}\|_F},$$

where $\tilde{S}^{\text{opt}} = S^{\text{opt}} - I_d$ and $\tilde{A} = A - I_d$. Subtracting the identity matrix from A and S^{opt} is due to the observation that both matrices have diagonal entries close to 1 when the support graph is sparse. This leads to an artificially inflated similarity degree between A and S^{opt}. Therefore, in order to have a better assessment of the similarity between the closed-form and optimal solutions, we measure the similarity between A and S^{opt} after softening the effect of their diagonal entries. The similarity degree of 1 means that the optimal and approximate solutions are exactly equal.

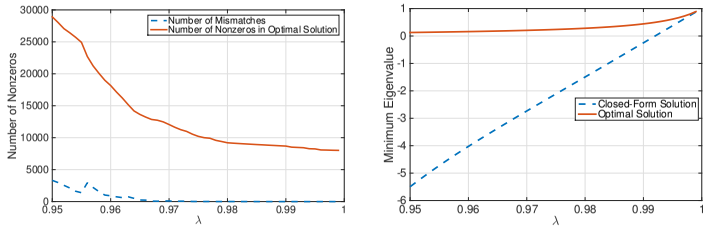
It can be observed that the approximation error is small and the similarity degree is high for a wide range of values of λ. For instance, if λ is greater than 0.4, then the 2-norm of the approximation error is less than 0.37 and the similarity degree is greater than 0.98. For these values of λ, the number of edges in the sparsity graph of the optimal solution ranges from 200 to 0. In all of these cases, the structure and values of the optimal solution can be estimated efficiently, without solving the optimization problem numerically.

4.3. Case Study on Transportation Networks

In recent years, the problem of short- and long-term traffic flow prediction and control has attracted much attention in Intelligent Transportation Systems (ITSs) (Figueiredo et al. 2001). Estimating the correlation between the traffic flows on different links of a transportation network is one of the crucial steps toward the traffic congestion control in the network; it can also serve as an initial block in different traffic forecasting methods. Substantial research has been devoted to extracting these dependencies and performing predictions based on the measured data (see Yin et al. (2017); Nassiri and Aghamohammadi (2017); Sun et al. (2012) and the references therein). In this case study, the objective is to construct a sparse matrix representing the conditional covariance between the traffic flows of different links in the network. The data is collected from the Caltrans Performance Measurement System (PeMS) database, which consists of traffic information of freeways on the a statewide scale across California (PeM 2017). We consider the data measured by the stations deployed in District 3 of California, which is collected and aggregated every 5 minutes from 1277 stations during March 6^{th} to March 12^{th} of the year 2017 (one-week interval). Due to the malfunctioning of some of the detectors, a non-negligible portion of the traffic flows was missing from the raw data set. Therefore, the following steps were taken before solving the GL problem in order to obtain a useful representation of the raw data:

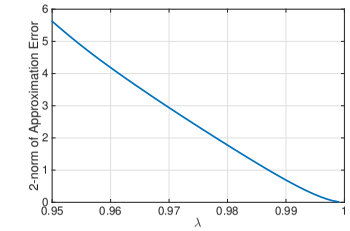
- Since 228 stations did not have sufficient number of measurements during the one-week period, they were removed from the sampled data.
- In a few stations, the detectors did not measure the traffic flow for some periods of time. For these data samples, we used a linear interpolation method to estimate the missing values.

After performing the aforementioned data-cleaning steps, a 1049×1049 normalized sample covariance matrix was constructed from the combined 2016 data samples (288 samples for

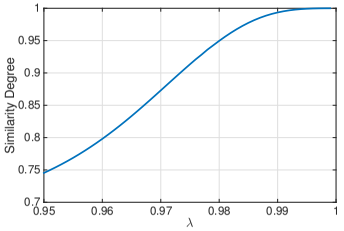


(a) Number of nonzeros and mismatches

(b) Minimum eigenvalue



(c) 2-norm of the approximation error



(d) Similarity degree

Figure 3: a) Number of edges in the sparsity graph of the optimal solution, compared to the number of mismatches. b) Minimum eigenvalues of the optimal and closed-form approximate solutions. c) The 2-norm of the difference between the optimal and approximate solutions. d) The similarity degree between the optimal and approximate solutions.

each day of the week). In Figure 3, the accuracy of the thresholding technique and its corresponding closed-form approximate solution is compared to the optimal solution of the GL problem for different values of the regularization coefficient.

Since the number of entries in the upper triangular part of the sample covariance matrix is large (roughly 550,000 entries), we have only considered large values of λ in order to obtain a sparse solution for the GL. Figure 3a shows the number of edges in the sparsity graph of the optimal solution, compared to the number of mismatches between the edge sets of the sparsity graphs of the optimal and closed-form solutions. It can be observed that as λ increases, the support graph of the optimal solution becomes sparser and the number of mismatches decreases. In particular, the number of mismatches is almost zero if λ is chosen to be greater than 0.97. Figure 3b depicts the minimum eigenvalues of the optimal and closed-form approximate solutions of the GL with respect to λ. The approximate solution becomes positive-definite if λ is greater than 0.991. Furthermore, Figures 3c and 3d show that, for those values of λ between 0.991 and 0.999, the 2-norm of the approximation error is between 0.5 and 0.01, and that the similarity degree is greater than 0.99. For this range of λ, the number of edges in the sparsity graph of the optimal solution is 7.82 to 7.40 times higher the number of nodes.

4.4. Case Study on Large-Scale Data

In this case study, we evaluate the performance of the proposed closed-form solution on massive randomly generated data sets. Given d (the dimension of each sample) and similar to Hsieh et al. (2014) and Yun and Toh (2011), a sparse inverse covariance matrix is generated for each test case according to the following procedure: first, a sparse matrix $U \in \mathbb{R}^{d\times d}$ is generated whose nonzero elements are randomly set to $+1$ or -1, with equal probability. Then, the inverse covariance matrix is set to $UU^\top + 2I$. Depending on the test case, the number of nonzero elements in U is controlled so that the resulted inverse covariance matrix has approximately $5d$ or $10d$ nonzero elements. $n = d/2$ number of i.i.d. samples are drawn from the corresponding multivariate Gaussian distribution in all experiments, except for the largest test case with $d = 80000$. This instance has more than 3.2 billion variables and only $n = 20000$ samples are collected to solve the GL due to the memory limitations. Furthermore, the regularization coefficient is chosen such that the estimated solution has approximately the same number of nonzero elements as the ground truth.

Table 1 reports the runtime of the closed-form solution, compared to two state-of-the-art methods for solving the GL, namely QUIC (Hsieh et al. 2014) and GLASSO (Friedman et al. 2008) algorithms, as well as elementary estimator (Yang et al. 2014). The GLASSO is the most widely-used algorithm for the GL, while the QUIC algorithm is commonly regarded as the fastest available solver for this problem. The elementary estimator is recently proposed in lieu of the GL to remove its computational burden, while preserving its desired high-dimensional properties. We use the source codes for latest versions of QUIC and GLASSO in our simulations. In particular, we use the QUIC 1.1 (available in `http://bigdata.ices.utexas.edu/software/1035/`) which is implemented in $C++$ with MATLAB interface. The GLASSO is downloaded from `http://statweb.stanford.edu/~tibs/glasso/` and is implemented in FORTRAN with MATLAB interface. We implemented the elementary estimator and the proposed closed-form solution in MATLAB using its sparse package. A time limit of 4 hours is considered in all experiments. Table 1 has the following columns:

- d: The dimension of the samples.
- m: The number of nonzero elements in the true inverse covariance matrix.
- Closed-form: The runtime of the proposed method.
- QUIC-C and GLASSO-C: The runtime of the QUIC and GLASSO without initialization.
- QUIC-W and GLASSO-W: The runtime of the QUIC and GLASSO using the warm-start Algorithm 1.
- Elem.: The runtime of the elementary estimator.

In all of the test cases, the resulted closed-form solution is positive-definite and hence, feasible. It can be seen that the proposed method significantly outperforms QUIC, GLASSO and elementary estimator in terms of its runtime. In particular, the presented method is on average 6, 36, and 951 times faster than elementary, QUIC, and GLASSO methods, respectively, provided that they can obtain the solution within the predefined time limit.

d	m	Closed-Form	QUIC-C	QUIC-W	GLASSO-C	GLASSO-W	Elem.
2000	9894	0.1	2.0	1.4	42.8	13.5	0.2
2000	20022	0.1	3.0	2.1	43.8	15.3	0.2
4000	20094	0.5	13.9	7.5	460.8	135.1	2.1
4000	40382	0.5	21.5	12.0	467.6	156.2	2.9
8000	40218	2.5	78.7	49.3	3675.1	1011.2	11.3
8000	79890	2.5	111.7	88.4	3784.3	1278.8	22.2
12000	60192	7.8	243.8	153.1	⋆	3233.0	31.8
12000	119676	7.4	333.6	251.0	⋆	3437.2	70.2
16000	80064	17.1	570.0	322.8	⋆	6545.0	67.2
16000	160094	18.5	787.4	616.4	⋆	9960.8	174.8
20000	99954	39.4	1266.5	539.4	⋆	⋆	107.8
20000	200018	37.4	1683.8	1392.5	⋆	⋆	211.5
40000	200290	495.4	⋆	⋆	⋆	⋆	⋆
80000	401798	1450.4	⋆	⋆	⋆	⋆	⋆

Table 1: The runtime of different methods.

Furthermore, for the cases where the GL can be solved to optimality using QUIC, the relative optimality gap of the closed-form solution, i.e., $(f(A) - f^*)/f^*$, is 2.1×10^{-3} on average. For the cases with $d = 40000$ and $d = 80000$, none of these methods converge to a meaningful solution, while the proposed method can obtain an accurate solution in less than 30 minutes. On the other hand, the warm-start Algorithm 1 accompanied by QUIC and GLASSO yields up to 2.35 and 4.45 times speedups in their runtime, respectively. Moreover, the warm-start algorithm doubles the size of the instances that are solvable using the GLASSO.

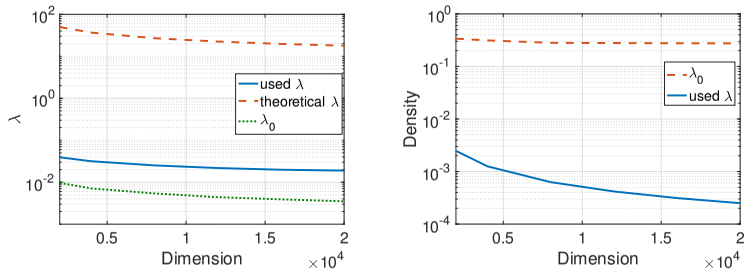
Table 2 compares the accuracy of the estimated inverse covariance matrix using different methods. This table includes the following columns:

- ℓ_F: The Frobenius norm of the difference between the true and estimated inverse covariance matrices, normalized by the Frobenius norm of the true inverse covariance matrix.
- TPR and FPR: The true positive rate (TPR) and false positive rate (FPR) defined as

$$\text{TPR} = \frac{\left|(i,j) : i \neq j, S_{ij} \neq 0, (\Sigma_*^{-1})_{ij} \neq 0\right|_0}{\left|(i,j) : i \neq j, (\Sigma_*^{-1})_{ij} \neq 0\right|_0},$$
$$\text{FPR} = \frac{\left|(i,j) : i \neq j, S_{ij} \neq 0, (\Sigma_*^{-1})_{ij} = 0\right|_0}{\left|(i,j) : i \neq j, (\Sigma_*^{-1})_{ij} = 0\right|_0},$$

  where S corresponds to the explicit formula, the optimal solution of the GL, or the elementary estimator.

It can be seen that, while the elementary estimator has slightly better estimation error, its TPR is significantly outperformed by the those of the GL and closed-form solutions. Furthermore, it can be seen that the closed-form estimator has almost the same accuracy as the optimal solution of the GL. The superiority of the proposed closed-form solution



(a) Different values of λ

(b) Density of the thresholded sample covariance matrix

Figure 4: a) Values of λ_0, used λ, and theoretical λ, b) The density of the true inverse and thresholded sample covariance matrices.

over the other methods becomes more evident in the larger instances, where it (almost) exactly recovers the true sparsity pattern of the inverse covariance matrix and results in small estimation error, while becoming the only viable method for estimating the inverse covariance matrix.

Finally, we show that the requirement $\lambda \geq \lambda_0$ in Theorem 27 does not impose any restriction on the practicality of this theorem under the finite-sampling regime. In particular, we show that in practice, the lower bound λ_0 on λ is significantly smaller than the theoretical value of λ that is derived for the high-dimensional consistency of the GL. To this goal, we compare λ_0 with the theoretical value of λ introduced in the seminal paper Ravikumar et al. (2011). In particular, Ravikumar et al. (2011) shows that the following value for λ is sufficient to guarantee consistency

$$\lambda = \frac{8}{\alpha}\sqrt{128(1+4\sigma^2)^2 \max(\Sigma_*)_{ii}^2}\sqrt{\frac{\log d + \log 4}{n}}, \tag{20}$$

where α is the mutual incoherence parameter, σ is the sub-Gaussian parameter of normalized random variables, and $(\Sigma_*)_{ii}$ is the i^{th} diagonal element of the true covariance matrix. Figure 4a shows the values for λ_0, theoretical λ defined as (20), and λ used in our simulations with respect to the dimension of the problem. On average, λ_0 is 640 and 6 times smaller than the theoretical and used λ, respectively. Furthermore, Figure 4b shows the density (the number of nonzero elements, normalized by the total number of entries) of the thresholded sample covariance matrix when λ is set to λ_0, compared to the density of the true inverse covariance matrix. Note that when $\lambda = \lambda_0$, the density of the thresholded sample covariance matrix is close to 0.3 on average while the average density of the true inverse covariance matrix is less than 0.0009. Based on these simulations, one can infer that λ_0 is an underestimator for the values of the regularization coefficient that correctly promote sparsity in the estimated solution, and the requirement $\lambda \geq \lambda_0$ is extremely mild for large-scale instances of the GL.

		Closed-Form			Graphical Lasso			Elementary		
d	m	ℓ_F	TPR	FPR	ℓ_F	TPR	FPR	ℓ_F	TPR	FPR
2000	9894	0.41	0.71	0.00	0.41	0.71	0.00	0.40	0.63	0.00
2000	20022	0.50	0.59	0.00	0.65	0.59	0.00	0.49	0.34	0.01
4000	20094	0.39	0.83	0.00	0.38	0.84	0.00	0.37	0.76	0.00
4000	40382	0.48	0.74	0.00	0.48	0.75	0.00	0.48	0.54	0.00
8000	40218	0.36	0.92	0.00	0.35	0.93	0.00	0.33	0.87	0.00
8000	79890	0.45	0.87	0.00	0.44	0.88	0.00	0.44	0.71	0.00
12000	60192	0.33	0.96	0.00	0.32	0.97	0.00	0.30	0.93	0.00
12000	119676	0.43	0.93	0.00	0.41	0.94	0.00	0.42	0.81	0.00
16000	80064	0.32	0.97	0.00	0.30	0.98	0.00	0.28	0.96	0.00
16000	160094	0.42	0.95	0.00	0.40	0.96	0.00	0.40	0.86	0.00
20000	99954	0.31	0.99	0.00	0.30	0.99	0.00	0.28	0.96	0.00
20000	200018	0.41	0.96	0.00	0.39	0.97	0.00	0.39	0.89	0.00
40000	200290	0.28	1.00	0.00	$\star$	$\star$	$\star$	$\star$	$\star$	$\star$
80000	401798	0.27	1.00	0.00	$\star$	$\star$	$\star$	$\star$	$\star$	$\star$

Table 2: The accuracy of different methods.

5. Conclusions

Graphical Lasso (GL) is a popular method for finding the conditional independence between the entries of a random vector. This technique aims at learning the sparsity pattern of the inverse covariance matrix from a limited number of samples, based on the regularization of a positive-definite matrix. Motivated by the computational complexity of solving the GL for large-scale problems, this paper provides conditions under which the GL behaves the same as the simple method of thresholding the sample covariance matrix. The conditions make direct use of the sample covariance matrix and are not based on the solution of the GL. More precisely, it is shown that the GL and thresholding techniques are equivalent if: (i) a certain matrix formed based on the sample covariance matrix is both sign-consistent and inverse-consistent, and (ii) the gap between the largest thresholded and the smallest un-thresholded entries of the sample covariance matrix is not too small. Although the GL is believed to be a difficult conic optimization problem, it is proved that it indeed has a closed-form solution in the case where the sparsity pattern of the solution is known to be acyclic. This result is then extended to general sparse graphs and an explicit formula is derived as an approximate solution of the GL, where the approximation error is also quantified in terms of the structure of the sparsity graph. The significant speedup and graceful scalability of the proposed explicit formula compared to other state-of-the-art methods is showcased on different real-world and randomly generated data sets.

Acknowledgments

This work was supported by grants from ONR, AFOSR and NSF.

Appendix

In what follows, the omitted technical proofs will be presented. A number of lemmas are required for this purpose.

Before presenting the proof of Theorem 12, consider the *normalized GL*, defined as

$$\min_{S\in\mathbb{S}^d_+} -\log\det(S)+\text{trace}(\tilde{\Sigma}S)+\sum_{i\neq j}\tilde{\lambda}_{ij}|S_{ij}|, \tag{21}$$

where $\tilde{\Sigma}$ is the normalized sample covariance, i.e., $\tilde{\Sigma}_{ij}=\frac{\Sigma_{ij}}{\sqrt{\Sigma_{ii}\Sigma_{jj}}}$ for every $(i,j)\in\{1,2,...,d\}^2$ (also known as sample correlation matrix). Similarly, $\tilde{\lambda}_{ij}$ is defined as $\frac{\lambda}{\sqrt{\Sigma_{ii}\Sigma_{jj}}}$. Upon denoting the optimal solution of the normalized GL as $\tilde{S}$, we consider the relationship between $\tilde{S}$ and S^{opt}. Recall that D is defined as a matrix collecting the diagonal elements of Σ.

Lemma 29 *We have* $S^{\text{opt}}=D^{-1/2}\tilde{S}D^{-1/2}$.

Proof Notice that the GL (2) can be re-written as follows

$$\min_{S\in\mathbb{S}^d_+} -\log\det(S)+\text{trace}(\tilde{\Sigma}D^{1/2}SD^{1/2})+\sum_{i\neq j}\lambda|S_{ij}|, \tag{22}$$

where we have used the equality

$$\text{trace}(\Sigma S)=\text{trace}(D^{1/2}\tilde{\Sigma}D^{1/2}S)=\text{trace}(\tilde{\Sigma}D^{1/2}SD^{1/2}).$$

Upon defining

$$\tilde{S}=D^{1/2}SD^{1/2} \tag{23}$$

and following some algebra, one can verify that (22) is equivalent to

$$\min_{\tilde{S}\in\mathbb{S}^d_+} -\log\det(\tilde{S})+\text{trace}(\tilde{\Sigma}\tilde{S})+\sum_{i\neq j}\tilde{\lambda}_{ij}|\tilde{S}_{ij}|+\log\det(D). \tag{24}$$

Dropping the constant term in (24) gives rise to the normalized GL (21). Therefore, $S^{\text{opt}}=D^{-1/2}\tilde{S}D^{-1/2}$ holds in light of 23. This completes the proof. ■

Proof of Theorem 12 Note that, due to the Definition 11 and Lemma 29, $\tilde{\Sigma}^{\text{res}}$ and $\tilde{S}$ have the same sparsity pattern as Σ^{res} and S^{opt}, respectively. Therefore, it suffices to show that the sparsity structures of $\tilde{\Sigma}^{\text{res}}$ and $\tilde{S}$ are the same.

To verify this, we focus on the optimality conditions for optimization (21). Define M as $I_d+\tilde{\Sigma}^{\text{res}}$. Due to Condition (1-i) and Lemma 6, M is inverse-consistent and has a unique inverse-consistent complement, which is denoted by N. First, will show that $(M+N)^{-1}$ is the optimal solution of (21). For an arbitrary pair $(i,j)\in\{1,...,d\}^2$, the KKT conditions, introduced in Lemma 23, imply that one of the following cases holds:

1) $i=j$: We have $(M+N)_{ij}=M_{ii}=\tilde{\Sigma}_{ii}$.

2) $(i,j) \in \text{supp}(\tilde{\Sigma}^{\text{res}})$: In this case, we have

$$(M+N)_{ij} = M_{ij} = \tilde{\Sigma}_{ij} - \tilde{\lambda}_{ij} \times \text{sign}(\tilde{\Sigma}_{ij}).$$

Note that since $|\Sigma_{ij}| > \lambda$, we have that $\text{sign}(M_{ij}) = \text{sign}(\tilde{\Sigma}_{ij})$. On the other hand, due to the sign-consistency of M, we have $\text{sign}(M_{ij}) = -\text{sign}\left(\left((M+N)^{-1}\right)_{ij}\right)$. This implies that

$$(M+N)_{ij} = M_{ij} = \tilde{\Sigma}_{ij} + \tilde{\lambda}_{ij} \times \text{sign}((M+N)^{-1}).$$

3) $(i,j) \notin \text{supp}(\tilde{\Sigma}^{\text{res}})$: One can verify that $(M+N)_{ij} = N_{ij}$. Therefore, due to Condition (1-iii), we have

$$\begin{aligned} |(M+N)_{ij}| &\leq \beta\left(\text{supp}(\tilde{\Sigma}^{\text{res}}), \|\tilde{\Sigma}^{\text{res}}\|_{\max}\right) \\ &\leq \min_{\substack{k\neq l \\ (k,l)\notin \text{supp}(\Sigma^{\text{res}})}} \frac{\lambda - |\Sigma_{kl}|}{\sqrt{\Sigma_{kk}\Sigma_{ll}}} \\ &= \min_{\substack{k\neq l \\ (k,l)\notin \text{supp}(\Sigma^{\text{res}})}} \tilde{\lambda}_{kl} - |\tilde{\Sigma}_{kl}|. \end{aligned} \tag{25}$$

This leads to

$$|(M+N)_{ij} - \tilde{\Sigma}_{ij}| \leq |(M+N)_{ij}| + |\tilde{\Sigma}_{ij}| \leq \min_{\substack{k\neq l \\ (k,l)\notin \text{supp}(\Sigma^{\text{res}})}} \left(\tilde{\lambda}_{kl} - |\tilde{\Sigma}_{kl}|\right) + |\tilde{\Sigma}_{ij}| \leq \tilde{\lambda}_{ij}. \tag{26}$$

Therefore, it can be concluded that $(M+N)^{-1}$ satisfies the KKT conditions for (21)[1]. On the other hand, note that $\text{supp}((\text{M}+\text{N})^{-1}) = \text{supp}(\tilde{\Sigma}^{\text{res}})$. This concludes the proof. ■

To proceed with the proof of Lemma 13, we need the following lemma.

Lemma 30 *Consider a matrix $M \in \mathbb{S}^d$ with positive-definite completion. Assume that $\|M^{(c)}\|_1 \leq \eta\|M - I_d\|_1$ and $\|M - I_d\|_1 < \frac{1}{\eta+1}$, for some number η. The relation*

$$\|M^{(c)}\|_1 \leq (1+\eta)^2 \frac{\|M - I_d\|_1^2}{1 - (\eta+1)\|M - I_d\|_1}$$

holds.

Proof Note that $M \in \mathbb{S}^d$ has a positive-definite completion and hence, is inverse-consistent due to Lemma 6. One can write

$$\|(M - I_d) + M^{(c)}\|_1 \leq \|M - I_d\|_1 + \|M^{(c)}\|_1 \leq (\eta+1)\|M - I_d\|_1 < 1.$$

Therefore,

$$\begin{aligned} (M + M^{(c)})^{-1} = (I_d + (M - I_d + M^{(c)}))^{-1} + I_d - (M - I_d + M^{(c)}) \\ + (M - I_d + M^{(c)})^2 \times \sum_{i=0}^{\infty} (-M + I_d - M^{(c)})^i. \end{aligned}$$

1. The KKT conditions for the normalized GL are equivalent to (14) after replacing λ with $\tilde{\lambda}_{ij}$

Since $\text{supp}((M + M^{(c)})^{-1}) \subseteq \text{supp}(M)$, it can be concluded that the (i,j) entries of $M^{(c)}$ and

$$(M - I_d + M^{(c)})^2 \times \sum_{i=0}^{\infty} (-M + I_d - M^{(c)})^i$$

are equal for every $(i,j) \in \text{supp}(M^{(c)})$. Since the (i,j) entry of $M^{(c)}$ is zero if $(i,j) \notin \text{supp}(M^{(c)})$, we have

$$\|M^{(c)}\|_1 \leq \left\| (M - I_d + M^{(c)})^2 \sum_{i=0}^{\infty} (M - I_d + M^{(c)})^i \right\|_1.$$

Since 1-norm is sub-multiplicative, the above inequality can be simplified as

$$\begin{aligned} \|M^{(c)}\|_1 &\leq (\|M - I_d\|_1 + \|M^{(c)}\|_1)^2 \times \sum_{i=0}^{\infty} (\|M - I_d\|_1 + \|M^{(c)}\|_1)^i \\ &= \frac{(\|M - I_d\|_1 + \|M^{(c)}\|_1)^2}{1 - \|M - I_d\|_1 - \|M^{(c)}\|_1} \\ &\leq \frac{(\|M - I_d\|_1 + \eta\|M - I_d\|\|_1)^2}{1 - \|M - I_d\|_1 - \eta\|M - I_d\|\|_1} \\ &= (1+\eta)^2 \frac{\|M - I_d\|_1^2}{1 - (\eta+1)\|M - I_d\|_1}. \end{aligned}$$

This completes the proof. ■

Proof of Lemma 13 Given an arbitrary graph $\mathcal{G}$, consider a matrix variable M with 1's on the diagonal such that $\text{supp}(M) \subseteq \mathcal{G}$. The first objective is to find a matrix in terms of M, denoted by the matrix function $N(M)$, satisfying the following properties

$$\begin{aligned} &\text{supp}\left((M + N(M))^{-1}\right) \subseteq \mathcal{G}, \\ &\text{supp}(N(M)) \subseteq \mathcal{G}^{(c)}. \end{aligned}$$

To this end, define the matrix function $A(M)$ as

$$A(M) = (M + N(M))^{-1}.$$

Observe that

- As long as $A(M)$ exists and $\text{supp}(A(M)) \subseteq \mathcal{G}$, there is a continuously differentiable mapping from $A(M)$ to M because M can be found by setting those entries of $A(M)^{-1}$ corresponding to the edges of $\mathcal{G}^{(c)}$ to zero. Moreover, the Jacobian of this mapping has full rank at $M = I_d$. Due to the inverse function theorem, the mapping from M to $A(M)$ exists and is continuously differentiable.
- Similarly, as long as $A(M)$ exists and $\text{supp}(A(M)) \subseteq \mathcal{G}$, there is a continuously differentiable mapping from $A(M)$ to $N(M)$.

- If $M = I_d$, then $N(M) = 0$.

It follows from the above properties that if M is sufficiently small, the function $N(M)$ exists and satisfies the following properties: (i) $0 = N(I_d)$, and (ii) $N(\cdot)$ is differentiable at $M = I_d$. This implies that there are sufficiently small nonzero numbers η and α_0 such that $\|N(M)\|_1 \leq \eta \|M - I_d\|_1$ whenever $\|M\|_{\max} \leq \alpha_0$. Now, it follows from Lemma 30 that

$$\|N(M)\|_1 \leq (1+\eta)^2 \frac{\|M - I_d\|_1^2}{1 - (\eta+1)\|M - I_d\|_1},$$

or

$$\|N(M)\|_{\max} \leq \frac{(1+\eta)^2 \times (\deg(\mathcal{G}))^2}{1 - (\eta+1)\alpha_0 \times \deg(\mathcal{G})} \|M\|_{\max}^2,$$

if $\|M\|_{\max} \leq \alpha_0$. The inequality (7) is satisfied for the number ζ defined as the maximum of

$$\frac{(1+\eta)^2 \times (\deg(\mathcal{G}))^2}{1 - (\eta+1)\alpha_0 \times \deg(\mathcal{G})}$$

and the finite number

$$\max \left\{ \frac{\beta(\mathcal{G}, \alpha)}{\alpha^2} \middle| \alpha \in (\alpha_0, 1) \right\}.$$

This completes the proof. ■

Proof of Lemma 14 It can be easily verified that

$$(M + M^{(c)})^{-1} = I - (M + M^{(c)} - I) + (M + M^{(c)})^{-1}(M + M^{(c)} - I)^2.$$

This implies that, for a given pair $(i, j) \in \mathcal{G}$, one can write

$$\left((M + M^{(c)})^{-1}\right)_{ij} = -M_{ij} + \left((M + M^{(c)})^{-1}\right)_{i:} \left((M + M^{(c)} - I)^2\right)_{:j}, \tag{28}$$

where $\left((M + M^{(c)})^{-1}\right)_{i:}$ and $\left((M + M^{(c)} - I)^2\right)_{:j}$ are the i^{th} row and j^{th} column of $(M + M^{(c)})^{-1}$ and $(M + M^{(c)} - I)^2$, respectively. Based on (28), the (i, j) entries of M and $(M + M^{(c)})^{-1}$ have opposite signs if

$$|M_{ij}| > \left| \left((M + M^{(c)})^{-1}\right)_{i:} \left((M + M^{(c)} - I)^2\right)_{:j} \right|. \tag{29}$$

To streamline the presentation, $\|M\|_{\max}$ is redefined as $\max_{i,j} |M_{ij}|$ in the rest of the proof. One can write

$$\begin{aligned}
\left\|(M+M^{(c)}-I)^2\right\|_{\max} &\leq \left\|(M-I)^2\right\|_{\max} + \left\|\left(M^{(c)}\right)^2\right\|_{\max} + \left\|M^{(c)}(M-I)\right\|_{\max} + \left\|(M-I)M^{(c)}\right\|_{\max} \\
&\leq \deg(\mathcal{G})\alpha^2 + (d - \deg(\mathcal{G}))\zeta(\mathcal{G})^2\alpha^4 + 2\deg(\mathcal{G})\zeta(\mathcal{G})\alpha^3 \\
&\leq 3\deg(\mathcal{G}) \max\{\alpha^2, \zeta(\mathcal{G})\alpha^3\} + (d - \deg(\mathcal{G}))\zeta(\mathcal{G})^2\alpha^4 \\
&\leq K\alpha^2,
\end{aligned} \tag{30}$$

for some K that only depends on $\deg(\mathcal{G}), \zeta(\mathcal{G})$, and d. Furthermore, assume that

$$\alpha \leq \frac{1}{2\deg(\mathcal{G})\sqrt{\zeta(\mathcal{G})}} = \alpha_0(\mathcal{G}). \tag{31}$$

Note that

$$(M + M^{(c)})^{-1} = I - (M + M^{(c)} - I)(M + M^{(c)})^{-1},$$

which implies that

$$\left\|(M + M^{(c)})^{-1}\right\|_{\max} = 1 + \deg(\mathcal{G}) \max\{\alpha, \zeta(\mathcal{G})\alpha^2\} \left\|(M + M^{(c)})^{-1}\right\|_{\max}, \tag{32}$$

where we have used the fact that $\text{supp}((M + M^{(c)})^{-1}) \subseteq \mathcal{G}$ and hence, its maximum degree is upper bounded by $\deg(\mathcal{G})$. (32), together with the assumption (31) implies that

$$\left\|(M + M^{(c)})^{-1}\right\|_{\max} \leq \frac{1}{1 - \deg(\mathcal{G}) \max\{\alpha, \zeta(\mathcal{G})\alpha^2\}} \leq 2. \tag{33}$$

Combining (30) and (33) with (29) completes the proof. ■

Proof of Lemma 15 Without loss of generality, assume that $\mathcal{G}$ is a tree. Note that if there are disjoint components, the argument made in the sequel can be applied to each connected component of $\mathcal{G}$ separately. Let d_{ij} denote the unique path between every two disparate nodes i and j in $\mathcal{G}$. Furthermore, define $\mathcal{N}(i)$ as the set of all neighbors of node i in $\mathcal{G}$. Consider a matrix M with positive-definite completion and with diagonal elements equal to 1 such that $\|M\|_{\max} \leq \alpha$ and $\text{supp}(M) = \mathcal{G}$. Let N be a matrix with the following entries

$$N_{ij} = \begin{cases} \prod_{(m,t) \in d_{ij}} M_{mt} & \text{if} \quad (i,j) \in (\text{supp}(M))^{(c)}, \\ 0 & \text{otherwise.} \end{cases} \tag{34}$$

Moreover, define

$$A_{ij} = \begin{cases} 1 + \sum_{m \in \mathcal{N}(i)} \frac{M_{mi}^2}{1 - M_{mi}^2} & \text{if} \quad i = j, \\ \frac{-M_{ij}}{1 - M_{ij}^2} & \text{if} \quad (i,j) \in \text{supp}(M), \\ 0 & \text{otherwise.} \end{cases} \tag{35}$$

The goal is to show that the matrix N is the unique inverse-consistent complement of M. First, note that $\text{supp}(N) = (\text{supp}(M))^{(c)}$ and $\text{supp}(M) = \text{supp}(A)$. Next, it is desirable to prove that $(M + N)^{-1} = A$ or equivalently $(M + N)A = I$. Upon defining $T = (M + N)A$, one can write

$$T_{ii} = \sum_{m=1}^{d} (M_{im} + \mathcal{N}_{im}) A_{mi} = 1 + \sum_{m \in \mathcal{N}(i)} \frac{M_{mi}^2}{1 - M_{mi}^2} - \sum_{m \in \mathcal{N}(i)} \frac{M_{mi}^2}{1 - M_{mi}^2} = 1.$$

Moreover, for every pair of nodes i and j, define D_{ij} as $\prod_{(k,t) \in d_{ij}} M_{kt}$ if $i \neq j$ and as 1 if $i = j$.

Consider a pair of distinct nodes i and j. Let t denote the node adjacent to j in d_{ij} (note that we may have $t = i$). It can be verified that

$$T_{ij} = \sum_{m=1}^{d}(M_{im}+N_{im})A_{mj} = D_{ij}\left(1+\sum_{m\in\mathcal{N}(j)}\frac{M_{mj}^2}{1-M_{mj}^2}\right) - D_{it}\left(\frac{M_{tj}}{1-M_{tj}^2}\right) - \sum_{\substack{m\in\mathcal{N}(j)\\ m\neq t}} D_{im}\frac{M_{mj}}{1-M_{mj}^2}. \tag{36}$$

Furthermore,

$$\begin{aligned} D_{ij} &= D_{it}M_{tj}, \\ D_{im} &= D_{it}M_{tj}M_{jm}, \qquad \forall\ m\in\mathcal{N}(j),\ m\neq t. \end{aligned} \tag{37}$$

Plugging (37) into (36) yields that

$$T_{ij} = D_{it}M_{tj}\left(\frac{1}{1-M_{tj}^2}+\sum_{\substack{m\in\mathcal{N}(j)\\ m\neq t}}\frac{M_{mj}^2}{1-M_{mj}^2}\right) - D_{it}\left(\frac{M_{tj}}{1-M_{tj}^2}\right) - D_{it}M_{tj}\sum_{\substack{m\in\mathcal{N}(j)\\ m\neq t}}\frac{M_{mj}^2}{1-M_{mj}^2} = 0.$$

Hence, $T = I$. Finally, we need to show that $M + N \succ 0$. To this end, it suffices to prove that $A \succ 0$. Note that A can be written as $I + \sum_{(i,j)\in\mathcal{G}} L^{(i,j)}$, where $L^{(i,j)}$ is defined as

$$L_{rl}^{(i,j)} = \begin{cases} \frac{M_{ij}^2}{1-M_{ij}^2} & \text{if} \quad r = l = i \text{ or } j, \\ \frac{-M_{ij}}{1-M_{ij}^2} & \text{if} \quad (r,l) = (i,j), \\ 0 & \text{otherwise.} \end{cases}$$

Consider the term x^TAx for an arbitrary vector $x\in\mathbb{R}^d$. One can verify that

$$\begin{aligned} x^TAx =& \sum_{i=1}^{d}x_i^2 + \sum_{(i,j)\in\mathcal{G}} x^TL^{(i,j)}x \\ =& \sum_{i=1}^{d}x_i^2 + \sum_{(i,j)\in\mathcal{G}}\left(\frac{M_{ij}^2}{1-M_{ij}^2}\right)x_i^2 + \left(\frac{M_{ij}^2}{1-M_{ij}^2}\right)x_j^2 - \left(\frac{2M_{ij}}{1-M_{ij}^2}\right)x_ix_j. \end{aligned} \tag{38}$$

Without loss of generality, assume that the graph is a rooted tree with the root at node d. Assume that each edge (i,j) defines a direction that is toward the root. Then, it follows from (38) that

$$\begin{aligned} x^TAx =& x_d^2 + \sum_{(i,j)\in\mathcal{G}} x_i^2 + \left(\frac{M_{ij}^2}{1-M_{ij}^2}\right)x_i^2 + \left(\frac{M_{ij}^2}{1-M_{ij}^2}\right)x_j^2 - \left(\frac{2M_{ij}}{1-M_{ij}^2}\right)x_ix_j \\ =& x_d^2 + \sum_{(i,j)\in\mathcal{G}}\left(\frac{1}{1-M_{ij}^2}\right)x_i^2 + \left(\frac{M_{ij}^2}{1-M_{ij}^2}\right)x_j^2 - \left(\frac{2M_{ij}}{1-M_{ij}^2}\right)x_ix_j \\ =& x_d^2 + \sum_{(i,j)\in\mathcal{G}}\frac{(x_i - M_{ij}x_j)^2}{1-M_{ij}^2} \geq 0. \end{aligned}$$

Therefore, $M + N \succeq 0$ and subsequently $M + N \succ 0$ (because it is invertible). Hence, according to Definition 5 and Lemma 6, the matrix N is the unique inverse-consistent compliment of M. On the other hand, it follows from the definition of N that $\|N\|_{\max} \leq \alpha^2$ and consequently $\beta(\mathcal{G},\alpha) \leq \alpha^2$. Now, suppose that $\mathcal{G}$ includes a path of length at least 2, e.g., the edges $(1,2)$ and $(2,3)$ belong to $\mathcal{G}$. By setting $M_{12} = M_{23} = \alpha$ and choosing sufficiently small values for those entries of M corresponding to the remaining edges in $\mathcal{G}$, the matrix M becomes positive-definite with a trivial positive-definite completion and we obtain $\|N\|_{\max} = \alpha^2$. This completes the proof. ■

Proof of Theorem 18 To prove this theorem, first consider the following matrix

$$\hat{S}_{ij} = \begin{cases} 1 + \sum\limits_{(i,m)\in\mathcal{E}^{\text{opt}}} \frac{(\tilde{\Sigma}_{im}^{\text{res}})^2}{1-(\tilde{\Sigma}_{im}^{\text{res}})^2} & \text{if} \quad i = j, \\ \frac{-\tilde{\Sigma}_{ij}^{\text{res}}}{1-(\tilde{\Sigma}_{ij}^{\text{res}})^2} & \text{if} \quad (i,j)\in\mathcal{E}^{\text{opt}}, \\ 0 & \text{otherwise.} \end{cases} \tag{39}$$

In what follows, we will show that $\hat{S} = \tilde{S}$, where $\tilde{S}$ is the optimal solution for the normalized GL. This, together with Lemma 29 implies that (11) is indeed optimal for the GL.

First, note that there exists a matrix N such that $\tilde{S}^{-1} = M + N$, where M is defined as

$$M_{ij} = \begin{cases} \tilde{\Sigma}_{ij} + \tilde{\lambda}_{ij}\times\text{sign}(\tilde{S}_{ij}) & \text{if} \quad (i,j)\in\text{supp}(\tilde{S}), \\ 1 & \text{if} \quad i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{40}$$

Clearly, $\text{supp}(\tilde{S}) = \text{supp}(M)$. Furthermore, $M = I_d + \tilde{T}(\lambda)$, where $(i,j)^{\text{th}}$ entry of $\tilde{T}(\lambda)$ is equal to $\tilde{\Sigma}_{ij} + \tilde{\lambda}_{ij}\text{sign}(S_{ij}^{\text{opt}})$ for every $(i,j)\in\text{supp}(S^{\text{opt}})$ and it is equal to zero otherwise. Subsequently, $M = D^{-1/2}(D + T(\lambda))D^{-1/2}$ and hence, $D + T(\lambda) \succ 0$ implies $M \succ 0$. By combining $N = (\tilde{S})^{-1} - M$ with (40) and exploiting the optimality conditions in (14), one can verify that $\text{supp}(N) \subseteq (\text{supp}(M))^{(c)}$ and $\text{supp}(\tilde{S}) = \text{supp}\left((M+N)^{-1}\right) \subseteq \text{supp}(M)$. Therefore, according to Lemma 6, the matrix N is the unique inverse-complement of M. Moreover, since M is sign-consistent, the equation $\text{sign}(M_{ij}) = -\text{sign}(\tilde{S}_{ij})$ holds for every $(i,j)\in\text{supp}(\tilde{S})$. This leads to the relations $\text{sign}(\Sigma_{ij}) = -\text{sign}(\tilde{S}_{ij})$ and

$$M_{ij} = \tilde{\Sigma}_{ij}^{\text{res}}, \tag{41a}$$

$$|\tilde{\Sigma}_{ij}| > \tilde{\lambda}_{ij}, \tag{41b}$$

for every $(i,j)\in\text{supp}(\tilde{S})$. Part 1 of the theorem is an immediate consequence of (41b). On the other hand, based on the argument made in the proof of Lemma 15, the matrix N can be obtained as

$$N_{ij} = \begin{cases} \prod_{(m,t)\in d_{ij}} M_{mt} & \text{if} \quad d_{ij}\neq\emptyset \text{ and } (i,j)\in(\text{supp}(M))^{(c)}, \\ 0 & \text{otherwise,} \end{cases} \tag{42}$$

where d_{ij} denotes the unique path between nodes i and j in $\text{supp}(\tilde{S})$ if they belong to the same connected component in $\text{supp}(\tilde{S})$, and d_{ij} is empty if there is no path between nodes

i and j. Similar to the proof of Lemma 15, one can show that (11) is equal to $(M+N)^{-1}$. This completes the proof of the second part of the theorem. ■

Proof of Theorem 19 Based on Lemmas 15 and 16, the conditions introduced in Theorem 12 can be reduced to conditions (2-ii) and (2-iii) in Theorem 19 if $\text{supp}(\Sigma^{\text{res}})$ is acyclic and therefore, $\mathcal{E}^{\text{opt}} = \mathcal{E}^{\text{res}}$. Moreover, suppose that M is set to $I_d + \tilde{\Sigma}^{\text{res}}$, and that the matrices N and A are defined as (34) and (35), respectively. Similar to the proof of Theorem 12, it can be verified that (39) satisfies all the KKT conditions for the normalized GL (21). Therefore, due to Lemma 29, (11) is the unique solution of the GL. The details are omitted for brevity. ■

Proof of Corollary 20 Given Σ and λ, the matrix Σ^{res} can be computed in $\mathcal{O}(d^2)$. Moreover, Condition (2-i) in Theorem 19 can be checked using the Depth-First-Search algorithm, which has the time complexity of $\mathcal{O}(d^2)$ in the worst case (Ahuja et al., 1993). If the graph is cyclic, Theorem 19 cannot be used. Otherwise, we consider Condition (2-ii). For matrices with acyclic support graphs, the Cholesky Decomposition can be computed in $\mathcal{O}(d)$, from which the positive-definiteness of the matrix can be checked (Vandenberghe and Andersen 2015). The complexity of checking Condition (2-iii) is equivalent to that of finding its left and right hand sides, which can be done in $\mathcal{O}(d)$ and $\mathcal{O}(d^2)$, respectively. Finally, since (11) can be used only if the support graph of Σ^{res} is acyclic, one can easily verify that the complexity of obtaining S^{opt} using (11) is at most $\mathcal{O}(d)$. This completes the proof of Corollary 20. ■

The remainder of this section is devoted to proving approximation bounds for the derived closed-form solution when the acyclic assumption on the support graph of the thresholded sample covariance matrix is not necessarily acyclic. The shorthand notations c, deg, $\mathcal{P}_{ij}$ and $P_{\max}$ will be used instead of $c(\text{supp}(\Sigma^{\text{res}}))$, $\deg(\text{supp}(\Sigma^{\text{res}}))$, $\mathcal{P}_{ij}(\text{supp}(\Sigma^{\text{res}}))$ and $P_{\max}(\text{supp}(\Sigma^{\text{res}}))$, respectively. First, the approximation error of the closed-form solution for the normalized GL will be analyzed. Then, the result will be generalized to the GL via the key equality in Lemma 29. To prove Theorem 26, the first step is to generalize the definition of the matrix N in (42) and show that this generalized matrix is an approximate inverse-consistent complement of $I_d + \tilde{\Sigma}^{\text{res}}$. Without loss of generality, assume that $\text{supp}(\Sigma^{\text{res}})$ is connected. If there are disjoint components in $\text{supp}(\Sigma^{\text{res}})$, the argument made in the sequel can be used for every connected component due to the decomposition rule for the GL (see Mazumder and Hastie (2012)). Let M be equal to $I_d + \tilde{\Sigma}^{\text{res}}$. Consider the matrix N as

$$N_{ij} = \begin{cases} \sum_{d_{ij} \in \mathcal{P}_{ij}} \prod_{(m,t) \in d_{ij}} M_{mt} & \text{if} \quad (i,j) \in (\text{supp}(M))^{(c)}, \\ \sum_{d_{ij} \in \mathcal{P}_{ij} \setminus \{(i,j)\}} \prod_{(m,t) \in d_{ij}} M_{mt} & \text{if} \quad (i,j) \in (\text{supp}(M)), \\ 0 & \text{otherwise.} \end{cases} \tag{43}$$

It can be verified that $M + N = R$, where

$$R_{ij} = \begin{cases} \sum_{d_{ij} \in \mathcal{P}_{ij}} \prod_{(m,t) \in d_{ij}} M_{mt} & \text{if} \quad i \neq j, \\ 1 & \text{if} \quad i = j. \end{cases} \tag{44}$$

For each simple path between the pair of nodes i and j, define its length as the multiplication of the entries of M corresponding to the edges of the path. Based on this definition, R_{ij} is

equal to the sum of the lengths of all nonidentical simple paths between nodes i and j in $\text{supp}(M)$. Denote d^s_{ij} as any shortest path between nodes i and j in $\text{supp}(M)$ (recall that $\text{supp}(M)$ is unweighted), and let R^s be given by

$$R^s_{ij} = \begin{cases} \prod_{(m,t) \in d^s_{ij}} M_{mt} & \text{if} \quad i \neq j, \\ 1 & \text{if} \quad i = j. \end{cases}$$

Note that R^s collects the length of the shortest path between every two nodes in $\text{supp}(M)$. The following lemmas are crucial to prove Theorem 26.

Lemma 31 *Given two nodes i and j in $\text{supp}(\Sigma^{\text{res}})$, suppose that $\mathcal{P}_{ij} \backslash d^s_{ij}$ is non-empty. Then, the length of every path d_{ij} in $\mathcal{P}_{ij} \backslash d^s_{ij}$ is at least $\lceil c/2 \rceil$.*

Proof Consider a path d_{ij} in $\mathcal{P}_{ij} \backslash d^s_{ij}$. The subgraph $d_{ij} \cup d^s_{ij}$ has a cycle. Since the length of this cycle is at least c, the segment of this cycle that resides in d_{ij} should have the length of at least $\lceil c/2 \rceil$; otherwise d^s_{ij} is not the shortest path between the nodes i and j. This implies that the length of d_{ij} is at least $\lceil c/2 \rceil$. ■

Lemma 32 *Let M be equal to $I_d + \tilde{\Sigma}^{\text{res}}$. The inequalities*

$$\left| R_{ij} - R^s_{k'j} M_{ik'} \right| \leq (|\mathcal{P}_{ij}|_0 - 1) \left(\|\tilde{\Sigma}^{\text{res}}\|_{\max} \right)^{\lceil \frac{c}{2} \rceil}, \tag{45a}$$

$$\left| R_{kj} - R^s_{k'j} M_{ik'} M_{ik} \right| \leq (|\mathcal{P}_{kj}|_0 - 1) \left(\|\tilde{\Sigma}^{\text{res}}\|_{\max} \right)^{\lceil \frac{c}{2} \rceil - 1} \tag{45b}$$

hold if $i \neq j$, where k' is the node adjacent to i in d^s_{ij} and $k \in \mathcal{N}(i) \backslash k'$.

Proof First, we show the validity of (45a). Due to (44), one can write

$$R_{ij} = R^s_{ij} + \sum_{d_{ij} \in \mathcal{P}_{ij} \backslash d^s_{ij}} \prod_{(m,t) \in d_{ij}} M_{mt}. \tag{46}$$

If $\mathcal{P}_{ij} \backslash d^s_{ij}$ is empty, then the equation $R_{ij} = R^s_{k'j} M_{ik'}$ and therefore (45a) hold. Now, assume that $\mathcal{P}_{ij} \backslash d^s_{ij}$ is not empty. Due to Lemma 31, we have

$$-\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max} \right)^{\lceil \frac{c}{2} \rceil} \leq \prod_{(m,t) \in d_{ij}} M_{mt} \leq \left(\|\tilde{\Sigma}^{\text{res}}\|_{\max} \right)^{\lceil \frac{c}{2} \rceil},$$

for every $d_{ij} \in \mathcal{P}_{ij} \backslash d^s_{ij}$. The above inequalities, together with (46) and the equation $R^s_{ij} = R^s_{k'j} M_{ik'}$, result in (45a). To prove (45b), define $\hat{d}_{kj}$ as $d^s_{ij} \cup \{(i,k)\}$ (note that $\hat{d}_{kj}$ is not necessarily equal to d^s_{kj}). It yields that

$$R_{kj} = R^s_{ij} M_{ik} + \sum_{d_{kj} \in \mathcal{P}_{kj} \backslash \hat{d}_{kj}} \prod_{(m,t) \in d_{kj}} M_{mt}. \tag{47}$$

In light of Lemma 31, the length of every path $d_{kj} \in \mathcal{P}_{kj} \backslash \hat{d}_{kj}$ is lower bounded by $\lceil c/2 \rceil - 1$. This implies that

$$-\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil \frac{c}{2} \rceil - 1} \leq \prod_{(m,t)\in d_{ij}} M_{mt} \leq \left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil \frac{c}{2} \rceil - 1}, \tag{48}$$

for every $d_{kj} \in \mathcal{P}_{kj} \backslash \hat{d}_{kj}$. Combining $R^s_{ij} M_{ik} = R^s_{k'j} M_{ik'} M_{ik}$ with (47) and (48) leads to the inequality (45b). ■

Lemma 33 *The following inequality holds*

$$\frac{\deg}{1 - \|\tilde{\Sigma}^{\text{res}}\|^2_{\max}} \leq \delta,$$

where δ defined as (17).

Proof The proof is straightforward and is omitted for brevity. ■

Proof of Theorem 26 Consider the normalized GL and define the following explicit formula for $\tilde{A}$

$$\tilde{A}_{ij} = \begin{cases} 1 + \sum\limits_{(i,m)\in\mathcal{E}^{\text{opt}}} \frac{(\tilde{\Sigma}^{\text{res}}_{im})^2}{1-(\tilde{\Sigma}^{\text{res}}_{im})^2} & \text{if} \quad i = j, \\ \frac{-\tilde{\Sigma}^{\text{res}}_{ij}}{1-(\tilde{\Sigma}^{\text{res}}_{ij})^2} & \text{if} \quad (i,j) \in \mathcal{E}^{\text{res}}, \\ 0 & \text{otherwise.} \end{cases} \tag{49}$$

Let M be equal to $I_d + \tilde{\Sigma}^{\text{res}}$. Furthermore, define

$$\tilde{\epsilon} = \delta \cdot (P_{\max}(\text{supp}(\Sigma^{\text{res}})) - 1) \cdot \left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\left\lceil \frac{c(\text{supp}(\Sigma^{\text{res}}))}{2} \right\rceil}.$$

In order to prove the theorem, we use the matrix N defined in (43), and first show that $M + N$ is an $\tilde{\epsilon}$-relaxed inverse of $\tilde{A}$ and that the pair $(\tilde{A}, M + N)$ satisfies the $\tilde{\epsilon}$-relaxed KKT conditions.

Consider the matrix T defined as $T = \tilde{A}(M + N)$ and recall that $M + N = R$. One can write

$$T_{ii} = \sum_{m=1}^{d} \tilde{A}_{im} R_{mi} = \left(1 + \sum_{m\in\mathcal{N}(i)} \frac{{M_{im}}^2}{1 - {M_{im}}^2}\right) - \sum_{m\in\mathcal{N}(i)} \frac{M_{im}}{1 - {M_{im}}^2} R_{mi}. \tag{50}$$

Note that since $\{(m,i)\} \in \mathcal{P}_{mi}$ for every $m \in \mathcal{N}(i)$, we have

$$R_{mi} = M_{mi} + \sum_{d_{mi}\in\mathcal{P}_{mi}\backslash\{(m,i)\}} \prod_{(r,t)\in d_{mi}} M_{rt}.$$

If $\mathcal{P}_{mi}\backslash\{(m,i)\}$ is empty, then $R_{mi} = M_{mi}$ and $T_{ii} = 1$. Otherwise, since the length of the minimum-length cycle is c, the length of every path $d_{mi} \in \mathcal{P}_{mi}\backslash\{(m,i)\}$ is at least $c - 1$. This yields that

$$M_{mi} - (|\mathcal{P}_{mi}|_0 - 1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{c-1} \leq R_{mi} \leq M_{mi} + (|\mathcal{P}_{mi}|_0 - 1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{c-1}. \tag{51}$$

Combining (51) and (50) leads to

$$|T_{ii} - 1| \leq (|\mathcal{P}_{mi}|_0 - 1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{c-1}\left(\sum_{m\in\mathcal{N}(i)} \frac{M_{im}}{1 - {M_{im}}^2}\right) \leq \deg(P_{\max} - 1)\frac{\|\tilde{\Sigma}^{\text{res}}\|^c_{\max}}{1 - \|\tilde{\Sigma}^{\text{res}}\|^2_{\max}} \leq \tilde{\epsilon}, \tag{52}$$

where the last inequality is due to Lemma 33 and the fact that $\lceil \frac{c}{2} \rceil \leq c$ for $c \geq 3$. Now, consider T_{ij} for a pair (i,j) such that $i \neq j$. We have

$$T_{ij} = \sum_{m=1}^{d} \tilde{A}_{im} R_{mj} = \left(1 + \sum_{m\in\mathcal{N}(i)} \frac{{M_{im}}^2}{1 - {M_{im}}^2}\right) R_{ij} - \sum_{m\in\mathcal{N}(i)} \frac{M_{im}}{1 - {M_{im}}^2} R_{mj}. \tag{53}$$

According to Lemma 31, one can write

$$R^s_{m'j} M_{im'} - (|\mathcal{P}_{ij}|_0 - 1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil \frac{c}{2} \rceil} \leq R_{ij} \leq R^s_{m'j} M_{im'} + (|\mathcal{P}_{ij}|_0 - 1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil \frac{c}{2} \rceil}, \tag{54a}$$

$$\begin{aligned} R^s_{m'j} M_{im'} M_{im} - (|\mathcal{P}_{mj}|_0 - 1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil \frac{c}{2} \rceil - 1} &\leq R_{mj} \\ &\leq R^s_{m'j} M_{im'} M_{im} + (|\mathcal{P}_{mj}|_0 - 1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil \frac{c}{2} \rceil - 1}, \end{aligned} \tag{54b}$$

where m' is the node adjacent to i in d^s_{ij} and $m \in \mathcal{N}(i)\backslash m'$. Note that if $\mathcal{N}(i)\backslash m'$ is empty, then $R_{ij} = R^s_{m'j} M_{im'}$ and $R_{mj} = R^s_{m'j} M_{im'} \tilde{\Sigma}^{\text{res}}_{im}$. In this case, an argument similar to the proof of Lemma 15 can be made to show that $T_{ij} = 0$. Now, assume that $\mathcal{N}(i)\backslash m'$ is not empty. One can write

$$|T_{ij} - F_{ij}| \overset{(a)}{=} |T_{ij}| \overset{(b)}{\leq} \tilde{\epsilon}, \tag{55}$$

where

$$\begin{aligned} F_{ij} = &\left(\frac{1}{1 - {M_{im'}}^2} + \sum_{m\in\mathcal{N}(i)\backslash m'} \frac{{M_{im}}^2}{1 - {M_{im}}^2}\right) R^s_{m'j} M_{im'} - \frac{M_{im'}}{1 - {M_{im'}}^2} R^s_{m'j} \\ &- \sum_{m\in\mathcal{N}(i)\backslash m'} \frac{{M_{im}}^2}{1 - {M_{im}}^2} R^s_{m'j} M_{im'} M_{im}. \end{aligned}$$

Note that the relation (a) can be verified by the fact that $F_{ij} = 0$ and the inequality (b) is obtained by combining (53) with (54a) and (54b). The inequalities (52) and (55) imply that $M + N$ is an $\tilde{\epsilon}$-relaxed inverse of $\tilde{A}$.

Now, it will be shown that the pair $(\tilde{A}, M+N)$ satisfies the $\tilde{\epsilon}$-relaxed KKT conditions. Note that $M_{ii}+N_{ii}=M_{ii}=\tilde{\Sigma}_{ii}$ and, hence, (15a) is satisfied. To prove (15b), since $\text{sign}(\tilde{A}_{ij})=-\text{sign}(M_{ij})=-\text{sign}(\tilde{\Sigma}_{ij})$, it can be concluded that

$$M_{ij}+N_{ij}=(\tilde{\Sigma}_{ij}-\tilde{\lambda}_{ij}\times\text{sign}(\Sigma_{ij}))+N_{ij}=(\tilde{\Sigma}_{ij}+\tilde{\lambda}_{ij}\times\text{sign}(\tilde{A}_{ij}))+N_{ij},$$

for every (i,j) such that $i\neq j$ and $\tilde{A}_{ij}\neq 0$. Due to the definition of N and the fact that $(i,j)\in\text{supp}(M)$, we have $|N_{ij}|\leq(P_{\max}-1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{c-1}$. Hence,

$$|M_{ij}+N_{ij}-(\tilde{\Sigma}_{ij}+\tilde{\lambda}_{ij}\times\text{sign}(\tilde{A}_{ij}))|\leq\epsilon,$$

for every (i,j) such that $i\neq j$ and $\tilde{A}_{ij}\neq 0$. Therefore, the pair $(\tilde{A}, M+N)$ satisfies (15b). Finally, consider a pair (i,j) such that $i\neq j$ and $\tilde{A}_{ij}=0$. One can write

$$M_{ij}+N_{ij}=R^s_{ij}+\sum_{d_{ij}\in\mathcal{P}_{ij}\backslash d^s_{ij}}\prod_{(m,t)\in d_{ij}}\tilde{\Sigma}^{\text{res}}_{mt}.$$

If $\mathcal{P}_{ij}\backslash d^s_{ij}$ is empty, a set of inequalities similar to (25) and (26) can be obtained to prove (15c). Now, assume that $\mathcal{P}_{ij}\backslash d^s_{ij}$ is not empty. The length of d^s_{ij} is at least 2 since there is no direct edge between nodes i and j. Hence, $|R^s_{ij}|\leq\|\tilde{\Sigma}^{\text{res}}\|^2_{\max}$. Furthermore, due to Lemma (31), the length of every path $d_{ij}\in\mathcal{P}_{ij}\backslash d^s_{ij}$ is at least $\lceil c/2\rceil$. This leads to

$$\begin{aligned}|M_{ij}+N_{ij}|&\leq\|\tilde{\Sigma}^{\text{res}}\|^2_{\max}+(P_{\max}-1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil\frac{c}{2}\rceil}\\&\leq\min_{\substack{k\neq l\\(k,l)\notin\text{supp}(\Sigma^{\text{res}})}}(\tilde{\lambda}_{kl}-|\tilde{\Sigma}^{\text{res}}_{kl}|)+(P_{\max}-1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil\frac{c}{2}\rceil}\\&\leq\tilde{\lambda}_{ij}-|\tilde{\Sigma}^{\text{res}}_{ij}|+(P_{\max}-1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil\frac{c}{2}\rceil},\end{aligned}$$

where the last inequality follows from Condition (2-ii) in the theorem. Therefore,

$$\begin{aligned}|M_{ij}+N_{ij}-\tilde{\Sigma}_{ij}|\leq|M_{ij}+N_{ij}|+|\tilde{\Sigma}_{ij}|&\leq\tilde{\lambda}_{ij}-|\tilde{\Sigma}^{\text{res}}_{ij}|+|\tilde{\Sigma}^{\text{res}}_{ij}|+(P_{\max}-1)\left(\|\tilde{\Sigma}^{\text{res}}\|_{\max}\right)^{\lceil\frac{c}{2}\rceil}\\&\leq\tilde{\lambda}_{ij}+\tilde{\epsilon}.\end{aligned}$$

This shows that $(\tilde{A}, M+N)$ indeed satisfies the $\tilde{\epsilon}$-relaxed KKT conditions for the normalized GL. Finally, we consider the explicit solution A defined as (13). The following statements hold:

1. the matrix $D^{1/2}(M+N)D^{1/2}$ is ϵ-relaxed inverse of A. To see this, note that

$$\begin{aligned}A\left(D^{1/2}(M+N)D^{1/2}\right)&=D^{-1/2}\tilde{A}D^{-1/2}D^{1/2}(M+N)D^{1/2}\\&=D^{-1/2}TD^{1/2}\\&=I_d+E,\end{aligned}$$

where $\|E\|_{\max}\leq\sqrt{\frac{\Sigma_{\max}}{\Sigma_{\min}}}\tilde{\epsilon}\leq\epsilon$.

2. The pair $(A, D^{1/2}(M+N)D^{1/2})$ satisfies the ϵ-relaxed KKT conditions. Note that it is already shown that $(\tilde{A}, M+N)$ satisfies the following inequalities

$$\begin{aligned}&(M+N)_{ij}=\tilde{\Sigma}_{ij} && \text{if}\quad i=j, && (56a)\\&\left|(M+N)_{ij}-\left(\tilde{\Sigma}_{ij}+\tilde{\lambda}_{ij}\times\text{sign}(\tilde{A}_{ij})\right)\right|\leq\tilde{\epsilon} && \text{if}\quad \tilde{A}_{ij}\neq 0, && (56b)\\&\left|(M+N)_{ij}-\tilde{\Sigma}_{ij}\right|\leq\tilde{\lambda}_{ij}+\tilde{\epsilon} && \text{if}\quad \tilde{A}_{ij}=0. && (56c)\end{aligned}$$

Replacing $M+N$ with $D^{1/2}(M+N)D^{1/2}$ and modifying (56) accordingly, one can verify that $(A, D^{1/2}(M+N)D^{1/2})$ satisfies ϵ-relaxed KKT conditions for the GL, where

$$\epsilon=\max\left\{\Sigma_{\max},\sqrt{\frac{\Sigma_{\max}}{\Sigma_{\min}}}\right\}\tilde{\epsilon}.$$

This completes the proof. ■

Proof of Theorem 27 Due to Theorem 26, the equation

$$D^{1/2}(M+N)D^{1/2}=A^{-1}+A^{-1}E \tag{57}$$

holds for every λ greater than or equal to λ_0, where $\|E\|_{\max}\leq\epsilon$. Since the pair $(A, D^{1/2}(M+N)D^{1/2})$ satisfies the ϵ-relaxed KKT conditions, it follows from (57) that

$$\begin{aligned}&(A)^{-1}_{ij}=\Sigma_{ij}-(A^{-1}E)_{ij}=\hat{\Sigma}_{ij} && \text{if}\quad i=j, && (58a)\\&(A)^{-1}_{ij}=\underbrace{\Sigma_{ij}+t_{ij}\epsilon-(A^{-1}E)_{ij}}_{\hat{\Sigma}_{ij}}+\lambda\times\text{sign}(A_{ij}) && \text{if}\quad A_{ij}\neq 0, && (58b)\\&\underbrace{\Sigma_{ij}+s_{ij}\epsilon-(A^{-1}E)_{ij}}_{\hat{\Sigma}_{ij}}-\lambda\leq(A)^{-1}_{ij}\leq\underbrace{\Sigma_{ij}+s_{ij}\epsilon-(A^{-1}E)_{ij}}_{\hat{\Sigma}_{ij}}+\lambda && \text{if}\quad A_{ij}=0, && (58c)\end{aligned}$$

for some numbers t_{ij} and s_{ij} in the interval $[-1,1]$. To complete the proof, it suffices to show that the matrix F defined as

$$\Sigma_{ij}-\hat{\Sigma}_{ij}=F_{ij}=\begin{cases}-(A^{-1}E)_{ij} & \text{if}\quad i=j,\\ t_{ij}\epsilon-(A^{-1}E)_{ij} & \text{if}\quad A_{ij}\neq 0,\\ s_{ij}\epsilon-(A^{-1}E)_{ij} & \text{if}\quad A_{ij}=0\end{cases} \tag{59}$$

satisfies the inequality $\|F\|_2\leq d_{\max}\left(1/\mu_{\min}(A)+1\right)\epsilon$. To this end, it is enough to prove that $\|A^{-1}E\|_2\leq(d_{\max}/\mu_{\min}(A))\epsilon$, since $\|F-A^{-1}E\|_2\leq d_{\max}(A)\epsilon$. One can write

$$\|A^{-1}E\|_2\leq\|A^{-1}\|_2\|E\|_2\leq d_{\max}(A)\|A^{-1}\|_2\|E\|_{\max}=\left(\frac{d_{\max}(A)}{\mu_{\min}(A)}\right)\epsilon,$$

which shows the validity of (18).

Next, we prove the inequality (19). The following chain of inequalities hold

$$\begin{aligned}
-\log\det(A)+\operatorname{trace}(\hat{\Sigma}A)+\lambda\|A\|_{1,\text{off}} &= \underbrace{-\log\det(A)+\operatorname{trace}(\Sigma A)+\lambda\|A\|_{1,\text{off}}}_{f(A)} \\
&\quad +\operatorname{trace}((\hat{\Sigma}-\Sigma)A) \\
&\overset{(a)}{\leq} -\log\det(S^{\text{opt}})+\operatorname{trace}(\hat{\Sigma}S^{\text{opt}})+\lambda\|S^{\text{opt}}\|_{1,\text{off}} \\
&= \underbrace{-\log\det(S^{\text{opt}})+\operatorname{trace}(\Sigma S^{\text{opt}})+\lambda\|S^{\text{opt}}\|_{1,\text{off}}}_{f^*} \\
&\quad +\operatorname{trace}((\hat{\Sigma}-\Sigma)S^{\text{opt}}),
\end{aligned}$$

where (a) is due to the fact that A is optimal for the GL with the perturbed sample covariances. This implies that

$$\begin{aligned}
f(A)-f^* &\leq \operatorname{trace}((\tilde{\Sigma}-\Sigma)(S^{\text{opt}}-A)) \\
&\leq \|\tilde{\Sigma}-\Sigma\|_2(\|S^{\text{opt}}\|_2+\|A\|_2) \\
&\leq \left(\mu_{\max}(A)+\mu_{\max}(S^{\text{opt}})\right) d_{\max}(A)\left(\frac{1}{\mu_{\min}(A)}+1\right)\epsilon.
\end{aligned}$$

■

References

Caltrans Performance Management System (PeMS), 2017. URL `http://pems.dot.ca.gov`.

Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows: theory, algorithms, and applications*. Pearson, 1993.

Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning*, 4(1): 1–106, 2012.

Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research*, 9:485–516, 2008.

Steven J. Benson, Yinyu Ye, and Xiong Zhang. Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM Journal on Optimization*, 10(2):443–461, 2000.

Peter Bühlmann and Sara Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

Emmanuel Candes and Justin Romberg. Sparsity and incoherence in compressive sampling. *Inverse Problems*, 23(3):969–985, 2007.

Thomas Frederick Coleman and Yuying Li, editors. *Large-scale numerical optimization*, volume 46. SIAM, 1990.

Patrick Danaher, Pei Wang, and Daniela M Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):373–397, 2014.

Jianqing Fan and Jinchi Lv. A selective overview of variable selection in high dimensional feature space. *Statistica Sinica*, 20(1):101, 2010.

Makan Fardad, Fu Lin, and Mihailo R. Jovanović. Sparsity-promoting optimal control for a class of distributed systems. *American Control Conference*, pages 2050–2055, 2011.

Salar Fattahi and Javad Lavaei. On the convexity of optimal decentralized control problem and sparsity path. In *American Control Conference (ACC), 2017*, pages 3359–3366. IEEE, 2017.

Olivier Fercoq, Alexandre Gramfort, and Joseph Salmon. Mind the duality gap: safer rules for the lasso. *arXiv preprint arXiv:1505.03410*, 2015.

Lino Figueiredo, Isabel Jesus, JA Tenreiro Machado, Jose Rui Ferreira, and JL Martins De Carvalho. Towards the development of intelligent transportation systems. *IEEE Intelligent Transportation Systems*, pages 1206–1211, 2001.

Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing*, volume 1. Basel: Birkhäuser, 2013.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

Jochen Garcke, Michael Griebel, and Michael Thess. Data mining with sparse grids. *Computing*, 67(3):225–253, 2001.

Laurent El Ghaoui, Vivian Viallon, and Tarek Rabbani. Safe feature elimination for the lasso and sparse supervised learning problems. *arXiv preprint arXiv:1009.4219*, 2010.

Cho-Jui Hsieh, Mátyás A. Sustik, Inderjit S. Dhillon, and Pradeep Ravikumar. Quic: quadratic approximation for sparse inverse covariance estimation. *Journal of Machine Learning Research*, 15(1):2911–2947, 2014.

Nicole Krämer, Juliane Schäfer, and Anne-Laure Boulesteix. Regularized estimation of large-scale gene association networks using graphical gaussian models. *BMC bioinformatics*, 10(1):384, 2009.

Han Liu, Kathryn Roeder, and Larry Wasserman. Stability approach to regularization selection (stars) for high dimensional graphical models. In *Advances in Neural Information Processing Systems*, pages 1432–1440, 2010.

Rahul Mazumder and Trevor Hastie. Exact covariance thresholding into connected components for large-scale graphical lasso. *Journal of Machine Learning Research*, 13:781–794, 2012.

Shanmugavelayutham Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends® in Theoretical Computer Science*, 1(2):117–236, 2005.

Habibollah Nassiri and Rafegh Aghamohammadi. A new analytic neuro-fuzzy model for work zone capacity estimation. *Transportation Research Board 96th Annual Meeting*, 17 (06061), 2017.

Eugene Ndiaye, Olivier Fercoq, Alexandre Gramfort, and Joseph Salmon. Gap safe screening rules for sparse multi-task and multi-class models. In *Advances in Neural Information Processing Systems*, pages 811–819, 2015.

Lishan Qiao, Songcan Chen, and Xiaoyang Tan. Sparsity preserving projections with applications to face recognition. *Pattern Recognition*, 43(1):331–341, 2010.

Pradeep Ravikumar, Martin J Wainwright, Garvesh Raskutti, and Bin Yu. High-dimensional covariance estimation by minimizing l1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935–980, 2011.

Somayeh Sojoudi. Equivalence of graphical lasso and thresholding for sparse graphs. *Journal of Machine Learning Research*, 17(115):1–21, 2016.

Somayeh Sojoudi and John Doyle. Study of the brain functional network using synthetic data. *52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 350–357, 2014.

Shiliang Sun, Rongqing Huang, and Ya Gao. Network-scale traffic modeling and forecasting with graphical lasso and neural networks. *Journal of Transportation Engineering*, 138(11): 1358–1367, 2012.

Robert Tibshirani, Jacob Bien, Jerome Friedman, Trevor Hastie, Noah Simon, Jonathan Taylor, and Ryan J Tibshirani. Strong rules for discarding predictors in lasso-type problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(2): 245–266, 2012.

Lieven Vandenberghe and Martin S. Andersen. Chordal graphs and semidefinite optimization. *Foundations and Trends® in Optimization*, 1(4):241–433, 2015.

Petra E. Vértes, Aaron F. Alexander-Bloch, Nitin Gogtay, Jay N. Giedd, Judith L. Rapoport, and Edward T. Bullmore. Simple models of human brain functional networks. *Proceedings of the National Academy of Sciences*, 109(15):5868–5873, 2012.

Daniela M. Witten, Jerome H. Friedman, and Noah Simon. New insights and faster computations for the graphical lasso. *Journal of Computational and Graphical Statistics*, 20 (4):892–900, 2011.

John Wright, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S. Huang, and Shuicheng Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031–1044, 2010.

Xindong Wu, Xingquan Zhu, Gong Qing Wu, and Wei Ding. Data mining with big data. *IEEE Transactions on Knowledge and Data Engineering*, 26(1):97–107, 2014.

Eunho Yang, Aurélie C Lozano, and Pradeep K Ravikumar. Elementary estimators for graphical models. In *Advances in neural information processing systems*, pages 2159–2167, 2014.

Hongbin Yin, S. C Wong, Jianmin Xu, and C. K. Wong. Urban traffic flow prediction using a fuzzy-neural approach. *Transportation Research Part C: Emerging Technologies*, 10(2): 85–98, 2017.

Ming Yuan and Yi Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

Sangwoon Yun and Kim-Chuan Toh. A coordinate gradient descent method for l_1-regularized convex minimization. *Computational Optimization and Applications*, 48(2): 273–307, 2011.

Richard Y Zhang and Javad Lavaei. Modified interior-point method for large-and-sparse low-rank semidefinite programs. In *Decision and Control (CDC), 2017 IEEE 56th Annual Conference on*, pages 5640–5647. IEEE, 2017.

Journal of Machine Learning Research 20 (2019) 1-23 Submitted 9/17; Revised 12/18; Published 1/19

An Approach to One-Bit Compressed Sensing Based on Probably Approximately Correct Learning Theory

Mehmet Eren Ahsen mehmeteren.ahsen@mssm.edu
Icahn School of Medicine at Mount Sinai
1468 Madison Avenue
New York, NY 10029

Mathukumalli Vidyasagar m.vidyasagar@utdallas.edu,m.vidyasagar@iith.ac.in
Department of Systems Engineering
University of Texas at Dallas
Richardson, TX 75080, USA, and
Department of Electrical Engineering
Indian Institute of Technology Hyderabad
Kandi, Telangana, India 502285

Editor: John Shawe-Taylor

Abstract

In this paper, the problem of one-bit compressed sensing (OBCS) is formulated as a problem in probably approximately correct (PAC) learning. It is shown that the Vapnik-Chervonenkis (VC-) dimension of the set of half-spaces in $\mathbb{R}^n$ generated by k-sparse vectors is bounded below by $k(\lfloor \lg(n/k) \rfloor + 1)$ and above by $\lfloor 2k \lg(en) \rfloor$. By coupling this estimate with well-established results in PAC learning theory, we show that a consistent algorithm can recover a k-sparse vector with $O(k \lg n)$ measurements, given only the signs of the measurement vector. This result holds for *all* probability measures on $\mathbb{R}^n$. The theory is also applicable to the case of noisy labels, where the signs of the measurements are flipped with some unknown probability.

1. Introduction

The field of "compressed sensing" has become very popular in recent years, with an explosion in the number of papers. Stated briefly, the core problem in compressed sensing is to recover a high-dimensional sparse (or nearly sparse) vector x from a small number of measurements of x. In the traditional problem formulation, the measurements are *linear*, consisting of m real numbers $y_i = \langle a_i, x \rangle, i = 1, \ldots, m$, where the measurement vectors $a_i \in \mathbb{R}^n$ are chosen by the learner. More recently, attention has focused on so-called one-bit compressed sensing, referred to hereafter as OBCS in the interest of brevity. In OBCS the measurements consist, not of the inner products $\langle a_i, x \rangle$, but rather *just the signs of these inner products*, i.e., a "one-bit" representation of these measurements. In much of the OBCS literature, the vectors a_i are chosen at random from some specified probability distribution, often Gaussian. Because the sign of $\langle a_i, x \rangle$ is unchanged if x is replaced by any positive multiple of x, it is obvious that under this model, one can at best aspire to recover the unknown vector x only to within a positive multiple, or equivalently, to recover the normalized vector $x/\|x\|_2$. This limitation can be overcome by choosing the measurements to consist of the signs of inner products $\langle a_i, x \rangle + b_i$, where again a_i, b_i are selected at random.

The current status of OBCS is that while several algorithms have been proposed, theoretical analysis is available for only a few algorithms. Moreover, in cases where theoretical analysis is available, the sample complexity is extremely high. In the present paper, we interpret OBCS as a problem in probably approximately correct (PAC) learning theory, which is a well-established branch of statistical learning theory. By doing so, we are able to draw from the wealth of results that are already available, and thereby address some of the currently outstanding issues. In PAC learning theory, a central role is played by the so-called Vapnik-Chervonenkis (VC) dimension of the collection of concepts to be learned. The principal result of the present paper is that the VC-dimension of the set of half-planes in $\mathbb{R}^n$ generated by k-sparse vectors is bounded below by $k \lg(n/k)$ and above by $2k \lg(en)$, plus some roundoff terms. Using this bound, we are able to establish the following results for the case where $x \in \mathbb{R}^n$ has no more than k nonzero components:

- In principle, OBCS is possible whenever the measurement vector (a_i, b_i) is drawn at random from *any arbitrary* probability distribution. Moreover, if a consistent algorithm[1] can be devised, then the number of measurements is $O(k \lg n)$.
- There is also a *lower bound* on the OBCS problem. Specifically, there exists a probability distribution on $\mathbb{R}^n$ such that, if (a_i, b_i) is drawn at random from this distribution, then the number of measurements required to learn each k-sparse n-dimensional vector is bounded *below* by $\Omega(k \ln(n/k))$.
- It is shown that OBCS is possible under random flipping of the signs of the measurements. This finding builds on earlier results on PAC learning with noisy labels.
- If the samples a_i are not independent, but are β-mixing, learning is still possible, and explicit estimates are available for the rate of learning.

The paper is organized as follows: In Section 2, a brief review is given of some recent papers in OBCS. In Section 3, some parts of PAC learning theory that are relevant to OBCS are reviewed. In particular, it is shown how OBCS can be formulated as a problem in PAC learning, so that OBCS can be addressed by finding upper bounds on the VC-dimension of half-spaces generated by k-sparse vectors. In Section 4, both upper and lower bounds are derived for the VC-dimension of half-spaces generated by k-sparse vectors. In Section 5, the case where the sign measurements are flipped at random is examined. Finally, Section 6 contains a discussion of some issues that merit further investigation.

2. Brief Review of One-Bit Compressed Sensing

By now there is a substantial literature regarding the traditional compressed sensing formulation, out of which only a few references are cited here in the interests of brevity. Book-length treatments of compressed sensing can be found in (Elad, 2010; Foucart and Rauhut, 2013; Rish and Grabarnik, 2015; Hastie et al., 2015). Amongst these, (Foucart

1. This term is standard in statistical learning theory and is defined later.

©2019 Mehmet Eren Ahsen and Mathukumalli Vidyasagar.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-504.html.

and Rauhut, 2013) contains a thorough discussion of virtually all aspects of compressed sensing theory. A recent volume (Eldar and Kutyniok, 2012) is a compendium of articles on a variety of topics. The first paper in this volume (Davenport et al., 2012) is a survey of the basic results in compressed sensing. Another paper (Negabhan et al., 2012) provides a very general framework for sparse regression that can be used, among other things, to analyze compressed sensing algorithms. Each of these papers contains an extensive bibliography.

Throughout this paper, n denotes some fixed and large integer. For $x \in \mathbb{R}^n$, let $\text{supp}(x)$ denote the support of a vector, and let Σ_k denote the set of k-sparse vectors in $\mathbb{R}^n$; that is,

$$\text{supp}(x) := \{i : x_i \neq 0\}, \Sigma_k := \{x \in \mathbb{R}^n : |\text{supp}(x)| \leq k\}.$$

Suppose $x \in \mathbb{R}^n$ is k-sparse, that is, $x \in \Sigma_k$, where both n and k are known integers with $k \ll n$. The basic problem in compressed sensing is to design an $m \times n$ matrix A where $m \ll n$, together with a decoder map $\Delta : \mathbb{R}^m \to \mathbb{R}^n$ such that $\Delta(Ax) = x$ for all $x \in \Sigma_k$, that is, x can be recovered exactly from the m-dimensional vector of linear measurements $y = Ax$. Variations of the problem include the case where x is not exactly sparse, and/or $y = Ax + \eta$ where η is a measurement noise. Given a vector $x \in \mathbb{R}^n$, an integer $k < n$, define

$$\sigma_k(x, \|\cdot\|_1) := \min_{z \in \Sigma_k} \|x - z\|_1$$

to be the k-**sparsity index** of the vector x. It is clear that $x \in \Sigma_k$ if and only if $\sigma_k(x, \|\cdot\|_1) = 0$. By far the most popular method for recovering a sparse vector is ℓ_1-norm minimization. If $y = Ax + \eta$, the approach is to define the "decoder" map $\Delta : \mathbb{R}^m \to \mathbb{R}^n$ by

$$\Delta(y) = \hat{x} := \underset{z}{\text{argmin}} \|z\|_1 \text{ s.t. } \|y - Az\|_2 \leq \epsilon, \tag{1}$$

where ϵ is a known upper bound on $\|\eta\|_2$. In a series of papers by Candès, Tao, Donoho, and others, it is demonstrated that if the matrix A is chosen so as to satisfy the so-called Restricted Isometry Property (RIP), then the decoder Δ defined in (1) produces a good approximation to x in the following sense: There exist constants C, D that depend on the measurement matrix A but not on the unknown vector x, such that

$$\|\hat{x} - x\|_1 \leq C\sigma_k(x, \|\cdot\|_1) + D\epsilon.$$

In particular, ℓ_1-norm minimization recovers x exactly if $x \in \Sigma_k$ and $\eta = 0$. See for example (Candès and Tao, 2005; Candès et al., 2006; Donoho, 2006a,b), as well as the survey paper (Davenport et al., 2012) and the comprehensive book (Foucart and Rauhut, 2013). Further, it is shown in (Candès and Tao, 2005) that if the elements of A are samples of independent and identically distributed (i.i.d.) normal random variables (denoted by $a_{ij} \sim \mathcal{N}(0, 1)$), then the resulting normalized matrix $(1/\sqrt{m})A$ satisfies the RIP with probability that can be made arbitrarily close to one by increasing the number of measurements.

The remainder of the section is devoted to a discussion of the one-bit compressed sensing (OBCS) problem. One bit compressed sensing is introduced in (Boufounos and Baraniuk, 2008). In that paper, it is assumed that the measurement y_i equals the bipolar quantity $y_i = \text{sign}(\langle a_i, x\rangle)$, as opposed to the real number $\langle a_i, x\rangle$. Because the measurements remain invariant if x is replaced by any positive multiple of x, there is no loss of generality in

assuming that $\|x\|_2 = 1$. A greedy algorithm called "renormalized fixed point iteration" is introduced, as follows:

$$\hat{x} := \underset{z}{\text{argmin}} \|z\|_1 + \lambda \sum_{i=1}^{m} f(|(YA)_i|) \text{ s.t. } \|z\|_2 = 1,$$

where $y_1, \ldots, y_m$ are some constants, $Y = \text{Diag}(y_1, \ldots, y_m)$, and the regularizing function $f(\cdot)$ is defined by

$$f(\alpha) := \begin{cases} \alpha^2/2 & \text{if } \alpha < 0, \\ 0 & \text{if } \alpha \geq 0. \end{cases}$$

The optimization problem is non-convex due to the constraint $\|z\|_2 = 1$. Only simulations are provided, but no theoretical results.

In (Gupta et al., 2010), the focus is on recovering the support set of the unknown vector x from noise-corrupted measurements of the form $\text{sign}(\langle a_i, x\rangle + \eta_i)$, where the noise vector $\boldsymbol{\eta}$ consists of pairwise independent Gaussian signals. A non-adaptive algorithm is presented that makes use of Hoeffding's inequality applied to the expected value of the covariance of the signs of two Gaussian random variables. An adaptive algorithm is also presented. In (Boufounos, 2009), a new greedy algorithm is presented called "matched signed pursuit." The optimization problem is not convex; as a result there are no theoretical resuts. The algorithm is similar to the CoSaMP algorithm for the conventional compressed sensing problem (Needell and Tropp, 2009).

In (Jacques et al., 2013), the authors introduce a constant

$$\epsilon_{\text{opt}} := \frac{k}{2em + 2k^{3/2}},$$

where e denotes the base of the natural logarithm, and show that it is the minimum achievable error no matter what reconstruction method is used. Next, they choose any $\epsilon \geq \epsilon_{\text{opt}}$ and show the following result: Let $A \sim \mathcal{N}^{m \times n}(0, 1)$ consist of mn pairwise independent normal random variables, and let $y_i = \text{sign}(\langle a_i, x\rangle)$. Fix $\delta \in (0, 1)$. If the number of measurements m satisfies

$$m \geq \frac{2}{\epsilon}\left(2k \ln n + 4k \ln \frac{17}{\epsilon} + \ln \frac{1}{\delta}\right).$$

then for every pair $x, s \in \Sigma_k$,

$$\text{sign}(Ax) = \text{sign}(As) \implies \|x - s\|_2 \leq \epsilon,$$

with probability $\geq 1 - \delta$. In words, this result means that if we can find a k-sparse vector s that is consistent with the observation vector y, then s is close to x. In fact, s can be made as close to x as desired by increasing the number of measurements m. Unfortunately, this result is not practical because finding such a vector s is equivalent to finding a minimal ℓ_0-norm solution consistent with the observations, which is known to be an NP-hard problem (Natarajan, 1995).

In (Plan and Vershynin, 2013a), the authors focus on vectors $x \in \mathbb{R}^n$ that satisfy an inequality of the form $\|x\|_1/\|x\|_2 \leq s$. Note that if x is s-sparse, then it satisfies the above

inequality, though of course the converse is not true. Thus they use the ratio $\|x\|_1/\|x\|_2$ as a proxy for $\|x\|_0$. They choose measurement vectors $a_i \in \mathbb{R}^n$ according to the Gaussian distribution, or more generally, any radially invariant distribution; this means that, under the chosen probability distribution on the vector $a \in \mathbb{R}^n$, the normalized vector $a/\|a\|_2$ is uniformly distributed on the sphere $S^{n-1} \subseteq \mathbb{R}^n$. With these randomly generated measurement vectors, the measured quantities are $y_i = \text{sign}(\langle a_i, x\rangle)$. The authors propose to estimate x via

$$\hat{x} := \underset{z}{\text{argmin}} \|z\|_1 \text{ s.t. } \text{sign}(\langle a_i, z\rangle) = y_i \ \forall i, \sum_{i=1}^{m} |\langle a_i, z\rangle| = m, \tag{2}$$

where m is the number of measurements. They show that if

$$\epsilon > C\left(\frac{s}{m}\ln(2n/s)\ln(2n/m + 2m/n)\right)^{1/5}$$

for some universal constant C, then with probability $\geq 1 - \exp(-c\epsilon m)$ where c is another universal constant, it is true that

$$\left\| \frac{x}{\|x\|_2} - \frac{\hat{x}}{\|\hat{x}\|_2} \right\|_2 \leq \epsilon \tag{3}$$

for all $x \in \mathbb{R}^n$ such that $\|x\|_1/\|x\|_2 \leq \sqrt{s}$. Although this is the first proposed convex algorithm to recover x, the number of measurement m is $O(\delta^{-5})$. From (3) we see that if we are able to carry out the ℓ_0-norm minimization, then m is $O(\delta^{-1})$. It is still an open question whether or not a practical algorithm can achieve this optimal dependence on δ in (3). In (Ai et al., 2014) the theory is extended to non-Gaussian noise signals that are sub-Gaussian.

In (Plan and Vershynin, 2013b), it is assumed that the measurements $y_i \in \{0, 1\}$ are drawn at random with the property that the expected value $E(y_i)$ satisfies

$$E(y_i) = \theta(\langle a_i, x\rangle),$$

where $\theta : \mathbb{R} \to [-1, 1]$ is an unknown function. If $\theta(\alpha) = \tanh(\alpha/2)$, then the problem is one of logistic regression, whereas if $\theta(\alpha) = \text{sign}(\alpha)$, then the problem becomes OBCS. A probabilistic approach is proposed in (Plan and Vershynin, 2013b), which has the advantage that the resulting optimization problem is convex. However, the disadvantage is that the number of measurements m is $O(\delta^{-6})$ where δ is the probability that the algorithm may fail. The large negative exponent of δ makes the algorithm somewhat impractical.

In all of the papers discussed until now, the measurement vector y_i equals $\text{sign}(\langle a_i, x\rangle)$ for suitably generated random vectors a_i. As mentioned above, with such a set of measurements one can at best aspire to recover only the normalized unknown vector $x/\|x\|_2$. In (Knudson et al., 2016), it is proposed to overcome this limitation by changing the *linear* measurements to *affine* measurements. Specifically, the measurements in (Knudson et al., 2016) are of the form $y_i = \text{sign}(\langle a_i, x\rangle + b_i)$, where $a_{ij} \sim \mathcal{N}(0, 1)$,[2] and $b_i \sim \mathcal{N}(0, \tau^2)$ where τ is some

2. Recall that each a_i is an n-vector.

specified constant. If a prior upper bound R for $\|x\|_2$ is available, then it is possible to choose $\tau = R$. Then the optimization problem in (2) is modified to[3]

$$(\hat{x}, \hat{v}) := \underset{z,v}{\text{argmin}}[\|z\|_1 + \tau|v|] \text{ s.t. } \text{sign}(\langle a_i, z\rangle + b_i v) = y_i \ \forall i, \sum_{i=1}^{m} |\langle a_i, z\rangle + b_i v| = m. \tag{4}$$

It is evident that the above formulation is similar to the formulation in (Plan and Vershynin, 2013a) applied to the augmented vector $(x, v) \in \mathbb{R}^{n+1}$. The following result is shown in (Knudson et al., 2016, Theorem 4): Fix τ, R, α such that $\alpha < \min\{1, \tau/2\}$. If

$$m \geq C\left(\frac{\sqrt{R^2 + \tau^2}}{\alpha}\right)^5 \log^2\left(\frac{2n}{x}\right),$$

then for all vectors $x \in \mathbb{R}^n$ with $\|x\|_1/\|s\|_2 \leq \sqrt{s}$, the solution $(\hat{x}, \hat{v})$ to the optimization problem in (4) satisfies

$$\|(\hat{x}/\hat{v}) - x\|_2 \leq \frac{4\sqrt{R^2 + \tau^2}}{\tau}\alpha,$$

with a probability exceeding

$$1 - C\exp\left(-\frac{c\alpha m}{\sqrt{R^2 + \tau^2}}\right),$$

where C and c are universal constants. If R is a known prior upper bound for $\|x\|_2$, then one can choose $\tau = R$ in the above, in which the bound simplifies to

$$\|(\hat{x}/\hat{v}) - x\|_2 \leq 4\sqrt{2}\alpha.$$

3. Preliminaries

In this section we present some preliminary results, while the main results are presented in the next section. As shown below, the one-bit compressed sensing (OBCS) problem can be naturally formulated as a problem in probably approximately correct (PAC) learning. In fact, several of the approaches proposed thus far for solving the OBCS problem are similar to existing methods in PAC learning, but do not take full advantage of the power and generality of PAC learning theory. Some of the things that "come for free" in PAC learning theory are: explicit estimates for the number of measurements m, ready extension to the case where successive measurement vectors a_i are not independent but form a β-mixing process, and ready extension to the case of noisy measurements. The negative is that the PAC learning approach does not always address the *efficiently computability* of the algorithms. In (Valiant, 1984) which launched PAC learning theory, efficient computability of the algorithm is a central consideration. However, subsequent literature does not always pay full attention to this aspect.

3. Note that v here equals u/τ in (Knudson et al., 2016, Eq. (6)). Also, since δ is used to denote the confidence of a PAC learning algorithm later in the paper, we use α instead of δ as in the cited equation.

3.1. Brief Introduction to the PAC Learning Problem

In this subsection, we give a brief introduction to PAC learning theory. By now the fundamentals of PAC learning theory are well-developed, and several book-length treatments are available, including (Vapnik, 1998; Anthony and Bartlett, 1999; Vidyasagar, 1997, 2003). The theory encompasses a wide variety of learning situations. However, OBCS is aligned closely with the most basic version of PAC learning, known as concept learning, which is formally described next.

The concept learnig problem is formulated over a metric space X (usually taken as $\mathbb{R}^n$ for some integer n), together with the associated Borel σ-algebra $\mathcal{S}$; a collection of sets $\mathcal{C} \subseteq \mathcal{S}$, known as the **concept class**; and a family $\mathcal{P}$ of probability measures on $(X, \mathcal{S})$. If $\mathcal{P} = \mathcal{P}^*$, the set of *all* probability measures on $(X, \mathcal{S})$, the problem is known as one of "distribution-free learning."

Learning takes place as follows: A fixed but unknown set $T \in \mathcal{C}$, known as the "target concept," is chosen, along with an unknown probability measure $P \in \mathcal{P}$. Then random samples $\{c_1, c_2, \ldots\}$ are generated independently in accordance with the chosen distribution P. This is the basic version of PAC learning studied in (Vapnik, 1998; Anthony and Bartlett, 1999; Vidyasagar, 1997). The case where the sample sequence can exhibit dependence, for example if they come from a Markov process with the stationary distribution P, is studied in (Vidyasagar, 2003). Using the sample c_i, an "oracle" generates a "label" $y_i \in \{0, 1\}$. In the case of noise-free measurements, $y_i = I_T(c_i)$, where T is the fixed but unknown target concept, and $I_T(\cdot)$ denotes the indicator function of the set T. In the case of noisy measurements, the label y_i equals $I_T(c_i)$ flipped with an unknown probability α. This situation is studied in Section 5. After m such samples are drawn and labelled, the available information $\{(c_i, y_i)\}_{i=1}^m \in (X \times \{0, 1\})^m$ is passed through an "algorithm" to generate a "hypothesis," or an approximation to the unknown target concept T. For present purposes, an "algorithm" is any indexed collection of maps $\{A_m\}_{m \geq 1}$ where

$$A_m : (X \times \{0, 1\})^m \to \mathcal{C}.$$

In other words, an algorithm is any systematic procedure for taking a finite sequence of labelled samples, and returning an element of the concept class $\mathcal{C}$. The concept

$$G_m(T; \mathbf{c}) := A_m\left(\{(c_i, y_i)\}_{i=1}^m\right)$$

is called the "hypothesis" generated by the first m samples when the sample sequence is $\mathbf{c} = (c_1, \ldots, c_m)$, and the label sequence is $\mathbf{y} = (y_1, \ldots, y_m)$. In the interests of reducing clutter, we will use G_m in the place of $G_m(T; \mathbf{c})$ unless the full form is needed for clarity. Note that A_m is a deterministic map, but the hypothesis G_m is random because it depends on the random learning sequence $\{c_i\}$.

To measure how well the hypothesis G_m approximates the unknown target concept T, we use the **generalization error** defined by

$$J(T, G_m) = E[|I_T(x) - I_{G_m}(x)|, P]. \tag{5}$$

Thus $J(T, G_m)$ is the expected value of the difference between the indicator function $I_T(\cdot)$ and the label generated by the oracle with the input $I_{G_m}(\cdot)$. Note that both I_T and I_{G_m}

assume values in $\{0, 1\}$. Hence $J(T, G_m)$ is also the probability that, when a random test sample $x \in X$ is generated in accordance with the probability distribution P, the sample is *misclassified* by the hypothesis G_m, in the sense that $I_T(x) \neq I_{G_m}(x)$. In particular, if the hypothesis G_m differs from the target concept T by a set of measure zero, then the generalization error would be zero.

The key quantity in PAC learning theory is the **learning rate**, defined by

$$r(m, \epsilon) := \sup_{P \in \mathcal{P}} \sup_{T \in \mathcal{C}} P^m\{\mathbf{c} \in X^m : J(T, G_m) > \epsilon\}. \tag{6}$$

Therefore $r(m, \epsilon)$ is the worst-case measure, over all probability distributions in $\mathcal{P}$ and all target concepts in $\mathcal{C}$, of the set of "bad" samples $\mathbf{c} = (c_1, \ldots, c_m)$ for which the corresponding hypothesis G_m has a generalization error larger than a prespecifie threshold ϵ. Thus, after m samples are generated together with their labels, and the hypothesis G_m is generated using the algorithm, it can be asserted with confidence $1 - r(m, \epsilon)$ that G_m will correctly classify the next randomly generated test sample with probability of at least $1 - \epsilon$.

Definition 1 *An algorithm $\{A_m\}$ is said to be* **probably approximately correct (PAC)** *if $r(m, \epsilon) \to 0$ as $m \to \infty$, for every fixed $\epsilon > 0$. The concept class $\mathcal{C}$ is said to be* **PAC learnable** *under the family of probability measures $\mathcal{P}$ if there exists a PAC algorithm.*

The objective of statistical learning theory is to determine conditions under which there exists a PAC algorithm for a given concept class, and if so, to find upper bounds for the learning rate $r(m, \epsilon)$.

3.2. OBCS as a Problem in PAC Learning

In order to embed the problem of one-bit compressed sensing into the framework of concept learning, we proceed as follows. We begin with the case where the measurements are of the form $\text{sign}(\langle a_i, x \rangle)$ where the a_i are chosen at random according to some arbitrary probability distribution, *which need not be the Gaussian.* Observe now that the closed half-space

$$H(x) := \{z \in \mathbb{R}^n : \langle z, x \rangle \geq 0\}$$

determines the vector x uniquely to within a positive scalar multiple, because x is the normal to the half-space. Conversely, the vector x uniquely determines the corresponding half-space $H(x)$, which remains invariant if x is replaced by a positive multiple of x. Thus the OBCS problem can be posed as that of determining the half-space $H(x)$ given the measurements $\text{sign}(\langle a_i, x \rangle), i = 1, \ldots, m$ where the a_i are selected at random in accordance with some probability measure P. Moreover, the one-bit measurement $\text{sign}(\langle a_i, x \rangle)$ equals $2I_{H(x)}(a_i) - 1$, where $I_{H(x)}(\cdot)$ denotes the indicator function of the half-space $H(x)$. Therefore, to within a simple affine transformation, the OBCS problem becomes that of determining an unknown half-space $H(x)$ from labelled samples $(a_i, I_{H(x)}(a_i)), i = 1, \ldots, m$, where the samples a_i are generated at random according to some prespecified probability measure. This is a PAC learning problem where the underlying space X is $\mathbb{R}^n$, with $\mathcal{S}$ being the associated Borel σ-algebra. The concept class $\mathcal{C}$ is the collection of all half-spaces $\{H_k^n(x)\}$ where

$$H_k^n(x) = \{a \in \mathbb{R}^n : \langle a, x \rangle \geq 0\} \tag{7}$$

as x varies over Σ_k, the set of k-sparse vectors in $\mathbb{R}^n$. The family of probability measures $\mathcal{P}$ can either be a singleton $\{P\}$ where P is specified *a priori*, or $\mathcal{P}^*$, the family of *all* probability measures on $\mathbb{R}^n$, or anything in-between.

When measurements are of the type $\text{sign}(\langle a_i, x\rangle)$, it is inherently impossible to determine the unknown vector x, except to within a positive scalar multiple. This is addressed by changing the measurements to be of the form $\text{sign}(\langle a_i, x\rangle + b_i)$, as suggested in (Knudson et al., 2016). Some slight modifications are required to address this modified formulation. In this case theunderlying space X is $\mathbb{R}^{n+1}$, and $\mathcal{S}$ is the associated σ-algebra. The concept class $\mathcal{C}$ is the collection of all half-spaces $\{H_k^{n+1}\}$ where

$$H_k^{n+1}(x) = \{(a,b) \in \mathbb{R}^{n+1} : \langle a, x\rangle + b \geq 0\} \tag{8}$$

as x varies over Σ_k. Finally, the family of probability measures $\mathcal{P}$ can either be a singleton $\{P\}$ where P is specified *a priori*, or $\mathcal{P}^*$, the family of *all* probability measures on $\mathbb{R}^n$, or anything in-between.

For a given $x \in \Sigma_k$, if $a \in \mathbb{R}^n$ belongs to the half-space $H_k^n(x)$, then the vector $(a,0) \in \mathbb{R}^{n+1}$ belongs to $H_k^{n+1}(x)$. However, the half-space $H_k^{n+1}(x)$ can also contain vectors of the form (a,b) with $b \neq 0$.

3.3. Interpretation of the Generalization Error

Suppose P is a probability measure on $(X, \mathcal{S})$, and define the quantity

$$d_P(A,B) := P(A \Delta B),\ \forall A, B \in \mathcal{S}.$$

Then d_P is a pseudometric on $\mathcal{S}$, in that d_P satisfies all the axioms of a metric, except that $d_P(A\Delta B) = 0$ does not necessarily imply that $A = B$, only that $A\Delta B$ has measure zero under P. In the traditional PAC learning problem formulation, the quantity of interest is the generalization error defined in (5). It is easy to see that an alternate expression for the generalization error is

$$J(T, G_m) = P(A\Delta B) = d_P(A,B).$$

Therefore the traditional PAC learning formulation does not distinguish between two concept classes that differ by a set of measure zero.

The above discussion explains the limitations of one-bit compressed sensing as described in (Plan and Vershynin, 2013b). In their case, they choose two vectors in $\mathbb{R}^2$, namely $x_1 = [\ 1 \quad 0\ ]$ and $x_1 = [\ 1 \quad 0.5\ ]$.[4] Their choice for P is the Bernoulli distribution on $\mathbb{R}^2$, which is purely atomic and assigns a weight of 0.25 to the four points $(1,1), (1,-1), (-1,1), (-1,-1)$. Now let us plot the half-planes H_{x_1}, H_{x_2} and their symmetric difference, which is the shaded region shown in Figure 1. Because none of the four points $(1,1), (1,-1), (-1,1), (-1,-1)$ (shown as red circles) belongs to the symmetric difference, x_1 and x_2 are indistinguishable in OBCS under this probability measure. In (Plan and Vershynin, 2013a), the authors conclude that OBCS cannot always recover an unknown vector x, depending on what P is. Indeed, x_1 and x_2 would be indistinguishable under *any* probability measure that assigns a value of zero to $H_{x_1} \Delta H_{x_2}$. Therefore, one must be careful to draw the right conclusion: When subsequent theorems in this paper show that OBCS is possible under *all* probability

4. They append a whole lot of zero components which are neglected here.

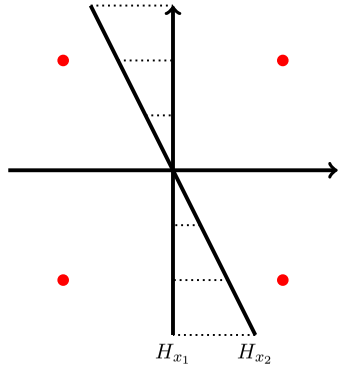


Figure 1: The half-planes H_{x_1}, H_{x_2}, and their symmetric difference.

measures on $\mathbb{R}^n$, *including* all purely atomic probability measures, what this means is that if $x \in \Sigma_k$, then OBCS will return a vector $\hat{x}$ such that $P(H_x \Delta H_{\hat{x}}) = 0$ whatever be P, and *not* that $x = \hat{x}$.

Now we examine the relationship of the generalization error $J(\hat{x}, x)$ to a couple of other quantities that are widely used in OBCS as error measures. First, define

$$\rho(\hat{x}, x) := E[|\text{sign}(\langle a, \hat{x}\rangle) - \text{sign}(\langle a, x\rangle)|, P], \tag{9}$$

where x is the true vector, $\hat{x}$ is its estimate. Note that $|\text{sign}(\langle a, \hat{x}\rangle) - \text{sign}(\langle a, x\rangle)|$ equals 0 or 2. Therefore we can also write

$$\rho(\hat{x}, x) = 2J(H_{\hat{x}}, H_x).$$

Next, we examine the relationship of $\rho(\hat{x}, x)$ to $\|\hat{x} - x\|_2$. Without loss of generality it can be assumed that both x and $\hat{x}$ have unit Euclidean norm. This can be achieved using some results from (Goemans and Williamson, 1995). Define

$$\alpha := \min_{\theta \in [0, 2\pi]} \frac{2}{\pi} \frac{\theta}{1 - \cos\theta} > 0.87856. \tag{10}$$

Then we have the following results.

Lemma 2 *Let P be any radially invariant probability measure on $\mathbb{R}^n$, and suppose $a \in \mathbb{R}^n$ is drawn at random according to P. Suppose $\|x\|_2 = \|\hat{x}\|_2 = 1$, and let $J(\hat{x}, x)$ denote the generalization error defined in* (5)*. Then*

$$\|\hat{x} - x\|_2^2 \leq \frac{4}{\alpha} J(\hat{x}, x). \tag{11}$$

Proof We make use of a couple of results from (Goemans and Williamson, 1995). First, (Goemans and Williamson, 1995, Lemma 3.2) states that

$$\Pr\{\text{sign}(\langle a, \hat{x}\rangle) \neq \text{sign}(\langle a, x\rangle)\} = \frac{1}{\pi}\arccos(\hat{x}^\top x),$$

Note that

$$J(\hat{x}, x) = \Pr\{\text{sign}(\langle a, \hat{x}\rangle) \neq \text{sign}(\langle a, x\rangle)\}.$$

Therefore the above is equivalent to

$$J(\hat{x}, x) = \frac{1}{\pi}\arccos(\hat{x}^\top x).$$

Second, (Goemans and Williamson, 1995, Lemma 3.4) states that

$$\frac{1}{\pi}\arccos(\hat{x}^\top x) \geq \frac{\alpha}{2}(1 - \hat{x}^\top x),$$

or equivalently

$$1 - \hat{x}^\top x \leq \frac{2}{\alpha}\arccos(\hat{x}^\top x).$$

Now note that, when both $\hat{x}$ and x are unit vectors, we have

$$\|\hat{x} - x\|_2^2 = \|\hat{x}\|_2^2 + \|x\|_2^2 - 2\hat{x}^\top x = 2(1 - \hat{x}^\top x).$$

Therefore

$$\begin{aligned} \|\hat{x} - x\|_2^2 &= 2(1 - \hat{x}^\top x) \\ &\leq \frac{4}{\alpha}\arccos(\hat{x}^\top x) = \frac{4}{\alpha}J(\hat{x}, x). \end{aligned}$$

■

3.4. PAC Learning via the Vapnik-Chervonenkis (VC) Dimension

One of the most useful concepts in PAC learning theory is defined next.

Definition 3 *A set $S \subseteq X$ of finite cardinality is said to be* **shattered** *by the concept class $\mathcal{C}$ if, for every subset $B \subseteq S$, there exists a concept $A \in \mathcal{C}$ such that $S \cap A = B$. The* **Vapnik-Chervonenkis- or VC-dimension** *of the concept class $\mathcal{C}$ is the largest integer d such that there exists a set S of cardinality d that is shattered by $\mathcal{C}$.*

Therefore a concept class $\mathcal{C}$ has VC-dimension d if two statements hold: (i) There exists a set of cardinality d that is shattered by $\mathcal{C}$, and *no set* of cardinality larger than d is shattered by $\mathcal{C}$. If there exist sets of arbitarily large cardinality that are shattered by $\mathcal{C}$, then its VC-dimension is defined to be infinite.

If a concept class has finite VC-dimension, then it is PAC learnable under *every probability distribution* on X. An algorithm is said to be **consistent** if it always produces a

hypothesis that classifies all the training samples correctly. In other words, an algorithm is consistent if the hypothesis G_m produced by applying the algorithm to the sequence $\{(c_i, I_T(c_i))\}_{i\geq 1}$ has the property that $I_T(c_i) = I_{G_m}(c_i)$ for all i and m. Note that a consistent algorithm always exists if the axiom of choice is assumed. However, in some situations, it is NP-hard or NP-complete to find a consistent algorithm.

With these notions in place, we have the following very fundamental result.

Theorem 4 *((Blumer et al., 1989); see also (Vidyasagar, 2003, Theorem 7.6)) Suppose a concept class $\mathcal{C}$ has finite VC-dimension. Then $\mathcal{C}$ is PAC learnable for every probability measure on X. Suppose that d is an upper bound for VC-dim($\mathcal{C}$), and let $\{A_m\}$ be any consistent algorithm. Moreover, the learning rate is bounded by*

$$r(m, \epsilon) \leq 2\left(\frac{2em}{d}\right)^d 2^{-m\epsilon/2},$$

where e denotes the base of the natural logarithm. Therefore $r(m, \epsilon) \leq \delta$ if

$$m \geq \max\left\{\frac{8d}{\epsilon}\lg\frac{8e}{\epsilon}, \frac{4}{\epsilon}\lg\frac{2}{\delta}\right\} \tag{12}$$

samples are chosen.

Note that the number of samples required to achieve an accuracy of ϵ with confidence $1 - \delta$ is $O((1/\epsilon)\ln(1/\delta))$. However, the main challenge in applying this result is in finding a consistent algorithm.

Theorem 4 shows that the finiteness of the VC-dimension of a concept class is a *sufficient* condition for PAC learnability. The next result shows that the condition is also necessary.

Theorem 5 *((Blumer et al., 1989; Ehrenfeucht et al., 1989); see also (Vidyasagar, 2003, Theorem 7.7)) Suppose a concept class $\mathcal{C}$ has VC-dimension $d \geq 2$. Then there exist probability measures on X such that any algorithm requires at least*

$$m \geq \left\{\frac{d-1}{32\epsilon}, \frac{1-\epsilon}{\epsilon}\ln\frac{1}{\delta}\right\} \tag{13}$$

samples, in order to learn to accuracy ϵ and confidence δ.

4. Estimates of the VC-Dimension

This section contains the main results of the paper, namely, to obtain explicit estimates of the VC-dimension of half-spaces generated by k-sparse vectors.

Theorem 6 *Let $\mathcal{H}_k^n$ denote the set of half-spaces $H_k^n(x)$ in $\mathbb{R}^n$ generated by k-sparse vectors, as defined in* (7)*. Then*

$$k(\lfloor\lg(n/k)\rfloor + 1) \leq \textit{VC-dim}(\mathcal{H}_k^n) \leq \lfloor 2k\lg(ne)\rfloor. \tag{14}$$

Remark: In (Neylon, 2006, Theorem 20), it is shown that if $k < 0.45\sqrt{n}$, then

$$\text{VC-dim}(\mathcal{H}_k^n) < 2k \lg n.$$

The right side of (14) can be rewritten as

$$\text{VC-dim}(\mathcal{H}_k^n) \leq 2k(\lg n + \lg e) \approx 2k(\lg n + 1.4).$$

Clearly, as n becomes larger, the additional term becomes insignificant. Moreover, there is no restriction on the magnitude of k in Theorem 6, as there is in (Neylon, 2006, Theorem 20). Also, the present Theorem 6 contains a lower bound on the VC-dimension in addition to an upper bound, which is not previously available in the literature.

Proof We begin with the upper bound in (14). It is shown that if a set $U = \{u_1, \ldots, u_l\} \subseteq \mathbb{R}^n$ is shattered by the collection of half-spaces $\mathcal{H}_k^n$, then $l \leq \lfloor 2k \lg(en) \rfloor$. The proof combines a few ideas that are standard in PAC learning theory, which are stated next.

The first result needed is (Dudley, 1978, Theorem 7.2).[5] It states the following: Suppose that $\mathcal{F}$ is a collection of functions mapping a given set Z into $\mathbb{R}$, such that $\mathcal{F}$ is a k-dimensional real vector space over the field $\mathbb{R}$. Define the associated collection of subsets of X by

$$\text{Pos}(f) := \{z \in Z : f(z) \geq 0\}, \text{Pos}(\mathcal{F}) := \{\text{Pos}(f), f \in \mathcal{F}\}.$$

Then VC-dim(Pos($\mathcal{F}$)) $= k$. To apply the above theorem to this particular instance, we fix the integer k as well as a support set $S \subseteq \{1, \ldots, n\}$ such that $|S| = k$, and choose $\mathcal{F}$ to be the set of functions $\{f(z) = \langle z, x \rangle : \text{supp}(x) \subseteq S\}$. This family of functions is clearly a k-dimensional linear space because the adjustable parameter here is the k-sparse vector x with support in the fixed set S. Therefore it follows that, if we define $\mathcal{H}_S = \{H_k^n(x) : \text{supp}(x) \subseteq S\}$, then VC-dim$(\mathcal{H}_S) = k$.

The next result needed is Sauer's lemma (Sauer, 1972), which states the following: Suppose $\mathcal{C}$ is a collection of subsets of X with finite VC-dimension d, and that $U = \{u_1, \ldots, u_l\} \subseteq X$ with $l > d$. Let $\mathcal{C} \cap U$ denote the collection $\{A \cap B : A \in \mathcal{C}, B \subseteq U\}$. Then

$$|\mathcal{C} \cap U| \leq \sum_{i=0}^{d} \begin{pmatrix} l \\ i \end{pmatrix} \leq \left(\frac{el}{d}\right)^d,$$

where e denotes the base of the natural logarithm. Strictly speaking, Sauer's lemma is the first inequality, which states that the number of subsets of U that can be generated by taking intersections with sets in the collection $\mathcal{C}$ is bounded by the summation shown. The second bound is derived in (Blumer et al., 1989). By applying Sauer's lemma to the problem at hand, it can be seen that, for a fixed support set S, the number of subsets of U that can be generated by intersecting with the collection of half-spaces $\mathcal{H}_S$ is bounded by $(el/k)^k$, because $\mathcal{H}_S$ has VC-dimension k. Note that a similar bound is derived in (Jacques et al., 2013), but without any reference to Sauer's lemma. Also, the result in (Jacques et al., 2013) is specifically for collections of half-spaces, whereas Sauer's lemma is applicable to completely general collections of sets.

5. Note that the definition of VC-dimension used in this reference is the VC-dimension as defined in Definition 3 *plus one*.

Now observe that $\mathcal{H}_k^n$ is just the union of the collections $\mathcal{H}_S$ as S ranges over all subsets of $\{1, \ldots, n\}$ with $|S| = k$. The number of such sets S is the combinatorial parameter n choose k, which is bounded by n^k.[6] Moreover, for each fixed support set S, the collection of subsets $\mathcal{H}_S \cap U$ has cardinality no larger than $(en/k)^k$, as shown above. Therefore

$$|\mathcal{H}_k^n \cap U| \leq \begin{pmatrix} n \\ k \end{pmatrix} \left(\frac{el}{k}\right)^k \leq \left(\frac{nel}{k}\right)^k.$$

The final step in the proof comes from (Vidyasagar, 2003, Lemma 4.6), which states the following (see specifically item 2 of this lemma): Suppose $\alpha, \beta > 0$, $\alpha\beta > 4$ and $l \geq 1$. Then

$$l \leq \alpha \lg(\beta l) \implies l < 2\alpha \lg(\alpha\beta). \tag{15}$$

In the present instance, the collection of sets $\mathcal{H}_k^n \cap U$ has cardinality no larger $(nel/k)^k$, whereas U has 2^l subsets in all. Therefore, if U is shattered by the collection $\mathcal{H}_k^n$, then we must have

$$2^l \leq \left(\frac{nel}{k}\right)^k,$$

or, after taking binary logarithms,

$$l \leq k \lg \frac{nel}{k},$$

which is of the form (15) with $\alpha = k, \beta = ne/k$. Substituting these values into (15) leads to the conclusion that

$$l \leq 2k \lg(ne),$$

provided $\alpha\beta = ne \geq 4$, which holds if $n \geq 2$. Because l is an integer, we can replace the right side by its integer part, which leads to the upper bound in (14).

Now we turn our attention to the lower bound. First we consider the simple case where $k = 1$. Given n, define $l = \lfloor \lg n \rfloor + 1$, so that $n \geq 2^{l-1}$. The first step is to show that the set of half-spaces $\mathcal{H}_1^n$ generated by "one-sparse vectors" has VC-dimension l. Let $s = l - 1 = \lfloor \lg n \rfloor$, and enumerate the 2^s bipolar row vectors in $\{-1, 1\}^s$ in some order, call them $v_1, \ldots v_{2^s}$. Now define the $n \times l$ matrix

$$M = \left[\begin{array}{c|c} 1 & v_1 \\ \vdots & \vdots \\ 1 & v_{2^s} \\ 0_{(n-2^s)\times 1} & 0_{(n-2^s)\times s} \end{array}\right] \in \mathbb{R}^{n \times l}. \tag{16}$$

In other words, the matrix M has a first column of ones, and then the 2^s bipolar vectors in $\{-1, 1\}^s$ in some order, padded by a block of zeros in case $n > 2^{l-1}$. As we shall see below, the "padding" is not used. Now define $U = \{u_0, u_1, \ldots, u_s\}$ denote the $s + 1 = l$ columns of the matrix M. Note that for notational convenience we start numbering the columns with 0 rather than 1. It is claimed that the collection of half-spaces $\mathcal{H}_1^n$ shatters this set U, thus showing that VC-dim$(\mathcal{H}_1^n) \geq l$.

6. Actually n^k is a pretty crude estimate, but as we shall see, it is good enough.

To show that the set U is shattered, let $B \subseteq U$ be an arbitrary subset. Thus B consists of some columns of the matrix M. We examine two cases separately. First, suppose $u_0 \in B$. Then we associate a unique integer r between 1 and 2^s as follows. Define a bipolar vector $\mathbf{i}_B \in \{-1,1\}^s$ by $i_j = 1$ if $u_j \in B$, and $i_j = -1$ if $u_j \notin B$. This bipolar vector $\mathbf{i}_B$ must be one of the vectors $v_1, \ldots, v_{2^s}$. Let r be the unique integer such that $\mathbf{i}_B = v_r$. Define the vector $x \in \mathbb{R}^n$ such that $x_r = 1$, and the remaining elements of x are all zero, and note that $x \in \Sigma_1$. Then $\langle u_0, x\rangle = 1$, while $\langle u_j, x\rangle = 1$ if $u_j \in B$ and $\langle u_j, x\rangle = -1$ if $u_j \notin B$. Therefore the associated half-space $H(x)$ includes precisely the elements of the specified set B. Next, suppose $u_0 \notin B$; in this case we basically flip the signs. Thus the bipolar vector $\mathbf{i}_B \in \{-1,1\}^s$ is chosen such that $i_j = -1$ if $u_j \in B$, and $i_j = 1$ if $u_j \notin B$. If this bipolar vector corresponds to row r in the ordering of $\{-1,1\}^s$, we choose $x \in \Sigma_1$ to have a -1 in row r and zeros elsewhere. This argument shows that the set $\mathcal{H}_1^n$ generated by all one-sparse vectors x has VC-dimension of at least $\lfloor \lg n \rfloor + 1$, which is consistent with the left side of (14) when $k = 1$.

To extend the above argument to general values of k, suppose n and k are specified, and define $l = \lfloor \lg(n/k) \rfloor + 1$ and $s = l - 1 = \lfloor \lg(n/k) \rfloor$. Then $n/k \geq 2^s$, or equivalently, $n \geq k2^s$. Define matrices $M_1, \ldots, M_k \in \{-1,1\}^{2^s \times l}$ in analogy with (16). Then define a matrix $M \in \{0,1\}^{n \times kl}$ as a block-diagonal matrix containing $M_1, \ldots, M_k$ on the diagonal blocks, padded by an appropriate number of zero rows so that the number of rows equals n. In other words, M has the form

$$M = \begin{bmatrix} M_1 & 0_{2^s \times l} & \ldots & 0_{2^s \times l} \\ 0_{2^s \times l} & M_2 & \ldots & 0_{2^s \times l} \\ \vdots & \vdots & \vdots & \vdots \\ 0_{2^s \times l} & \ldots & 0_{2^s \times l} & M_k \\ & & 0_{(n-k2^s) \times kl} & \end{bmatrix} \in \mathbb{R}^{n \times kl}.$$

Define U to be the set of columns of the matrix M, and note that $|U| = kl = k(\lfloor \lg(n/k) \rfloor + 1)$. It is now shown that the set U is shattered by the collection $\mathcal{H}_k^n$ of half-spaces generated by k-sparse vectors. Partition U as $U_1 \cup U_2 \cup \ldots \cup U_k$, where each U_i consists of l column vectors. Then any specified subset $B \subseteq U$ can be expressed as a union $B_1 \cup \cdots \cup B_k$ where $B_i \subseteq U_i$ for each i. Now it is possible to mimic the arguments of the previous paragraph to show that the set U can be shattered by the collection of half-spaces $\mathcal{H}_k^n$. For each subset B_i, identify an integer r_i between 1 and 2^s such that the bipolar vector $\mathbf{i}_{B_i}$ is the r_i-th in the enumeration of $\{-1,1\}^s$. For each index i between 1 and k, let $x_i \in \mathbb{R}^{2^s}$ contain a 1 in row r_i and zeros elsewhere. Define $x \in \mathbb{R}^n$ by stacking x_1 through x_k, followed by $n - k2^s$ zeros. This shows that it is possible to shatter a set of cardinality $k(1 + \lfloor \lg(n/k) \rfloor)$, which is the right inequality in (14). ■

Theorem 6 is applicable to the case where measurements are of the form $\text{sign}(\langle a_i, x\rangle)$. Such measurements can at best lead to the recovery of the direction of a k-sparse vector x, but not its magnitude. In situations where it is desired to recover a sparse vector in its entirety, the measurements are changed to

$$y_i = \text{sign}(\langle a_i, x\rangle + b_i\theta), \tag{17}$$

where x varies over $\Sigma_k \subseteq \mathbb{R}^n$ and $\theta \in \mathbb{R}$. The concept class in this case is given by

$$H_k^{n+1}(x, \theta) = \{(a, b) \in \mathbb{R}^{n+1} : \langle a, x\rangle + b\theta \geq 0\}. \tag{18}$$

By inspecting the equation (17), we deduce that

$$\mathcal{H}_k^n(x) \subseteq \mathcal{H}_k^{n+1}(x) \subseteq \mathcal{H}_{k+1}^{n+1}([x\ \theta]^\top), \tag{19}$$

where $x \in \Sigma_k$ and $\theta \in \mathbb{R}$ so that the vector $[x\ \theta]^\top$ is $k+1$-sparse. We are now ready to state a bound on the VC-dimenstion of the collection of half-spaces $\mathcal{H}_k^{n+1}$.

Theorem 7 *Let $\mathcal{H}_k^{n+1}$ denote the set of half-spaces $H_k^n(x)$ in $\mathbb{R}^n$ as defined in (17). Then*

$$k(1 + \lfloor \lg(n/k) \rfloor) \leq \textit{VC-dim}(\mathcal{H}_k^{n+1}) \leq \lfloor 2(k+1)\lg(e(n+1)) \rfloor. \tag{20}$$

Proof From (19) we conclude that

$$\text{VC-dim}(H_k^n) \leq \text{VC-dim}(\mathcal{H}_k^{n+1}) \leq \text{VC-dim} H_{k+1}^{n+1}([x\ \theta]^\top),$$

then the desired result follows Theorem 6. ■

5. OBCS with Noisy Measurements

In this section we study the one-bit compressed sensing problem when the information available to the learner is a randomly "flipped" version of the true output $\text{sign}(\langle a_i, x\rangle)$ or $\text{sign}(\langle a_i, x\rangle + b_i)$, where x is an unknown k-sparse vector. A study of the problem of PAC learning with noisy measurements was initiated in (Valiant, 1985), shortly after the publication of (Valiant, 1984). Over the years several different models of PAC learning with noisy labels have been studied. Rather than present an exhaustive listing of these, we focus on just a few papers that are most germane to the version of the problem studied here.

In the case of noise-free measurements, the results on learnability were stated in terms of a consistent algorithm, which always exists if one were to assume the axiom of choice. In contrast, in the case where the labels are noisy, it might not be possible to construct a hypothesis that is consistent with the data. Therefore the notion of consistency is replaced by the notion of minimizing empirical risk. Suppose we are given a labelled sample sequence $\{(c_i, y_i) \in X \times \{0,1\}\}_{i \geq 1}$. Suppose $F \in \mathcal{C}$ is a hypothesis. Then the **empirical risk** of the hypothesis with respect to this labelled sequence, after m samples, is defined as

$$\hat{J}_m(T, F) := \frac{1}{m} \sum_{i=1}^{m} |y_i - I_F(c_i)|. \tag{21}$$

Definition 8 *An algorithm $\{A_m\}_{m \geq 1}$ is said to **minimize empirical risk**, or to be a **MER algorithm**, if for all sample sequences $\{(c_i, y_i) \in X \times \{0,1\}\}_{i \geq 1}$, and all integers m, it is the case that*

$$\hat{J}_m(G_m) = \min_{F \in \mathcal{C}} \hat{J}_m(F), \tag{22}$$

where

$$G_m = A_m((c_1, y_1), \ldots (c_m, y_m))$$

is the output of the algorithm after m samples, given the sequence $\{(c_i, y_i) \in X \times \{0,1\}\}_{i \geq 1}$.

Note that if the labels are noise-free, then $y_i = I_T(c_i)$, and

$$\hat{J}_m(T,F) := \frac{1}{m}\sum_{i=1}^{m} |I_T(c_i) - I_F(c_i)|$$

is the empirical estimate of the distance $D_P(T,F)$. In this case, a MER algorithm becomes a consistent algorithm.

We begin by summarizing some relevant results from (Angluin and Laird, 1988). In this paper, the probability of a 1 turning into 0 is assumed to be the same as the probability of a 0 turning into a 1; call it α. Clearly α must be < 0.5 in order for the problem to be learnable, because if the labels are flipped with probability 0.5, then the labels are pure noise and do not convey any information. In this paper, it is *not* assumed that the flipping probability α is known; rather, it is assumed that an upper bound $\bar{\alpha} < 0.5$ for α is known. Moreover, a procedure is given for generating such an upper bound. Suppose $\epsilon, \delta \in (0,1)$ are specified, where ϵ is the accuracy and δ is the confidence, and the objective is to choose the number of samples m to be sufficiently large that $r(m,\epsilon) \leq \delta$, where the learning rate $r(m,\epsilon)$ is defined in (6). The procedure given in this paper generates an upper bound $\bar{\alpha} < 0.5$ for α with confidence $\geq 1 - \delta/2$ using $r = 1 + \lceil \lg[(1-2\alpha)^{-1}] \rceil$ rounds. In case the concept class $\mathcal{C}$ has finite cardinality, say $|\mathcal{C}| \leq N$, then the following result can be established.

Theorem 9 *(See (Angluin and Laird, 1988, Theorem 2).) Given $\epsilon, \delta > 0$, draw*

$$m \geq \frac{2}{\epsilon^2(1-2\bar{\alpha})^2} \ln \frac{2N}{\delta} \tag{23}$$

i.i.d. samples according to an unknown probability measure $\mathcal{P} \in \mathcal{P}^*$. Let $T \in \mathcal{C}$ be any unknown target concept, and let G_m denote the output of a MER algorithm based on m noisy labels. Then

$$\sup_{P \in \mathcal{P}^*} \max_{T \in \mathcal{C}} P^m\{x \in X^m : d_P(G_m, T) \geq \epsilon\} \leq \delta. \tag{24}$$

If an infinite concept class $\mathcal{C}$ has finite VC-dimension, then for each $\epsilon > 0$ and $P \in \mathcal{P}^*$, $\mathcal{C}$ has a finite ϵ-cover with respect to the metric d_P, whose cardinality does not depend on $P \in \mathcal{P}^*$. Using this fact and Theorem 9, it is possible to prove bounds analogous to those in (25) for concept classes with finite VC-dimension. This is done in (Laird, 1988).

Next we discuss the paper (Natarajan et al., 2013). In this paper, the probability of a 1 becoming 0 is *not* assumed to equal the probability of a 0 becoming a 1. However, it is assumed that *both probabilities are known*. This allows the authors to construct an unbiased estimator for the true label, which may not be possible if only bounds for these probabilities are known.

Finally, we mention in passing the paper (Simon, 1996), even though the learning problem studied there is more general than the one studied here. Let $\mathcal{F} \subseteq [0,1]^X$ be a family of functions mapping X into $[0,1]$, and let $f \in \mathcal{F}$ be fixed but unknown. Let $P \in \mathcal{P}^*(X)$ be an unknown probability measure on X, and generate samples $x_1, \ldots, x_m$ in X that are i.i.d. according to P. For each sample x_j, a corresponding label $l_j \in \{0,1\}$ is generated according to

$$l_j = \begin{cases} 1, & \text{w.p. } f(x_j), \\ 0, & \text{w.p. } 1 - f(x_j). \end{cases}$$

From the data $\{(x_j, l_j)\}_{i=1}^m$, the objective is to cconstruct an approximation to $f(\cdot)$.

Now we come to the objective of the present section. By focusing on the *prediction error* (defined below) instead of the distance $d_P(G_m, T)$, it is possible to streamline the statement of the theorem.

To make the problem formulation precise, we use the notation in Section 3.1, whereby X is a set, $\mathcal{S}$ is a σ-algebra of subsets of X, and P is a probability measure on X. To incorporate the randomness, we enlarge X by defining $X_N = X \times \{0,1\}$ as the sample space; define $\mathcal{S}_N$ to be the σ-algebra of subsets in X_N generated by cylinder sets of the form $S \times \{0\}$ and $S \times \{1\}$ for all $S \in \mathcal{S}$; and define a probability measure P_N on X_N by defining

$$P_N(S \times \{0\}) = (1-\alpha)P(S), P_N(S \times \{1\}) = \alpha P(S). \tag{25}$$

Let (c, L) denote a typical element in the sample space X_N. Then

$$\Pr\{L = 0\} = P_N(X \times \{0\}) = 1 - \alpha, \Pr\{L = 1\} = P_N(X \times \{1\}) = \alpha.$$

Here the event $L = 0$ corresponds to the label not being flipped, while the event $L = 1$ corresponds to the label being flipped. It is clear that P_N is a product measure, so that the flipping of labels is independent of the generation of training samples.

Learning takes place as follows: Independent samples $\{(c_i, \omega_i) \in X \times \{0,1\}\}_{i \geq 1}$ are generated in accordance with the above probability measure P_N. Let T be a fixed but unknown target concept. Then for each i, a label y_i is generated as

$$y_i = |I_T(c_i) - \omega_i|. \tag{26}$$

This is equivalent to saying that $y_i = I_T(c_i)$ with probability $1 - \alpha$ and $y_i = 1 - I_T(c_i)$ with probability α. As before, an algorithm is an indexed family of maps $A_m : (X \times \{0,1\})^m \to \mathcal{C}$ for each $m \geq 1$. The algorithm A_m is applied to the set of labelled samples $\{(c_i, y_i)\}_{i=1}^m$, giving rise to a hypothesis G_m.

To assess how well a hypothesis F (however it is derived) approximates the unknown target concept T, we generate a random test input $x \in X$ according to P, and then predict that the oracle output on x will be $I_F(x)$. The error criterion therefore is the **prediction error**, and equals

$$J_N(T,F) := E[|f(I_T(x)) - I_F(x)|, P_N], \tag{27}$$

where $f(I_T(x))$ is the noisy label and $I_F(x)$ is the indicator function of F. The premise in the above definition is that, while the oracle output is noisy, our prediction is not noisy.

Note that, for a given $x \in X$, the quantity $|f(I_T(x)) - I_F(x)|$ equals $|I_T(x) - I_F(x)|$ with probability $1 - \alpha$, and equals $1 - |I_T(x) - I_F(x)|$ with probabilty α. Therefore

$$\begin{aligned} J_N(T,F) &= \int_X [(1-\alpha)|I_T(x) - I_F(x)| + \alpha(1 - |I_T(x) - I_F(x)|)]P(dx) \\ &= \alpha + (1-2\alpha)\int_X |I_T(x) - I_F(x)|P(dx) \\ &= \alpha + (1-2\alpha)d_P(T,F). \end{aligned} \tag{28}$$

Therefore, if we were to define the minimum achievable prediction error $J_N^*(T)$ as

$$J_N^*(T) = \min_{F \in \mathcal{C}} J_N(T,F),$$

then $J_N^*(T) = \alpha$ for each $T \in \mathcal{C}$. Therefore, to quantify the performance of a learning algorithm, we compare the actual prediction error $J_N(G_m, T)$ with $J_N^*(T)$, where G_m is the output of the algorithm based on m samples. With these observations, we can state the following result.

Theorem 10 *Suppose $\mathcal{C}$ is a concept class of subsets of X with VC-dim$(\mathcal{C}) \leq d$, and let $T \in \mathcal{C}$ be a fixed but unknown target concept. Suppose $P \in \mathcal{P}^*$ is a fixed but unknown probability measure on X, $c_1, \ldots, c_m \in X$ are i.i.d. samples drawn according to P, and let G_m be the output of a MER algorithm. Then*

$$\Pr\{J_N(T, G_m) > J_N^*(T) + \epsilon\} \leq c(m, \epsilon), \tag{29}$$

where

$$c(m, \epsilon) = \left[4\left(\frac{0.2em}{d}\right)^{10d} + 1\right] \exp(-0.08m\epsilon^2). \tag{30}$$

Because the estimates derived above are broadly similar to those in (Angluin and Laird, 1988; Laird, 1988), we omit the proof of Theorem 10. The proof can be found in (Ahsen and Vidyasagar, 2017).

Note that the bound for the learning rate in Theorem 10 is qualitatively similar to that in Theorem 4. Therefore the bound (30) can be used to show the following: The quantity $\Pr\{J_N(T, G_m) > J_N^*(T) + \epsilon\}$ can be made smaller than a specified constant δ by choosing

$$m \geq \max\{O((1/\epsilon)\lg(1/\epsilon)), O((1/\epsilon)\lg(1/\delta))\} \tag{31}$$

samples.

Note that the error rate α does not appear explicitly in the right side of (30). However, if we examine (28), we see that α *does* appear in the estimate for the error $d_P(G_m, T)$. Specifically

$$d_P(G_m, T) = \frac{J_N(G_m, T) - J_N^*(T)}{1 - 2\alpha}.$$

Therefore

$$d_P(G_m, T) \leq \epsilon \iff J_N(G_m, T) - J_N^*(T) \leq (1 - 2\alpha)\epsilon. \tag{32}$$

Substituting this into Theorem 10 gives the following estimate.

Corollary 11 *Let all symbols be as in Theorem 10. Then*

$$\Pr\{d_P(G_m, T) \geq \epsilon\} \leq c(m, (1 - 2\alpha)\epsilon) = c(m, \bar{\epsilon}), \tag{33}$$

where $\bar{\epsilon} = (1 - 2\alpha)\epsilon)$, and $c(\cdot, \cdot)$ is defined in (30).

In view of Corollary 11, we infer that, in order to make the quantity $\Pr\{d_P(G_m, T) \geq \epsilon\}$ smaller than a specified constant δ, it is sufficient to choose

$$m \geq \max\{O((1/\bar{\epsilon})\lg(1/\bar{\epsilon})), O((1/\bar{\epsilon})\lg(1/\delta))\} \tag{34}$$

samples. Thus, for a fixed mislabelling probability α, the rate of growth of the sample complexity with respect to ϵ and δ is of the same order as in the noise-free case. However, as $\alpha \to 0.5^-$, the constant under the O symbol becomes larger and larger.

Note that Corollary 11 continues to hold if α is replaced by any upper bound for α.

6. Discussion

In this paper, the problem of one-bit compressed sensing (OBCS) has been formulated as a problem in probably approximately correct (PAC) learning theory. In particular, it has been shown that the VC-dimension of the set of half-spaces in $\mathbb{R}^n$ generated by k-sparse vectors is bounded by $O(k \lg n)$. Therefore, in principle at least, the OBCS problem can be solved using only $O(k \lg n)$ samples. This is possible in principle even when the measurements are corrupted by noise, except that as the mislabelling probability α approaches 0.5, the constant under the O symbol becomes larger and larger. However, in general, it is NP-hard to find a consistent algorithm when measurements are free from noise, and to find an algorithm that minimizes empirical risk when measurements are noisy.

One of the main advantages of formulating OBCS as a problem in PAC learning is that extending these results to the case where the samples $\{a_i\}$ (or $\{(a_i, b_i)\}$ as the case may be) are not i.i.d. essentially "comes for free." It is now known that, if a concept class has finite VC-dimension, then empirical means converge to their true values not only for i.i.d. samples $\{a_i\}$ (or $\{(a_i, b_i)\}$ as the case may be), but also when the samples come from a β-mixing stochastic process, e.g., from a Markov process. The convergence result is established in (Nobel and Dembo, 1993), and explicit rates of convergence are proved in (Karandikar and Vidyasagar, 2002). As it is fairly straight-forward to adapt the various theorems given here to the case of β-mixing processes using the above-mentioned results, the details are omitted.

Acknowledgement

The authors thank the reviewers for very helpful comments.

References

Mehmet Eren Ahsen and Mathukumalli Vidyasagar. An approach to one-bit compressed sensing based on probably approximately correct learning theory. arXiv:1710.07973va1, 2017.

A. Ai, A. Lapanowski, Y. Plan, and R. Vershynin. One-bit compressed sensing with non-Gaussian measurements. *Linear Algebra and Its Applications*, 441:222–239, 2014.

Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2: 343–370, 1988.

Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

P. T. Boufounos and R. G. Baraniuk. 1-bit compressive sensing. In *Proceedings of the Conference on Information Sciences and Systems*, 2008.

Petros T. Boufounos. Greedy sparse signal reconstruction from sign measurements. In *Proceedings of the Asilomar Conference on Signals, Systems, and Computation*, 2009.

Emmanuel J. Candès and Terence Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, December 2005.

Emmanuel J. Candès, J. Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications in Pure and Applied Mathematics*, 59(8): 1207–1223, August 2006.

Mark A. Davenport, Marco F. Duarte, Yonina C. Eldar, and Gitta Kutyniok. Introduction to compressed sensing. In Yonina C. Eldar and Gitta Kutyniok, editors, *Compressed Sensing: Theory and Applications*, pages 1–68. Cambridge University Press, Cambridge, UK, 2012.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4): 1289–1306, April 2006a.

D. L. Donoho. For most large underdetermined systems of linear equations, the minimal ℓ_1-norm solution is also the sparsest solution. *Communications in Pure and Applied Mathematics*, 59(6):797–829, 2006b.

R. M. Dudley. Central limit theorems for empirical measures. *The Annals of Probability*, 6(6):899–929, 1978.

A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82:247–261, 1989.

Michael Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* Springer-Verlag, New York, 2010.

Yonina C. Eldar and Gitta Kutyniok, editors. *Compressed Sensing: Theory and Applications.* Cambridge University Press, Cambridge, UK, 2012.

Simon Foucart and Holger Rauhut. *A Mathematical Introduction to Compressive Sensing.* Springer-Verlag, 2013.

M. X. Goemans and D. P. Williamson. Improved algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995.

Ankit Gupta, Robert Nowak, and Benjamin Recht. Sample complexity for 1-bit compressive sensing and sparse classification. In *Proceedings of the International Symposium on Information Theory*, 2010.

Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations.* CRC Press, Taylor & Francis Group, Boca Raton, FL, 2015.

Laurent Jacques, Jason N. Laska, Petros T. Boufounos, and Richard G. Baraniuk. Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors. *IEEE Transactions on Information Theory*, 59(4):2082–2102, April 2013.

Rajeeva L. Karandikar and M. Vidyasagar. Rates of uniform convergence of empirical means with mixingprocesses. *Statistics & Probability Letters*, 58:297–307, 2002.

Karin Knudson, Rayan Saab, and Rachel Ward. One-bit compressive sensing with norm estimation. *IEEE Transactions on Information Theory*, 62(5):2748–2758, 2016.

Philip D. Laird. *Learning from Good and Bad Data.* Kluwer Academic Press, 1988.

B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24:227–234, 1995.

N. Natarajan, I. Dhillon, P. Ravikumar, and A. Tewari. Learning with noisy labels. *Neural Information Processig Systems*, 26, 2013.

Deanna Needell and Joel A Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

S. Negabhan, P. Ravikumar, M. J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Statistical Science*, 27(4):538–557, December 2012.

Tyler Neylon. Sparse solutions for linear prediction problems. Ph.D. thesis, Department of Mathematics, New York University, 2006.

A. Nobel and A. Dembo. Rates of uniform convergence of empirical means with mixing-processes. *Statistics & Probability Letters*, 17:169–172, 1993.

Yaniv Plan and Ronan Vershynin. One-bit compressed sensing by linear programming. *Communications on Pure and Applied Mathematics*, 66:1275–1297, 2013a.

Yaniv Plan and Ronan Vershynin. Robust 1-bit compressed sensing and sparse logistic regression: A convex programming approach. *IEEE Transactions on Information Theory*, 59(1):482–494, January 2013b.

Irina Rish and Genady Grabarnik. *Sparse Modeling: Theory, Algorithms, and Applications.* CRC Press, Taylor & Francis Group, Boca Raton, FL, 2015.

N. Sauer. On the densities of families of sets. *Journal of Combinatorial Theory, Series A*, 13:145–147, 1972.

Hans Ulrich Simon. General bounds on the number of examples needed for learning probabilistic concepts. *Journal of Computer and System Sciences*, 52(2):239–254, 1996.

L. G. Valiant. Learning conjuctions of disjunctions. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 560–566, 1985.

Leslie G. Valiant. A theory of the learnable. *Journal of the ACM*, 29(11):1134–1142, 1984.

V. N. Vapnik. *Statistical Learning Theory.* John Wiley, New York, 1998.

M. Vidyasagar. *A Theory of Learning and Generalization.* Springer-Verlag, London, 1997.

M. Vidyasagar. *Learning and Generalization: With Applications to Neural Networks and Control Systems*. Springer-Verlag, London, 2003.

Journal of Machine Learning Research 20 (2019) 1-49 Submitted 9/17; Published 2/19

Scalable Kernel K-Means Clustering with Nyström Approximation: Relative-Error Bounds

Shusen Wang SHUSEN.WANG@STEVENS.EDU
Department of Computer Science
Stevens Institute of Technology
Hoboken, NJ 07030, USA

Alex Gittens GITTEA@RPI.EDU
Computer Science Department
Rensselaer Polytechnic Institute
Troy, NY 12180, USA

Michael W. Mahoney MMAHONEY@STAT.BERKELEY.EDU
International Computer Science Institute and Department of Statistics
University of California at Berkeley
Berkeley, CA 94720, USA

Editor: Sanjiv Kumar

Abstract

Kernel k-means clustering can correctly identify and extract a far more varied collection of cluster structures than the linear k-means clustering algorithm. However, kernel k-means clustering is computationally expensive when the non-linear feature map is high-dimensional and there are many input points.

Kernel approximation, e.g., the Nyström method, has been applied in previous works to approximately solve kernel learning problems when both of the above conditions are present. This work analyzes the application of this paradigm to kernel k-means clustering, and shows that applying the linear k-means clustering algorithm to $\frac{k}{\epsilon}(1+o(1))$ features constructed using a so-called rank-restricted Nyström approximation results in cluster assignments that satisfy a $1+\epsilon$ approximation ratio in terms of the kernel k-means cost function, relative to the guarantee provided by the same algorithm without the use of the Nyström method. As part of the analysis, this work establishes a novel $1+\epsilon$ relative-error trace norm guarantee for low-rank approximation using the rank-restricted Nyström approximation.

Empirical evaluations on the 8.1 million instance MNIST8M dataset demonstrate the scalability and usefulness of kernel k-means clustering with Nyström approximation. This work argues that spectral clustering using Nyström approximation—a popular and computationally efficient, but theoretically unsound approach to non-linear clustering—should be replaced with the efficient and theoretically sound combination of kernel k-means clustering with Nyström approximation. The superior performance of the latter approach is empirically verified.

Keywords: kernel k-means clustering, the Nyström method, randomized linear algebra

©2019 Shusen Wang, Alex Gittens, and Michael W. Mahoney.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-517.html.

1. Introduction

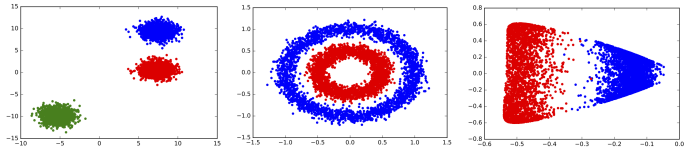
Cluster analysis divides a data set into several groups using information found only in the data points. Clustering can be used in an exploratory manner to discover meaningful groupings within a data set, or it can serve as the starting point for more advanced analyses. As such, applications of clustering abound in machine learning and data analysis, including, *inter alia*: genetic expression analysis (Sharan et al., 2002), market segmentation (Chaturvedi et al., 1997), social network analysis (Handcock et al., 2007), image segmentation (Haralick and Shapiro, 1985), anomaly detection (Chandola et al., 2009), collaborative filtering (Ungar and Foster, 1998), and fast approximate learning of non-linear models (Si et al., 2014).

Linear k-means clustering is a standard and well-regarded approach to cluster analysis that partitions input vectors $\{\mathbf{a}_1,\dots,\mathbf{a}_n\}\subset\mathbb{R}^d$ into k clusters, in an unsupervised manner, by assigning each vector to the cluster with the nearest centroid. Formally, linear k-means clustering seeks to partition the set $[n]=\{1,\dots,n\}$ into k disjoint sets $\mathcal{J}_1,\dots,\mathcal{J}_k$ by solving

$$\underset{\mathcal{J}_1,\dots,\mathcal{J}_k}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{k}\sum_{j\in\mathcal{J}_i}\left\|\mathbf{a}_j-\frac{1}{|\mathcal{J}_i|}\sum_{l\in\mathcal{J}_i}\mathbf{a}_l\right\|_2^2. \tag{1}$$

Lloyd's algorithm (Lloyd, 1982) is a standard approach for finding local minimizers of (1), and is a staple in data mining and machine learning.

Despite its popularity, linear k-means clustering is not a universal solution to all clustering problems. In particular, linear k-means clustering strongly biases the recovered clusters towards isotropy and sphericity. Applied to the data in Figure 1(a), Lloyd's algorithm is perfectly capable of partitioning the data into three clusters which fit these assumptions. However, the data in Figure 1(b) do not fit these assumptions: the clusters are ring-shaped and have coincident centers, so minimizing the linear k-means objective does not recover these clusters.



(a) (b) (c)

Figure 1: Figures 1(a) and 1(b) show two sets of two-dimensional points, with different colors indicating different clusters. Figure 1(a) is separable using linear k-means; Figures 1(b) is inseparable using linear k-means. Figure 1(c) shows the first two dimensions of the feature vectors $\mathbf{k}_1,\dots,\mathbf{k}_n\in\mathbb{R}^n$ derived from a kernel.

To extend the scope of k-means clustering to include anistotropic, non-spherical clusters such as those depicted in Figure 1(b), Schölkopf et al. (1998) proposed to perform linear k-means clustering in a nonlinear feature space instead of the input space. After choosing a feature function $\boldsymbol{\phi}:\mathbb{R}^d\mapsto\mathcal{F}$ to map the input vectors non-linearly into feature vectors,

they propose minimizing the objective function

$$\underset{\mathcal{J}_1,\ldots,\mathcal{J}_k}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{k}\sum_{j\in\mathcal{J}_i}\left\|\boldsymbol{\phi}(\mathbf{a}_j) - \frac{1}{|\mathcal{J}_i|}\sum_{l\in\mathcal{J}_i}\boldsymbol{\phi}(\mathbf{a}_l)\right\|_2^2, \tag{2}$$

where $\{\mathcal{J}_1,\ldots,\mathcal{J}_k\}$ denotes a k-partition of $[n]$. The "kernel trick" enables us to minimize this objective without explicitly computing the potentially high-dimensional features, as inner products in feature space can be computed implicitly by evaluating the kernel function

$$\kappa\big(\mathbf{a}_i,\,\mathbf{a}_j\big) \;=\; \big\langle\boldsymbol{\phi}(\mathbf{a}_i),\,\boldsymbol{\phi}(\mathbf{a}_j)\big\rangle.$$

Thus the information required to solve the kernel k-means problem (2), is present in the kernel matrix $\mathbf{K} = [\kappa(\mathbf{a}_i,\mathbf{a}_j)]_{ij} \in \mathbb{R}^{n\times n}$.

Let $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$ be the full eigenvalue decomposition (EVD) of the kernel matrix and $\mathbf{k}_1,\ldots,\mathbf{k}_n \in \mathbb{R}^n$ be the rows of $\mathbf{V}\mathbf{\Lambda}^{1/2} \in \mathbb{R}^{n\times n}$. It can be shown (see Appendix B.3) that the solution of (2) is identical to the solution of

$$\underset{\mathcal{J}_1,\ldots,\mathcal{J}_k}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{k}\sum_{j\in\mathcal{J}_i}\left\|\mathbf{k}_j - \frac{1}{|\mathcal{J}_i|}\sum_{l\in\mathcal{J}_i}\mathbf{k}_l\right\|_2^2. \tag{3}$$

To demonstrate the power of kernel k-means clustering, consider the dataset in Figure 1(b). We use the Gaussian RBF kernel

$$\kappa(\mathbf{a},\mathbf{a}') = \exp\big(-\tfrac{1}{2\sigma^2}\|\mathbf{a}-\mathbf{a}'\|_2^2\big)$$

with $\sigma = 0.3$, and form the corresponding kernel matrix of the data in Figure 1(b). Figure 1(c) scatterplots the first two coordinates of the feature vectors $\mathbf{k}_1,\ldots,\mathbf{k}_n$. Clearly, the first coordinate of the feature vectors already separates the two classes well, so k-means clustering using the non-linear features $\mathbf{k}_1,\cdots,\mathbf{k}_n$ has a better chance of separating the two classes.

Although it is more generally applicable than linear k-means clustering, kernel k-means clustering is computationally expensive. As a baseline, we consider the cost of optimizing (3). The formation of the kernel matrix $\mathbf{K}$ given the input vectors $\mathbf{a}_1,\ldots,\mathbf{a}_n \in \mathbb{R}^d$ costs $\mathcal{O}(n^2d)$ time. The objective in (3) can then be (approximately) minimized using Lloyd's algorithm at a cost of $\mathcal{O}(n^2k)$ time per iteration. This requires the n-dimensional non-linear feature vectors obtained from the full EVD of $\mathbf{K}$; computing these feature vectors takes $\mathcal{O}(n^3)$ time, because $\mathbf{K}$ is, in general, full-rank. Thus, approximately solving the kernel k-means clustering problem by optimizing (3) costs $\mathcal{O}(n^3 + n^2d + Tn^2k)$ time, where T is the number of iterations of Lloyd's algorithm.

Kernel approximation techniques, including the Nyström method (Nyström, 1930; Williams and Seeger, 2001; Gittens and Mahoney, 2016) and random feature maps (Rahimi and Recht, 2007), have been applied to decrease the cost of solving kernelized machine learning problems: the idea is to replace $\mathbf{K}$ with a low-rank approximation, which allows for more efficient computations. Chitta et al. (2011, 2012) proposed to apply kernel approximations to efficiently approximate kernel k-means clustering. Although kernel approximations mitigate the computational challenges of kernel k-means clustering, the aforementioned works do not provide guarantees on the clustering performance: how

accurate must the low-rank approximation of $\mathbf{K}$ be to ensure near optimality of the approximate clustering obtained via this method?

We propose a provable approximate solution to the kernel k-means problem based on the Nyström approximation. Our method has three steps: first, extract c ($c \ll n$) features using the Nyström method; second, reduce the features to s dimensions ($k \le s < c$) using the truncated SVD;[1] third, apply any off-the-shelf linear k-means clustering algorithm upon the s-dimensional features to obtain the final clusters. The total time complexity of the first two steps is $\mathcal{O}(ndc + nc^2)$. The time complexity of the third step depends on the specific linear k-means algorithm; for example, using Lloyd's algorithm, the per-iteration complexity is $\mathcal{O}(nsk)$, and the number of iterations may depend on s.[2]

Our method comes with a strong approximation ratio guarantee. Suppose we set $s = k/\epsilon$ and $c = \tilde{\mathcal{O}}(\mu s/\epsilon)$ for any $\epsilon \in (0,1)$, where $\mu \in [1, \frac{n}{s}]$ is the coherence parameter of the dominant s-dimensional singular space of the kernel matrix $\mathbf{K}$. Also suppose the standard kernel k-means and our approximate method use the same linear k-means clustering algorithm, e.g., Lloyd's algorithm or some other algorithm that comes with different provable approximation guarantees. As guaranteed by Theorem 2, when the quality of the clustering is measured by the cost function defined in (2), with probability at least 0.9, our algorithm returns a clustering that is at most $1+\mathcal{O}(\epsilon)$ times worse than the standard kernel k-means clustering. Our theory makes explicit the trade-off between accuracy and computation: larger s and c lead to high accuracy and also high computational cost.

Spectral clustering (Shi and Malik, 2000; Ng et al., 2002) is a popular alternative to kernel k-means clustering that can also partition non-linearly separable data such as those in Figure 1(b). Unfortunately, because it requires computing an $n\times n$ affinity matrix and the top k eigenvectors of the corresponding graph Laplacian, spectral clustering is inefficient for large n. Fowlkes et al. (2004) applied the Nyström approximation to increase the scalability of spectral clustering. Since then, spectral clustering with Nyström approximation has been used in many works, e.g., (Arikan, 2006; Berg et al., 2004; Chen et al., 2011; Wang et al., 2016b; Weiss et al., 2009; Zhang and Kwok, 2010). Despite its popularity in practice, this approach does not come with guarantees on the approximation ratio for the obtained clustering. Our algorithm, which combines kernel k-means with Nyström approximation, is an equally computationally efficient alternative that comes with strong bounds on the approximation ratio, and can be used wherever spectral clustering is applied.

1.1. Contributions

Using tools developed in (Boutsidis et al., 2015; Cohen et al., 2015; Feldman et al., 2013), we rigorously analyze the performance of approximate kernel k-means clustering with the Nyström approximation, and show that a rank-$\frac{k}{\epsilon}$ Nyström approximation delivers a $1+\mathcal{O}(\epsilon)$

1. This is why our variant is called the rank-restricted Nyström approximation. The rank-restriction serves two purposes. First, although we do not know whether the rank-restriction is necessary for the $1+\epsilon$ bound, we are unable to establish the bound without it. Second, the rank-restriction makes the third step, linear k-means clustering, much less costly. For the computational benefit, previous works (Boutsidis et al., 2009, 2010, 2015; Cohen et al., 2015; Feldman et al., 2013) have considered dimensionality reduction for linear k-means clustering.
2. Without the rank-restriction, the per-iteration cost would be $\mathcal{O}(nck)$, and the number of iterations may depend on c.

approximation ratio guarantee, relative to the guarantee provided by the same algorithm without the use of the Nyström method.

As part of the analysis of kernel k-means with Nyström approximation, we establish the first relative-error bound for rank-restricted Nyström approximation,[3] which has independent interest.

Kernel k-means clustering and spectral clustering are competing solutions to the nonlinear clustering problem, neither of which scales well with n. Fowlkes et al. (2004) introduced the use of Nyström approximations to make spectral clustering scalable; this approach has become popular in machine learning. We identify fundamental mathematical problems with this heuristic. These concerns and an empirical comparison establish that our proposed combination of kernel k-means with rank-restricted Nyström approximation is a theoretically well-founded and empirically competitive alternative to spectral clustering with Nyström approximation.

Finally, we demonstrate the scalability of this approach by measuring the performance of an Apache Spark implementation of a distributed version of our approximate kernel k-means clustering algorithm using the MNIST8M data set, which has 8.1 million instances and 10 classes.

1.2. Relation to Prior Work

The key to our analysis of the proposed approximate kernel k-means clustering algorithm is a novel relative-error trace norm bound for a *rank-restricted* Nyström approximation. We restrict the rank of the Nyström approximation in a non-standard manner (see Remark 1). Our relative-error trace norm bound is not a simple consequence of the existing bounds for non-rank-restricted Nyström approximation such as the ones provided by Gittens and Mahoney (2016). The relative-error bound which we provide for the rank-restricted Nyström approximation is potentially useful in other applications involving the Nyström method.

The projection-cost preservation (PCP) property (Cohen et al., 2015; Feldman et al., 2013) is an important tool for analyzing approximate linear k-means clustering. We apply our novel relative-error trace norm bound as well as existing tools in (Cohen et al., 2015) to prove that the rank-restricted Nyström approximation enjoys the PCP property. We do not rule out the possibility that the non-rank-restricted (rank-c) Nyström approximation satisfies the PCP property and/or also enjoys a $1+\epsilon$ approximation ratio guarantee when applied to kernel k-means clustering. However, the cost of the linear k-means clustering step in the algorithm is proportional to the dimensionality of the feature vectors, so the rank-restricted Nyström approximation, which produces s-dimensional feature vectors, where $s < c$, is more computationally desirable.

Musco and Musco (2017) similarly establishes a $1+\epsilon$ approximation ratio for the kernel k-means objective when a Nyström approximation is used in place of the full kernel matrix. Specifically, Musco and Musco (2017) shows that when $c = \mathcal{O}(\frac{k}{\epsilon}\log\frac{k}{\epsilon})$ columns of $\mathbf{K}$ are sampled using *ridge leverage score (RLS) sampling* (Alaoui and Mahoney, 2015; Cohen et al., 2017; Musco and Musco, 2017) and are used to form a Nyström approximation,

3. Similar relative-error bounds were independently developed by contemporaneous work of Tropp et al. (2017), in service of the analysis of a novel streaming algorithm for fixed-rank approximation of positive semidefinite matrices. Preliminary versions of this work and theirs were simultaneously submitted to arXiv in June 2017.

Table 1: Commonly used parameters. It holds that $k \le s \le c \le n$.

Notation	Definition
n	number of samples
d	number of features (attributes)
k	number of clusters
s	target rank of the Nyström approximation
c	sketch size of the Nyström approximation

then applying linear k-means clustering to the c-dimensional Nyström features returns a clustering that has objective value at most $1+\epsilon$ times as large as the objective value of the best clustering. Our theory is independent of that in Musco and Musco (2017), and differs in that (1) Musco and Musco (2017) applies specifically to Nyström approximations formed using RLS sampling, whereas our guarantees apply to any sketching method that satisfies the "subspace embedding" and "matrix multiplication" properties (see Lemma 10 for definitions of these two properties); (2) Musco and Musco (2017) establishes a $1+\epsilon$ approximation ratio for the non-rank-restricted RLS-Nyström approximation, whereas we establish a $1+\epsilon$ approximation ratio for the (more computationally efficient) rank-restricted Nyström approximation.

1.3. Paper Organization

In Section 2, we start with a definition of the notation used throughout this paper as well as a background on matrix sketching methods. Then, in Section 3, we present our main theoretical results: Section 3.1 presents an improved relative-error rank-restricted Nyström approximation; Section 3.2 presents the main theoretical results on kernel k-means with Nyström approximation; and Section 3.3 studies kernel k-means with kernel principal component analysis. Section 4 discusses and evaluates the theoretical and empirical merits of kernel k-means clustering versus spectral clustering, when each is approximated using Nyström approximation. Section 5 empirically compares the Nyström method and the random feature maps for the kernel k-means clustering on medium-scale data. Section 6 presents a large-scale distributed implementation in Apache Spark and its empirical evaluation on a data set with 8.1 million points. Section 7 provides a brief conclusion. Proofs are provided in the Appendices.

2. Notation

This section defines the notation used throughout this paper. A set of commonly used parameters is summarized in Table 1.

Matrices and vectors. We take $\mathbf{I}_n$ to be the $n \times n$ identity matrix, $\mathbf{0}$ to be a vector or matrix of all zeros of the appropriate size, and $\mathbf{1}_n$ to be the n-dimensional vector of all ones.

Sets. The set $\{1, 2, \cdots, n\}$ is written as $[n]$. We call $\{\mathcal{J}_1, \cdots, \mathcal{J}_k\}$ a k-partition of $[n]$ if $\mathcal{J}_1 \cup \cdots \cup \mathcal{J}_k = [n]$ and $\mathcal{J}_p \cap \mathcal{J}_q = \emptyset$ when $p \neq q$. Let $|\mathcal{J}|$ denote the cardinality of the set $\mathcal{J}$.

Singular value decomposition (SVD). Let $\mathbf{A} \in \mathbb{R}^{n\times d}$ and $\rho = \text{rank}(\mathbf{A})$. A (compact) singular value decomposition (SVD) is defined by

$$\mathbf{A} \;=\; \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \;=\; \textstyle\sum_{i=1}^{\rho} \sigma_i(\mathbf{A})\mathbf{u}_i\mathbf{v}_i^T,$$

where $\mathbf{U}$, $\mathbf{\Sigma}$, $\mathbf{V}$ are an $n \times \rho$ column-orthogonal matrix, a $\rho \times \rho$ diagonal matrix with nonnegative entries, and a $d \times \rho$ column-orthogonal matrix, respectively. If $\mathbf{A}$ is symmetric positive semi-definite (SPSD), then $\mathbf{U} = \mathbf{V}$, and this decomposition is also called the (reduced) eigenvalue decomposition (EVD). By convention, we take $\sigma_1(\mathbf{A}) \geq \cdots \geq \sigma_\rho(\mathbf{A})$.

Truncated SVD. The matrix $\mathbf{A}_s = \sum_{i=1}^{s} \sigma_i(\mathbf{A})\mathbf{u}_i\mathbf{v}_i^T$ is a rank-s truncated SVD of $\mathbf{A}$, and is an optimal rank-s approximation to $\mathbf{A}$ when the approximation error is measured in a unitarily invariant norm.

The Moore-Penrose inverse of $\mathbf{A}$ is defined by $\mathbf{A}^\dagger = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T$.

Leverage score and coherence. Let $\mathbf{U} \in \mathbb{R}^{n\times\rho}$ be defined in the above and $\mathbf{u}_i$ be the i-th row of $\mathbf{U}$. The row leverage scores of $\mathbf{A}$ are $l_i = \|\mathbf{u}_i\|_2^2$ for $i \in [n]$. The row coherence of $\mathbf{A}$ is $\mu(\mathbf{A}) = \frac{n}{\rho}\max_i \|\mathbf{u}_i\|_2^2$. The leverage scores for a matrix $\mathbf{A}$ can be computed exactly in the time it takes to compute the matrix $\mathbf{U}$; and the leverage scores can be approximated (in theory (Drineas et al., 2012) and in practice (Gittens and Mahoney, 2016)) in roughly the time it takes to apply a random projection matrix to the matrix $\mathbf{A}$.

Matrix norms. We use three matrix norms in this paper:

$$\begin{aligned}
\text{Frobenius Norm:} \quad & \|\mathbf{A}\|_F \;=\; \sqrt{\textstyle\sum_{i,j} a_{ij}^2} \;=\; \sqrt{\textstyle\sum_i \sigma_i^2(\mathbf{A})};\\
\text{Spectral Norm:} \quad & \|\mathbf{A}\|_2 \;=\; \max_{\|\mathbf{x}\|_2=1} \left\|\mathbf{A}\mathbf{x}\right\|_2 \;=\; \sigma_1(\mathbf{A});\\
\text{Trace Norm:} \quad & \|\mathbf{A}\|_* \;=\; \textstyle\sum_i \sigma_i(\mathbf{A}).
\end{aligned}$$

Any square matrix satisfies $\text{tr}(\mathbf{A}) \leq \|\mathbf{A}\|_*$. If additionally $\mathbf{A}$ is SPSD, then $\text{tr}(\mathbf{A}) = \|\mathbf{A}\|_*$.

Matrix sketching Here, we briefly review matrix sketching methods that are commonly used within randomized linear algebra (RLA) (Mahoney, 2011).

Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, we call $\mathbf{C} = \mathbf{A}\mathbf{P} \in \mathbb{R}^{m\times c}$ (typically $c \ll n$) a *sketch* of $\mathbf{A}$ and $\mathbf{P} \in \mathbb{R}^{n\times c}$ a *sketching matrix*. Within RLA, sketching has emerged as a powerful primitive, where one is primarily interested in using random projections and random sampling to construct randomzed sketches (Mahoney, 2011; Drineas and Mahoney, 2016). In particular, sketching is useful as it allows large matrices to be replaced with smaller matrices which are more amenable to efficient computation, but provably retain almost optimal accuracy in many computations (Mahoney, 2011; Woodruff, 2014). The columns of $\mathbf{C}$ typically comprise a rescaled subset of the columns of $\mathbf{A}$, or random linear combinations of the columns of $\mathbf{A}$; the former type of sketching is called *column selection* or *random sampling*, and the latter is referred to as *random projection*.

Column selection forms $\mathbf{C} \in \mathbb{R}^{m\times c}$ using a randomly sampled and rescaled subset of the columns of $\mathbf{A} \in \mathbb{R}^{m\times n}$. Let $p_1, \cdots, p_n \in (0, 1)$ be the sampling probabilities associated

with the columns of $\mathbf{A}$ (so that, in particular, $\sum_{i=1}^n p_i = 1$). The columns of the sketch are selected identically and independently as follows: each column of $\mathbf{C}$ is randomly sampled from the columns of $\mathbf{A}$ according to the sampling probabilities and rescaled by $\frac{1}{\sqrt{cp_i}}$, where i is the index of the column of $\mathbf{A}$ that was selected. In our matrix multiplication formulation for sketching, column selection corresponds to a sketching matrix $\mathbf{P} \in \mathbb{R}^{n\times c}$ that has exactly one non-zero entry in each column, whose position and magnitude correspond to the index of the column selected from $\mathbf{A}$. *Uniform sampling* is column sampling with $p_1 = \cdots = p_n = \frac{1}{n}$, and *leverage score sampling* takes $p_i = \frac{l_i}{\sum_j l_j}$ for $i \in [n]$, where l_i is the i-th leverage score of some matrix (typically $\mathbf{A}$, $\mathbf{A}_k$, or an randomized approximation thereto) (Drineas et al., 2012).

Gaussian projection is a type of random projection where the sketching matrix is taken to be $\mathbf{P} = \frac{1}{\sqrt{c}}\mathbf{G} \in \mathbb{R}^{n\times c}$; here the entries of $\mathbf{G}$ are i.i.d. $\mathcal{N}(0, 1)$ random variables. Gaussian projection is inefficient relative to column sampling: the formation of a Gaussian sketch of a dense $m \times n$ matrix requires $\mathcal{O}(mnc)$ time. The *Subsampled Randomized Hadamard Transform (SRHT)* is a more efficient alternative that enjoys similar properties to the Gaussian projection (Drineas et al., 2011; Lu et al., 2013; Tropp, 2011), and can be applied to a dense $m \times n$ matrix in only $\mathcal{O}(mn \log c)$ time. The *CountSketch* is even more efficient: it can be applied to any matrix $\mathbf{A}$ in $\mathcal{O}(\text{nnz}(\mathbf{A}))$ time (Clarkson and Woodruff, 2013; Meng and Mahoney, 2013; Nelson and Nguyên, 2013), where $\text{nnz}(\cdot)$ denotes the number of nonzero entries in a matrix.

3. Our Main Results: Improved SPSD Matrix Approximation and Kernel k-means Approximation

In this section, we present our main theoretical results. We start, in Section 3.1, by presenting Theorem 1, a novel result on SPSD matrix approximation with the rank-restricted Nyström method. This result is of independent interest, and so we present it in detail, but in this paper we will use it to establish our main result. Then, in Section 3.2, we present Theorem 2, which is our main result for approximate kernel k-means with the Nyström approximation. In Section 3.3, we establish novel guarantees on kernel k-means with dimensionality reduction.

3.1. The Nyström Method

The Nyström method (Nyström, 1930) is the most popular kernel approximation method in the machine learning community. Let $\mathbf{K} \in \mathbb{R}^{n\times n}$ be an SPSD matrix and $\mathbf{P} \in \mathbb{R}^{n\times c}$ be a sketching matrix. The Nyström method approximates $\mathbf{K}$ with $\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T$, where $\mathbf{C} = \mathbf{K}\mathbf{P}$ and $\mathbf{W} = \mathbf{P}^T\mathbf{K}\mathbf{P}$. The Nyström method was introduced to the machine learning community by Williams and Seeger (2001); since then, numerous works have studied its theoretical properties, e.g., (Drineas and Mahoney, 2005; Gittens and Mahoney, 2016; Jin et al., 2013; Kumar et al., 2012; Wang and Zhang, 2013; Yang et al., 2012).

Empirical results in (Gittens and Mahoney, 2016; Wang et al., 2016b; Yang et al., 2012) demonstrated that the accuracy of the Nyström method significantly increases when the spectrum of $\mathbf{K}$ decays fast. This suggests that the Nyström approximation captures the dominant eigenspaces of $\mathbf{K}$, and that error bounds comparing the accuracy of the Nyström

Table 2: Sufficient sketch sizes for several sketching methods used to construct Nyström approximations to the matrix $\mathbf{K}$. Let $\mathbf{K}_s = \mathbf{V}_s\mathbf{\Sigma}_s\mathbf{V}_s^T$ be the truncated SVD of $\mathbf{K}$, then μ denotes the row coherence of $\mathbf{V}_s \in \mathbb{R}^{n\times s}$ and the leverage score sampling is done using the row leverage scores of $\mathbf{V}_s$.

sketching	**sketch size** (c)	**time complexity** (T)
uniform sampling	$\mathcal{O}(\mu s/\epsilon + \mu s \log s)$	$\mathcal{O}(nc)$
leverage sampling	$\mathcal{O}(s/\epsilon + s\log s)$	$\tilde{\mathcal{O}}(n^2 s)$
Gaussian projection	$\mathcal{O}(s/\epsilon)$	$\mathcal{O}(n^2 c)$
SRHT	$\mathcal{O}((s+\log n)(\epsilon^{-1}+\log s))$	$\mathcal{O}(n^2 \log c)$
CountSketch	$\mathcal{O}(s/\epsilon + s^2)$	$\mathcal{O}(n^2)$

approximation of $\mathbf{K}$ to that of the best rank-s approximation $\mathbf{K}_s$ (for $s < c$) would provide a meaningful measure of the performance of the Nyström kernel approximation. Gittens and Mahoney (2016) established the first relative-error bounds showing that for sufficiently large c, the trace norm error $\|\mathbf{K} - \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\|_*$ is comparable to $\|\mathbf{K} - \mathbf{K}_s\|_*$. Such results quantify the benefits of spectral decay to the performance of the Nyström method, and are sufficient to analyze the performance of Nyström approximations in applications such as kernel ridge regression (Alaoui and Mahoney, 2015; Bach, 2013) and kernel support vector machines (Cortes et al., 2010).

However, Gittens and Mahoney (2016) did not analyze the performance of rank-restricted Nyström approximations; they compared the approximation accuracies of the rank-c matrix $\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T$ and the rank-s matrix $\mathbf{K}_s$ (recall $s < c$). In our application to approximate kernel k-means clustering, it is the rank-s matrix $(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s$ that is of relevance. Given $\mathbf{C}$ and $\mathbf{W}$, the truncated SVD $\left((\mathbf{W}^\dagger)^{1/2}\mathbf{C}^T\right)_s = \tilde{\mathbf{U}}_s\tilde{\mathbf{\Sigma}}_s\tilde{\mathbf{V}}_s^T$ can be found using $\mathcal{O}(nc^2)$ time. Then the rank-s Nyström approximation can be written as

$$(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s \;=\; \left(\mathbf{C}(\mathbf{W}^\dagger)^{1/2}\right)_s\left(\mathbf{C}(\mathbf{W}^\dagger)^{1/2}\right)_s^T \;=\; \tilde{\mathbf{V}}_s\tilde{\mathbf{\Sigma}}_s^2\tilde{\mathbf{V}}_s^T. \tag{4}$$

Theorem 1 provides a relative-error trace norm approximation guarantee for the sketch (4) and is novel; a proof is provided in Appendix A.

Theorem 1 (Relative-Error Rank-Restricted Nyström Approximation) *Let $\mathbf{K} \in \mathbb{R}^{n\times n}$ be an SPSD matrix, s be the target rank, and $\epsilon \in (0,1)$ be an error parameter. Let $\mathbf{P} \in \mathbb{R}^{n\times c}$ be a sketching matrix corresponding to one of the sketching methods listed in Table 2. Let $\mathbf{C} = \mathbf{K}\mathbf{P}$ and $\mathbf{W} = \mathbf{P}^T\mathbf{K}\mathbf{P}$. Then*

$$\left\|\mathbf{K} - (\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s\right\|_* \;\leq\; (1+\epsilon)\left\|\mathbf{K} - \mathbf{K}_s\right\|_*$$

holds with probability at least 0.9. In addition, there exists an $n \times s$ column orthogonal matrix $\mathbf{Q}$ such that $(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s = \mathbf{K}^{1/2}\mathbf{Q}\mathbf{Q}^T\mathbf{K}^{1/2}$.

Remark 1 (Rank Restrictions) *The traditional rank-restricted Nyström approximation, $\mathbf{C}(\mathbf{W}_s)^\dagger\mathbf{C}^T$, (Drineas and Mahoney, 2005; Fowlkes et al., 2004; Gittens and Mahoney, 2016; Li et al., 2015) is not known to satisfy a relative-error bound of the form guaranteed in Theorem 1. Pourkamali-Anaraki and Becker (2016) pointed out the drawbacks of the traditional rank-restricted Nyström approximation and proposed the use of the rank-restricted Nyström approximation $(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s$ in applications requiring kernel approximations, but provided only empirical evidence of its performance. This work provides guarantees on the approximation error of the rank-restricted Nyström approximation $(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s$, and applies this approximation to the kernel k-means clustering problem. The contemporaneous work Tropp et al. (2017) provides similar guarantees on the approximation error of $(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s$, and uses this Nyström approximation as the basis of a streaming algorithm for fixed-rank approximation of positive-semidefinite matrices.*

3.2. Main Result for Approximate Kernel k-means

In this section we establish the approximation ratio guarantees for the objective function of kernel k-means clustering. We first define γ-approximate k-means algorithms (where $\gamma \geq 1$), then present our main result in Theorem 2.

Let $\mathbf{A}$ be a matrix with n rows $\mathbf{a}_1, \cdots, \mathbf{a}_n$. The objective function for linear k-means clustering over the rows of $\mathbf{A}$ is

$$f(\mathcal{J}_1, \cdots, \mathcal{J}_k; \mathbf{A}) \;=\; \frac{1}{n}\sum_{i=1}^{k}\sum_{j\in\mathcal{J}_i}\left\|\mathbf{a}_j \;-\; \frac{1}{|\mathcal{J}_i|}\sum_{l\in\mathcal{J}_i}\mathbf{a}_l\right\|_2^2.$$

The minimization of f w.r.t. the k-partition $\{\mathcal{J}_1, \cdots, \mathcal{J}_k\}$ is NP-hard (Garey et al., 1982; Aloise et al., 2009; Dasgupta and Freund, 2009; Mahajan et al., 2009; Awasthi et al., 2015), but approximate solutions can be obtained in polynomial time. γ-approximate algorithms capture one useful notion of approximation.

Definition 1 (γ-Approximate Algorithms) *A linear k-means clustering algorithm $\mathcal{A}_\gamma$ takes as input a matrix $\mathbf{Z}$ with n rows and outputs $\{\mathcal{J}_1', \cdots, \mathcal{J}_k'\}$. We call $\mathcal{A}_\gamma$ a γ-approximate algorithm if, for any such matrix $\mathbf{Z}$,*

$$f(\mathcal{J}_1', \cdots, \mathcal{J}_k'; \mathbf{Z}) \;\leq\; \gamma \cdot \min_{\mathcal{J}_1, \cdots, \mathcal{J}_k} f(\mathcal{J}_1, \cdots, \mathcal{J}_k; \mathbf{Z}).$$

Here $\{\mathcal{J}_1, \cdots, \mathcal{J}_k\}$ and $\{\mathcal{J}_1', \cdots, \mathcal{J}_k'\}$ are k-partitions of $[n]$.

Many $(1+\epsilon)$-approximation algorithms have been proposed, but they are computationally expensive (Chen, 2009; Har-Peled and Mazumdar, 2004; Kumar et al., 2004; Matousek, 2000). There are also relatively efficient constant factor approximate algorithms, e.g., (Arthur and Vassilvitskii, 2007; Kanungo et al., 2002; Song and Rajasekaran, 2010).

Let $\boldsymbol{\phi}$ be a feature map, $\mathbf{\Phi}$ be the matrix with rows $\phi(\mathbf{a}_1), \ldots, \phi(\mathbf{a}_n)$, and $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^T \in \mathbf{R}^{n\times n}$ be the associated kernel matrix. Analogously, we denote the objective function for kernel k-means clustering by

$$f(\mathcal{J}_1, \cdots, \mathcal{J}_k\,;\,\mathbf{\Phi}) \;=\; \frac{1}{n}\sum_{i=1}^{k}\sum_{j\in\mathcal{J}_i}\left\|\boldsymbol{\phi}(\mathbf{a}_j) \;-\; \frac{1}{|\mathcal{J}_i|}\sum_{l\in\mathcal{J}_i}\boldsymbol{\phi}(\mathbf{a}_l)\right\|_2^2,$$

where $\{\mathcal{J}_1, \cdots, \mathcal{J}_k\}$ is a k-partition of $[n]$.

Theorem 2 (Kernel k-Means with Nyström Approximation) *Choose a sketching matrix $\mathbf{P} \in \mathbb{R}^{n\times c}$ and sketch size c consistent with Table 2. Let $\mathbf{CW}^{\dagger}\mathbf{C}^T$ be the previously defined Nyström approximation of $\mathbf{K}$. Let $\mathbf{B} \in \mathbb{R}^{n\times s}$ be any matrix satisfying $\mathbf{BB}^T = (\mathbf{CW}^{\dagger}\mathbf{C}^T)_s$. Let the k-partition $\{\tilde{\mathcal{J}}_1, \cdots, \tilde{\mathcal{J}}_k\}$ be the output of a γ-approximate algorithm applied to the rows of $\mathbf{B}$. With probability at least 0.9,*

$$f(\tilde{\mathcal{J}}_1, \cdots, \tilde{\mathcal{J}}_k\,;\,\mathbf{\Phi}) \;\;\leq\;\; \gamma\big(1+\epsilon+\tfrac{k}{s}\big)\cdot \min_{\mathcal{J}_1,\cdots,\mathcal{J}_k} f(\mathcal{J}_1, \cdots, \mathcal{J}_k\,;\,\mathbf{\Phi}).$$

Remark 2 *Kernel k-means clustering is an NP-hard problem. Therefore, instead of comparing with $\min f$, we compare with clusterings obtained using γ-approximate algorithms. Theorem 2 shows that, when uniform sampling to form the Nyström approximation, if $s = \mathcal{O}(\frac{k}{\epsilon})$ and $c = \tilde{\mathcal{O}}(\frac{\mu s}{\epsilon})$, then the returned clustering has an objective value that is at most a factor of ϵ larger than the objective value of the kernel k-means clustering returned by the γ-approximate algorithm.*

Remark 3 *Assume we are in a practical setting where c, the budget of column samples one can use to form a Nyström approximation, and k, the number of desired cluster centers, are fixed. The pertinent question is how to choose s to produce a high-quality approximate clustering. Theorem 2 shows that for uniform sampling, the error ratio is*

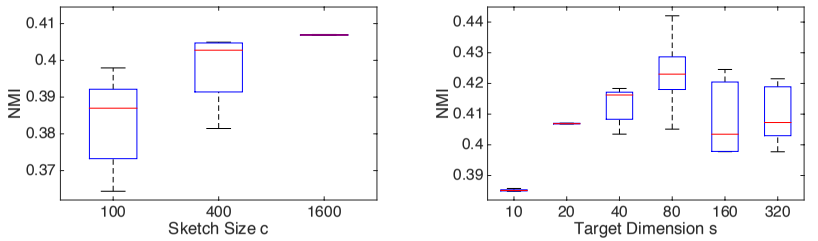
$$1+\epsilon+\tfrac{k}{s} \;=\; 1+\tilde{\mathcal{O}}(\tfrac{s\mu}{c})+\tfrac{k}{s}.$$

To balance the two sources of error, s must be larger than k, but not too large a fraction of c. To minimize the above error ratio, s should be selected on the order of $\sqrt{kc/\mu}$. Since the matrix coherence μ (≥ 1) is unknown, it can be heuristically treated as a constant.

We empirically study the effect of the values of c and s using a data set comprising 8.1 million samples. Note that computing the kernel k-means clustering objective function requires the formation of the entire kernel matrix $\mathbf{K}$, which is infeasible for a data set of this size; instead, we use normalized mutual information (NMI) (Strehl and Ghosh, 2002)—a standard measure of the performance of clustering algorithms— to measure the quality of the clustering obtained by approximating kernel k-means clustering using Nyström approximations formed through uniform sampling. NMI scores range from zero to one, with a larger score indicating better performance. We report the results in Figure 2. The complete details of the experiments, including the experimental setting and time costs, are given in Section 6.

From Figure 2(a) we observe that larger values of c lead to better and more stable clusterings: the mean of the NMI increases and its standard deviation decreases. This is reasonable and in accordance with our theory. However, larger values of c incur more computations, so one should choose c to trade off computation and accuracy.

Figure 2(b) shows that for fixed k and c, the clustering performance is not monotonic in s, which matches Theorem 2 (see the discussion in Remark 3). Setting s as small as k results in poor performance. Setting s over-large not only incurs more computations, but also negatively affects clustering performance; this may suggest the necessity of rank-restriction. Furthermore, in this example, $\sqrt{kc} = 126.5$, which corroborates the suggestion made in Remark 3 that setting s around $\sqrt{kc/\mu}$ (where μ is unknown but can be treated as a constant larger than 1) can be a good choice.



(a) Fix $s = 20$ and vary c. (b) Fix $c = 1600$ and vary s.

Figure 2: Performance of approximate kernel k-means clustering on the MNIST8M data set, for which $n = 8.1 \times 10^6$, $d = 784$, and $k = 10$. We use Nyström approximations formed using uniform sampling, and the built-in k-means algorithm of Spark MLlib (a parallelized variant of the k-means++ method) with the setting MaxIter $= 100$.

Remark 4 *Musco and Musco (2017) established a $1+\epsilon$ approximation ratio for the kernel k-means objective value when a non-rank-restricted Nyström approximation is formed using ridge leverage scores (RLS) sampling; their analysis is specific to RLS sampling and does not extend to other sketching methods. By way of comparison, our analysis covers several popular sampling schemes and applies to rank-restricted Nyström approximations, but does not extend to RLS sampling.*

3.3. Approximate Kernel k-Means with KPCA and Power Method

The use of dimensionality reduction to increase the computational efficiency of k-means clustering has been widely studied, e.g. in (Boutsidis et al., 2010, 2015; Cohen et al., 2015; Feldman et al., 2013; Zha et al., 2002). Kernel principal component analysis (KPCA) is particularly well-suited to this application (Dhillon et al., 2004; Ding et al., 2005). Applying Lloyd's algorithm on the rows of $\mathbf{\Phi}$ or $\mathbf{K}^{-1/2}$ has an $\mathcal{O}(n^2k)$ per-iteration complexity; if s features are extracted using KPCA and Lloyd's algorithm is applied to the resulting s-dimensional feature map, then the per-iteration cost reduces to $\mathcal{O}(nsk)$. Proposition 3 states that, to obtain a $1+\epsilon$ approximation ratio in terms of the kernel k-means objective function, it suffices to use $s = \frac{k}{\epsilon}$ KPCA features. This proposition is a simple consequence of (Cohen et al., 2015).

Proposition 3 (KPCA) *Let $\mathbf{\Phi}$ be a matrix with n rows, and $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^T \in \mathbf{R}^{n\times n}$ be the corresponding kernel matrix. Let $\mathbf{K}_s = \mathbf{V}_s\mathbf{\Lambda}_s\mathbf{V}_s^T$ be the truncated SVD of $\mathbf{K}$ and take $\mathbf{B} = \mathbf{V}_s\mathbf{\Lambda}_s^{1/2} \in \mathbb{R}^{n\times s}$. Let the k-partition $\{\tilde{\mathcal{J}}_1, \cdots, \tilde{\mathcal{J}}_k\}$ be the output of a γ-approximate algorithm applied to the rows of $\mathbf{B}$. Then*

$$f(\tilde{\mathcal{J}}_1, \cdots, \tilde{\mathcal{J}}_k\,;\,\mathbf{\Phi}) \;\;\leq\;\; \gamma\,(1+\tfrac{k}{s})\cdot \min_{\mathcal{J}_1,\cdots,\mathcal{J}_k} f(\mathcal{J}_1, \cdots, \mathcal{J}_k\,;\,\mathbf{\Phi}).$$

Algorithm 1 Approximate Kernel k-Means using the Power Method.

Input: kernel matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$, number of clusters k, target dimension s ($\geq k$), sketch size c ($\geq s$), number of iterations t (≥ 1)
Draw a Gaussian projection matrix $\mathbf{P} \in \mathbb{R}^{n\times c}$;
for all $j \in [t]$ **do**
  $\mathbf{P} \longleftarrow \mathbf{KP}$;
end for
Orthogonalize $\mathbf{P}$ to obtain $\mathbf{U} \in \mathbb{R}^{n\times c}$;
Compute $\mathbf{C} = \mathbf{KU}$ and $\mathbf{W} = \mathbf{U}^T\mathbf{KU}$;
Compute a $\mathbf{B} \in \mathbb{R}^{n\times s}$ satisfying $(\mathbf{CW}^\dagger\mathbf{C}^T)_s = \mathbf{BB}^T$;
return $\mathbf{B}$.

In practice, the truncated SVD (equivalently EVD) of $\mathbf{K}$ is computed using the power method or Krylov subspace methods. These numerical methods do not compute the exact decomposition $\mathbf{K}_s = \mathbf{V}_s\mathbf{\Lambda}_s\mathbf{V}_s^T$, so Proposition 3 is not directly applicable. It is useful to have a theory that captures the effect of realisticly inaccurate estimates like $\mathbf{K}_s \approx \tilde{\mathbf{V}}_s\tilde{\mathbf{\Lambda}}_s\tilde{\mathbf{V}}_s^T$ on the clustering process. As one particular example, consider that general-purpose implementations of the truncated SVD attempt to mitigate the fact that the computed decompositions are inaccurate by returning very high-precision solutions, e.g. solutions that satisfy $\|(\mathbf{I}_n - \mathbf{V}_s\mathbf{V}_s^T)\tilde{\mathbf{V}}_s\|_2 \leq 10^{-10}$. Understanding the trade-off between the precision of the truncated SVD solution and the impact on the approximation ratio of the approximate kernel k-means solution allows us to more precisely manage the computational complexity of our algorithms. Are such high-precision solutions necessary for kernel k-means clustering?

Theorem 4 answers this question by establishing that highly accurate eigenspaces are not significantly more useful in approximate kernel k-means clustering than eigenspace estimates with lower accuracy. A low-precision solution obtained by running the power iteration for a few rounds suffices for kernel k-means clustering applications. We prove Theorem 4 in Appendix C.

Theorem 4 (The Power Method) *Let $\mathbf{\Phi}$ be a matrix with n rows, $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^T \in \mathbb{R}^{n\times n}$ be the corresponding kernel matrix, and σ_i be the i-th singular value of $\mathbf{K}$. Fix an error parameter $\epsilon \in (0,1)$. Run Algorithm 1 with $t = \mathcal{O}(\frac{\log(n/\epsilon)}{\log(\sigma_s/\sigma_{s+1})})$ to obtain $\mathbf{B} \in \mathbb{R}^{n\times s}$. Let the k-partition $\{\tilde{\mathcal{J}}_1, \cdots, \tilde{\mathcal{J}}_k\}$ be the output of a γ-approximate algorithm applied to the rows of $\mathbf{B}$. If $c = s + \mathcal{O}(\log\frac{1}{\delta})$, then*

$$f(\tilde{\mathcal{J}}_1, \cdots, \tilde{\mathcal{J}}_k\,;\,\mathbf{\Phi}) \;\leq\; \gamma\left(1+\epsilon+\tfrac{k}{s}\right)\cdot \min_{\mathcal{J}_1,\cdots,\mathcal{J}_k} f(\mathcal{J}_1, \cdots, \mathcal{J}_k\,;\,\mathbf{\Phi}).$$

holds with probability at least $1-\delta$. If $c = s$, then the above inequality holds with probability $0.9 - \mathcal{O}(s^{-\tau})$, where τ is a positive constant (Tao and Vu, 2010).

Note that the power method requires forming the entire kernel matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$, which may not fit in memory even in a distributed setting. Therefore, in practice, the power method may not be as efficient as the Nyström approximation with uniform sampling, which avoids forming $\mathbf{K}$.

Theorem 2, Proposition 3, and Theorem 4 are highly interesting from a theoretical perspective. These results demonstrate that $s = \frac{k}{\epsilon}\big(1 + o(1)\big)$ features are sufficient to

ensure a $(1+\epsilon)$ approximation ratio. Prior work (Dhillon et al., 2004; Ding et al., 2005) set $s = k$ and did not provide approximation ratio guarantees. Indeed, a lower bound in the linear k-means clustering case due to (Cohen et al., 2015) shows that $s = \Omega(\frac{k}{\epsilon})$ is necessary to obtain a $1+\epsilon$ approximation ratio.

4. Comparison to Spectral Clustering with Nyström Approximation

In this section, we provide a brief discussion and empirical comparison of our clustering algorithm, which uses the Nyström method to approximate kernel k-means clustering, with the popular alternative algorithm that uses the Nyström method to approximate spectral clustering.

4.1. Background

Spectral clustering is a method with a long history (Cheeger, 1969; Donath and Hoffman, 1972, 1973; Fiedler, 1973; Guattery and Miller, 1995; Spielman and Teng, 1996). Within machine learning, spectral clustering is more widely used than kernel k-means clustering (Ng et al., 2002; Shi and Malik, 2000), and the use of the Nyström method to speed up spectral clustering has been popular since Fowlkes et al. (2004). Both spectral clustering and kernel k-means clustering can be approximated in time linear in n by using the Nyström method with uniform sampling. Practitioners reading this paper may ask:

> *How does the approximate kernel k-means clustering algorithm presented here, which uses Nyström approximation, compare to the popular heuristic of combining spectral clustering with Nyström approximation?*

Based on our theoretical results and empirical observations, our answer to this reader is:

> *Although they have equivalent computational costs, kernel k-means clustering with Nyström approximation is both more theoretically sound and more effective in practice than spectral clustering with Nyström approximation.*

We first formally describe spectral clustering, and then substantiate our claim regarding the theoretical advantage of our approximate kernel k-means method. Our discussion is limited to the normalized and symmetric graph Laplacians used in Fowlkes et al. (2004), but spectral clustering using asymmetric graph Laplacians encounters similar issues.

4.2. Spectral Clustering with Nyström Approximation

The input to the spectral clustering algorithm is an affinity matrix $\mathbf{K} \in \mathbb{R}_+^{n\times n}$ that measures the pairwise similarities between the points being clustered; typically $\mathbf{K}$ is a kernel matrix or the adjacency matrix of a weighted graph constructed using the data points as vertices. Let $\mathbf{D} = \mathsf{diag}(\mathbf{K}\mathbf{1}_n)$ be the diagonal degree matrix associated with $\mathbf{K}$, and $\mathbf{L} = \mathbf{I}_n - \mathbf{D}^{-1/2}\mathbf{K}\mathbf{D}^{-1/2}$ be the associated normalized graph Laplacian matrix. Let $\mathbf{V}_k \in \mathbb{R}^{n\times k}$ denote the bottom k eigenvectors of $\mathbf{L}$, or equivalently, the top k eigenvectors of $\mathbf{D}^{-1/2}\mathbf{K}\mathbf{D}^{-1/2}$. Spectral clustering groups the data points by performing linear k-means clustering on the normalized rows of $\mathbf{V}_k$. Fowlkes et al. (2004) popularized the application of the Nyström approximation to spectral clustering. This algorithm computes an approximate spectral

clustering by: (1) forming a Nyström approximation to $\mathbf{K}$, denoted by $\tilde{\mathbf{K}}$; (2) computing the degree matrix $\tilde{\mathbf{D}} = \mathsf{diag}(\tilde{\mathbf{K}}\mathbf{1}_n)$ of $\tilde{\mathbf{K}}$; (3) computing the top k singular vectors $\tilde{\mathbf{V}}_k$ of $\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{K}}\tilde{\mathbf{D}}^{-1/2}$, which are equivalent to the bottom k eigenvectors of $\tilde{\mathbf{L}} = \mathbf{I}_n - \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{K}}\tilde{\mathbf{D}}^{-1/2}$; (4) performing linear k-means over the normalized rows of $\tilde{\mathbf{V}}_k \in \mathbb{R}^{n\times k}$.

To the best of our knowledge, spectral clustering with Nyström approximation does not have a bounded approximation ratio relative to exact spectral clustering. In fact, it seems unlikely that the approximation ratio could be bounded, as there are fundamental problems with the application of the Nyström approximation to the affinity matrix.

- The affinity matrix $\mathbf{K}$ used in spectral clustering must be elementwise nonnegative. However, the Nyström approximation of such a matrix can have numerous negative entries, so $\tilde{\mathbf{K}}$ is, in general, not proper input for the spectral clustering algorithm. In particular, the approximated degree matrix $\tilde{\mathbf{D}} = \mathsf{diag}(\tilde{\mathbf{K}}\mathbf{1}_n)$ may have negative diagonal entries, so $\tilde{\mathbf{D}}^{-1/2}$ is not guaranteed to be a real matrix; such exceptions must be handled heuristically. The approximate asymmetric Laplacian $\tilde{\mathbf{L}} = \mathbf{I}_n - \tilde{\mathbf{K}}\tilde{\mathbf{D}}^{-1}$ does avoid the introduction of complex values; however, the negative entries in $\tilde{\mathbf{D}}^{-1}$ negate whole columns of $\tilde{\mathbf{K}}$, leading to less meaningful negative similarities/distances.

- Even if $\tilde{\mathbf{D}}^{-1/2}$ is real, the matrix $\tilde{\mathbf{L}} = \mathbf{I}_n - \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{K}}\tilde{\mathbf{D}}^{-1/2}$ may not be SPSD, much less a Laplacian matrix. Thus the bottom eigenvectors of $\tilde{\mathbf{L}}$ cannot be viewed as useful coordinates for linear k-means clustering in the same way that the eigenvectors of $\mathbf{L}$ can be.

- Such approximation is also problematic in terms of matrix approximation accuracy. Even when $\tilde{\mathbf{K}}$ approximates $\mathbf{K}$ well, which can be theoretically guaranteed, the approximate Laplacian $\tilde{\mathbf{L}} = \mathbf{I}_n - \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{K}}\tilde{\mathbf{D}}^{-1/2}$ can be far from $\mathbf{L}$. This is because a small perturbation in $\tilde{\mathbf{D}}$ can have an out-sized influence on the eigenvectors of $\tilde{\mathbf{L}}$.

- One may propose to approximate $\mathbf{N} = \mathbf{D}^{-1/2}\mathbf{K}\mathbf{D}^{-1/2}$, rather than $\mathbf{K}$, with a Nyström approximation $\tilde{\mathbf{N}}$; this ensures that the approximate normalized graph Laplacian $\tilde{\mathbf{L}} = \mathbf{I}_n - \tilde{\mathbf{N}}$ is SPSD. However, this approach requires forming the entirety of $\mathbf{K}$ in order to compute the degree matrix $\mathbf{D}$, and thus has quadratic (with n) time and memory costs. Furthermore, although the resulting approximation, $\tilde{\mathbf{L}}$, is SPSD, it is not a graph Laplacian: its off-diagonal entries are not guaranteed to be non-positive, and its smallest eigenvalue may be nonzero.

In summary, spectral clustering using the Nyström approximation (Fowlkes et al., 2004), which has proven to be a useful heuristic, and which is composed of theoretically principled parts, is less principled when viewed in its entirety. Approximate kernel k-means clustering using Nyström approximation is an equivalently efficient, but theoretically more principled alternative.

4.3. Empirical Comparison with Approximate Spectral Clustering using Nyström Approximation

To complement our discussion of the relative merits of the two methods, we empirically compared the performance of our novel method of approximate kernel k-means clustering

Table 3: Summary of the data sets used in our comparisons.

dataset	**#instances** (n)	**#features** (d)	**#clusters** (k)
MNIST (LeCun et al., 1998)	60,000	780	10
Mushrooms (Frank and Asuncion, 2010)	8,124	112	2
PenDigits (Frank and Asuncion, 2010)	7,494	16	10

using the Nyström method with the popular method of approximate spectral clustering using the Nyström method. We used three classification data sets, described in Table 3. The data sets used are available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

Let $\mathbf{a}_1, \ldots, \mathbf{a}_n \in \mathbb{R}^d$ be the input vectors. We take both the affinity matrix for spectral clustering and the kernel matrix for kernel k-means to be the RBF kernel matrix $\mathbf{K} = [\kappa(\mathbf{a}_i, \mathbf{a}_j)]_{ij} \in \mathbb{R}^{n\times n}$, where $\kappa(\mathbf{a}, \mathbf{a}') = \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{a} - \mathbf{a}'\|_2^2\right)$ and σ is the kernel width parameter. We choose σ based on the average interpoint distance in the data sets as

$$\sigma \quad = \quad \beta \cdot \sqrt{\tfrac{1}{n^2}\textstyle\sum_{i=1}^n \sum_{j=1}^n \|\mathbf{a}_i - \mathbf{a}_j\|_2^2}, \tag{5}$$

where we take $\beta = 0.2$, 1, or 5.

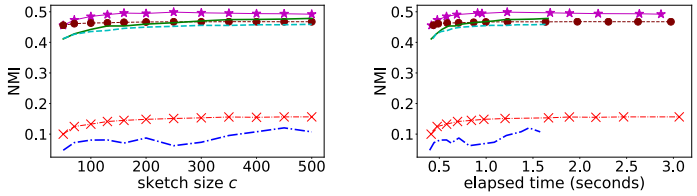
The algorithms under comparison are all implemented in Python 3.5.2. Our implementation of approximate spectral clustering follows the code in (Fowlkes et al., 2004). To compute linear k-means clusterings, we use the function `sklearn.cluster.KMeans` present in the scikit-learn package. Our algorithm for approximate kernel k-means clustering is described in more detail in Section 5.1. We ran the computations on a MacBook Pro with a 2.5GHz Intel Core i7 CPU and 16GB of RAM.

We compare approximate spectral clustering (SC) with approximate kernel k-means clustering (KK), with both using the rank-restricted Nyström method with uniform sampling.[4] We used normalized mutual information (NMI) (Strehl and Ghosh, 2002) to evaluate clustering performance: the NMI falls between 0 (representing no mutual information between the true and approximate clusterings) and 1 (perfect correlation of the two clusterings), so larger NMI indicates better performance. The target dimension s is taken to be k; and, for each method, the sketch size c is varied from $5s$ to $50s$. We record the time cost of the two methods, excluding the time spent on the k-means clustering required in both algorithms.[5] We repeat this procedure 100 times and report the averaged NMI and average elapsed time.

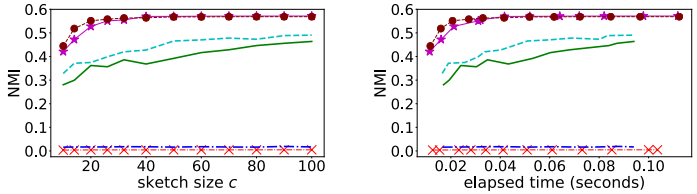
We note that, at small sketch sizes c, exceptions often arise during approximate spectral clustering due to negative entries in the degree matrix. (This is an example, as discussed in Section 4.2, of when approximate spectral clustering heuristics do not perform well.) We discard the trials where such exceptions occur.

Our results are summarized in Figure 3. Figure 3 illustrates the NMI of SC and KK as a function of the sketch size c and as a function of elapsed time for both algorithms. While

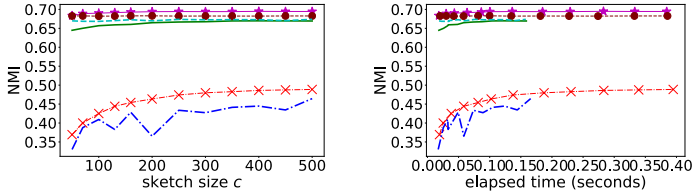
4. Uniform sampling is appropriate for the value of σ used in Eqn. (5); see Gittens and Mahoney (2016) for a detailed discussion of the effect of varying σ.
5. For both SC and KK with Nyström approximation, the extracted feature matrices have dimension $n \times k$, so the k-means clusterings required by both SC and KK have identical cost.



(a) M $n = 60,000$, $d = 780$, $k = 10$.



(b) Mushroom $n = 8,124$, $d = 112$, $k = 2$.



(c) en igits $n = 7,494$, $d = 16$, $k = 10$.

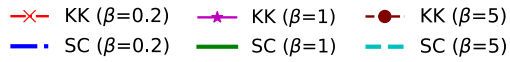


Figure 3: Comparisons between the runtimes and accuracies of approximate spectral clustering (SC) and approximate kernel k-means clustering (KK), both using uniform column sampling Nyström approximation.

Algorithm 2 Approximate Kernel k-Means Clustering using Nyström Approximation.

Input: data set $\mathbf{a}_1, \ldots, \mathbf{a}_n \in \mathbb{R}^d$, number of clusters k, target dimension s ($\geq k$), arbitrary integer ℓ ($> s$), sketch size c ($> \ell$), kernel function κ.
// Step 1: The Nyström Method
Form sketches $\mathbf{C} = \mathbf{KP} \in \mathbb{R}^{n\times c}$ and $\mathbf{W} = \mathbf{P}^T\mathbf{C} \in \mathbb{R}^{c\times c}$, where $\mathbf{K} = [\kappa(\mathbf{a}_i, \mathbf{a}_j)]_{ij} \in \mathbb{R}^{n\times n}$ is the kernel matrix and $\mathbf{P} \in \mathbb{R}^{n\times c}$ is some sketching matrix, e.g., uniform sampling;
Compute a matrix $\mathbf{R} \in \mathbb{R}^{n\times \ell}$ such that $\mathbf{RR}^T = \mathbf{CW}_\ell^{-1}\mathbf{C}^T$;
// Step 2: Dimensionality Reduction
Compute the rank-s truncated SVD $\mathbf{R}_s = \tilde{\mathbf{U}}_s\tilde{\mathbf{\Sigma}}_s\tilde{\mathbf{V}}_s^T$;
Let $\mathbf{B} = \tilde{\mathbf{U}}_s\tilde{\mathbf{\Sigma}}_s (= \mathbf{R}\tilde{\mathbf{V}}_s) \in \mathbb{R}^{n\times s}$;
// Step 3: Linear k-Means Clustering
Perform k-means clustering over the rows of $\mathbf{B}$;
return the clustering results.

there are quantitative differences between the results on the three data sets, the plots all show that KK is more accurate as a function of the sketch size or elapsed time than SC.

5. Single-Machine Medium-Scale Experiments

In this section, we empirically compare the Nyström method and random feature maps (Rahimi and Recht, 2007) for kernel k-means clustering. We conduct experiments on the data listed in Table 3. For the Mushrooms and PenDigits data, we are able to evaluate the objective function value of kernel k-means clustering.

5.1. Single-Machine Implementation of Approximate Kernel k-Means

Our algorithm for approximate kernel k-means clustering comprises three steps: **Nyström approximation**, **dimensionality reduction**, and **linear k-means clustering**. Both the single-machine as well as the distributed variants of the algorithm are governed by three parameters: s, the number of features used in the clustering; ℓ, a regularization parameter; and c, the sketch size. These parameters satisfy $k \leq s < \ell \leq c \ll n$.

1. **Nyström approximation.** Let c be the sketch size and $\mathbf{P} \in \mathbb{R}^{n\times c}$ be a sketching matrix. Let $\mathbf{C} = \mathbf{KP}$ and $\mathbf{W} = \mathbf{P}^T\mathbf{KP} = \mathbf{P}^T\mathbf{C}$. The standard Nyström approximation is $\mathbf{CW}^\dagger\mathbf{C}^T$; small singular values in $\mathbf{W}$ can lead to instability in the Moore-Penrose inverse, so a widely used heuristic is to choose $\ell < c$ and use $\mathbf{CW}_\ell^{-1}\mathbf{C}^T$ instead of the standard Nyström approximation.[6] We set $\ell = \lceil c/2 \rceil$ (arbitrarily). Let $\mathbf{W}_\ell = \mathbf{U}_{\mathbf{W},\ell}\mathbf{\Lambda}_{\mathbf{W},\ell}\mathbf{U}_{\mathbf{W},\ell}^T$ be the truncated SVD of $\mathbf{W}$ and return $\mathbf{R} = \mathbf{CU}_{\mathbf{W},\ell}\mathbf{\Lambda}_{\mathbf{W},\ell}^{-1/2} \in \mathbb{R}^{n\times\ell}$ as the output of the Nyström method.

6. The Nyström approximation $\mathbf{CW}^\dagger\mathbf{C}^T$ is correct in theory, but the Moore-Penrose inverse often causes numerical errors in practice. The Moore-Penrose inverse drops all the zero singular values, however, due to the finite numerical precision, it is difficult to determine whether a singular value, say 10^{-12}, should be zero or not, and this makes the computation unstable: if such a small singular value is believed to be zero, it will be dropped; otherwise, the Moore-Penrose inverse will invert it to obtain a singular value of 10^{12}. Dropping some portion of the smallest singular values is a simple heuristic that avoids this instability. This is why we heuristically use $\mathbf{CW}_\ell^{-1}\mathbf{C}^T$ instead of $\mathbf{CW}^\dagger\mathbf{C}^T$. Currently we do not have theory for this heuristic. Chiu and Demanet (2013) considers the theoretical implications of this regularization heuristic, but their results do not apply to our problem.

2. **Dimensionality reduction.** Let $\tilde{\mathbf{V}}_s \in \mathbb{R}^{\ell \times s}$ contain the dominant s right singular vectors of $\mathbf{R}$. Let $\mathbf{B} = \mathbf{R}\tilde{\mathbf{V}}_s \in \mathbb{R}^{n \times s}$. It can be verified that $\mathbf{B}\mathbf{B}^T = (\mathbf{C}\mathbf{W}_\ell^{-1}\mathbf{C}^T)_s$, which is our desired rank-restricted Nyström approximation.

3. **Linear k-means clustering.** With $\mathbf{B} \in \mathbb{R}^{n \times s}$ at hand, use an arbitrary off-the-shelf linear k-means clustering algorithm to cluster the rows of $\mathbf{B}$.

See Algorithm 2 for the single-machine version of this approximate kernel k-means clustering algorithm. Observe that we can use uniform sampling to form $\mathbf{C}$ and $\mathbf{W}$, and thereby avoid computing most of $\mathbf{K}$.

Let $\mathbf{R} \in \mathbb{R}^{n \times c}$ be the feature matrix computed by random feature maps (RFM). To make the comparison fair, we perform **dimensionality reduction** for RFM (Rahimi and Recht, 2007) in the same way as described in Algorithm 2 to compute $\mathbf{B} = \mathbf{R}\tilde{\mathbf{V}}_s$, and apply linear k-means clustering on the rows of $\mathbf{B}$.

5.2. Comparing Nyström, Random Feature Maps, and Two-Step Method

We empirically compare the clustering performances of kernel approximations formed using Nyström, random feature map (RFM) (Rahimi and Recht, 2007), and the two-step method (Chitta et al., 2011) on the data sets detailed in Table 3.

We use the RBF kernel with width parameter given by (5); Figure 3 indicates that $\beta = 1$ is a good choice for these data sets. We conduct dimensionality reduction for both Nyström and RFM to obtain s-dimensional features, and consider three choices: $s = k$, $s = \lceil \sqrt{ck} \rceil$, and without dimensionality reduction (equivalently, $s = c$).

The quality of the clusterings is quantified using both normalized mutual information (NMI) (Strehl and Ghosh, 2002) and the objective function value:

$$\frac{1}{n} \sum_{i=1}^{k} \sum_{j \in \mathcal{J}_i} \left\| \mathbf{k}_j - \frac{1}{|\mathcal{J}_i|} \sum_{l \in \mathcal{J}_i} \mathbf{k}_l \right\|_2^2, \tag{6}$$

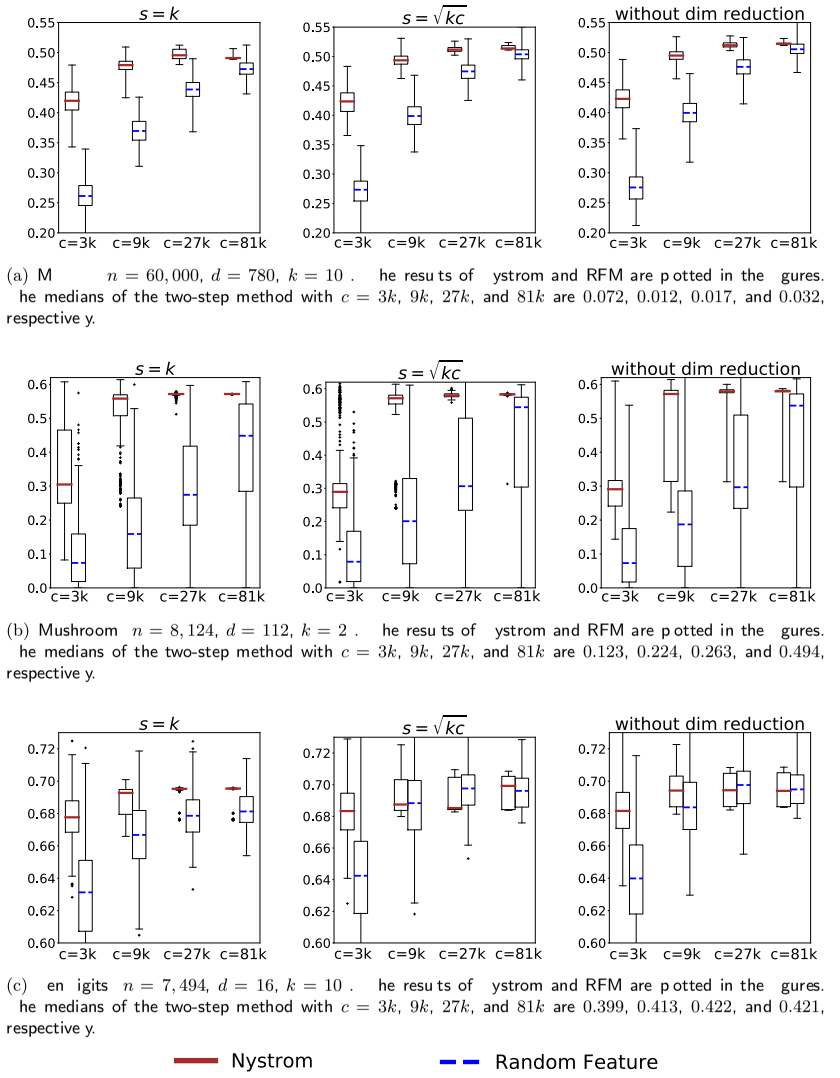
where $\mathbf{k}_1, \cdots, \mathbf{k}_n \in \mathbb{R}^n$ are the columns of the kernel matrix $\mathbf{K}$, and the disjoint sets $\mathcal{J}_1, \cdots, \mathcal{J}_k$ reflect the clustering.

We repeat the experiments 500 times and report the results in Figures 4 and 5. The experiments show that as measured by both NMIs and objective values, the Nyström method outperforms RFM in most cases. Both the Nyström method and RFM are consistently superior to the two-step method of (Chitta et al., 2011), which requires a large sketch size. All the compared methods improve as the sketch size c increases.

Judging from these medium-scale experiments, the target rank s has little impact on the NMI and clustering objective value. This phenomenon is not general; in the large-scale experiments of the next section we see that setting s properly allows one to obtain a better NMI than an over-small or over-large s.

6. Large-Scale Experiments using Distributed Computing

In this section, we empirically study our approximate kernel k-means clustering algorithm on large-scale data. We state a distributed version of the algorithm, implement it in Apache

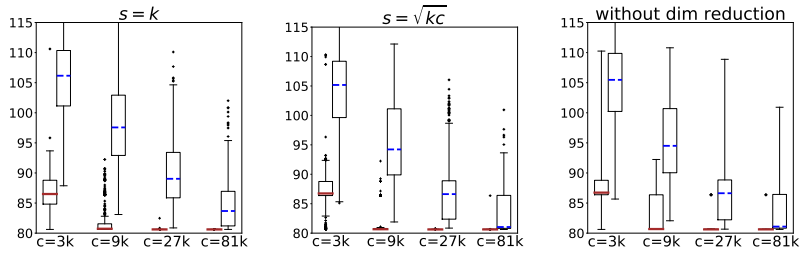


(a) M $n = 60{,}000$, $d = 780$, $k = 10$. he resu ts of ystrom and RFM are p otted in the gures. he medians of the two-step method with $c = 3k$, $9k$, $27k$, and $81k$ are 0.072, 0.012, 0.017, and 0.032, respective y.

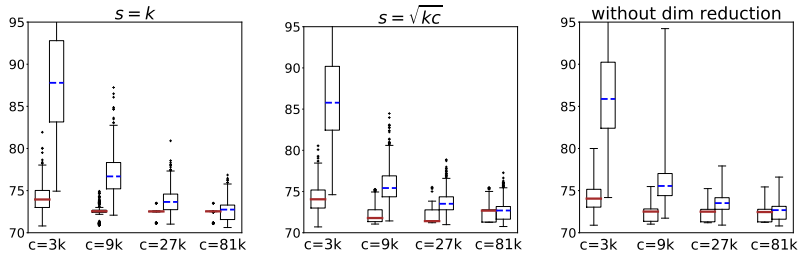
(b) Mushroom $n = 8{,}124$, $d = 112$, $k = 2$. he resu ts of ystrom and RFM are p otted in the gures. he medians of the two-step method with $c = 3k$, $9k$, $27k$, and $81k$ are 0.123, 0.224, 0.263, and 0.494, respective y.

(c) en igits $n = 7{,}494$, $d = 16$, $k = 10$. he resu ts of ystrom and RFM are p otted in the gures. he medians of the two-step method with $c = 3k$, $9k$, $27k$, and $81k$ are 0.399, 0.413, 0.422, and 0.421, respective y.

Figure 4: Quality of approximate kernel k-means clusterings using Nyström, random feature maps (Rahimi and Recht, 2007), and the two-step method (Chitta et al., 2011). (Dimensionality reduction is not applicable to the two-step method). The y-axis reports the normalized mutual information (NMI).



(a) Mushroom $n = 8,124$, $d = 112$, $k = 2$. he resu ts of ystrom and RFM are p otted in the gures. he medians of the objective va ues of the two-step method with $c = 3k$, $9k$, $27k$, and $81k$ are 101.5, 92.8, 89.5, and 85.3, respective y. random partition resu ts in a median objective function va ue of 119.5.



(b) en igits $n = 7,494$, $d = 16$, $k = 10$. he resu ts of ystrom and RFM are p otted in the gures. he medians of the two-step method with $c = 3k$, $9k$, $27k$, and $81k$ are 210.5, 198.2, 193.5, and 193.4, respective y. random partition resu ts in a median objective function va ue of 395.4.



Figure 5: Quality of approximate kernel k-means clusterings obtained using rank-restricted Nyström, RFMs (Rahimi and Recht, 2007), and the two-step method (Chitta et al., 2011). (Dimensionality reduction is not applicable to the two-step method). The y-axis reports the kernel k-means clustering objective function value.

Algorithm 3 Distributed Approximate Kernel k-Means Clustering using Nyström Approximation.

Input: data set $\mathbf{a}_1, \ldots, \mathbf{a}_n \in \mathbb{R}^d$ distributed among m machines, number of clusters k, target dimension s ($\geq k$), arbitrary integer $\ell > s$, sketch size c ($> \ell$), kernel function κ.
// **Step 1: The Nyström Method**
Sample c vectors from $\mathbf{a}_1, \ldots, \mathbf{a}_n$ to form $\mathbf{a}'_1, \ldots, \mathbf{a}'_c$ and send to the driver;
Driver computes $\mathbf{W} = [\kappa(a'_i, a'_j)]_{ij} \in \mathbb{R}^{c \times c}$ and a matrix $\mathbf{Z} \in \mathbb{R}^{c \times \ell}$ satisfying $\mathbf{Z}\mathbf{Z}^T = \mathbf{W}_\ell^{-1}$;
Broadcast $\mathbf{Z}$ and $\mathbf{a}'_1, \ldots, \mathbf{a}'_c$ to all executors;
for all $j \in [n]$ **do**
  Each executor locally computes $\mathbf{c}_j = [\kappa(\mathbf{a}_j, \mathbf{a}'_1); \ldots; \kappa(\mathbf{a}_j, \mathbf{a}'_c)] \in \mathbb{R}^c$ and $\mathbf{r}_j = \mathbf{Z}^T\mathbf{c}_j \in \mathbb{R}^\ell$;
end for
// **Step 2: Dimensionality Reduction**
Let $\tilde{\mathbf{V}}_s$ contain the dominant s right singular vectors of the distributed row matrix $\mathbf{R} = [\mathbf{r}_1, \ldots, \mathbf{r}_n]^T \in \mathbb{R}^{n \times \ell}$ (computed using a distributed truncated SVD algorithm);
Broadcast $\tilde{\mathbf{V}}_s \in \mathbb{R}^{\ell \times s}$ to all executors;
for all $j \in [n]$ **do**
  Each executor locally computes $\mathbf{b}_j = \tilde{\mathbf{V}}_s^T\mathbf{r}_j \in \mathbb{R}^s$;
end for
// **Step 3: Linear k-Means Clustering**
Perform k-means clustering over the Nyström features $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^s$ (using a distributed linear k-means clustering algorithm).
return the clustering results.

Spark[7], and evaluate its performance on NERSC's Cori supercomputer. We investigate the effect of increased parallelism, sketch size c, and target dimension s.

Algorithm 3 is a distributed version of our method described in Section 5.1. Again, we use uniform sampling to form $\mathbf{C}$ and $\mathbf{W}$ to avoid computing most of $\mathbf{K}$. We mainly focus on the **Nyström approximation** step, as the other two steps are well supported by distributed computing systems such as Apache Spark.

6.1. Experimental Setup

We implemented Algorithm 3 in the Apache Spark framework (Zaharia et al., 2010, 2012), using the Scala API. We computed the Nyström approximation using the matrix operations provided by Spark, and invoked the MLlib library for machine learning in Spark (Meng et al., 2016) to perform the dimensionality reduction and linear k-means clustering steps. For the linear k-means clustering, we set the maximum number of iterations to 100.

We ran our experiments on Cori Phase I, a NERSC supercomputer, located at Lawrence Berkeley National Laboratory. Cori Phase I is a Cray XC40 system with 1632 compute nodes, each of which has two 2.3GHz 16-core Haswell processors and 128GB of DRAM. The Cray Aries high-speed interconnect linking the compute nodes is configured in a dragonfly topology.

We used the MNIST8M data set to conduct our empirical evaluations; this data set has $n = 8.1 \times 10^6$ instances, $d = 784$ features, and $k = 10$ clusters. We vary c and s and set $\ell = \frac{c}{2}$. We chose the RBF kernel width parameter according to (5) with $\beta = 1.0$. We use

7. This implementation is available at `https://github.com/wangshusen/SparkKernelKMeans.git`.

Table 4: The mean and standard deviation of the runtimes of Algorithm 3, in seconds, as a function of the number of compute nodes. We report the total runtime as well as the runtimes of the three stages: **Nyström approximation**, **dimensionality reduction (DR)**, and **linear k-means clustering**. The total time is the sum of the three stages and all the Spark overheads, e.g., Spark initialization. We fix $k = 10$ and $c = 400$. In the upper table, we set $s = 20$, and the NMI is 0.400 ± 0.009; in the lower table, we set $s = 80$, and the NMI is 0.410 ± 0.010.

	8 Nodes	16 Nodes	32 Nodes	64 Nodes	128 Nodes
Nyström	2998.6 ± 263.9	1268.8 ± 106.4	665.2 ± 39.6	369.7 ± 126.1	181.4 ± 22.2
DR	37.4 ± 58.3	63.8 ± 72.5	38.9 ± 18.2	87.3 ± 37.8	183.9 ± 89.1
k-means	117.2 ± 132.3	153.1 ± 98.8	119.9 ± 76.9	223.2 ± 88.4	391.0 ± 155.9
Total	3201.5 ± 344.0	1532.0 ± 146.9	867.9 ± 97.5	734.7 ± 152.0	828.2 ± 210.4

	8 Nodes	16 Nodes	32 Nodes	64 Nodes	128 Nodes
Nyström	3008.4 ± 385.6	1312.9 ± 141.2	696.0 ± 111.8	342.5 ± 16.9	197.0 ± 24.1
DR	53.0 ± 41.9	58.7 ± 22.7	94.6 ± 31.8	179.4 ± 55.8	470.6 ± 118.5
k-means	58.2 ± 24.7	80.9 ± 31.8	104.6 ± 28.7	211.3 ± 103.1	501.7 ± 162.9
Total	3168.1 ± 434.1	1492.6 ± 135.3	940.6 ± 139.3	775.9 ± 151.6	1232.4 ± 232.1

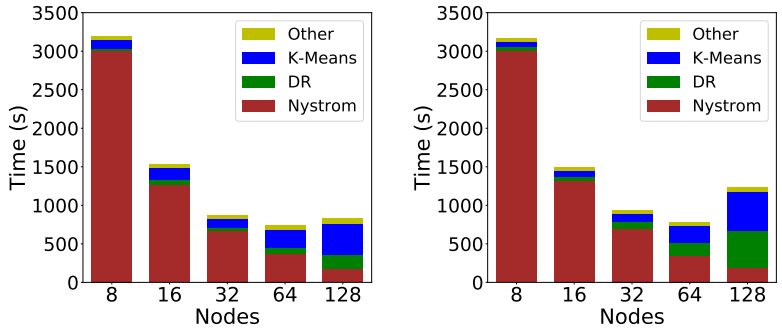
Cori's default setting of Spark configurations. For each setting of parameters, we repeated the experiments 10 times and recorded the NMIs and elapsed time.

6.2. Effect of Increased Parallelism

We varied the number of nodes to test the impact of increased parallelism on each of the three steps in Algorithm 3. We set the sketch size to $c = 400$ and the target dimension to $s = 20$ or 80. Table 4 reports the mean and standard deviation of the elapsed times. As a reference point, using 32 nodes, our algorithm takes 15 minutes on average to group the 8.1 million input instances into 10 clusters. We also plot the elapsed times in Figure 6.

The **Nyström method** scales very well with the increase of nodes: its elapsed time is inversely proportional to the number of nodes. This is because the **Nyström method** requires only 2 rounds of communications; the elapsed time is spent mostly on computation on the executors.

Dimensionality reduction (DR) and **linear k-means clustering** are highly iterative and thus have high latency, incur straggler delays, and have large scheduler and communication overheads. See (Gittens et al., 2016) for an in-depth discussion of the performance concerns when using Spark for distributed matrix computations. As the number of nodes increases from 32 to 64 and higher, **DR** and **k-means** exhibit anti-scaling behavior: as the number of nodes goes up, their runtimes increase rather than decrease. For **DR**, the input dimension is $\ell = \frac{c}{2} = 200$, and the target dimension is $s = 20$ or 80; while for **k-means**, the input dimension is $s = 20$ or 80. Clearly, the issue here is not that **DR** and **k-means** are computationally intensive. Instead, as the number of nodes increase, although the per-node computational time decreases, the Spark communications overheads are increasing, making **DR** and **k-means** less performant. Such



(a) $k = 10$, $s = 20$, $c = 400$. (b) $k = 10$, $s = 80$, $c = 400$.

Figure 6: The mean of the runtimes of Algorithm 3, in seconds, as a function of the number of compute nodes.

behavior has been characterized and studied in other Spark implementations of linear algebra algorithms (Gittens et al., 2016).

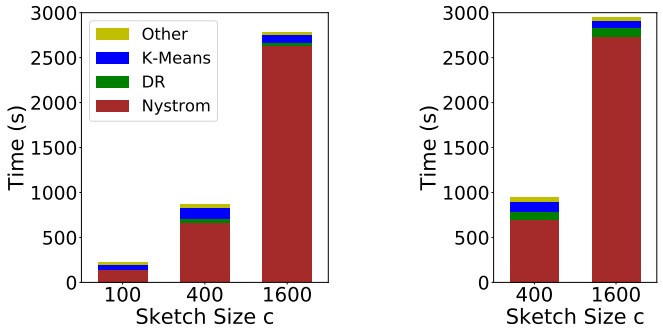
Table 5: The NMI and elapsed time (seconds) for varying sketch sizes c. Here "T" denotes the elapsed time. We use 32 nodes. In the upper table, we set $s = 20$; in the lower table, we set $s = 80$.

	$c = 100$	$c = 400$	$c = 1,600$
NMI	0.3833 ± 0.0117	0.3975 ± 0.0112	0.4069 ± 0.0001
T(Nyström)	138.5 ± 62.1	665.2 ± 39.5	2634.5 ± 94.4
T(DR)	1.2 ± 0.5	38.9 ± 18.2	31.3 ± 6.6
T(k-Means)	60.9 ± 28.7	119.9 ± 76.9	86.9 ± 27.3
T(Total)	227.4 ± 101.3	867.9 ± 97.5	2785.4 ± 83.2

	$c = 400$	$c = 1600$
NMI	0.4101 ± 0.0101	0.4233 ± 0.0131
T(Nyström)	696.0 ± 111.8	2728.1 ± 190.3
T(DR)	94.6 ± 31.8	97.9 ± 20.6
T(k-Means)	104.6 ± 28.7	88.0 ± 23.9
T(Total)	940.6 ± 139.3	2952.2 ± 174.5

6.3. Effect of Sketch Size c

We executed our Spark implementation using 32 compute nodes, setting $s = 20$ or 80, and varying the target dimension c. Table 5 reports the observed normalized mutual information



(a) $k = 10$ and $s = 20$.

(b) $k = 10$ and $s = 80$.

Figure 7: The mean of the runtimes of Algorithm 3, in seconds, as a function of the sketch size c.

(NMI) (Strehl and Ghosh, 2002) and elapsed times. As predicted by our theory, for fixed k and s, larger c always leads to better performance.

In Figure 7 we plot the runtime as a function of the sketch size c. The plots shows the significant influence of c on the running time. In Figure 7, the runtime of Nyström grows superlinearly with c. According to our analysis, the computational and communication costs of Nyström are both superlinear in c. Therefore, the user should keep in mind this trade-off between the clustering performance and the computational cost.

6.4. Effect of Target Dimension s

We executed our Spark implementation using 32 computational nodes, fixing $k = 10$ and $c = 1600$ and varying the target dimension s. Table 6 reports the observed NMIs and elapsed times. For fixed k and c, a moderately large s leads to better clustering performance. However, as s grows, $\frac{k}{s}$ decreases but $\frac{s}{c}$ increases, so as predicted in Remark 3, the clustering performance is nonmonotonic in s. Indeed, as s grows from 80 to 160, the NMI deteriorates. In practice, for fixed k and c, one should set s moderately large, but not over-large.

In Figure 8 we plot the runtime as a function of the target dimension s. Note that s does not affect the time cost of the Nyström method. As s grows, the time costs of dimensionality reduction and linear k-means clustering both increase. This implies that a moderate s is good for computational purpose. Previously, in Figure 2, we plot the NMI as a function of s while fixing k and c. Figure 2 shows that a proper s leads to better NMI than an over-large or over-small s. In sum, setting s properly has both computational and accuracy benefits, which corroborates our theory.

Table 6: The NMI and elapsed time (seconds) for varying target dimensions s. Here "T" denotes the elapsed time. We use 32 nodes and fix $c = 1600$.

	$s = 10$	$s = 20$	$s = 40$	$s = 80$	$s = 160$	$s = 320$
NMI	0.3852 ± 0.0003	0.4069 ± 0.0001	0.4130 ± 0.0062	0.4233 ± 0.0131	0.4086 ± 0.0125	0.4099 ± 0.0098
T(Nyström)	2702.2 ± 131.6	2634.5 ± 94.4	2815.3 ± 403.6	2728.1 ± 190.3	2849.9 ± 511.2	2757.1 ± 271.9
T(DR)	21.1 ± 3.2	31.3 ± 6.6	57.5 ± 9.2	97.9 ± 20.6	168.2 ± 29.0	570.5 ± 151.0
T(k-Means)	84.0 ± 6.4	86.9 ± 27.3	90.6 ± 9.0	88.0 ± 23.9	99.8 ± 12.2	169.3 ± 64.2
T(Total)	2840.9 ± 132.0	2785.4 ± 83.2	3003.4 ± 413.7	2952.2 ± 174.5	3157.5 ± 549.2	3537.5 ± 256.3

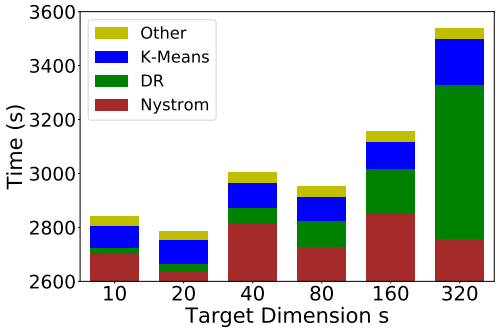


Figure 8: The mean of the runtimes of Algorithm 3, in seconds, as a function of the target dimension s. We use 32 nodes and fix $k = 10$ and $c = 1600$.

7. Conclusion

We provided principled algorithms for computing approximate kernel k-means clusterings. In particular, we showed that the combination of linear k-means with rank-restricted Nyström approximation is theoretically sound, practically useful, and scalable to large data sets. This should be contrasted with approximate spectral clustering using Nyström approximation. Although the latter is a widely-used approach to scalable non-linear clustering, it has theoretical deficiencies and practical limitations. Experiments demonstrated that approximate kernel k-means clustering using the rank-restricted Nyström approximation consistently outperforms approximate spectral clustering.

Our analysis uses the concept of projection-cost preservation and builds upon the existing theory of randomized linear algebra. Our main result is a $1+\epsilon$ approximation ratio guarantee for kernel k-means clustering: when $s = \frac{k}{\epsilon}$ Nyström features are used, a $1 + \mathcal{O}(\epsilon)$ approximation ratio is guaranteed with high probability. As an intermediate theoretical result of independent interest, we introduced a novel rank-restricted Nyström approximation and proved that it gives a $1+\epsilon$ relative-error low-rank approximation guarantee in the trace norm.

To complement our theory, we investigated the performance of the rank-restricted Nyström approximation when applied to kernel k-means clustering. We produced an Apache Spark implementation of a distributed version of our approximate kernel k-means algorithm, and applied it to cluster 8.1 million vectors; the results demonstrate the usefulness and simplicity of kernel k-means with rank-restricted Nyström approximation for clustering moderately large-scale data sets. Combined with recent work on user-friendly frameworks for distributed computing systems (Zaharia et al., 2010, 2012), large-scale machine learning (Meng et al., 2016), and large-scale randomized linear algebra (Gittens et al., 2016), our results suggest the use of kernel k-means with rank-restricted Nyström approximations for simple, computationally efficient, and theoretically principled clustering of large-scale data sets.

Acknowledgments

We thank the authors of Musco and Musco (2017) for bringing to our attention their results on approximate kernel k-means clustering through the use of ridge leverage score Nyström approximations, the authors of Tropp et al. (2017) for informing us of their contemporaneous guarantees on the approximation error of rank-restricted Nyström approximations, and the anonymous reviewers for their helpful suggestions. We would like to acknowledge ARO, DARPA, NSF, and ONR for providing partial support of this work.

Appendix A. Proof of Theorem 1

In this section, we will provide a proof of Theorem 1, our main quality-of-approximation result for rank-restricted Nyström approximation. We will start, in Section A.1, by establishing three technical lemmas; then, in Section A.2, we will establish an important structural result on the rank-restricted Nyström approximation as well as key approximate matrix multiplication properties for low-rank matrix approximation; and finally, in Section A.3, we will use these results to prove Theorem 1.

A.1. Technical Lemmas

Lemma 5 is a very well known result. See (Boutsidis et al., 2014; Zhang, 2015). Here we offer a simplified proof.

Lemma 5 *Let $\mathbf{A} \in \mathbb{R}^{n\times m}$ be any matrix and $\mathbf{U} \in \mathbb{R}^{n\times c}$ have orthonormal columns. Let s be any positive integer no greater than c. Then*

$$(\mathbf{U}^T\mathbf{A})_s \;=\; \operatorname*{argmin}_{\operatorname{rank}(\mathbf{Z})\le s} \big\|\mathbf{A}-\mathbf{U}\mathbf{Z}\big\|_F^2.$$

Proof Let $\mathbf{U}^\perp \in \mathbb{R}^{n\times(n-c)}$ be the orthogonal complement of $\mathbf{U}$. It holds that

$$\begin{aligned}
\big\|\mathbf{A}-\mathbf{U}\mathbf{Z}\big\|_F^2 &= \left\|\mathbf{A} - \begin{bmatrix}\mathbf{U} & \mathbf{U}^\perp\end{bmatrix}\begin{bmatrix}\mathbf{Z}\\ \mathbf{0}\end{bmatrix}\right\|_F^2\\
&= \left\|\begin{bmatrix}\mathbf{U}^T\\ (\mathbf{U}^\perp)^T\end{bmatrix}\mathbf{A} - \begin{bmatrix}\mathbf{Z}\\ \mathbf{0}\end{bmatrix}\right\|_F^2\\
&= \left\|\begin{bmatrix}\mathbf{U}^T\mathbf{A}-\mathbf{Z}\\ (\mathbf{U}^\perp)^T\mathbf{A}\end{bmatrix}\right\|_F^2\\
&= \big\|\mathbf{U}^T\mathbf{A}-\mathbf{Z}\big\|_F^2 + \big\|(\mathbf{U}^\perp)^T\mathbf{A}\big\|_F^2,
\end{aligned}$$

where the second equality follows from the unitary invariance of the Frobenius norm. Then

$$\operatorname*{argmin}_{\operatorname{rank}(\mathbf{Z})\le s} \big\|\mathbf{A}-\mathbf{U}\mathbf{Z}\big\|_F^2 \;=\; \operatorname*{argmin}_{\operatorname{rank}(\mathbf{Z})\le s} \big\|\mathbf{U}^T\mathbf{A}-\mathbf{Z}\big\|_F^2 \;=\; (\mathbf{U}^T\mathbf{A})_s,$$

by which the lemma follows. ■

Lemma 6 is a new result established in this work. The lemma plays an important role in analyzing the rank-restricted Nyström method.

Lemma 6 *Let $\mathbf{A} \in \mathbb{R}^{n\times m}$ be any matrix and $\mathbf{U} \in \mathbb{R}^{n\times c}$ has orthonormal columns. Let $s \le c$ be arbitrary integer. Let $\mathbf{Q} \in \mathbb{R}^{n\times s}$ be the orthonormal bases of the rank s matrix $\mathbf{U}(\mathbf{U}^T\mathbf{A})_s \in \mathbb{R}^{n\times m}$. Then $\mathbf{Q}\mathbf{Q}^T\mathbf{A} = \mathbf{U}(\mathbf{U}^T\mathbf{A})_s$.*

Proof Let $\mathbf{Z}^\star = \mathbf{Q}^T\mathbf{U}(\mathbf{U}^T\mathbf{A})_s \in \mathbb{R}^{s\times m}$. By the definition of $\mathbf{Q}$, we have

$$\mathbf{Q}\mathbf{Q}^T\mathbf{U}(\mathbf{U}^T\mathbf{A})_s \;=\; \mathbf{U}(\mathbf{U}^T\mathbf{A})_s. \tag{7}$$

It holds that

$$\big\|\mathbf{U}(\mathbf{U}^T\mathbf{A})_s-\mathbf{A}\big\|_F^2 \;=\; \min_{\operatorname{rank}(\mathbf{X})\le s}\big\|\mathbf{U}\mathbf{X}-\mathbf{A}\big\|_F^2 \;\le\; \min_{\operatorname{rank}(\mathbf{Z})\le s}\big\|\mathbf{Q}\mathbf{Z}-\mathbf{A}\big\|_F^2 \;\le\; \big\|\mathbf{Q}\mathbf{Z}^\star-\mathbf{A}\big\|_F^2,$$

where the equality follows from Lemma 5; the former inequality follows from that $\operatorname{range}(\mathbf{Q}) \subset \operatorname{range}(\mathbf{U})$. Because $\mathbf{U}(\mathbf{U}^T\mathbf{A})_s = \mathbf{Q}\mathbf{Z}^\star$ by (7), the lefthand and righthand sides are equal, and thereby

$$\min_{\operatorname{rank}(\mathbf{Z})\le s}\big\|\mathbf{Q}\mathbf{Z}-\mathbf{A}\big\|_F^2 \;=\; \big\|\mathbf{Q}\mathbf{Z}^\star-\mathbf{A}\big\|_F^2.$$

Since $\mathbf{Z}$ is $s\times m$, it follows that

$$\min_{\mathbf{Z}}\big\|\mathbf{Q}\mathbf{Z}-\mathbf{A}\big\|_F^2 \;=\; \min_{\operatorname{rank}(\mathbf{Z})\le s}\big\|\mathbf{Q}\mathbf{Z}-\mathbf{A}\big\|_F^2 \;=\; \big\|\mathbf{Q}\mathbf{Z}^\star-\mathbf{A}\big\|_F^2. \tag{8}$$

Let $\mathbf{Q}^\perp \in \mathbb{R}^{n\times(n-s)}$ be the orthogonal complement of $\mathbf{Q}$. It holds that

$$\begin{aligned}\left\|\mathbf{A}-\mathbf{QZ}\right\|_F^2 &= \left\|\mathbf{A}-\begin{bmatrix}\mathbf{Q} & \mathbf{Q}^\perp\end{bmatrix}\begin{bmatrix}\mathbf{Z}\\ \mathbf{0}\end{bmatrix}\right\|_F^2\\ &= \left\|\begin{bmatrix}\mathbf{Q}^T\\ (\mathbf{Q}^\perp)^T\end{bmatrix}\mathbf{A}-\begin{bmatrix}\mathbf{Z}\\ \mathbf{0}\end{bmatrix}\right\|_F^2\\ &= \left\|\begin{bmatrix}\mathbf{Q}^T\mathbf{A}-\mathbf{Z}\\ (\mathbf{Q}^\perp)^T\mathbf{A}\end{bmatrix}\right\|_F^2\\ &= \left\|\mathbf{Q}^T\mathbf{A}-\mathbf{Z}\right\|_F^2+\left\|(\mathbf{Q}^\perp)^T\mathbf{A}\right\|_F^2,\end{aligned}$$

where the second equality follows from the unitary invariance of the Frobenius norm. Obviously, $\mathbf{Z}=\mathbf{Q}^T\mathbf{A}$ is the unique minimizer of $\min_{\mathbf{Z}}\|\mathbf{QZ}-\mathbf{A}\|_F^2$; otherwise $\mathbf{Q}^T\mathbf{A}-\mathbf{Z}$ in the righthand side is non-zero, making the objective function increase.

On the one hand, Eqn. (8) shows that $\mathbf{Z}^\star=\mathbf{Q}^T\mathbf{U}(\mathbf{U}^T\mathbf{A})_s\in\mathbb{R}^{s\times m}$ is one minimizer of $\min_{\mathbf{Z}}\|\mathbf{QZ}-\mathbf{A}\|_F^2$. On the other hand, we have that $\mathbf{Z}=\mathbf{Q}^T\mathbf{A}$ is the unique minimizer of $\min_{\mathbf{Z}}\|\mathbf{QZ}-\mathbf{A}\|_F^2$. By the uniqueness, we have

$$\mathbf{Z}^\star \;=\; \mathbf{Q}^T\mathbf{U}(\mathbf{U}^T\mathbf{A})_s \;=\; \mathbf{Q}^T\mathbf{A}.$$

It follows that

$$\mathbf{QQ}^T\mathbf{U}(\mathbf{U}^T\mathbf{A})_s \;=\; \mathbf{QQ}^T\mathbf{A}.$$

The lemma follows from (7) and the above equality. ■

Lemma 7 is a new result established by this work. Let $\mathbf{A}$ be any symmetric matrix. Lemma 7 analyzes the power scheme: using $\mathbf{A}^t\mathbf{P}$, instead of $\mathbf{AP}$, as a sketch of $\mathbf{A}$. The lemma shows that the power scheme leads to an improvement of $\left(\frac{\sigma_{s+1}^2}{\sigma_s^2}\right)^{t-1}$, where σ_i is the i-th biggest singular value of $\mathbf{A}$. Lemma 7 with $t=1$ is identical to (Boutsidis et al., 2014, Lemma 9). The power scheme has been studied by Gittens and Mahoney (2016); Halko et al. (2011); Woodruff (2014); their results do not allow the rank restriction $\mathrm{rank}(\mathbf{X})\le s$. To prove Theorem 1, we only need $t=1$; we will use Lemma 7 to extend Theorem 1 to the power method.

Lemma 7 *Let $\mathbf{A}\in\mathbb{R}^{n\times n}$ be any SPSD matrix and $\mathbf{A}_s=\mathbf{V}_s\mathbf{\Sigma}_s\mathbf{V}_s^T$ be the truncated SVD. Let $\mathbf{P}\in\mathbb{R}^{n\times c}$ satisfy that $\mathbf{P}^T\mathbf{V}_s\in\mathbb{R}^{c\times s}$ has full column rank. Let $t\ge 1$ be any integer and $\mathbf{C}=\mathbf{A}^t\mathbf{P}$. Then for $\xi=2$ or F,*

$$\min_{\mathrm{rank}(\mathbf{X})\le s}\left\|\mathbf{A}-\mathbf{CX}\right\|_\xi^2 \;\le\; \left\|\mathbf{A}-\mathbf{A}_s\right\|_\xi^2+\left(\tfrac{\sigma_{s+1}^2}{\sigma_s^2}\right)^{t-1}\left\|(\mathbf{A}-\mathbf{A}_s)\mathbf{P}(\mathbf{V}_s^T\mathbf{P})^\dagger\right\|_\xi^2,$$

where σ_i is the i-th largest singular value of $\mathbf{A}$.

Proof We construct the rank s matrix $\tilde{\mathbf{A}}_s=\mathbf{C}(\mathbf{V}_s^T\mathbf{P})^\dagger\mathbf{\Sigma}_s^{1-t}\mathbf{V}_s^T$ and use it to facilitate our proof. Since $\tilde{\mathbf{A}}_s$ has rank s, and its column space is in range$(\mathbf{C})$, it holds that

$$\begin{aligned}&\min_{\mathrm{rank}(\mathbf{X})\le s}\left\|\mathbf{A}-\mathbf{CX}\right\|_\xi^2 \;\le\; \left\|\mathbf{A}-\tilde{\mathbf{A}}_s\right\|_\xi^2\\ &=\; \left\|(\mathbf{A}-\mathbf{A}_s)+(\mathbf{A}_s-\tilde{\mathbf{A}}_s)\right\|_\xi^2 \;\le\; \left\|\mathbf{A}-\mathbf{A}_s\right\|_\xi^2+\left\|\mathbf{A}_s-\tilde{\mathbf{A}}_s\right\|_\xi^2.\end{aligned}$$

The latter inequality follows from that the row space of $\mathbf{A}_s-\tilde{\mathbf{A}}_s$ is in range$(\mathbf{V}_s)$, that $(\mathbf{A}-\mathbf{A}_s)\mathbf{V}_s=\mathbf{0}$, and the matrix Pythagorean theorem. By the assumption $\mathrm{rank}(\mathbf{V}_s^T\mathbf{P})=s$, it holds that $(\mathbf{V}_s^T\mathbf{P})(\mathbf{V}_s^T\mathbf{P})^\dagger=\mathbf{I}_s$. We have

$$\begin{aligned}\left\|\mathbf{A}_s-\tilde{\mathbf{A}}_s\right\|_\xi^2 &= \left\|\mathbf{A}_s-(\mathbf{A}_s^t+\mathbf{A}^t-\mathbf{A}_s^t)\mathbf{P}(\mathbf{V}_s^T\mathbf{P})^\dagger\mathbf{\Sigma}_s^{1-t}\mathbf{V}_s^T\right\|_\xi^2\\ &= \left\|\mathbf{A}_s-\mathbf{V}_s\mathbf{\Sigma}_s^t\underbrace{(\mathbf{V}_s^T\mathbf{P})}_{s\times c}\underbrace{(\mathbf{V}_s^T\mathbf{P})^\dagger}_{c\times s}\mathbf{\Sigma}_s^{1-t}\mathbf{V}_s^T-(\mathbf{A}^t-\mathbf{A}_s^t)\mathbf{P}(\mathbf{V}_s^T\mathbf{P})^\dagger\mathbf{\Sigma}_s^{1-t}\mathbf{V}_s^T\right\|_\xi^2\\ &= \left\|\mathbf{A}_s-\mathbf{V}_s\mathbf{\Sigma}_s\mathbf{V}_s^T-(\mathbf{A}^t-\mathbf{A}_s^t)\mathbf{P}(\mathbf{V}_s^T\mathbf{P})^\dagger\mathbf{\Sigma}_s^{1-t}\mathbf{V}_s^T\right\|_\xi^2\\ &= \left\|(\mathbf{A}^t-\mathbf{A}_s^t)\mathbf{P}(\mathbf{V}_s^T\mathbf{P})^\dagger\mathbf{\Sigma}_s^{1-t}\mathbf{V}_s^T\right\|_\xi^2\\ &\le \left\|(\mathbf{A}-\mathbf{A}_s)^{t-1}\right\|_2^2\left\|\mathbf{\Sigma}_s^{1-t}\right\|_2^2\left\|(\mathbf{A}-\mathbf{A}_s)\mathbf{P}(\mathbf{V}_s^T\mathbf{P})^\dagger\right\|_\xi^2\\ &= \left(\tfrac{\sigma_{s+1}^2}{\sigma_s^2}\right)^{t-1}\left\|(\mathbf{A}-\mathbf{A}_s)\mathbf{P}(\mathbf{V}_s^T\mathbf{P})^\dagger\right\|_\xi^2,\end{aligned}$$

by which the lemma follows. ■

A.2. Structural Result and Approximate Matrix Multiplication Properties

Lemma 8 is a structural result of independent interest that is important for establishing Theorem 1. By *structural result*, we mean that it is a linear algebraic result that holds for any matrix $\mathbf{P}$, and in particular for any matrix $\mathbf{P}$ that is a randomized sketching matrix. In particular, given this structural result, randomness in an algorithm enters only via $\mathbf{P}$. The importance of establishing such structural results within randomized linear algebra has been highlighted previously (Drineas et al., 2011; Mahoney, 2011; Gittens and Mahoney, 2016; Mahoney and Drineas, 2016; Wang et al., 2017).

Lemma 8 *Let $\mathbf{K}\in\mathbb{R}^{n\times n}$ be any SPSD matrix, $\mathbf{P}\in\mathbb{R}^{n\times c}$ be any matrix, and $s\le c$ be any positive integer. Let $\mathbf{D}=\mathbf{K}^{1/2}\mathbf{P}\in\mathbb{R}^{n\times c}$, $\rho=\mathrm{rank}(\mathbf{D})\le c$, and $\mathbf{U}\in\mathbb{R}^{n\times\rho}$ be the left singular vectors of $\mathbf{D}$. Let $\mathbf{Q}\in\mathbb{R}^{n\times s}$ be any orthonormal bases of $\mathbf{U}(\mathbf{U}^T\mathbf{K}^{1/2})_s$. It holds that*

$$\begin{aligned}\left(\mathbf{CW}^\dagger\mathbf{C}^T\right)_s &= \mathbf{K}^{1/2}\mathbf{QQ}^T\mathbf{K}^{1/2},\\ \left\|\mathbf{K}-\left(\mathbf{CW}^\dagger\mathbf{C}^T\right)_s\right\|_* &= \min_{\mathrm{rank}(\mathbf{Z})\le s}\left\|\mathbf{K}^{1/2}-(\mathbf{K}^{1/2}\mathbf{P})\mathbf{Z}\right\|_F^2.\end{aligned}$$

Proof Let $\mathbf{D}=\mathbf{K}^{1/2}\mathbf{P}=\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\in\mathbb{R}^{n\times c}$ be the SVD of $\mathbf{D}$. It holds that $\mathbf{C}=\mathbf{KP}=\mathbf{K}^{1/2}\mathbf{D}$ and $\mathbf{W}=\mathbf{P}^T\mathbf{KP}=\mathbf{D}^T\mathbf{D}$. It follows the SVD of $\mathbf{D}$ that

$$\mathbf{C}(\mathbf{W}^\dagger)^{1/2} \;=\; \mathbf{K}^{1/2}\mathbf{D}\left((\mathbf{D}^T\mathbf{D})^\dagger\right)^{1/2} \;=\; \mathbf{K}^{1/2}(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)\left(\mathbf{V}\mathbf{\Sigma}^{-2}\mathbf{V}^T\right)^{1/2} \;=\; \mathbf{K}^{1/2}\mathbf{U}\mathbf{V}^T.$$

We need the following lemma to prove Lemma 8. The following lemma is not hard to prove.

Lemma 9 *Let the integers n,c,ρ,s satisfy $n\ge c\ge\rho\ge s$. Let $\mathbf{A}\in\mathbb{R}^{n\times\rho}$ be any matrix with rank at least s and $\mathbf{V}\in\mathbb{R}^{c\times\rho}$ has orthonormal columns. Then $(\mathbf{AV}^T)_s=\mathbf{A}_s\mathbf{V}^T$.*

If $\mathrm{rank}(\mathbf{K}^{1/2}\mathbf{U}) < s$, then obviously $\mathbf{K}^{1/2}\mathbf{U}\mathbf{V}^T = (\mathbf{K}^{1/2}\mathbf{U}\mathbf{V}^T)_s = (\mathbf{K}^{1/2}\mathbf{U})_s\mathbf{V}^T$. We thus assume $\mathrm{rank}(\mathbf{K}^{1/2}\mathbf{U}\mathbf{V}^T) \geq s$ and can apply the above lemma to show $(\mathbf{K}^{1/2}\mathbf{U}\mathbf{V}^T)_s = (\mathbf{K}^{1/2}\mathbf{U})_s\mathbf{V}^T$. In either case, we have

$$(\mathbf{K}^{1/2}\mathbf{U}\mathbf{V}^T)_s \;=\; (\mathbf{K}^{1/2}\mathbf{U})_s\mathbf{V}^T.$$

It follows from (9) that $\big(\mathbf{C}(\mathbf{W}^\dagger)^{1/2}\big)_s = (\mathbf{K}^{1/2}\mathbf{U})_s\mathbf{V}^T$. Thus

$$\big(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\big)_s \;=\; (\mathbf{K}^{1/2}\mathbf{U})_s(\mathbf{U}^T\mathbf{K}^{1/2})_s \;=\; (\mathbf{K}^{1/2}\mathbf{U})_s\mathbf{U}^T\mathbf{U}(\mathbf{U}^T\mathbf{K}^{1/2})_s$$

Let $\mathbf{Q} \in \mathbb{R}^{n\times s}$ be the orthonormal bases of the rank s matrix $\mathbf{U}(\mathbf{U}^T\mathbf{K}^{1/2})_s$. Lemma 6 ensures that

$$\mathbf{Q}\mathbf{Q}^T\mathbf{K}^{1/2} \;=\; \mathbf{U}(\mathbf{U}^T\mathbf{K}^{1/2})_s.$$

It follows that

$$\big(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\big)_s \;=\; \mathbf{K}^{1/2}\mathbf{Q}\mathbf{Q}^T\mathbf{K}^{1/2}, \tag{9}$$

by which the former conclusion of the lemma follows.

It is well known that $\|\mathbf{Y}\|_* = \mathrm{tr}(\mathbf{Y})$ for any SPSD matrix $\mathbf{Y}$ and that $\mathrm{tr}(\mathbf{A}^T\mathbf{A}) = \|\mathbf{A}\|_F^2$ for any matrix $\mathbf{A}$. It follows from (9) that

$$\begin{aligned}
&\big\|\mathbf{K} - \big(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\big)_s\big\|_* \;=\; \big\|\mathbf{K}^{1/2}\big(\mathbf{I}_n - \mathbf{Q}\mathbf{Q}^T\big)\mathbf{K}^{1/2}\big\|_* \\
&= \mathrm{tr}\Big(\mathbf{K}^{1/2}\big(\mathbf{I}_n - \mathbf{Q}\mathbf{Q}^T\big)\big(\mathbf{I}_n - \mathbf{Q}\mathbf{Q}^T\big)\mathbf{K}^{1/2}\Big) \\
&= \big\|\big(\mathbf{I}_n - \mathbf{Q}\mathbf{Q}^T\big)\mathbf{K}^{1/2}\big\|_F^2 \;=\; \big\|\mathbf{K}^{1/2} - \mathbf{U}(\mathbf{U}^T\mathbf{K}^{1/2})_s\big\|_F^2;
\end{aligned}$$

here the second equality holds because $\mathbf{K}^{1/2}(\mathbf{I}_n - \mathbf{Q}\mathbf{Q}^T)\mathbf{K}^{1/2}$ is SPSD and $(\mathbf{I}_n - \mathbf{Q}\mathbf{Q}^T)$ is orthogonal projection matrix; the last equality follows from Lemma 6. It follows from Lemma 5 that

$$\begin{aligned}
&\big\|\mathbf{K} - \big(\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T\big)_s\big\|_* \;=\; \big\|\mathbf{K}^{1/2} - \mathbf{U}(\mathbf{U}^T\mathbf{K}^{1/2})_s\big\|_F^2 \\
&= \min_{\mathrm{rank}(\mathbf{X})\leq s}\big\|\mathbf{K}^{1/2} - \mathbf{U}\mathbf{X}\big\|_F^2 \;=\; \min_{\mathrm{rank}(\mathbf{Z})\leq s}\big\|\mathbf{K}^{1/2} - \mathbf{D}\mathbf{Z}\big\|_F^2,
\end{aligned}$$

by which the latter conclusion of the lemma follows. ■

Lemma 10 formally defines the key *approximate matrix multiplication properties* (subspace embedding property and approximate orthogonality property) which—when sketching dimensions are chosen appropriately (see Table 7)—are shared by the uniform sampling, leverage score sampling, Gaussian projection, SRHT, and CountSketch sketching methods. Establishing approximate matrix multiplication bounds is the key step in proving many approximate regression and low-rank approximation results (Mahoney, 2011), and this lemma will be used crucually in the proof of Theorem 1.

Lemma 10 (Key Approximate Matrix Multiplication Properties for a Sketch) *Let $\eta, \epsilon, \delta_1, \delta_2 \in (0,1)$ be fixed parameters. Fix $\mathbf{Y} \in \mathbb{R}^{n\times d}$ and let $\mathbf{V} \in \mathbb{R}^{n\times s}$ have orthonormal columns. Let $\mathbf{P} \in \mathbb{R}^{n\times c}$ be one of the sketching methods listed in Table 7. When c is larger than the quantity in the middle column of the corresponding row of Table 7, the subspace embedding property*

$$\big\|\mathbf{V}^T\mathbf{P}\mathbf{P}^T\mathbf{V} - \mathbf{I}_s\big\|_2 \;\leq\; \eta$$

holds with probability at least $1-\delta_1$. When c is larger than the quantity in the right-hand column of the corresponding row of Table 7, the matrix multiplication property

$$\big\|\mathbf{V}^T\mathbf{P}\mathbf{P}^T\mathbf{Y} - \mathbf{V}^T\mathbf{Y}\big\|_F^2 \;\leq\; \epsilon\|\mathbf{Y}\|_F^2$$

holds with probability at least $1-\delta_2$.

Table 7: Sufficient sketching dimensions for the sketching properties of Lemma 10 to hold. The leverage score sampling referred to here is with respect to the leverage scores of the matrix $\mathbf{V}$ described in Lemma 10. Similarly, μ is the row coherence of $\mathbf{V}$.

Sketching	Subspace Embedding	Matrix Multiplication
Leverage Score Sampling	$\frac{s}{\eta^2}\log\frac{s}{\delta_1}$	$\frac{s}{\epsilon\delta_2}$
Uniform Sampling	$\frac{\mu s}{\eta^2}\log\frac{s}{\delta_1}$	$\frac{\mu s}{\epsilon\delta_2}$
SRHT	$\frac{s+\log n}{\eta^2}\log\frac{s}{\delta_1}$	$\frac{s+\log n}{\epsilon\delta_2}$
Gaussian Projection	$\frac{s+\log(1/\delta_1)}{\eta^2}$	$\frac{s}{\epsilon\delta_2}$
CountSketch	$\frac{s^2}{\delta_1\eta^2}$	$\frac{s}{\epsilon\delta_2}$

Proof This lemma is a reproduction of (Wang et al., 2016b, Lemma 2), and is a collation of heterogenously stated results from the literature. The random sampling estimates were established by Drineas et al. (2008); Wang et al. (2016a); Woodruff (2014). The Gaussian projection was firstly analyzed by Johnson and Lindenstrauss (1984); see Woodruff (2014) for a proof of the stated sufficient sketching dimensions. The SRHT estimates are from Drineas et al. (2011); Lu et al. (2013); Tropp (2011). The CountSketch estimates are from Meng and Mahoney (2013); Nelson and Nguyên (2013). ■

A.3. Completing the Proof of Theorem 1

Proof Let $\mathbf{K}_s = \mathbf{V}_s\mathbf{\Lambda}_s\mathbf{V}_s^T \in \mathbb{R}^{n\times n}$ be the truncated SVD of $\mathbf{K}$. It follows from Lemma 10 and the sketch sizes defined in Table 2 that

$$\begin{aligned}
\big\|\mathbf{V}_s^T\mathbf{P}\mathbf{P}^T\mathbf{V}_s - \mathbf{I}_s\big\|_2 &\leq \eta, \\
\big\|(\mathbf{K}^{1/2} - \mathbf{K}_s^{1/2})\mathbf{P}\mathbf{P}^T\mathbf{V}_s\big\|_F^2 &\leq \epsilon\big\|\mathbf{K}^{1/2} - \mathbf{K}_s^{1/2}\|_F^2
\end{aligned}$$

hold simultaneously with probability at least 0.9.

It follows from Lemma 8 that

$$\begin{aligned}\left\|\mathbf{K}-\left(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\right)_{s}\right\|_{*} &= \min_{\mathrm{rank}(\mathbf{Z})\leq s}\left\|\mathbf{K}^{1/2}-(\mathbf{K}^{1/2}\mathbf{P})\mathbf{Z}\right\|_{F}^{2}\\ &\leq \left\|\mathbf{K}^{1/2}-\mathbf{K}_{s}^{1/2}\right\|_{F}^{2}+\left\|(\mathbf{K}^{1/2}-\mathbf{K}_{s}^{1/2})\mathbf{P}(\mathbf{V}_{s}^{T}\mathbf{P})^{\dagger}\right\|_{F}^{2}\\ &= \left\|\mathbf{K}-\mathbf{K}_{s}\right\|_{*}+\left\|(\mathbf{K}^{1/2}-\mathbf{K}_{s}^{1/2})\mathbf{P}\mathbf{P}^{T}\mathbf{V}_{s}(\mathbf{V}_{s}^{T}\mathbf{P}\mathbf{P}^{T}\mathbf{V}_{s})^{\dagger}\right\|_{F}^{2}\\ &\leq \left\|\mathbf{K}-\mathbf{K}_{s}\right\|_{*}+\sigma_{s}^{-2}\left(\mathbf{V}_{s}^{T}\mathbf{P}\mathbf{P}^{T}\mathbf{V}_{s}\right)\left\|(\mathbf{K}^{1/2}-\mathbf{K}_{s}^{1/2})\mathbf{P}\mathbf{P}^{T}\mathbf{V}_{s}\right\|_{F}^{2},\end{aligned}$$

where the former inequality follows from Lemma 7 (with $t=1$). It follows that

$$\left\|\mathbf{K}-\left(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{T}\right)_{s}\right\|_{*} \leq \left\|\mathbf{K}-\mathbf{K}_{s}\right\|_{*}+\tfrac{\epsilon}{(1-\eta)^{2}}\left\|\mathbf{K}-\mathbf{K}_{s}\right\|_{*}.$$

We let η be a constant and obtain the former claim of Theorem 1. The latter claim of Theorem 1 follows from Lemma 8. ■

Appendix B. Proof of Theorem 2

In this section, we will provide a proof of Theorem 2, our main result for kernel k-means approximation. We will start, in Sections B.1, B.2, and B.3, by presenting several technical tools of independent interest (Lemmas 11, 12, and 13, respectively). Then, in Section B.4, we will use use these tools to prove Theorem 2.

The proof of Theorem 2 will proceed by combining Theorem 1 with Lemmas 11, 12, and 13. For convenience, the structure of proof is given in Figure 9.

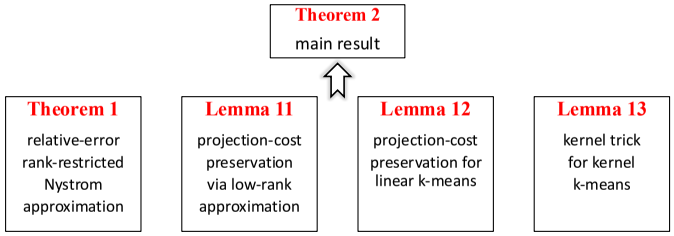


Figure 9: The structure of the proof of Theorem 2.

We use the notation of *orthogonal projection matrix* throughout this section. A matrix $\mathbf{M}\in\mathbb{R}^{n\times n}$ is an orthogonal projection matrix if and only if there exists a matrix $\mathbf{Q}$ with orthonormal columns such that $\mathbf{M}=\mathbf{Q}\mathbf{Q}^{T}$. It follows that $\mathbf{M}^{T}=\mathbf{M}$, $\mathbf{M}\mathbf{M}=\mathbf{M}$, $\mathbf{M}\preceq\mathbf{I}_{n}$ and that $\mathbf{I}_{n}-\mathbf{M}$ is also an orthogonal projection matrix.

B.1. Projection-Cost Preservation via Low-Rank Approximation

We first define projection-cost preservation, then show that certain low-rank approximations are projection-cost preserving. Projection-cost preservation was named and systematized by Cohen et al. (2015), but the idea has been used in earlier works (Boutsidis et al., 2010, 2015; Feldman et al., 2013; Liang et al., 2014).

Definition 2 (Rank k Projection-Cost Preservation) *Fix a matrix $\mathbf{A}\in\mathbb{R}^{n\times d}$ and error parameters $\epsilon_1,\epsilon_2>0$. The matrix $\mathbf{B}\in\mathbb{R}^{n\times d}$ has the rank k projection-cost preserving property with respect to $\mathbf{A}$ if for any rank k orthogonal projection matrices $\mathbf{\Pi}\in\mathbb{R}^{n\times n}$,*

$$(1-\epsilon_1)\left\|\mathbf{A}-\mathbf{\Pi}\mathbf{A}\right\|_{F}^{2} \leq \left\|\mathbf{B}-\mathbf{\Pi}\mathbf{B}\right\|_{F}^{2}+\alpha \leq (1+\epsilon_2)\left\|\mathbf{A}-\mathbf{\Pi}\mathbf{A}\right\|_{F}^{2},$$

for some fixed non-negative α that can depend on $\mathbf{A}$ and $\mathbf{B}$ but is independent of $\mathbf{\Pi}$.

The next lemma provides a way to construct a projection-cost preserving sketch of $\mathbf{A}$. It states that if a rank-s matrix $\tilde{\mathbf{A}}_s$ in the span of $\mathbf{A}$ is constructed so that $\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^{T}$ approximates $\mathbf{A}\mathbf{A}^{T}$ sufficiently well in terms of the trace norm, then $\tilde{\mathbf{A}}_s$ has the projection-cost preserving property. Observe that this lemma explicitly relates projection-cost preserving sketches to approximate matrix multiplication (Drineas et al., 2006; Mahoney, 2011), where the latter is measures with respect to the trace norm (Gittens and Mahoney, 2016). We provide a proof of Lemma 11 in Appendix D.

Lemma 11 (Projection-Cost Preservation via Low-Rank Approximation) *Fix an error parameter $\epsilon\in(0,1)$. Let $\mathbf{A}\in\mathbb{R}^{n\times d}$ and choose a rank-s matrix $\tilde{\mathbf{A}}_s$ that satisfies the following conditions:*

(i) there is an orthogonal projection matrix $\mathbf{M}$ such that $\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^{T}=\mathbf{A}\mathbf{M}\mathbf{A}^{T}$, and

(ii) $\|\mathbf{A}\mathbf{A}^{T}-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^{T}\|_{*}\leq(1+\epsilon)\left\|\mathbf{A}\mathbf{A}^{T}-\mathbf{A}_s\mathbf{A}_s^{T}\right\|_{*}$.

Then there exists a fixed $\alpha\geq 0$ such that, for any rank k projection matrices $\mathbf{\Pi}\in\mathbb{R}^{n\times n}$,

$$\left\|(\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}\right\|_{F}^{2} \leq \left\|(\mathbf{I}_n-\mathbf{\Pi})\tilde{\mathbf{A}}_s\right\|_{F}^{2}+\alpha \leq \left(1+\epsilon+\tfrac{k}{s}\right)\left\|(\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}\right\|_{F}^{2}.$$

B.2. Linear k-Means Clustering and Projection-Cost Preservation

Linear k-means clustering is formally defined by the optimization problem (1). To relate the linear k-means clustering problem to the projection-cost preservation property, we use an equivalent formulation of (1) as an optimization over the set of rank-k projection matrices of the form $\mathbf{X}\mathbf{X}^{T}$, where $\mathbf{X}$ is a *cluster indicator matrix*. This approach was adopted by Boutsidis et al. (2009, 2010, 2015); Cohen et al. (2015); Ding et al. (2005). Following them, we define the cluster indicator matrix in the following and give an example in Figure 10.

Definition 3 (Cluster Indicator Matrices) *Let n and k be given, and let $\mathcal{J}_1,\ldots,\mathcal{J}_k$ be a k-partition of the set $[n]$. The cluster indicator matrix corresponding to $\mathcal{J}_1,\ldots,\mathcal{J}_k$ is the $n\times k$ matrix $\mathbf{X}$ defined by*

$$x_{ij} = \begin{cases}\frac{1}{\sqrt{|\mathcal{J}_j|}} & \text{if } i\in\mathcal{J}_j;\\ 0 & \text{otherwise.}\end{cases}$$

We take $\mathcal{X}_{n,k}$ to be the collection of cluster indicator matrices corresponding to all the k-partitions of $[n]$.

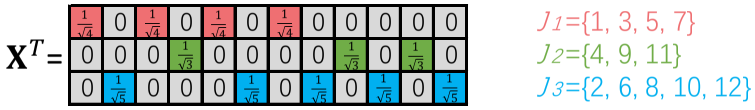


Figure 10: An example of a cluster indicator matrix $\mathbf{X} \in \mathcal{X}_{n,k}$. In this example, $n = 12$, $k = 3$, and $\mathbf{X}$ corresponds to the indicated 3-partition $\mathcal{J}_1$, $\mathcal{J}_2$, $\mathcal{J}_3$.

A cluster indicator matrix $\mathbf{X}$ has exactly one non-zero entry in each row and has orthonormal columns. That is, because of the normalization of the non-zero entries, a cluster indicator matrix $\mathbf{X}$ is an orthogonal matrix, but one with the additional constraint that there is only one non-zero entry in each row. (Indeed, the insight of Boutsidis et al. (2009, 2010, 2015) was that the rank-constrained optimization over all orthonormal matrices, which provides the usual PCA/SVD-based low-rank approximation, is a relaxation of the optimization over this smaller set of orthonormal matrices.) Among other things, it follows that $\mathbf{X}\mathbf{X}^T$ is a rank-k projection matrix.

Given this, assume $i \in \mathcal{J}_j$; it can be verified that the i-th row of $\mathbf{X}\mathbf{X}^T\mathbf{A} \in \mathbb{R}^{n\times d}$ is the centroid of $\mathcal{J}_j$:

$$(\mathbf{X}\mathbf{X}^T\mathbf{A})_{i:} \;=\; \mathbf{x}_{i:}\mathbf{X}^T\mathbf{A} \;=\; \frac{1}{|\mathcal{J}_j|}\sum_{l\in\mathcal{J}_j}\mathbf{a}_{l:}.$$

Consider the objective function of (1). When $\mathbf{X}$ is the cluster indicator matrix corresponding to the input k-partition, this objective can be rewritten in terms of $\mathbf{X}\mathbf{X}^T$ as

$$\sum_{j=1}^{k}\sum_{i\in\mathcal{J}_j}\left\|\mathbf{a}_{i:} - \frac{1}{|\mathcal{J}_j|}\sum_{l\in\mathcal{J}_j}\mathbf{a}_{l:}\right\|_2^2 = \sum_{j=1}^{k}\sum_{i\in\mathcal{J}_j}\left\|\mathbf{a}_{i:} - (\mathbf{X}\mathbf{X}^T\mathbf{A})_{i:}\right\|_2^2$$
$$= \sum_{i=1}^{n}\left\|\mathbf{a}_{i:} - (\mathbf{X}\mathbf{X}^T\mathbf{A})_{i:}\right\|_2^2 \;=\; \left\|\mathbf{A} - \mathbf{X}\mathbf{X}^T\mathbf{A}\right\|_F^2.$$

Therefore, the linear k-means clustering problem (1) is equivalent to the optimization problem

$$\operatorname*{argmin}_{\mathbf{X}\in\mathcal{X}_{n,k}} \frac{1}{n}\left\|\mathbf{A} - \mathbf{X}\mathbf{X}^T\mathbf{A}\right\|_F^2. \tag{10}$$

In the remainder of this section, we use this equivalence between k-partitionings of $[n]$ and cluster indicator matrices without comment; e.g., we refer to the outputs of linear k-means clustering algorithms as cluster indicator matrices. Because the linear k-means clustering problem (10) is NP-hard, in practice it is solved by variants of Lloyd's algorithm.

Definition 4 defines an appropriate notion of approximation for linear k-means clustering algorithms. Definition 4 is an equivalent statement of Definition 1.

Definition 4 (γ-Approximate Algorithms) *We call an algorithm $\mathcal{A}_\gamma$ ($\gamma \geq 1$) a γ-approximate linear k-means algorithm if it produces a cluster indicator matrix $\tilde{\mathbf{X}}$ such that*

$$\left\|\mathbf{A} - \tilde{\mathbf{X}}\tilde{\mathbf{X}}^T\mathbf{A}\right\|_F^2 \;\leq\; \gamma\cdot\min_{\mathbf{X}\in\mathcal{X}_{n,k}}\left\|\mathbf{A} - \mathbf{X}\mathbf{X}^T\mathbf{A}\right\|_F^2$$

for any conformal matrix $\mathbf{A}$.

Given this, we now state Lemma 12, which relates projection-cost preservation to linear k-means clustering. Let $\mathbf{A} \in \mathbb{R}^{n\times d}$ be the input matrix and $\mathbf{B} \in \mathbb{R}^{n\times s}$ be a smaller matrix ($s < d$) satisfying the projection-cost preservation property. The projection-cost preserving property ensures that the k-partition obtained by applying linear k-means clustering to the rows of $\mathbf{B}$, instead of $\mathbf{A}$, is a good k-partition for the rows of $\mathbf{A}$. This lemma was established by Cohen et al. (2015). To make this paper self-contained, we provide a proof of Lemma 12 in Appendix E.

Lemma 12 (Projection-Cost Preservation for Linear k-Means) *Fix $\mathbf{A} \in \mathbb{R}^{n\times d}$ and assume $\mathbf{B} \in \mathbb{R}^{n\times s}$ satisfies the rank k projection-cost preservation property in Definition 2 with probability at least $1-\delta$. Let the error parameters $\epsilon_1, \epsilon_2 \in (0,1)$ be as in Definition 2. If $\tilde{\mathbf{X}}_\mathbf{B}$ is the result of applying a γ-approximate linear k-means clustering algorithm to the rows of $\mathbf{B}$, then with probability at least $1-2\delta$,*

$$\left\|\mathbf{A} - \tilde{\mathbf{X}}_\mathbf{B}\tilde{\mathbf{X}}_\mathbf{B}^T\mathbf{A}\right\|_F^2 \;\leq\; \gamma\cdot\tfrac{1+\epsilon_2}{1-\epsilon_1}\cdot\min_{\mathbf{X}\in\mathcal{X}_{n,k}}\left\|\mathbf{A} - \mathbf{X}\mathbf{X}^T\mathbf{A}\right\|_F^2.$$

B.3. Kernel k-Means Clustering

Given input vectors $\mathbf{a}_1, \ldots, \mathbf{a}_n$, the kernel k-means clustering algorithm applies linear k-means clustering to feature vectors $\boldsymbol{\phi}(\mathbf{a}_1), \ldots, \boldsymbol{\phi}(\mathbf{a}_n)$. For convenience, we let $\boldsymbol{\phi}(\mathbf{a}_1)^T, \ldots, \boldsymbol{\phi}(\mathbf{a}_n)^T$ constitute the rows of the matrix $\boldsymbol{\Phi}$. Lemma 13 argues that all the information in $\boldsymbol{\Phi}$ relevant to the kernel k-means clustering problem is present in the kernel matrix $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$. Variants of this lemma are well-known (Schölkopf and Smola, 2002).

Lemma 13 (Kernel Trick for Kernel k-Means) *Let $\boldsymbol{\Phi}$ be a matrix with n rows and let $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T \in \mathbb{R}^{n\times n}$. Let $\mathbf{K} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$ be the EVD of $\mathbf{K}$. Then for any matrix $\mathbf{X}$ with orthonormal columns,*

$$\left\|\boldsymbol{\Phi} - \mathbf{X}\mathbf{X}^T\boldsymbol{\Phi}\right\|_F^2 \;=\; \left\|(\mathbf{V}\boldsymbol{\Lambda}^{1/2}) - \mathbf{X}\mathbf{X}^T(\mathbf{V}\boldsymbol{\Lambda}^{1/2})\right\|_F^2.$$

Proof Because $\mathbf{X}$ has orthonormal columns, $\mathbf{X}\mathbf{X}^T$ is an orthogonal projection matrix, and $(\mathbf{I}_n - \mathbf{X}\mathbf{X}^T)$ is the orthogonal projection onto the complementary space. It follows that, as claimed,

$$\left\|\boldsymbol{\Phi} - \mathbf{X}\mathbf{X}^T\boldsymbol{\Phi}\right\|_F^2 \;=\; \mathrm{tr}\Big((\boldsymbol{\Phi}^T(\mathbf{I}_n - \mathbf{X}\mathbf{X}^T)\boldsymbol{\Phi}\Big) \;=\; \mathrm{tr}\Big((\mathbf{I}_n - \mathbf{X}\mathbf{X}^T)\mathbf{K}\Big)$$
$$= \mathrm{tr}\Big((\mathbf{I}_n - \mathbf{X}\mathbf{X}^T)\mathbf{V}\boldsymbol{\Lambda}^{1/2}\boldsymbol{\Lambda}^{1/2}\mathbf{V}(\mathbf{I}_n - \mathbf{X}\mathbf{X}^T)\Big) \;=\; \left\|(\mathbf{V}\boldsymbol{\Lambda}^{1/2}) - \mathbf{X}\mathbf{X}^T(\mathbf{V}\boldsymbol{\Lambda}^{1/2})\right\|_F^2.$$

The equalities are justified by, respectively, the definition of the squared Frobenius norm and the idempotence of projections; the cyclicity of the trace and the fact that $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$; the cyclicity of the trace and the EVD of $\mathbf{K}$; and the definition of the squared Frobenius norm. ■

B.4. Completing the Proof of Theorem 2

To complete the proof of Theorem 2, we first establish Lemma 14, and then use it to prove Theorem 15, which is equivalent to Theorem 2. Let $\mathbf{\Phi}$ be a matrix with n rows, and $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^T \in \mathbb{R}^{n\times n}$ be the corresponding kernel matrix.

Lemma 14 *Assume that $\mathbf{B} \in \mathbb{R}^{n\times s}$ satisfies*

$$\mathrm{tr}\big(\mathbf{K} - \mathbf{B}\mathbf{B}^T\big) \;\le\; (1+\epsilon)\big\|\mathbf{K} - \mathbf{K}_s\big\|_* \tag{11}$$

and that there is an orthogonal projection matrix $\mathbf{M}$ such that

$$\mathbf{B}\mathbf{B}^T \;=\; \mathbf{K}^{1/2}\mathbf{M}\mathbf{K}^{1/2}. \tag{12}$$

Use the rows of $\mathbf{B}$ as input to a γ-approximation algorithm (see Definition 4) and let $\tilde{\mathbf{X}}_\mathbf{B}$ be the output cluster indicator matrix (see Definition 3), then

$$\big\|\mathbf{\Phi} - \tilde{\mathbf{X}}_\mathbf{B}\tilde{\mathbf{X}}_\mathbf{B}^T\mathbf{\Phi}\big\|_F^2 \;\le\; \gamma\big(1+\epsilon+\tfrac{k}{s}\big)\cdot \min_{\mathbf{X}\in\mathcal{X}_{n,k}} \big\|\mathbf{\Phi} - \mathbf{X}\mathbf{X}^T\mathbf{\Phi}\big\|_F^2.$$

Proof Because assumptions (11) and (12) are satisfied, Lemma 11 ensures that $\mathbf{B}$ is a rank k projection-cost preserving sketch of $\mathbf{K}$. That is, there exists a constant $\alpha \ge 0$ independent of $\mathbf{\Pi}$ such that

$$\big\|(\mathbf{I}_n - \mathbf{\Pi})\mathbf{K}^{1/2}\big\|_F^2 \;\le\; \big\|(\mathbf{I}_n - \mathbf{\Pi})\mathbf{B}\big\|_F^2 + \alpha \;\le\; \big(1+\epsilon+\tfrac{k}{s}\big)\big\|(\mathbf{I}_n - \mathbf{\Pi})\mathbf{K}^{1/2}\big\|_F^2$$

holds for any rank-k orthogonal projection matrix $\mathbf{\Pi}$. It follows by Lemma 12 that $\tilde{\mathbf{X}}_\mathbf{B}$ gives an almost optimal linear k-means clustering over the rows of $\mathbf{K}^{1/2}$:

$$\big\|\mathbf{K}^{1/2} - \tilde{\mathbf{X}}_\mathbf{B}\tilde{\mathbf{X}}_\mathbf{B}^T\mathbf{K}^{1/2}\big\|_F^2 \;\le\; \gamma\big(1+\epsilon+\tfrac{k}{s}\big)\cdot \min_{\mathbf{X}\in\mathcal{X}_{n,k}} \big\|\mathbf{K}^{1/2} - \mathbf{X}\mathbf{X}^T\mathbf{K}^{1/2}\big\|_F^2.$$

Finally, Lemma 13 states that clustering the rows of $\mathbf{K}^{1/2}$ is equivalent to clustering the rows of $\mathbf{\Phi}$, so we reach the desired conclusion that

$$\big\|\mathbf{\Phi} - \tilde{\mathbf{X}}_\mathbf{B}\tilde{\mathbf{X}}_\mathbf{B}^T\mathbf{\Phi}\big\|_F^2 \;\le\; \gamma\big(1+\epsilon+\tfrac{k}{s}\big)\cdot \min_{\mathbf{X}\in\mathcal{X}_{n,k}} \big\|\mathbf{\Phi} - \mathbf{X}\mathbf{X}^T\mathbf{\Phi}\big\|_F^2.$$

■

Finally, we state and prove Theorem 15. Theorem 15 shows that approximate kernel k-means clustering using the Nyström method exhibits a $1+\epsilon+\frac{k}{s}$ approximation ratio. Observe that Theorem 15 is equivalent to Theorem 2, and thus establishing it also establishes Theorem 2. Recall the Nyström approximation is $\mathbf{K} \approx \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T$, where $\mathbf{C} = \mathbf{K}\mathbf{P}$, $\mathbf{W} = \mathbf{P}^T\mathbf{K}\mathbf{P}$, and $\mathbf{P}$ is a sketching matrix.

Theorem 15 *Choose a sketching sketching matrix $\mathbf{P} \in \mathbb{R}^{n\times c}$ and sketch size c consistent with Table 2. Let $\mathbf{B} \in \mathbb{R}^{n\times s}$ be any matrix satisfying $\mathbf{B}\mathbf{B}^T = (\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s$. Let the cluster indicator matrix $\mathbf{X}_\mathbf{B}$ be the output of any γ-approximate k-means clustering algorithm applied to the rows of $\mathbf{B}$. With probability at least 0.9,*

$$\big\|\mathbf{\Phi} - \mathbf{X}_\mathbf{B}\mathbf{X}_\mathbf{B}^T\mathbf{\Phi}\big\|_F^2 \;\le\; \gamma\big(1+\epsilon+\tfrac{k}{s}\big)\min_{\mathbf{X}\in\mathcal{X}_{n,k}} \big\|\mathbf{\Phi} - \mathbf{X}\mathbf{X}^T\mathbf{\Phi}\big\|_F^2.$$

Proof Theorem 1 shows that (11) holds with probability at least 0.9. Theorem 1 also shows that $\mathbf{B}\mathbf{B}^T = \mathbf{K}^{1/2}\mathbf{Q}\mathbf{Q}^T\mathbf{K}^{1/2}$ where $\mathbf{Q}$ is a $n\times s$ matrix with orthonormal columns, so (12) holds surely. The theorem now follows from Lemma 14. ■

Appendix C. Proof of Theorem 4

Section C.1 analyzes rank-restricted Nystöm with power method. Section C.2 completes the proof of Theorem 4. Here we re-state the algorithm of Theorem 4: first, set any $c \ge s$ and draw a Gaussian projection matrix $\mathbf{P} \in \mathbb{R}^{n\times c}$; second, run the power iteration $\mathbf{P} \longleftarrow \mathbf{K}\mathbf{P}$ for t times; third, orthogonalize $\mathbf{P} \in \mathbb{R}^{n\times c}$ to obtain $\mathbf{U} \in \mathbb{R}^{n\times c}$; fourth, compute $\mathbf{C} = \mathbf{K}\mathbf{U}$ and $\mathbf{W} = \mathbf{U}^T\mathbf{K}\mathbf{U}$; last, find any $\mathbf{B} \in \mathbb{R}^{n\times s}$ satisfying $\mathbf{B}\mathbf{B}^T = (\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s$.

C.1. Rank-Restricted Nyström with Power Method

Theorem 16 analyzes the rank-restricted Nyström with power method. The theorem will be used to prove Theorem 4.

Theorem 16 *Let $\mathbf{K} \in \mathbb{R}^{n\times n}$ be an SPSD matrix, s be the target rank, c ($\ge s$) be arbitrary, and $\epsilon \in (0,1)$ be an approximation parameter. Let $\mathbf{C} \in \mathbb{R}^{n\times c}$ and $\mathbf{W} \in \mathbb{R}^{c\times c}$ be computed by the above algorithm with $t = \mathcal{O}(\frac{\log(n/\epsilon)}{\log(\sigma_s/\sigma_{s+1})})$ power iterations, where σ_i is the i-th singular value of $\mathbf{K}$. If $c = s + \mathcal{O}(\log\frac{1}{\delta})$, then*

$$\big\|\mathbf{K} - (\mathbf{C}\mathbf{W}^\dagger\mathbf{C}^T)_s\big\|_* \;\le\; (1+\epsilon)\big\|\mathbf{K} - \mathbf{K}_s\big\|_*$$

with probability at least $1-\delta$. If $c = s$, then this bound holds with a constant probability (that depends on s).

Proof Let $\mathbf{V}_{-s} \in \mathbb{R}^{n\times(n-s)}$ be the orthogonal complement of $\mathbf{V}_s$. Let $\mathbf{P}$ be an $n\times c$ standard Gaussian matrix. Then $\mathbf{V}_{-s}^T\mathbf{P}$ is $(n-s)\times c$ standard Gaussian matrix, and $\mathbf{V}_s^T\mathbf{P}$ is $s\times c$ standard Gaussian matrix.

It is well known that the largest singular value of an $(n-s)\times c$ standard Gaussian matrix is at most $\sqrt{n-s}+\sqrt{c}+\zeta$ with probability $1-\exp(-\zeta^2/2)$. See (Vershynin, 2010).

Consider $c > s$. The least singular value of an $s\times c$ standard Gaussian matrix is at least $\sqrt{c}-\sqrt{s}-\zeta$ with probability $1-\exp(-\zeta^2/2)$. Combining the above results, we have that

$$\frac{\sigma_1^2(\mathbf{V}_{-s}^T\mathbf{P})}{\sigma_s^2(\mathbf{V}_s^T\mathbf{P})} \;\le\; \frac{\sqrt{n-s}+\sqrt{c}+\sqrt{2\log(1/\delta)}}{\sqrt{c}-\sqrt{s}-\sqrt{2\log(1/\delta)}}$$

holds with probability $1-2\delta$.

Consider $c = s$. Rudelson and Vershynin (2008); Tao and Vu (2010) showed that the least singular value of an $s\times s$ standard Gaussian matrix $\mathbf{G}$ satisfy

$$\mathbb{P}\big\{\sigma_s(\mathbf{G}) \le \tfrac{\delta_1}{\sqrt{s}}\big\} \;=\; \delta_1 + \mathcal{O}(s^{-\tau}),$$

where $\tau > 0$ is a constant. It follows that

$$\frac{\sigma_1^2(\mathbf{V}_{-s}^T\mathbf{P})}{\sigma_s^2(\mathbf{V}_s^T\mathbf{P})} \;\le\; \frac{\sqrt{n-s}+\sqrt{s}+\sqrt{2\log(1/\delta_2)}}{\delta_1/\sqrt{s}}$$

holds with probability $1-\delta_1-\delta_2-\mathcal{O}(s^{-\tau})$.

For either $c>s$ or $c=s$, it follows from Lemma 17, which will be proved subsequently, that

$$\begin{aligned}\left\|\mathbf{K}-(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T)_s\right\|_* &\leq \left\|\mathbf{K}-\mathbf{K}_s\right\|_* + \tfrac{\sigma_1^2(\mathbf{V}_{-s}^T\mathbf{P})}{\sigma_s^2(\mathbf{V}_s^T\mathbf{P})}\left(\tfrac{\sigma_{s+1}(\mathbf{K})}{\sigma_s(\mathbf{K})}\right)^{2t}\left\|\mathbf{K}-\mathbf{K}_s\right\|_* \\ &\leq \left\|\mathbf{K}-\mathbf{K}_s\right\|_* + \mathcal{O}(ns)\left(\tfrac{\sigma_{s+1}(\mathbf{K})}{\sigma_s(\mathbf{K})}\right)^{2t}\left\|\mathbf{K}-\mathbf{K}_s\right\|_*\end{aligned}$$

holds with constant probability, by which the theorem follows. ■

Lemma 17 *Let $\mathbf{K}\in\mathbb{R}^{n\times n}$ be any fixed SPSD matrix, s and t be any positive integer, and $\mathbf{V}_s\in\mathbb{R}^{n\times s}$ be the top s singular vectors of $\mathbf{K}$. Let $\mathbf{P}\in\mathbb{R}^{n\times c}$ satisfy that $\mathbf{P}^T\mathbf{V}_s\in\mathbb{R}^{c\times s}$ has full column rank. Let $\mathbf{U}\in\mathbb{R}^{n\times c}$ be the orthonormal bases of $\mathbf{K}^t\mathbf{P}$. Let $\mathbf{C}=\mathbf{K}\mathbf{U}$ and $\mathbf{W}=\mathbf{U}^T\mathbf{K}\mathbf{U}$. Then*

$$\left\|\mathbf{K}-(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T)_s\right\|_* \leq \left\|\mathbf{K}-\mathbf{K}_s\right\|_* + \left(\tfrac{\sigma_{s+1}(\mathbf{K})}{\sigma_s(\mathbf{K})}\right)^{2t}\tfrac{\sigma_1^2(\mathbf{V}_{-s}^T\mathbf{P})}{\sigma_s^2(\mathbf{V}_s^T\mathbf{P})}\left\|\mathbf{K}-\mathbf{K}_s\right\|_*.$$

Here $\mathbf{V}_{-s}\in\mathbb{R}^{n\times(n-s)}$ is the orthogonal complement of $\mathbf{V}_s$.

Proof It follows from Lemma 8 that

$$\begin{aligned}\left\|\mathbf{K}-\left(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\right)_s\right\|_* &= \min_{\mathrm{rank}(\mathbf{Z})\leq s}\left\|\mathbf{K}^{1/2}-(\mathbf{K}^{1/2}\mathbf{U})\mathbf{Z}\right\|_F^2 \\ &= \min_{\mathrm{rank}(\mathbf{Z})\leq s}\left\|\mathbf{A}-(\mathbf{A}\mathbf{U})\mathbf{Z}\right\|_F^2\end{aligned}$$

where we let $\mathbf{A}=\mathbf{K}^{1/2}$. By definition, $\mathbf{U}$ contains the orthonormal bases of $\mathbf{K}^t\mathbf{P}=\mathbf{A}^{2t}\mathbf{P}$, and thus there exists an $s\times s$ non-singular matrix $\mathbf{R}$ such that $\mathbf{A}^{2t}\mathbf{P}=\mathbf{U}\mathbf{R}$. It follows that

$$\begin{aligned}\left\|\mathbf{K}-\left(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\right)_s\right\|_* &= \min_{\mathrm{rank}(\mathbf{Z})\leq s}\left\|\mathbf{A}-(\mathbf{A}\mathbf{A}^{2t}\mathbf{P}\mathbf{R}^{-1})\mathbf{Z}\right\|_F^2 \\ &= \min_{\mathrm{rank}(\mathbf{Y})\leq s}\left\|\mathbf{A}-(\mathbf{A}^{2t+1}\mathbf{P})\mathbf{Y}\right\|_F^2,\end{aligned}$$

where the latter equality follows from that the column spaces of $\mathbf{A}^{2t+1}\mathbf{P}$ and $\mathbf{A}^{2t+1}\mathbf{P}\mathbf{R}^{-1}$ are the same. It follows from Lemma 7 that

$$\begin{aligned}&\left\|\mathbf{K}-\left(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\right)_s\right\|_* = \min_{\mathrm{rank}(\mathbf{Y})\leq s}\left\|\mathbf{A}-(\mathbf{A}^{2t+1}\mathbf{P})\mathbf{Y}\right\|_F^2 \\ &\leq \left\|\mathbf{A}-\mathbf{A}_s\right\|_F^2 + \left(\tfrac{\sigma_{s+1}^2(\mathbf{A})}{\sigma_s^2(\mathbf{A})}\right)^{2t}\left\|(\mathbf{A}-\mathbf{A}_s)\mathbf{P}(\mathbf{V}_s^T\mathbf{P})^{\dagger}\right\|_F^2 \\ &\leq \left\|\mathbf{A}-\mathbf{A}_s\right\|_F^2 + \left(\tfrac{\sigma_{s+1}^2(\mathbf{A})}{\sigma_s^2(\mathbf{A})}\right)^{2t}\left\|\mathbf{A}-\mathbf{A}_s\right\|_F^2\left\|\mathbf{V}_{-s}^T\mathbf{P}\right\|_2^2\left\|(\mathbf{V}_s^T\mathbf{P})^{\dagger}\right\|_2^2,\end{aligned}$$

where we define $\mathbf{V}_{-s}\in\mathbb{R}^{n\times(n-s)}$ as the orthogonal complement of $\mathbf{V}_s$. By our definition $\mathbf{A}=\mathbf{K}^{1/2}$, it follows that

$$\begin{aligned}&\left\|\mathbf{K}-\left(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T\right)_s\right\|_* \\ &\leq \left\|(\mathbf{K}-\mathbf{K}_s)^{1/2}\right\|_F^2 + \left(\tfrac{\sigma_{s+1}(\mathbf{K})}{\sigma_s(\mathbf{K})}\right)^{2t}\tfrac{\sigma_1^2(\mathbf{V}_{-s}^T\mathbf{P})}{\sigma_s^2(\mathbf{V}_s^T\mathbf{P})}\left\|(\mathbf{K}-\mathbf{K}_s)^{1/2}\right\|_F^2 \\ &= \left\|\mathbf{K}-\mathbf{K}_s\right\|_* + \left(\tfrac{\sigma_{s+1}(\mathbf{K})}{\sigma_s(\mathbf{K})}\right)^{2t}\tfrac{\sigma_1^2(\mathbf{V}_{-s}^T\mathbf{P})}{\sigma_s^2(\mathbf{V}_s^T\mathbf{P})}\left\|\mathbf{K}-\mathbf{K}_s\right\|_*,\end{aligned}$$

by which the lemma follows. ■

C.2. Completing the Proof of Theorem 4

Finally, we state and prove Theorem 18. Observe that Theorem 18 is equivalent to Theorem 4, and thus establishing it also establishes Theorem 4.

Theorem 18 *Let c, t, and $\mathbf{B}\in\mathbb{R}^{n\times s}$ be defined in the begining of this section. Set $t=\mathcal{O}(\frac{\log(n/\epsilon)}{\log(\sigma_s/\sigma_{s+1})})$. Let the cluster indicator matrix $\mathbf{X}_\mathbf{B}$ be the output of any γ-approximate k-means clustering algorithm applied to the rows of $\mathbf{B}$. If $c=s+\mathcal{O}(\log\frac{1}{\delta})$, then*

$$\left\|\mathbf{\Phi}-\mathbf{X}_\mathbf{B}\mathbf{X}_\mathbf{B}^T\mathbf{\Phi}\right\|_F^2 \leq \gamma\left(1+\epsilon+\tfrac{k}{s}\right)\min_{\mathbf{X}\in\mathcal{X}_{n,k}}\left\|\mathbf{\Phi}-\mathbf{X}\mathbf{X}^T\mathbf{\Phi}\right\|_F^2$$

holds with probability at least $1-\delta$. If $c=s$, then the above inequality holds with probability $0.9-\mathcal{O}(s^{-\tau})$, where τ is a constant.

Proof Since $\mathbf{B}\mathbf{B}^T=(\mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^T)_s$ by definition, Theorem 16 ensures that (11) holds with some constant probability. Theorem 1 shows that $\mathbf{B}\mathbf{B}^T=\mathbf{K}^{1/2}\mathbf{Q}\mathbf{Q}^T\mathbf{K}^{1/2}$ where $\mathbf{Q}$ is a $n\times s$ matrix with orthonormal columns, so (12) holds surely. The theorem now follows from Lemma 14. ■

Appendix D. Proof of Lemma 11

To prove Lemma 11, we first establish a key lemma, Lemma 19. This lemma will make use of the following two assumptions.

Assumption 1 *Assume that $\tilde{\mathbf{A}}_s$ satisfies $\mathrm{tr}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\leq(1+\epsilon)\left\|\mathbf{A}\mathbf{A}^T-\mathbf{A}_s\mathbf{A}_s^T\right\|_*$.*

Assumption 2 *Assume there exists such an orthogonal projection matrix $\mathbf{M}$ that $\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T=\mathbf{A}\mathbf{M}\mathbf{A}^T$.*

Lemma 19 *Let $\mathbf{A}\in\mathbb{R}^{n\times d}$ be any fixed matrix. Fix an error parameter $\epsilon\in(0,1)$. Let the rank s matrix $\tilde{\mathbf{A}}_s$ satisfy Assumptions 1 and 2. Then for any rank k orthogonal projection matrix $\mathbf{\Pi}\in\mathbb{R}^{n\times n}$,*

$$\mathrm{tr}\Big(\mathbf{\Pi}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\mathbf{\Pi}\Big) \leq \left(\epsilon+\tfrac{k}{s}\right)\left\|\mathbf{A}-\mathbf{\Pi}\mathbf{A}\right\|_F^2.$$

Proof It holds that

$$\begin{aligned}&\mathrm{tr}\Big((\mathbf{I}_n-\mathbf{\Pi})\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\Big) \\ &= \mathrm{tr}\Big(\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)-\mathbf{\Pi}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\Big) \\ &= \mathrm{tr}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)-\mathrm{tr}\big(\mathbf{\Pi}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\mathbf{\Pi}\big) \\ &\leq (1+\epsilon)\left\|\mathbf{A}\mathbf{A}^T-\mathbf{A}_s\mathbf{A}_s^T\right\|_*-\mathrm{tr}\big(\mathbf{\Pi}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\mathbf{\Pi}\big),\end{aligned}$$

where the inequality follows from Assumption 1. It can be equivalently written as

$$\begin{aligned}&\mathrm{tr}\Big(\mathbf{\Pi}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\mathbf{\Pi}\Big)\\&\leq\ (1+\epsilon)\big\|\mathbf{A}\mathbf{A}^T-\mathbf{A}_s\mathbf{A}_s^T\big\|_*-\mathrm{tr}\Big((\mathbf{I}_n-\mathbf{\Pi})\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)(\mathbf{I}_n-\mathbf{\Pi})\Big).\end{aligned}$$

Let $\mathbf{M}$ be an orthogonal projection matrix defined in Assumption 2; by this assumption, it holds that $\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T=\mathbf{A}\mathbf{M}\mathbf{A}^T$ and $\mathrm{rank}(\mathbf{M}\mathbf{A})=s$. It follows that

$$\begin{aligned}&\mathrm{tr}\Big((\mathbf{I}_n-\mathbf{\Pi})\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)(\mathbf{I}_n-\mathbf{\Pi})\Big)\ =\ \mathrm{tr}\Big((\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}(\mathbf{I}-\mathbf{M})\mathbf{A}^T(\mathbf{I}_n-\mathbf{\Pi})\Big)\\&=\ \big\|(\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}(\mathbf{I}-\mathbf{M})\big\|_F^2\ =\ \Big\|\mathbf{A}-\underbrace{\mathbf{\Pi}\mathbf{A}}_{\text{rank }k}-\underbrace{(\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}\mathbf{M}}_{\text{rank }s}\Big\|_F^2\\&\geq\ \min_{\mathrm{rank}(\mathbf{Y})\leq s+k}\big\|\mathbf{A}-\mathbf{Y}\big\|_F^2\ =\ \big\|\mathbf{A}-\mathbf{A}_{s+k}\big\|_F^2,\end{aligned}$$

where the inequality follows from that matrix rank is subadditive function. It follows that

$$\begin{aligned}&\mathrm{tr}\Big(\mathbf{\Pi}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\mathbf{\Pi}\Big)\\&\leq\ (1+\epsilon)\big\|\mathbf{A}\mathbf{A}^T-\mathbf{A}_s\mathbf{A}_s^T\big\|_*-\big\|\mathbf{A}-\mathbf{A}_{s+k}\big\|_F^2\\&=\ (1+\epsilon)\big\|\mathbf{A}-\mathbf{A}_s\big\|_F^2-\big\|\mathbf{A}-\mathbf{A}_{s+k}\big\|_F^2\\&=\ \epsilon\big\|\mathbf{A}-\mathbf{A}_s\big\|_F^2+\sum_{i=s+1}^{n}\sigma_i^2(\mathbf{A})-\sum_{i=s+k+1}^{n}\sigma_i^2(\mathbf{A})\\&=\ \epsilon\big\|\mathbf{A}-\mathbf{A}_s\big\|_F^2+\sum_{i=s+1}^{s+k}\sigma_i^2(\mathbf{A})\\&\leq\ \epsilon\big\|\mathbf{A}-\mathbf{A}_s\big\|_F^2+\frac{k}{s}\sum_{i=k+1}^{s+k}\sigma_i^2(\mathbf{A}),\end{aligned}$$

where the last inequality follows by the decrease of the singular values. Finally, we obtain that

$$\mathrm{tr}\Big(\mathbf{\Pi}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\mathbf{\Pi}\Big)\ \leq\ \epsilon\big\|\mathbf{A}-\mathbf{A}_k\big\|_F^2+\tfrac{k}{s}\big\|\mathbf{A}-\mathbf{A}_k\big\|_F^2\ \leq\ \big(\epsilon+\tfrac{k}{s}\big)\big\|\mathbf{A}-\mathbf{\Pi}\mathbf{A}\big\|_F^2,$$

by which the lemma follows. ■

Given Lemma 19, we now provide the proof of Lemma 11.

Proof For any orthogonal projection matrix $\mathbf{M}$, it holds that $\mathbf{M}=\mathbf{M}\mathbf{M}$. It follows that

$$\begin{aligned}&\big\|(\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}\big\|_F^2-\big\|(\mathbf{I}_n-\mathbf{\Pi})\tilde{\mathbf{A}}_s\big\|_F^2\ =\ \mathrm{tr}\big((\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}\mathbf{A}^T\big)-\mathrm{tr}\big((\mathbf{I}_n-\mathbf{\Pi})\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\\&=\ \mathrm{tr}\big((\mathbf{I}_n-\mathbf{\Pi})(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T)\big)\ =\ \mathrm{tr}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)-\mathrm{tr}\big(\mathbf{\Pi}(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T)\big)\\&=\ \alpha-\mathrm{tr}\big(\mathbf{\Pi}(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T)\big),\end{aligned}$$

where the last equality follows by letting $\alpha=\mathrm{tr}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\geq 0$ which is independent of $\mathbf{\Pi}$. The above equality can be equivalently written as

$$\big\|(\mathbf{I}_n-\mathbf{\Pi})\tilde{\mathbf{A}}_s\big\|_F^2+\alpha\ \ =\ \ \big\|(\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}\big\|_F^2+\mathrm{tr}\big(\mathbf{\Pi}(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T)\mathbf{\Pi}\big).\tag{13}$$

Under Assumption 2 that $\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\preceq\mathbf{A}\mathbf{A}^T$, it follows from (13) that

$$\begin{aligned}&\big\|(\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}\big\|_F^2\\&\leq\ \big\|(\mathbf{I}_n-\mathbf{\Pi})\tilde{\mathbf{A}}_s\big\|_F^2+\alpha\\&=\ \big\|(\mathbf{I}_n-\mathbf{\Pi})\mathbf{A}\big\|_F^2+\mathrm{tr}\big(\mathbf{\Pi}(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T)\mathbf{\Pi}\big).\end{aligned}$$

Under Assumptions 1 and 2, we can apply Lemma 19 to bound the right-hand side:

$$\mathrm{tr}\Big(\mathbf{\Pi}\big(\mathbf{A}\mathbf{A}^T-\tilde{\mathbf{A}}_s\tilde{\mathbf{A}}_s^T\big)\mathbf{\Pi}\Big)\ \leq\ \big(\epsilon+\tfrac{k}{s}\big)\big\|\mathbf{A}-\mathbf{\Pi}\mathbf{A}\big\|_F^2,$$

by which the lemma follows. ■

Appendix E. Proof of Lemma 12

Because $\mathbf{B}$ enjoys the projection-cost preservation property, there exists a constant $\alpha\geq 0$ such that for any rank k orthogonal projection matrices $\mathbf{\Pi}_1$ and $\mathbf{\Pi}_2$, the two inequalities

$$(1-\epsilon_1)\big\|\mathbf{A}-\mathbf{\Pi}_1\mathbf{A}\big\|_F^2\ \leq\ \big\|\mathbf{B}-\mathbf{\Pi}_1\mathbf{B}\big\|_F^2+\alpha,\tag{14}$$

$$\big\|\mathbf{B}-\mathbf{\Pi}_2\mathbf{B}\big\|_F^2+\alpha\ \leq\ (1+\epsilon_2)\big\|\mathbf{A}-\mathbf{\Pi}_2\mathbf{A}\big\|_F^2\tag{15}$$

hold simultaneously with probability at least $1-2\delta$. Let $\mathbf{\Pi}_1=\tilde{\mathbf{X}}_{\mathbf{B}}\tilde{\mathbf{X}}_{\mathbf{B}}^T$ and $\mathbf{X}_{\mathbf{A}}^\star=\mathrm{argmin}_{\mathbf{X}\in\mathcal{X}_{n,k}}\|\mathbf{A}-\mathbf{X}\mathbf{X}^T\mathbf{A}\|_F^2$. It follows from (14) and the definition of γ-approximate k-means algorithm that

$$\begin{aligned}&(1-\epsilon_1)\big\|\mathbf{A}-\tilde{\mathbf{X}}_{\mathbf{B}}\tilde{\mathbf{X}}_{\mathbf{B}}^T\mathbf{A}\big\|_F^2\ \leq\ \big\|\mathbf{B}-\tilde{\mathbf{X}}_{\mathbf{B}}\tilde{\mathbf{X}}_{\mathbf{B}}^T\mathbf{B}\big\|_F^2+\alpha\\&\leq\ \gamma\cdot\min_{\mathbf{X}\in\mathcal{X}_{n,k}}\big\|\mathbf{B}-\mathbf{X}\mathbf{X}^T\mathbf{B}\big\|_F^2+\alpha\ \leq\ \gamma\big\|\mathbf{B}-\mathbf{X}_{\mathbf{A}}^\star{\mathbf{X}_{\mathbf{A}}^\star}^T\mathbf{B}\big\|_F^2+\alpha.\end{aligned}$$

Let $\mathbf{\Pi}_2=\mathbf{X}_{\mathbf{A}}^\star{\mathbf{X}_{\mathbf{A}}^\star}^T$. It follows from (15) that

$$\big\|\mathbf{B}-\mathbf{X}_{\mathbf{A}}^\star{\mathbf{X}_{\mathbf{A}}^\star}^T\mathbf{B}\big\|_F^2+\alpha\ \leq\ (1+\epsilon_2)\big\|\mathbf{A}-\mathbf{X}_{\mathbf{A}}^\star{\mathbf{X}_{\mathbf{A}}^\star}^T\mathbf{A}\big\|_F^2.$$

Combining the above results, we have that

$$\begin{aligned}&(1-\epsilon_1)\big\|\mathbf{A}-\tilde{\mathbf{X}}_{\mathbf{B}}\tilde{\mathbf{X}}_{\mathbf{B}}^T\mathbf{A}\big\|_F^2\ \leq\ \gamma\big\|\mathbf{B}-\mathbf{X}_{\mathbf{A}}^\star{\mathbf{X}_{\mathbf{A}}^\star}^T\mathbf{B}\big\|_F^2+\alpha\\&\leq\ \gamma\big(\big\|\mathbf{B}-\mathbf{X}_{\mathbf{A}}^\star{\mathbf{X}_{\mathbf{A}}^\star}^T\mathbf{B}\big\|_F^2+\alpha\big)\ \leq\ \gamma(1+\epsilon_2)\big\|\mathbf{A}-\mathbf{X}_{\mathbf{A}}^\star{\mathbf{X}_{\mathbf{A}}^\star}^T\mathbf{A}\big\|_F^2,\end{aligned}$$

where the second inequality follows from that $\gamma\geq 1$ and $\alpha\geq 0$. It follows that

$$\big\|\mathbf{A}-\tilde{\mathbf{X}}_{\mathbf{B}}\tilde{\mathbf{X}}_{\mathbf{B}}^T\mathbf{A}\big\|_F^2\ \leq\ \gamma\tfrac{1+\epsilon_2}{1-\epsilon_1}\big\|\mathbf{A}-\mathbf{X}_{\mathbf{A}}^\star{\mathbf{X}_{\mathbf{A}}^\star}^T\mathbf{A}\big\|_F^2\ =\ \gamma\tfrac{1+\epsilon_2}{1-\epsilon_1}\min_{\mathbf{X}\in\mathcal{X}_{n,k}}\big\|\mathbf{A}-\mathbf{X}\mathbf{X}^T\mathbf{A}\big\|_F^2$$

This concludes our proof.

References

Ahmed Alaoui and Michael W. Mahoney. Fast Randomized Kernel Ridge Regression with Statistical Guarantees. In *Advances in Neural Information Processing Systems (NIPS)*. 2015.

Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. NP-hardness of Euclidean sum-of-squares clustering. *Machine Learning*, 75(2):245–248, 2009.

Okan Arikan. Compression of motion capture databases. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 890–897. ACM, 2006.

David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *Annual ACM-SIAM Symposium on Discrete Algorithms*, 2007.

Pranjal Awasthi, Moses Charikar, Ravishankar Krishnaswamy, and Ali Kemal Sinop. The hardness of approximation of Euclidean k-means. *arXiv preprint arXiv:1502.03316*, 2015.

Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *International Conference on Learning Theory (COLT)*, 2013.

Tamara L. Berg, Alexander C. Berg, Jaety Edwards, and David A. Forsyth. Who's in the picture. *Advances in Neural Information Processing Systems (NIPS)*, 2004.

Christos Boutsidis, Petros Drineas, and Michael W. Mahoney. Unsupervised Feature Selection for the k-means Clustering Problem. In *Advances in Neural Information Processing Systems (NIPS)*. 2009.

Christos Boutsidis, Anastasios Zouzias, and Petros Drineas. Random projections for k-means clustering. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. Near-Optimal Column-Based Matrix Reconstruction. *SIAM Journal on Computing*, 43(2):687–717, 2014.

Christos Boutsidis, Anastasios Zouzias, Michael W. Mahoney, and Petros Drineas. Randomized Dimensionality Reduction for k-Means Clustering. *IEEE Transactions on Information Theory*, 61(2):1045–1062, 2015.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):1–58, 2009.

Anil Chaturvedi, J. Douglas Carroll, Paul E. Green, and John A. Rotondo. A feature-based approach to market segmentation via overlapping k-centroids clustering. *Journal of Marketing Research*, 34(3):370–377, 1997.

Jeff Cheeger. A lower bound for the smallest eigenvalue of the Laplacian. In *Problems in Analysis, Papers dedicated to Salomon Bochner*, pages 195–199. Princeton University Press, 1969.

Ke Chen. On coresets for k-median and k-means clustering in metric and Euclidean spaces and their applications. *SIAM Journal on Computing*, 39(3):923–947, 2009.

Wen-Yen Chen, Yangqiu Song, Hongjie Bai, Chih-Jen Lin, and Edward Y. Chang. Parallel Spectral Clustering in Distributed Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):568–586, 2011.

Radha Chitta, Rong Jin, Timothy C. Havens, and Anil K. Jain. Approximate Kernel k-means: Solution to Large Scale Kernel Clustering. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2011.

Radha Chitta, Rong Jin, and Anil K. Jain. Efficient Kernel Clustering Using Random Fourier Features. In *IEEE International Conference on Data Mining (ICDM)*, 2012.

Jiawei Chiu and Laurent Demanet. Sublinear Randomized Algorithms for Skeleton Decompositions. *SIAM Journal on Matrix Analysis and Applications*, 34(3):1361–1383, 2013.

Kenneth L. Clarkson and David P. Woodruff. Low rank approximation and regression in input sparsity time. In *Annual ACM Symposium on Theory of Computing (STOC)*, 2013.

Michael B Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *Annual ACM Symposium on Theory of Computing (STOC)*, 2015.

Michael B Cohen, Cameron Musco, and Christopher Musco. Input sparsity time low-rank approximation via ridge leverage score sampling. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2017.

Corinna Cortes, Mehryar Mohri, and Ameet Talwalkar. On the impact of kernel approximation on learning accuracy. In *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

Sanjoy Dasgupta and Yoav Freund. Random projection trees for vector quantization. *IEEE Transactions on Information Theory*, 55(7):3229–3242, 2009.

Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: spectral clustering and normalized cuts. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2004.

Chris Ding, Xiaofeng He, and Horst D Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *SIAM International Conference on Data Mining (SDM)*, 2005.

William E Donath and Alan J Hoffman. Algorithms for partitioning graphs and computer logic based on eigenvectors of connection matrices. *IBM Technical Disclosure Bulletin*, 15(3):938–944, 1972.

William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5):420–425, 1973.

Petros Drineas and Michael W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

Petros Drineas and Michael W Mahoney. RandNLA: randomized numerical linear algebra. *Communications of the ACM*, 59(6):80–90, 2016.

Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1): 132–157, 2006.

Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, September 2008.

Petros Drineas, Michael W. Mahoney, S. Muthukrishnan, and Tamás Sarlós. Faster least squares approximation. *Numerische Mathematik*, 117(2):219–249, 2011.

Petros Drineas, Malik Magdon-Ismail, Michael W. Mahoney, and David P. Woodruff. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13:3441–3472, 2012.

Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: constant-size coresets for k-means, PCA and projective clustering. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2013.

Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23 (2):298–305, 1973.

Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Michael R. Garey, D. S. Johnson, and Hans S. Witsenhausen. The complexity of the generalized Lloyd-Max problem. *IEEE Transactions on Information Theory*, 28(2):255–256, 1982.

Alex Gittens and Michael W. Mahoney. Revisiting the Nyström method for improved large-scale machine learning. *Journal of Machine Learning Research*, 17(1):3977–4041, 2016.

Alex Gittens, Jey Kottalam, Jiyan Yang, Michael F. Ringenburg, Jatin Chhugani, Evan Racah, Mohitdeep Singh, Yushu Yao, Curt Fischer, Oliver Ruebel, Benjamin Bowen, Norman G. Lewis, Michael W. Mahoney, Venkat Krishnamurthy, and Prabhat. A Multi-Platform Evaluation of the Randomized CX Low-Rank Matrix Factorization in Spark. In *IEEE International Parallel and Distributed Processing Symposium Workshops*, 2016.

Stephen Guattery and Gary L. Miller. On the performance of spectral graph partitioning methods. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1995.

Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

Mark S. Handcock, Adrian E. Raftery, and Jeremy M. Tantrum. Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 170(2):301–354, 2007.

Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Annual ACM Symposium on Theory of Computing (STOC)*, 2004.

Robert M. Haralick and Linda G. Shapiro. Image segmentation techniques. *Computer Vision, Graphics, and Image Processing*, 29(1):100–132, 1985.

Rong Jin, Tianbao Yang, Mehrdad Mahdavi, Y Li, and Z Zhou. Improved bounds for the Nyström method with application to kernel classification. *IEEE Transactions on Information Theory*, 59(10):6939–6949, 2013.

William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics*, 26(189-206), 1984.

Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. A local search approximation algorithm for k-means clustering. In *Annual Symposium on Computational Geometry*, 2002.

Amit Kumar, Yogish Sabharwal, and Sandeep Sen. A simple linear time $(1 + \epsilon)$-approximation algorithm for k-means clustering in any dimensions. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2004.

Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling methods for the Nyström method. *Journal of Machine Learning Research*, 13:981–1006, 2012.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Mu Li, Wei Bi, J.T. Kwok, and Bao-Liang Lu. Large-scale Nystr̈m kernel matrix approximation using randomized SVD. *IEEE Transactions on Neural Networks and Learning Systems*, 26(1):152–164, Jan 2015.

Yingyu Liang, Maria-Florina F Balcan, Vandana Kanchanapally, and David Woodruff. Improved distributed principal component analysis. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

Stuart Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

Yichao Lu, Paramveer Dhillon, Dean P Foster, and Lyle Ungar. Faster ridge regression via the subsampled randomized Hadamard transform. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is NP-hard. In *International Workshop on Algorithms and Computation*, pages 274–285. Springer, 2009.

Michael W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011.

Michael W. Mahoney and Petros Drineas. Structural properties underlying high-quality randomized numerical linear algebra algorithms. In P. Bühlmann, P. Drineas, M. Kane, and M. van de Laan, editors, *Handbook of Big Data*, pages 137–154. CRC Press, 2016.

Jiri Matousek. On approximate geometric k-clustering. *Discrete & Computational Geometry*, 24(1):61–84, 2000.

Xiangrui Meng and Michael W Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *Annual ACM Symposium on Theory of Computing (STOC)*, 2013.

Xiangrui Meng, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman, DB Tsai, Manish Amde, Sean Owen, Doris Xin, Reynold Xin, Michael J. Franklin, Reza Zadeh, Mateh Zaharia, and Ameet Talwalkar. MLlib: Machine Learning in Apache Spark. *Journal of Machine Learning Research*, 17(34):1–7, 2016.

Cameron Musco and Christopher Musco. Recursive sampling for the Nystrom method. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

John Nelson and Huy L Nguyên. OSNAP: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2013.

Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: analysis and an algorithm. *Advances in Neural Information Processing Systems (NIPS)*, 2002.

Evert J. Nyström. Über die praktische auflösung von integralgleichungen mit anwendungen auf randwertaufgaben. *Acta Mathematica*, 54(1):185–204, 1930.

Farhad Pourkamali-Anaraki and Stephen Becker. Randomized Clustered Nyström for Large-Scale Kernel Machines. *arXiv preprint arXiv:1612.06470*, 2016.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

Mark Rudelson and Roman Vershynin. The Littlewood–Offord problem and invertibility of random matrices. *Advances in Mathematics*, 218(2):600–633, 2008.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, 2002.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

Roded Sharan, Rani Elkon, and Ron Shamir. Cluster analysis and its applications to gene expression data. In *Bioinformatics and Genome Analysis*, pages 83–108. Springer, 2002.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

Si Si, Cho-Jui Hsieh, and Inderjit Dhillon. Memory efficient kernel approximation. In *International Conference on Machine Learning (ICML)*, 2014.

Mingjun Song and Sanguthevar Rajasekaran. Fast algorithms for constant approximation k-means clustering. *Transactions on Machine Learning and Data Mining*, 3(2):67–79, 2010.

Daniel A. Spielman and Shang-Hua Teng. Spectral partitioning works: Planar graphs and finite element meshes. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 1996.

Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3(Dec):583–617, 2002.

Terence Tao and Van Vu. Random matrices: The distribution of the smallest singular values. *Geometric And Functional Analysis*, 20(1):260–297, 2010.

Joel A Tropp. Improved analysis of the subsampled randomized hadamard transform. *Advances in Adaptive Data Analysis*, 3(01n02):115–126, 2011.

Joel A Tropp, Alp Yurtsever, Madeleine Udell, and Volkan Cevher. Fixed-rank approximation of a positive-semidefinite matrix from streaming data. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

L. H. Ungar and D. P. Foster. Clustering Methods for Collaborative Filtering. In *Proc. Recommender Systems, Papers from 1998 Workshop*, 1998. Technical Report WS-98-08.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Shusen Wang and Zhihua Zhang. Improving CUR matrix decomposition and the Nyström approximation via adaptive sampling. *Journal of Machine Learning Research*, 14:2729–2769, 2013.

Shusen Wang, Luo Luo, and Zhihua Zhang. SPSD matrix approximation vis column selection: Theories, algorithms, and extensions. *Journal of Machine Learning Research*, 17(49):1–49, 2016a.

Shusen Wang, Zhihua Zhang, and Tong Zhang. Towards more efficient SPSD matrix approximation and CUR matrix decomposition. *Journal of Machine Learning Research*, 17(210):1–49, 2016b.

Shusen Wang, Alex Gittens, and Michael W Mahoney. Sketched ridge regression: Optimization perspective, statistical perspective, and model averaging. In *International Conference on Machine Learning (ICML)*, 2017.

Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

Christopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

David P Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014.

Tianbao Yang, Yu-Feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou. Nyström method vs random Fourier features: A theoretical and empirical comparison. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster computing with working sets. *HotCloud*, 10(10-10):95, 2010.

Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J. Franklin, Scott Shenker, and Ion Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *USENIX Conference on Networked Systems Design and Implementation (NSDI)*, 2012.

Hongyuan Zha, Xiaofeng He, Chris Ding, Ming Gu, and Horst D Simon. Spectral relaxation for k-means clustering. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

Kai Zhang and James T. Kwok. Clustered Nyström method for large scale manifold learning and dimension reduction. *IEEE Transactions on Neural Networks*, 21(10):1576–1587, 2010.

Zhihua Zhang. The singular value decomposition, applications and beyond. *arXiv:1510.08532*, 2015.

Journal of Machine Learning Research 20 (2019) 1-34 Submitted 9/17; Revised 10/18; Published 2/19

Train and Test Tightness of LP Relaxations in Structured Prediction

Ofer Meshi MESHI@GOOGLE.COM
Google

Ben London BLONDON@AMAZON.COM
Amazon

Adrian Weller ADRIAN.WELLER@ENG.CAM.AC.UK
University of Cambridge and The Alan Turing Institute

David Sontag DSONTAG@CSAIL.MIT.EDU
MIT CSAIL

Editor: Ivan Titov

Abstract

Structured prediction is used in areas including computer vision and natural language processing to predict structured outputs such as segmentations or parse trees. In these settings, prediction is performed by MAP inference or, equivalently, by solving an integer linear program. Because of the complex scoring functions required to obtain accurate predictions, both learning and inference typically require the use of approximate solvers. We propose a theoretical explanation for the striking observation that approximations based on linear programming (LP) relaxations are often tight (exact) on real-world instances. In particular, we show that learning with LP relaxed inference encourages integrality of training instances, and that this training tightness generalizes to test data.

1. Introduction

Many applications of machine learning can be formulated as prediction problems over structured output spaces (Bakir et al., 2007; Nowozin et al., 2014). In such problems output variables are predicted *jointly* in order to take into account mutual dependencies between them, such as high-order correlations or structural constraints (e.g., matchings or spanning trees). Unfortunately, the improved expressive power of these models comes at a computational cost, and indeed, exact prediction and learning become NP-hard in general. Despite this worst-case intractability, efficient approximations often achieve very good performance in practice. In particular, one type of approximation which has proved effective in many applications is based on *linear programming (LP) relaxation*. In this approach the prediction problem is first cast as an integer LP (ILP), and then the integrality constraints are relaxed to obtain a tractable program. In addition to achieving high prediction accuracy, it has been observed that LP relaxations are often *tight* in practice. That is, the solution to the relaxed program happens to be optimal for the original hard problem (i.e., an *integral* solution is found). This is particularly surprising since the LPs have complex scoring functions that are not constrained to be from any tractable family. It has been an interesting open question to understand why these real-world instances behave so differently from the theoretical worst case.

This paper addresses this question and aims to provide a theoretical explanation for the frequent tightness of LP relaxations in the context of structured prediction. In particular, we show that an *approximate* training objective, although designed to produce accurate predictors, also induces tightness of the LP relaxation as a byproduct. Interestingly, our analysis also suggests that exact training may have the opposite effect. To explain tightness on future (i.e., test) instances, we prove several generalization bounds relating average tightness on the training data to expected tightness with respect to the generating distribution. Our bounds imply that if many predictions on training instances are tight, then predictions on test instances are also likely to be tight. Moreover, if predictions on training instances are *close to* integral solutions (in terms of L1 distance), then predictions on test instances will likely be similarly close to integral solutions. Our results help to understand previous empirical findings, and to our knowledge provide the first theoretical justification for the widespread success of LP relaxations for structured prediction in settings where the training data is not (algorithmically) separable.

2. Background

In this section we review the formulation of the structured prediction problem, its LP relaxation, and the associated learning problem. Consider a prediction task where the goal is to map a real-valued input vector x to a discrete output vector $y = (y_1, \ldots, y_n)$. In this work we focus on a simple class of models based on linear classifiers.[1] Particularly, in this setting prediction is performed via a linear discriminant rule: $y(x; w) = \arg\max_{y'} w^\top \phi(x, y')$, where $\phi(x, y) \in \mathbb{R}^d$ is a function mapping input-output pairs to feature vectors, and $w \in \mathbb{R}^d$ is the corresponding weight vector. Since the output space is often huge (exponential in n), it will generally be intractable to maximize over all possible outputs.

In many applications the score function has a particular structure. Specifically, we will assume that the score decomposes as a sum of simpler score functions: $w^\top \phi(x, y) = \sum_c w_c^\top \phi_c(x, y_c)$, where y_c is an assignment to a (non-exclusive) subset of the variables. For example, it is common to use such a decomposition that assigns scores to single and pairs of output variables corresponding to nodes and edges of a graph G: $w^\top \phi(x, y) = \sum_{i \in V(G)} w_i^\top \phi_i(x, y_i) + \sum_{ij \in E(G)} w_{ij}^\top \phi_{ij}(x, y_i, y_j)$. Viewing this as a function of y, we can write the prediction problem as:

$$\max_y \sum_c \theta_c(y_c; x, w) \ , \tag{1}$$

where $\theta_c(y_c; x, w) = w_c^\top \phi_c(x, y_c)$ (we will sometimes omit the dependence on x and w in the sequel).

Due to its combinatorial nature, the prediction problem is generally NP-hard, but fortunately, efficient approximations have been proposed. Here we will be particularly interested in approximations based on LP relaxations. We begin by formulating prediction

1. Most of our results also hold more generally for non-linear models (e.g., deep neural factors).

©2019 Ofer Meshi, Ben London, Adrian Weller, and David Sontag.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-535.html.

as the following ILP:[2]

$$\max_{\substack{\mu\in\mathcal{M}_L\\ \mu\in\{0,1\}^q}} \sum_c \sum_{y_c} \mu_c(y_c)\theta_c(y_c) + \sum_i \sum_{y_i} \mu_i(y_i)\theta_i(y_i) \quad = \theta^\top \mu \ , \tag{2}$$

$$\text{where } \mathcal{M}_L = \left\{ \mu \geq 0 : \begin{array}{ll} \sum_{y_{c\setminus i}} \mu_c(y_c) = \mu_i(y_i) & \forall c, i \in c, y_i \\ \sum_{y_i} \mu_i(y_i) = 1 & \forall i \end{array} \right\}.$$

Here, $\mu_c(y_c)$ is an indicator variable for a factor c and local assignment y_c, and q is the total number of factor assignments (dimension of μ). The set $\mathcal{M}_L$ is known as the local marginal polytope (Wainwright and Jordan, 2008), and $\sum_{y_{c\setminus i}} \mu_c(y_c)$ is the marginalization of μ_c w.r.t. variable i. First, notice that there is a one-to-one correspondence between feasible μ's and assignments y's, which is obtained by setting μ to indicators over local assignments (y_c and y_i) consistent with y. Second, while solving ILPs is NP-hard in general, it is easy to obtain a tractable program by relaxing the integrality constraints ($\mu \in \{0,1\}^q$), which may introduce fractional solutions to the LP. This relaxation, sometimes called the *basic linear programming relaxation* (Thapper and Živný, 2012), is the first level of the Sherali-Adams hierarchy (Sherali and Adams, 1990). This hierarchy provides successively tighter LP relaxations of an ILP. Notice that since the relaxed program is obtained by removing constraints from the original problem, its optimal value upper bounds the ILP optimum. Finally, we note that most of our results below also hold for cases where more complex constraints than those in Eq. (2) are used. For example, some assignments may be forbidden since they do not correspond to feasible global structures such as spanning trees (e.g., Martins et al., 2009b; Koo et al., 2010).

In order to achieve high prediction accuracy, the parameters w are learned from training data. In this supervised learning setting, the model is fit to labeled examples $\{(x^{(m)}, y^{(m)})\}_{m=1}^M$, where the goodness of fit is measured by a task-specific loss $\Delta(y(x^{(m)}; w), y^{(m)})$. We assume that $\Delta(y, y') \geq 0$ for all y, y', and that $\Delta(y, y) = 0$. For example, a commonly used task-loss is the Hamming distance: $\Delta_{\text{Hamming}}(y, y') = \frac{1}{n}\sum_i \mathbb{1}[y_i \neq y'_i]$.

In the *structured SVM* (SSVM) framework (Taskar et al., 2003; Tsochantaridis et al., 2004), the empirical risk is upper bounded by a convex surrogate called the structured hinge loss, which yields the training objective:

$$\min_w \ \sum_m \max_y \left[w^\top \Big(\phi(x^{(m)}, y) - \phi(x^{(m)}, y^{(m)}) \Big) + \Delta(y, y^{(m)}) \right] \ . \tag{3}$$

For brevity, we have omitted the standard regularization term from Eq. (3), however, all of our results below in sections 4—6 still hold with regularization.[3] The objective in Eq. (3) is a convex function of w and hence can be optimized in various ways. But, notice that the objective includes a maximization over outputs y for each training example. This loss-augmented prediction task needs to be solved repeatedly during training (e.g., to evaluate subgradients), which makes training intractable in general (see also Sontag et al., 2010). Similar to prediction, LP relaxation can be applied to the structured loss (Taskar et al., 2003; Kulesza and Pereira, 2007), which yields the relaxed training objective:

$$\min_w \ \sum_m \max_{\mu\in\mathcal{M}_L} \left[\theta_m^\top(\mu - \mu_m) + {\ell_m}^\top \mu \right] \ , \tag{4}$$

where $\theta_m \in \mathbb{R}^q$ is a score vector in which each entry represents $w_c^\top \phi_c(x^{(m)}, y_c)$ for some c and y_c, and μ_m is the integral vector corresponding to $y^{(m)}$. Assuming that the task-loss decomposes as the model score, $\Delta(y, y') = \sum_c \Delta_c(y_c, y'_c)$, we define the vector $\ell_m \in \mathbb{R}^q$ with entries $\ell_{m,c,y_c} = \Delta_c(y_c^{(m)}, y_c)$ for each value y_c. Notice that ℓ has the same dimension as μ, and we can define $\Delta(y^{(m)}, y) = \sum_c \sum_{y'_c} \Delta_c(y_c^{(m)}, y'_c)\mu_c(y'_c) = {\ell_m}^\top \mu$, where μ is the vector of indicators corresponding to y (i.e., $\mu_c(y'_c) = \mathbb{1}\{y'_c = y_c\}$). With this definition, ℓ generalizes Δ to any $\mu \in \mathcal{M}_L$.

After reviewing related work in Section 3, we propose a theoretical justification for the observed tightness of LP relaxations for structured prediction. To this end, we make two complementary arguments: in Section 4 we argue that optimizing the relaxed training objective of Eq. (4) also has the effect of encouraging tightness of training instances; then, in sections 5 and 6 we show that tightness generalizes from train to test data.

3. Related Work

Many structured prediction problems can be expressed as ILPs (Roth and Yih, 2005; Martins et al., 2009a; Rush et al., 2010). Despite being NP-hard in general (Roth, 1996; Shimony, 1994), various effective approximations have been proposed. These approximations include search-based methods (Daumé III et al., 2009; Zhang et al., 2014) and natural LP relaxations to the hard ILPs (Schlesinger, 1976; Koster et al., 1998; Chekuri et al., 2004; Wainwright et al., 2005). Tightness of LP relaxations for special classes of problems has been studied extensively in recent years and has been demonstrated by restricting either the structure of the model or its score function. For example, the pairwise LP relaxation is known to be tight for tree-structured models or for supermodular scores (see, e.g., Wainwright and Jordan, 2008; Thapper and Živný, 2012); certain stability conditions guarantee tightness of LP relaxations for Ferromagnetic Potts models (Lang et al., 2018); and the cycle relaxation (for binary pairwise models) is known to be tight both for planar Ising models with no external field (Barahona, 1993) and for almost balanced models (Weller et al., 2016). Hybrid conditions, combining structure and score, by forbidding signed minors have recently been shown to also guarantee tight relaxations (Rowland et al., 2017; Weller, 2016). To facilitate efficient prediction, one could restrict the model class to be tractable. For example, Taskar et al. (2004) learn supermodular scores, and Meshi et al. (2013) learn tree structures.

However, the sufficient conditions mentioned above are by no means necessary, and indeed, many score functions that are useful in practice do not satisfy them but still produce integral solutions (Roth and Yih, 2004; Lacoste-Julien et al., 2006; Sontag et al., 2008; Finley and Joachims, 2008; Martins et al., 2009b; Koo et al., 2010). For example, Martins et al. (2009b) showed that predictors that are learned with LP relaxations yield tight LPs for 92.88% of the test data on a dependency parsing problem (see Table 2 therein). Koo et al. (2010) observed similar behavior for dependency parsing on a number of languages, as can

2. For convenience we introduce singleton factors θ_i, which could be set to 0 if needed.
3. In particular, our bounds apply to the maximization over y in Eq. (3), so still hold when regularization w.r.t. w is added.

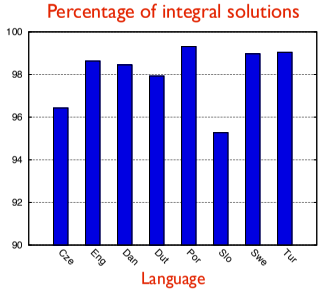


Figure 1: Percentage of integral solutions for dependency parsing from Koo et al. (2010).

be seen in Fig. 1.[4] Lacoste-Julien et al. (2006) found that about 80% of test instances had tight LP relaxations in a quadratic assignment formulation for word alignment, with only 0.2% fractional values overall. The same phenomenon has been observed for a multi-label classification task, where test integrality reached 100% (Finley and Joachims, 2008, Table 3).

Learning structured output predictors from labeled data was proposed in various forms by Collins (2002); Taskar et al. (2003); Tsochantaridis et al. (2004). These formulations generalize training methods for binary classifiers, such as the Perceptron algorithm and support vector machines (SVMs), to the case of structured outputs. The learning algorithms repeatedly perform prediction, necessitating the use of approximate inference within training as well as at test time. A common approach, introduced right at the inception of structured SVMs by Taskar et al. (2003), is to use LP relaxations for this purpose.

The closest work to ours is by Kulesza and Pereira (2007), which showed that not all approximations are equally good, and that it is important to match the inference algorithms used at train and test time. The authors defined the concept of *algorithmic separability* which refers to the case where an approximate inference algorithm achieves zero loss on a data set. The authors studied the use of LP relaxations for structured learning, giving generalization bounds for the true risk of LP-based prediction. However, since the generalization bounds in Kulesza and Pereira (2007) are focused on prediction *accuracy*, the only settings in which tightness on test instances can be guaranteed are when the training data is algorithmically separable, which is seldom the case in real-world structured prediction tasks (the models are far from perfect). In contrast, our paper's main result (Theorem 1), guarantees that the expected fraction of test instances for which an LP relaxation is tight is close to that which was achieved on training data. This then allows us to talk about the generalization of *computation*. For example, suppose one uses LP relaxation-based algorithms that iteratively tighten the relaxation, such as Sontag and Jaakkola (2008); Sontag et al. (2008), and observes that 20% of the instances in the training data are integral using the basic relaxation and that after tightening the remaining 80% are now integral too. Our generalization bound then guarantees that approximately the same ratio will hold at test time (assuming sufficient training data).

4. Kindly provided by the authors.

Finley and Joachims (2008) also studied the effect of various approximate inference methods in the context of structured prediction. Their theoretical and empirical results support the superiority of LP relaxations in this setting. Martins et al. (2009b) established conditions which guarantee algorithmic separability for LP relaxed training, and derived risk bounds for a learning algorithm which uses a combination of exact and relaxed inference.

Finally, Globerson et al. (2015) recently studied the performance of structured predictors for 2D grid graphs with binary labels from an information-theoretic perspective. They proved lower bounds on the minimum achievable expected Hamming error in this setting, and proposed a polynomial-time algorithm that achieves this error. Our work is different since we focus on LP relaxations as an approximation algorithm, we handle a general form of the problem without making any assumptions on the model or error measure (except score decomposition), and we concentrate solely on the computational aspects while ignoring any accuracy concerns.

4. Tightness at Training

In this section we show that the *relaxed* training objective in Eq. (4), although designed to achieve high accuracy, also induces tightness of the underlying LP relaxation. In fact, although we focus here on the basic LP relaxation (first-level), the results below hold for higher-level LP relaxations as well.[5]

In order to simplify notation we focus on a single training instance and drop the index m. Denote the solutions to the relaxed and integer LPs as:

$$\mu_L \in \arg\max_{\mu \in \mathcal{M}_L} \theta^\top \mu \qquad \mu_I \in \arg\max_{\substack{\mu \in \mathcal{M}_L \\ \mu \in \{0,1\}^q}} \theta^\top \mu \ , \tag{5}$$

respectively. Also, let μ_T be the integral vector corresponding to the ground-truth output $y^{(m)}$. Now consider the following decomposition:

$$\underset{\text{relaxed-hinge}}{\theta^\top(\mu_L - \mu_T)} = \underset{\text{integrality gap}}{\theta^\top(\mu_L - \mu_I)} + \underset{\text{exact-hinge}}{\theta^\top(\mu_I - \mu_T)} \tag{6}$$

This equality states that the difference in scores between the relaxed optimum and ground-truth (*relaxed-hinge*) can be written as a sum of the *integrality gap* and the difference in scores between the exact optimum and the ground-truth (*exact-hinge*); notice that all three terms are non-negative. This simple decomposition has several interesting implications.

First, we can immediately derive the following bound on the integrality gap:

$$\begin{aligned} \theta^\top(\mu_L - \mu_I) &= \theta^\top(\mu_L - \mu_T) - \theta^\top(\mu_I - \mu_T) && (7)\\ &\leq \theta^\top(\mu_L - \mu_T) && (8)\\ &\leq \theta^\top(\mu_L - \mu_T) + \ell^\top \mu_L && (9)\\ &\leq \max_{\mu \in \mathcal{M}_L} \left(\theta^\top(\mu - \mu_T) + \ell^\top \mu\right), && (10) \end{aligned}$$

where Eq. (10) is precisely the relaxed training objective from Eq. (4). Therefore, optimizing the approximate training objective of Eq. (4) minimizes an upper bound on the integrality

5. Similar analysis for SDP and quadratic relaxations is left as future work—see recent work by Lê-Huu and Paragios (2018).

gap. Hence, driving down the approximate objective may also reduce the integrality gap of training instances, although in Section 4.1 we also study cases where loose bounds can lead to non-zero integrality gap. One case where the integrality gap becomes zero is when the data is algorithmically separable (i.e., $\mu_L = \mu_T$, so Eq. (8) equals 0). In this case the relaxed-hinge term vanishes (the exact-hinge must also vanish), and training integrality is assured.

Second, Eq. (6) holds for *any integral* μ, and not just the ground-truth μ_T. In other words, the only property of μ_T used here is its integrality. Indeed, in Section 7 we verify empirically that training a model using *random labels* still attains the same level of tightness as training with the ground-truth labels.[6] On the other hand, accuracy drops drastically, as expected. This analysis suggests that *tightness is not coupled with accuracy* of the predictor. Finley and Joachims (2008) explained tightness of LP relaxations by noting that fractional solutions always incur a loss during training. Our analysis suggests an alternative explanation, emphasizing the difference in scores (Eq. (7)) rather than the loss, and decoupling tightness from accuracy.

Third, the results above still hold in the presence of global constraints. Often such constraints can be expressed as linear inequalities, so in this case the local polytope $\mathcal{M}_L$ can be redefined by adding these constraints to those in Eq. (2) to form a tighter polytope. In particular, Eq. (8) holds since μ_T satisfies the global constraints.

Finally, we do not make any assumption here about the form of the model scores θ. Therefore, these results apply more generally, even when factor scores are obtained from non-linear functions of the inputs, such as deep neural networks (e.g., Chen et al., 2015).

4.1. When Relaxed Training is Better than Exact Training

Unfortunately, the bound in Eq. (10) might sometimes be loose. Indeed, to get the bound we have discarded the exact-hinge term (Eq. (8)), added the task-loss (Eq. (9)), and maximized the loss-augmented objective (Eq. (10)). At the same time, Eq. (7) provides a precise characterization of the integrality gap. Specifically, the gap is determined by the difference between the relaxed-hinge and the exact-hinge terms. This implies that even when the relaxed-hinge is not zero, a small integrality gap can still be attained if the exact-hinge is also large. In fact, the *only way* to get a large integrality gap is by setting the exact-hinge much smaller than the relaxed-hinge. But when can this happen? As we now show, it is less likely to happen when training with relaxed inference than when training with exact inference.

A key point is that the relaxed and exact hinge terms are upper bounded by the relaxed and exact *training objectives* respectively (the latter additionally depend on the task loss Δ). Therefore, minimizing the training objective will also likely reduce the corresponding hinge term (this is also demonstrated empirically in Section 7). Using this insight, we observe that relaxed training reduces the relaxed-hinge term without directly reducing the exact-hinge term, and thereby induces a small integrality gap. On the other hand, this also suggests that *exact training may actually increase the integrality gap*, since it reduces the exact-hinge without also reducing directly the relaxed-hinge term. This finding is consistent with previous empirical evidence. Specifically, Martins et al. (2009b, Table 2) showed that on

6. This is not true for random models (w), which often yield loose relaxations.

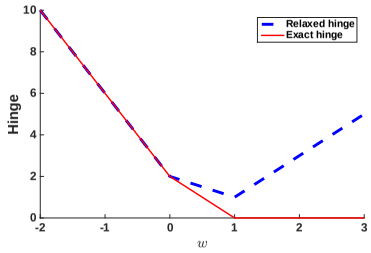


Figure 2: Exact- and relaxed- hinge (see Eq. (6)) as a function of w for the learning scenario in Section 4.1.

a dependency parsing problem, training with the relaxed objective achieved 92.9% integral solutions, while exact training achieved only 83.5% integral solutions. An even stronger effect was observed by Finley and Joachims (2008, Table 3) for multi-label classification, where relaxed training resulted in 99.6% integral instances, with exact training attaining only 17.7% ('Yeast' dataset).

In Section 7 we provide further empirical support for our explanation, however, we next show its possible limitation by providing a counter-example. The counter-example demonstrates that despite training with a relaxed objective, the exact-hinge can in some cases actually be *substantially smaller* than the relaxed-hinge, leading to a loose relaxation. Although this illustrates the limitations of the explanation above, we point out that the corresponding learning task is far from natural.

We construct a learning scenario where relaxed training obtains zero exact-hinge and non-zero relaxed-hinge, so the relaxation is not tight. Consider a model where $x \in \mathbb{R}^3$, $y \in \{0,1\}^3$, and the prediction is given by:

$$y(x;w) = \arg\max_y \Big(x_1 y_1 + x_2 y_2 + x_3 y_3 + w\left[\mathbb{1}\{y_1 \neq y_2\} + \mathbb{1}\{y_1 \neq y_3\} + \mathbb{1}\{y_2 \neq y_3\}\right] \Big). \tag{11}$$

The corresponding LP relaxation is then:

$$\max_{\mu \in \mathcal{M}_L} \Big(x_1\mu_1(1) + x_2\mu_2(1) + x_3\mu_3(1) + w\left[\mu_{12}(01) + \mu_{12}(10) + \mu_{13}(01) + \mu_{13}(10) + \mu_{23}(01) + \mu_{23}(10)\right] \Big). \tag{12}$$

Next, we construct a training set where the first instance is: $x^{(1)} = (2,2,2), y^{(1)} = (1,1,0)$, and the second is: $x^{(2)} = (0,0,0), y^{(2)} = (1,1,0)$. Fig. 2 shows the relaxed and exact losses as a function of w, obtained by plugging Eq. (11) and Eq. (12) in Eq. (3) and Eq. (4), respectively.[7] Observe that $w = 1$ minimizes the relaxed objective (Eq. (4)). However, with this weight vector the relaxed-hinge for the second instance $(x^{(2)}, y^{(2)})$ is equal to 1, while the exact-hinge for this instance is 0 (the data is separable with $w = 1$). Consequently, there is an integrality gap of 1 for the second instance, and the relaxation is loose (the first

7. For simplicity we use $\ell = 0$ in this example, but a similar result holds with $\ell = 1/2 \cdot \ell_{\text{Hamming}}$.

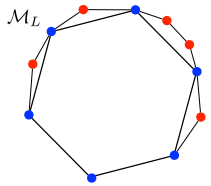


Figure 3: Illustration of the local marginal polytope $\mathcal{M}_L$, with its vertices partitioned into integral vertices $\mathcal{V}_I$ (blue), and fractional vertices $\mathcal{V}_F$ (red).

instance is actually tight). Notice that in this data the same output corresponds to two very different inputs.

5. Generalization of Tightness

Our argument in Section 4 concerns only the tightness of train instances. However, the empirical evidence discussed above pertains to test data. To bridge this gap, in this section we prove a generalization bound which shows that train tightness implies test tightness.

We first define a loss function which measures the lack of integrality (or, fractionality) of the LP solution for a given instance. To this end, we consider the discrete set of *vertices* of the local polytope $\mathcal{M}_L$ (excluding its convex hull), denoting by $\mathcal{V}_I$ and $\mathcal{V}_F$ the sets of fully-integral and non-integral (i.e., fractional) vertices,[8] respectively (so $\mathcal{V}_I \cap \mathcal{V}_F = \emptyset$, and $\mathcal{V}_I \cup \mathcal{V}_F$ consists of all vertices of $\mathcal{M}_L$);[9] see Fig. 3. Considering only vertices does not reduce generality, since the solution to a linear program is always at a vertex.

Next, let

$$I^*(w,x) \triangleq \max_{\mu \in \mathcal{V}_I} \theta(x;w)^\top \mu \qquad \text{and} \qquad F^*(w,x) \triangleq \max_{\mu \in \mathcal{V}_F} \theta(x;w)^\top \mu \tag{13}$$

denote the respective best integral and fractional scores. By convention, we set $F^*(w,x) \triangleq -\infty$ whenever $\mathcal{V}_F = \emptyset$. The fractionality of inference with (w,x) can be measured by the quantity

$$D(w,x) \triangleq F^*(w,x) - I^*(w,x). \tag{14}$$

Observe that $D(w,x) > 0$ whenever the LP has a fractional solution that is better than the integral solution. We can now define the *integrality loss*,

$$\mathcal{L}_0(w,x) \triangleq \begin{cases} 1 & D(w,x) > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{15}$$

8. It is enough that one coordinate is fractional to belong to $\mathcal{V}_F$.
9. We assume that all feasible integral solutions are vertices of $\mathcal{M}_L$, which is the case for the type of relaxations considered here (see Wainwright and Jordan, 2008).

This loss function equals 1 if and only if the optimal fractional solution has a (strictly) higher score than the optimal integral solution. The loss will be 0 whenever the non-integral and integral optima are equal—that is, for our purpose we consider the relaxation to be tight in this case. The expected integrality loss measures the probability of obtaining a fractional LP solution (over draws of an input, x). Note that this loss ignores the ground truth assignment.

To support our generalization analysis, we define a related loss function, which we call the *integrality ramp loss*. For a predetermined margin parameter, γ, the integrality ramp loss is given by

$$\mathcal{L}_\gamma(w,x) \triangleq \begin{cases} 1 & D(w,x) > 0 \\ 1 + D(w,x)/\gamma & -\gamma < D(w,x) \leq 0 \\ 0 & D(w,x) \leq -\gamma \end{cases}. \tag{16}$$

Importantly, the integrality ramp loss upper-bounds the integrality loss. For the ramp loss to be zero, the best integral solution has to be better than the best fractional solution by a margin of at least γ, which is a stronger requirement than mere tightness. In Appendix A we give examples of models that are guaranteed to satisfy this requirement, and in Section 7 we also show this often happens in practice.

We point out that both $\mathcal{L}_0(w,x)$ and $\mathcal{L}_\gamma(w,x)$ are generally hard to compute, a point which we address in Section 6. For the time being, we are only interested in proving that tightness is a generalizing property, so we will not worry about computational efficiency.

We are now ready to state the main theorem of this section, a generalization bound for tightness. Our proof (deferred to Appendix B.1) uses a PAC-Bayesian analysis, similar to London et al. (2016), though the main result is stated for a deterministic predictor.

Theorem 1. *Let $\mathbb{D}$ denote a distribution over $\mathcal{X}$. Let $\phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ denote a feature mapping such that $\sup_{x,y} \|\phi(x,y)\|_2 \leq B < \infty$. Then, for any $\gamma > 0$, $\delta \in (0,1)$ and $m \geq 1$, with probability at least $1-\delta$ over draws of $(x^{(1)}, \ldots, x^{(m)}) \in \mathcal{X}^m$, according to $\mathbb{D}^m$, every weight vector, w, with $\|w\|_2 \leq R < \infty$, satisfies*

$$\mathop{\mathbb{E}}_{x \sim \mathbb{D}}[\mathcal{L}_0(w,x)] \leq \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}_\gamma(w, x^{(i)}) + \frac{8}{m} + 2\sqrt{\frac{d \ln(mBR/\gamma) + \ln \frac{2}{\delta}}{2m}}. \tag{17}$$

Theorem 1 shows that if we observe high integrality (equivalently, low fractionality) on a finite sample of training data, then it is likely that integrality of test data will not be much lower, provided sufficient number of samples. It is worth noting that, though we focus on linear models to simplify our presentation, it is possible to extend this result to accommodate non-linear models (e.g., scores θ computed by a deep neural network), provided some assumptions are made on the smoothness of the model and loss.

As the following corollary states, Theorem 1 actually applies more generally to any two disjoint sets of vertices, and is not limited to $\mathcal{V}_I$ and $\mathcal{V}_F$.

Corollary 1. *Let $\mathcal{V}_\alpha$ be any set of vertices of $\mathcal{M}_L$ with at most α fractional values (where $0 \leq \alpha \leq 1$), and let $\bar{\mathcal{V}}_\alpha$ be the rest of the vertices of $\mathcal{M}_L$. Then Theorem 1 holds with $\mathcal{V}_\alpha$ and $\bar{\mathcal{V}}_\alpha$ replacing $\mathcal{V}_I$ and $\mathcal{V}_F$ in the definition of I^* and F^* in Eq. (13), respectively.*

For example, we can set $\mathcal{V}_\alpha$ to be any set of vertices with at most 10% fractional values, and $\bar{\mathcal{V}}_\alpha$ to be the rest of the vertices of $\mathcal{M}_L$. This gives a different meaning to the integrality loss, but the rest of our analysis holds unchanged. Consequently, our generalization result implies that it is likely to observe a similar portion of instances with at most 10% fractional values at test time as we did at training.

Moreover, Theorem 1 also holds in the presence of global constraints (e.g., spanning tree constraints). As mentioned in Section 4, the polytope $\mathcal{M}_L$ is replaced by its intersection with the global constraints polytope, but the rest of our derivation remains unchanged.

Note that the loss function in Eq. (15) does not measure the actual number of fractional values, nor their *distance* to integrality. In Section 6, we analyze a notion of tightness that accounts for the L1 distance to integrality.

Compared to the generalization bound of Kulesza and Pereira (2007), our bounds only consider the tightness of a prediction, ignoring label errors. Thus, for example, if learning happens to settle on a set of parameters in a tractable regime (e.g., supermodular potentials or stable instances (Makarychev et al., 2014)) for which the LP relaxation is tight for most training instances, our generalization bound guarantees that with high probability the LP relaxation will also be tight on most test instances. In contrast, in Kulesza and Pereira (2007), tightness on test instances can only be guaranteed when the training data is algorithmically separable (i.e., LP-relaxed inference predicts perfectly).

5.1. γ-Tight Relaxations

In this section we study the stronger notion of tightness required by our surrogate fractionality loss (Eq. (16)), and show examples of models that satisfy it.

Definition 1. An LP relaxation is called γ-*tight* if $I^*(w,x) \geq F^*(w,x) + \gamma$ (so $\mathcal{L}_\gamma(w,x) = 0$). That is, the best integral value is larger than the best non-integral value by at least γ.[10]

We focus on binary pairwise models and show two cases where the model is guaranteed to be γ-tight. Proofs are provided in Appendix A. Our first example involves *balanced* models, which are binary pairwise models that have supermodular scores, or can be made supermodular by "flipping" a subset of the variables (for more details, see Appendix A).

Proposition 1. *A balanced model with a unique optimum is* $(\alpha/2)$*-tight, where* α *is the difference between the best and second-best (integral) solutions.*

This result is of particular interest when learning structured predictors where the edge scores depend on the input. Whereas one could learn supermodular models by enforcing linear inequalities (Taskar et al., 2004), we know of no tractable means of ensuring the model is balanced. Instead, one could learn over the full space of models using LP relaxation. If the learned models are balanced on the training data, Proposition 1 together with Theorem 1 tell us that the LP relaxation is likely to be tight on test data as well.

Our second example regards models with singleton scores that are much stronger than the pairwise scores. Consider a binary pairwise model[11] in minimal representation, where $\bar{\theta}_i$ are node scores and $\bar{\theta}_{ij}$ are edge scores in this representation (see Appendix A for full details). Further, for each variable i, define the set of neighbors with *attractive* edges $N_i^+ = \{j \in N_i | \bar{\theta}_{ij} > 0\}$, and the set of neighbors with *repulsive* edges $N_i^- = \{j \in N_i | \bar{\theta}_{ij} < 0\}$.

10. Notice that scaling up $\theta(w,x)$ will also increase γ, but our bound in Eq. (17) also grows with the norm of $\theta(w,x)$ (via the term BR). Therefore, we assume here that $\|\theta(w,x)\|_2$ is bounded.
11. This case easily generalizes to variables with more than 2 possible values.

Proposition 2. *If all variables satisfy the condition:*

$$\bar{\theta}_i \geq -\sum\nolimits_{j \in N_i^-} \bar{\theta}_{ij} + \beta, \quad \text{or} \quad \bar{\theta}_i \leq -\sum\nolimits_{j \in N_i^+} \bar{\theta}_{ij} - \beta$$

for some $\beta > 0$*, then the model is* $(\beta/2)$*-tight.*

Finally, we point out that in both of the examples above, the conditions can be verified efficiently and if they hold, the value of γ can be computed efficiently.

6. Analysis of the Integrality Distance

For structured prediction, the maximizer, μ_L, is often more important than the maximum value, $\theta^\top \mu_L$. That is, we do not really care whether the optimum of the relaxed problem equals that of the integral one; we just want relaxed inference to yield the optimal integral assignment—or, lacking that, an assignment that is "close to" the optimal integral one. If we assume that the relaxed problem has a unique solution, then an integrality gap of zero implies that the assignments are the same. However, lacking this assumption, there may be multiple, disparate solutions, so the assignments may differ. In general, it is difficult to characterize the distance between relaxed and exact assignments as a function of a nonzero integrality gap.

Thus, in addition to studying the integrality gap, we are also interested in what we will call the *integrality distance*, defined as $\|\mu_L - \mu_I\|_1$, which is the Manhattan distance between the maximizers of the relaxed and exact programs. The integrality distance is conceptually similar to *persistence* (see Wainwright and Jordan, 2008 for definition) in that a persistent fractional solution will have a subset of variables with zero integrality distance. The integrality distance is also related to the integrality gap, although the distance could sometimes be more useful: when the integrality distance is small, the relaxed solution is close to the exact solution, which is what we may ultimately care about; moreover, when the integrality distance is zero, the integrality gap must also be zero, regardless of whether we assume uniqueness.

In this section, we relate the integrality distance to several loss functions that are commonly analyzed in the literature on structured prediction. We then show that, similar to the integrality gap, the integrality distance also generalizes from an empirical sample to the population average. Moreover, we show that the integrality distance is upper-bounded by a constant multiple of the structured hinge loss—a convex loss function that is commonly used for training. Importantly, unlike the bound in Theorem 1, this bound is computationally tractable. Combining these results, we obtain a high-probability bound on the expected integrality distance that can be efficiently evaluated from training data, and whose additive error decreases with the number of examples. Finally, a simple argument shows how this bound applies to an integral rounding of a fractional solution.

6.1. Structured Loss Functions

We will focus on the integrality distance of the singleton (i.e., node) marginals, denoted $\mu_{\mathrm{u}} \triangleq (\mu_i)_{i=1}^n$, since they are sufficient for decoding a labeling, y. Let

$$D_1(\mu, \mu') \triangleq \frac{1}{2n} \left\| \mu_{\mathrm{u}} - \mu'_{\mathrm{u}} \right\|_1 \tag{18}$$

denote the normalized Manhattan distance. When both inputs are integral, D_1 is equivalent to the normalized Hamming distance.

Given a model, w, an input, x, and an assignment, μ, let

$$\mathcal{L}_1(w, x, \mu) \triangleq D_1\left(\mu, \mu_L(x; w)\right) \tag{19}$$

denote the L_1 *loss*. This loss function is a generalization of the Hamming loss, which is commonly used to measure the prediction error of exact inference. If the third argument is a reference (i.e., "ground truth") labeling, μ_T, then $\mathcal{L}_1(w, x, \mu_T)$ measures the prediction error of approximate inference. However, if the third argument is the exact, integral MAP state, μ_I, then $\mathcal{L}_1(w, x, \mu_I)$ is the normalized integrality distance. This latter quantity is what we will focus on upper-bounding.

Let

$$\mathcal{L}_{\mathrm{h}}(w, x, \mu) \triangleq \max_{\mu' \in \mathcal{M}_L} D_1(\mu, \mu') + \theta^\top \left(\mu' - \mu\right). \tag{20}$$

denote a loss function commonly referred to as the (relaxed) *structured hinge loss*. This loss is minimized when μ scores higher than all alternate assignments, μ', by a margin that is at least $D_1(\mu, \mu')$. Note that when μ is the exact MAP state, the structured hinge loss computes a *loss-augmented* integrality gap, using Manhattan distance for loss augmentation (see Eq. (4)).

A related loss function is the (relaxed) *structured ramp loss*,

$$\mathcal{L}_{\mathrm{r}}(w, x, \mu) \triangleq \max_{\mu' \in \mathcal{M}_L} D_1(\mu, \mu') + \theta^\top \left(\mu' - \mu_L(x; w)\right), \tag{21}$$

which can be considered a normalized version of the hinge loss. $\mathcal{L}_{\mathrm{r}}$ is bounded by $[0, 1]$, whereas $\mathcal{L}_{\mathrm{h}}$ might be unbounded (depending on the features and weights).

The hinge loss is often used in max-margin training, since it is convex in w. The ramp loss is not convex in w, but it is bounded, Lipschitz, and has a convenient relationship to the L_1 (or Hamming) and hinge losses:

$$\mathcal{L}_1(w, x, \mu) \leq \mathcal{L}_{\mathrm{r}}(w, x, \mu) \leq \mathcal{L}_{\mathrm{h}}(w, x, \mu). \tag{22}$$

Thus, the ramp loss is often used as an analytical tool to derive generalization bounds, such as those that follow.

6.2. Generalization Bound

Similar to Section 5, we now show that the integrality distance on a training sample generalizes to the data distribution. Like Theorem 1, the proof (in Appendix B.2) uses PAC-Bayesian analysis.

Theorem 2. *Let $\mathbb{D}$ denote a distribution over $\mathcal{X}$. Let $\phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ denote a feature mapping such that $\sup_{x,y} \|\phi(x, y)\|_2 \leq B < \infty$; and μ_I is defined in Eq. (5). Then, for any $\delta \in (0, 1)$ and $m \geq 1$, with probability at least $1 - \delta$ over draws of $(x^{(1)}, \ldots, x^{(m)}) \in \mathcal{X}^m$, according to $\mathbb{D}^m$, every weight vector, w, with $\|w\|_2 \leq R < \infty$, satisfies*

$$\mathop{\mathbb{E}}_{x \sim \mathbb{D}}[\mathcal{L}_1(w, x, \mu_I)] \leq \frac{1}{m} \sum_{i=1}^m \mathcal{L}_r(w, x^{(i)}, \mu_I^{(i)}) + \frac{8}{m} + 2\sqrt{\frac{d \ln(mBR) + \ln \frac{2}{\delta}}{2m}}. \tag{23}$$

Further, Eq. (23) holds when $\mathcal{L}_r$ is replaced with $\mathcal{L}_h$.

Theorem 2 says that the integrality distance on the training set generalizes to future examples. More precisely, the expected integrality distance on a random instance is upper-bounded by the average integrality ramp (or hinge) loss on the training set, plus two terms that vanish as the number of training examples grows. Thus, the more training data we have, the better we can estimate the expected integrality distance.

Remark 1. There is nothing special about μ_I to Theorem 2. Indeed, we could use any integral assignment as a reference labeling for the loss functions and the proof would be the same. For example, we could replace μ_I with μ_T (a ground truth labeling) and obtain a risk bound for learning with approximate inference, which is a well-studied topic (e.g., Kulesza and Pereira, 2007; London et al., 2016).

6.3. Relationship to Max-Margin Training

In practice, computing the integrality loss is generally infeasible, since it requires exact inference. Therefore, the upper bounds in Theorems 1 and 2 cannot be evaluated. However, the empirical relaxed hinge loss with respect to the ground truth labels *can* be evaluated efficiently. In this section, we show how minimizing this quantity actually minimizes the integrality distance. That is, max-margin training with approximate inference—which is commonly used anyway to learn graphical models—reduces not only the prediction error, but also the inference approximation error.

The key insight that enables this result comes from the following technical lemma.

Lemma 1. *For any w and x, if μ_T is the reference (ground truth) labeling of x and μ_I is the exact MAP state under w, then*

$$\mathcal{L}_h(w, x, \mu_I) \;\leq\; 2\,\mathcal{L}_h(w, x, \mu_T), \tag{24}$$

meaning the integrality hinge loss is at most twice the hinge loss with respect to the true labeling.

Proof First, we decompose the integrality hinge loss as follows:

$$\begin{aligned}
\mathcal{L}_{\mathrm{h}}(w, x, \mu_I) &= \max_{\mu \in \mathcal{M}_L} D_1(\mu_I, \mu) + \theta^\top (\mu - \mu_I) \\
&\leq \max_{\mu \in \mathcal{M}_L} D_1(\mu_I, \mu) + \theta^\top (\mu - \mu_T) \\
&\leq D_1(\mu_I, \mu_T) + \max_{\mu \in \mathcal{M}_L} D_1(\mu_T, \mu) + \theta^\top (\mu - \mu_T) \\
&= D_1(\mu_I, \mu_T) + \mathcal{L}_{\mathrm{h}}(w, x, \mu_T).
\end{aligned} \tag{25}$$

The second term on the right-hand side is the hinge loss of the approximate predictor with respect to the true labeling, which can be evaluated efficiently. The first term on the right-hand side is the Hamming loss of exact inference, which cannot be evaluated efficiently. However, this latter quantity can be upper-bounded as follows:

$$\begin{aligned} D_1(\mu_I, \mu_T) &\leq \max_{\mu \in \mathcal{M}} D_1(\mu, \mu_T) + \theta^\top (\mu - \mu_T) \\ &\leq \max_{\mu \in \mathcal{M}_L} D_1(\mu, \mu_T) + \theta^\top (\mu - \mu_T) \\ &= \mathcal{L}_{\mathrm{h}}(w, x, \mu_T). \end{aligned} \tag{26}$$

Combining Eq. (25) and (26) completes the proof. ■

Note that Lemma 1 also yields an upper bound on the integrality ramp loss, since it is upper-hounded by the integrality hinge loss.

Using Lemma 1, we thus obtain the following corollary of Theorem 2.

Corollary 2. *Let $\mathbb{D}$ denote a distribution over $\mathcal{X} \times \mathcal{Y}$. Let $\phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ denote a feature mapping such that $\sup_{x,y} \|\phi(x,y)\|_2 \leq B < \infty$. Then, for any $\delta \in (0,1)$ and $m \geq 1$, with probability at least $1-\delta$ over draws of $(x^{(1)}, y^{(1)}, \ldots, x^{(m)}, y^{(m)}) \in (\mathcal{X} \times \mathcal{Y})^m$, according to $\mathbb{D}^m$, every weight vector, w, with $\|w\|_2 \leq R < \infty$, satisfies*

$$\mathop{\mathbb{E}}_{x \sim \mathbb{D}}[\mathcal{L}_1(w, x, \mu_I)] \leq \frac{2}{m} \sum_{i=1}^{m} \mathcal{L}_h(w, x^{(i)}, \mu_T^{(i)}) + \frac{8}{m} + 2\sqrt{\frac{d \ln(mBR) + \ln \frac{2}{\delta}}{2m}}, \tag{27}$$

where $\mu_T^{(i)}$ denotes the integral vector corresponding to the ground-truth labeling $y^{(i)}$.

Corollary 2 says that max-margin training with relaxed inference directly minimizes the integrality distance on future examples. Importantly, if the constants B and R are known, then this bound can be efficiently evaluated from training data. It is worth noting that Corollary 2 actually holds for *any* integral assignment, not just the ground truth labels. Nonetheless, we feel the bound is more insightful when stated with respect to the ground truth labels, which are given in the learning setup. In this case $\mathbb{D}$ is defined as a joint distribution over $\mathcal{X}$ and $\mathcal{Y}$, and the bound holds with high probability over draws of both inputs and labels. It is also worth noting that Corollary 2 holds in the presence of global constraints—provided the ground truth (or whichever integral assignments are used) are in the feasible set.

6.4. Decoding a Solution

When the solution to an LP relaxation is fractional, we often round the solution to an integral assignment. Rounding schemes have been studied extensively (e.g., Raghavan and Tompson, 1987; Kleinberg and Tardos, 2002; Chekuri et al., 2004; Ravikumar et al., 2010). Arguably, the simplest method is to select the local assignments $\mu_i(y_i)$ with the highest values. One question that arises is how far the rounding, denoted $\mu_R(x; w)$, is from the exact solution; once this relationship is determined, one can apply our prior generalization analysis to the rounding. It turns out that the distance from μ_R to μ_I can be upper-bounded by a multiple of the integrality distance.

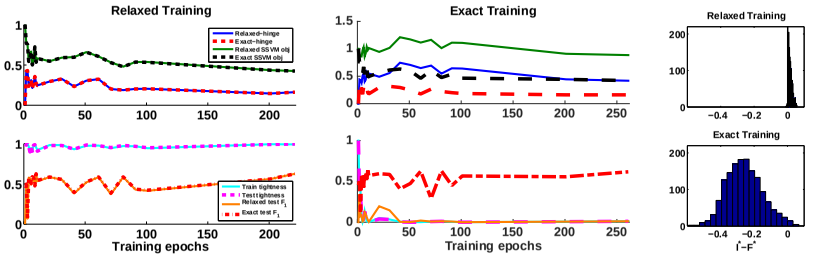


Figure 4: Training with the 'Yeast' multi-label classification dataset. Various quantities of interest are shown as a function of training iterations. (Left) Training with LP relaxation. (Middle) Training with ILP. (Right) Integrality margin (bin widths are scaled differently).

Lemma 2. *Suppose that every output variable has the same domain—i.e., $\mathcal{Y}_1 = \mathcal{Y}_2 = \ldots = \mathcal{Y}_n$—and that each domain has size k. If $\mu_R(x; w)$ is the rounding of the fractional solution, $\mu_L(x; w)$, then*

$$D_1(\mu_R, \mu_I) \;\leq\; k\, D_1(\mu_L, \mu_I). \tag{28}$$

Proof Consider any output variable. If the fractional solution assigns the majority of the local belief to the "correct" label (i.e., the label chosen by exact inference), then the rounding of that variable will be exact. However, if the fractional solution puts most of the local belief on an "incorrect" label, then the rounding of that variable will have D_1 distance 1 from the correct label. Since the incorrect label must have had a fractional value of at least $1/k$, it follows that the fractional solution has D_1 distance at least $1/k$, which is no less than $1/k$ that of the rounding. Applying this logic to every variable completes the proof. ■

Lemma 2 can be combined with Corollary 2 to generate bounds on the expected integrality distance of rounding; the bound simply scales by k.

7. Experiments

In this section we present some numerical results to support our theoretical analysis. We run experiments for a multi-label classification task and an image segmentation task. For training we have implemented the block-coordinate Frank-Wolfe algorithm for structured SVM (Lacoste-Julien et al., 2013), using GLPK as the LP solver. We use a standard L_2 regularizer, chosen via cross-validation.

7.1. Multi-Label Classification

For multi-label classification we adopt the experimental setting of Finley and Joachims (2008). In this setting labels are represented by binary variables, the model consists of

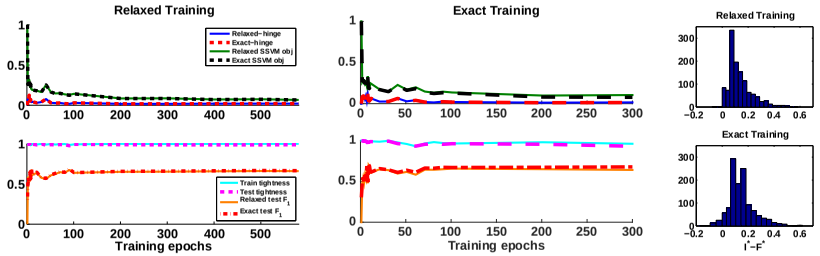


Figure 5: Training with the 'Scene' multi-label classification dataset. Various quantities of interest are shown as a function of training iterations. (Left) Training with LP relaxation. (Middle) Training with ILP. (Right) Integrality margin.

singleton and pairwise factors forming a fully connected graph over the labels, and the task loss is the normalized Hamming distance.

Fig. 4 shows relaxed and exact training iterations for the 'Yeast' dataset (14 labels). We plot the relaxed and exact hinge terms (Eq. (6)), the exact and relaxed SSVM training objectives[12] (Eq. (3) and Eq. (4), respectively), fraction of train and test instances having integral solutions, as well as test accuracy (measured by F_1 score). We use a simple scheme to round fractional solutions found with relaxed inference. First, we note that the relaxed-hinge values are nicely correlated with the relaxed training objective, and likewise the exact-hinge is correlated with the exact objective (left and middle, top). Second, observe that with relaxed training, the relaxed-hinge and the exact-hinge are very close (left, top), so the integrality gap, given by their difference, remains small (almost 0 here). On the other hand, with exact training there is a large integrality gap (middle, top). Indeed, we can see that the percentage of integral solutions is almost 100% for relaxed training (left, bottom), and close to 0% with exact training (middle, bottom). In Fig. 4 (right) we also show histograms of the difference between the optimal integral and fractional values, i.e., the integrality margin $(I^*(w,x) - F^*(w,x))$, under the final learned w for all training instances. It can be seen that with relaxed training this margin is positive (although small), while exact training results in larger negative values. Finally, we note that train and test integrality levels are very close to each other, almost indistinguishable (left and middle, bottom), which provides empirical support to our generalization result from Section 5.

We next train a model using random labels (with similar label counts as the true data). In this setting the learned model obtains 100% tight training instances (not shown), which supports our observation that any integral point can be used in place of the ground-truth, and that accuracy is not important for tightness. Finally, in order to verify that tightness is not coincidental, we test the tightness of the relaxation induced by a random weight vector w. We find that random models are never tight (in 20 trials), which shows that tightness of the relaxation does not come by chance.

12. The displayed objective values are averaged over train instances and exclude regularization.

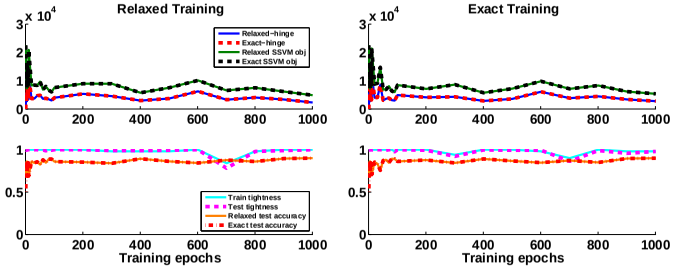


Figure 6: Training for foreground-background segmentation with the Weizmann Horse dataset. Various quantities of interest are shown as a function of training iterations. (Left) Training with LP relaxation. (Right) Training with ILP.

We now proceed to perform experiments on the 'Scene' dataset (6 labels). The results, in Fig. 5, differ from the 'Yeast' results in case of exact training (middle). Specifically, we observe that in this case the relaxed-hinge and exact-hinge are close in value (middle, top), as for relaxed training (left, top). As a consequence, the integrality gap is very small and the relaxation is tight for almost all train (and test) instances. These results illustrate that sometimes optimizing the exact objective can reduce the relaxed objective (and relaxed-hinge) as well. Further, in this setting we observe a larger integrality margin (right), namely the integral optimum is strictly better than the fractional one.

We conjecture that the LP instances are easy in this case due to the dominance of the singleton scores.[13] Specifically, the features provide a strong signal which allows label assignment to be decided mostly based on the local score, with little influence coming from the pairwise terms. To test this conjecture we inject Gaussian noise into the input features, forcing the model to rely more on the pairwise interactions. We find that with the noisy singleton scores the results are indeed more similar to the 'Yeast' dataset, where a large integrality gap is observed and fewer instances are tight (see Appendix C in the supplement).

7.2. Image Segmentation

Finally, we conduct experiments on a foreground-background segmentation problem using the Weizmann Horse dataset (Borenstein et al., 2004). The data consists of 328 images, of which we use the first 50 for training and the rest for testing. Here a binary output variable is assigned to each pixel, and there are $\sim 58K$ variables per image on average. We extract singleton and pairwise features as described in Domke (2013). Fig. 6 shows the same quantities as in the multi-label setting, except for the accuracy measure—here we compute the percentage of correctly classified pixels rather than F_1. We observe a very similar behavior to that of the 'Scene' multi-label dataset (Fig. 5). Specifically, both relaxed and exact training produce a small integrality gap and high percentage of tight instances.

13. With ILP training, the condition in Proposition 2 is satisfied for 65% of all variables, although only 1% of the training instances satisfy it for all their variables.

Unlike the 'Scene' dataset, here only 1.2% of variables satisfy the condition in Proposition 2 (using LP training). In all of our experiments the learned model scores were never balanced (Proposition 1), although for the segmentation problem we believe the models learned are close to balanced, both for relaxed and exact training.

8. Conclusion

In this paper we present a theoretical analysis of the tightness of LP relaxations often observed in structured prediction applications. Our analysis is based on a careful examination of the integrality gap, the integrality distance, and their relation to the training objective. It shows how training with LP relaxations, although designed with accuracy considerations in mind, also induces tightness of the relaxation. Our derivation also suggests that exact training may sometimes have the opposite effect, increasing the integrality gap.

To explain tightness at test time, we show that tightness generalizes in the following two senses: first, if most training predictions are integral, then most test instances will also be integral; secondly, if training predictions are on average "close to" integral, then test predictions will be similarly close to integral in expectation.

Acknowledgments

AW acknowledges support from the David MacKay Newton research fellowship at Darwin College, The Alan Turing Institute under EPSRC grant EP/N510129/1 & TU/B/000074, and the Leverhulme Trust via the CFI. DS acknowledges support from NSF CCF-1723344 and NSF CAREER award #1350965.

Appendix A. γ-Tight LP Relaxations

In this section we provide full derivations for the results in Section 5.1. We make extensive use of the results in Weller et al. (2016), some of which are restated here for completeness. We start by defining a model in minimal representation, which will be convenient for the derivations that follow. Specifically, in the case of binary variables ($y_i \in \{0,1\}$) with pairwise factors, we define a value η_i for each variable, and a value η_{ij} for each pair. The mapping between the over-complete vector μ and the minimal vector η is as follows. For singleton factors, we have:

$$\mu_i = \begin{pmatrix} 1-\eta_i \\ \eta_i \end{pmatrix}$$

Similarly, for the pairwise factors, we have:

$$\mu_{ij} = \begin{pmatrix} 1+\eta_{ij}-\eta_i-\eta_j & \eta_j-\eta_{ij} \\ \eta_i-\eta_{ij} & \eta_{ij} \end{pmatrix},$$

The corresponding mapping to minimal parameters is then:

$$\bar{\theta}_i = \theta_i(1) - \theta_i(0) + \sum_{j \in N_i} (\theta_{ij}(1,0) - \theta_{ij}(0,0))$$
$$\bar{\theta}_{ij} = \theta_{ij}(1,1) + \theta_{ij}(0,0) - \theta_{ij}(0,1) - \theta_{ij}(1,0)$$

In this representation, the LP relaxation is given by (up to constants):

$$\max_{\eta \in \mathbb{L}} f(\eta) := \sum_{i=1}^{n} \bar{\theta}_i \eta_i + \sum_{ij \in \mathcal{E}} \bar{\theta}_{ij} \eta_{ij}$$

where $\mathbb{L}$ is the appropriate transformation of $\mathcal{M}_L$ to the equivalent reduced space of η:

$$\begin{aligned} 0 &\le \eta_i \le 1 && \forall i \\ \max(0, \eta_i + \eta_j - 1) &\le \eta_{ij} \le \min(\eta_i, \eta_j) && \forall ij \in \mathcal{E} \end{aligned}$$

If $\bar{\theta}_{ij} > 0$ ($\bar{\theta}_{ij} < 0$), then the edge is called *attractive* (*repulsive*). If all edges are attractive, then the LP relaxation is known to be tight (Wainwright and Jordan, 2008). When not all edges are attractive, in some cases it is possible to make them attractive by *flipping* a subset of the variables ($y_i \leftarrow 1 - y_i$), which flips the signs of edge potentials for edges with exactly one end in the flipped subset.[14] In such cases the model is called *balanced*.

In the sequel we will make use of the known fact that all vertices of the local polytope are half-integral (take values in $\{0, \frac{1}{2}, 1\}$) (Padberg, 1989). We are now ready to prove the propositions (restated here for convenience).

14. The flip-set, if exists, is easy to find by making a single pass over the graph (see Weller (2015) for more details).

A.1. Proof of Proposition 1

Proposition 1 *A balanced model with a unique optimum is $(\alpha/2)$-tight, where α is the difference between the best and second-best (integral) solutions.*

Proof Weller et al. (2016) define for a given variable i the function $F^i_{\mathbb{L}}(z)$, which returns for every $0 \leq z \leq 1$ the constrained optimum:

$$F^i_{\mathbb{L}}(z) = \max_{\substack{\eta \in \mathbb{L} \\ \eta_i = z}} f(\eta)$$

Given this definition, they show that for a balanced model, $F^i_{\mathbb{L}}(z)$ is a *linear function* (Weller et al., 2016, Theorem 6).

Let m be the optimal score, let η^1 be the unique optimum integral vertex in minimal form so $f(\eta^1) = m$, and by assumption any other integral vertex has value at most $m - \alpha$. Denote the state of η^1 at coordinate i by $z^* = \eta^1_i$, and consider computing the constrained optimum holding η_i to various states. By assumption, any other integral vertex has value at most $m - \alpha$, therefore,

$$F^i_{\mathbb{L}}(z^*) = m$$
$$F^i_{\mathbb{L}}(1 - z^*) \leq m - \alpha$$

(the second line holds with equality if there exists a second-best solution η^2 s.t. $\eta^2_i \neq \eta^1_i$). Since $F^i_{\mathbb{L}}(z)$ is a linear function, we have that:

$$F^i_{\mathbb{L}}(1/2) \leq m - \alpha/2 \tag{29}$$

Next, towards contradiction, suppose that there exists a fractional vertex η^f with value $f(\eta^f) > m - \alpha/2$. Let j be a fractional coordinate, so $\eta^f_j = \frac{1}{2}$ (since vertices are half-integral). Our assumption implies that $F^j_{\mathbb{L}}(1/2) > m - \alpha/2$, but this contradicts Eq. (29). Therefore, we conclude that any fractional solution has value at most $f(\eta^f) \leq m - \alpha/2$. ■

It is possible to check in polynomial time if a model is balanced, if it has a unique optimum, and compute α. This can be done by computing the difference in value to the second-best. In order to find the second-best: one can constrain each variable in turn to differ from the state of the optimal solution, and recompute the MAP solution; finally, take the maximum over all these trials.

A.2. Proof of Proposition 2

Proposition 2 *If all variables satisfy the condition:*

$$\bar{\theta}_i \geq -\sum_{j \in N_i^-} \bar{\theta}_{ij} + \beta, \quad \textit{or} \quad \bar{\theta}_i \leq -\sum_{j \in N_i^+} \bar{\theta}_{ij} - \beta$$

for some $\beta > 0$, then the model is $(\beta/2)$-tight.

Proof For any binary pairwise models, given singleton terms $\{\eta_i\}$, the *optimal* edge terms are given by (for details see Weller et al., 2016):

$$\eta_{ij}(\eta_i, \eta_j) = \begin{cases} \min(\eta_i, \eta_j) & \text{if } \bar{\theta}_{ij} > 0 \\ \max(0, \eta_i + \eta_j - 1) & \text{if } \bar{\theta}_{ij} < 0 \end{cases}$$

Now, consider a variable i and let N_i be the set of its neighbors in the graph. Further, define the sets $N_i^+ = \{j \in N_i | \bar{\theta}_{ij} > 0\}$ and $N_i^- = \{j \in N_i | \bar{\theta}_{ij} < 0\}$, corresponding to attractive and repulsive edges, respectively. We next focus on the parts of the objective affected by the value at η_i (recomputing optimal edge terms); recall that all vertices are half-integral:

$\eta_i = 1$	$\eta_i = 1/2$	$\eta_i = 0$
$\bar{\theta}_i + \sum_{\substack{j \in N_i^+ \\ \eta_j = 1}} \bar{\theta}_{ij} + \frac{1}{2} \sum_{\substack{j \in N_i^+ \\ \eta_j = \frac{1}{2}}} \bar{\theta}_{ij} + \sum_{\substack{j \in N_i^- \\ \eta_j = 1}} \bar{\theta}_{ij} + \frac{1}{2} \sum_{\substack{j \in N_i^- \\ \eta_j = \frac{1}{2}}} \bar{\theta}_{ij}$	$\frac{1}{2}\bar{\theta}_i + \frac{1}{2} \sum_{\substack{j \in N_i^+ \\ \eta_j \in \{\frac{1}{2}, 1\}}} \bar{\theta}_{ij} + \frac{1}{2} \sum_{\substack{j \in N_i^- \\ \eta_j = 1}} \bar{\theta}_{ij}$	0

It is easy to verify that the condition $\bar{\theta}_i \geq -\sum_{j \in N_i^-} \bar{\theta}_{ij} + \beta$ guarantees that $\eta_i = 1$ in the optimal solution. We next bound the difference in objective values resulting from setting $\eta_i = 1/2$.

$$\Delta f = \frac{1}{2} \left(\bar{\theta}_i + \sum_{\substack{j \in N_i^+ \\ \eta_j = 1}} \bar{\theta}_{ij} + \sum_{\substack{j \in N_i^- \\ \eta_j \in \{\frac{1}{2}, 1\}}} \bar{\theta}_{ij} \right) \geq \frac{1}{2} \left(\bar{\theta}_i + \sum_{j \in N_i^-} \bar{\theta}_{ij} \right) \geq \beta/2$$

Similarly, when $\bar{\theta}_i \leq -\sum_{j \in N_i^+} \bar{\theta}_{ij} - \beta$, then $\eta_i = 0$ in any optimal solution. The difference in objective values from setting $\eta_i = 1/2$ in this case is:

$$\Delta f = -\frac{1}{2} \left(\bar{\theta}_i + \sum_{\substack{j \in N_i^+ \\ \eta_j \in \{\frac{1}{2}, 1\}}} \bar{\theta}_{ij} + \sum_{\substack{j \in N_i^- \\ \eta_j = 1}} \bar{\theta}_{ij} \right) \geq -\frac{1}{2} \left(\bar{\theta}_i + \sum_{j \in N_i^+} \bar{\theta}_{ij} \right) \geq \beta/2$$

Notice that for more fractional coordinates the difference in values can only increase, so in any case the fractional solution is worse by at least $\beta/2$. ■

Appendix B. Generalization Bound Proofs

To prove Theorems 1 and 2, we will use the following PAC-Bayes bound. There are other ways of proving generalization bounds for structured predictors (e.g., Weiss and Taskar, 2010), and other PAC-Bayesian analyses of structured prediction (e.g., McAllester, 2007), but we prefer the following for its simplicity. Our analysis is based on London et al.'s (2016), with certain simplifications due to differences in our objective.

Lemma 3. *Let $\mathbb{D}$ denote a distribution over an instance space, $\mathcal{Z}$. Let $\mathcal{H}$ denote a hypothesis class. Let $\mathcal{L} : \mathcal{H} \times \mathcal{Z} \to [0, 1]$ denote a bounded loss function. Let $\mathbb{P}$ denote a fixed prior distribution over $\mathcal{H}$. Then, for any $\delta \in (0, 1)$ and $m \geq 1$, with probability at least $1 - \delta$ over draws of $(Z^{(1)}, \ldots, Z^{(m)}) \in \mathcal{Z}^m$, according to $\mathbb{D}^m$, every posterior distribution, $\mathbb{Q}$, over $\mathcal{H}$, satisfies*

$$\mathop{\mathbb{E}}_{Z \sim \mathbb{D}} \mathop{\mathbb{E}}_{h \sim \mathbb{Q}} [\mathcal{L}(h, Z)] \leq \frac{1}{m} \sum_{i=1}^{m} \mathop{\mathbb{E}}_{h \sim \mathbb{Q}} [\mathcal{L}(h, Z^{(i)})] + 2\sqrt{\frac{D_{\mathrm{KL}}(\mathbb{Q} \| \mathbb{P}) + \ln \frac{2}{\delta}}{2m}} \tag{30}$$

Proof To simplify notation, let

$$\varphi(h, Z^{(i)}) \triangleq \frac{1}{m}\left(\mathop{\mathbb{E}}_{Z\sim\mathbb{D}}[\mathcal{L}(h,Z)] - \mathcal{L}(h, Z^{(i)})\right).$$

For any free parameter, $\epsilon \in \mathbb{R}$, observe that

$$\mathop{\mathbb{E}}_{Z\sim\mathbb{D}}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}[\mathcal{L}(h,Z)] - \frac{1}{m}\sum_{i=1}^{m}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}[\mathcal{L}(h,Z^{(i)})] = \frac{1}{\epsilon}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}\left[\sum_{i=1}^{m}\epsilon\,\varphi(h,Z^{(i)})\right]. \tag{31}$$

The next step uses Donsker and Varadhan's (1975) *change of measure* inequality, which states that, if X is a random variable taking values in Ω, then for any two distributions, $\mathbb{P}$ and $\mathbb{Q}$, on Ω,

$$\mathop{\mathbb{E}}_{X\sim\mathbb{Q}}[X] \le D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln \mathop{\mathbb{E}}_{X\sim\mathbb{P}}\left[e^{X}\right].$$

Applying change of measure to the righthand side of Equation 31, we have

$$\frac{1}{\epsilon}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}\left[\sum_{i=1}^{m}\epsilon\,\varphi(h,Z^{(i)})\right] \le \frac{1}{\epsilon}\left(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln \mathop{\mathbb{E}}_{h\sim\mathbb{P}}\left[\exp\left(\sum_{i=1}^{m}\epsilon\,\varphi(h,Z^{(i)})\right)\right]\right). \tag{32}$$

By Markov's inequality, with probability $1-\delta$ over draws of a training set, $\mathbf{Z} \triangleq (Z^{(1)},\ldots,Z^{(m)})$, according to $\mathbb{D}^m$,

$$\begin{aligned}
\mathop{\mathbb{E}}_{h\sim\mathbb{P}}\left[\exp\left(\sum_{i=1}^{m}\epsilon\,\varphi(h,Z^{(i)})\right)\right] &\le \frac{1}{\delta}\mathop{\mathbb{E}}_{\mathbf{Z}\sim\mathbb{D}^m}\mathop{\mathbb{E}}_{h\sim\mathbb{P}}\left[\exp\left(\sum_{i=1}^{m}\epsilon\,\varphi(h,Z^{(i)})\right)\right] \\
&= \frac{1}{\delta}\mathop{\mathbb{E}}_{h\sim\mathbb{P}}\mathop{\mathbb{E}}_{\mathbf{Z}\sim\mathbb{D}^m}\left[\prod_{i=1}^{m}\exp\left(\epsilon\,\varphi(h,Z^{(i)})\right)\right] \\
&= \frac{1}{\delta}\mathop{\mathbb{E}}_{h\sim\mathbb{P}}\prod_{i=1}^{m}\mathop{\mathbb{E}}_{Z^{(i)}\sim\mathbb{D}}\left[\exp\left(\epsilon\,\varphi(h,Z^{(i)})\right)\right].
\end{aligned} \tag{33}$$

In the last line, we leveraged the fact that the expectation of a product of i.i.d. random variables (in this case, $\varphi(h,Z^{(i)})$) is the product of their expectations. To upper-bound each expectation, we use Hoeffding's inequality, which states that if X is a zero-mean random variable, such that $a \le X \le b$ almost surely, then, for all $\epsilon \in \mathbb{R}$,

$$\mathbb{E}\left[e^{\epsilon X}\right] \le \exp\left(\frac{\epsilon^2(b-a)^2}{8}\right).$$

Since $\varphi(h,Z^{(i)})$ has mean zero, and

$$\mathop{\mathbb{E}}_{Z\sim\mathbb{D}}[\mathcal{L}(h,Z)] - \frac{1}{m} \le \varphi(h,Z^{(i)}) \le \mathop{\mathbb{E}}_{Z\sim\mathbb{D}}[\mathcal{L}(h,Z)] - 0,$$

we have that

$$\mathop{\mathbb{E}}_{Z^{(i)}\sim\mathbb{D}}\left[\exp\left(\epsilon\,\varphi(h,Z^{(i)})\right)\right] \le \exp\left(\frac{\epsilon^2}{8m^2}\right). \tag{34}$$

Combining Equations 31 to 34, we have for any $\epsilon \in \mathbb{R}$, with probability at least $1-\delta$, all $\mathbb{Q}$ satisfy

$$\mathop{\mathbb{E}}_{Z\sim\mathbb{D}}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}[\mathcal{L}(h,Z)] - \frac{1}{m}\sum_{i=1}^{m}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}[\mathcal{L}(h,Z^{(i)})] \le \frac{1}{\epsilon}\left(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln\frac{1}{\delta}\right) + \frac{\epsilon}{8m}. \tag{35}$$

The ϵ that minimizes Equation 35 is straightforward: setting the derivative equal to 0 and solving for $\epsilon^\star$, we have $\epsilon^\star = \sqrt{8m(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln\frac{1}{\delta})}$. Unfortunately, this optimal value depends on the choice of posterior, $\mathbb{Q}$, which means that the bound would only hold for a single posterior, not *all* posteriors simultaneously. To derive a bound that holds for all posteriors, we employ a covering technique, attributed to Seldin et al. (2012). We first define a countably infinite sequence of ϵ values. For each value, we assign an exponentially decreasing probability that Equation 35 fails to hold, such that with probability at least $1-\delta$ it holds for all values simultaneously. Then, for any posterior, we select an index, $j^\star$, that approximately optimizes Equation 35, and use lower and upper bounds on $\epsilon_{j^\star}$ to simplify the bound.

For $j = 0, 1, 2, \ldots$, let

$$\epsilon_j \triangleq 2^j\sqrt{8m\ln\frac{2}{\delta}}. \tag{36}$$

For each ϵ_j, we assign $\delta_j \triangleq \delta 2^{-(j+1)}$ probability that Equation 35 does not hold, substituting (ϵ_j, δ_j) for (ϵ, δ). Thus, with probability at least $1 - \sum_{j=0}^{\infty}\delta_j = 1 - \delta\sum_{j=0}^{\infty}2^{-(j+1)} = 1-\delta$, all $j = 0, 1, 2, \ldots$ satisfy

$$\mathop{\mathbb{E}}_{Z\sim\mathbb{D}}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}[\mathcal{L}(h,Z)] - \frac{1}{m}\sum_{i=1}^{m}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}[\mathcal{L}(h,Z^{(i)})] \le \frac{1}{\epsilon_j}\left(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln\frac{1}{\delta_j}\right) + \frac{\epsilon_j}{8m}$$

For any given posterior, $\mathbb{Q}$, we choose an index, $j^\star$, by taking

$$j^\star \triangleq \left\lfloor \frac{1}{2\ln 2}\ln\left(\frac{D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P})}{\ln(2/\delta)} + 1\right)\right\rfloor,$$

which implies

$$\sqrt{2m\left(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln\frac{2}{\delta}\right)} \le \epsilon_{j^\star} \le \sqrt{8m\left(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln\frac{2}{\delta}\right)}.$$

We further have (from London et al. (2016)) that

$$D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln\frac{1}{\delta_{j^\star}} \le \frac{3}{2}\left(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln\frac{2}{\delta}\right).$$

Thus, with probability at least $1-\delta$, all posteriors satisfy

$$\begin{aligned}
&\mathop{\mathbb{E}}_{Z\sim\mathbb{D}}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}[\mathcal{L}(h,Z)] - \frac{1}{m}\sum_{i=1}^{m}\mathop{\mathbb{E}}_{h\sim\mathbb{Q}}[\mathcal{L}(h,Z^{(i)})] \\
&\quad\le \frac{1}{\epsilon_{j^\star}}\left(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln\frac{1}{\delta_{j^\star}}\right) + \frac{\epsilon_{j^\star}}{8m} \\
&\quad\le \frac{D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln(1/\delta_{j^\star})}{\sqrt{2m(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln(2/\delta))}} + \frac{\sqrt{8m(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln(2/\delta))}}{8m} \\
&\quad\le \frac{3\left(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln(2/\delta)\right)}{2\sqrt{2m(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln(2/\delta))}} + \frac{\sqrt{8m(D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln(2/\delta))}}{8m} \\
&\quad= 2\sqrt{\frac{D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln(2/\delta)}{2m}},
\end{aligned}$$

which completes the proof. ■

B.1. Proof of Theorem 1

The proof proceeds in three stages: First, we construct prior and posterior distributions. Both are uniform, but the posterior is centered at a learned hypothesis, and its support shrinks as the training data grows. This construction essentially means that we should trust the learned model more as we get more training data. In the second step, we upper-bound the KL divergence term—which is trivial given the uniform constructions. Finally, we show that, under the posterior construction, the loss of a random hypothesis is "close to" the loss of the learned hypothesis; specifically, that the difference between their losses is a small additive term that vanishes as the training set grows.

Let $\mathbb{P}$ denote a uniform prior over $\{w \in \mathbb{R}^d : \|w\|_2 \le R\}$. Given a learned weight vector, w, we construct a posterior, $\mathbb{Q}$, as a uniform distribution over $\{w' \in \mathbb{R}^d : \|w'\|_2 \le R,\ \|w'-w\|_2 \le 2\gamma/(mB)\}$. It can easily be shown (London et al., 2016, Appendix C.3) that

$$D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) \le d\ln(mBR/\gamma). \tag{37}$$

The next part of the proof "derandomizes" the loss functions in Equation 30, replacing the randomized hypothesis, with weights $w' \sim \mathbb{Q}$, with the deterministic predictor based on w. To do so, we will bound the difference $|\mathcal{L}_\gamma(w,x) - \mathcal{L}_\gamma(w',x)|$. We first note that the ramp function is $(1/\gamma)$-Lipschitz; that is, for any γ, x, w and w',

$$\left|\mathcal{L}_\gamma(w,x) - \mathcal{L}_\gamma(w',x)\right| \le \frac{1}{\gamma}\left|D(w,x) - D(w',x)\right|. \tag{38}$$

To upper-bound $|D(w,x) - D(w',x)|$, let

$$\begin{aligned}
\mu_F &\in \mathop{\arg\max}_{\mu\in\mathcal{V}_F} \theta(x;w)^\top\mu \quad\text{and}\quad \mu'_F \in \mathop{\arg\max}_{\mu\in\mathcal{V}_F} \theta(x;w')^\top\mu, \\
\mu_I &\in \mathop{\arg\max}_{\mu\in\mathcal{V}_I} \theta(x;w)^\top\mu \quad\text{and}\quad \mu'_I \in \mathop{\arg\max}_{\mu\in\mathcal{V}_I} \theta(x;w')^\top\mu,
\end{aligned}$$

and observe that

$$\begin{aligned}
\left|D(w,x) - D(w',x)\right| &= \left|\theta(x;w)^\top(\mu_F - \mu_I) - \theta(x;w')^\top(\mu'_F - \mu'_I)\right| \\
&\le \left|\theta(x;w)^\top\mu_F - \theta(x;w')^\top\mu'_F\right| \qquad (39) \\
&\quad + \left|\theta(x;w')^\top\mu'_I - \theta(x;w)^\top\mu_I\right|. \qquad (40)
\end{aligned}$$

We will upper-bound Equations 39 and 40 separately.

Starting with Equation 39, assume that $\theta(x;w)^\top\mu_F \ge \theta(x;w')^\top\mu'_F$. (If the inequality goes in the other direction, we simply swap the left and right terms, which is equivalent inside the absolute value.) We then have that

$$\begin{aligned}
\left|\theta(x;w)^\top\mu_F - \theta(x;w')^\top\mu'_F\right| &= \theta(x;w)^\top\mu_F - \theta(x;w')^\top\mu'_F \\
&\le \theta(x;w)^\top\mu_F - \theta(x;w')^\top\mu_F \\
&= (w-w')^\top\phi(x,\mu_F),
\end{aligned}$$

due to the optimality of μ'_F for $\theta(x;w')$. Then, using Cauchy-Schwarz,

$$(w-w')^\top\phi(x,\mu_F) \le \left\|w-w'\right\|_2 \|\phi(x,\mu_F)\|_2 \le \left\|w-w'\right\|_2 B.$$

An identical analysis of Equation 40 yields

$$\begin{aligned}
\left|\theta(x;w')^\top\mu'_I - \theta(x;w)^\top\mu_I\right| &= \theta(x;w')^\top\mu'_I - \theta(x;w)^\top\mu_I \\
&\le \theta(x;w')^\top\mu'_I - \theta(x;w)^\top\mu'_I \\
&= (w'-w)^\top\phi(x,\mu'_I) \\
&\le \left\|w'-w\right\|_2 \left\|\phi(x,\mu'_I)\right\|_2 \\
&\le \left\|w'-w\right\|_2 B.
\end{aligned}$$

By construction, every $w' \sim \mathbb{Q}$ has distance at most $2\gamma/(mB)$ from w. Therefore, substituting the above inequalities into Equations 39 and 40, we have that

$$\left|D(w,x) - D(w',x)\right| \le \left\|w-w'\right\|_2 B + \left\|w'-w\right\|_2 B \le \frac{2\gamma}{mB}\cdot 2B = \frac{4\gamma}{m};$$

and combining this bound with Equation 38 yields

$$\left|\mathcal{L}_\gamma(w,x) - \mathcal{L}_\gamma(w',x)\right| \le \frac{1}{\gamma}\cdot\frac{4\gamma}{m} = \frac{4}{m}.$$

Thus, the loss of any random weight vector, w', is at most $4/m$ above or below that of the learned weights, w. We can therefore derandomize the randomized loss by bounding its distance to the deterministic loss:

$$\left|\mathcal{L}_\gamma(w,x) - \mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_\gamma(w',x)]\right| \le \mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}\left[\left|\mathcal{L}_\gamma(w,x) - \mathcal{L}_\gamma(w',x)\right|\right] \le \frac{4}{m}.$$

Recalling $\mathcal{L}_0(w,x) \le \mathcal{L}_\gamma(w,x)$, we then have that

$$\mathop{\mathbb{E}}_{x\sim\mathbb{D}}[\mathcal{L}_0(w,x)] \le \mathop{\mathbb{E}}_{x\sim\mathbb{D}}[\mathcal{L}_\gamma(w,x)] \le \mathop{\mathbb{E}}_{x\sim\mathbb{D}}\mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_\gamma(w',x)] + \frac{4}{m} \tag{41}$$

$$\text{and} \quad \frac{1}{m}\sum_{i=1}^{m}\mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_\gamma(w',x^{(i)})] \le \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_\gamma(w,x^{(i)}) + \frac{4}{m}. \tag{42}$$

All that remains is to put the pieces together. Via Lemma 3, with probability at least $1-\delta$ over draws of the training data,

$$\mathop{\mathbb{E}}_{x\sim\mathbb{D}}\mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_\gamma(w',x)] \le \frac{1}{m}\sum_{i=1}^{m}\mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_\gamma(w',x^{(i)})] + 2\sqrt{\frac{D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) + \ln\frac{2}{\delta}}{2m}}.$$

To complete the proof, we use Equation 37 to upper-bound the KL divergence, Equation 41 to lower-bound the randomized risk and Equation 42 to upper-bound the empirical randomized risk.

B.2. Proof of Theorem 2

As we did in the previous proof, we define $\mathbb{P}$ as a uniform prior over $\{w \in \mathbb{R}^d : \|w\|_2 \le R\}$. Given w, we construct a posterior, $\mathbb{Q}$, as a uniform distribution over $\{w' \in \mathbb{R}^d : \|w'\|_2 \le R,\ \|w'-w\|_2 \le 2/(mB)\}$ (we omit γ), which yields $D_{\mathrm{KL}}(\mathbb{Q}\|\mathbb{P}) \le d\ln(mBR)$.

In what follows we use the fact that when one of the arguments of $D_1(\mu,\mu')$ (Eq. (18))—say, μ—is integral, $D_1(\mu,\mu')$ can be expressed as $\ell(\mu)^\top\mu'$, where

$$\ell(\mu) \triangleq \frac{1}{n}\begin{bmatrix}\mathbf{1}-\mu_{\mathrm{u}}\\ \mathbf{0}\end{bmatrix}. \tag{43}$$

To derandomize the loss of $w' \sim \mathbb{Q}$, we need to bound the difference $|\mathcal{L}_{\mathrm{r}}(w,x,\mu_I) - \mathcal{L}_{\mathrm{r}}(w',x,\mu_I)|$. Note that w' is being evaluated with respect to the MAP assignment under w, i.e., $\mu_I \in \arg\max_{\mu\in\mathcal{M}}\theta(x;w)^\top\mu$. To simplify notation, we will use the following shorthand:

$$\theta \triangleq \theta(x;w) \quad \text{and} \quad \theta' \triangleq \theta(x;w')\,;$$

$$\mu_L \in \mathop{\arg\max}_{\mu\in\mathcal{M}_L}\ \theta^\top\mu \quad \text{and} \quad \mu'_L \in \mathop{\arg\max}_{\mu\in\mathcal{M}_L}\ (\theta')^\top\mu\,;$$

$$\tilde{\mu}_L \in \mathop{\arg\max}_{\mu\in\mathcal{M}_L}\ (\ell(\mu_I)+\theta)^\top\mu \quad \text{and} \quad \tilde{\mu}'_L \in \mathop{\arg\max}_{\mu\in\mathcal{M}_L}\ (\ell(\mu_I)+\theta')^\top\mu.$$

We then have that the difference of ramp losses decomposes as

$$\begin{aligned}
&\left|\mathcal{L}_{\mathrm{r}}(w,x,\mu_I) - \mathcal{L}_{\mathrm{r}}(w',x,\mu_I)\right| \\
&\quad= \left|\Big((\ell(\mu_I)+\theta)^\top\tilde{\mu}_L - \theta^\top\mu_L\Big) - \Big((\ell(\mu_I)+\theta')^\top\tilde{\mu}'_L - (\theta')^\top\mu'_L\Big)\right| \\
&\quad\le \left|(\ell(\mu_I)+\theta)^\top\tilde{\mu}_L - (\ell(\mu_I)+\theta')^\top\tilde{\mu}'_L\right| &(44)\\
&\qquad+ \left|(\theta')^\top\mu'_L - \theta^\top\mu_L\right|. &(45)
\end{aligned}$$

Just as before, we will upper-bound Equations 44 and 45 separately.

We start with Equation 45, which is slightly simpler. Without loss of generality, assume that $(\theta')^\top\mu'_L \ge \theta^\top\mu_L$. Using the optimality of μ_L with respect to θ, we then have that

$$\begin{aligned}
\left|(\theta')^\top\mu'_L - \theta^\top\mu_L\right| &= (\theta')^\top\mu'_L - \theta^\top\mu_L \\
&\le (\theta')^\top\mu'_L - \theta^\top\mu'_L \\
&= (w'-w)^\top\phi(x,\mu'_L) \\
&\le \left\|w'-w\right\|_2\left\|\phi(x,\mu'_L)\right\|_2 \\
&\le \left\|w'-w\right\|_2 B.
\end{aligned}$$

By construction, every $w' \sim \mathbb{Q}$ has distance at most $2/(mB)$ from w, so

$$\left|(\theta')^\top\mu'_L - \theta^\top\mu_L\right| \le \left\|w'-w\right\|_2 B \le \frac{2}{mB}\cdot B = \frac{2}{m}. \tag{46}$$

Applying the same technique to Equation 44, we assume, without loss of generality, that $(\ell(\mu_I)+\theta)^\top\tilde{\mu}_L \ge (\ell(\mu_I)+\theta')^\top\tilde{\mu}'_L$. Then,

$$\begin{aligned}
\left|(\ell(\mu_I)+\theta)^\top\tilde{\mu}_L - (\ell(\mu_I)+\theta')^\top\tilde{\mu}'_L\right| &= (\ell(\mu_I)+\theta)^\top\tilde{\mu}_L - (\ell(\mu_I)+\theta')^\top\tilde{\mu}'_L \\
&\le (\ell(\mu_I)+\theta)^\top\tilde{\mu}_L - (\ell(\mu_I)+\theta')^\top\tilde{\mu}_L \\
&= (\theta-\theta')^\top\tilde{\mu}_L \\
&= (w-w')^\top\phi(x,\tilde{\mu}_L) \\
&\le \left\|w-w'\right\|_2 B \le \frac{2}{m}.
\end{aligned} \tag{47}$$

Substituting Equation 47 for 44, and Equation 46 for 45, we have that

$$\left|\mathcal{L}_{\mathrm{r}}(w,x,\mu_I) - \mathcal{L}_{\mathrm{r}}(w',x,\mu_I)\right| \le \frac{4}{m}\ ;$$

hence,

$$\left|\mathcal{L}_{\mathrm{r}}(w,x,\mu_I) - \mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_{\mathrm{r}}(w',x,\mu_I)]\right| \le \mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}\left[\left|\mathcal{L}_{\mathrm{r}}(w,x,\mu_I) - \mathcal{L}_{\mathrm{r}}(w',x,\mu_I)\right|\right] \le \frac{4}{m}.$$

Then, using the inequalities in Equation 22, we have that

$$\mathop{\mathbb{E}}_{x\sim\mathbb{D}}[\mathcal{L}_1(w,x,\mu_I)] \le \mathop{\mathbb{E}}_{x\sim\mathbb{D}}[\mathcal{L}_{\mathrm{r}}(w,x,\mu_I)] \le \mathop{\mathbb{E}}_{x\sim\mathbb{D}}\mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_{\mathrm{r}}(w',x,\mu_I)] + \frac{4}{m}, \tag{48}$$

and

$$\frac{1}{m}\sum_{i=1}^{m}\mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_{\mathrm{r}}(w',x^{(i)},\mu_I^{(i)})] \le \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_{\mathrm{r}}(w,x^{(i)},\mu_I^{(i)}) + \frac{4}{m}. \tag{49}$$

Finally, combining Lemma 3 with Equations 48 and 49 to lower- and upper-bound the randomized losses, we have that, with probability at least $1-\delta$ over draws of the training

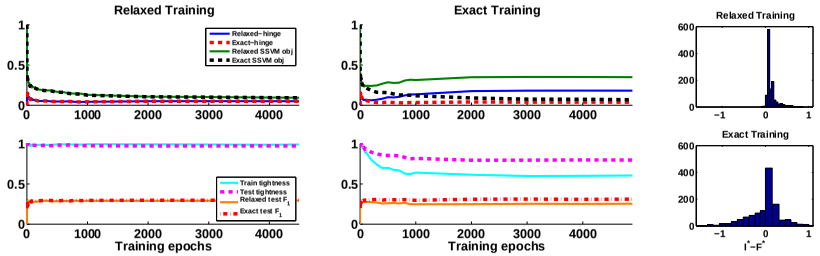


Figure 7: Training with a noisy version of the 'Scene' dataset. Various quantities of interest are shown as a function of training iterations. (Left) Training with LP relaxation. (Middle) Training with ILP. (Right) Integrality margin (bin widths are scaled differently).

set,

$$\begin{aligned}\mathop{\mathbb{E}}_{x\sim\mathbb{D}}[\mathcal{L}_1(w,x,\mu_I)] &\le \mathop{\mathbb{E}}_{x\sim\mathbb{D}}\mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_{\mathrm{r}}(w',x,\mu_I)] + \frac{4}{m} \\ &\le \frac{1}{m}\sum_{i=1}^{m}\mathop{\mathbb{E}}_{w'\sim\mathbb{Q}}[\mathcal{L}_{\mathrm{r}}(w',x^{(i)},\mu_I^{(i)})] + \frac{4}{m} + 2\sqrt{\frac{d\ln(mBR)+\ln\frac{2}{\delta}}{2m}} \\ &\le \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_{\mathrm{r}}(w,x^{(i)},\mu_I^{(i)}) + \frac{8}{m} + 2\sqrt{\frac{d\ln(mBR)+\ln\frac{2}{\delta}}{2m}}.\end{aligned}$$

Noting that the hinge loss uniformly upper-bounds the ramp loss completes the proof.

Appendix C. Additional Experimental Results

In this section we present additional experimental results for the 'Scene' dataset. Specifically, we inject random Gaussian noise to the input features in order to reduce the signal in the singleton scores and increase the role of the pairwise interactions. This makes the problem harder since the prediction needs to account for global information.

In Fig. 7 we observe that with exact training the exact loss is minimized, causing the exact-hinge to decrease, since it is upper bounded by the loss (middle, top). On the other hand, the relaxed-hinge (and relaxed loss) *increase* during training, which results in a large integrality gap and fewer tight instances. In contrast, with relaxed training the relaxed loss is minimized, which causes the relaxed-hinge to decrease. Since the exact-hinge is upper bounded by the relaxed-hinge it also decreases, but both hinge terms decrease similarly and remain very close to each other. This results in a small integrality gap and tightness of almost all instances.

Finally, in contrast to other settings, in Fig. 7 we observe that with exact training the test tightness is noticeably higher (about 20%) than the train tightness (Fig. 7, middle, bottom).

This does not contradict our bound from Theorem 1, since in fact the test fractionality is even *lower* than the bound suggests. On the other hand, this result does indicate that train and test tightness may sometimes behave differently, which means that we might need to increase the size of the trainset in order to get a tighter bound.

References

G. H. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data*. The MIT Press, 2007.

F. Barahona. On cuts and matchings in planar graphs. *Mathematical Programming*, 60: 53–68, 1993.

E. Borenstein, E. Sharon, and S. Ullman. Combining top-down and bottom-up segmentation. In *CVPR*, 2004.

C. Chekuri, S. Khanna, J. Naor, and L. Zosin. A linear programming formulation and approximation algorithms for the metric labeling problem. *SIAM J. on Discrete Mathematics*, 18(3):608–625, 2004.

L.-C. Chen, A. Schwing, A. Yuille, and R. Urtasun. Learning deep structured models. In *International Conference on Machine Learning*, 2015.

M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *EMNLP*, 2002.

Hal Daumé III, John Langford, and Daniel Marcu. Search-based structured prediction. *Machine Learning*, 75(3):297–325, 2009.

J. Domke. Learning graphical model parameters with approximate marginal inference. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(10), 2013.

M. Donsker and S. Varadhan. Asymptotic evaluation of certain Markov process expectations for large time. *Communications on Pure and Applied Mathematics*, 28(1):1–47, 1975.

T. Finley and T. Joachims. Training structural SVMs when exact inference is intractable. In *Proceedings of the 25th International Conference on Machine learning*, pages 304–311, 2008.

A. Globerson, T. Roughgarden, D. Sontag, and C. Yildirim. How hard is inference for structured prediction? In *ICML*, 2015.

J. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. *Journal of the ACM*, 49(5):616–639, 2002.

T. Koo, A. M. Rush, M. Collins, T. Jaakkola, and D. Sontag. Dual decomposition for parsing with non-projective head automata. In *EMNLP*, 2010.

A. Koster, S.P.M. van Hoesel, and A.W.J. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 23:89–97, 1998.

A. Kulesza and F. Pereira. Structured learning with approximate inference. In *Advances in Neural Information Processing Systems 20*, pages 785–792, 2007.

S. Lacoste-Julien, B. Taskar, D. Klein, and M. I. Jordan. Word alignment via quadratic assignment. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 112–119, 2006.

S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *ICML*, pages 53–61, 2013.

Hunter Lang, David Sontag, and Aravindan Vijayaraghavan. Optimality of approximate inference algorithms on stable instances. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics (AISTATS)*. JMLR: W&CP, 2018.

D. Khuê Lê-Huu and Nikos Paragios. Continuous Relaxation of MAP Inference: A Nonconvex Perspective. In *CVPR 2018 - IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–19, 2018.

B. London, B. Huang, and L. Getoor. Stability and generalization in structured prediction. *Journal of Machine Learning Research*, 17, 2016.

Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Bilu–Linial stable instances of max cut and minimum multiway cut. *Proc. 22nd Symposium on Discrete Algorithms (SODA)*, 2014.

A. Martins, N. Smith, and E. P. Xing. Concise integer linear programming formulations for dependency parsing. In *ACL*, 2009a.

A. Martins, N. Smith, and E. P. Xing. Polyhedral outer approximations with application to natural language parsing. In *Proceedings of the 26th International Conference on Machine Learning*, 2009b.

D. McAllester. Generalization bounds and consistency for structured labeling. In G. Bakir, T. Hofmann, B. Schölkopf, A. Smola, B. Taskar, and S. Vishwanathan, editors, *Predicting Structured Data*. MIT Press, 2007.

O. Meshi, E. Eban, G. Elidan, and A. Globerson. Learning max-margin tree predictors. In *UAI*, 2013.

Sebastian Nowozin, Peter V. Gehler, Jeremy Jancsary, and Christoph Lampert. *Advanced Structured Prediction*. MIT Press, 2014.

Manfred Padberg. The boolean quadric polytope: Some characteristics, facets and relatives. *Mathematical Programming*, 45(1):139–172, 1989.

P. Raghavan and C. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

P. Ravikumar, A. Agarwal, and M. J. Wainwright. Message-passing for graph-structured linear programs: Proximal methods and rounding schemes. *JMLR*, 11:1043–1080, 2010.

D. Roth. On the hardness of approximate reasoning. *Artificial Intelligence*, 82, 1996.

D. Roth and W. Yih. A linear programming formulation for global inference in natural language tasks. In *CoNLL, The 8th Conference on Natural Language Learning*, 2004.

D. Roth and W. Yih. Integer linear programming inference for conditional random fields. In *ICML*, pages 736–743. ACM, 2005.

M. Rowland, A. Pacchiano, and A. Weller. Conditions beyond treewidth for tightness of higher-order LP relaxations. In *In Artificial Intelligence and Statistics (AISTATS)*, 2017.

Alexander M. Rush, David Sontag, Michael Collins, and Tommi Jaakkola. On dual decomposition and linear programming relaxations for natural language processing. In *EMNLP*, 2010.

M. I. Schlesinger. Syntactic analysis of two-dimensional visual signals in noisy conditions. *Kibernetika*, 4:113—130, 1976.

Y. Seldin, F. Laviolette, N. Cesa-Bianchi, J. Shawe-Taylor, and P. Auer. PAC-Bayesian inequalities for martingales. *IEEE Transactions on Information Theory*, 58(12):7086–7093, 2012.

H. D. Sherali and W. P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM J. on Disc. Math.*, 3(3):411–430, 1990.

Y. Shimony. Finding the MAPs for belief networks is NP-hard. *Aritifical Intelligence*, 68(2): 399–410, 1994.

D. Sontag and T. Jaakkola. New outer bounds on the marginal polytope. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1393–1400. MIT Press, Cambridge, MA, 2008.

D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. In *UAI*, pages 503–510, 2008.

D. Sontag, O. Meshi, T. Jaakkola, and A. Globerson. More data means less inference: A pseudo-max approach to structured learning. In *NIPS*, 2010.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems*. MIT Press, 2003.

B. Taskar, V. Chatalbashev, and D. Koller. Learning associative Markov networks. In *Proc. ICML*. ACM Press, 2004.

J. Thapper and S. Živný. The power of linear programming for valued CSPs. In *FOCS*, 2012.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML*, pages 104–112, 2004.

M. Wainwright and M. I. Jordan. *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers Inc., Hanover, MA, USA, 2008.

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

D. Weiss and B. Taskar. Structured Prediction Cascades. In *AISTATS*, 2010.

A. Weller. Characterizing tightness of LP relaxations by forbidding signed minors. In *Uncertainty in Artificial Intelligence (UAI)*, 2016.

Adrian Weller. Bethe and related pairwise entropy approximations. In *Uncertainty in Artificial Intelligence (UAI)*, 2015.

Adrian Weller, Mark Rowland, and David Sontag. Tightness of LP relaxations for almost balanced models. In *AISTATS*, 2016.

Yuan Zhang, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Greed is good if randomized: New inference for dependency parsing. In *EMNLP*, 2014.

Journal of Machine Learning Research 20 (2019) 1-37 Submitted 9/17; Revised 11/18; Published 2/19

Approximations of the Restless Bandit Problem

Steffen Grünewälder S.GRUNEWALDER@LANCASTER.AC.UK

Azadeh Khaleghi A.KHALEGHI@LANCASTER.AC.UK

Department of Mathematics and Statistics
Lancaster University
Lancaster, UK

Editor: Peter Auer

Abstract

The multi-armed restless bandit problem is studied in the case where the pay-off distributions are stationary φ-mixing. This version of the problem provides a more realistic model for most real-world applications, but cannot be optimally solved in practice, since it is known to be PSPACE-hard. The objective of this paper is to characterize a sub-class of the problem where *good* approximate solutions can be found using tractable approaches. Specifically, it is shown that under some conditions on the φ-mixing coefficients, a modified version of UCB can prove effective. The main challenge is that, unlike in the i.i.d. setting, the distributions of the sampled pay-offs may not have the same characteristics as those of the original bandit arms. In particular, the φ-mixing property does not necessarily carry over. This is overcome by carefully controlling the effect of a sampling policy on the pay-off distributions. Some of the proof techniques developed in this paper can be more generally used in the context of online sampling under dependence. Proposed algorithms are accompanied with corresponding regret analysis.

1. Introduction

As one of the simplest examples of sequential optimization under uncertainty, multi-armed bandit problems arise in various modern real-world applications, such as online advertisement, and Internet routing. These problems are typically studied under the assumption that the pay-offs are independently and identically distributed (i.i.d.), and the arms are independent. However, this assumption does not necessarily hold in many practical situations. Consider, for example, the problem of online advertisement in which the aim is to garner as many clicks as possible from a user. Grouping adverts into categories and associating with each category an arm, this problem turns into a multi-armed bandit. There is dependence over time and across the arms since, for example, we expect a user to be more likely to select adverts that are related to her selections in the recent past.

In this paper, we consider the multi-armed bandit problem in the case where the pay-offs are dependent and each arm evolves over time regardless of whether or not it is played. This is an instance of the so-called *restless* bandit problem (Whittle, 1988; Guha et al., 2010; Ortner et al., 2014). Since in this setting an optimal policy can leverage the inter-dependencies between the samples and switch between the arms at appropriate times, it can obtain an overall pay-off much higher than that given by playing the *best arm*, i.e. the distribution with the highest expected pay-off, see Example 1 in (Ortner et al., 2014). However, finding the best such *switching strategy* is PSPACE-hard, even in the case where the process distributions are Markovian with known dynamics (Papadimitriou and Tsitsiklis, 1999). Therefore, it is useful to consider relaxations of the problem with the aim to devise computationally tractable solutions that effectively approximate the optimal switching strategy. Approximations of the Markovian restless bandit problem *under known dynamics* have been previously considered, see, e.g. (Guha et al., 2010) and references therein. Our focus in this paper is on a more general setting, where the rewards have unknown distributions, exhibit long-range dependencies, and have Markov chains as a special case.

We are interested in a sub-class of the restless bandit problem, where the pay-offs have long-range dependencies. Since the nature of the problem calls for finite-time analysis, we further require that the dependence weakens over time, so as to make use of concentration inequalities in this setting. To this end, a natural approach is to assume that the pay-off distributions are stationary φ-mixing. The so-called φ-mixing coefficients $\varphi_n,\ n \in \mathbb{N}$, of a sequence of random variables $\langle X_t \rangle_{t\in\mathbb{N}}$ measure the amount of dependence between the sub-sequences of $\langle X_t \rangle_{t\in\mathbb{N}}$ separated by n time-steps. A process is said to be φ-mixing if this dependence vanishes with n. This notion is more formally defined in Section 2. In Markov chains φ-mixing coefficients are closely related to mixing times. The mixing time of a Markov chain is a measure of how fast its distribution approaches the stationary distribution. In particular, it is defined to be the time that it takes for the distribution to be within $1/4^{\text{th}}$ of the stationary distribution as measured in total variation distance (Levin et al., 2008)[Sec 4.5]. A classical result by Davydov shows that the decrease in distance of the distribution of the Markov chain to the stationary distribution is controlled (up to a factor of $1/2$) by the φ-mixing coefficients of the Markov chain (Doukhan, 1994)[pp.88]. While φ-mixing coefficients are related to the well-studied mixing properties of Markov chains, φ-mixing processes correspond to a wide variety of stochastic processes of which Markov chains are a special case (Doukhan, 1994).

As discussed earlier, the optimal, yet notoriously infeasible strategy for this version of the problem is to switch between the arms. In this paper, we first address the question of *when* a relaxation obtained by identifying the arm with the highest stationary mean would lead to a viable approximation in this setting. For this purpose, we characterize the approximation error in terms of the amount of dependence between the pay-offs, and show that if φ_1 is small, the optimum of the relaxed problem is close to that given by the optimal switching strategy. Observe that this condition translates directly to the pay-off distributions being weakly dependent. Next, we address the question of *how* an optimistic approach can be devised to identify the best arm. To this end, we propose a UCB-type algorithm and show that it achieves logarithmic regret with respect to the highest stationary mean. Interestingly, the amount of dependence in the form of $\sum_i \varphi_i$ appears in the bound, and in the case where the pay-offs are i.i.d., we recover the regret bound of Auer et al. (2002).

A familiar real-world example for the bandit problem considered in this paper corresponds to recommendation systems where the objective is to present personalized adverts, news-feeds or Massive Open Online Course (MOOC) material to each user. In these cases, depending on the specific application, each bandit arm could correspond to an appropriate subject category, indicating, for example, a genre of products or a class of topics. Once an arm is played, an item of the corresponding category could be selected at random and presented to the user. There are long-range dependencies between the pay-offs as reflected by the users' memory of their past observations. However, the dependence naturally decays over time, while users' short-term memory, which in turn influences the dominant mixing coefficients, can be controlled by capping the maximum frequency at which each specific recommendation is given to users. With minor modifications, this example carries over to a larger class of real-world sequential decision making problems which naturally possess a dependency structure imposed by users' memory.

©2019 Steffen Grünewälder and Azadeh Khaleghi.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-547.html.

Note that even this relaxed version of the problem is far from straightforward. The main challenge lies in obtaining confidence intervals around empirical estimates of the stationary means. Since Hoeffding-type concentration bounds exist for φ-mixing processes, it may be tempting to use such inequalities directly with standard UCB algorithms designed for the i.i.d. setting, to find the best arm. However, as we demonstrate in the paper, unlike in the i.i.d. setting, a policy in this framework may introduce strong couplings between past and future pay-offs in such a way that the distribution of the sampled sequence may not even be φ-mixing. This is the reason why a standard UCB algorithm designed for i.i.d. settings is not suitable here, even when equipped with a concentration bound for φ-mixing processes. In fact, this oversight seems to have occurred in previous literature, specifically in the Improved-UCB-based approach of Audiffren and Ralaivola (2015). We refer to Section 4.1 for a more detailed discussion. We circumvent these difficulties by carefully taking *random times*[1] into account and controlling the effect of a policy on the pay-off distributions. Some of our technical results can be more generally used to address the problem of online sampling under dependence.

Finally, while the study of the multi-armed bandit problem with *strongly* dependent pay-offs at its full generality is beyond the scope of this paper, we provide a complementary example for this regime. Specifically, we consider a setting where the bandit arms are governed by stationary Gaussian processes with slowly decaying covariance functions. Such high-dependence scenarios are quite common in practice. For instance, the throughput of radio channels changes slowly over time and the problem of choosing the best channel can be modeled by a bandit problem with strongly dependent pay-offs. The intuitive reason why in this setting it may also be possible to efficiently obtain approximately optimal solutions is that the strong dependencies can allow for the prediction of future rewards even from scarce observations. We give a simple switching strategy for this instance of the problem and show that it significantly outperforms a policy that aims for the best arm. Our regret bound for this algorithm directly reflects the dependence between the pay-offs: the higher the dependence the lower the regret.

A summary of our main contributions is listed below.

i. In an attempt to derive a computationally tractable solution for the restless bandit problem, we first identify a case where the optimal switching strategy can be approximated by playing the arm with the highest stationary mean. To this end, we show that the loss of settling for the highest stationary mean as opposed to finding the best switching strategy is controlled by the amount of inter-dependence as reflected by φ_1. This is shown in Proposition 9.

ii. We provide a detailed example, namely Example 1, where we demonstrate the challenges in the non-i.i.d. bandit problem. In particular, we show that a policy in this framework may introduce strong couplings between past and future pay-offs in such a way that the resulting pay-off sequence may have a completely different dependency structure.

iii. We develop technical machinery to circumvent the difficulties introduced by the inter-dependence between the rewards, and allow us to control the effect of a policy on the pay-off distributions. Some of our derivations concerning sampling under dependence can be of independent interest. We further propose a UCB-type algorithm, namely Algorithm 1 that deploys these tools to identify the arm with the highest stationary mean.

1. These correspond to random variables which determine the time at which an arm is sampled.

iv. We provide an upper bound on the regret of Algorithm 1 (with respect to the highest stationary mean). This is provided in Theorem 12. The regret bound is a function of the amount of inter-dependence as reflected by the φ-mixing coefficients, and in the case where the pay-offs are i.i.d., we recover the regret bound of Auer et al. (2002)[Thm. 1]. This result along with Proposition 9 allow us to argue that in the case where dependence is low Algorithm 1 can be used to approximate the best switching strategy.

The remainder of the paper is organized as follows. In Section 2 we introduce preliminary notation and definitions. We formulate the problem in Section 3 and give our main results in Section 4. We conclude in Section 5 with a discussion of open problems.

2. Preliminaries

We start with some useful notation before discussing basic definitions concerning stochastic processes. Since the bandit problem involves multiple arms (processes), we extend the definition of a φ-mixing process to what we call a jointly φ-mixing process, to be able to model the multi-armed bandit problem. Indeed, in many natural settings, the process is jointly φ-mixing. For example, as we demonstrate in Proposition 5, independent Markov chains are jointly φ-mixing.

Notation. Let $\mathbb{N}_+ := \{1, 2, \ldots\}$ and $\overline{\mathbb{N}} := \mathbb{N} \cup \{\infty\}$ denote the set and extended set of natural numbers respectively. We introduce the abbreviation $\mathbf{a}_{m..n},\ m, n \in \mathbb{N}_+, m \leq n$, for sequences $a_m, a_{m+1}, \ldots, a_n$. Given a finite subset $C \subset \mathbb{N}_+$ and a sequence $\mathbf{a}$, we let $\mathbf{a}_C := \{a_i : i \in C\}$ denote the set of elements of $\mathbf{a}$ indexed by C. If X_C is a sequence of random variables indexed by $C \subset \mathbb{N}_+$, we denote by $\sigma(X_C)$ the smallest σ-algebra generated by X_C.

Notion of φ-dependence. Part of our results concern the so-called φ-dependence between σ-algebras defined as follows, see, e.g. (Doukhan, 1994).

Definition 1 *Consider a probability space $(\Omega, \mathcal{A}, P)$ and let $\mathcal{U}$ and $\mathcal{V}$ denote σ-subalgebras of $\mathcal{A}$ respectively. The φ-dependence between $\mathcal{U}$ and $\mathcal{V}$ is is given by*

$$\varphi(\mathcal{U}, \mathcal{V}) := \sup\{|P(V) - P(V|U)| : U \in \mathcal{U}, P(U) > 0, V \in \mathcal{V}\}.$$

If X and Y are two random variables measurable with respect to $\mathcal{A}$ we simplify notation by letting $\varphi(X, Y) := \varphi(\sigma(X), \sigma(Y))$ denote the φ-dependence between their corresponding σ-algebras; distinction will be clear from the context. Similarly, if X_A and X_B are finite sequences of random variables, with $A, B \subset \mathbb{N}_+$ their φ-dependence can be similarly defined as

$$\varphi(X_A, X_B) := \varphi(\sigma(X_A), \sigma(X_B)).$$

In words, $\varphi(X_A, X_B)$ measures the maximal difference between the probability of an event V and its conditional probability given an event U, where U and V are determined by random variables indexed by A and B respectively. The notion of φ-dependence carries over from probability measures to expectations. In particular, consider a real-valued random variable X defined on some probability space $(\Omega, \mathcal{A}, P)$, and denote by $\mathcal{G}$ some collected information in the form of a σ-subalgebra of $\mathcal{A}$. Let $E(X|\mathcal{G})$ denote Kolmogorov's conditional expectation, i.e. a $\mathcal{G}$-measurable random variable Z such that $\int_B Z = \int_B X$ for all $B \in \mathcal{G}$. As follows from Theorem 2 below, due to Bradley (2007)[vol. 1 pp. 124], the difference between $E(X|\mathcal{G})$ and EX is effectively upper-bounded by $\varphi(\mathcal{G}, \sigma(X))$.

Theorem 2 (Bradley (2007)) *Let $(\Omega, \mathcal{A}, P)$ be a probability space, let X be a real-valued random variable with $\|X\|_1 < \infty$ and let $\mathcal{G}$ be some σ-subalgebra of $\mathcal{A}$. Then*

$$2\varphi(\mathcal{G}, \sigma(X)) = \sup \|E(Y|\mathcal{G}) - E(Y)\|_1 / \|Y\|_1, \tag{1}$$

where the supremum is taken over all $\sigma(X)$-measurable random variables Y with $\|Y\|_1 < \infty$. Furthermore, for any $B \in \mathcal{G}$ it holds that

$$\int_B |E(X|\mathcal{G}) - E(X)|\, dP \leq 2P(B)\|X\|_\infty \varphi(\mathcal{G}, \sigma(X)). \tag{2}$$

Observe that because X is trivially $\sigma(X)$-measurable (1) given in the theorem implies that

$$\|E(X|\mathcal{G}) - E(X)\|_1 \leq 2\|X\|_1 \varphi(\mathcal{G}, \sigma(X)).$$

Stochastic Processes & φ-mixing Properties. Let $(\mathcal{X}, \mathcal{B}_\mathcal{X})$ be a measurable space; we let $\mathcal{X} \subset [0,1]$ [2] and denote by $\mathcal{B}_\mathcal{X}^{(m)}$ the Borel σ-algebra on $\mathcal{X}^m,\ m \in \mathbb{N}_+$. We denote by $\mathcal{X}^\infty$ the set of all $\mathcal{X}$-valued infinite sequences indexed by $\mathbb{N}_+$. A stochastic process can be modeled as a probability measure over the space $(\mathcal{X}^\infty, \mathcal{B})$ where $\mathcal{B}$ denotes the σ-algebra on $\mathcal{X}^\infty$ generated by the cylinder sets. Associated with the stochastic process is a sequence of random variables $X_1, X_2, \ldots$, where $X_t : \mathcal{X}^\infty \to \mathcal{X}$ is the projection onto the t'th element, i.e. $X_t(\omega) = \omega_t$ for $\omega \in \mathcal{X}^\infty$ and $t \in \mathbb{N}_+$. A process ρ is stationary if $\rho(X_{1..m} \in B) = \rho(X_{i+1..i+m} \in B)$ for all Borel sets $B \in \mathcal{B}_\mathcal{X}^{(m)}$, $i, m \in \mathbb{N}_+$. The term *stochastic process* refers to either the process distribution ρ or the associated sequence of random variables $X_t,\ t \in \mathbb{N}_+$; reference will be clear from the context.

Definition 3 (Stationary φ-mixing Process) *Consider a stationary stochastic process $\langle X_i \rangle_{i \in \mathbb{N}_+}$. Its φ-mixing coefficients are given by*

$$\varphi_n := \sup_{u,v \in \mathbb{N}_+} \varphi(X_{1..u}, X_{u+n..u+n+v-1}),\ n \in \mathbb{N}_+$$

and measure the φ-dependence between $\sigma(X_{1..u})$ and $\sigma(X_{u+n..u+n+v-1})$

$$\underbrace{X_1, \ldots, X_u} \ \leftarrow \textit{gap of length } n \rightarrow \ \underbrace{X_{u+n}, \ldots, X_{u+n+v-1}}$$

The process is said to be φ-mixing if $\lim_{n\to\infty} \varphi_n = 0$.

When modeling a bandit problem in this paper, we are concerned with some $k \in \mathbb{N}_+$ stochastic processes with a joint distribution that is stationary φ-mixing. More specifically, for a fixed $k \in \mathbb{N}$, let $(\Omega, \mathcal{A}, P)$ be a probability space where $\Omega := \Omega_1 \times \ldots \times \Omega_k$ with $\Omega_i := \mathcal{X}^\infty,\ i \in 1..k$, P a probability measure and $\mathcal{A}$ obtained via the cylinder sets. Let $\mathcal{B}_\mathcal{X}^{(m,k)}$ denote the Borel σ-algebra on $(\mathcal{X}^m)^k,\ m \in \mathbb{N}_+$. In much the same way as with the single-process described above, associated with the joint process is a sequence of random variables $\langle X_{t,j} \rangle,\ t \in \mathbb{N}_+, j \in 1..k$ where $X_{t,j} : \Omega \to \mathcal{X}$ is the projection on to the t, j-th element, i.e. $X_{t,j}(\omega) = \omega_{t,j}$ for $\omega \in \Omega,\ t \in \mathbb{N}_+, j \in 1..k$. The measure P is stationary if $P(X_{1..m,1..k} \in B) = P(X_{i+1..i+m,1..k} \in B)$ for every $B \in \mathcal{B}_\mathcal{X}^{(m,k)}$ and $i, m \in \mathbb{N}_+$. As above, the term stochastic process is used interchangeably to correspond to the sequence of random variables $\langle X_{t,j} \rangle,\ t \in \mathbb{N}_+, j \in 1..k$ or their corresponding joint measure P.

2. More generally $\mathcal{X}$ can be a finite set or a closed interval $[a, b]$, for $a < b, a, b \in \mathbb{R}$.

Definition 4 (Jointly Stationary φ-mixing Processes) *For a fixed $k \in \mathbb{N}_+$, consider a stationary process $\langle X_{t,j} \rangle,\ t \in \mathbb{N}_+, j \in 1..k$. Its φ-mixing coefficients are given by*

$$\varphi_n := \sup_{u,v \in \mathbb{N}_+} \varphi(X_{1..u,1..k}, X_{u+n..u+n+v-1,1..k}),\ n \in \mathbb{N}_+$$

and measure the φ-dependence between $\sigma(X_{1..u,1..k})$ and $\sigma(X_{u+n..u+n+v-1,1..k})$. The process is said to be jointly φ-mixing if $\lim_{n\to\infty} \varphi_n = 0$.

We use φ_n to denote the φ-mixing coefficient corresponding to a single process or to a joint process given by Definitions 3 and 4 respectively; the notion will be apparent from the context. Under the assumption that the joint process is stationary φ-mixing, there could be dependence between the processes, as the mixing requirement needs only to be fulfilled by the joint process. The assumption that the process is jointly φ-mixing is fulfilled by a variety of well-known models, including independent Markov chains. More generally, as follows from Proposition 5 below, if we have k independent ψ-mixing processes then the joint process is φ-mixing. The ψ-mixing property is defined in the same way as φ-mixing whereby the φ-dependence given by Definition 1 is replaced with

$$\psi(\mathcal{U}, \mathcal{V}) := \sup\{|1 - \rho(U \cap V)/(\rho(V)\rho(U))| : U \in \mathcal{U},\ \rho(U) > 0, V \in \mathcal{V}, \rho(V) > 0\},$$

and the ψ-mixing coefficients are defined in a manner analogous to Definitions 3. A ψ-mixing process is also φ-mixing and the ψ-mixing coefficients upper bound φ-mixing coefficients.

Proposition 5 *Let $(\Omega, \mathcal{A}, P)$ be some probability space with k mutually independent processes defined on it. If each of these processes is ψ-mixing then the joint process is also ψ-mixing and for all $i \in \mathbb{N}$, $1 + \tilde{\psi}_i \leq (1 + \psi_i)^k$, where $\tilde{\psi}_i$ are the mixing coefficients of the joint process and the ψ_i are upper-bounds on the mixing coefficients of the individual processes.*

The proof is provided in Appendix A.2. Using the above result, it can be shown that independent Markov chains are jointly φ-mixing. More specifically, we have the following example.

Corollary 6 (Independent Markov chains are jointly φ-mixing.) *A set of $k \in \mathbb{N}_+$ mutually independent, stationary ergodic, finite-state Markov processes $\rho_i,\ i = 1..k$ give rise to a jointly stationary φ-mixing process.*

Proof By Theorem 3.1 of Bradley (2005) each process $\rho_i,\ i = 1..k$ is ψ-mixing. Moreover, by Proposition 5 above, k mutually independent ψ-mixing processes are jointly φ-mixing. As a result $\rho_i,\ i = 1..k$, are jointly stationary φ-mixing. ■

The significance of the above observation is that jointly φ-mixing processes have mutually independent Markov processes as special case.

3. Problem Formulation: the Jointly φ-mixing Bandit Problem.

We assume in the sequel that a total of $k < \infty$ bandit arms are given, where for each $i \in 1..k$, arm i corresponds to a stationary process that generates a time series of pay-offs $X_{1,i}, X_{2,i}, \ldots$ Furthermore, we assume that the joint process over the k arms is φ-mixing in the sense of Definition 4, and that its sequence of mixing coefficients is summable, i.e. $\|\varphi\| := \sum_{i=1}^\infty \varphi_i < \infty$. Each process has

stationary mean $\mu_i,\ i = 1..k$ and we denote by $\mu^* := \max\{\mu_1, \ldots, \mu_k\}$ the highest stationary mean. We sometimes denote the arm with the highest stationary mean as the *best arm*. At every time-step $t \in \mathbb{N}_+$, a player chooses one of k arms according to a policy π_t and receives a reward X_{t,π_t}. The player's objective is to maximize the sum of the pay-offs received. The policy has access only to the pay-offs gained at earlier stages and to the arms it has chosen. Let $\langle \mathcal{F}_t \rangle_{t\geq 0}$ be a filtration that tracks the pay-offs obtained in the past t rounds, i.e. $\mathcal{F}_0 = \{\emptyset, \Omega\}$, and $\mathcal{F}_t = \sigma(X_{1,\pi_1}, \ldots, X_{t,\pi_t}),\ t \geq 1$. A policy is a sequence of mappings $\pi_t : \Omega \to \{1, \ldots, k\}$, $t \geq 1$, each of which is measurable with respect to $\mathcal{F}_{t-1}$. Note that the assumption that $\pi_t,\ t \geq 1$ is measurable with respect to $\mathcal{F}_{t-1}$ is equivalent to the assumption that the policy can be written as a function of the past pay-offs and chosen arms, see, e.g. (Shiryaev, 1991)[Thm. 3, pp.174]. Let $\Pi = \{\pi := \langle \pi_t \rangle_{t\geq 1} : \pi_t \text{ is } \mathcal{F}_{t-1}\text{-measurable for all } t \geq 1\}$ denote the space of all possible policies. We also let $\mathcal{G}_t$ denote the filtration that keeps track of all the information available up to time t (including unobserved pay-offs). More specifically, let $\langle \mathcal{G}_t \rangle_{t\geq 0}$, $\mathcal{G}_0 = \mathcal{N}$ and $\mathcal{G}_t = \sigma(\bigcup_{i=1}^k \bigcup_{s=1}^t \sigma(X_{s,i}) \cup \mathcal{N})$ for $t \geq 1$, where $\mathcal{N}$ is the family of measurable sets of P-measure zero. We define the maximal value that can be achieved in n rounds as

$$v_n^* = \sup_{\pi\in\Pi} \sum_{t=1}^{n} E X_{t,\pi_t}. \tag{3}$$

The regret that builds up over n rounds for any strategy π is

$$\mathcal{R}_\pi(n) := v_n^* - E(X_{t,\pi_t}). \tag{4}$$

To simplify notation, we may use $\mathcal{R}(n)$ when the policy π is clear from the context.

Remark 7 *Requiring $\|\varphi\| < \infty$, which is standard in the literature on empirical process theory involving mixing processes, does not lead to a strong assumption. This condition is already fulfilled by any stationary ergodic finite-state Markov chain, see, e.g. (Bradley, 2005). In fact, as follows from Theorem 3.4 of Bradley (2005) the φ-mixing coefficient corresponding to any (not necessarily stationary) φ-mixing Markov process has an exponentially fast rate of decay, giving rise to a summable sequence of φ-mixing coefficients. Note that our focus in the sequel is on the more general case of φ-mixing (not necessarily Markov) processes.*

4. Main Results

We consider the restless bandit problem in a setting where the reward distributions are jointly φ-mixing as formulated in Section 3. Recall that, while the optimal strategy in this case is to switch between the arms, obtaining the best switching strategy is PSPACE-hard. We address the question of *when* and *how* a good and computationally tractable approximation of the optimal policy can be obtained in this setting. The former question is answered in Section 4.2 where we characterize (in terms of φ_1) the loss of settling for the highest stationary mean as opposed to following the best switching strategy. We show that for small φ-mixing coefficients, the optimum of this relaxed problem is close to that given by v_n^*. To answer the latter, we devise a UCB-type algorithm in Section 4.3 to identify the arm with the highest stationary mean. The main challenge lies in building confidence intervals around empirical estimates of the stationary means. Indeed, as we demonstrate in Section 4.1, unlike in the i.i.d. setting, a policy in this framework may introduce strong couplings between past and future pay-offs in such a way that the resulting pay-off sequence may not even be φ-mixing. As a result, a standard UCB algorithm designed for an i.i.d. setting is not suitable

here, even when equipped with a Hoeffding-type concentration bound for φ-mixing processes. We circumvent these difficulties in Sections 4.2 and 4.3 by controlling the effect of a policy on the pay-off distributions. Part of the analysis in these two sections relies on some technical results for φ-mixing processes outlined in Appendix A.1, which may be of independent interest. Finally, our results for the weakly dependent reward distributions are complemented in Section 4.4, where we study an example of a class of strongly dependent processes and give a simple switching strategy that significantly outperforms a policy that aims for the best arm.

4.1. Policies, Random Times, and the φ-mixing Property

Recall that a policy $\pi_t,\ t \in \mathbb{N}_+$ is a function which based on the past (observed) data samples one of k bandit arms at time-step t. Therefore, since this decision is based on samples generated by a random process, the times at which bandit arms are played can be naturally modeled via *random times*. More formally, denote by $\tau_{i,j} : \Omega \to \mathbb{N}_+$ a random variable which determines the time at which the j-th arm is sampled for the i-th time, $i \in \mathbb{N}_+$ and $j \in 1..k$, and observe that for any $t \in \mathbb{N}$ the event $\{\tau_{i,j} = t\}$ is in $\mathcal{F}_{t-1}$. We denote by

$$X_{\tau_i,j} := \sum_{t\in\mathbb{N}_+} \chi\{\tau_{i,j} = t\} X_{t,j}$$

the pay-off obtained from sampling arm j at random time $\tau_{i,j}$ for any $i \in \mathbb{N}_+,\ j \in 1..k$, where χ is the indicator function, i.e. $\chi\{\tau_{i,j} = t\}$ is 1 for all $\omega \in \Omega$ such that $\tau_{i,j}(\omega) = t$, and is 0 otherwise.

The main challenge in devising a policy for the φ-mixing bandit problem is that, depending on the policy used, the dependence structure of the pay-off sequence $X_{\tau_1,j},\ X_{\tau_2,j}, \ldots$, for $j \in 1..k$ may be completely different from that of $X_{1,j},\ X_{2,j}, \ldots$. Note that this differs from the simpler i.i.d. setting where the distribution of the j-th arm (with all its characteristics) carries over to that of the sampled sequence. This is illustrated in Example 1 below.

Example 1 *Consider a two-armed bandit problem where the second arm is deterministically set to 0, i.e. $X_{t,2} = 0,\ t \in \mathbb{N}_+$ and the first arm has a process distribution described by a two state Markov chain with the initial distribution and stationary distribution over the states both being $(1/2\ \ 1/2)^\top$ and with the following transition matrix,*

$$T = \begin{pmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{pmatrix}, \quad \text{with some } \epsilon \in (0, 1).$$

Observe that for this process, if ϵ is small, with high probability the Markov chain stays in its current state. Now consider a policy π, and denote by $\tau_1, \tau_2, \ldots$ the sequence of random times at which π samples the first arm according to the following simple rule. Set $\tau_1 = 1$. For subsequent random times, if $X_{\tau_n,1} = X_{1,1}$ for $n \in \mathbb{N}_+$ then $\tau_{n+1} = \tau_n + 1$. Otherwise, τ_{n+1} is set to be significantly larger than τ_n to guarantee that the distribution of $X_{\tau_{n+1},1}$ given $X_{\tau_n,1}$ is close to the stationary distribution of the Markov chain, during which time the first arm is sampled. The sequence $X_{\tau_1,1}, X_{\tau_2,1}, \ldots$ so generated is highly dependent on $X_{1,1}$ and is not φ-mixing. In fact, the expected pay-offs given the first observation, i.e. $E(X_{\tau_n,1}|X_{1,1}),\ n \in \mathbb{N}_+$, are very different from the stationary mean $EX_{1,1}$ if ϵ is small. In particular, $EX_{1,1} = 0.5$ while $E(X_{\tau_n,1}|X_{1,1} = 1)$ is at least $1/(1 + 2\epsilon) - 1/10$ due to Equation (41) on page 28 and, hence, for $\epsilon \leq 0.01$ we have that $E(X_{\tau_n,1}|X_{1,1} = 1) \geq 0.8$.

A more detailed treatment of the above example is given in Appendix A.4.

Indeed, it is a policy's access to the (observed) past data which can lead to strong couplings between past and future pay-offs in this framework. This point has been overlooked in the work of Audiffren and Ralaivola (2015) which relies on Improved-UCB (Auer and Ortner, 2010) to identify the arm with the highest stationary mean, by eliminating potentially sub-optimal arms. The elimination process depends on the data, and the time-steps at which a particular arm is played depend on the remaining arms. Hence, random times and the policy's memory have to be carefully taken into consideration, as the process distribution of the sampled sequence is different from that of the corresponding arm. This notion has not been accounted for in their algorithm, and the confidence intervals involved correspond to the distributions of the arms, and not to those of the sampled sequences, and are therefore invalid in this non-i.i.d. setting.

4.2. Approximation Error

We start by translating φ-mixing properties to those of expectations in order to control the difference between what a switching strategy can achieve as compared to the highest stationary mean. Prior to delving into the bandit problem we consider a single bounded real-valued stationary φ-mixing process $\langle X_t\rangle_{t\in\mathbb{N}_+}$ sampled at random times $\tau_1 < \tau_2 < \ldots$, where $\tau_i,\ i \in \mathbb{N}_+$, is fully defined by the past observations $X_{\tau_1}, \ldots, X_{\tau_{i-1}}$. We control the difference between the mean of the sampled process $\langle X_{\tau_i}\rangle_{i\in\mathbb{N}_+}$ from the stationary mean of the original process. The following proposition shows that the difference in the means is controlled by the φ-mixing coefficients and the increments in the stopping times; the proof is provided in Appendix A.3.

Proposition 8 *Assume that $\langle X_t\rangle_{t\in\mathbb{N}_+}$ is a stationary φ-mixing process with mixing coefficients $\langle\varphi_i\rangle_{i\in\mathbb{N}_+}$ such that $EX_t = \mu,\ t \in \mathbb{N}_+$, and $\sup_{t\in\mathbb{N}_+} |X_t| \leq c$ for some $c \in [0,\infty)$. Furthermore, let $\tau_1, \tau_2, \ldots$ be a sequence of random times such that $\tau_i + \ell \leq \tau_{i+1}$ a.s. for some $\ell \geq 1$ and all $i \in \mathbb{N}_+$, and all τ_{i+1} are $\sigma(X_{\tau_1}, \ldots, X_{\tau_i})$- measurable with $\tau_1 \in \mathbb{N}_+$ being a fixed time. Then for any $n \in \mathbb{N}_+$*

$$\Big|\frac{1}{n}\sum_{i=1}^{n} EX_{\tau_i} - \mu\Big| \leq 2c\varphi_\ell.$$

In other words, when using the sample mean of the sampled process as an estimate of the stationary mean then the bias of the estimator is bounded through the φ_ℓ-mixing coefficient. This result has further implications, in particular, it is telling us something about the leverage a switching policy has. A switching policy selects effectively random times for each arm at which the arm is played and this result is saying that the summed pay-off it can gather cannot be more than $2cn\varphi_\ell$ larger than the stationary mean of the arm. Now, a policy is free to play the arms at any time and we only know that $\tau_{i+1} \geq \tau_i + 1$ for any random time τ_i, i.e. $\ell = 1$. This intuition underlies the following proposition.

Proposition 9 *Consider the jointly stationary φ-mixing bandit problem formulated in Section 3. Let $\mu_1, \ldots, \mu_k$ be the means of the stationary distributions and let $\mu^* := \max\{\mu_1, \ldots, \mu_k\}$. Let φ_1 be the first φ-mixing coefficient as given by Definition 4. For every $n \geq 1$ we have*

$$v_n^* - n\mu^* \leq 2n\varphi_1.$$

Proof Consider an arbitrary policy $\langle\pi_t\rangle_{t\in\mathbb{N}_+}$ and an arbitrary $t \in \mathbb{N}_+$ then

$$EX_{t,\pi_t} = \sum_{j=1}^{k}\sum_{i=1}^{t} EX_{t,j} \times \chi\{\tau_{i,j} = t\}.$$

Recall the definition of $\mathcal{G}_{t-1}$ in Section 3 and observe that $\tau_{i,j}$ is $\mathcal{G}_{t-1}$-measurable. Hence, we have with $B = \{\tau_{i,j} = t\}$ that

$$EX_{t,j} \times \chi\{\tau_{i,j} = t\} = EE(X_{t,j}|\mathcal{G}_{t-1}) \times \chi\{\tau_{i,j} = t\} = \int_B E(X_{t,j}|\mathcal{G}_{t-1}).$$

We can extend the φ-mixing property of the joint process from $\sigma(X_{11}, \ldots, X_{tk})$to $\mathcal{G}_t$ by applying Lemma A.2 and we get

$$\Big|\int_B (E(X_{t,j}|\mathcal{G}_{t-1}) - EX_{1,j})\Big| \leq 2\varphi_1 P(B).$$

Since the different sets $\{\tau_{i,j} = t\}, j \in \{1, \ldots, k\}, i \in \{1, \ldots, t\}$ are disjoint

$$\begin{aligned} EX_{t,\pi_t} - \mu^* &\leq \sum_{j=1}^{k}\sum_{i=1}^{t}(EX_{t,j} \times \chi\{\tau_{i,j} = t\} - P(\tau_{i,j} = t)EX_{1,j}) \\ &\leq 2\varphi_1 \sum_{j=1}^{k}\sum_{i=1}^{t} P(\tau_{i,j} = t) \\ &\leq 2\varphi_1. \end{aligned}$$

■

Observe that this relaxation introduces an inevitable linear component to the regret as shown by Proposition 9. However, we argue that if the reward distributions are weakly dependent in the sense that φ_1 is small, we may settle for the best arm instead of following the best switching strategy.

4.3. An Optimistic Approach

In this section we propose a UCB-type algorithm to identify the arm with the highest stationary mean in a jointly φ-mixing bandit problem. Consider the bandit problem described in Section 3, where we have k arms each with a bounded stationary pay-off sequence such that the joint process is stationary φ-mixing. Suppose that the processes are weakly dependent in the sense that $\varphi_1 \leq \epsilon$ for some small ϵ. As discussed in Section 4.2, in this case a policy to settle for the best arm can serve as a good approximation for the best switching strategy. More specifically, let

$$\overline{\mathcal{R}}_\pi(n) := n\mu^* - \sum_{t=1}^{n} EX_{t,\pi_t}$$

denote the regret of a policy π with respect to the arm with the highest stationary mean. From Proposition 9 we have $\frac{1}{n}(\mathcal{R}_\pi(n) - \overline{\mathcal{R}}_\pi(n)) \leq \epsilon$ and our objective in this section is to minimize $\overline{\mathcal{R}}_\pi$.

Recall that in light of the arguments provided in Section 4.1 it is crucial to take a policy's access to past (observed) data into account when devising a strategy for the bandit problem in this framework.

To address the challenge induced by the inter-dependent reward sequences obtained at random times, our approach relies on the following key observation. Suppose we obtain a sequence of $m \in \mathbb{N}_+$ consecutive samples $X_{\tau,i}, X_{\tau+1,i}, \ldots, X_{\tau+m,i}$ from arm $i \in 1..k$ starting at a random time τ. For a long batch, i.e. large enough m, the average expectations $\frac{1}{m}\sum_{j=0}^{m-1} EX_{\tau+j,i}$ become close to the stationary mean μ_i. More formally we have Lemma 10 below.

Lemma 10 *For a fixed $i \in 1..k$ and $m \in \mathbb{N}_+$, consider the consecutive samples $X_{\tau,i}, X_{\tau+1,i}, \ldots, X_{\tau+m,i}$, where $\tau : \Omega \to \mathbb{N}_+$ is a random time at which the i-th arm is sampled. Let μ_i denote the stationary mean of arm i. We have*

$$\left|\mu_i - \frac{1}{m}\sum_{j=0}^{m-1} EX_{\tau+j,i}\right| \leq \frac{2}{m}\left\|\varphi\right\|.$$

Proof For simplicity of notation we denote $X_{t,i}$ by X_t and $X_{\tau,i}$ by X_τ. Recall that $\mathcal{G}_t$ denotes the filtration that keeps track of all the information available up to time t (including unobserved pay-offs). Observe that $\chi\{\tau = t\}$ is $\mathcal{G}_{t-1}$-measurable so that the event $\{\tau = t\}$ is in $\mathcal{G}_{t-1}$ for all $t \in \mathbb{N}_+$. As a result, for any $t \in \mathbb{N}_+$ and $j \in 0..m-1$ we have

$$E(\chi\{\tau = t\}X_{t+j}|\mathcal{G}_{t-1}) = \chi\{\tau = t\}E(X_{t+j}|\mathcal{G}_{t-1}), \tag{5}$$

see, e.g. (Shiryaev, 1991)[pp.216]. We obtain

$$\begin{aligned}
\left|m\mu_i - \sum_{j=0}^{m-1} EX_{\tau+j,i}\right| &= \left|\sum_{t\in\mathbb{N}_+}\sum_{j=0}^{m-1} E(\chi\{\tau = t\}X_{t+j}) - E\chi\{\tau = t\}EX_t\right| && (6)\\
&= \left|\sum_{t\in\mathbb{N}_+}\sum_{j=0}^{m-1} EE(\chi\{\tau = t\}X_{t+j}|\mathcal{G}_{t-1}) - E\chi\{\tau = t\}EX_t\right| && (7)\\
&= \left|\sum_{t\in\mathbb{N}_+}\sum_{j=0}^{m-1} E(\chi\{\tau = t\}E(X_{t+j}|\mathcal{G}_{t-1})) - E\chi\{\tau = t\}EX_t\right| && (8)\\
&\leq \sum_{t\in\mathbb{N}_+}\sum_{j=0}^{m-1} E\big(\chi\{\tau = t\}|E(X_{t+j}|\mathcal{G}_{t-1}) - EX_t|\big) && (9)\\
&= \sum_{t\in\mathbb{N}_+}\sum_{j=0}^{m-1} E\big(\chi\{\tau = t\}|E(X_{t+j}|\mathcal{G}_{t-1}) - EX_{t+j}|\big) && (10)\\
&= \sum_{t\in\mathbb{N}_+}\sum_{j=0}^{m-1} \int_{\{\tau=t\}} |E(X_{t+j}|\mathcal{G}_{t-1}) - EX_{t+j}|\\
&\leq \sum_{t\in\mathbb{N}_+}\sum_{j=0}^{m-1} 2\varphi(\mathcal{G}_{t-1}, \sigma(X_{t+j}))E\chi\{\tau = t\} && (11)\\
&\leq \sum_{t\in\mathbb{N}_+}\sum_{j=1}^{m} 2\varphi_j E\chi\{\tau = t\} && (12)\\
&\leq 2\left\|\varphi\right\| && (13)
\end{aligned}$$

Algorithm 1 A UCB-type Algorithm for φ-mixing bandits.

Input: Number k of arms; Sum $\|\varphi\| := \sum_{i=1}^{\infty}\varphi_i$ of mixing coefficients[3]

Initialization: Play each arm once and update the empirical mean for each arm

for $i = 1..k$ **do**
  $\overline{X}_i \leftarrow X_i$
  $s_i \leftarrow 1$ ▷ $s_i,\ i = 1..k$ denotes the number of times arm i has been *selected*

Main Loop:

for $t = 1..\infty$ **do**

Arm Selection: Select the arm that maximizes the following UCB

$$j \leftarrow \min\left(\operatorname*{argmax}_{u\in 1..k} \overline{X}_u + \sqrt{\frac{8\xi(\frac{1}{8}+\ln t)}{2^{s_u}}} + \frac{\|\varphi\|}{2^{s_u-1}}\right),\ \text{where } \xi := 1 + 8\left\|\varphi\right\|$$

▷ *The min operator is used to give precedence to the smaller arm-index in the case of a tie.*

Update: Play arm j for 2^{s_j} consecutive iterations and update the empirical mean accordingly.

$$t_j \leftarrow t \qquad t \leftarrow t + 2^{s_j} \qquad \overline{X}_j \leftarrow \frac{1}{2^{s_j}}\sum_{t'=t_j}^{t-1} X_{t',j} \qquad s_j \leftarrow s_j + 1$$

where (6) and (7) are due to stationarity and the law of total expectation respectively, (8) follows from (5), (10) follows from stationarity, (11) follows from Theorem 2, namely Inequality (2), and noting that $\|X\|_\infty = 1$, (12) follows directly from Definition 4, and (13) follows from the definition of $\|\varphi\|$. ■

Inspired by this result, we provide Algorithm 1 which, given the number k of arms and the sum $\|\varphi\|$ of the φ-mixing coefficients, works as follows. First, each arm is sampled once for initialization. Next, from $t = k+1$ on, arms are played in batches of exponentially growing length. Specifically, at each round arm j with the highest upper-confidence on its empirical mean is selected, and played for 2^{s_j} consecutive time-steps, where s_j denotes the number of times that arm j has been selected so far. The upper confidence bound is calculated based on a Hoeffding-type bound for φ-mixing processes given by Corollary 2.1 of Rio (1999). The 2^{s_j} samples obtained by playing the selected arm are used in turn to calculate (from scratch) the arm's empirical mean. The algorithm does not require the values of the individual φ-mixing coefficients, but only their sum $\|\varphi\|$. In fact, any upper-bound $\vartheta \geq \|\varphi\|$ may be used, in which case ϑ would replace $\|\varphi\|$ in the regret bound of Theorem 12.

To analyze the regret of Algorithm 1, first recall that in an i.i.d. setting we trivially have $\overline{\mathcal{R}}(n) = n\mu^* - \mu_j\sum_{j=1}^k ET_j(n) = \sum_{j=1}^k \Delta_j ET_j(n)$ where $T_j(n)$ is the total number of times that arm j is played by the algorithm in n rounds and $\Delta_j := \mu^* - \mu_j,\ j \in 1..k$. In our framework, this equality does not necessarily hold due to the inter-dependencies between the pay-offs. However, as shown in Proposition 11 below, an analogous result *in the form of an upper-bound* holds for our algorithm.

3. Note that only the sum of the coefficients is needed, not the individual coefficients φ_i.

Proposition 11 *Consider the regret $\overline{\mathcal{R}}(n)$ of Algorithm 1 after n rounds of play. We have,*

$$\overline{\mathcal{R}}(n) \leq \sum_{j=1}^{k} \Delta_j E T_j(n) + 2k \left(\sum_{l=0}^{n} \varphi_l \right) \log n$$

Proof Denote by $\tau_{i,j} : \Omega \to \mathbb{N}_+$ the random time at which the j-th arm is sampled for the i-th time. Note that for any $t \in \mathbb{N}$ the event $\{\tau_{i,j} = t\}$ is measurable with respect to the filtration $\mathcal{G}_{t-1}$ that keeps track of all the information available up to time t. First note that

$$\begin{aligned} E(\chi\{\tau_{i,j} = t\} X_{t+l,j}) &= E E(\chi\{\tau_{i,j} = t\} X_{t+l,j} | \mathcal{G}_{t-1}) \\ &= E\left(\chi\{\tau_{i,j} = t\} E\left(X_{t+l,j} | \mathcal{G}_{t-1}\right)\right) \\ &\geq (\mu_j - 2\varphi_l) P\left(\tau_{i,j} = t\right) \end{aligned} \tag{14}$$

where the second equality follows from the fact that the event $\{\tau_{i,j} = t\}$ is $\mathcal{G}_{t-1}$-measurable and (14) follows from Theorem 2. We have,

$$\begin{aligned} \overline{\mathcal{R}}(n) &= n\mu^* - \sum_{t=1}^{n} E X_{t,\pi_t} \\ &= n\mu^* - \sum_{t=1}^{n} \sum_{j=1}^{k} \sum_{m=1}^{\log n} \sum_{l=0}^{\min\{2^m - 1, n-t\}} E\chi\{\tau_{m,j} = t\} X_{t+l,j} \\ &\leq n\mu^* - \sum_{t=1}^{n} \sum_{j=1}^{k} \sum_{m=1}^{\log n} \sum_{l=0}^{\min\{2^m - 1, n-t\}} P\left(\tau_{m,j} = t\right) (\mu_j - 2\varphi_l) \\ &\leq n\mu^* - \sum_{t=1}^{n} \sum_{j=1}^{k} \sum_{m=1}^{\log n} \sum_{l=0}^{\min\{2^m - 1, n-t\}} P\left(\tau_{m,j} = t\right) \mu_j + 2k \left(\sum_{l=0}^{n} \varphi_l \right) \log n \\ &= \sum_{j=1}^{k} \Delta_j E T_j(n) + 2k \left(\sum_{l=0}^{n} \varphi_l \right) \log n \end{aligned}$$

where the last inequality follows from (14).

■

An upper-bound on $\overline{\mathcal{R}}(n)$ is given by Theorem 12 below with proof provided in Appendix B. Recall that $\Delta_i := \mu^* - \mu_i$ is the difference between the highest stationary mean μ^* and the stationary mean μ_i of arm i.

Theorem 12 (Regret Bound.) *For the regret $\overline{\mathcal{R}}(n)$ of Algorithm 1 after n rounds of play. We have,*

$$\overline{\mathcal{R}}(n) \leq \sum_{\substack{i=1 \\ \mu_i \neq \mu^*}}^{k} \frac{32(1 + 8\|\varphi\|)\ln n}{\Delta_i} + (1 + 2\pi^2/3)(\sum_{i=1}^{k} \Delta_i) + \|\varphi\| \log n$$

Remark. Interestingly, $\|\varphi\|$ appears in the bound of Theorem 12. Indeed, in the case where the pay-offs are i.i.d., we recover (up to some constant) the regret bound of Auer et al. (2002)[Thm. 1].

4.4. Strongly Dependent Reward Distributions: a Complementary Example

At the other end of the extreme lie bandit problems with strongly dependent pay-off distributions. Our objective in this section is to give an example where a simple switching strategy can be obtained in this case to leverage the strong inter-dependencies between the samples. This approach gives a much higher overall pay-off than what would be given by settling for the best arm, and is computationally efficient. The intuition is that in many cases strong dependencies may allow for the prediction of future rewards even from scarce observations of a sample path.

We consider a class of stochastic processes for which we can easily control the level of dependency. A natural choice is to use stationary Gaussian processes on $\mathbb{N}_+$. Recall that a Gaussian process is fully specified by its covariance function cov: $\mathbb{N}_+ \times \mathbb{N}_+ \to \mathbb{R}$. Also, for any covariance function Kolmogorov's consistency theorem guarantees the existence of a Gaussian process with this particular covariance function, see, e.g. (Giné and Nickl, 2016). A Gaussian process is stationary if it has constant mean on $\mathbb{N}_+$ and its covariance can be written as $\mathrm{cov}(s) = \mathrm{cov}(t, t+s)$ for all $t \in \mathbb{N}_+, s \in \mathbb{N}$. We measure the degree of dependence of the process by means of Hölder-continuity of the covariance function. In particular, we assume that there exists some $c > 0$ and $\alpha \in (0, 1]$ such that for all $s, t \in \mathbb{N}$, $|\mathrm{cov}(s) - \mathrm{cov}(t)| \leq c|s - t|^\alpha$ and we assume that the covariance function is non-negative. A low c and α correspond to highly dependent processes since the covariance decreases slowly over time. A slowly decreasing covariance also implies large φ-mixing coefficients, since $|\mathrm{cov}(t)|/\mathrm{cov}^2(0) \leq 2(\varphi(X_0, X_t)\varphi(X_t, X_0))^{1/2}$, by Rio (1999)[Thm. 1.4]. Consider a k-armed bandit problem where each arm is distributed according to a stationary Gaussian process with stationary mean μ_i, $i = 1..k$. For simplicity, we assume that the processes are mutually independent with the same, *unknown*, covariance function $\mathrm{cov}(\cdot)$. While $\mathrm{cov}(\cdot)$ is assumed unknown, we have access to an upper-bound on its rate of decay. That is, we are given constants c and α such that $\mathrm{cov}(\cdot)$ is α, c-Hölder continuous. We further assume that an upper bound on the stationary means of the processes is known and that $\mathrm{cov}(0) = 1$. In order to obtain direct control on the regret $\mathcal{R}_\pi(n)$ we would need to make inference about the best possible switching strategy. Instead, we provide guarantees for the regret $\mathcal{R}^+_\pi(n)$ of policy π with respect to the best policy that can choose arms in hindsight, i.e. $\mathcal{R}^+_\pi(n) := \sum_{t=1}^{n} E(\max_{i \leq k}(X_{t,i} - X_{t,\pi_t}))$. Note that $\mathcal{R}^+_\pi(n) \geq \mathcal{R}_\pi(n)$.

We provide a simple algorithm, namely, Algorithm 2, that exploits the dependence between the pay-offs. Starting from an exploration phase, the algorithm alternates between exploration and exploitation, denoted Phase I and Phase II respectively. In Phase I it sweeps through all k arms to observe the corresponding pay-offs. In Phase II it plays the arm with the highest observed pay-off for $m - k$ rounds, where m is a (large) constant given by (1) which reflects the degree of dependence between the samples in the processes. We need not estimate the stationary distributions in this algorithm, as bounds on the differences between the stationary means suffice. Indeed, these differences are of minor relevance unless they are high as compared to the dependence between the individual processes. We have the following regret bound whose proof is given in Appendix C.2.

Proposition 13 *Given Δ such that $\max_{i,j} |\mu_i - \mu_j| \leq \Delta$ and α, c such that $|1 - \mathrm{cov}(t)| \leq ct^\alpha$ for all $t \geq 0$, the regret $\mathcal{R}^+(n)$ of Algorithm 2 after n rounds is at most,*

$$(n + m^\star)k(k-1)\left(\frac{\Delta + \sqrt{2}}{m^\star} + \frac{a_{m^\star} c^{1/2}}{8\pi(1 - b_{m^\star})} \left(2\pi^{1/2} - (1 - \Delta\sqrt{b_{m^\star}/4}) \exp\left(-\frac{\Delta^2 b_{m^\star}}{4} \right) \right) \right),$$

with $a_{m^\star} = 8c(m^\star)^\alpha, b_{m^\star} = c((m^\star - k)^\alpha + k^\alpha)$.

Algorithm 2 An Algorithm for highly dependent arms.

Input: a bound Δ on the difference between the stationary means, $\max_{i,j\leq k}|\mu_i-\mu_j|\leq\Delta$; Hölder coefficients α, c such that $|1-\mathrm{cov}(t)|\leq ct^\alpha$ for all $t\geq 0$.

$$m^\star \leftarrow \left\lceil \left(\frac{\sqrt{\pi}(\Delta+\sqrt{2})}{2\alpha c^{3/2}}\right)^{\frac{1}{1+\alpha}} \right\rceil .$$

if $m^\star \leq k$ **then** $m^\star \leftarrow k+1$

if $\Delta < \sqrt{8(m^*)^\alpha}/(\sqrt{2c}(m^\star-k)^\alpha+k^\alpha))$ **then**

$$m^\star \leftarrow \min \operatorname*{argmax}_{m\in\mathbb{N}}\{m : \Delta \geq \sqrt{8m^\alpha}/(\sqrt{2c}(m-k)^\alpha+k^\alpha))\}.$$

for $l=0\ldots\infty$ **do**

  Phase I: observe pay-offs $X_{lm^\star+1,1}=x_1,\ldots,X_{lm^\star+k,k}=x_k$

$$i^* \leftarrow \min \operatorname*{argmax}_{i\leq k}\{x_1,\ldots,x_k\}$$

  Phase II: play arm i^* for $m^\star-k$ steps.

To interpret the bound consider for simplicity $\alpha=1$ and the case of a highly dependent process and, hence, a very small c. If we choose to play the arm with the highest stationary mean at all rounds, then standard bounds on the normal distribution give us a bound of order $n\exp(-(\Delta^2/4))$ on the regret. In this case, the regret of Algorithm 2 is of order

$$nk^2c^{3/4}$$

because $b_{m^\star}=cm^\star\approx c^{1/4}\pi^{1/4}((\Delta+\sqrt{2})/2)^{1/2}$ is insignificant for small c, $c^{1/2}a_{m^\star}=8c^{3/2}m^\star\approx 8c^{3/4}\pi^{1/4}((\Delta+\sqrt{2})/2)^{1/2}$ and the bracket on the right side is about $2\pi^{1/2}-1$. The gap Δ itself is not of high importance because in Phase I the algorithm selects the arm with highest current pay-off and the value stays stable over a long period as c is small. Both regret bounds are linear in n because the oracle has a significant advantage in this setting: at any given time t the oracle chooses the arm with the highest pay-off in hindsight. However, for a moderate number of arms $k^2c^{3/4}$ is significantly smaller than $\exp(-\Delta^2/4)$ unless Δ is considerably large. The advantage of the switching algorithm vanishes if k is large compared to the smoothness of the process, because eventually the exploration phase (Phase I) will dominate and the smoothness of the arms cannot be exploited by this algorithm. This example demonstrates that large dependence in the stochastic process can be exploited to build switching algorithms that have a significant edge over algorithms that aim to select a single arm and algorithms like Algorithm 1 are outperformed by simple switching algorithms.

5. Outlook

This paper is an initial attempt to characterize special sub-classes of the restless bandit problem where good approximate solutions can be found using simple, computationally tractable approaches. We provide a UCB-type algorithm to approximate the optimal strategy in the case where the pay-offs

are jointly stationary φ-mixing and are only weakly dependent. A natural open problem here is the derivation of a lower-bound. Moreover, while our algorithm only requires knowledge of the sum of the φ-mixing coefficients $\|\varphi\|=\sum_i\varphi_i$ as opposed to that of each individual φ_i, the online estimation of the mixing coefficients can prove useful. Specifically, in light of Proposition 9, if φ_1 is estimated from data, the algorithm can have a real-time estimate of its maximum loss with respect to the best switching strategy after n rounds of play. Further, the results can be strengthened if the algorithm can adaptively estimate $\|\varphi\|$ instead of relying on it as input. Another interesting regime corresponds to strongly dependent pay-off distributions. We provide an example using stationary Gaussian Processes where a simple switching strategy can leverage the dependencies to outperform a best arm policy. An open problem would be to weaken the assumptions on the process distributions and obtain results analogous to the weakly dependent case for the strongly dependent framework.

Acknowledgments.

We are grateful to the anonymous reviewers whose comments have greatly contributed to the readability and organization of the paper.

Appendix A. Proofs for φ-mixing Processes

A.1. Technical Results

The following key lemma allows us to control φ-mixing coefficients corresponding to disjoint events.

Lemma A.1 *Let $(\Omega,\mathcal{A},P)$ be a probability space and let $\mathcal{B},\mathcal{C}$ be two σ-subalgebras of $\mathcal{A}$. If there exists a $\varphi\geq 0$ such that for all $B\in\mathcal{B}$ and $C\in\mathcal{C}$ it holds that $|P(B)P(C)-P(B\cap C)|\leq\varphi P(C)$ then for any disjoint sequence $\langle B_n\rangle_{n\in\mathbb{N}}$, $B_n\in\mathcal{B}$ for all $n\in\mathbb{N}$, and any $C\in\mathcal{C}$, we have*

$$\sum_{n=0}^{\infty}|P(B_n)P(C)-P(B_n\cap C)|\leq 2\varphi P(C).$$

Proof Let $c_n=P(B_n)P(C)-P(B_n\cap C)$, $I_+=\{n\in\mathbb{N}:c_n\geq 0\}$, $I_-=\{n\in\mathbb{N}:c_n<0\}$. Now, since $\bigcup_{n\in I_+}B_n\in\mathcal{B}$ and $\bigcup_{n\in I_-}B_n\in\mathcal{B}$ we have

$$\begin{aligned}P(C)\varphi &\geq P\Big(\bigcup_{n\in I_+}B_n\Big)P(C)-P\Big(\Big(\bigcup_{n\in I_+}B_n\Big)\cap C\Big)\\ &=\sum_{n\in I_+}(P(B_n)P(C)-P(B_n\cap C))=\sum_{n\in I_+}|P(B_n)P(C)-P(B_n\cap C)|\,.\end{aligned}$$

Similarly, $P(C)\varphi\geq\sum_{n\in I_-}|P(B_n)P(C)-P(B_n\cap C)|$. Hence

$$\begin{aligned}2P(C)\varphi &\geq\sum_{n\in I_+}|P(B_n)P(C)-P(B_n\cap C)|+\sum_{n\in I_-}|P(B_n)P(C)-P(B_n\cap C)|\\ &=\sum_{n\in\mathbb{N}}|P(B_n)P(C)-P(B_n\cap C)|\,.\end{aligned}$$

■

The following technical lemma is used in the proof of Proposition 9.

Lemma A.2 *Given some probability space $(\Omega, \mathcal{A}, P)$, three σ-subalgebras $\mathcal{B}, \mathcal{C}, \mathcal{D} \subset \mathcal{A}$ and $\varphi > 0$ such that $|P(U)P(V) - P(U \cap V)| \leq \varphi P(V)$ for all $U \in \mathcal{B}, V \in \mathcal{C}$ and $P(A) = 0$ for all $A \in \mathcal{D}$ then it holds that $|P(U)P(V) - P(U \cap V)| \leq \varphi P(V)$ for all $U \in \mathcal{B}, V \in \sigma(\mathcal{C} \cup \mathcal{D})$.*

Proof **(i)** The set

$$\mathcal{E} := \{A : A \in \mathcal{A}, \text{ there exists a } B \in \mathcal{C} \text{ such that for all } C \in \mathcal{A}, P(A \cap C) = P(B \cap C)\}$$

is a Dynkin system (Fremlin, 2010)[136A]. To see this, note that (1) $\emptyset \in \mathcal{E}$ since $\emptyset \in \mathcal{C}$, (2) for $A \in \mathcal{E}$ take any $B \in \mathcal{C}$ and observe that $P(C \cap (\Omega \backslash A)) = P(C \backslash (C \cap A)) = P(C) - P(C \cap A) = P(C) - P(C \cap B) = P(C \cap (\Omega \backslash B))$ for every $C \in \mathcal{A}$ and $\Omega \backslash B \in \mathcal{C}$, (3) let $\langle A_n \rangle_{n \in \mathbb{N}}$ be a disjoint sequence in $\mathcal{E}$ with corresponding elements $B_n \in \mathcal{C}$ then $P(\bigcup_{n \in \mathbb{N}} A_n) = \sum_{n=0}^{\infty} P(A_n) = \sum_{n=0}^{\infty} P(B_n)$. Let $B'_0 = B_0$ and iteratively let $B'_n = B_n \backslash B'_{n-1}$ then $\langle B'_n \rangle_{n \in \mathbb{N}}$ is a disjoint sequence such that $P(B_n) = P(B'_n)$: for any $m, n \in \mathbb{N}$ we have $P(B_n \cap B_m) = P(A_n \cap B_m) = P(A_n \cap A_m) = 0$ and, hence, $P(B_n \backslash B'_{n-1}) \geq P(B_n) - \sum_{i=0}^{n-1} P(B_n \cap B_i) = P(B_n)$. Therefore, $\sum_{n=0}^{\infty} P(B_n) = \sum_{n=0}^{\infty} P(B'_n) = P(\bigcup_{n \in \mathbb{N}} B'_n) = P(\bigcup_{n \in \mathbb{N}} B_n)$ and $\mathcal{E}$ is a Dynkin system.

(ii) Let $\mathcal{D}' := \{A \cap B : A \in \mathcal{C}, B \in \mathcal{D}\}$. Observe that $\mathcal{C} \cup \mathcal{D} \cup \mathcal{D}'$ is closed under intersection: if $A, B \in \mathcal{C}$ then $A \cap B \in \mathcal{C}$ and similarly for $A, B \in \mathcal{D}$. If $A \in \mathcal{C}$ and $B \in \mathcal{D}$ then $A \cap B$ is in an element of $\mathcal{D}'$. If $A \in \mathcal{C}$ and B is an element of $\mathcal{D}'$ then $B = C \cap D$ for some $C \in \mathcal{C}$, $D \in \mathcal{D}$ and $A \cap B = (A \cap C) \cap D$ which is again an element of $\mathcal{D}'$ (and equivalently for $A \in \mathcal{D}$). Furthermore, $\mathcal{C} \cup \mathcal{D} \cup \{A \cap B : A \in \mathcal{C}, B \in \mathcal{D}\} \subset \mathcal{E}$. Note that $\mathcal{C}$ is a subset of $\mathcal{E}$, and if $B \in \mathcal{D}$ then we obtain $P(\emptyset \cap C) = 0 = P(B \cap C)$ for any $C \in \mathcal{A}$ and $\emptyset \in \mathcal{C}$. Finally, if we have $A \cap B$ where $A \in \mathcal{C}$ and $B \in \mathcal{D}$ then $P(A \cap B \cap C) = 0 = P(\emptyset \cap C)$ for any $C \in \mathcal{A}$. Hence, $\sigma(C \cup D) \subset \mathcal{E}$ by the monotone class theorem (Fremlin, 2010)[136B].

(iii) Now, let us consider a $U \in \mathcal{B}$ and $V \in \sigma(\mathcal{C} \cup \mathcal{D})$. Due to steps (i) and (ii) we can select a $V' \in \mathcal{C}$ such that $P(V \cap C) = P(V' \cap C)$ for all $C \in \mathcal{A}$. Then with this V' we have that $|P(U)P(V) - P(U \cap V)| = |P(U)P(V') - P(U \cap V')| \leq \varphi P(V') = \varphi P(V)$. ■

Another important result in this context concerns the φ-mixing property of a process that starts at a random time. This result is needed to be able to use Hoeffding bounds for batches of observations which occur in our algorithm. To state this result concisely we first define the following two families of events

$$\mathcal{U} := \left\{\{\tau_n < \infty\} \cap (X_{\tau_n,i}, \ldots, X_{\tau_n+u-1,i})^{-1}[B] : B \in \mathcal{B}_{\mathcal{X}}^{(u)}\right\},$$
$$\mathcal{V} := \left\{\{\tau_n < \infty\} \cap (X_{\tau_n+u+l-1,i}, \ldots, X_{\tau_n+u+l+v-2,i})^{-1}[B] : B \in \mathcal{B}_{\mathcal{X}}^{(u)}\right\},$$

where the notation $(X_{\tau_n,i}, \ldots, X_{\tau_n+u-1,i})^{-1}[B]$ denotes the set

$$\{\omega : \omega \in \Omega, (X_{\tau_n(\omega),i}(\omega), \ldots, X_{\tau_n(\omega)+u-1,i}(\omega)) \in B\}$$

and $\mathcal{X}$ is the space in which the random variables $X_{t,i}$ attain values in.

Lemma 14 *In the jointly-stationary φ-mixing bandit problem formulated in Section 3, consider an arm $i \leq k$ and an increasing sequence of starting times $\langle \tau_n \rangle_{n \in \mathbb{N}_+}$ of batches in which arm i is played which are almost surely finite. The following holds for all $n \in \mathbb{N}_+$ and $1 \leq l \leq 2^{n-1} - 1$*

$$\sup\{|P(V) - P(U \cap V)/P(U)| : U \in \mathcal{U}, V \in \mathcal{V}, P(U) > 0, \\ u, v \in \mathbb{N}_+, u + v + l - 1 \leq 2^{n-1}\} \leq 2\varphi_l. \qquad (15)$$

Proof For any U, V as defined in (15) there exist sequences $\langle U_t \rangle_{t \in \mathbb{N}_+}$, and $\langle V_t \rangle_{t \in \mathbb{N}_+}$, where

$$U_t \in \sigma\left(\{(X_{t,i}, \ldots, X_{t+u-1,i})^{-1}[B] \cap \{\tau_n < \infty\} : B \in \mathcal{B}_{\mathcal{X}}^{(u)}\}\right) \text{ and}$$
$$V_t \in \sigma\left(\{(X_{t+u+l-1,i}, \ldots, X_{t+u+l+v-2,i})^{-1}[B] \cap \{\tau_n < \infty\} : B \in \mathcal{B}_{\mathcal{X}}^{(v)}\}\right)$$

such that $U = \bigcup_{t \in \mathbb{N}_+} \{\tau_n = t\} \cap U_t$ and $V = \bigcup_{t \in \mathbb{N}_+} \{\tau_n = t\} \cap V_t$. Observe also that due to stationarity $P(V_t) = P(V_1)$ for all $t \geq 1$. Since, $U_t \cap \{\tau_n = t\} \in \mathcal{G}_t$ the φ-mixing property gives us

$$|P(\{\tau_n = t\} \cap U_t \cap V_t) - P(\{\tau_n = t\} \cap U_t)P(V_t)| \leq \varphi_l P(\{\tau_n = t\} \cap U_t)$$

and since $\tau_n < \infty$ almost surely for each n we have

$$|P(V) - P(V_1)| \leq \sum_{t=1}^{\infty} |P(\{\tau_n = t\} \cap V_t) - P(\tau_n = t)P(V_t)| \leq \varphi_l.$$

Combining these gives the result as

$$\begin{aligned}
&|P(U)P(V) - P(U \cap V)| \\
&\quad = |P(U)P(V) - \sum_{t=1}^{\infty} P(\{\tau_n = t\} \cap U_t \cap V_t)| \\
&\quad \leq |P(U)P(V) - \sum_{t=1}^{\infty} P(\{\tau_n = t\} \cap U_t)P(V_t)| + \sum_{t=1}^{\infty} \varphi_l P(\{\tau_n = t\} \cap U_t) \\
&\quad \leq 2\varphi_l P(U) + |P(U)P(V_1) - \sum_{t=1}^{\infty} P(\{\tau_n = t\} \cap U_t)P(V_1)| = 2\varphi_l P(U).
\end{aligned}$$

■

A.2. Proof of Proposition 5

Let $(\Omega, \mathcal{A}, P)$ be some probability space with k mutually independent processes defined on it. If each of these processes is ψ-mixing then the joint process is also ψ-mixing and for all $i \in \mathbb{N}$, $1 + \tilde{\psi}_i \leq (1 + \psi_i)^k$, where the $\tilde{\psi}_i$ are the mixing coefficients of the joint process and the ψ_i are upper bounds on the mixing coefficients of the individual processes.

Proof Fix some $n, a, u, v > 0$, $a \geq u + n$ and let us denote the individual processes with $\langle X_{n,i} \rangle_{n \in \mathbb{N}_+}$, $i \leq k$. Furthermore, let $A = \{1, \ldots, u\}, B = \{a, \ldots, a + v - 1\}$ and $\mathcal{G} = \sigma(X_{A,1}, \ldots, X_{A,k}), \mathcal{H} = \sigma(X_{B,1}, \ldots, X_{B,k})$. The proof structure is the following: in step (i) we show for "simple" events that the ψ-mixing property carries over to the joint process. In step (ii), we construct a new probability space in which to each of these "simple" events a product of events is associated. The approach is useful since more complex events can be easily approximated by products. In step (iii) we make use of this approximation and we relate arbitrary events to unions of products for each of which the ψ-mixing property derived in (i) applies.

(i) Consider a set $U \in \mathcal{G}$ of the form $U = \bigcap_{i\leq k} U_i$ where $U_i \in \sigma(X_{A,i})$ for all $i \leq k$ and a set $V \in \mathcal{H}$ of the form $V = \bigcap_{i\leq k} V_i$, $V_i \in \sigma(X_{B,i})$, for all $i \leq k$. The mutual independence of the processes implies that

$$
\begin{aligned}
|P(U)P(V) - P(U\cap V)| &= \Big|\prod_{i=1}^{k} P(U_i)P(V_i) - \prod_{i=1}^{k} P(U_i\cap V_i)\Big| \\
&\leq P(U_1)P(V_1)\,\Big|\prod_{i=2}^{k} P(U_i)P(V_i) - \prod_{i=2}^{k} P(U_i\cap V_i)\Big| \\
&\quad + |P(U_1)P(V_1) - P(U_1\cap V_1)|\prod_{i=2}^{k} P(U_i\cap V_i) \\
&\leq \Big(\prod_{i=1}^{2} P(U_i)P(V_i)\Big)\,\Big|\prod_{i=3}^{k} P(U_i)P(V_i) - \prod_{i=3}^{k} P(U_i\cap V_i)\Big| \\
&\quad + P(U_1)P(V_1)\,|P(U_2)P(V_2) - P(U_2\cap V_2)|\prod_{i=3}^{k} P(U_i\cap V_i) \\
&\quad + \psi_n\, P(U_1)P(V_1)\prod_{i=2}^{k} P(U_i\cap V_i) \\
&\leq \psi_n \sum_{i=1}^{k}\Big(\prod_{j=1}^{i} P(U_j)P(V_j)\Big)\Big(\prod_{j=i+1}^{k} P(U_j\cap V_j)\Big) \\
&\leq \psi_n\, P(U)P(V)\sum_{i=1}^{k}\prod_{j=i+1}^{k}\frac{P(U_j\cap V_j)}{P(U_j)P(V_j)} \\
&\leq \psi_n\, P(U)P(V)\sum_{i=0}^{k-1}(1+\psi_n)^{k-i-1} = ((1+\psi_n)^k - 1)\,P(U)P(V).
\end{aligned}
$$

(ii) Next, we demonstrate the ψ-mixing property for $U \in \mathcal{G}$ and $V \in \mathcal{H}$. We use a product measure approach. To make use of this let the index set $C = A \cup B$ and consider the independent σ-algebras $\sigma(X_{C,1}), \ldots, \sigma(X_{C,k})$. Let P_i be the restriction of P to $\sigma(X_{C,i})$ for all $i \leq k$ and define the product space $(\Omega^k, \widehat{\otimes}_{i\leq k}\sigma(X_{C,i}), \mu)$, where μ is the product measure of $P_1, \ldots, P_k$ and $\widehat{\otimes}_{i\leq k}\sigma(X_{C,i})$ denotes the product σ-algebra of $\sigma(X_{C,1}), \ldots, \sigma(X_{C,k})$. The map $\phi : \Omega \to \Omega^k$, $\phi(\omega)(i) = \omega$ for all $i \leq k$, is inverse-measure preserving due to Fremlin (2010)[272J,254Xc]. In particular, $P(\bigcap_{i\leq k} U_i) = P\phi^{-1}[U_1 \times \ldots \times U_k] = \mu(U_1 \times \ldots \times U_k)$ for $U_i \in \sigma(X_{A,i})$. The important property is the following: if $U \in \mathcal{G}$ then there exists an $F \in \widehat{\otimes}_{i\leq k}\sigma(X_{A,i}) \subseteq \widehat{\otimes}_{i\leq k}\sigma(X_{C,i})$ such that $U = \phi^{-1}[F]$. To see this consider the set

$$\mathcal{S} := \{\phi^{-1}[F] : F \in \widehat{\otimes}_{i\leq k}\sigma(X_{A,i})\}.$$

A standard argument shows that $\mathcal{S}$ is a σ-algebra, i.e. $\emptyset \in \mathcal{S}$, if $U \in \mathcal{S}$ then $\Omega\backslash U = \phi^{-1}[\Omega^k\backslash U] \in \mathcal{S}$ and if $\langle U_n\rangle_{n\in\mathbb{N}}$ a sequence in $\mathcal{S}$ then $\bigcup_{n\in\mathbb{N}} U_n = \phi^{-1}[\bigcup_{n\in\mathbb{N}} F_n] \in \mathcal{S}$ for suitable sets F_n. Furthermore, $\mathcal{S} \supseteq \{\bigcap_{i\leq k} U_i : U_i \in \sigma(X_{A,i}), i \leq k\}$ and the latter set is closed under intersection.

Hence, the monotone class theorem tells us that $\mathcal{G} = \sigma\{\bigcap_{i\leq k} U_i : U_i \in \sigma(X_{A,i}), i \leq k\} \subset \mathcal{S}$ and the result follows. Similarly, we can link $V \in \mathcal{G}$ to $\widehat{\otimes}_{i\leq k}\sigma(X_{B,i})$.

(iii) We'd like to demonstrate that the ψ-mixing property carries over to arbitrary elements $U \in \widehat{\otimes}_{i\leq k}\sigma(X_{A,i})$ and $V \in \widehat{\otimes}_{i\leq k}\sigma(X_{B,i})$. First, observe that if $U = U_1 \times \ldots \times U_k$ and $V = V_1 \times \ldots \times V_k$, $U_i \in \sigma(X_{A,i}), V_i \in \sigma(X_{B,i})$, $i \leq k$, then using $(U_1 \times \ldots \times U_k) \cap (V_1 \times \ldots \times V_k) = (U_1 \cap V_1) \times \ldots \times (U_k \cap V_k)$ with the inverse-measure preserving property of ϕ and (i) it follows that

$$
\begin{aligned}
&|\mu(U_1 \times \ldots \times U_k)\mu(V_1 \times \ldots \times V_k) - \mu(U_1 \times \ldots \times U_k \cap V_1 \times \ldots \times V_k)| \\
&= \big|P\big(\textstyle\bigcap_{i\leq k} U_i\big)P\big(\bigcap_{i\leq k} V_i\big) - P\big(\bigcap_{i\leq k} U_i \cap \bigcap_{i\leq k} V_i\big)\big| \\
&\leq ((1+\psi_n)^k - 1)\,P\big(\textstyle\bigcap_{i\leq k} U_i\big)P\big(\bigcap_{i\leq k} V_i\big) \\
&= ((1+\psi_n)^k - 1)\,\mu(U_1 \times \ldots \times U_k)\mu(V_1 \times \ldots \times V_k).
\end{aligned}
$$

The advantage of the product approach is that we can approximate the sets U and V with cylinders of the form $U_1 \times \ldots \times U_k$ and this allows us to carry the ψ-mixing property to $\mathcal{G}$ and $\mathcal{H}$. As follows from Fremlin (2010)[Thm. 251Ie, pp. 200, 251W] for every $\epsilon \in (0, 1]$ there exist sequences $U_{1,1}, \ldots, U_{m_1,1}, \ldots, U_{1,k}, \ldots, U_{m_1,k}$ and $V_{1,1}, \ldots, V_{m_2,1}, \ldots, V_{1,k}, \ldots, V_{m_2,k}$, with $U_{i,j}, V_{i',j} \in \sigma(X_{C,j})$ for all $i \leq m_1, i' \leq m_2, j \leq k$, with the following four properties.

$$\mu\Big(\bigcup_{i\leq m_1} U_{i,1} \times \ldots \times U_{i,k}\Big) \geq \sum_{i\leq m_1} \mu(U_{i,1} \times \ldots \times U_{i,k}) - \epsilon, \tag{16}$$

$$\mu\Big(\bigcup_{i\leq m_2} V_{i,1} \times \ldots \times V_{i,k}\Big) \geq \sum_{i\leq m_2} \mu(V_{i,1} \times \ldots \times V_{i,k}) - \epsilon/m_1, \tag{17}$$

$$\mu\Big(U \triangle \bigcup_{i\leq m_1} U_{i,1} \times \ldots \times U_{i,k}\Big) \leq \epsilon, \text{ and} \tag{18}$$

$$\mu\Big(V \triangle \bigcup_{i\leq m_2} V_{i,1} \times \ldots \times V_{i,k}\Big) \leq \epsilon/m_1, \tag{19}$$

where $\triangle$ denotes the symmetric difference between sets.

From (16) and (17) we obtain $\sum_{i\leq m_1} \mu(U_{i,1} \times \ldots \times U_{i,k}) \leq 1 + \epsilon$ and $\sum_{i\leq m_2} \mu(V_{i,1} \times \ldots \times V_{i,k}) \leq 1 + \epsilon/m_1$ respectively. Moreover, we have

$$\big|\mu(U) - \mu\big(\bigcup_{i\leq m_1} U_{i,1} \times \ldots \times U_{i,k}\big)\big| \leq \mu\big(U \triangle \bigcup_{i\leq m_1} U_{i,1} \times \ldots \times U_{i,k}\big) \leq \epsilon \tag{20}$$

where the second inequality follows from (18). Similarly, from (19) we obtain

$$\big|\mu(V) - \mu\big(\bigcup_{i\leq m_2} V_{i,1} \times \ldots \times V_{i,k}\big)\big| \leq \epsilon/m_1. \tag{21}$$

Furthermore, to obtain a similar result for $U \cap V$,below, we rely on the following elementary set manipulation. Consider sets A_1, A_2, A_3, A_4 then

$$
\begin{aligned}
&(A_1 \cap A_2)\triangle(A_3 \cap A_4) = ((A_1 \cap A_2)\backslash(A_3 \cap A_4)) \cup ((A_3 \cap A_4)\backslash(A_1 \cap A_2)) \\
&= ((A_1 \cap A_2)\backslash A_3) \cup ((A_1 \cap A_2)\backslash A_4) \cup ((A_3 \cap A_4)\backslash A_1) \cup ((A_3 \cap A_4)\backslash A_2) \\
&\subseteq (A_1\backslash A_3) \cup (A_2\backslash A_4) \cup (A_3\backslash A_1) \cup (A_4\backslash A_2) \\
&= (A_1\triangle A_3) \cup (A_2\triangle A_4).
\end{aligned} \tag{22}
$$

We have,

$$
\begin{aligned}
&\big|\mu(U\cap V)-\mu\big(\big(\bigcup_{i\le m_1} U_{i,1}\times\ldots\times U_{i,k}\big)\cap\big(\bigcup_{i\le m_2} V_{i,1}\times\ldots\times V_{i,k}\big)\big)\big| \\
&\le\big|\mu\big((U\cap V)\triangle\big(\big(\bigcup_{i\le m_1} U_{i,1}\times\ldots\times U_{i,k}\big)\cap\big(\bigcup_{i\le m_2} V_{i,1}\times\ldots\times V_{i,k}\big)\big)\big)\big| \\
&\le\mu\big(U\triangle\big(\bigcup_{i\le m_1} U_{i,1}\times\ldots\times U_{i,k}\big)\big)+\mu\big(V\triangle\big(\bigcup_{i\le m_2} V_{i,1}\times\ldots\times V_{i,k}\big)\big) &(23)\\
&\le\epsilon+\epsilon/m_1 &(24)\\
&\le 2\epsilon &(25)
\end{aligned}
$$

where, (23) is due to (22) and (24) follows from (18) and (19). Now, applying (20), (21) and (25) we obtain,

$$
\begin{aligned}
&|\mu(U)\mu(V)-\mu(U\cap V)| \\
&\le|\mu(U)\mu(V)-\mu\big(\bigcup_{i\le m_1}\bigcup_{j\le m_2}(U_{i,1}\times\ldots\times U_{i,k})\cap(V_{j,1}\times\ldots\times V_{j,k})\big)|+2\epsilon \\
&\le 4\epsilon+\big|\mu\big(\bigcup_{i\le m_1} U_{i,1}\times\ldots\times U_{i,k}\big)\mu\big(\bigcup_{i\le m_2} V_{i,1}\times\ldots\times V_{i,k}\big) \\
&\qquad-\mu\big(\bigcup_{i\le m_1}\bigcup_{j\le m_2}(U_{i,1}\times\ldots\times U_{i,k})\cap(V_{j,1}\times\ldots\times V_{j,k})\big)\big|. &(26)
\end{aligned}
$$

Furthermore,

$$
\begin{aligned}
&\big|\sum_{i\le m_1}\sum_{j\le m_2}\mu(U_{i,1}\times\ldots\times U_{i,k})\mu(V_{j,1}\times\ldots\times V_{j,k}) \\
&\qquad\qquad-\mu\big(\bigcup_{i\le m_1} U_{i,1}\times\ldots\times U_{i,k}\big)\mu\big(\bigcup_{i\le m_2} V_{i,1}\times\ldots\times V_{i,k}\big)\big| \\
&\le\Big(\mu\big(\bigcup_{i\le m_1} U_{i,1}\times\ldots\times U_{i,k}\big)+\epsilon\Big)\sum_{j\le m_2}\mu(V_{j,1}\times\ldots\times V_{j,k}) \\
&\qquad\qquad-\mu\big(\bigcup_{i\le m_1} U_{i,1}\times\ldots\times U_{i,k}\big)\mu\big(\bigcup_{i\le m_2} V_{i,1}\times\ldots\times V_{i,k}\big) \\
&\le 2\epsilon+\epsilon^2 \\
&\le 3\epsilon. &(27)
\end{aligned}
$$

Let $U_1,\ldots,U_m$ be such that $\mu(\bigcup_{i\le m}U_i)\ge\sum_{i\le m}\mu(U_i)-\epsilon$. Then $\sum_{i\le m}\mu\big(\bigcup_{j<i}U_j\cap U_i\big)\le\epsilon$. This is because

$$
\begin{aligned}
\mu\big(\bigcup_{i\le m}U_i\big)&=\mu\big(\bigcup_{i\le m}U_i\backslash\big(\bigcup_{j<i}U_j\cap U_i\big)\big)\\
&=\sum_{i\le m}\mu\big(U_i\backslash\big(\bigcup_{j<i}U_j\cap U_i\big)\big)\\
&=\sum_{i\le m}\mu(U_i)-\sum_{i\le m}\mu\big(\bigcup_{j<i}U_j\cap U_i\big)\big).
\end{aligned}
$$

Now, for any measurable set V we have by the same argument that

$$
\begin{aligned}
\mu\big(\bigcup_{i\le m}U_i\cap V\big)&=\mu\big(\bigcup_{i\le m}U_i\cap V\backslash\big(\bigcup_{j<i}U_j\cap U_i\big)\big)\\
&=\sum_{i\le m}\mu\big(U_i\cap V\backslash\big(\bigcup_{j<i}U_j\cap U_i\big)\big)\\
&=\sum_{i\le m}\mu(U_i\cap V)-\sum_{i\le m}\mu\big(\bigcup_{j<i}U_j\cap U_i\big)\big)\\
&\ge\sum_{i\le m}\mu(U_i\cap V)-\epsilon.
\end{aligned}
$$

Hence, using (16) and (17), we can obtain

$$
\begin{aligned}
&\big|\sum_{i\le m_1}\sum_{j\le m_2}\mu\big((U_{i,1}\times\ldots\times U_{i,k})\cap(V_{j,1}\times\ldots\times V_{j,k})\big)\\
&\quad-\mu\big(\bigcup_{i\le m_1}\bigcup_{j\le m_2}(U_{i,1}\times\ldots\times U_{i,k})\cap(V_{j,1}\times\ldots\times V_{j,k})\big)\big|\\
&\le\sum_{i\le m_1}\sum_{j\le m_2}\mu\big((U_{i,1}\times\ldots\times U_{i,k})\cap(V_{j,1}\times\ldots\times V_{j,k})\big)\\
&\quad-\sum_{i\le m_1}\mu\big((U_{i,1}\times\ldots\times U_{i,k})\cap\big(\bigcup_{j\le m_2}(V_{j,1}\times\ldots\times V_{j,k})\big)+\epsilon\\
&\le\epsilon+\sum_{i\le m_1}\epsilon/m_1\\
&=2\epsilon. &(28)
\end{aligned}
$$

We obtain an bound on $|\mu(U)\mu(V)-\mu(U\cap V)|$ by using (26), (27) and (28) together with the result from Step (i), i.e.,

$$
\begin{aligned}
&|\mu(U)\mu(V)-\mu(U\cap V)|\\
&\qquad\le 9\epsilon+\sum_{i\le m_1}\sum_{j\le m_2}|\mu(U_{i,1}\times\ldots\times U_{i,k})\mu(V_{j,1}\times\ldots\times V_{j,k})\\
&\qquad\qquad-\mu\big((U_{i,1}\times\ldots\times U_{i,k})\cap(V_{j,1}\times\ldots\times V_{j,k})\big)|\\
&\qquad\le 9\epsilon+\big((1+\psi_n)^k-1\big)\sum_{i\le m_1}\mu(U_{i,1}\times\ldots\times U_{i,k})\sum_{j\le m_2}\mu(V_{j,1}\times\ldots\times V_{j,k}) &(29)\\
&\qquad\le\big((1+\psi_n)^k-1\big)\,\mu(U)\mu(V)+9\epsilon+5\epsilon\big((1+\psi_n)^k-1\big),
\end{aligned}
$$

where the last inequality follows from the same arguments as used in inequalities (26) and (27). Since ϵ is arbitrary we derive the upper bound for arbitrary elements $U\in\widehat{\otimes}_{i\le k}\sigma(X_{A,i})$ and $V\in\widehat{\otimes}_{i\le k}\sigma(X_{B,i})$.

Note that this last step is the only part of the proof where ψ-mixing is necessary. In particular, we could not derive (29) from the line preceding it by only assuming φ-mixing.

To conclude, if $U \in \mathcal{G}, V \in \mathcal{H}$ then we know from step (ii) that there are elements $E \in \widehat{\otimes}_{i\leq k}\sigma(X_{A,i}), F \in \widehat{\otimes}_{i\leq k}\sigma(X_{B,i})$ such that

$$\begin{aligned}|P(U\cap V) - P(U)P(V)| &= |P\phi^{-1}[E\cap F] - P\phi^{-1}[E]P\phi^{-1}[F]| \\ &= |\mu(E\cap F) - \mu(E)\mu(F)| \\ &\leq ((1+\psi_n)^k - 1)P(U)P(V).\end{aligned}$$

(iv) The joint process is ψ-mixing since $\lim_{n\to\infty}(1-\psi_n)^k - 1 = 0$. ■

A.3. Proof of Proposition 8

In what follows, we denote the probability space by $(\Omega, \mathcal{A}, P)$ and let $\langle \mathcal{E}_t\rangle_{t\geq 1}$ be the filtration $\mathcal{E}_t = \sigma\{\sigma(X_1,\ldots,X_t)\cup\mathcal{N}\}$ where $\mathcal{N}$ is the family of sets of measure zero. The random times are $\tau_i : \Omega \to \mathbb{N}_+$ and they are $\mathcal{A} - \mathcal{P}(\mathbb{N}_+)$ measurable, where $\mathcal{P}(\mathbb{N}_+)$ denotes the power set of $\mathbb{N}_+ = \{1, 2\ldots\}$. Furthermore, let the random variables X_i attain values in $\mathcal{X} \subset \mathbb{R}$ and assume that they are $\mathcal{A} - \mathcal{B}_\mathcal{X}$ measurable, where $\mathcal{B}_\mathcal{X}$ denotes the Borel σ-algebra on $\mathcal{X}$. We define the random times τ_i inductively together with a filtration that tracks the observed information. Let $\mathcal{H}_0 = \{\emptyset, \Omega\}$, let τ_1 be a $\mathcal{H}_0$-measurable random time and let $\mathcal{H}_1 = \sigma(X_{\tau_1})$. Then, for given $\tau_1,\ldots,\tau_i$ and $\mathcal{H}_1,\ldots,\mathcal{H}_i$ let τ_{i+1} be some $\mathcal{H}_i$-measurable random variable such that $\tau_{i+1} > \tau_i$ almost surely and define $\mathcal{H}_{i+1} = \sigma(X_{\tau_1},\ldots,X_{\tau_{i+1}})$. We can observe that $\sigma(X_{\tau_1},\ldots,X_{\tau_i}) = \sigma(X_{\tau_1},\ldots,X_{\tau_t},\tau_1,\ldots,\tau_t)$. To see this, note that τ_1 is $\sigma(X_{\tau_1},\ldots,X_{\tau_t})$-measurable, hence $\sigma(X_{\tau_1},\ldots,X_{\tau_t},\tau_1,\ldots,\tau_t) = \sigma(X_{\tau_1},\ldots,X_{\tau_t},\tau_2,\ldots,\tau_t)$. Moreover, since τ_2 is $\sigma(X_{\tau_1},\tau_1) = \sigma(X_{\tau_1})$-measurable, we have that $\sigma(X_{\tau_1},\ldots,X_{\tau_t},\tau_2,\ldots,\tau_t) = \sigma(X_{\tau_1},\ldots,X_{\tau_t},\tau_3,\ldots,\tau_t)$, and the observation can simply be verified by induction. Recall the statement of Proposition 8 as follows.

Assume that $\langle X_t\rangle_{t\in\mathbb{N}_+}$ is a stationary φ-mixing process with mixing coefficients $\langle\varphi_i\rangle_{i\in\mathbb{N}_+}$ such that $EX_t = \mu,\ t\in\mathbb{N}_+$, and $\sup_{t\in\mathbb{N}_+}|X_t| \leq c$ for some $c\in[0,\infty)$. Furthermore, let $\tau_1,\tau_2,\ldots$ be a sequence of random times such that $\tau_i + \ell \leq \tau_{i+1}$ a.s. for some $\ell \geq 1$ and all $i\in\mathbb{N}_+$, and all τ_{i+1} are $\sigma(X_{\tau_1},\ldots,X_{\tau_i})$- measurable with $\tau_1 \in \mathbb{N}_+$ being a fixed time. Then for any $n\in\mathbb{N}_+$

$$\Big|\frac{1}{n}\sum_{i=1}^n EX_{\tau_i} - \mu\Big| \leq 2c\varphi_\ell.$$

Proof The following technical result is important in this context.

Lemma A.3 *For all $i,t \geq 1$ and $A \in \mathcal{H}_{i-1}$ we have that $A\cap\{\tau_i = t\} \in \mathcal{E}_t$. In particular $\{\tau_i = t\}\in\mathcal{E}_t$.*

Proof For $i = 1$ the result holds trivially since $A\cap\{\tau_i = t\}$ is either Ω or $\emptyset$ and since $\mathcal{E}_t$ is a σ-algebra it contains Ω and $\emptyset$. For any other $i \geq 2$ observe that if the claim holds for all $1\leq j\leq i-1$ then since τ_i is $\mathcal{H}_{i-1}$-measurable and because $\tau_i > \tau_{i-1}$ almost surely implies that $U := \{\tau_i = t\}\cap\{\tau_{i-1}\geq t\} \in \mathcal{N}\subset\mathcal{E}_t$ and we have

$$\{\tau_i = t\} = U \cup \bigcup_{s=1}^{t-1}\{\tau_i = t\}\cap\{\tau_{i-1} = s\} \in \mathcal{E}_t.$$

We can also observe that $\{B\cap\{\tau_i = t\} : B\in\sigma(X_{\tau_i})\}\subset\mathcal{E}_t$ because

$$\{B\cap\{\tau_i = t\} : B\in\sigma(X_{\tau_i})\} = \{X_t^{-1}[B']\cap\{\tau_i = t\} : B'\in\mathcal{B}_\mathcal{X}\}$$

and both $X_t^{-1}[B']$ and $\{\tau_i = t\}$ lie in $\mathcal{E}_t$.

Also, for any $j\in\{1,\ldots,i-1\}$ we have $\{B\cap\{\tau_i = t\} : B\in\sigma(X_{\tau_j})\}\subset\mathcal{E}_t$: let $U = \{\tau_i = t\}\cap\{\tau_j \geq t+1-(i-j)\}$ then $U\in\mathcal{N}$ and

$$\{B\cap\{\tau_i = t\} : B\in\sigma(X_{\tau_j})\} = U\cup\bigcup_{s=1}^{t-(i-j)}\{X_s^{-1}[B']\cap\{\tau_i = t\}\cap\{\tau_j = s\} : B'\in\mathcal{B}_\mathcal{X}\}$$

and $X_s^{-1}[B']\cap\{\tau_j = s\}\in\mathcal{E}_s\subset\mathcal{E}_t$. We have shown that

$$\Big\{\{\tau_i = t\}\cap\bigcap_{j=1}^{i-1}B_j : B_j\in\sigma(X_{\tau_j})\Big\}\subset\mathcal{E}_t.$$

This implies directly that

$$\sigma\Big\{\{\tau_i = t\}\cap\bigcap_{j=1}^{i-1}B_j : B_j\in\sigma(X_{\tau_j})\Big\}\subset\mathcal{E}_t.$$

By induction one can verify that $\{A\cap\{\tau_i = t\} : A\in\sigma(X_{\tau_1},\ldots,X_{\tau_{i-1}})\}$ is included in the left side. ■

We also need the following observation: Let $s,t\in\mathbb{N}_+$, $s < t$ then for any $B\in\mathcal{E}_s$ we have $|\int_B(E(X_t|\mathcal{E}_s) - E(X_t))| \leq 2\varphi_{t-s}cP(B)$. This follows from Lemma A.2 by remembering that $\mathcal{E}_s = \sigma(\sigma(X_1,\ldots,X_s)\cup\mathcal{N})$, i.e. the φ-mixing property implies this upper bound for $\sigma(X_1,\ldots,X_s)$ instead of $\mathcal{E}_s$ and Lemma A.2 allows us to extend this property to $\sigma(\sigma(X_1,\ldots,X_s)\cup\mathcal{N})$ by using in Lemma A.2 the following σ-algebras: $\mathcal{B} = \sigma(X_t), \mathcal{C} = \sigma(X_1,\ldots,X_s)$ and $\mathcal{D} = \mathcal{N}$.

Now, coming to the main proof consider first τ_1. Note that τ_1 is independent of any X_i since for $U\in\sigma(X_i), V\in\sigma(\tau_1) = \{\emptyset,\Omega\}$ we have either $V = \Omega$ and $P(U\cap V) = P(U) = P(U)P(V)$ or $V = \emptyset$ and $P(U\cap V) = 0 = P(U)P(V)$. Independence and stationarity of the process $\langle X_t\rangle_{t\in\mathbb{N}_+}$ give us for any $t\in\mathbb{N}_+$ that

$$E\left(X_t\times\chi\{\tau_1 = t\}\right) = P(\tau_1 = t)E(X_t) = P(\tau_1 = t)E(X_1).$$

Now, since the X_t are bounded and τ_1 attains a value in $\mathbb{N}_+$ we know that

$$\sum_{t=1}^\infty E\left(|X_t|\times\chi\{\tau_1 = t\}\right) = \sum_{t=1}^\infty P(\tau_1 = t)E|X_1|$$

is finite and B. Levi's Theorem (Fremlin, 2010)[123A] tells us that $\sum_{t=1}^\infty|X_t|\times\chi\{\tau_1 = t\}$ is integrable. Also $|\sum_{t=1}^{t'}X_t\times\chi\{\tau_1 = t\}|$ is upper bounded by the integrable function $\sum_{t=1}^\infty|X_t|\times\chi\{\tau_1 = t\}$ for all $t'\in\mathbb{N}_+$ and Lebesgue's Dominated Convergence Theorem (Fremlin, 2010)[123C] gives us

$$E(X_{\tau_1}) = E\big(\sum_{t=1}^\infty X_t\times\chi\{\tau_1 = t\}\big) = \sum_{t=1}^\infty E(X_t\times\chi\{\tau_1 = t\}) = \sum_{t=1}^\infty P(\tau_1 = t)E(X_t) = E(X_1).$$

We perform induction over τ_i, $i \geq 2$. The set $\{\tau_i = t\}$ is an element of $\mathcal{H}_{i-1}$ and for any $s < t$, $s, t \in \mathbb{N}_+$ we have that $B = \{\tau_i = t\} \cap \{\tau_{i-1} = s\} \in \mathcal{H}_{i-1}$. Observe that Lemma A.3 tells us that $B \in \mathcal{E}_s$ and, hence, that

$$\int_B E(X_{\tau_i}|\mathcal{H}_{i-1}) = \int_B X_t = \int_B E(X_t|\mathcal{E}_s).$$

Due to stationarity this implies that

$$\Big|\int_B (E(X_{\tau_i}|\mathcal{H}_{i-1}) - E(X_t))\Big| = \Big|\int_B (E(X_t|\mathcal{E}_s) - E(X_1))\Big| \leq 2c\varphi_{t-s}P(B). \tag{30}$$

Because τ_i is $\mathcal{H}_{i-1}$ measurable we can now write

$$\begin{aligned}
E(X_{\tau_i}) &= EE(X_{\tau_i}|\mathcal{H}_{i-1}) \\
&= EE(\sum_{t=1}^{\infty} X_t \times \chi\{\tau_i = t\}|\mathcal{H}_{i-1}) && (31)\\
&= \sum_{t=1}^{\infty} EE(X_t \times \chi\{\tau_i = t\}|\mathcal{H}_{i-1}) && (32)\\
&= \sum_{t=1}^{\infty} E(E(X_t|\mathcal{H}_{i-1}) \times \chi\{\tau_i = t\}) && (33)\\
&= \sum_{t=1}^{\infty} E(E(X_t|\mathcal{H}_{i-1}) \times \chi\Big(\{\tau_i = t\} \cap \bigcup_{s=1}^{t-\ell}\{\tau_{i-1} = s\}\Big)) \\
&= \sum_{t=1}^{\infty}\sum_{s=1}^{t-\ell} E(E(X_t|\mathcal{H}_{i-1}) \times \chi\Big(\{\tau_i = t\} \cap \{\tau_{i-1} = s\}\Big), && (34)
\end{aligned}$$

where, with the same argument as above, using B. Levi's Theorem and Lebesgue's Dominated Convergence Theorem for the expectation operator and for the conditional expectation operator, the infinite summation in (31) is moved outside to give (32). Moreover, (33) is due to the fact that τ_i is $\mathcal{H}_{i-1}$-measurable. Finally, we obtain,

$$\begin{aligned}
&|E(X_{\tau_i}) - E(X_1)| && (35)\\
&= \left|\sum_{t=1}^{\infty}\sum_{s=1}^{t-\ell}(E(E(X_t|\mathcal{H}_{i-1}) \times \chi\left(\{\tau_i = t\} \cap \{\tau_{i-1} = s\}\right)) - P(\tau_i = t, \tau_{i-1} = s)E(X_1))\right| && (36)\\
&\leq \sum_{t=1}^{\infty}\sum_{s=1}^{t-\ell}\left|(E(E(X_t|\mathcal{H}_{i-1}) \times \chi\left(\{\tau_i = t\} \cap \{\tau_{i-1} = s\}\right)) - P(\tau_i = t, \tau_{i-1} = s)E(X_1))\right| \\
&\leq 2c\sum_{t=1}^{\infty}\sum_{s=1}^{t-\ell}\varphi_{t-s}P(\tau_i = t, \tau_{i-1} = s) && (37)\\
&\leq 2c\varphi_\ell\sum_{t=1}^{\infty}\sum_{s=1}^{t-\ell}P(\tau_i = t, \tau_{i-1} = s) && (38)\\
&= 2c\varphi_\ell,
\end{aligned}$$

where, (36) follows from (34), (37) follows from (30) with $B := \{\tau_i = t\} \cap \{\tau_{i-1} = s\} \in \mathcal{H}_{i-1}$, and (38) is due to the fact that $\langle\varphi_n\rangle_{n\in\mathbb{N}_+}$ is a decreasing sequence, see, e.g. (Bradley, 2007)[vol. 1 pp. 69], and that $t - s \geq \ell$. This proves the statement. ■

A.4. Details of Example 1

We give in this section the details of the construction in Example 1. This example demonstrates that the sequence $\langle X_{\tau_n}\rangle_{n\in\mathbb{N}_+}$ of samples of one of the arms of a two armed bandit problem does not need to be φ-mixing even though the original process is φ-mixing. The argument uses standard results from Markov chains as they can be found in (Levin et al., 2008) and a well known perturbation result for Markov chains. We provide the details of the argument for completeness.

Assume we have a two arm bandit problem where the pay-off for arm two is zero at all time and the pay-off distribution of arm one is described by a Markov chain with two states, transition probabilities $p_{11} = p_{22} = 1 - \epsilon, p_{12} = p_{21} = \epsilon$, for some $\epsilon \in (0, 1)$, and probability $1/2$ to be in state 1 at time $t = 0$. The player gains a pay-off of 1 if the Markov chain is in state 1 and a pay-off of 0 if the Markov chain is in state 2. The Markov chain is irreducible and aperiodic. Furthermore, the Markov chain induces a stationary pay-off distribution. This implies that we are dealing with a jointly stationary φ-mixing process with mixing coefficients φ_k being upper bounded by $\varphi_k \leq (1 - 2\epsilon)^k$: one can derive the particular bound by considering an eigendecomposition of the transition matrix T which yields eigenvalues $\lambda_1 = 1, \lambda_2 = 1 - 2\epsilon$ and eigenvectors $u_1 = \sqrt{2}(1/2 \quad 1/2)^\top, u_2 = \sqrt{2}(1/2 \quad -1/2)^\top$, i.e. with $U = (u_1 \quad u_2)$ and Λ being the diagonal matrix with entries λ_1 and λ_2 we have $T = U\Lambda U^\top$. The stationary distribution over the states is $s = (1/2 \quad 1/2)^\top$ and for any vector $v = (v_1 \quad v_2)^\top, v_1, v_2 \geq 0, v_1 + v_2 = 1$, we have $\Lambda^k U^\top v = (1/\sqrt{2})(1 \quad (1 - 2\epsilon)^k(v_1 - v_2))^\top$. Hence, $T^k v - s = (1/2)(1 - 2\epsilon)^k(v_1 - v_2)(1 \quad -1)^\top$ and $\left\|T^k v - s\right\|_\infty \leq (1/2)(1 - 2\epsilon)^k$. The mixing coefficients can now be bounded in the following way. Let $X_{t,i}, t \geq 1, i \in \{1, 2\}$, be random variables that represent the pay-off of arm i gained at time t. Consider a particular realization where $X_{1,1} = x_1, \ldots, X_{n,1} = x_n$ and $X_{n+k,1} = x_{n+k}, \ldots, X_{n+k+m,1} = x_{n+k+m}$ for some $x_1, \ldots, x_n, x_{n+k+1}, \ldots, x_{n+k+m} \in \{0, 1\}$ then $P(X_{1,1} = x_1, \ldots, X_{n,1} = x_n, X_{n+k,1} = x_{n+k}, \ldots, X_{n+k+m,1} = x_{n+k+m}) = P(X_{1,1} = x_1, \ldots, X_{n,1} = x_n)P(X_{n+k,1} = x_{n+k}, \ldots, X_{n+k+m,1} = x_{n+m+k}|X_{n,1} = x_n)$ and

$$\begin{aligned}
&|P(X_{n+k,1} = x_{n+k}, \ldots, X_{n+k+m,1} = x_{n+m+k}|X_{n,1} = x_n) \\
&\qquad - P(X_{n+k,1} = x_{n+k}, \ldots, X_{n+k+m,1} = x_{n+m+k})| \\
&= P(X_{n+k+1,1} = x_{n+k+1}, \ldots, X_{n+k+m,1} = x_{n+m+k}|X_{n+k,1} = x_{n+k}) \\
&\qquad \times |P(X_{n+k,1} = x_{n+k}) - P(X_{n+k,1} = x_{n+k}|X_{n,1} = x_n)| \\
&\leq (1/2)(1 - 2\epsilon)^k P(X_{n+k+1,1} = x_{n+k+1}, \ldots, X_{n+k+m,1} = x_{n+m+k}|X_{n+k,1} = x_{n+k}) \\
&= (1 - 2\epsilon)^k P(X_{n+k,1} = x_{n+k}, \ldots, X_{n+k+m,1} = x_{n+m+k}). && (39)
\end{aligned}$$

since $P(X_{n+k,1} = x_{n+k}) = 1/2$. Let $A = \{1, \ldots, n\}$, $B = \{n + k, \ldots, n + k + m\}$ and consider $\sigma(X_A), \sigma(X_B)$. The events in these σ-algebras are finite unions of events of the form $\{X_{1,1} = x_1, \ldots, X_{n,1} = x_n\}$ and $\{X_{n+k,1} = x_{n_k}, \ldots, X_{n+k+m,1} = x_{n+k+m}\}$.

For any $U \in \sigma(X_A)$ we know that U consists at most of finitely many such events $U_1, \ldots, U_l$, $U_i \cap U_j = \emptyset$, for all $i, j \leq l$. Similarly for $V \in \sigma(X_B)$ we know that $V = V_1 \cup \ldots \cup V_o$, $V_i \cap V_j = \emptyset$,

for all $i, j \leq o$. The above argument which leads to the bound (39) allows us to conclude that for any $U \in \sigma(X_A)$ and $V \in \sigma(X_B)$

$$|P(U)P(V) - P(U \cap V)| = \sum_{i \leq l} \sum_{j \leq o} |P(U_i)P(V_j) - P(U_i \cap V_j)|$$
$$\leq (1-2\epsilon)^k \sum_{i \leq l} \sum_{j \leq o} P(U_i)P(V_j) \leq (1-2\epsilon)^k.$$

Hence the Markov chain is φ-mixing. The second arm has a constant reward and does not introduce any dependencies in the pay-off over time. Therefore we have a jointly stationary φ-mixing process with the mixing coefficient being equal to the mixing coefficients of the Markov chain.

Now fix some $\delta > 0$ and consider the following policy π^δ. At $t = 1$ the policy plays arm 1 receiving pay-off $X_{1,1}$. Then at any other $t \geq 2$ the arm is selected according to the following rules: if at $t-1$ arm 1 has been played and $X_{t-1,1} = X_{1,1}$ then the policy chooses at t arm 1; if at $t-1$ arm 1 has been played and $X_{t-1,1} \neq X_{1,1}$ then the policy chooses at t arm 2 and plays arm 2 for the next $k := \lceil \log(2\delta)/\log(1-2\epsilon) \rceil$ rounds before switching back to arm 1. We'd like to show that the sequence of pay-offs generated by policy π^δ at arm 1 is not φ-mixing. Let $\tau_1, \tau_2, \ldots$ be the sequence of random times at which arm 1 is played (by construction $\tau_1 = 1$ and $\tau_2 = 2$). The evolution of the process $\langle X_{\tau_n} \rangle_{n \geq 1}$ can also be described by transition matrices. The transition probabilities to move from a state at $t = 1$ to a state at $t = 2$ are just the probabilities summarized in T. For all other t ($t \geq 2$) the transition matrix is either

$$\tilde{T} = \begin{pmatrix} 1-\epsilon & \epsilon \\ (T^k)_{21} & (T^k)_{22} \end{pmatrix} \quad \text{or} \quad \tilde{T} = \begin{pmatrix} (T^k)_{11} & (T^k)_{12} \\ \epsilon & 1-\epsilon \end{pmatrix} \tag{40}$$

depending on X_{τ_1}, i.e. if $X_{\tau_1} = 1$ then the former is describing the evolution and if $X_{\tau_1} = 0$ the latter is the one describing the evolution of the Markov chain. In the following we discuss the case that $X_{\tau_1} = 1$, but the same arguments apply to the case $X_{\tau_1} = 0$. We can observe that $\|v^\top(\tilde{T} - \hat{T})\|_\infty \leq \delta$ for all v with non-negative entries and $v_1 + v_2 = 1$, where

$$\hat{T} = \begin{pmatrix} 1-\epsilon & \epsilon \\ 1/2 & 1/2 \end{pmatrix}$$

in case that $X_{\tau_1} = 1$. The claim can be verified through

$$\left\| v^\top \tilde{T} - v^\top \hat{T} \right\|_\infty = \left\| v^\top \begin{pmatrix} 0 & 0 \\ \tilde{T}_{21} - 1/2 & \tilde{T}_{22} - 1/2 \end{pmatrix} \right\|_\infty$$
$$= v_2 \left\| T^k \begin{pmatrix} 0 \\ 1 \end{pmatrix} - \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix} \right\|_\infty \leq v_2 \delta \leq \delta.$$

The Markov chains associated to $\tilde{T}$ and $\hat{T}$ are both irreducible and aperiodic. This implies in particular the existence of stationary distributions, with the associated probabilities to be in state one and two summarized in vectors $\tilde{s}, \hat{s} \in [0,1]^2$, and the convergence of $\tilde{T}^l, \hat{T}^l$ to $\tilde{s}, \hat{s}$ in l (measured in $\|\|_\infty$, (Levin et al., 2008)[Thm. 4.9]). In particular, there exist constants $\tilde{c}, \hat{c} > 0$ and $\tilde{\alpha}, \hat{\alpha} \in (0,1)$ such that for all $l \geq 1$ and any vector $v \in [0,1]^2$ with $v_1 + v_2 = 1$

$$\|v^\top \tilde{T}^l - \tilde{s}\|_\infty \leq \tilde{c}\tilde{\alpha}^l \quad \text{and} \quad \|v^\top \hat{T}^l - \hat{s}\|_\infty \leq \hat{c}\hat{\alpha}^l.$$

We can also calculate the stationary distribution of $\hat{T}$ explicitly to get $\hat{s} = (1/(1+2\epsilon) \;\; 2\epsilon/(1+2\epsilon))^\top$.

It is known that Markov chains with slightly perturbed transition matrices have similar stationary distributions. Due to Cho and Meyer (2001) there exists a constant $c > 0$ that is only dependent on $\hat{T}$ (and independent of $\tilde{T}$) such that $\|\tilde{s} - \hat{s}\|_\infty \leq c\|\tilde{T} - \hat{T}\|_{\infty,1} \leq 2\delta c$ where $\|\tilde{T} - \hat{T}\|_{\infty,1} := \max_{i \in \{1,2\}} \sum_{j=1}^2 |\tilde{T}_{i,j} - \hat{T}_{i,j}| \leq 2\delta$. Combining these inequalities yields $\|v^\top \tilde{T}^l - \hat{s}\|_\infty \leq 2\delta c + \tilde{c}\tilde{\alpha}^l$ for any v with non-negative entries and $v_1 + v_2 = 1$.

Consider now the events $U = \{X_{\tau_1,1} = 1\}$ and $V = \{X_{\tau_n,1} = 1\}$ for $n \geq 2$. We have $P(U) = 1/2 = P(V)$ where the second equality follows from

$$2P(X_{\tau_n,1} = 0) = P(X_{\tau_n,1} = 0 | X_{\tau_1,1} = 0) + P(X_{\tau_n,1} = 0 | X_{\tau_1,1} = 1)$$
$$= P(X_{\tau_n,1} = 1 | X_{\tau_1,1} = 1) + P(X_{\tau_n,1} = 1 | X_{\tau_1,1} = 0)$$
$$= 2P(X_{\tau_n,1} = 1).$$

Furthermore, with $\tilde{T}$ being the matrix defined on the left side of (40)

$$\left| \frac{P(U \cap V)}{P(U)} - \frac{1}{1+2\epsilon} \right| = \left| (1-\epsilon \;\; \epsilon) \tilde{T}^{n-1} \begin{pmatrix} 1 \\ 0 \end{pmatrix} - \frac{1}{1+2\epsilon} \right|$$
$$\leq \left\| (1-\epsilon \;\; \epsilon) \tilde{T}^{n-1} - \frac{1}{1+2\epsilon} \begin{pmatrix} 1 \\ 2\epsilon \end{pmatrix} \right\|_\infty$$

and the last term is upper bounded by $2\delta c + \tilde{c}\tilde{\alpha}^{n-1}$. Recalling that c does not depend on $\tilde{T}$ and, hence, not on δ we see that we can make the term $2\delta c$ arbitrary small. Furthermore, by considering a large n we can make the second term arbitrary small. In particular, let $\epsilon = 1/10$ then there exists a π^δ and an $N \in \mathbb{N}$ such that for all $n \geq N$

$$\left| \frac{P(U \cap V)}{P(U)} - \frac{1}{1+2\epsilon} \right| \leq 1/10. \tag{41}$$

Hence, for all $n \geq N$

$$|P(U)P(V) - P(U \cap V)| \geq P(U)\Big(\Big| \frac{1}{2} - \frac{1}{1+2\epsilon} \Big| - 1/10 \Big)$$
$$\geq P(U)/5$$

and the process is not φ-mixing.

Appendix B. Proof of Theorem 12: Regret Bound for φ-mixing Bandits

For the regret $\overline{\mathcal{R}}(n)$ of Algorithm 1 after n rounds of play. We have,

$$\overline{\mathcal{R}}(n) \leq \sum_{\substack{i=1 \\ \mu_i \neq \mu^*}}^k \frac{32(1 + 8\|\varphi\|)\ln n}{\Delta_i} + (1 + 2\pi^2/3)(\sum_{i=1}^k \Delta_i) + \|\varphi\| \log n$$

Proof Thanks to Proposition 11, in order to bound the regret $\overline{\mathcal{R}}(n)$ it suffices to calculate the expected number of times $T_i(n)$ that a suboptimal arm is played in n rounds. For any $s, t \in \mathbb{N}_+$, let $c_{t,s} := \sqrt{(8\zeta((1/8) + \ln t))/2^s} + \|\varphi\|/2^{s-1}$, where $\zeta = 1 + 8\|\varphi\|$. Recall that Algorithm 1

plays its selected arms in batches of exponentially growing length so that if arm j for $j \in 1..k$ is selected at round t, it is played for $2^{s_j(t)}$ consecutive time-steps, where $s_j(t)$ denotes the number of times that arm j has been selected up-to time t. Below, we denote by $\overline{X}_{j,s} := \frac{1}{2^s}\sum_{t'=t}^{t+2^s-1} X_{t',j}$ with $s := s_j(t)$, the Algorithm's estimate of the stationary mean of arm j selected at time t. As usual, a superscript "$*$" refers to the quantities for the arm with the highest stationary mean. Fix some $l \in \mathbb{N}_+$. We have,

$$\begin{aligned}
T_i(n) &= 1 + \sum_{t=k+1}^{n} \chi\{\pi_t = i\} \\
&\le 2^{l+1} + \sum_{t=2^{l+1}+1}^{n} \sum_{m=l+1}^{\log t} 2^m \chi\Big\{ \min_{s=1..\log t} \overline{X}^*_s + c_{t,s} \le \frac{1}{2^{m-1}} \sum_{u=t}^{t+2^{m-1}-1} X_{u,i} + c_{t,m-1} \Big\} \\
&\le 2^{l+1} + \sum_{t=1}^{\infty} \sum_{m=l+1}^{\log t} \sum_{s=1}^{\log t} 2^m \chi\Big\{ \overline{X}^*_s + c_{t,s} \le \frac{1}{2^{m-1}} \sum_{u=t}^{t+2^{m-1}-1} X_{u,i} + c_{t,m-1} \Big\}
\end{aligned}$$

For every $t \in \mathbb{N}$ and every $s \in 1..\log t$ we have that $\overline{X}^*_s + c_{t,s} \le \frac{1}{2^{m-1}}\sum_{u=t}^{t+2^{m-1}-1} X_{u,i} + c_{t,m-1}$ implies that

$$\overline{X}^*_s \le \mu^* - c_{t,s} \tag{42}$$

$$\frac{1}{2^{m-1}} \sum_{u=t}^{t+2^{m-1}-1} X_{u,i} \ge \mu_i + c_{t,m-1} \tag{43}$$

$$\mu^* < \mu_i + 2c_{t,m-1} \tag{44}$$

Now, observe that for a fixed $t \in 1..n$ we have,

$$\begin{aligned}
P(\overline{X}^*_s \le \mu^* - c_{t,s}) &\le P\left(\left| \overline{X}^*_s - \mu^* \right| \ge c_{t,s} \right) \\
&\le P\left(\left| \sum_{j=0}^{2^s-1} X^*_{\tau_s+j} - EX^*_{\tau_s+j} \right| + \left| \sum_{j=0}^{2^s-1} (EX^*_{\tau_s+j} - \mu^*) \right| \ge 2^s c_{t,s} \right) \\
&\le P\left(\left| \sum_{j=0}^{2^s-1} X^*_{\tau_s+j} - EX^*_{\tau_s+j} \right| \ge 2^s c_{t,s} - 2\left\|\varphi\right\| \right) && (45) \\
&\le \sqrt{e} \exp\Big(- \frac{(2^s c_{t,s} - 2\left\|\varphi\right\|)^2}{2^{s+1}\zeta} \Big) && (46) \\
&\le t^{-4}, && (47)
\end{aligned}$$

where, (45) follows from Lemma 10 and (46) follows from a Hoeffding-type bound for φ-mixing processes given by Corollary 2.1 of Rio (1999) which is applicable due to Lemma 14 (pp. 17).

Moreover, noting that $\|\varphi\| \ge 0$ we similarly obtain,

$$\begin{aligned}
P\Big(\frac{1}{2^{m-1}} \sum_{u=t}^{t+2^{m-1}-1} X_{u,i} \ge \mu_i + c_{t,m-1}\Big) &\le P\Big(\frac{1}{2^{m-1}} \sum_{u=t}^{t+2^{m-1}-1} X_{u,i} \ge \mu_i + c_{t,m-1} - \frac{2}{2^{m-1}}\left\|\varphi\right\|\Big) \\
&\le P\Big(\Big| \sum_{u=t}^{t+2^{m-1}-1} X_{u,i} - 2^{m-1}\mu_i \Big| \ge 2^{m-1} c_{t,m-1} - 2\left\|\varphi\right\|\Big) \\
&\le t^{-4}. && (48)
\end{aligned}$$

Let $L := \log \frac{32(1+4\|\varphi\|)\ln n}{\Delta_i^2}$. Since, for $t \ge 2^L+1$ and every $m \ge L+1$ we have $\mu^* - \mu_i - 2c_{t,m-1} \ge 0$, it follows that (44) is false for all $m \ge L+1$. Therefore we have,

$$\begin{aligned}
E(T_i(n)) &\le 2^{L+1} + \sum_{t=1}^{\infty} \sum_{m=L+1}^{\log t} \sum_{s=1}^{\log t} P(\overline{X}^*_s \le \mu^* - c_{t,s}, \frac{1}{2^{m-1}} \sum_{u=t}^{t+2^{m-1}-1} X_{u,i} \ge \mu_i + c_{t,m-1}) \\
&\le 2^{L+1} + \sum_{t=1}^{\infty} \sum_{m=L+1}^{\log t} \sum_{s=1}^{\log t} 2^{m+1} t^{-4} && (49) \\
&\le \frac{32(1+4\left\|\varphi\right\|)\ln n}{\Delta_i^2} + 1 + 2\pi^2/3
\end{aligned}$$

where (49) follows from (47) and (48). ■

Appendix C. Proofs for Strongly Dependent Reward Distributions

C.1. Basic Bounds

Consider two independent and normally distributed random variables X, Y with mean $\mu_X > \mu_Y$ and variance σ_X^2, σ_Y^2. Let $\Delta = \mu_X - \mu_Y > 0$ and $\sigma^2 = \sigma_X^2 + \sigma_Y^2$. We derive in this section the following bounds which are crucial for the derivation of the regret. We use the notation $z^+ = \max\{0, z\}$ and $\phi(x) = (2\pi)^{-1/2}\exp(-x^2/2)$ for the density function of the standard normal distribution.

$$E(Y-X)^+ \le \sigma\phi(\Delta/\sigma), \tag{50}$$

$$E(Y-X)^+ \ge 0, \tag{51}$$

$$E(X-Y)^+ \le \sigma\phi(\Delta/\sigma) + \Delta, \tag{52}$$

$$E(X-Y)^+ \ge \Delta. \tag{53}$$

The derivation is based on basic properties of Gaussian random variables. Recall that $Z = X - Y$ is normally distributed with mean Δ and variance σ^2. Therefore,

$$\begin{aligned}
\sqrt{2\pi}\sigma E(X-Y)^+ &= \int_0^\infty z \exp\left(-\frac{(z-\Delta)^2}{2\sigma^2}\right) \\
&= \int_{-\Delta}^\infty z \exp\left(-\frac{z^2}{2\sigma^2}\right) + \Delta \int_{-\Delta}^\infty \exp\left(-\frac{z^2}{2\sigma^2}\right) \\
&= \sigma^2 \exp\left(-\frac{\Delta^2}{2\sigma^2}\right) + \Delta\sigma \int_{-\infty}^{\Delta/\sigma} \exp\left(-\frac{z^2}{2}\right)
\end{aligned}$$

Standard bounds on the cdf, as can be found in (Dudley, 2002)[Lem. 12.1.6], lead to (53), i.e.

$$\begin{aligned}\sqrt{2\pi}\sigma E(X-Y)^+ &= \sigma^2 \exp\left(-\frac{\Delta^2}{2\sigma^2}\right) + \Delta\sigma\sqrt{2\pi}\left(1 - \frac{1}{\sqrt{2\pi}}\int_{\Delta/\sigma}^{\infty}\exp\left(-\frac{z^2}{2}\right)\right)\\ &\geq \sigma^2 \exp\left(-\frac{\Delta^2}{2\sigma^2}\right) + \Delta\sigma\sqrt{2\pi}\left(1 - \frac{\sigma}{\sqrt{2\pi}\Delta}\exp\left(-\frac{\Delta^2}{2\sigma^2}\right)\right)\\ &= \Delta\sigma\sqrt{2\pi}.\end{aligned}$$

Similarly, if we consider $Z = Y - X$ which has mean $-\Delta$

$$\begin{aligned}\sqrt{2\pi}\sigma E(Y-X)^+ &= \int_0^{\infty} z\exp\left(-\frac{(z+\Delta)^2}{2\sigma^2}\right)\\ &= \sigma^2\exp\left(-\frac{\Delta^2}{2\sigma^2}\right) - \Delta\sigma\int_{\Delta/\sigma}^{\infty}\exp\left(-\frac{z^2}{2}\right).\end{aligned}$$

Applying the result from Dudley (2002)[Lem. 12.1.6] leads here to the trivial lower bound 0 and, hence, inequality (51). We use the following inequalities to obtain upper bounds on $E(Y-X)^+$ and $E(X-Y)^+$. Let Z be a standard normal random variable. Then

$$\Pr(Z \geq c) \geq \begin{cases}\phi(c)/(2c) & \text{if } c \geq 1,\\ \phi(c)(1-c)/2 & \text{if } 0 \leq c < 1.\end{cases}$$

The first bound can be found in (Dudley, 2014). The second bound is a straightforward adaptation of the techniques used to derive the first bound. Applying these bounds we get the following upper bounds on $E(Y-X)^+$. If $\Delta/\sigma \geq 1$ then

$$\begin{aligned}\sqrt{2\pi}\sigma E(Y-X)^+ &\leq \sigma^2\exp\left(-\frac{\Delta^2}{2\sigma^2}\right) - \Delta\sigma\frac{\sigma}{2\Delta}\exp\left(-\frac{\Delta^2}{2\sigma^2}\right)\\ &= \frac{\sigma^2}{2}\exp\left(-\frac{\Delta^2}{2\sigma^2}\right)\end{aligned}$$

and if $0 \leq \Delta/\sigma < 1$

$$\begin{aligned}\sqrt{2\pi}\sigma E(Y-X)^+ &\leq \sigma^2\exp\left(-\frac{\Delta^2}{2\sigma^2}\right) - \Delta\sigma\frac{(1-\Delta/\sigma)}{2}\exp\left(-\frac{\Delta^2}{2\sigma^2}\right)\\ &= (\sigma^2 - \Delta\sigma/2 + \Delta^2/2)\exp\left(-\frac{\Delta^2}{2\sigma^2}\right)\\ &\leq \sigma^2\exp\left(-\frac{\Delta^2}{2\sigma^2}\right)\end{aligned}$$

and the inequality (52) follows. The upper bounds on $E(X-Y)^+$ are derived in the same way. For $\Delta/\sigma \geq 1$ these are

$$\begin{aligned}\sqrt{2\pi}\sigma E(X-Y)^+ &\leq \sigma^2\exp\left(-\frac{\Delta^2}{2\sigma^2}\right) - (\sigma^2/2)\exp\left(-\frac{\Delta^2}{2\sigma^2}\right) + \Delta\sigma\sqrt{2\pi}\\ &= \frac{\sigma^2}{2}\exp\left(-\frac{\Delta^2}{2\sigma^2}\right) + \Delta\sigma\sqrt{2\pi}\end{aligned}$$

and for $0 \leq \Delta/\sigma \leq 1$

$$\begin{aligned}\sqrt{2\pi}\sigma E(X-Y)^+ &\leq \sigma^2\exp\left(-\frac{\Delta^2}{2\sigma^2}\right) - \frac{\Delta\sigma(1-\Delta/\sigma)}{2}\exp\left(-\frac{\Delta^2}{2\sigma^2}\right) + \Delta\sigma\sqrt{2\pi}\\ &= (\sigma^2 - \Delta\sigma/2 + \Delta^2/2)\exp\left(-\frac{\Delta^2}{2\sigma^2}\right) + \Delta\sigma\sqrt{2\pi}\end{aligned}$$

and the inequality (50) follows.

C.2. Proof of Proposition 13

We bound the regret of the two phases individually. Some technical steps are moved further below to streamline the discussion.

Regret of Phase I. We sweep through all k arms at times 1 to k, $m+1$ to $m+k$, etc. During each of these phases we build up regret. This regret can be bounded by

$$\begin{aligned}&\sum_{i=1}^{k} E(\max_{j\neq i} X_{m+i,j} - X_{m+i,i})^+\\ &\qquad\leq \sum_{i=1}^{k}\sum_{j\neq i} E(X_{m+i,j} - X_{m+i,i})^+\\ &\qquad\leq (k-1)\sum_{i=1}^{k}\max_{j\neq i} E(X_{m+i,j} - X_{m+i,i})^+\\ &\qquad\leq (k-1)\sum_{i=1}^{k}\Delta_i + \sqrt{2}(k-1)\sum_{i=1}^{k}\phi(\Delta_i/\sqrt{2})\\ &\qquad\leq (k-1)\sum_{i=1}^{k}(\Delta_i + \sqrt{2}),\end{aligned} \tag{54}$$

where $\Delta_i = \mu^* - \mu_i$ is the difference between the highest stationary mean of the k arms and the stationary mean of arm i. Here, we use Inequality (52) and the assumption that the variance of the individual processes is 1. Therefore, the σ appearing in the bound (52) is $\sqrt{2}$ since σ^2 is the sum of the variances of the two individual processes.

Regret of Phase II. To control the regret building up in the second phase we condition on the observations in a sweep at time lm, $l \in \mathbb{N}$, i.e. on the observed pay-offs $x_1, \ldots, x_k$. Arm i^* is selected such that $x_{i^*} \geq \max_{i\leq k} x_i$ and this arm is played for $m-k$ steps. We need to control $E(\max_{i\leq k}(X_{t',i} - X_{t',\pi_{t'}})^+)$ for all $lm+k+1 \leq t' \leq (l+1)m$. Due to stationarity this is equal to controlling $E(\max_{i\leq k}(X_{t,i} - X_{t,\pi_t})^+)$, for all $k+1 \leq t \leq m$.

We use in the following $P_{x_1,\ldots,x_k}$, P_{x_i,x_u} for the conditional distributions given that $X_{11} = x_1, \ldots, X_{kk} = x_k$ and $X_{ii} = x_i, X_{uu} = x_u$, and we use ν for the marginal measure. In Appendix

C.2.1 below we show that for any $k+1 \leq t \leq m$ it holds that

$$E(\max_{i\leq k}(X_{t,i} - X_{t,\pi_t})^+) \leq \sum_{u=1}^{k}\sum_{i\neq u}\int P_{x_1,\ldots,x_k}(u = i^*) \times \int (X_{t,i} - X_{t,u})^+ dP_{x_i,x_u} d\nu(x_1,\ldots,x_k). \quad (55)$$

The inner integral in (55) can be bounded by using inequality (50)

$$\int (X_{t,i} - X_{t,u})^+ dP_{x_i,x_u} \leq \sigma_{t,i}\phi\left(\frac{\Delta_{t,i}}{\sigma_{t,i}}\right), \quad (56)$$

where $\Delta_{t,i} = E(X_{t,u} - X_{t,i}|X_{i,i} = x_i, X_{u,u} = x_u)$ which equals $E(X_{t,u}|X_{u,u} = x_u) - E(X_{t,i}|X_{i,i} = x_i)$ due to the independence between arm u and i. These are posterior means of Gaussian processes given observations x_u and x_i, respectively. Observe that the posterior mean for $X_{t,u}$ is related to the posterior mean of the zero mean Gaussian process $X_{t,u} - \mu_u$ through $E(X_{t,u}|X_{u,u} = x_u) = \mu_u + E(X_{t,u} - \mu_u|X_{u,u} - \mu_u = x_u - \mu_u)$. The posterior equation for the zero mean Gaussian process $X_{t,u} - \mu_u$ given observation $x_u - \mu_u$ at time u is $\text{cov}(t-u)(x_u - \mu_u)$ since $\text{cov}(0) = 1$. Therefore, $E(X_{t,u}|X_{u,u} = x_u) = \mu_u + \text{cov}(t-u)(x_u - \mu_u)$ and, hence, $E(X_{t,u}|X_{u,u} = x_u) = (1 - \text{cov}(t-u))\mu_u + \text{cov}(t-u)x_u$. Similarly, $E(X_{t,i}|X_{i,i} = x_i) = (1 - \text{cov}(t-i))\mu_i + \text{cov}(t-i)x_i$ By defining $\tilde{\Delta}_i = x_u - x_i$ and by using the Hölder-continuity assumption, we obtain the following lower bound.

$$\begin{aligned}\Delta_{t,i} &= (\mu_u - \mu_i)(1 - \text{cov}(t-k)) + \tilde{\Delta}_i \text{cov}(t-k) + \varepsilon(t)\\ &\geq \tilde{\Delta}_i - |\mu_u - \mu_i||\text{cov}(0) - \text{cov}(t-k)| - \tilde{\Delta}_i|\text{cov}(0) - \text{cov}(t-k)| - |\varepsilon(t)|\\ &\geq \tilde{\Delta}_i - c(t-k)^\alpha(\Delta + \tilde{\Delta}_i) - |\varepsilon(t)| \quad (57)\end{aligned}$$

with $\varepsilon(t)$ being a term which can be bounded by $|\varepsilon(t)| \leq (\Delta + \tilde{\Delta}_i)ck^\alpha$ (see Appendix C.2.2). For the conditional variance,

$$\sigma^2_{t,i} = E(X^2_{t,u}|X_{u,u} = x_u) - E(X_{t,u}|X_{u,u} = x_u)^2 + E(X^2_{t,i}|X_{i,i} = x_i) - E(X_{t,i}|X_{i,i} = x_i)^2,$$

we obtain the following upper-bound

$$\sigma^2_{t,i} = 2 - \text{cov}^2(t-u) - \text{cov}^2(t-i) \leq 2c(t-u)^\alpha + 2c(t-i)^\alpha \leq 4ct^\alpha, \quad (58)$$

where we have used the posterior equation for the covariance, that the covariance is by assumption non-negative, and that, under the Hölder-continuity assumption, for a non-negative covariance $\text{cov}(l) \geq \max\{0, (1 - cl^\alpha)\}$ and, hence,

$$\begin{aligned}\text{cov}^2(l) &\geq \text{cov}(l)\max\{0, 1 - cl^\alpha\}\\ &= \begin{cases}0 & \text{if } cl^\alpha \geq 1,\\ \text{cov}(l)(1 - cl^\alpha) & \text{otherwise,}\end{cases}\\ &\geq \begin{cases}0 & \text{if } cl^\alpha \geq 1,\\ 1 - 2cl^\alpha + c^2l^{2\alpha} & \text{otherwise,}\end{cases}\\ &\geq 1 - 2cl^\alpha.\end{aligned}$$

Combining (56), (57) and (58) we obtain

$$\begin{aligned}&\int (X_{t,i} - X_{t,u})^+ dP_{x_i,x_u}\\ &\leq (2/\pi)^{1/2}c^{1/2}t^{\alpha/2}\exp\left(-\frac{(\tilde{\Delta}_i - c((t-k)^\alpha + k^\alpha)(\tilde{\Delta}_i + \Delta))^2}{8ct^\alpha}\right)\\ &= (2/\pi)^{1/2}c^{1/2}t^{\alpha/2}\exp\left(-\frac{\tilde{\Delta}_i^2}{8ct^\alpha}\Big(1 - c((t-k)^\alpha + k^\alpha)\Big(1 + \frac{\Delta}{\tilde{\Delta}_i}\Big)\Big)^2\right). \quad (59)\end{aligned}$$

The bound is maximized for $t = m$. Substituting this bound back into (55) and after a few manipulations (see Appendix C.2.3) we see that

$$\begin{aligned}&\sum_{t=k+1}^{m} E\Big(\max_{i\leq k}(X_{t,i} - X_{t,\pi_t})^+\Big)\\ &\qquad \leq (m-k)\frac{a_m c^{1/2}k(k-1)}{8\pi(1-b_m)}\left(2\pi^{1/2} - (1 - \Delta\sqrt{b_m/4})\exp\left(-\frac{\Delta^2 b_m}{4}\right)\right) \quad (60)\end{aligned}$$

if $m > k$ and $\Delta < \sqrt{a_m}/(b_m\sqrt{2})$, where $a_m = 8cm^\alpha$ and $b_m = c((m-k)^\alpha + k^\alpha)$.

Combined Regret. Combining (54) with (60) and using $\Delta_i \leq \Delta$ we can observe that the combined regret for any m steps is bounded by

$$k(k-1)\left(\Delta + \sqrt{2} + (m-k)\frac{a_m c^{1/2}}{8\pi(1-b_m)}\left(2\pi^{1/2} - (1 - \Delta\sqrt{b_m/4})\exp\left(-\frac{\Delta^2 b_m}{4}\right)\right)\right)$$

if $m > k$ and $\Delta < \sqrt{a_m}/(b_m\sqrt{2})$. Given a time horizon n we have $\lceil n/m \rceil$ many iterations of Phase I and II and the overall regret is bounded by

$$\begin{aligned}&(n/m+1)k(k-1)\left(\Delta + \sqrt{2} + \frac{a_m c^{1/2}(m-k)}{8\pi(1-b_m)}\left(2\pi^{1/2} - (1 - \Delta\sqrt{b_m/4})\exp\left(-\frac{\Delta^2 b_m}{4}\right)\right)\right)\\ &\leq (n+m)k(k-1)\left(\frac{\Delta + \sqrt{2}}{m} + \frac{a_m c^{1/2}}{8\pi(1-b_m)}\left(2\pi^{1/2} - (1 - \Delta\sqrt{b_m/4})\exp\left(-\frac{\Delta^2 b_m}{4}\right)\right)\right). \quad (61)\end{aligned}$$

This is the regret bound stated in the proposition text.

Instead of trying to find the m that minimizes (61) we optimize m over the considerably simpler expression

$$\frac{\Delta + \sqrt{2}}{m} + \frac{2c^{3/2}m^\alpha}{\sqrt{\pi}}\left(= \frac{\Delta + \sqrt{2}}{m} + \frac{a_m c^{1/2}}{4\sqrt{\pi}}\right). \quad (62)$$

The $(1 - b_m)$ term that we leave out is of minor relevance since b_m is small and the negative term in the bracket that we leave out is not larger than 1. By ignoring this latter term we lose only another constant. Minimizing (62) with respect to m and rounding up yields

$$m^\star = \left\lceil \left(\frac{\sqrt{\pi}(\Delta + \sqrt{2})}{2\alpha c^{3/2}}\right)^{\frac{1}{1+\alpha}} \right\rceil.$$

C.2.1. PROOF OF INEQUALITY (55)

Due to stationarity and since our policy depends only on the observations at the last sweep we have that $E(\max_{i\le k}(X_{t',i}-X_{t',\pi_{t'}})^+)=E(\max_{i\le k}(X_{t,i}-X_{t,\pi_t})^+)$ for any t', $lm+k+1\le t'\le lm$, $l\in\mathbb{N}$, and corresponding $t=t'-lm$. Now, consider any t, $k+1\le t\le m$ and, using i^* for the choice of arm given the observations $X_{1,1},\dots,X_{k,k}$, rewrite the regret in the following way

$$
\begin{aligned}
&E\Big(\max_{i\le k}(X_{t,i}-X_{t,\pi_t})^+\Big)\\
&=\sum_{u=1}^{k}E\Big(\max_{i\le k}\chi\{u=i^*\}\times(X_{t,i}-X_{t,i^*})^+\Big)\\
&=\sum_{u=1}^{k}E\Big(\max_{i\neq u}\chi\{u=i^*\}\times(X_{t,i}-X_{t,u})^+\Big)\\
&\le\sum_{u=1}^{k}\sum_{i\neq u}E\Big(\chi\{u=i^*\}\times(X_{t,i}-X_{t,u})^+\Big)\\
&=\sum_{u=1}^{k}\sum_{i\neq u}\int\int\chi\{u=i^*\}\times(X_{t,i}-X_{t,u})^+dP_{x_1,\dots,x_k}d\nu(x_1,\dots,x_k)\\
&=\sum_{u=1}^{k}\sum_{i\neq u}\int P_{x_1,\dots,x_k}(u=i^*)\times\int(X_{t,i}-X_{t,u})^+dP_{x_1,\dots,x_k}d\nu(x_1,\dots,x_k) \qquad (63)\\
&=\sum_{u=1}^{k}\sum_{i\neq u}\int P_{x_1,\dots,x_k}(u=i^*)\times\int(X_{t,i}-X_{t,u})^+dP_{x_i,x_u}d\nu(x_1,\dots,x_k) \qquad (64)
\end{aligned}
$$

where (63) follows because $\chi\{u=i^*\}$ is independent of $(X_{t,i}-X_{t,u})^+$ given $X_{11}=x_1,\dots,X_{kk}=x_k$ and (64) follows since $(X_{t,i}-X_{t,u})^+$ only depends on $X_{i,i}$ and $X_{u,u}$.

C.2.2. BOUND ON $\varepsilon(t)$.

The term $\varepsilon(t)$ that we introduced is equal to

$$
\begin{aligned}
\varepsilon(t)=&\mu_u(\mathrm{cov}(t-k)-\mathrm{cov}(t-u))-\mu_i(\mathrm{cov}(t-k)-\mathrm{cov}(t-i))\\
&+x_u(\mathrm{cov}(t-u)-\mathrm{cov}(t-k))-x_i(\mathrm{cov}(t-i)-\mathrm{cov}(t-k)).
\end{aligned}
$$

Since, due to our Hölder assumption, $|\mathrm{cov}(t-k)-\mathrm{cov}(t-u)|\le c(k-u)^\alpha\le ck^\alpha$ and $|\mathrm{cov}(t-k)-\mathrm{cov}(t-i)|\le ck^\alpha$, we have the following bound.

$$
|\varepsilon(t)|\le(\Delta+\tilde{\Delta}_i)ck^\alpha.
$$

C.2.3. PROOF OF INEQUALITY (60)

Denote in the following $X_u:=X_{u,u}$ and $X_i=X_{i,i}$ and recall that $\tilde{\Delta}_i=X_u-X_i$ is normally distributed with mean $\mu_u-\mu_i$ and variance 2. Writing

$$
f(X_u,X_i)=(2/\pi)^{1/2}c^{1/2}m^{\alpha/2}\exp\left(-\frac{\tilde{\Delta}_i^2}{8cm^\alpha}\Big(1-c((m-k)^\alpha+k^\alpha)\Big(1+\frac{\Delta}{\tilde{\Delta}_i}\Big)\Big)^2\right)
$$

we have

$$
\begin{aligned}
&\int P_{x_1,\dots,x_k}(u=i^*)\times\int(X_{t,i}-X_{t,u})^+dP_{x_i,x_u}\,d\nu(x_1,\dots,x_k)\\
&\le\int P_{x_1,\dots,x_k}(X_u\ge X_i)\times\int(X_{t,i}-X_{t,u})^+dP_{x_i,x_u}\,d\nu(x_1,\dots,x_k)\\
&\le\int\chi\{x_u\ge x_i\}\times f(x_u,x_i)\,d\nu(x_1,\dots,x_k) \qquad (65)\\
&=\int\chi\{x_u-x_i\ge0\}\times f(x_u,x_i)\,d\nu(x_u,x_i), \qquad (66)
\end{aligned}
$$

where the inequality in (65) follows because $\{X_u\ge X_i\}$ is only dependent on x_u and x_i and by bounding the inner integral with (59). Multiplying the density of $\tilde{\Delta}_i$ by f and using $a=8cm^\alpha$, $b=c((m-k)^\alpha+k^\alpha)$, $d=(ac)^{1/2}/(4\pi)$, in the case where $\Delta<\sqrt{a}/(b\sqrt{2})$, we obtain

$$
\begin{aligned}
&\int\chi\{x_u-x_i\ge0\}\times f(x_u,x_i)\,d\nu(x_u,x_i)\\
&\qquad=d\int_0^\infty\exp\left(-\frac{(\tilde{\Delta}_i(1-b)-\Delta b)^2}{a}-\frac{(\tilde{\Delta}_i-(\mu_u-\mu_i))^2}{4}\right)d\tilde{\Delta}_i\\
&\qquad\le d\int_0^\infty\exp\left(-\frac{(\tilde{\Delta}_i(1-b)-\Delta b)^2}{a}\right)d\tilde{\Delta}_i\\
&\qquad=\frac{d}{1-b}\int_{-\Delta b}^\infty\exp\left(-\frac{\tilde{\Delta}_i^2}{a}\right)d\tilde{\Delta}_i\\
&\qquad=\frac{\sqrt{a}d}{(1-b)\sqrt{2}}\int_{-\infty}^{\frac{\sqrt{2}\Delta b}{\sqrt{a}}}\exp\left(-\frac{\tilde{\Delta}_i^2}{2}\right)d\tilde{\Delta}_i\\
&\qquad=\frac{\sqrt{a}d\sqrt{2\pi}}{(1-b)\sqrt{2}}\left(1-\frac{1}{\sqrt{2\pi}}\int_{\frac{\sqrt{2}\Delta b}{\sqrt{a}}}^\infty\exp\left(-\frac{\tilde{\Delta}_i^2}{2}\right)d\tilde{\Delta}_i\right)\\
&\qquad\le\frac{\sqrt{\pi a}d}{(1-b)}\left(1-\frac{1-\sqrt{2}\Delta b/\sqrt{a}}{2\sqrt{2\pi}}\exp\left(-\frac{\Delta^2b^2}{a}\right)\right)\\
&\qquad=\frac{ac^{1/2}}{4(1-b)\sqrt{\pi}}\left(1-\frac{1-\sqrt{2}\Delta b/\sqrt{a}}{2\sqrt{2\pi}}\exp\left(-\frac{\Delta^2b^2}{a}\right)\right)\\
&\qquad\le\frac{ac^{1/2}}{8\pi(1-b)}\left(2\pi^{1/2}-(1-\Delta b/\sqrt{a})\exp\left(-\frac{\Delta^2b^2}{a}\right)\right)
\end{aligned}
$$

and for $m>k$, since then $a/4\ge b$, this can be further bounded by

$$
\frac{ac^{1/2}}{8\pi(1-b)}\left(2\pi^{1/2}-(1-\Delta\sqrt{b/4})\exp\left(-\frac{\Delta^2b}{4}\right)\right).
$$

References

J. Audiffren and L. Ralaivola. Cornering stationary and restless mixing bandits with remix-ucb. In *Advances in Neural Information Processing Systems*, 2015.

P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multi-armed bandit problem. *Machine Learning*, 2002.

P. Auer and R. Ortner. UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica*, 61(1-2):55–65, 2010.

R. C. Bradley. Basic properties of strong mixing conditions. A survey and some open questions. *Probability surveys*, 2(2):107–144, 2005.

R. C. Bradley. *Introduction to Strong Mixing Conditions, Vols. 1, 2 and 3.* Kendrick Press, 2007.

G. E. Cho and C. D. Meyer. Comparison of perturbation bounds for the stationary distribution of a markov chain. *Linear Algebra and its Applications*, 2001.

P. Doukhan. *Mixing: Properties and Examples.* Springer Lecture Notes, 1994.

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

R. M. Dudley. *Uniform Central Limit Theorems.* Cambridge University Press, second edition, 2014.

D. H. Fremlin. *Measure Theory.* Torres Fremlin, Colchester, England, 2010. Vol. 1 and 2.

E. Giné and R. Nickl. *Mathematical Foundations of Infinite-dimensional Statistical Models.* Cambridge University Press, 2016.

S. Guha, K. Munagala, and P. Shi. Approximation algorithms for restless bandit problems. *Journal of the ACM (JACM)*, 58(1):3, 2010.

D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov Chains and Mixing Times.* American Mathematical Society, 2008.

R. Ortner, D. Ryabko, P. Auer, and R. Munos. Regret bounds for restless Markov bandits. *Theoretical Computer Science*, 558:62–76, 2014.

C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of optimal queuing network control. *Mathematics of Operations Research*, 1999.

E. Rio. *Théorie asymptotique des processus aléatoires faiblement dépendants.* Springer, 1999.

A. N. Shiryaev. *Probability.* Springer, 1991.

P. Whittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, 1988.

Journal of Machine Learning Research 20 (2019) 1-38 Submitted 10/17; Revised 12/18; Published 2/19

Automated Scalable Bayesian Inference via Hilbert Coresets

Trevor Campbell TREVOR@STAT.UBC.CA
Department of Statistics
University of British Columbia
Vancouver, BC V6T 1Z4, Canada

Tamara Broderick TBRODERICK@CSAIL.MIT.EDU
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Editor: Zhihua Zhang

Abstract

The automation of posterior inference in Bayesian data analysis has enabled experts and nonexperts alike to use more sophisticated models, engage in faster exploratory modeling and analysis, and ensure experimental reproducibility. However, standard automated posterior inference algorithms are not tractable at the scale of massive modern data sets, and modifications to make them so are typically model-specific, require expert tuning, and can break theoretical guarantees on inferential quality. Building on the Bayesian coresets framework, this work instead takes advantage of data redundancy to shrink the data set itself as a preprocessing step, providing fully-automated, scalable Bayesian inference with theoretical guarantees. We begin with an intuitive reformulation of Bayesian coreset construction as sparse vector sum approximation, and demonstrate that its automation and performance-based shortcomings arise from the use of the supremum norm. To address these shortcomings we develop Hilbert coresets, i.e., Bayesian coresets constructed under a norm induced by an inner-product on the log-likelihood function space. We propose two Hilbert coreset construction algorithms—one based on importance sampling, and one based on the Frank-Wolfe algorithm—along with theoretical guarantees on approximation quality as a function of coreset size. Since the exact computation of the proposed inner-products is model-specific, we automate the construction with a random finite-dimensional projection of the log-likelihood functions. The resulting automated coreset construction algorithm is simple to implement, and experiments on a variety of models with real and synthetic data sets show that it provides high-quality posterior approximations and a significant reduction in the computational cost of inference.

Keywords: Bayesian inference, scalable, automated, coreset, Hilbert, Frank–Wolfe

1. Introduction

Bayesian probabilistic models are a standard tool of choice in modern data analysis. Their rich hierarchies enable intelligent sharing of information across subpopulations, their posterior distributions provide many avenues for principled parameter estimation and uncertainty quantification, and they can incorporate expert knowledge through the prior. In all but the simplest models, however, the posterior distribution is intractable to compute exactly, and we must resort to approximate inference algorithms. Markov chain Monte Carlo (MCMC) (Gelman et al., 2013, Chapters 11, 12) methods are the gold standard, due primarily to their guaranteed asymptotic exactness. Variational Bayes (VB) (Jordan et al., 1999; Wainwright and Jordan, 2008) is also becoming widely used due to its tractability, detectable convergence, and parameter estimation performance in practice.

One of the most important recent developments in the Bayesian paradigm has been the automation of these standard inference algorithms. Rather than having to develop, code, and tune specific instantiations of MCMC or VB for each model, practitioners now have "black-box" implementations that require only a basic specification of the model as inputs. For example, while standard VB requires the specification of model gradients—whose formulae are often onerous to obtain—and an approximating family—whose rigorous selection is an open question—ADVI (Ranganath et al., 2014; Kucukelbir et al., 2015, 2017) applies standard transformations to the model so that a multivariate Gaussian approximation can be used, and computes gradients with automatic differentiation. The user is then left only with the much simpler task of specifying the log-likelihood and prior. Similarly, while Hamiltonian Monte Carlo (Neal, 2011) requires tuning a step size and path length parameter, NUTS (Hoffman and Gelman, 2014) provides a method for automatically determining reasonable values for both. This level of automation has many benefits: it enables experts and nonexperts alike to use more sophisticated models, it facilitates faster exploratory modeling and analysis, and helps ensure experimental reproducibility.

But as modern data sets continue to grow larger over time, it is important for inference to be not only automated, but *scalable* while retaining *theoretical guarantees* on the quality of inferential results. In this regard, the current set of available inference algorithms falls short. Standard MCMC algorithms may be "exact", but they are typically not tractable for large-scale data, as their complexity per posterior sample scales at least linearly in the data set size. Variational methods on the other hand are often scalable, but posterior approximation guarantees continue to elude researchers in all but a few simple cases. Other scalable Bayesian inference algorithms have largely been developed by modifying standard inference algorithms to handle distributed or streaming data processing. Examples include subsampling and streaming methods for variational Bayes (Hoffman et al., 2013; Broderick et al., 2013; Campbell et al., 2015), subsampling methods for MCMC (Welling and Teh, 2011; Ahn et al., 2012; Bardenet et al., 2014; Korattikara et al., 2014; Maclaurin and Adams, 2014; Bardenet et al., 2015), and distributed "consensus" methods for MCMC (Scott et al., 2016; Srivastava et al., 2015; Rabinovich et al., 2015; Entezari et al., 2016). These methods either have no guarantees on the quality of their inferential results, or require expensive iterative access to a constant fraction of the data, but more importantly they tend to be model-specific and require extensive expert tuning. This makes them poor candidates for automation on the large class of models to which standard automated inference algorithms are applicable.

An alternative approach, based on the observation that large data sets often contain redundant data, is to modify the data set itself such that its size is reduced while preserving its original statistical properties. In Bayesian regression, for example, a large data set can be compressed using random linear projection (Geppert et al., 2017; Ahfock et al., 2017; Bardenet and Maillard, 2015). For a wider class of Bayesian models, one can construct a

©2019 Trevor Campbell and Tamara Broderick.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-613.html.

small weighted subset of the data, known as a *Bayesian coreset*[1] (Huggins et al., 2016), whose weighted log-likelihood approximates the full data log-likelihood. The coreset can then be passed to any standard (automated) inference algorithm, providing posterior inference at a significantly reduced computational cost. Note that since the coresets approach is agnostic to the particular inference algorithm used, its benefits apply to the continuing developments in both MCMC (Robert et al., 2018) and variational (Dieng et al., 2017; Li and Turner, 2016; Liu and Wang, 2016) approaches.

Bayesian coresets, in contrast to other large-scale inference techniques, are simple to implement, computationally inexpensive, and have theoretical guarantees relating coreset size to both computational complexity and the quality of approximation (Huggins et al., 2016). However, their construction cannot be easily automated, as it requires computing the *sensitivity* (Langberg and Schulman, 2010) of each data point, a model-specific task that involves significant technical expertise. This approach also often necessitates a bounded parameter space to ensure bounded sensitivities, precluding many oft-used continuous likelihoods and priors. Further, since Bayesian coreset construction involves i.i.d. random subsampling, it can only reduce approximation error compared to uniform subsampling by a constant, and cannot update its notion of importance based on what points it has already selected.

In this work, we develop a scalable, theoretically-sound Bayesian approximation framework with the same level of automation as ADVI and NUTS, the algorithmic simplicity and low computational burden of Bayesian coresets, and the inferential performance of hand-tuned, model-specific scalable algorithms. We begin with an intuitive reformulation of Bayesian coreset construction as sparse vector sum approximation, in which the data log-likelihood functions are vectors in a vector space, sensitivity is a weighted uniform (i.e. supremum) norm on those vectors, and the construction algorithm is importance sampling. This perspective illuminates the use of the uniform norm as the primary source of the shortcomings of Bayesian coresets. To address these issues we develop Hilbert coresets, i.e., Bayesian coresets using a norm induced by an inner-product on the log-likelihood function space. Our contributions include two candidate norms: one a weighted L^2 norm, and another based on the Fisher information distance (Johnson and Barron, 2004). Given these norms, we provide an importance sampling-based coreset construction algorithm and a more aggressive "direction of improvement"-aware coreset construction based on the Frank–Wolfe algorithm (Frank and Wolfe, 1956; Guélat and Marcotte, 1986; Jaggi, 2013). Our contributions include theoretical guarantees relating the performance of both to coreset size. Since the proposed norms and inner-products cannot in general be computed in closed-form, we automate the construction using a random finite-dimensional projection of the log-likelihood functions inspired by Rahimi and Recht (2007). We test Hilbert coresets empirically on multivariate Gaussian inference, logistic regression, Poisson regression, and von Mises-Fisher mixture modeling with both real and synthetic data; these experiments show that Hilbert coresets provide high quality posterior approximations with a significant reduction in the computational cost of inference compared to standard automated inference algorithms. All proofs are deferred to Appendix A.

1. The concept of a coreset originated in computational geometry and optimization (Agarwal et al., 2005; Feldman and Langberg, 2011; Feldman et al., 2013; Bachem et al., 2015; Lucic et al., 2016; Bachem et al., 2016; Feldman et al., 2011; Han et al., 2016).

2. Background

In the general setting of Bayesian posterior inference, we are given a data set $(y_n)_{n=1}^N$ of N observations, a likelihood $p(y_n|\theta)$ for each observation given the parameter $\theta \in \Theta$, and a prior density $\pi_0(\theta)$ on Θ. We assume throughout that the data are conditionally independent given θ. The Bayesian posterior is given by the density

$$\pi(\theta) := \frac{1}{Z}\exp(\mathcal{L}(\theta))\pi_0(\theta),$$

where the log-likelihood $\mathcal{L}(\theta)$ and marginal likelihood Z are defined by

$$\mathcal{L}_n(\theta) := \log p(y_n \,|\, \theta), \qquad \mathcal{L}(\theta) := \sum_{n=1}^N \mathcal{L}_n(\theta), \qquad Z := \int \exp(\mathcal{L}(\theta))\pi_0(\theta)\,\mathrm{d}\theta.$$

In almost all cases in practice, an exact closed-form expression of π is not available due to the difficulty of computing Z, forcing the use of approximate Bayesian inference algorithms. While Markov chain Monte Carlo (MCMC) algorithms (Gelman et al., 2013, Chapters 11, 12) are often preferred for their theoretical guarantees asymptotic in running time, they are typically computationally intractable for large N. One way to address this is to construct a small, weighted subset of the original data set whose log-likelihood approximates that of the full data set, known as a *Bayesian coreset* (Huggins et al., 2016). This coreset can then be passed to a standard MCMC algorithm. The computational savings from running MCMC on a much smaller data set can allow a much faster inference procedure while retaining the theoretical guarantees of MCMC. In particular, the aim of the Bayesian coresets framework is to find a set of nonnegative weights $w := (w_n)_{n=1}^N$, a small number of which are nonzero, such that the weighted log-likelihood

$$\mathcal{L}(w,\theta) := \sum_{n=1}^N w_n \mathcal{L}_n(\theta) \quad \text{satisfies} \quad |\mathcal{L}(w,\theta) - \mathcal{L}(\theta)| \leq \epsilon\,|\mathcal{L}(\theta)|\,,\ \forall \theta \in \Theta. \tag{2.1}$$

The algorithm proposed by Huggins et al. (2016) to construct a Bayesian coreset is as follows. First, compute the *sensitivity* σ_n of each data point,

$$\sigma_n := \sup_{\theta\in\Theta} \left| \frac{\mathcal{L}_n(\theta)}{\mathcal{L}(\theta)} \right|, \tag{2.2}$$

and then subsample the data set by taking M independent draws with probability proportional to σ_n (resulting in a coreset of size $\leq M$) via

$$\sigma := \sum_{n=1}^N \sigma_n \qquad (M_1, \ldots, M_N) \sim \mathsf{Multi}\left(M, \left(\frac{\sigma_n}{\sigma}\right)_{n=1}^N\right) \qquad W_n = \frac{\sigma}{\sigma_n}\frac{M_n}{M}. \tag{2.3}$$

Since $\mathbb{E}[W_n] = 1$, we have that $\mathbb{E}[\mathcal{L}(W,\theta)] = \mathcal{L}(\theta)$, and we expect that $\mathcal{L}(W,\theta) \to \mathcal{L}(\theta)$ in some sense as M increases. This is indeed the case; Braverman et al. (2016); Feldman and Langberg (2011) showed that with high probability, the coreset likelihood $\mathcal{L}(W,\theta)$ satisfies Eq. (2.1) with $\epsilon^2 = O\left(\frac{1}{M}\right)$, and Huggins et al. (2016) extended this result to the case of Bayesian coresets in the setting of logistic regression. Typically, exact computation of the sensitivities σ_n is not tractable, so upper bounds are used instead (Huggins et al., 2016).

3. Coresets as Sparse Vector Sum Approximation

This section develops an intuitive perspective of Bayesian coresets as sparse vector sum approximation under a uniform norm, and draws on this perspective to uncover the limitations of the framework and avenues for extension. Consider the vector space of functions $g : \Theta \to \mathbb{R}$ with bounded uniform norm weighted by the total log-likelihood $\mathcal{L}(\theta)$,

$$\|g\| := \sup_{\theta \in \Theta} \left| \frac{g(\theta)}{\mathcal{L}(\theta)} \right|. \tag{3.1}$$

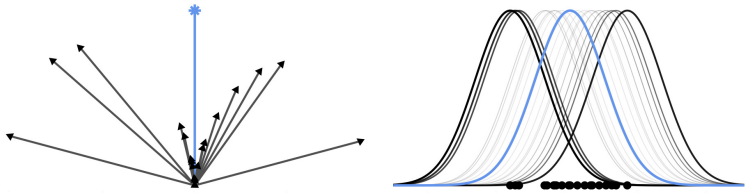
In this space, the data log-likelihood functions $\mathcal{L}_n(\theta)$ have vectors $\mathcal{L}_n$ with norm $\sigma_n := \|\mathcal{L}_n\|$ as defined in Eq. (2.2), the total log-likelihood has vector $\mathcal{L} := \sum_{n=1}^N \mathcal{L}_n$, and the coreset guarantee in Eq. (2.1) corresponds to approximation of $\mathcal{L}$ with the vector $\mathcal{L}(w) := \sum_{n=1}^N w_n \mathcal{L}_n$ under the vector norm with error at most ϵ, i.e. $\|\mathcal{L}(w) - \mathcal{L}\| \leq \epsilon$. Given this formulation, we can write the problem of constructing the best coreset of size M as the minimization of approximation error subject to a constraint on the number of nonzero entries in w,

$$\min_{w \in \mathbb{R}^N} \quad \|\mathcal{L}(w) - \mathcal{L}\|^2 \quad \text{s.t.} \quad w \geq 0, \quad \sum_{n=1}^N \mathbb{1}\left[w_n > 0\right] \leq M. \tag{3.2}$$

Eq. (3.2) is a convex optimization with binary constraints, and thus is difficult to solve efficiently in general; we are forced to use approximate methods. The uniform Bayesian coresets framework provides one such approximate method, where $\mathcal{L}/N$ is viewed as the expectation of a uniformly random subsample of $(\mathcal{L}_n)_{n=1}^N$, and importance sampling is used to reduce the expected error of the estimate. Choosing importance probabilities proportional to $\sigma_n = \|\mathcal{L}_n\|$ results in a high-probability bound on approximation error given below in Theorem 3.2. The proof of Theorem 3.2 in Appendix A is much simpler than similar results available in the literature (Feldman and Langberg, 2011; Braverman et al., 2016; Huggins et al., 2016) due to the present vector space formulation. Theorem 3.2 depends on two constants (σ and $\overline{\eta}$) that capture important aspects of the geometry of the optimization problem:

$$\sigma_n := \|\mathcal{L}_n\| \qquad \sigma := \sum_{n=1}^N \sigma_n \qquad \overline{\eta}^2 := \max_{n,m \in [N]} \left\| \frac{\mathcal{L}_n}{\sigma_n} - \frac{\mathcal{L}_m}{\sigma_m} \right\|^2, \tag{3.3}$$

where $[N] := \{1, 2, \ldots, N\}$. The quantity $\sigma \geq 0$ captures the scale of the problem; all error guarantees on $\|\mathcal{L}(w) - \mathcal{L}\|$ should be roughly linearly proportional to σ. The quantity $0 \leq \overline{\eta} \leq 2$ captures how well-aligned the vectors $(\mathcal{L}_n)_{n=1}^N$ are, and thus the inherent difficulty of approximating $\mathcal{L}$ with a sparse weighted subset $\mathcal{L}(w)$. For example, if all vectors are aligned then $\overline{\eta} = 0$, and the problem is trivial since we can achieve 0 error with a single scaled vector $\mathcal{L}_n$. Theorem 3.2 also depends on an approximate notion of the dimension of the span of the log-likelihood vectors $(\mathcal{L}_n)_{n=1}^N$, given by Definition 3.1. Note in particular that the approximate dimension of a set of vectors in $\mathbb{R}^d$ is at most d, corresponding to the usual notion of dimension in this setting.



(a) (b)

Figure 1: (1a): The sparse vector approximation problem, depicting the sum $\mathcal{L}$ in blue and the vectors $\mathcal{L}_n$ in grey. (1b): Uniform Bayesian coresets behavior on the simple exercise of learning a Gaussian mean. Depicted are data likelihoods in black, scaled posterior density in blue, and data as black scatter points. The sensitivity of each datapoint is indicated by the line thickness of the likelihood—a thicker, darker line denotes higher sensitivity, while thinner, lighter lines denote lower sensitivity.

Definition 3.1. The *approximate dimension* $\dim\left(u_n\right)_{n=1}^N$ of N vectors in a normed vector space is the minimum value of $d \in \mathbb{N}$ such that all vectors u_n can be approximated using linear combinations of a set of d unit vectors $(v_j)_{j=1}^d$, $\|v_j\| = 1$:

$$\forall\, n \in [N],\ \exists\, \alpha_n \in [-1,1]^d \text{ s.t. } \left\| \frac{u_n}{\|u_n\|} - \sum_{j=1}^d \alpha_{nj} v_j \right\| \leq \frac{d}{\sqrt{N}}.$$

Theorem 3.2. *Fix any $\delta \in (0,1)$. With probability $\geq 1 - \delta$, the output of the uniform coreset construction algorithm in Eq. (2.3) satisfies*

$$\|\mathcal{L}(W) - \mathcal{L}\| \leq \frac{\sigma}{\sqrt{M}} \left(\frac{3}{2} \dim\left(\mathcal{L}_n\right)_{n=1}^N + \overline{\eta} \sqrt{2 \log \frac{1}{\delta}} \right).$$

The analysis, discussion, and algorithms presented to this point are independent of the particular choice of norm given in Eq. (3.1); one might wonder if the uniform norm used above is the best choice, or if there is another norm more suited to Bayesian inference in some way. For instance, the supremum in Eq. (3.1) can diverge in an unbounded or infinite-dimensional parameter space Θ, requiring an artificial restriction placed on the space (Huggins et al., 2016). This precludes the application to the many common models and priors that have unbounded parameter spaces, even logistic regression with full support $\Theta = \mathbb{R}^d$. The optimization objective function in Eq. (3.1) is also typically nonconvex, and finding (or bounding) the optimum is a model-specific task that is not easily automated.

Perhaps most importantly, the uniform norm lacks a sense of "directionality" as it does not correspond to an inner-product. This implies that the bound in Theorem 3.2 does not scale properly with the alignment of vectors (note how the error does not approach 0 as $\overline{\eta} \to 0$) and that it depends on the approximate dimension (which may be hard to compute). Moreover, the lack of directionality makes the coreset construction algorithm behave counterintuitively and limits its performance in a fundamental way. Fig. 1a provides

a pictorial representation of this limitation. Recall that the goal of coreset construction is to find a *sparse* weighted subset of the vectors $(\mathcal{L}_n)_{n=1}^N$ (grey) that approximates $\mathcal{L}$ (blue). In this example, there are vectors which, when scaled, could individually nearly perfectly replicate $\mathcal{L}$. But the importance sampling algorithm in Eq. (2.3) will instead tend to sample those vectors with large norm that are pointed away from $\mathcal{L}$, requiring a much larger coreset to achieve the same approximation error. This is a consequence of the lack of directionality of the uniform norm; it has no concept of the alignment of certain vectors with $\mathcal{L}$, and is forced to mitigate worst-case error by sampling those vectors with large norm. Fig. 1b shows the result of this behavior in a 1D Gaussian inference problem. In this figure, the likelihood functions of the data are depicted in black, with their uniform norm (or sensitivity) indicated by thickness and opacity. The posterior distribution is displayed in blue, with its log-density scaled by $1/N$ for clarity. The importance sampling algorithm in Eq. (2.3) will tend to sample those data that are *far away* from the posterior mean, with likelihoods that are different than the scaled posterior, despite the fact that there are data close to the mean whose likelihoods are near-perfect approximations of the scaled posterior. Using the intuition from Fig. 1, it is not difficult to construct examples where the expected error of importance sampling is arbitrarily worse than the error of the optimal coreset of size M.

4. Hilbert Coresets

It is clear that a notion of directionality of the vectors $(\mathcal{L}_n)_{n=1}^N$ is key to developing both efficient, intuitive coreset construction algorithms and theory that correctly reflects problem difficulty. Therefore, in this section we develop methods for constructing Bayesian coresets in a Hilbert space (*Hilbert coresets*), i.e., using a norm corresponding to an inner product. The notion of directionality granted by the inner product provides two major advantages over uniform coresets: coreset points can be chosen intelligently based on the residual posterior approximation error vector; and theoretical guarantees on approximation quality can directly incorporate the difficulty of the approximation problem via the alignment of log-likelihood vectors. We provide two coreset construction algorithms which take advantage of these benefits. The first method, developed in Section 4.1, is based on viewing $\mathcal{L}/N$ as the expectation of a uniformly random subsample of $(\mathcal{L}_n)_{n=1}^N$, and then using importance sampling to reduce the expected error of the estimate. The second method, developed in Section 4.2, is based on viewing the cardinality-unconstrained version of Eq. (3.2) as a quadratic optimization over an appropriately-chosen polytope, and then using the Frank–Wolfe algorithm Frank and Wolfe (1956); Guélat and Marcotte (1986); Jaggi (2013) to compute a sparse approximation to the optimum. Theoretical guarantees on posterior approximation error are provided for both. In Section 4.3, we develop streaming/distributed extensions of these methods and provide similar approximation guarantees. Note that this section treats the general case of Bayesian coreset construction with a Hilbert space norm; the selection of a particular norm and its automated computation is left to Section 5.

4.1. Coreset Construction via Importance Sampling

Taking inspiration from the uniform Bayesian coreset construction algorithm, the first Hilbert coreset construction method, Algorithm 1, involves i.i.d. sampling from the vectors $(\mathcal{L}_n)_{n=1}^N$

Algorithm 1 IS: Hilbert coresets via importance sampling

Require: $(\mathcal{L}_n)_{n=1}^N$, M, $\|\cdot\|$
- $\forall n \in [N]$ $\sigma_n \leftarrow \|\mathcal{L}_n\|$, and $\sigma \leftarrow \sum_{n=1}^N \sigma_n$ ▷ compute norms
- $(M_1, \dots, M_N) \leftarrow \mathsf{Multi}\left(M, (\frac{\sigma_n}{\sigma})_{n=1}^N\right)$ ▷ subsample the data
- $W_n \leftarrow \frac{\sigma}{\sigma_n}\frac{M_n}{M}$ for $n \in [N]$ ▷ reweight the subsample
- **return** W

with probabilities $(p_n)_{n=1}^N$ and reweighting the subsample. In contrast to the case of the weighted uniform norm in Eq. (3.1), the choice $p_n \propto \sigma_n$ exactly minimizes the expected squared coreset error under a Hilbert norm (see Eq. (A.2) in Appendix A), yielding

$$\mathbb{E}\left[\|\mathcal{L}(W) - \mathcal{L}\|^2\right] = \frac{\sigma^2\eta^2}{M} \qquad \eta^2 := 1 - \frac{\|\mathcal{L}\|^2}{\sigma^2}, \tag{4.1}$$

where $0 \leq \eta \leq 1$, similar to $\overline{\eta}$, captures how well-aligned the vectors $(\mathcal{L}_n)_{n=1}^N$ are. However, in a Hilbert space η is a tighter constant: $\eta \leq \overline{\eta}/\sqrt{2}$ by Lemma A.4. Theorem 4.1, whose proof in Appendix A relies on standard martingale concentration inequalities, provides a high-probability guarantee on the quality of the output approximation. This result depends on $\overline{\eta}$ from Eq. (3.3) and η from Eq. (4.1).

Theorem 4.1. *Fix any $\delta \in (0,1)$. With probability $\geq 1-\delta$, the output W of Algorithm 1 satisfies*

$$\|\mathcal{L}(W) - \mathcal{L}\| \leq \frac{\sigma}{\sqrt{M}}\left(\eta + \eta_M\sqrt{2\log\frac{1}{\delta}}\right)$$

where

$$\eta_M := \min\left(\overline{\eta}, \eta\sqrt{\frac{2M\eta^2}{\overline{\eta}^2\log\frac{1}{\delta}}}\, H^{-1}\left(\frac{\overline{\eta}^2\log\frac{1}{\delta}}{2M\eta^2}\right)\right)$$
$$H(y) := (1+y)\log(1+y) - y.$$

In contrast to Theorem 3.2, Theorem 4.1 takes advantage of the inner product to incorporate a notion of problem difficulty into the bound. For example, since $H(y) \sim y^2$ as $y \to 0$, we have that $H^{-1}(y) \sim \sqrt{y}$ and so $\lim_{y\to 0}\sqrt{y^{-1}}H^{-1}(y) = 1$. Combined with the fact that $\eta \leq \overline{\eta}$, we have $\lim_{M\to\infty}\eta_M = \eta$, and so the bound in Theorem 4.1 is asymptotically equivalent to $\frac{\sigma\eta}{\sqrt{M}}\left(1+\sqrt{2\log\frac{1}{\delta}}\right)$ as $M \to \infty$. Given that importance sampling can only improve convergence over uniformly random subsampling by a constant, this constant reduction is significant. Note that Theorem 4.1, in conjunction with the fact that $\eta \leq \overline{\eta}$, immediately implies the simpler result in Corollary 4.2.

Corollary 4.2. *Fix any $\delta \in (0,1)$. With probability $\geq 1-\delta$, the output W of Algorithm 1 satisfies*

$$\|\mathcal{L}(W) - \mathcal{L}\| \leq \frac{\sigma\overline{\eta}}{\sqrt{M}}\left(1+\sqrt{2\log\frac{1}{\delta}}\right).$$

4.2. Coreset Construction via Frank–Wolfe

The major advantages of Algorithm 1 are its simplicity and sole requirement of computing the norms $(\|\mathcal{L}_n\|)_{n=1}^N$. Like the original uniform Bayesian coreset algorithm in Eq. (2.3), however, it does not take into account the residual error in the coreset approximation in order to choose new samples intelligently. The second Hilbert coreset construction method, Algorithm 2, takes advantage of the directionality of the Hilbert norm to incrementally build the coreset by selecting vectors aligned with the "direction of greatest improvement."

The development of Algorithm 2 involves two major steps. First, we replace the cardinality constraint on w in Eq. (3.2) with a polytope constraint:

$$\min_{w\in\mathbb{R}^N} \quad (w-1)^T K(w-1) \quad \text{s.t.} \quad w \geq 0, \quad \sum_{n=1}^N \sigma_n w_n = \sigma, \tag{4.2}$$

where $K \in \mathbb{R}^{N\times N}$ is a kernel matrix defined by $K_{ij} := \langle \mathcal{L}_i, \mathcal{L}_j\rangle$, and we take advantage of the Hilbert norm to rewrite $\|\mathcal{L}(w) - \mathcal{L}\|^2 = (w-1)^T K(w-1)$. The polytope is designed to contain the point $w = 1 := [1, 1, \ldots, 1]^T \in \mathbb{R}^N$—which is optimal with cost 0 since $\mathcal{L}(1) = \mathcal{L}$—and have vertices $\frac{\sigma}{\sigma_n} 1_n$ for $n \in [N]$, where 1_n is the n^{th} coordinate unit vector. Next, taking inspiration from the large-scale optimization literature Frank and Wolfe (1956); Guélat and Marcotte (1986); Jaggi (2013); Lacoste-Julien and Jaggi (2015); Clarkson (2010); Reddi et al. (2016); Balasubramanian and Ghadimi (2018); Hazan and Luo (2016), we solve the convex optimization in Eq. (4.2) using the Frank–Wolfe algorithm (Frank and Wolfe, 1956). Frank–Wolfe is an iterative algorithm for solving convex optimization problems of the form $\min_{x\in\mathcal{D}} g(x)$, where each iteration has three steps: 1) given the t^{th} iterate x_t, we first find a search direction $d_t = s_t - x_t$ by solving the linear program $s_t = \arg\min_{s\in\mathcal{D}} \nabla f(x_t)^T s$; 2) we find a step size by solving the 1-dimensional optimization $\gamma_t = \arg\min_{\gamma\in[0,1]} f(x_t + \gamma d_t)$; and 3) we update $x_{t+1} \leftarrow x_t + \gamma_t d_t$. In Eq. (4.2), we are optimizing a convex objective over a polytope, so the linear optimization can be solved by searching over all vertices of the polytope. And since we designed the polytope such that its vertices each have a single nonzero component, the algorithm adds at most a single data point to the coreset at each iteration; after initialization followed by $M-1$ iterations, this produces a coreset of size $\leq M$.

We initialize w_0 to the vertex most aligned with $\mathcal{L}$, i.e.

$$w_0 = \frac{\sigma}{\sigma_{f_0}} 1_{f_0} \quad \text{where} \quad f_0 = \arg\max_{n\in[N]} \left\langle \mathcal{L}, \frac{1}{\sigma_n}\mathcal{L}_n \right\rangle. \tag{4.3}$$

Let w_t be the iterate at step t. The gradient of the cost is $2K(w_t - 1)$, and we are solving a convex optimization on a polytope, so the Frank–Wolfe direction may be computed by searching over its vertices:

$$d_t := \frac{\sigma}{\sigma_{f_t}} 1_{f_t} - w_t \quad \text{where} \quad f_t = \arg\max_{n\in[N]} \left\langle \mathcal{L} - \mathcal{L}(w_t), \frac{1}{\sigma_n}\mathcal{L}_n \right\rangle. \tag{4.4}$$

The Frank–Wolfe algorithm applied to Eq. (4.2) thus corresponds to a simple greedy approach in which we select the vector $\mathcal{L}_{f_t}$ with direction most aligned with the residual error $\mathcal{L} - \mathcal{L}(w_t)$. We perform line search to update $w_{t+1} = w_t + \gamma d_t$ for some $\gamma \in [0, 1]$. Since the objective is

Algorithm 2 FW: Hilbert coresets via Frank–Wolfe

Require: $(\mathcal{L}_n)_{n=1}^N$, M, $\langle\cdot,\cdot\rangle$

$\forall n \in [N]$ $\sigma_n \leftarrow \sqrt{\langle \mathcal{L}_n, \mathcal{L}_n\rangle}$, and $\sigma \leftarrow \sum_{n=1}^N \sigma_n$ ▷ compute norms

$f \leftarrow \arg\max_{n\in[N]} \left\langle \mathcal{L}, \frac{1}{\sigma_n}\mathcal{L}_n \right\rangle$ ▷ greedy initial vertex f selection

$w \leftarrow \frac{\sigma}{\sigma_f} 1_f$ ▷ initialize w with full weight on f

for $t \in \{1, \ldots, M-1\}$ **do**

$f \leftarrow \arg\max_{n\in[N]} \left\langle \mathcal{L} - \mathcal{L}(w), \frac{1}{\sigma_n}\mathcal{L}_n \right\rangle$ ▷ find the FW vertex index f

$\gamma \leftarrow \frac{\left\langle \frac{\sigma}{\sigma_f}\mathcal{L}_f - \mathcal{L}(w), \mathcal{L} - \mathcal{L}(w) \right\rangle}{\left\langle \frac{\sigma}{\sigma_f}\mathcal{L}_f - \mathcal{L}(w), \frac{\sigma}{\sigma_f}\mathcal{L}_f - \mathcal{L}(w) \right\rangle}$ ▷ closed-form line search for step size γ

$w \leftarrow (1-\gamma)w + \gamma\frac{\sigma}{\sigma_f} 1_f$ ▷ add/reweight data point f in coreset

end for

return w

quadratic, the exact solution for unconstrained line search is available in closed form per Eq. (4.5); Lemma 4.3 shows that this is actually the solution to constrained line search in $\gamma \in [0, 1]$, ensuring that w_{t+1} remains feasible.

$$w_{t+1} = w_t + \gamma_t d_t \quad \text{where} \quad \gamma_t = \frac{\left\langle \frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t} - \mathcal{L}(w_t), \mathcal{L} - \mathcal{L}(w_t) \right\rangle}{\left\langle \frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t} - \mathcal{L}(w_t), \frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t} - \mathcal{L}(w_t) \right\rangle}. \tag{4.5}$$

Lemma 4.3. *For all $t \in \mathbb{N}$, $\gamma_t \in [0, 1]$.*

Theorem 4.4 below provides a guarantee on the quality of the approximation output by Algorithm 2 using the combination of the initialization in Eq. (4.3) and exact line search in Eq. (4.5). This result depends on the constants $\overline{\eta}$ from Eq. (3.3), η from Eq. (4.1), and ν, defined by

$$\nu^2 := 1 - \frac{r^2}{\sigma^2\overline{\eta}^2},$$

where r is the distance from $\mathcal{L}$ to the nearest boundary of the convex hull of $\{\sigma\mathcal{L}_n/\sigma_n\}_{n=1}^N$. Since $\mathcal{L}$ is in the relative interior of this convex hull by Lemma A.5, we are guaranteed that $\nu < 1$. The proof of Theorem 4.4 in Appendix A relies on a technique from Guélat and Marcotte (1986, Theorem 2) and a novel bound on the logistic equation.

Theorem 4.4. *The output w of Algorithm 2 satisfies*

$$\|\mathcal{L}(w) - \mathcal{L}\| \leq \frac{\sigma\eta\overline{\eta}\nu}{\sqrt{\overline{\eta}^2\nu^{-2(M-2)} + \eta^2(M-1)}} \leq \frac{\sigma\overline{\eta}}{\sqrt{M}}.$$

In contrast to previous convergence analyses of Frank–Wolfe optimization, Theorem 4.4 exploits the quadratic objective and exact line search to capture both the logarithmic $1/\sqrt{M}$ convergence rate for small values of M, and the linear ν^M rate for large M. Alternatively,

one can remove the computational cost of computing the exact line search via Eq. (4.5) by simply setting $\gamma_t = \frac{2}{3t+4}$. In this case, Theorem 4.4 is replaced with the weaker result (see the note at the end of the proof of Theorem 4.4 in Appendix A)

$$\|\mathcal{L}(w) - \mathcal{L}\| \leq \frac{2\sigma\overline{\eta}}{\sqrt{3M+1}}.$$

4.3. Distributed Coreset Construction

An advantage of Hilbert coresets—and coresets in general—is that they apply to streaming and distributed data with little modification, and retain their theoretical guarantees. In particular, if the data set $(y_n)_{n=1}^N$ is distributed (not necessarily evenly) among C processors, and either Algorithm 1 or Algorithm 2 are run for M iterations on each processor, the resulting merged coreset of size $\leq MC$ has an error guarantee given by Corollary 4.5 or Corollary 4.6. Note that the weights from each distributed coreset are not modified when merging. These results both follow from Theorems 4.1 and 4.4 with straightforward usage of the triangle inequality, the union bound, and the fact that $\overline{\eta}$ for each subset is bounded above by $\overline{\eta}$ for the full data set.

Corollary 4.5. *Fix any $\delta \in (0,1)$. With probability $\geq 1-\delta$, the coreset constructed by running Algorithm 1 on C nodes and merging the result satisfies*

$$\|\mathcal{L}(w) - \mathcal{L}\| \leq \frac{\sigma\overline{\eta}}{\sqrt{M}}\left(1 + \sqrt{2\log\frac{C}{\delta}}\right).$$

Corollary 4.6. *The coreset constructed by running Algorithm 2 on C nodes and merging the results satisfies*

$$\|\mathcal{L}(w) - \mathcal{L}\| \leq \frac{\sigma\overline{\eta}}{\sqrt{M}}.$$

5. Norms and Random Projection

The algorithms and theory in Section 4 address the scalability and performance of Bayesian coreset construction, but are specified for an arbitrary Hilbert norm; it remains to choose a norm suitable for automated Bayesian posterior approximation. There are two main desiderata for such a norm: it should be a good indicator of posterior discrepancy, and it should be efficiently computable or approximable in such a way that makes Algorithms 1 and 2 efficient for large N, i.e., $O(N)$ time complexity. To address the desideratum that the norm is an indicator of posterior discrepancy, we propose the use of one of two Hilbert norms. It should, however, be noted that these are simply reasonable suggestions, and other Hilbert norms could certainly be used in the algorithms set out in Section 4. The first candidate is the expectation of the squared 2-norm difference between the log-likelihood gradients under a weighting distribution $\hat{\pi}$,

$$\|\mathcal{L}(w) - \mathcal{L}\|_{\hat{\pi},F}^2 := \mathbb{E}_{\hat{\pi}}\left[\|\nabla\mathcal{L}(\theta) - \nabla\mathcal{L}(w,\theta)\|_2^2\right], \tag{5.1}$$

where the weighting distribution $\hat{\pi}$ has the same support as the posterior π. This norm is a weighted version of the Fisher information distance (Johnson and Barron, 2004). The inner product induced by this norm is defined by

$$\langle\mathcal{L}_n, \mathcal{L}_m\rangle_{\hat{\pi},F} := \mathbb{E}_{\hat{\pi}}\left[\nabla\mathcal{L}_n(\theta)^T\nabla\mathcal{L}_m(\theta)\right]. \tag{5.2}$$

Although this norm has a connection to previously known discrepancies between probability distributions, it does require that the likelihoods are differentiable. One could instead employ a simple weighted L^2 norm on the log-likelihoods, given by

$$\|\mathcal{L}(w) - \mathcal{L}\|_{\hat{\pi},2} := \mathbb{E}_{\hat{\pi}}\left[\left(\mathcal{L}(\theta) - \mathcal{L}(w,\theta)\right)^2\right] \tag{5.3}$$

with induced inner product

$$\langle\mathcal{L}_n, \mathcal{L}_m\rangle_{\hat{\pi},2} := \mathbb{E}_{\hat{\pi}}\left[\mathcal{L}_n(\theta)\mathcal{L}_m(\theta)\right]. \tag{5.4}$$

In both cases, the weighting distribution $\hat{\pi}$ would ideally be chosen equal to π to emphasize discrepancies that are in regions of high posterior mass. Though we do not have access to the true posterior without incurring significant computational cost, there are many practical options for setting $\hat{\pi}$, including: the Laplace approximation (Bishop, 2006, Section 4.4), a posterior based on approximate sufficient statistics (Huggins et al., 2017), a discrete distribution based on samples from an MCMC algorithm run on a small random subsample of the data, the prior, independent posterior conditionals (see Eq. (7.1) in Section 7.3), or any other reasonable method for finding a low-cost posterior approximation. This requirement of a low-cost approximation is not unusual, as previous coreset formulations have required similar preprocessing to compute sensitivities, e.g., a k-clustering of the data (Huggins et al., 2016; Lucic et al., 2016; Braverman et al., 2016). We leave the general purpose selection of a weighting function $\hat{\pi}$ for future work.

The two suggested norms $\|\cdot\|_{\hat{\pi},2/F}$ often do not admit exact closed-form evaluation due to the intractable expectations in Eqs. (5.2) and (5.4). Even if closed-form expressions are available, Algorithm 2 is computationally intractable when we only have access to inner products between pairs of individual log-likelihoods $\mathcal{L}_n$, $\mathcal{L}_m$, since obtaining the Frank–Wolfe direction involves the $O(N^2)$ computation $\arg\max_{n\in[N]}\sum_{m=1}^N\langle\mathcal{L}_m, \mathcal{L}_n/\sigma_n\rangle$. Further, the analytic evaluation of expectations is a model- (and $\hat{\pi}$-) specific procedure that cannot be easily automated. To both address these issues and automate Hilbert coreset construction, we use *random features* (Rahimi and Recht, 2007), i.e. a random projection of the vectors $(\mathcal{L}_n)_{n=1}^N$ into a finite-dimensional vector space using samples from $\hat{\pi}$. For the weighted Fisher information inner product in Eq. (5.2), we approximate $\langle\mathcal{L}_n, \mathcal{L}_m\rangle_{\hat{\pi},F}$ with an unbiased estimate given by

$$(d_j)_{j=1}^J \overset{\text{i.i.d.}}{\sim} \mathsf{Unif}(\{1,\ldots,D\}) \quad (\mu_j)_{j=1}^J \overset{\text{i.i.d.}}{\sim} \hat{\pi}$$

$$\langle\mathcal{L}_n, \mathcal{L}_m\rangle_{\hat{\pi},F} \approx \frac{D}{J}\sum_{j=1}^J (\nabla\mathcal{L}_n(\mu_j))_{d_j}(\nabla\mathcal{L}_m(\mu_j))_{d_j},$$

where subscripts indicate the selection of a component of a vector. If we define the J-dimensional vector

$$\hat{\mathcal{L}}_n := \sqrt{\frac{D}{J}}\left[(\nabla\mathcal{L}_n(\mu_1))_{d_1}, (\nabla\mathcal{L}_n(\mu_2))_{d_2}, \ldots, (\nabla\mathcal{L}_n(\mu_J))_{d_J}\right]^T, \tag{5.5}$$

Algorithm 3 Bayesian Hilbert coresets with random projection

Require: $(\mathcal{L}_n)_{n=1}^N$, $\hat{\pi}$, M, J
▷ sample feature points and gradient dimension indices
$(\mu_j)_{j=1}^J \overset{\text{i.i.d.}}{\sim} \hat{\pi}$, $(d_j)_{j=1}^J \overset{\text{i.i.d.}}{\sim} \mathsf{Unif}(\{1,\dots,D\})$
▷ construct the random projection using one of the norms from Eqs. (5.5) and (5.6)
If $\|\cdot\|_{\hat{\pi},F}$: $\forall n \in [N]$, $v_n \leftarrow \sqrt{D/J}\,[(\nabla\mathcal{L}_n(\mu_1))_{d_1},\dots,(\nabla\mathcal{L}_n(\mu_J))_{d_J}]^T$
If $\|\cdot\|_{\hat{\pi},2}$: $\forall n \in [N]$, $v_n \leftarrow \sqrt{1/J}\,[\mathcal{L}_n(\mu_1),\dots,\mathcal{L}_n(\mu_J)]^T$
▷ return the coreset constructed using random feature vectors
return $\mathrm{FW}\left((v_n)_{n=1}^N, M, (\cdot)^T(\cdot)\right)$ or $\mathrm{IS}\left((v_n)_{n=1}^N, M, \|\cdot\|_2\right)$

we have that for all $n, m \in [N]$,

$$\langle \mathcal{L}_n, \mathcal{L}_m \rangle_{\hat{\pi},F} \approx \hat{\mathcal{L}}_n^T \hat{\mathcal{L}}_m.$$

Therefore, for $n \in [N]$, $\hat{\mathcal{L}}_n$ serves as a random finite-dimensional projection of $\mathcal{L}_n$ that can be used in Algorithms 1 and 2. Likewise, for the weighted L^2 inner product in Eq. (5.4), we have

$$(\mu_j)_{j=1}^J \overset{\text{i.i.d.}}{\sim} \hat{\pi} \quad \langle \mathcal{L}_n, \mathcal{L}_m \rangle_{\hat{\pi},2} \approx \frac{1}{J}\sum_{j=1}^J \mathcal{L}_n(\mu_j)\mathcal{L}_m(\mu_j),$$

and so defining

$$\hat{\mathcal{L}}_n := \sqrt{\frac{1}{J}}\,[\mathcal{L}_n(\mu_1), \mathcal{L}_n(\mu_2), \dots, \mathcal{L}_n(\mu_J)]^T, \tag{5.6}$$

we have that for all $n, m \in [N]$,

$$\langle \mathcal{L}_n, \mathcal{L}_m \rangle_{\hat{\pi},2} \approx \hat{\mathcal{L}}_n^T \hat{\mathcal{L}}_m,$$

and once again $\forall n \in [N]$, $\hat{\mathcal{L}}_n$ serves as a finite-dimensional approximation of $\mathcal{L}_n$. The construction of the random projections is both easily automated and enables the efficient computation of inner products with vector sums. For example, to obtain the Frank–Wolfe direction, rather than computing $\arg\max_{n\in[N]} \sum_{m=1}^N \langle \mathcal{L}_m, \mathcal{L}_n/\sigma_n \rangle$, we can simply compute $\hat{\mathcal{L}} = \sum_{n=1}^N \hat{\mathcal{L}}_n$ in $O(NJ)$ time once at the start of the algorithm and then $\arg\max_{n\in[N]} \frac{1}{\sigma_n}\hat{\mathcal{L}}^T\hat{\mathcal{L}}_n$ in $O(NJ)$ time at each iteration. Further, since $\hat{\mathcal{L}}_n^T\hat{\mathcal{L}}_m$ is an unbiased estimate of $\langle \mathcal{L}_n, \mathcal{L}_m \rangle_{\hat{\pi},2/F}$, we expect the error of the approximation to decrease with the random finite projection dimension J. Theorem 5.2 (below), whose proof may be found in Appendix A, shows that under reasonable conditions this is indeed the case: the difference between the true output error and random projection output error shrinks as J increases. Note that Theorem 5.2 is quite loose, due to its reliance on a max-norm quadratic form upper bound.

Definition 5.1. (Boucheron et al., 2013, p. 24) A random variable X is *sub-Gaussian with constant* ξ^2 if

$$\forall \lambda \in \mathbb{R}, \quad \mathbb{E}\left[e^{\lambda X}\right] \le e^{\frac{\lambda^2\xi^2}{2}}.$$

Theorem 5.2. *Let $\mu \sim \hat{\pi}$, $d \sim \mathsf{Unif}(\{1,\dots,D\})$, and suppose $D\nabla\mathcal{L}_n(\mu)_d\nabla\mathcal{L}_m(\mu)_d$ (given $\|\cdot\|_{\hat{\pi},F}$) or $\mathcal{L}_n(\mu)\mathcal{L}_m(\mu)$ (given $\|\cdot\|_{\hat{\pi},2}$) is sub-Gaussian with constant ξ^2. Fix any $\delta \in (0,1)$. With probability $\ge 1-\delta$, the output of Algorithm 3 satisfies*

$$\|\mathcal{L} - \mathcal{L}(w)\|_{\hat{\pi},2/F}^2 \le \|\hat{\mathcal{L}} - \hat{\mathcal{L}}(w)\|_2^2 + \|w-1\|_1^2\sqrt{\frac{2\xi^2}{J}\log\frac{2N^2}{\delta}}.$$

6. Synthetic Evaluation

In this section, we compare Hilbert coresets to uniform coresets and uniformly random subsampling in a synthetic setting where expressions for the exact and coreset posteriors, along with the KL-divergence between them, are available in closed-form. In particular, the methods are used to perform posterior inference for the unknown mean $\mu \sim \mathcal{N}(\mu_0, I)$ of a 2-dimensional multivariate normal distribution with known covariance I from a collection of $N = 1{,}000$ i.i.d. observations $(y_n)_{n=1}^N$:

$$\mu \sim \mathcal{N}(\mu_0, I) \quad (y_n)_{n=1}^N \,|\, \mu \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu, I).$$

6.1. Methods

We ran 1000 trials of data generation followed by uniformly random subsampling (`Rand`), uniform coresets (`Unif`), Hilbert importance sampling (`IS`), and Hilbert Frank–Wolfe (`FW`). For the two Hilbert coreset constructions, we used the weighted Fisher information distance in Eq. (5.1). In this simple setting, the exact posterior distribution is a multivariate Gaussian with mean μ_π and covariance Σ_π given by

$$\mu \,|\, (y_n)_{n=1}^N \sim \mathcal{N}(\mu_\pi, \Sigma_\pi) \qquad \Sigma_\pi = \frac{1}{1+N}I \qquad \mu_\pi = \Sigma_\pi\left(\mu_0 + \sum_{n=1}^N y_n\right).$$

For uniform coreset construction, we subsampled the data set as per Eq. (2.3), where the sensitivity of y_n (see Appendix C for the derivation) is given by

$$\sigma_n = \frac{1}{N}\left(1 + \frac{(y_n-\bar{y})^T(y_n-\bar{y})}{\frac{1}{N}\sum_{m=1}^N y_m^T y_m - \bar{y}^T\bar{y}}\right), \quad \bar{y} := \frac{1}{N}\sum_{n=1}^N y_n. \tag{6.1}$$

This resulted in a multivariate Gaussian uniform coreset posterior approximation with mean $\hat{\mu}_\pi$ and covariance $\hat{\Sigma}_\pi$ given by

$$\hat{\Sigma}_\pi = \frac{1}{1+\sum_{n=1}^N W_n}I \qquad \hat{\mu}_\pi = \hat{\Sigma}_\pi\left(\mu_0 + \sum_{n=1}^N W_n y_n\right). \tag{6.2}$$

Generating a uniformly random subsample posterior approximation involved a similar technique, instead using probabilities $\frac{1}{N}$ for all $n \in [N]$. For the Hilbert coreset algorithms, we used the true posterior as the weighting distribution, i.e., $\hat{\pi} = \pi$. This was chosen to illustrate the ideal case in which the true posterior π is well-approximated by the weighting

distribution $\hat{\pi}$. Given this choice, the Fisher information distance inner product is available in closed-form:

$$\langle \mathcal{L}_n, \mathcal{L}_m \rangle_{\pi,F} = \frac{2}{1+N} + (\mu_\pi - y_n)^T (\mu_\pi - y_m). \tag{6.3}$$

Note that the norm $\|\mathcal{L}_n\|_{\pi,F}$ implied by Eq. (6.3) and the uniform sensitivity from Eq. (6.1) are functionally very similar; both scale with the squared distance from y_n to an estimate of μ. Since all the approximate posteriors are multivariate Gaussians of the form Eq. (6.2)—with weights W_n differing depending on the construction algorithm—we were able to evaluate posterior approximation quality exactly using the KL-divergence from the approximate coreset posterior $\tilde{\pi}$ to π, given by

$$\mathrm{D_{KL}}(\pi \| \tilde{\pi}) = \frac{1}{2}\left\{ \mathrm{tr}\left(\Sigma_{\tilde{\pi}}^{-1}\Sigma_\pi\right) + (\mu_{\tilde{\pi}} - \mu_\pi)^T \Sigma_{\tilde{\pi}}^{-1}(\mu_{\tilde{\pi}} - \mu_\pi) - 2 + \log\frac{|\Sigma_{\tilde{\pi}}|}{|\Sigma_\pi|}\right\}.$$

6.2. Results

The results of this test appear in Fig. 2. The visual comparison of the different coreset constructions in Fig. 2a–2d makes the advantages of Hilbert coresets constructed via Frank–Wolfe clear. As more coreset points are added, all approximate posteriors converge to the true posterior; however, the Frank–Wolfe method requires many fewer coreset points to converge on a reliable estimate. While both the Hilbert and uniform coresets subsample the data favoring those points at greater distance from the center, the Frank–Wolfe method first selects a point close to the center (whose scaled likelihood well-approximates the true posterior), and then refines its estimate with points far away from the center. This intuitive behavior results in a higher-quality approximation of the posterior across all coreset sizes. Note that the black coreset points across $M = 5$, 50, and 500 show a single trace of a coreset being constructed, while the green posterior predictive ellipses show the noise in the coreset construction across multiple runs at each fixed value of M. The quantitative results in Figs. 2e and 2f—which plot the KL-divergence between each coreset approximate posterior and the truth as the projection dimension J or the number of coreset iterations M varies—confirm the qualitative evaluations. In addition, Fig. 2e confirms the theoretical result from Theorem 4.4, i.e., that Frank–Wolfe exhibits linear convergence in this setting. Fig. 2f similarly confirms Theorem 5.2, i.e., that the posterior error of the projected Hilbert coreset converges to that of the exact Hilbert coreset as the dimension J of the random projection increases.

7. Experiments

In this section we evaluate the performance of Hilbert coresets compared with uniform coresets and uniformly random subsampling, using MCMC on the full data set as a benchmark. We test the algorithms on logistic regression, Poisson regression, and directional clustering models applied to numerous real and synthetic data sets. Based on the results of the synthetic comparison presented in Section 6, for clarity we focus the tests on comparing uniformly random subsampling to Hilbert coresets constructed using Frank–Wolfe with the weighted Fisher information distance from Eq. (5.1). Additional results on importance sampling, uniform coresets, and the weighted 2-norm from Eq. (5.3) are deferred to Appendix B.

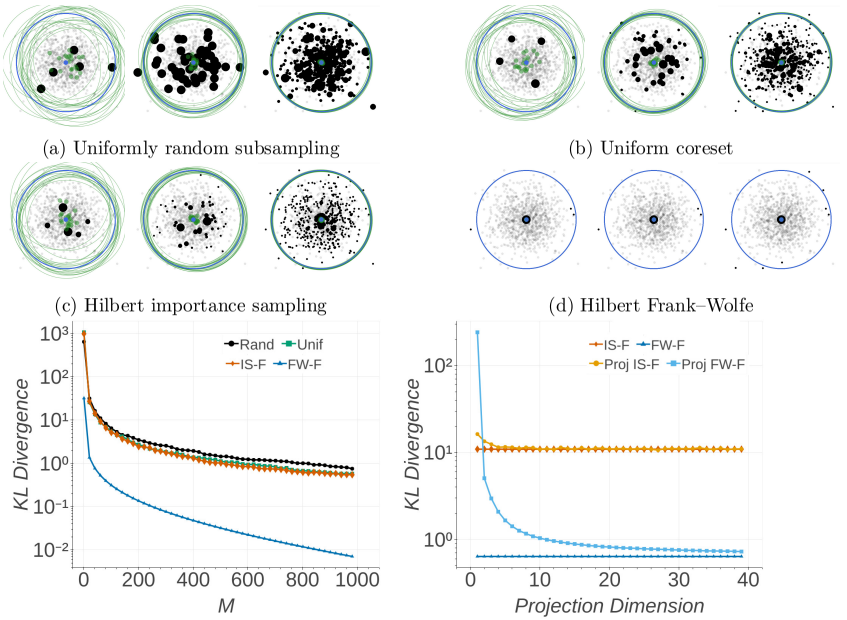


Figure 2: (2a-2d): Comparison of different coreset constructions for Gaussian inference, showing example coreset posterior predictive 3σ ellipses (green), the true data generating distribution 3σ ellipse (blue), and a single trace of coreset construction (black) for $M = 5$, 50, and 500. The radius of each coreset point indicates its weight. (2e): A comparison of approximate posteriors using exact norms versus coreset construction iterations M. (2f): A comparison of exact and projected methods versus projection dimension J, with fixed $M = 50$.

7.1. Models

In the **logistic regression** setting, we are given a set of data points $(x_n, y_n)_{n=1}^N$ each consisting of a feature $x_n \in \mathbb{R}^D$ and a label $y_n \in \{-1, 1\}$, and the goal is to predict the label of a new point given its feature. We thus seek to infer the posterior distribution of the parameter $\theta \in \mathbb{R}^{D+1}$ governing the generation of y_n given x_n via

$$\theta \sim \mathcal{N}(0, I) \qquad y_n \,|\, x_n, \theta \overset{\text{indep}}{\sim} \mathsf{Bern}\left(\frac{1}{1+e^{-z_n^T\theta}}\right) \qquad z_n := [x_n,\, 1]^T.$$

In the **Poisson regression** setting, we are given a set of data points $(x_n, y_n)_{n=1}^N$ each consisting of a feature $x_n \in \mathbb{R}^D$ and a count $y_n \in \mathbb{N}$, and the goal is to learn a relationship

between features x_n and the associated mean count. We thus seek to infer the posterior distribution of the parameter $\theta \in \mathbb{R}^D$ governing the generation of y_n given x_n via

$$\theta \sim \mathcal{N}(0, I) \qquad y_n \,|\, x_n, \theta \overset{\text{indep}}{\sim} \text{Poiss}\left(\log\left(1 + e^{\theta^T z_n}\right)\right) \qquad z_n := [x_n, 1]^T.$$

Finally, in the **directional clustering** setting, we are given a data set of points $(x_n)_{n=1}^N$ on the unit $(D-1)$-sphere, i.e. $x_n \in \mathbb{R}^D$ with $\|x_n\|_2 = 1$, and the goal is to separate them into K clusters. For this purpose we employ a *von Mises-Fisher (vMF) mixture model* (Banerjee et al., 2005). The component likelihood in this model is the von Mises-Fisher distribution $\text{vMF}(\mu, \tau)$ with concentration $\tau \in \mathbb{R}_+$ and mode $\mu \in \mathbb{R}^D$, $\|\mu\|_2 = 1$, having density

$$f_{\text{vMF}}(x; \mu, \tau) = C_D(\tau) e^{\tau x^T \mu} \qquad C_D(\tau) = \frac{\tau^{D/2-1}}{(2\pi)^{D/2} I_{D/2-1}(\tau)}$$

with support on the unit $(D-1)$-sphere $\mathbb{S}^{D-1}$, where I_p denotes the modified Bessel function of the first kind of order p. We place uniform priors on both the component modes and mixture weights, and set $\tau = 50$, resulting in the generative model

$$(\mu_k)_{k=1}^K \overset{\text{i.i.d.}}{\sim} \text{Unif}\left(\mathbb{S}^{D-1}\right) \quad (\omega_k)_{k=1}^K \sim \text{Dir}(1, \ldots, 1)$$
$$(x_n)_{n=1}^N \,|\, (\omega_k, \mu_k)_{k=1}^K \overset{\text{i.i.d.}}{\sim} \sum_{k=1}^K \omega_k \text{vMF}(\mu_k, \tau).$$

7.2. Datasets

We tested the coreset construction methods for each model on a number of data sets. For **logistic regression**, the `Synthetic` data set consisted of $N = 10{,}000$ data points (with 1,000 held out for testing) with covariate $x_n \in \mathbb{R}^2$ sampled i.i.d. from $\mathcal{N}(0, I)$, and label $y_n \in \{-1, 1\}$ generated from the logistic likelihood with parameter $\theta = [3, 3, 0]^T$. The `Phishing`[2] data set consisted of $N = 11{,}055$ data points (with 1,105 held out for testing) each with $D = 68$ features. In this data set, each covariate corresponds to the features of a website, and the goal is to predict whether or not a website is a phishing site. The `ChemReact`[3] data set consisted of $N = 26{,}733$ data points (with 2,673 held out for testing) each with $D = 10$ features. In this data set, each covariate represents the features of a chemical experiment, and the label represents whether a chemical was reactive in that experiment or not.

For **Poisson regression**, the `Synthetic` data set consisted of $N = 10{,}000$ data points (with 1,000 held out for testing) with covariate $x_n \in \mathbb{R}$ sampled i.i.d. from $\mathcal{N}(0, 1)$, and count $y_n \in \mathbb{N}$ generated from the Poisson likelihood with $\theta = [1, 0]^T$. The `BikeTrips`[4] data set consisted of $N = 17{,}386$ data points (with 1,738 held out for testing) each with $D = 8$ features. In this data set, each covariate corresponds to weather and season information for a particular hour during the time between 2011–2012, and the count is the number of bike trips taken during that hour in a bikeshare system in Washington, DC. The `AirportDelays`[5] data set consisted of $N = 7{,}580$ data points (with 758 held out for testing) each with $D = 15$ features. In this data set, each covariate corresponds to the weather information of a day during the time between 1987–2008, and the count is the number of flights leaving Boston Logan airport delayed by more than 15 minutes that day.

Finally, for **directional clustering**, the `Synthetic` data set consisted of $N = 10{,}000$ data points (with 1,000 held out for testing) generated from an equally-weighted vMF mixture with 6 components, one centered at each of the axis poles.

2. https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html
3. http://komarix.org/ac/ds/
4. http://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset
5. Airport information from http://stat-computing.org/dataexpo/2009/the-data.html, with historical weather information from https://www.wunderground.com/history/.

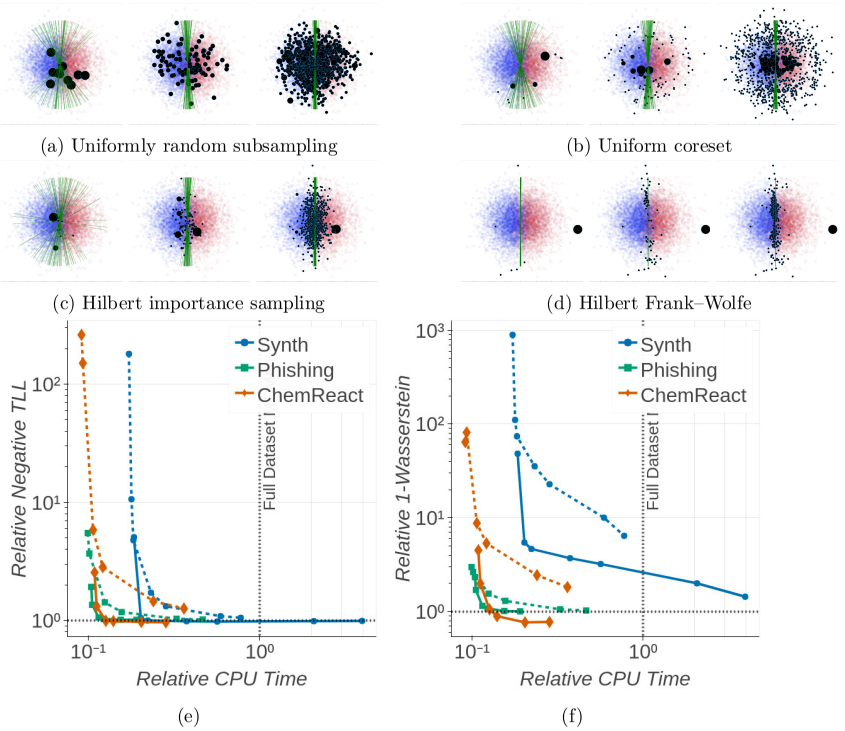
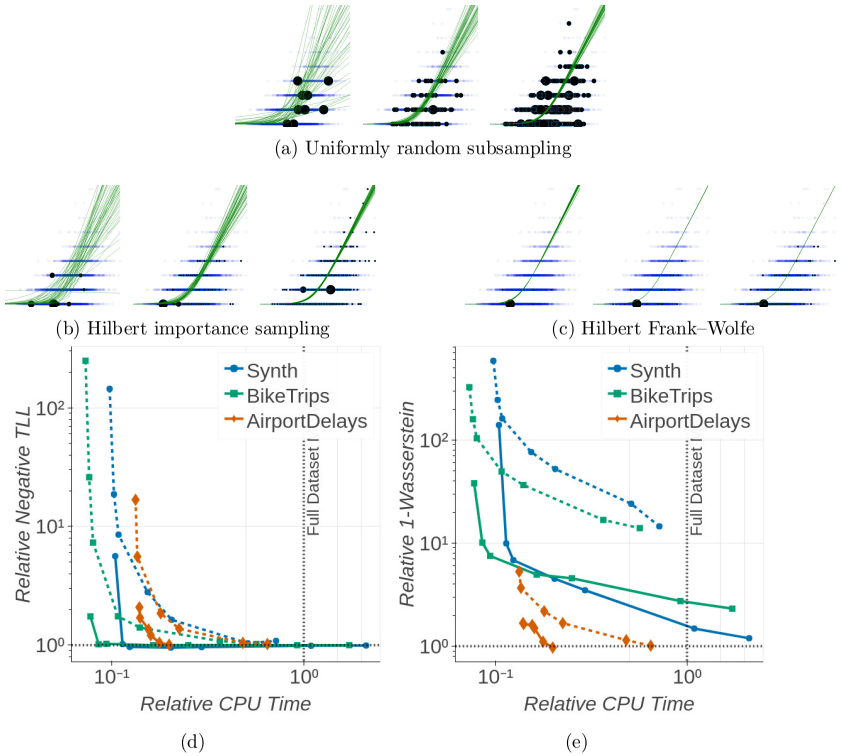


Figure 3: (3a-3d): Comparison of different coreset constructions for logistic regression on the `Synthetic` data set (with blue & red labeled data), showing example coreset posterior mean classification boundaries (green), and a single trace of coreset construction (black) for $M = 10$, 100, and 1000. The radius of each coreset point indicates its weight. (3e, 3f): A comparison of negative test log-likelihood (3e) and 1-Wasserstein distance (3f) versus computation time for Frank–Wolfe (solid) and uniform random subsampling (dashed) on the logistic regression model. Both axes are normalized using results from running MCMC on the full data set; see Section 7.3.



(a) Uniformly random subsampling

(b) Hilbert importance sampling

(c) Hilbert Frank–Wolfe

(d)

(e)

Figure 4: (4a-4c): Comparison of different coreset constructions for Poisson regression on the `Synthetic` data set, showing example coreset posterior Poisson mean curves (green), and a single trace of coreset construction (black) for $M = 10$, 100, and 1,000. The radius of each coreset point indicates its weight. (4d, 4e): A comparison of negative test log-likelihood (4d) and 1-Wasserstein distance (4e) versus computation time for Frank–Wolfe (solid) and uniform random subsampling (dashed) on the Poisson regression model. Both axes are normalized using results from running MCMC on the full data set; see Section 7.3.

7.3. Methods

We ran 50 trials of uniformly random subsampling and Hilbert Frank–Wolfe using the approximate Fisher information distance in Eq. (5.1), varying $M \in \{10, 50, 100, 500, 1{,}000, 5{,}000, 10{,}000\}$. For both **logistic regression** and **Poisson regression**, we used the Laplace approximation (Bishop, 2006, Section 4.4) as the weighting distribution $\hat{\pi}$ in the Hilbert coreset, with the random projection dimension set to $D = 500$. Posterior inference in each of the 50 trials was conducted using random walk Metropolis-Hastings with an isotropic

multivariate Gaussian proposal distribution. We simulated a total of 100,000 steps, with 50,000 warmup steps including proposal covariance adaptation with a target acceptance rate of 0.234, and thinning of the latter 50,000 by a factor of 5, yielding 10,000 posterior samples.

For **directional clustering** the weighting distribution $\hat{\pi}$ for the Hilbert coreset was constructed by finding maximum likelihood estimates of the cluster modes $(\hat{\mu}_k)_{k=1}^K$ and weights $\hat{\omega}$ using the EM algorithm, and then setting $\hat{\pi}$ to an independent product of approximate posterior conditionals,

$$\bar{x}_k := \sum_{n=1}^N z_{nk} x_n \qquad \bar{z}_k := \sum_{n=1}^N z_{nk}$$
$$\mu_k \overset{\text{indep}}{\sim} \text{vMF}\left(\frac{\bar{x}_k}{\|\bar{x}_k\|}, \tau\|\bar{x}_k\|\right) \qquad \omega \overset{\text{indep}}{\sim} \text{Dir}\left(1 + \bar{z}_1, \ldots, 1 + \bar{z}_k\right), \tag{7.1}$$

where $(z_n)_{n=1}^N$, $z_n \in \mathbb{R}_+^K$ are the smoothed cluster assignments. The random projection dimension was set to $D = 500$, and the number of clusters K was set to 6. Posterior inference in each of the 50 trials was conducted used Gibbs sampling (introducing auxiliary label variables for the data) with a total of 100,000 steps, with 50,000 warmup steps and thinning of the latter 50,000 by a factor of 5, yielding 10,000 posterior samples. Note that this approach is exact for the full data set; for the coreset constructions with weighted data, we replicate each data point by its ceiled weight, and then rescale the assignment variables to account for the fractional weight. In particular, for coreset weights $(w_n)_{n=1}^N$, we sample labels for points with $w_n > 0$ via

$$\gamma_k \propto \omega_k f_{\text{vMF}}(x_n; \mu_k, \tau) \quad z_n \overset{\text{indep}}{\sim} \text{Multi}(\texttt{ceil}(w_n), \gamma) \quad z_n \leftarrow \frac{w_n}{\texttt{ceil}(w_n)} z_n,$$

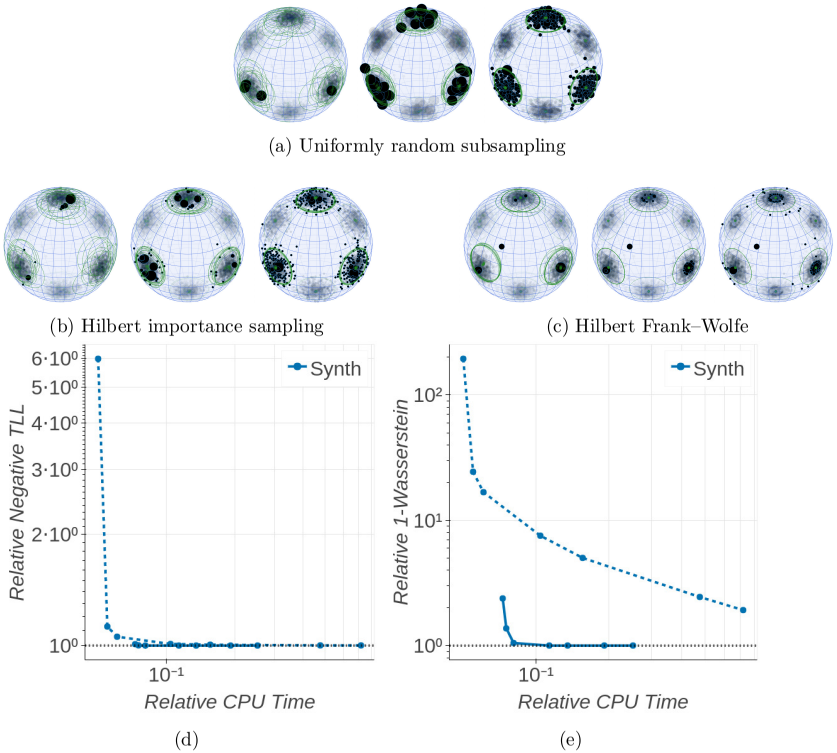
and sample the cluster centers and weights via Eq. (7.1).

For all models, we evaluate two metrics of posterior quality: negative log-likelihood on the held-out test set, averaged over posterior MCMC samples; and 1-Wasserstein distance of the posterior samples to samples obtained from running MCMC on the full data set. All negative test log-likelihood results are shifted by the maximum possible test log-likelihood and normalized by the test log-likelihood obtained from the full data set posterior. All 1-Wasserstein distance results are normalized by the median pairwise 1-Wasserstein distance between 10 trials of MCMC on the full data set. All computation times are normalized by the median computation time for MCMC on the full data set across the 10 trials. These normalizations allow the results from multiple data sets to be plotted coherently on the same axes.

We ran the same experiments described above on Hilbert importance sampling for all data sets, and uniform coresets on the logistic regression model with $a = 3$ and $K = 4$ (see Huggins et al. (2016, Sec. 4.2)). We also compared Hilbert coresets with the weighted 2-norm in Eq. (5.3) to the weighted Fisher information distance in Eq. (5.1). The results of these experiments are deferred to Appendix B for clarity.

7.4. Results and discussion

Figs. 3, 4 and 5 show the experimental results for logistic regression, Poisson regression, and directional clustering, respectively. The visual comparisons of coreset construction for



(a) Uniformly random subsampling

(b) Hilbert importance sampling

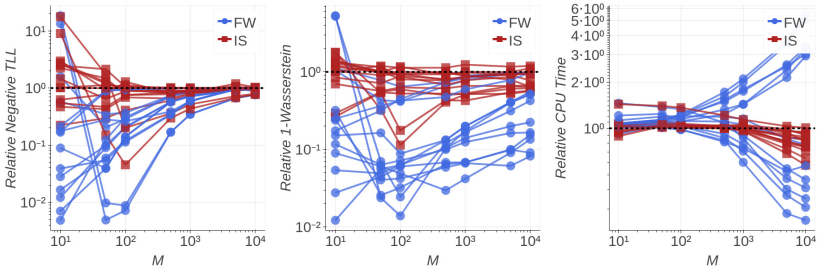
(c) Hilbert Frank–Wolfe

(d)

(e)

Figure 5: (5a-5c): Comparison of different coreset constructions for directional clustering, showing example coreset posterior mean clusters (green), and a single trace of coreset construction (black) for $M = 10$, 100, and 1,000. The radius of each coreset point indicates its weight. (5d, 5e): A comparison of negative test log-likelihood (5d) and 1-Wasserstein distance (5e) versus computation time for Frank–Wolfe (solid) and uniform random subsampling (dashed) on the directional clustering model. Both axes are normalized using results from running MCMC on the full data set; see Section 7.3.

all models mimic the results of the synthetic evaluation in Fig. 2. For all the algorithms, the approximate posterior converges to the true posterior on the full data set as more coreset points are added; and the Frank–Wolfe Hilbert coreset construction selects the most useful points incrementally, creating intuitive coresets that outperform all the other methods. For example, in the logistic regression model, the uniform coreset construction sensitivities are based on the proximity of the data to the centers of a K-clustering, and do not directly incorporate information about the classification boundary. This construction



(a) (b) (c)

Figure 6: A comparison of (6a) negative test log-likelihood, (6b) 1-Wasserstein posterior distance estimate, and (6c) computation time versus coreset construction iterations M for Frank–Wolfe (blue), importance sampling (red), and uniformly random subsampling (dashed black) across all models and data sets. All metrics are normalized to the median value for uniformly random subsampling; see Section 7.3.

therefore generally favors sampling points on the periphery of the data set and assigns high weight to those near the center. The Hilbert importance sampling algorithm, in contrast, directly considers the logistic regression problem; it favors sampling points lying along the boundary and assigns high weight to points orthogonal to it, thereby fixing the boundary plane more accurately. The Hilbert Frank–Wolfe algorithm selects a single point closely aligned with the classification boundary normal, and then refines its estimate with points near the boundary. This enables it to use far fewer coreset points to achieve a more accurate posterior estimate than the sampling-based methods. Similar statements hold for the two other models: in the Poisson regression model, the Hilbert Frank–Wolfe algorithm chooses a point closest to the true parameter and then refines its estimate using far away points; and in the directional clustering model, the Hilbert Frank–Wolfe algorithm initially selects points near the cluster centers and then refines the estimates with points in each cluster far from its center. The quantitative results demonstrate the strength of the Hilbert Frank–Wolfe coreset construction algorithm: for a given computational time budget, this algorithm provides orders of magnitude reduction in both error metrics over uniformly random subsampling. In addition, the Hilbert Frank–Wolfe coreset achieves the same negative test log-likelihood as the full data set in roughly a tenth of the computation time. These statements hold across all models and data sets considered.

Fig. 6 provides a summary of the performance of Hilbert coresets as a function of construction iterations M across all models and data sets. This demonstrates its power not only as a scalable inference method but also as a data set compression technique. For any value of M and across a wide variety of models and data sets, Hilbert coresets (both importance sampling and Frank–Wolfe-based constructions) provide a significant improvement in posterior approximation quality over uniformly random subsampling, with comparable computation time. This figure also shows a rather surprising result: not only does the Frank–Wolfe-based method provide improved posterior estimates, it also can sometimes

have *reduced overall computational cost* compared to uniformly random subsampling for fixed M. This is due to the fact that M is an upper bound on the coreset size; the Frank–Wolfe algorithm often selects the same point multiple times, leading to coresets of size $\ll M$, whereas subsampling techniques always have coresets of size $\approx M$. Since the cost of posterior inference scales with the coreset size and dominates the cost of setting up either coreset construction algorithm, the Hilbert Frank–Wolfe method has a reduced overall cost. Generally speaking, we expect the Hilbert coreset methods to be slower than random subsampling for small M, where $\hat{\pi}$ setup and random projection dominates the time cost, but the Frank–Wolfe method to sometimes be faster for large M where the smaller coreset provides a significant inferential boost.

Although more detailed results for Hilbert importance sampling, uniform coresets, and the weighted 2-norm from Eq. (5.3) are deferred to Appendix B, we do provide a brief summary here. Fig. 7 shows that Hilbert importance sampling provides comparable performance to both uniform coresets and uniformly random subsampling on all models and data sets. The Hilbert Frank–Wolfe coreset construction algorithm typically outperforms all random subsampling methods. Finally, Figs. 8 and 9 show that the weighted 2-norm and Fisher information norm perform similarly in all cases considered.

8. Conclusion

This paper presented a fully-automated, scalable, and theoretically-sound Bayesian inference framework based on Hilbert coresets. The algorithms proposed in this work are simple to implement, reliably provide high-quality posterior approximation at a fraction of the cost of running inference on the full data set, and enable experts and nonexperts alike to conduct sophisticated modeling and exploratory analysis at scale. There are many avenues for future work, including exploring the application of Hilbert coresets in more complex high-dimensional models, using alternate Hilbert norms, connecting the norms proposed in the present work to more well-known measures of posterior discrepancy, investigating different choices of weighting function, obtaining tighter bounds on the quality of the random projection result, and using variants of the Frank–Wolfe algorithm (e.g. away-step, pairwise, and fully-corrective FW (Lacoste-Julien and Jaggi, 2015)) with stronger convergence guarantees.

Acknowledgments

This research is supported by a Google Faculty Research Award, an MIT Lincoln Laboratory Advanced Concepts Committee Award, and ONR grant N00014-17-1-2072. We also thank Sushrutha Reddy for finding and correcting a bug in the proof of Theorem 3.2.

Appendix A. Technical results and proofs

In this section we provide proofs of the main results from the paper, along with supporting technical lemmas. Lemma A.1 is the martingale extension of Hoeffding's inequality (Boucheron et al., 2013, Theorem 2.8, p. 34) known as Azuma's inequality. Lemma A.2 is the martingale extension of Bennet's inequality (Boucheron et al., 2013, Theorem 2.9, p. 35).

Lemma A.3 provides bounds on the expectation and martingale differences of the norm of a vector constructed by i.i.d. sampling from a discrete distribution. Finally, Lemma A.5 is a geometric result and Lemma A.6 bounds iterates of the logistic equation, both of which are used in the proof of the Frank-Wolfe error bound. Lemma A.4 provides a relationship between two vector alignment constants in the main text.

Lemma A.1 (Azuma's Inequality). *Suppose $(Y_m)_{m=0}^M$ is a martingale adapted to the filtration $(\mathcal{F}_m)_{m=0}^M$. If there is a constant ξ such that for each $m \in \{1, \dots, M\}$,*

$$|Y_m - Y_{m-1}| \leq \xi \quad a.s.,$$

then for all $\epsilon \geq 0$,

$$\mathbb{P}\left(Y_M - Y_0 > \epsilon\right) \leq e^{-\frac{\epsilon^2}{2M\xi^2}}.$$

Lemma A.2 (Martingale Bennet Inequality). *Suppose $(Y_m)_{m=0}^M$ is a martingale adapted to the filtration $(\mathcal{F}_m)_{m=0}^M$. If there are constants ξ and τ^2 such that for each $m \in \{1, \dots, M\}$,*

$$|Y_m - Y_{m-1}| \leq \xi \qquad \text{and} \qquad \mathbb{E}\left[(Y_m - Y_{m-1})^2 \,|\, \mathcal{F}_{m-1}\right] \leq \tau^2 \quad a.s.,$$

then for all $\epsilon \geq 0$,

$$\mathbb{P}\left(Y_M - Y_0 > \epsilon\right) \leq e^{-\frac{M\tau^2}{\xi^2} H\left(\frac{\epsilon\xi}{M\tau^2}\right)}, \quad H(x) := (1+x)\log(1+x) - x.$$

Lemma A.3. *Suppose U and $\{U_m\}_{m=1}^M$ are i.i.d. random vectors in a normed vector space with discrete support on $(u_n)_{n=1}^N$ with probabilities $(p_n)_{n=1}^N$, and*

$$Y := \left\| \frac{1}{M} \sum_{m=1}^{M} U_m - \mathbb{E}\left[U\right] \right\|.$$

Then we have the following results.

1. *Suppose $\dim\left(u_n\right)_{n=1}^N \leq d$ where $\dim$ is given by Definition 3.1, $\alpha_{n\cdot} \in [-1,1]^d$ are the coefficients used to approximate u_n in Definition 3.1, and A_j is a random variable equal to α_{nj} when $U = u_n$. Then*

$$\mathbb{E}\left[Y\right] \leq \frac{d}{\sqrt{M}} \left(\sum_{n=1}^{N} \|u_n\| \sqrt{\frac{p_n(1-p_n)}{N}} + \sqrt{\mathrm{Var}\left[\|U A_j\|\right]} \right).$$

2. *If the norm is a Hilbert norm,*

$$\mathbb{E}\left[Y\right] \leq \frac{1}{\sqrt{M}} \sqrt{\mathbb{E}\left[\|U\|^2\right] - \|\mathbb{E}\left[U\right]\|^2}.$$

3. *The random variable $Y_m := \mathbb{E}\left[Y \,|\, \mathcal{F}_m\right]$ with $\mathcal{F}_m$ the σ-algebra generated by $U_1, \dots, U_m$ is a martingale that satisfies, for $m \geq 1$, both*

$$|Y_m - Y_{m-1}| \leq \frac{1}{M} \max_{n,\ell} \|u_n - u_\ell\|$$

and

$$\mathbb{E}\left[(Y_m - Y_{m-1})^2 \mid \mathcal{F}_{m-1}\right] \leq \frac{1}{M^2}\mathbb{E}\left[\|U - U_1\|^2\right]$$

almost surely.

Proof. 1. Using the triangle inequality, denoting the number of times vector u_n is sampled as M_n,

$$\begin{aligned}\mathbb{E}[Y] \leq& \frac{1}{M}\sum_{n=1}^{N}\frac{d\,\|u_n\|}{\sqrt{N}}\mathbb{E}\left[|M_n - Mp_n|\right] \\ &+\frac{1}{M}\sum_{j=1}^{d}\mathbb{E}\left[\left|\sum_{n=1}^{N}(M_n - Mp_n)\,\|u_n\|\alpha_{nj}\right|\right].\end{aligned}$$

Bounding $\mathbb{E}[|\cdot|] \leq \sqrt{\mathbb{E}[(\cdot)^2]}$ via Jensen's inequality and evaluating the multinomial variances yields the desired result.

2. This follows from Jensen's inequality to write $\mathbb{E}[Y] \leq \sqrt{\mathbb{E}[Y^2]}$ and the expansion of the squared norm.

3. $(Y_m)_{m=0}^M$ is a standard Doob martingale with $Y_0 = \mathbb{E}[Y]$. Letting $U'_\ell = U_\ell$ for $\ell \neq m$ and U'_m be an independent random variable with $U'_m \overset{d}{=} U_m$, by the triangle inequality we have

$$\begin{aligned}&|Y_m - Y_{m-1}| \\ =&|\mathbb{E}[Y \mid \mathcal{F}_m] - Y_{m-1}| \\ =&\left|\mathbb{E}\left[\left\|\frac{1}{M}\sum_{\ell=1}^{M}U_\ell - \mathbb{E}[U]\right\| \mid \mathcal{F}_m\right] - Y_{m-1}\right| \\ =&\left|\mathbb{E}\left[\left\|\frac{1}{M}\left(U_m - U'_m\right) + \frac{1}{M}\sum_{\ell=1}^{M}U'_\ell - \mathbb{E}[U]\right\| \mid \mathcal{F}_m\right] - Y_{m-1}\right| \\ \leq&\left|\frac{1}{M}\mathbb{E}\left[\left\|U_m - U'_m\right\| \mid \mathcal{F}_m\right] + \mathbb{E}\left[\left\|\frac{1}{M}\sum_{\ell=1}^{M}U'_\ell - \mathbb{E}[U]\right\| \mid \mathcal{F}_m\right] - Y_{m-1}\right| \\ =&\frac{1}{M}\mathbb{E}\left[\left\|U_m - U'_m\right\| \mid \mathcal{F}_m\right] \qquad\qquad (A.1) \\ \leq&\frac{1}{M}\max_{n,\ell}\|u_n - u_\ell\|.\end{aligned}$$

Next, using Eq. (A.1) and Jensen's inequality, we have that

$$\begin{aligned}\mathbb{E}\left[(Y_m - Y_{m-1})^2 \mid \mathcal{F}_{m-1}\right] &\leq \mathbb{E}\left[\mathbb{E}\left[\frac{1}{M}\left\|U_m - U'_m\right\| \mid \mathcal{F}_m\right]^2 \mid \mathcal{F}_{m-1}\right] \\ &\leq \frac{1}{M^2}\mathbb{E}\left[\left\|U_m - U'_m\right\|^2 \mid \mathcal{F}_{m-1}\right] \\ &= \frac{1}{M^2}\mathbb{E}\left[\left\|U_m - U'_m\right\|^2\right].\end{aligned}$$

□

Proof of Theorem 3.2. Set $\delta \in (0,1)$. Rearranging the results of Lemma A.1, we have that with probability $\geq 1-\delta$,

$$Y_M \leq Y_0 + \sqrt{2M\xi^2\log\frac{1}{\delta}}.$$

We now apply the results of Lemma A.3(1) and Lemma A.3(3), noting that $\sum_{n=1}^N\sqrt{p_n(1-p_n)}$ is maximized when $p_n = 1/N$, where the discrete distribution is specified by atoms $u_n = \sigma\mathcal{L}_n/\sigma_n$ with probabilities σ_n/σ for $n \in [N]$. Further, when applying Lemma A.3(1), note that

$$\begin{aligned}\operatorname{Var}\|UA_j\| &= \sigma^2\operatorname{Var}|A_j| \\ &= \sigma^2\operatorname{Var}\left(|A_j| - \frac{1}{2}\right) \leq \sigma^2\mathbb{E}\left[\left(|A_j| - \frac{1}{2}\right)^2\right] \leq \frac{\sigma^2}{4}.\end{aligned}$$

This yields

$$Y_0 \leq \frac{3\sigma d}{2\sqrt{M}} \qquad\qquad \xi = \frac{\sigma\overline{\eta}}{M}.$$

Substituting these results into the above expression,

$$Y_M \leq \frac{\sigma}{\sqrt{M}}\left(\frac{3}{2}\dim\left(\mathcal{L}_n\right)_{n=1}^N + \overline{\eta}\sqrt{2\log\frac{1}{\delta}}\right).$$

□

Proof of Theorem 4.1. Set $\delta \in (0,1)$. Rearranging the results of Lemmas A.1 and A.2, we have that with probability $\geq 1-\delta$,

$$Y_M \leq Y_0 + \min\left(\sqrt{2M\xi^2\log\frac{1}{\delta}}, \frac{M\tau^2}{\xi}H^{-1}\left(\frac{\xi^2}{M\tau^2}\log\frac{1}{\delta}\right)\right).$$

We now apply the results of Lemma A.3(3), where the discrete distribution is specified by atoms $u_n = \mathcal{L}_n/p_n$ with probabilities p_n for $n \in [N]$. Define M_n to be the number of times index n is sampled; then $(M_1, \ldots, M_N) \sim \mathsf{Multi}(M, (p_n)_{n=1}^N)$. Then since our vectors are in a Hilbert space, we use Lemma A.3(2) and Lemma A.3(3) to find that

$$\begin{aligned}Y_M &= \left\|\frac{1}{M}\sum_{m=1}^{M}U_m - \mathbb{E}[U]\right\| = \left\|\sum_{n=1}^{N}\frac{M_n}{Mp_n}\mathcal{L}_n - \mathcal{L}\right\| \\ Y_0 &\leq \sqrt{\frac{1}{M}\left(\sum_{n=1}^{N}\frac{\|\mathcal{L}_n\|^2}{p_n} - \|\mathcal{L}\|^2\right)}\end{aligned}$$

$$\xi = \frac{1}{M}\max_{m,n}\left\|\frac{\mathcal{L}_n}{p_n} - \frac{\mathcal{L}_m}{p_m}\right\|$$
$$\tau^2 = \frac{1}{M^2}\mathbb{E}\left[\left\|U_m - U'_m\right\|^2\right] = \frac{2}{M^2}\left(\sum_{n=1}^N \frac{\|\mathcal{L}_n\|^2}{p_n} - \|\mathcal{L}\|^2\right).$$

Minimizing both τ^2 and Y_0 over $(p_n)_{n=1}^N$ by setting the derivative to 0 yields

$$p_n = \frac{\|\mathcal{L}_n\|}{\sigma} \qquad \sigma := \sum_{n=1}^N \|\mathcal{L}_n\|. \tag{A.2}$$

Finally, we have that

$$Y_0 \leq \sqrt{\frac{1}{M}\sigma^2\eta^2} \qquad \tau^2 = \frac{2\sigma^2}{M^2}\left(1 - \frac{\|\mathcal{L}\|^2}{\sigma^2}\right) = \frac{2\sigma^2\eta^2}{M^2} \qquad \xi = \frac{\sigma}{M}\overline{\eta},$$

and $\frac{M_n}{M}\frac{1}{p_n} = W_n$ from Algorithm 1, so

$$\begin{aligned}\|\mathcal{L}(W) - \mathcal{L}\| &\leq \frac{\sigma\eta}{\sqrt{M}} + \min\left(\sqrt{2\frac{\sigma^2\overline{\eta}^2}{M}\log\frac{1}{\delta}}, \frac{2\sigma\eta^2}{\overline{\eta}}H^{-1}\left(\frac{\overline{\eta}^2}{2M\eta^2}\log\frac{1}{\delta}\right)\right)\\ &= \frac{\sigma}{\sqrt{M}}\left(\eta + \eta_M\sqrt{2\log\frac{1}{\delta}}\right)\\ \eta_M &:= \min\left(\overline{\eta}, \eta\sqrt{\frac{2M\eta^2}{\overline{\eta}^2\log\frac{1}{\delta}}}H^{-1}\left(\frac{\overline{\eta}^2\log\frac{1}{\delta}}{2M\eta^2}\right)\right).\end{aligned}$$

□

Lemma A.4. *Given a Hilbert norm, $\overline{\eta}$ from Eq. (3.3) and η from Eq. (4.1) satisfy*

$$\eta \leq \frac{\overline{\eta}}{\sqrt{2}}.$$

Proof. Noting $\|\mathcal{L}\|^2 = \langle\mathcal{L},\mathcal{L}\rangle$ and expanding the definition from Eq. (4.1),

$$\eta^2 = 1 - \sum_{n,m=1}^N \frac{\sigma_n\sigma_m}{\sigma^2}\left\langle\frac{\mathcal{L}_n}{\sigma_n}, \frac{\mathcal{L}_m}{\sigma_m}\right\rangle \leq 1 - \min_{n,m\in[N]}\left\langle\frac{\mathcal{L}_n}{\sigma_n}, \frac{\mathcal{L}_m}{\sigma_m}\right\rangle = \frac{1}{2}\overline{\eta}^2$$

where the inequality follows from $\sum_{n,m=1}^N \frac{\sigma_n\sigma_m}{\sigma^2} = 1$. □

Lemma A.5. *$\mathcal{L}$ is in the relative interior of the convex hull of $\left\{\frac{\sigma}{\|\mathcal{L}_n\|}\mathcal{L}_n\right\}_{n=1}^N$.*

Proof. First, since K (the kernel matrix of inner products defined in Eq. (4.2)) is a symmetric postive semidefinite $N \times N$ matrix, there exists an $N \times N$ matrix U such that $\|\mathcal{L}(w)\|^2 = w^T K w = w^T U^T U w = \|u(w)\|_2^2$, where $u(w) := \sum_{n=1}^N w_n u_n$, $u := \sum_{n=1}^N u_n$, and $u_n \in \mathbb{R}^N$ are the columns of U. Therefore the mapping $\mathcal{L}(w) \to u(w)$ is a linear isometry from the Hilbert space to $\mathbb{R}^N$, so if u is in the relative interior of the convex hull of $\left\{\frac{\sigma}{\|u_n\|}u_n\right\}_{n=1}^N$, the result follows. Let y be any other point in the convex hull in $\mathbb{R}^N$, with coefficients γ_n. If we set

$$\lambda = \min_{n:\gamma_n > \frac{\|u_n\|}{\sigma}} \frac{\gamma_n}{\gamma_n - \frac{\|u_n\|}{\sigma}}$$

where the minimum of an empty set is defined to be ∞, then $\lambda u + (1-\lambda)y$ is in the convex hull and $\lambda > 1$. Since for any point y we can find such a λ, the result follows from (Rockafeller, 1970, Theorem 6.4, p. 47). □

Lemma A.6. *The logistic recursion,*

$$x_{n+1} \leq \alpha x_n(1 - x_n),$$

for $x_0, \alpha \in [0,1]$ satisfies

$$\forall n \in \mathbb{N}, \quad x_n \leq \frac{x_0}{\alpha^{-n} + x_0 n}.$$

Proof. The proof proceeds by induction. The bound holds at $n = 0$ since

$$x_0 \leq \frac{x_0}{\alpha^0 + 0} = x_0.$$

Since $1 - x \leq 1/(1+x)$, for any $n \geq 0$,

$$x_{n+1} \leq \alpha x_n(1 - x_n) \leq \alpha\frac{x_n}{1 + x_n}.$$

Assuming the bound holds for any $n \geq 0$, and noting $x/(1+x)$ is monotone increasing for $x \geq 0$, we can substitute the bound yielding

$$x_{n+1} \leq \alpha\frac{\frac{x_0}{\alpha^{-n} + nx_0}}{1 + \frac{x_0}{\alpha^{-n} + nx_0}} = \alpha\frac{x_0}{\alpha^{-n} + nx_0 + x_0} = \frac{x_0}{\alpha^{-(n+1)} + \alpha^{-1}(n+1)x_0}.$$

The final result follows since $\alpha^{-1} \geq 1$. □

Proof of Lemma 4.3. Let w_t be the weight vector at iteration t in Algorithm 2, and let f_t and d_t be the Frank-Wolfe vertex index and direction, respectively, from Eq. (4.4). For brevity, denote the cost $J(w) := (w-1)^T K(w-1)$. For any $\gamma \in \mathbb{R}$, if we let $w_{t+1} = w_t + \gamma d_t$ we have that

$$J(w_{t+1}) = J(w_t) + 2\gamma d_t^T K(w_t - 1) + \gamma^2 d_t^T K d_t. \tag{A.3}$$

Minimizing Eq. (A.3) over $\gamma \in \mathbb{R}$ yields Eq. (4.5) (expressed as a quadratic form with gram matrix K),

$$\gamma_t = \frac{d_t^T K(1 - w_t)}{d_t^T K d_t}. \tag{A.4}$$

Suppose $\gamma_t < 0$. Then $d_t^T K(1-w_t) < 0$; but d_t maximizes this product over feasible directions, so

$$0 > d_t^T K(1-w_t) > (1-w_t)^T K(1-w_t) = J(w_t) \geq 0, \tag{A.5}$$

which is a contradiction. Now suppose $\gamma_t > 1$. Then

$$d_t^T K(1-w_t) > d_t^T K d_t,$$

and Eq. (A.5) holds again, so if we were to select $\gamma = 1$ in Eq. (A.3), we would have

$$0 \leq J(w_{t+1}) < J(w_t) + d_t^T K(w_t - 1) \leq 0,$$

which is another contradiction, so $\gamma_t \leq 1$. Therefore $\gamma_t \in [0,1]$ □

Proof of Theorem 4.4. Using the same notation as the proof of Lemma 4.3 above, first note that $J(w_0) \leq \sigma^2\eta^2$ as initialized by Eq. (4.3): for any $\xi \in \mathbb{R}_+^N$ with $\sum_n \xi_n = 1$,

$$\frac{J(w_0)}{\sigma^2} = 1 - 2\left\langle \frac{\mathcal{L}_{f_0}}{\sigma_{f_0}}, \frac{\mathcal{L}}{\sigma} \right\rangle + \frac{\|\mathcal{L}\|^2}{\sigma^2} \leq 1 - 2\sum_{n=1}^N \xi_n \left\langle \frac{\mathcal{L}_n}{\sigma_n}, \frac{\mathcal{L}}{\sigma} \right\rangle + \frac{\|\mathcal{L}\|^2}{\sigma^2}$$

since f_0 maximizes $\langle \mathcal{L}, \mathcal{L}_n/\sigma_n \rangle$ over $n \in [N]$, and picking $\xi_n = \sigma_n/\sigma$ yields

$$\frac{J(w_0)}{\sigma^2} \leq 1 - 2\sum_{n=1}^N \left\langle \frac{\mathcal{L}_n}{\sigma}, \frac{\mathcal{L}}{\sigma} \right\rangle + \frac{\|\mathcal{L}\|^2}{\sigma^2} = 1 - \frac{\|\mathcal{L}\|^2}{\sigma^2} = \eta^2.$$

By Lemma 4.3, we are guaranteed that each Frank-Wolfe iterate using exact line search is feasible, and substituting Eq. (A.4) into Eq. (A.3) yields

$$\begin{aligned} J(w_{t+1}) &= J(w_t) - \frac{\left(d_t^T K(1-w_t)\right)^2}{d_t^T K d_t} \\ &= J(w_t)\left(1 - \left\langle \frac{\frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t} - \mathcal{L}(w_t)}{\|\frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t} - \mathcal{L}(w_t)\|}, \frac{\mathcal{L} - \mathcal{L}(w_t)}{\|\mathcal{L} - \mathcal{L}(w_t)\|} \right\rangle^2\right). \end{aligned} \tag{A.6}$$

We now employ a technique due to Guélat and Marcotte (1986): by Lemma A.5, $\mathcal{L}$ is in the relative interior of the convex hull of the $\left\{\frac{\sigma}{\sigma_n}\mathcal{L}_n\right\}_{n=1}^N$, so there exists an $r > 0$ such that for any feasible w,

$$\mathcal{L}(w) + (\|\mathcal{L} - \mathcal{L}(w)\| + r)\frac{\mathcal{L} - \mathcal{L}(w)}{\|\mathcal{L} - \mathcal{L}(w)\|}$$

is also in the convex hull. Thus, since the Frank-Wolfe vertex $\frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t}$ maximizes $\langle \mathcal{L}(w) - \mathcal{L}(w_t), \mathcal{L} - \mathcal{L}(w_t) \rangle$ over feasible w, we have that

$$\left\langle \frac{\frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t} - \mathcal{L}(w_t)}{\|\frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t} - \mathcal{L}(w_t)\|}, \frac{\mathcal{L} - \mathcal{L}(w_t)}{\|\mathcal{L} - \mathcal{L}(w_t)\|} \right\rangle \geq \left\langle \frac{(\|\mathcal{L} - \mathcal{L}(w_t)\| + r)\frac{\mathcal{L} - \mathcal{L}(w_t)}{\|\mathcal{L} - \mathcal{L}(w_t)\|}}{\|\frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t} - \mathcal{L}(w_t)\|}, \frac{\mathcal{L} - \mathcal{L}(w_t)}{\|\mathcal{L} - \mathcal{L}(w_t)\|} \right\rangle$$

$$\begin{aligned} &= \frac{\sqrt{J(w_t)} + r}{\|\frac{\sigma}{\sigma_{f_t}}\mathcal{L}_{f_t} - \mathcal{L}(w_t)\|} \\ &\geq \frac{\sqrt{J(w_t)} + r}{\sigma\overline{\eta}}. \end{aligned}$$

Substituting this into Eq. (A.6) yields

$$J(w_{t+1}) \leq J(w_t)\left(1 - \left(\frac{\sqrt{J(w_t)} + r}{\sigma\overline{\eta}}\right)^2\right) \leq J(w_t)\left(\nu^2 - \frac{J(w_t)}{\sigma^2\overline{\eta}^2}\right),$$

where $\nu := 1 - \frac{r^2}{\sigma^2\overline{\eta}^2}$. Defining $x_t := \frac{J(w_t)}{\sigma^2\overline{\eta}^2\nu^2}$, we have that $0 \leq x_t \leq 1$ and

$$x_{t+1} \leq \nu^2 x_t(1 - x_t),$$

and so Lemma A.6 implies that

$$\frac{J(w_t)}{\sigma^2\overline{\eta}^2\nu^2} \leq \frac{\frac{J(w_0)}{\sigma^2\overline{\eta}^2\nu^2}}{\nu^{-2t} + \frac{J(w_0)}{\sigma^2\overline{\eta}^2\nu^2}t}.$$

Further, since the function $\frac{a}{a+b}$ is monotonically increasing in a for all $a, b \geq 0$, we can use the bound on the initial objective $J(w_0)$, yielding

$$J(w_t) \leq \frac{\sigma^2\eta^2\overline{\eta}^2\nu^2}{\overline{\eta}^2\nu^{2-2t} + \eta^2 t}$$

The proof concludes by noting that we compute $M-1$ iterations after initialization to construct a coreset of size $\leq M$. The second stated bound results from the fact that $\overline{\eta} \geq \eta$ and $\nu \leq 1$.

The weaker bound in the note after the theorem is a result of a technique very similar to that commonly found in past work (Clarkson, 2010; Jaggi, 2013): starting from Eq. (A.3), we bound $d_t^T K d_t \leq \sigma^2\overline{\eta}^2$ and $d_t^T K(w_t - 1) \leq -J(w_t)$, and then use recursion to prove that $J(w_t) \leq \frac{4\sigma^2\overline{\eta}^2}{3t+4}$ given $\gamma_t = \frac{2}{3t+4}$. □

Proof of Theorem 5.2. Suppose $\max_{m,n} \left|\langle \mathcal{L}_n, \mathcal{L}_m \rangle - v_n^T v_m\right| \leq \epsilon$. Then

$$\begin{aligned} &(w-1)^T K(w-1) - (w-1)^T V(w-1) \leq \\ &\qquad \sum_{m,n} |w_n - 1|\,|w_m - 1|\left|\langle \mathcal{L}_n, \mathcal{L}_m \rangle - v_n^T v_m\right| \leq \|w-1\|_1^2\,\epsilon. \end{aligned}$$

We now bound the probability that the above inequality holds, assuming $D\nabla\mathcal{L}_n(\mu)_d\nabla\mathcal{L}_m(\mu)_d$ (when using $\mathcal{D}_{\hat{\pi},F}$) or $\mathcal{L}_n(\mu)\mathcal{L}_m(\mu)$ (when using $\mathcal{D}_{\hat{\pi},2}$) is sub-Gaussian with constant ξ^2. For brevity denote the true vector $\mathcal{L}_n$ as $\mathcal{L}_n$ and its random projection as v_n. Then

$$\mathbb{P}\left(\max_{m,n}\left|\langle \mathcal{L}_n, \mathcal{L}_m \rangle - v_n^T v_m\right| \geq \epsilon\right)$$

$$
\begin{aligned}
&\leq \sum_{m,n} \mathbb{P}\left(\left|\langle \mathcal{L}_n, \mathcal{L}_m\rangle - v_n^T v_m\right| \geq \epsilon\right) \\
&\leq N^2 \max_{m,n} \mathbb{P}\left(\left|\langle \mathcal{L}_n, \mathcal{L}_m\rangle - v_n^T v_m\right| \geq \epsilon\right) \\
&= N^2 \max_{m,n} \mathbb{P}\left(\left|\langle \mathcal{L}_n, \mathcal{L}_m\rangle - \frac{1}{J}\sum_{j=1}^{J} v_{nj} v_{mj}\right| \geq \epsilon\right) \\
&\leq 2N^2 e^{-\frac{J\epsilon^2}{2\xi^2}},
\end{aligned}
$$

using Hoeffding's inequality for sub-Gaussian variables. Thus if we fix $\delta \in (0,1)$, with probability $\geq 1-\delta$,

$$
\sqrt{\frac{2\xi^2}{J}\log\frac{2N^2}{\delta}} \geq \epsilon.
$$

Therefore, with probability $\geq 1-\delta$,

$$
\|\mathcal{L}(w) - \mathcal{L}\|^2 \leq \|v(w) - v\|^2 + \|w-1\|_1^2 \sqrt{\frac{2\xi^2}{J}\log\frac{2N^2}{\delta}}.
$$

□

Appendix B. Additional Results

This section contains supplementary quantitative evaluations. Fig. 7 compares Hilbert importance sampling to uniformly random subsampling and uniform coresets. These results demonstrate that all subsampling techniques perform similarly, with Hilbert coresets often the best choice of the three. The Frank–Wolfe constructions outperform subsampling techniques across all models and data sets considered. Figs. 8 and 9 compare the weighted 2-norm from Eq. (5.3) to the weighted Fisher information norm from Eq. (5.1) in both importance sampling and Frank–Wolfe-based Hilbert coreset constructions. These results show that the 2-norm and F-norm perform similarly in all cases.

Appendix C. Derivation of the Gaussian uniform coreset sensitivity

The sensitivity of observation y_n used in the construction of a Bayesian coreset (Huggins et al., 2016) (ignoring constants) is

$$
N\sigma_n = \sup_{\mu\in\mathbb{R}^d} \frac{N\mathcal{L}_n(\mu)}{\mathcal{L}(\mu)} = \sup_{\mu\in\mathbb{R}^d} \frac{N\left(y_n-\mu\right)^T\left(y_n-\mu\right)}{\sum_{m=1}^{N}\left(y_m-\mu\right)^T\left(y_m-\mu\right)}.
$$

By noting that

$$
\frac{1}{N}\sum_{m=1}^{N}\left(y_m-\mu\right)^T\left(y_m-\mu\right) = \frac{1}{N}\sum_{m=1}^{N} y_m^T y_m - \bar{y}^T\bar{y} + \left(\mu-\bar{y}\right)^T\left(\mu-\bar{y}\right),
$$

where $\bar{y} := \frac{1}{N}\sum_{m=1}^{N} y_m$, we can keep the denominator constant by varying μ on the ball centered at $\bar{y}$ of constant radius. The maximum of the numerator while keeping the

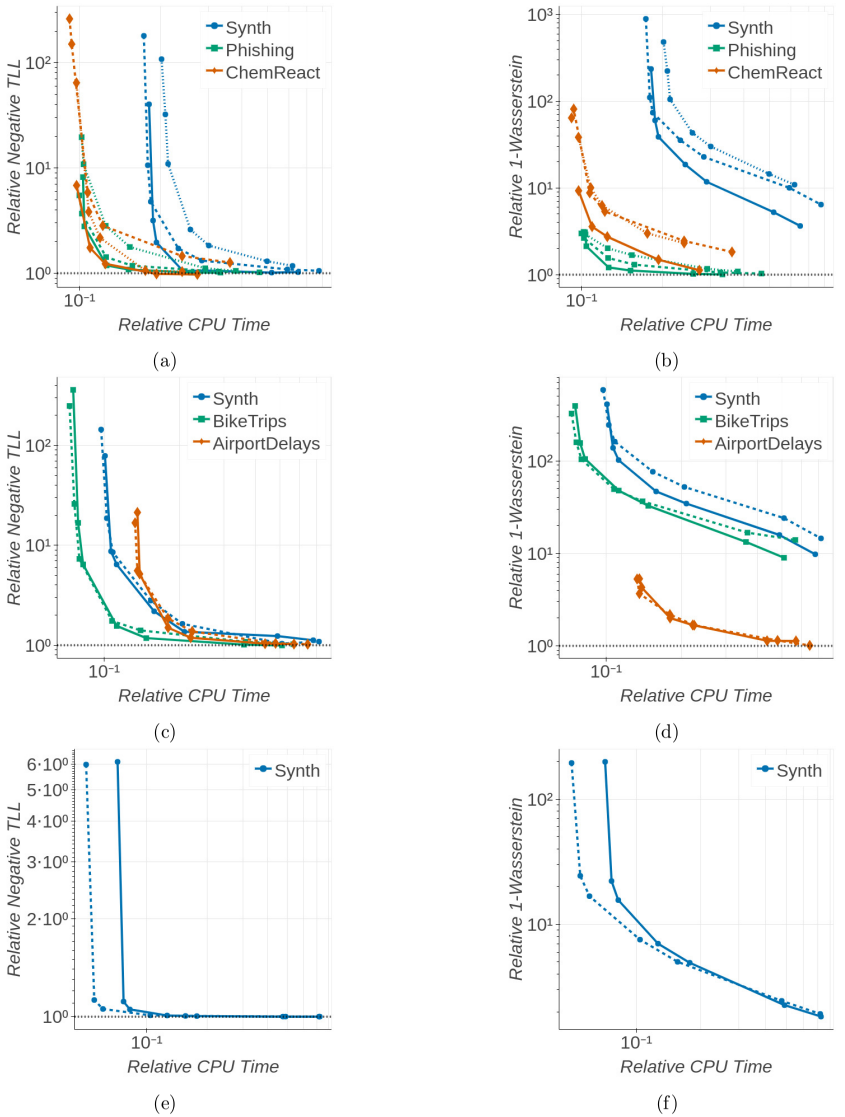


Figure 7: Comparisons for IS-F (solid), uniform coresets (dotted), and uniform random subsampling (dashed) on (7a, 7b) logistic regression, (7c, 7d) Poisson regression, and (7e, 7f) directional clustering. Both axes are normalized; see Section 7.3.

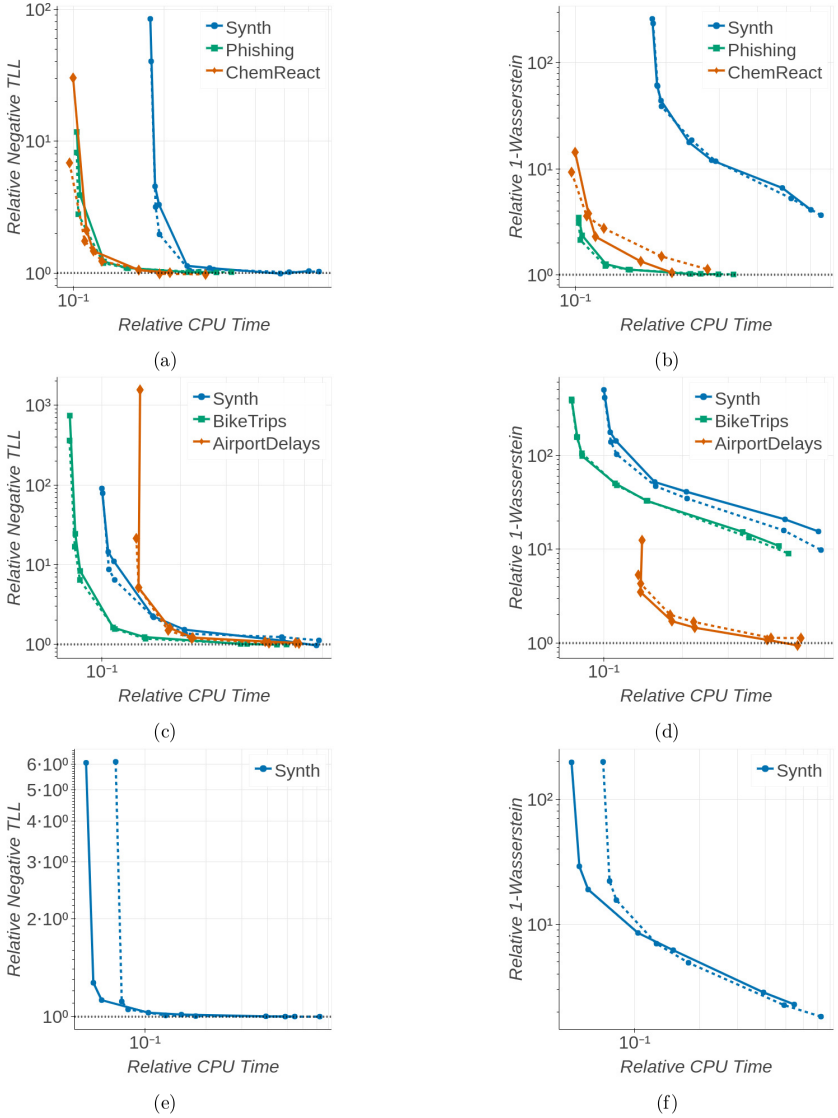


Figure 8: Comparisons for IS-2 (solid) and IS-F (dashed) on (8a, 8b) logistic regression, (8c, 8d) Poisson regression, and (8e, 8f) directional clustering. Both axes are normalized; see Section 7.3.

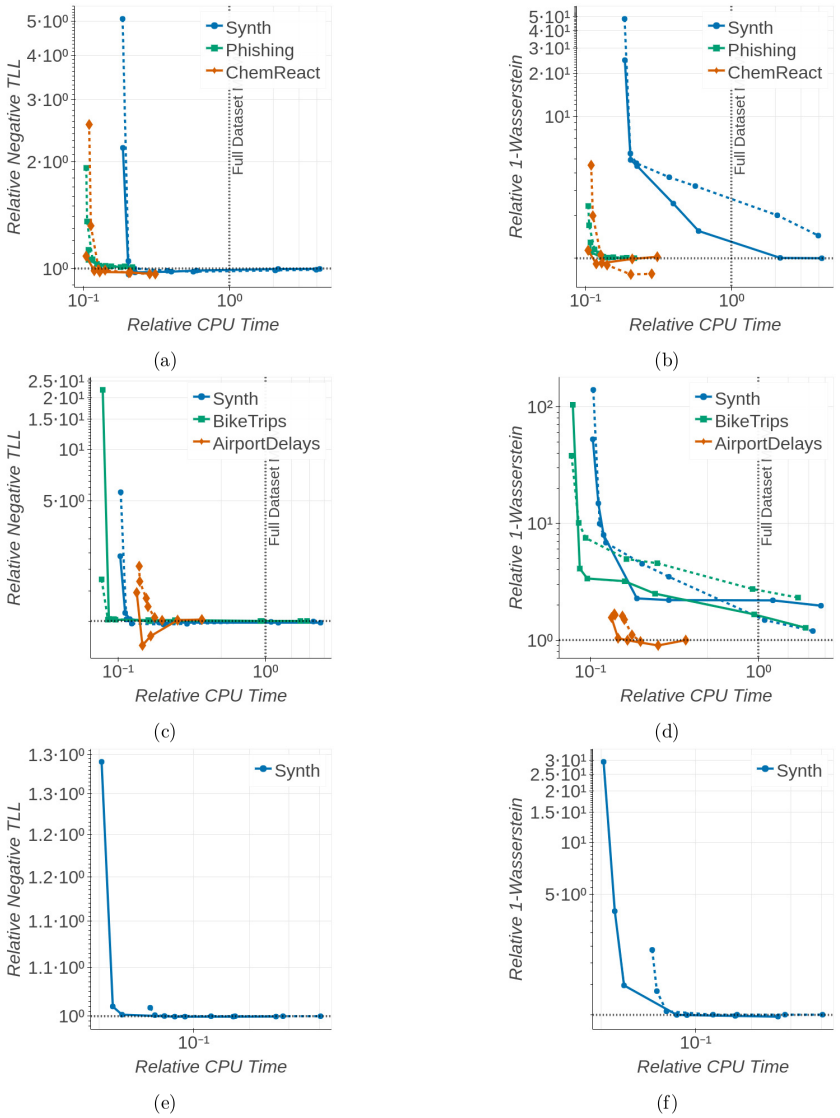


Figure 9: Comparisons for FW-2 (solid) and FW-F (dashed) on (9a, 9b) logistic regression, (9c, 9d) Poisson regression, and (9e, 9f) directional clustering. Both axes are normalized; see Section 7.3.

denominator constant happens when μ lies on the 1d affine space between $\bar{y}$ and y_n; so we can reparametrize $\mu = \lambda\bar{y} + (1-\lambda)y_n$ for $\lambda \in \mathbb{R}$, yielding the optimization

$$\sup_{\mu\in\mathbb{R}^d} \frac{N\mathcal{L}_n(\mu)}{\mathcal{L}(\mu)} = \sup_{\lambda\in\mathbb{R}} \frac{\lambda^2 (y_n-\bar{y})^T (y_n-\bar{y})}{\frac{1}{N}\sum_{m=1}^N y_m^T y_m - \bar{y}^T\bar{y} + (1-\lambda)^2 (y_n-\bar{y})^T (y_n-\bar{y})}$$

for which the optimum occurs at $\lambda^\star = \left(\frac{\frac{1}{N}\sum_{m=1}^N y_m^T y_m - \bar{y}^T\bar{y}}{(y_n-\bar{y})^T(y_n-\bar{y})} + 1\right)$ with value

$$N\sigma_n = \sup_{\mu\in\mathbb{R}^d} \frac{N\mathcal{L}_n(\mu)}{\mathcal{L}(\mu)} = 1 + \frac{(y_n-\bar{y})^T (y_n-\bar{y})}{\frac{1}{N}\sum_{m=1}^N y_m^T y_m - \bar{y}^T\bar{y}}.$$

References

Pankaj K Agarwal, Sariel Har-Peled, and Kasturi R Varadarajan. Geometric approximation via coresets. *Combinatorial and computational geometry*, 52:1–30, 2005.

Daniel Ahfock, William Astle, and Sylvia Richardson. Statistical properties of sketching algorithms. *arXiv:1706.03665*, 2017.

Sungjin Ahn, A. Korattikara, and Max Welling. Bayesian posterior sampling via stochastic gradient Fisher scoring. In *International Conference on Machine Learning*, 2012.

Olivier Bachem, Mario Lucic, and Andreas Krause. Coresets for nonparametric estimation—the case of DP-means. In *International Conference on Machine Learning*, 2015.

Olivier Bachem, Mario Lucic, S. Hamed Hassani, and Andreas Krause. Approximate k-means++ in sublinear time. In *AAAI Conference on Artificial Intelligence*, 2016.

Krishnakumar Balasubramanian and Saeed Ghadimi. Zeroth-order (non)-convex stochastic optimization via conditional gradient and gradient updates. In *Advances in Neural Information Processing Systems*, 2018.

Arindam Banerjee, Inderjit Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von Mises-Fisher distributions. *Journal of Machine Learning Research*, 6:1345–1382, 2005.

Rémi Bardenet and Odalic-Ambrym Maillard. A note on replacing uniform subsampling by random projections in MCMC for linear regression of tall datasets. Technical report, HAL Id: hal-01248841, 2015.

Rémi Bardenet, Arnaud Doucet, and Chris C Holmes. Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. In *International Conference on Machine Learning*, pages 405–413, 2014.

Rémi Bardenet, Arnaud Doucet, and Chris Holmes. On Markov chain Monte Carlo methods for tall data. *arXiv:1505.02827*, 2015.

Christopher Bishop. *Pattern recognition and machine learning*. Springer, 2006.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: a nonasymptotic theory of independence*. Oxford University Press, 2013.

Vladimir Braverman, Dan Feldman, and Harry Lang. New frameworks for offline and streaming coreset constructions. *arXiv:1612.00889*, 2016.

Tamara Broderick, Nicholas Boyd, Andre Wibisono, Ashia Wilson, and Michael Jordan. Streaming variational Bayes. In *Advances in Neural Information Processing Systems*, 2013.

Trevor Campbell, Julian Straub, John W. Fisher III, and Jonathan How. Streaming, distributed variational inference for Bayesian nonparametrics. In *Advances in Neural Information Processing Systems*, 2015.

Kenneth Clarkson. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. *ACM Transactions on Algorithms*, 6(4), 2010.

Adji Dieng, Dustin Tran, Rajesh Ranganath, John Paisley, and David Blei. Variational inference via χ upper bound minimization. In *Advances in Neural Information Processing Systems*, 2017.

Reihaneh Entezari, Radu Craiu, and Jeffrey Rosenthal. Likelihood inflating sampling algorithm. *arXiv:1605.02113*, 2016.

Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *ACM Symposium on Theory of Computing*, pages 569–578, 2011.

Dan Feldman, Matthew Faulkner, and A. Krause. Scalable training of mixture models via coresets. In *Advances in Neural Information Processing Systems*, pages 2142–2150, 2011.

Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: constant-size coresets for k-means, pca and projective clustering. In *ACM–SIAM Symposium on Discrete Algorithms*, pages 1434–1453, 2013.

Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

Andrew Gelman, John Carlin, Hal Stern, David Dunson, Aki Vehtari, and Donald Rubin. *Bayesian data analysis*. CRC Press, 3rd edition, 2013.

Leo Geppert, Katja Ickstadt, Alexander Munteanu, Jesn Quedenfeld, and Christian Sohler. Random projections for Bayesian regression. *Statistics and Computing*, 27:79–101, 2017.

Jacques Guélat and Patrice Marcotte. Some comments on Wolfe's 'away step'. *Mathematical Programming*, 35:110–119, 1986.

Lei Han, Ting Yang, and Tong Zhang. Local uncertainty sampling for large-scale multi-class logistic regression. *arXiv:1604.08098*, 2016.

Elad Hazan and Haipeng Luo. Variance-reduced and projection-free stochastic optimization. In *International Conference on Machine Learning*, 2016.

Matthew Hoffman and Andrew Gelman. The No-U-Turn Sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15:1351–1381, 2014.

Matthew D Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.

Jonathan Huggins, Trevor Campbell, and Tamara Broderick. Coresets for Bayesian logistic regression. In *Advances in Neural Information Processing Systems*, 2016.

Jonathan Huggins, Ryan Adams, and Tamara Broderick. PASS-GLM: polynomial approximate sufficient statistics for scalable Bayesian GLM inference. In *Advances in Neural Information Processing Systems*, 2017.

Martin Jaggi. Revisiting Frank-Wolfe: projection-free sparse convex optimization. In *International Conference on Machine Learning*, 2013.

Oliver Johnson and Andrew Barron. Fisher information inequalities and the central limit theorem. *Probability Theory and Related Fields*, 129:391–409, 2004.

Michael Jordan, Zoubin Ghahramani, Tommi Jaakkola, and Lawrence Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

Anoop Korattikara, Yutian Chen, and Max Welling. Austerity in MCMC land: cutting the Metropolis-Hastings budget. In *International Conference on Machine Learning*, 2014.

Alp Kucukelbir, Rajesh Ranganath, Andrew Gelman, and David Blei. Automatic variational inference in Stan. In *Advances in Neural Information Processing Systems*, 2015.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David Blei. Automatic differentiation variational inference. *Journal of Machine Learning Research*, 18:1–45, 2017.

Simon Lacoste-Julien and Martin Jaggi. On the global linear convergence of Frank-Wolfe optimization variants. In *Advances in Neural Information Processing Systems*, 2015.

Michael Langberg and Leonard Schulman. Universal ϵ-approximators for integrals. In *Annual ACM–SIAM Symposium on Discrete Algorithms*, pages 598–607, 2010.

Yingzhen Li and Richard Turner. Rényi divergence variational inference. In *Advances in Neural Information Processing Systems*, 2016.

Qiang Liu and Dilin Wang. Stein variational gradient descent: a general purpose Bayesian inference algorithm. In *Advances in Neural Information Processing Systems*, 2016.

Mario Lucic, Olivier Bachem, and Andreas Krause. Strong coresets for hard and soft bregman clustering with applications to exponential family mixtures. In *International Conference on Artificial Intelligence and Statistics*, 2016.

Dougal Maclaurin and Ryan Adams. Firefly Monte Carlo: exact MCMC with subsets of data. In *Conference on Uncertainty in Artificial Intelligence*, 2014.

Radford Neal. MCMC using Hamiltonian dynamics. In Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng, editors, *Handbook of Markov chain Monte Carlo*, chapter 5. CRC Press, 2011.

Maxim Rabinovich, Elaine Angelino, and Michael Jordan. Variational consensus Monte Carlo. *arXiv:1506.03074*, 2015.

Ali Rahimi and Ben Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, 2007.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *International Conference on Artificial Intelligence and Statistics*, 2014.

Sashank Reddi, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic Frank-Wolfe methods for nonconvex optimization. *arXiv:1607.08254*, 2016.

Christian Robert, Víctor Elvira, Nick Tawn, and Changye Wu. Accelerating MCMC algorithms. *arXiv:1804.02719*, 2018.

R. Tyrrell Rockafeller. *Convex analysis*. Princeton University Press, 1970.

Steven Scott, Alexander Blocker, Fernando Bonassi, Hugh Chipman, Edward George, and Robert McCulloch. Bayes and big data: the consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management*, 11:78–88, 2016.

S Srivastava, Volkan Cevher, Q Tran-Dinh, and David Dunson. WASP: scalable Bayes via barycenters of subset posteriors. In *International Conference on Artificial Intelligence and Statistics*, 2015.

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

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

Journal of Machine Learning Research 20 (2019) 1-24 Submitted 10/17; Published 2/19

Smooth neighborhood recommender systems

Ben Dai BENDAI2-C@MY.CITYU.EDU.HK
Junhui Wang J.H.WANG@CITYU.EDU.HK
School of Data Science
City University of Hong Kong
Kowloon Tong, 999077, Hong Kong

Xiaotong Shen XSHEN@UMN.EDU
School of Statistics
University of Minnesota
Minneapolis, MN 55455, USA

Annie Qu ANNIEQU@ILLINOIS.EDU
Department of Statistics
University of Illinois at Urbana Champaign
Champaign, IL 61820, USA

Editor: Gert Lanckriet

Abstract

Recommender systems predict users' preferences over a large number of items by pooling similar information from other users and/or items in the presence of sparse observations. One major challenge is how to utilize user-item specific covariates and networks describing user-item interactions in a high-dimensional situation, for accurate personalized prediction. In this article, we propose a smooth neighborhood recommender in the framework of the latent factor models. A similarity kernel is utilized to borrow neighborhood information from continuous covariates over a user-item specific network, such as a user's social network, where the grouping information defined by discrete covariates is also integrated through the network. Consequently, user-item specific information is built into the recommender to battle the 'cold-start" issue in the absence of observations in collaborative and content-based filtering. Moreover, we utilize a "divide-and-conquer" version of the alternating least squares algorithm to achieve scalable computation, and establish asymptotic results for the proposed method, demonstrating that it achieves superior prediction accuracy. Finally, we illustrate that the proposed method improves substantially over its competitors in simulated examples and real benchmark data–*Last.fm* music data.

Keywords: Blockwise coordinate decent, Cold-start, Kernel smoothing, Neighborhood, Personalized prediction, Singular value decomposition, Social networks.

1. Introduction

Recommender systems predict users' preferences over a large number of items by pooling similar information from other users or items when observations are sparse, which are particularly useful in personalized prediction. It has become an essential part of e-commerce, with applications in movie rentals (MovieLens; Miller et al. 2003), restaurant guides (Entree; Burke 2002), book recommendations (Amazon; Linden et al. 2003), and personalized e-news (Daily learner; Billsus and Pazzani 2000).

Two main streams emerge for training recommender systems: collaborative filtering, which predicts users' behaviors based on similar users (Bell and Koren, 2007), and content-based filtering, which builds user and item profiles based on domain knowledge and recommends items with similar profiles (Lang, 1995; Melville et al., 2002). Specifically, collaborative filtering predicts unknown ratings by averaging over similar users' ratings with weights; such as the latent factor approach (Feuerverger et al., 2012), latent Dirichlet allocation (LDA; Blei et al. 2003), probabilistic latent semantic analysis (pLSA; Hofmann 2004), regularized singular value decomposition (regularized SVD; Paterek 2007), and restricted Boltzmann machines (RBM; Salakhutdinov et al. 2007). Among them, the regularized SVD approach has become popular due to its high predictive performance and scalability in real applications. In addition, Koren (2008) further generalizes SVD to model users' implicit feedbacks, and Forbes and Zhu (2011) incorporates content information in the regularized SVD approach through a regression-type of constraint. For content-based filtering, keywords analysis extracts features from items previously rated by a user to develop a profile of the user's interests, and recommendation is made by comparing the user profile and potential items (Lang, 1995). The naive Bayes (Billsus and Pazzani, 2000), decision tree (Pazzani et al., 1996) and kNN (Middleton et al., 2004) formulate this type of recommendation as a classification problem, where each item can be labeled by users as "like" or "dislike". Hybrid recommender systems (Burke, 2002) utilize geo-social correlations to accommodate new users and items through location-based recommendation systems; Bi et al. (2017) proposes a group-specific latent factor model by utilizing missingness-related characteristics to accommodate new users or items without any observed ratings.

To achieve better prediction accuracy, several main challenges remain in training recommender systems. First, smooth structure is contained in the user-item interactions, yet they are not fully utilized in model training. For instances, friends tend to share similar interests and preferences on various items, and relevant items tend to receive comparable ratings from users. Particularly, Figure 1 for the *Last.fm* data indicates that friends behave more similarly than randomly selected user pairs, which can be well characterized by a smoothing pattern of users' preference on items over the user-item specific network. However, such a user-item specific network usually involves a large number of parameters, which imposes great challenges in model building without a sufficient amount of observations. Therefore, how to incorporate the user-item specific network into a recommender system remains one key factor for improving prediction accuracy. Second, covariate information such as demographic and social network information of users, tags and content information of items, can be easily accessible (Nguyen and Zhu, 2013). In the existing literature, a number of methods have been proposed to utilize the side and relational information through Laplacian regularization (Gu et al., 2010), Gaussian generative models (Zhou et al., 2012), probability propagation (Yang et al., 2011), aggregating regression (Demir et al., 2017; Zhao et al., 2016; Zhao and Guo, 2017), Markov logic network (Salakhutdinov and Mnih, 2008; Richardson and Domingos, 2006), and the probabilistic soft logic (Kouki et al., 2015; Bach et al., 2017). Yet, most of the aforementioned methods are rather ad hoc and lack theoretical justification. Mainly, it remains largely unknown how the users' social networks and items' tagging information impact the accuracy of prediction. More importantly, a rating mechanism usu-

©2019 Ben Dai, Junhui Wang, Xiaotong Shen and Annie Qu.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-629.html.

ally does not follow a parametric form in terms of covariates, which is commonly assumed in the literature due to the lack of sufficient observations for each user and item. How to employ a nonparametric approach to model the covariate-assisted recommender system in a high-dimensional setting continues to be an open question. Third, missing completely at random (MCAR) is often assumed by existing methods, leading to inaccurate prediction as the MCAR assumption is typically unrealistic for recommender systems. For example, users tend not to rate items that are of little interest to them, as illustrated in Figure 4. Fourth, most methods fail to recommend for new users or items without any observed ratings, which is referred as the "cold-start" problem. Thus, utilizing the covariate information to fully and efficiently solve the "cold-start" problem is attractive in devising recommender systems.

In this paper, we propose a novel approach based on the idea of a similarity-based neighborhood system pooling similar user-item pairs to improve prediction performance. Specifically, for each user-item pair, the proposed approach incorporates similar observed pairs through kernel weighting based on covariates as well as a user-item specific network. The weight function quantifies a smooth rating mechanism in terms of the closeness of continuous covariates within a neighborhood of the connected user-item pairs in the network. One novelty of the proposed approach is that it builds discrete covariates into a user-item specific network with the same discrete covariate values corresponding to connectivity. This enables us to handle high-dimensional covariates while not being burdened by the "curse of dimensionality." Unlike existing methods (Zhu et al., 2016; Bi et al., 2017), our approach is nonparametric, yet it goes beyond the traditional nonparametric framework which focuses primarily on continuous covariates. This provides a flexible framework to handle continuous covariates, discrete covariates and networks all together without specifying a functional relation. Moreover, the proposed approach also tackles the "cold-start" issue as it utilizes observed pairs in the neighborhood of any new user-item pairs in prediction.

The proposed approach takes full advantage of the smoothing pattern of the rating mechanism over covariates, while integrating user-item dependencies through user-item specific networks into a recommender system nonparametrically. This leads to a higher prediction accuracy for a recommender, as demonstrated in the numerical examples in Section 5. Significantly, our approach outperforms the state-of-the-art prediction performance for the *Last.fm* music benchmark dataset by nearly 20%. Additionally, we perform an error analysis, showing that the error rate of the proposed method is governed by the degree of smoothness of a neighborhood system with respect to continuous covariates, given that the grouping information is precisely defined by discrete covariates and networks. Most critically, as suggested by Theorem 1, the method performs well even in a high-dimensional situation in which the overall size of observed ratings is of the same magnitude as the number of unknown parameters.

The rest of the paper is organized as follows. Section 2 briefly introduces the regularized latent factor model, Section 3 presents the proposed recommender system and its implementation via the alternating least squares algorithm. Section 4 establishes the asymptotic results for the proposed method. Section 5 examines the numerical performance of the proposed method in simulation studies and a real application to the *Last.fm* dataset (`http://www.last.fm`). A brief summary is given in Section 6, and the Appendix contains the technical proofs.

2. Regularized latent factors

In this section, we provide the notations in recommender systems, and the framework of a regularized latent factor model. Consider a recommender system of n users' preference scores on m items, where r_{ui} denotes the preference score of user u on item i. Suppose a user-item specific covariate vector $\boldsymbol{x}_{ui} \in \mathcal{X} \subset \mathbb{R}^d$ is observed (e.g., a user's demographics and an item's content information). One key challenge for training a recommender system is that the preference matrix $R = (r_{ui}) \in \mathbb{R}^{n\times m}$ is only partially observed with a high missing percentage. Denote the index set of observed preference scores as Ω, then $|\Omega| \ll nm$.

A recommender system can be formulated in the framework of a latent factor model:

$$r_{ui} = \theta_{ui} + \epsilon_{ui} = \boldsymbol{p}_u^T \boldsymbol{q}_i + \varepsilon_{ui}, \quad 1 \le u \le n, 1 \le i \le m, \tag{1}$$

where $\theta_{ui} = E(r_{ui})$ is the expected preference score of a user-item pair (u, i), and ε_{ui} is independent from $\boldsymbol{x}_{ui}$ with mean zero and finite variance. The latent factor model assumes that θ_{ui} can be represented by user and item latent factors: $\theta_{ui} = \boldsymbol{p}_u^T \boldsymbol{q}_i$, where $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$ are K-dimensional latent vectors representing user u's preference and item i's profile, respectively, and K is the number of latent factors for both users and items, which is also the rank in the latent factor model (Mukherjee et al., 2015).

To estimate these personalized parameters, a regularized SVD method (Paterek, 2007; Zhu et al., 2016) estimates $\boldsymbol{P} = (\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n)^T$ and $\boldsymbol{Q} = (\boldsymbol{q}_1, \cdots, \boldsymbol{q}_m)^T$ by

$$\min_{\boldsymbol{P},\boldsymbol{Q}} \frac{1}{|\Omega|} \sum_{(u,i)\in\Omega} (r_{ui} - \boldsymbol{p}_u^T \boldsymbol{q}_i)^2 + \lambda \Big\{ \sum_{u=1}^{n} J(\boldsymbol{p}_u) + \sum_{i=1}^{m} J(\boldsymbol{q}_i) \Big\}, \tag{2}$$

where λ is a nonnegative tuning parameter, and $J(\cdot)$ can be any penalty function such as the L_q-penalty with $q = 0, 1, 2$ (Zhu et al., 2016) or the alignment penalty (Nguyen and Zhu, 2013). Note that r_{ui} in (2) is often replaced by the residual $r_{ui} - \mu - \boldsymbol{x}_{ui}^T \boldsymbol{\beta}$, where $(\mu, \boldsymbol{\beta})$ is a vector of regression coefficients to be minimized in (2). Alternatively, $\boldsymbol{p}_u = \boldsymbol{s}_u - \boldsymbol{x}_u^T \boldsymbol{\alpha}$ and $\boldsymbol{q}_i = \boldsymbol{t}_i - \boldsymbol{x}_i^T \boldsymbol{\beta}$, where $(\boldsymbol{s}_u, \boldsymbol{t}_i)$ are latent factors, and $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ are regression coefficients to incorporate the covariate effects (Agarwal et al., 2011).

There are a number of major challenges for the regularized SVD model in (2). First, a user-item specific network is often available to capture user-item dependence but is ignored in (2). Here networks consists of information from existing users' social network, or item network, or the network constructed from available discrete covariates of the user-item pair (u, i). It can provide additional information regarding the preference similarity between connected pairs. This is the case for the *Last.fm* data, where a user specific network impacts users' preference on items, indicated by Figure 1. Second, a linear model in (2) may be inadequate to incorporate user and item covariates, especially when the linearity assumption is violated. Third, the objective function (2) assumes implicitly that missing occurs completely at random so that the first loss function in (2) can approximate the loss function for the complete data. When the missing is informative, missing characteristics such as the missing pattern for each user can be utilized for subgrouping (Bi et al., 2017). Fourth, the regularized SVD in (2) suffers from the "cold-start" problem for new users or items in the absence of observed ratings. For instance, if a user u or an item i is completely missing from Ω, the corresponding $\boldsymbol{p}_u$ or $\boldsymbol{q}_i$ is estimated incorrectly as 0 due to the penalty $J(\cdot)$ in (2).

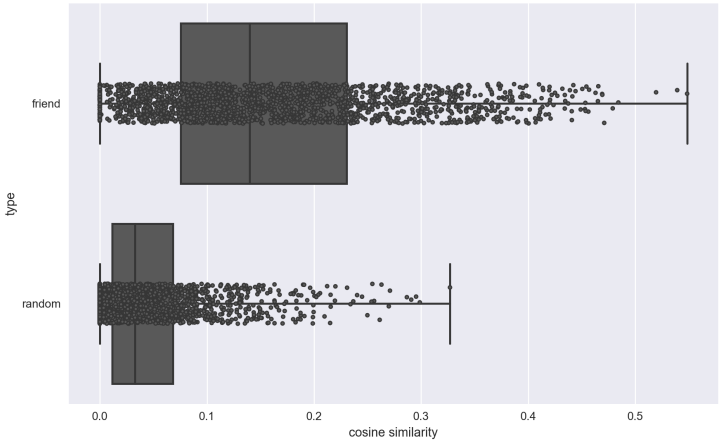


Figure 1: Comparison of the cosine similarity of logarithm of users' listening counts vector between friends and randomly selected users in the *Last.fm* dataset.

3. Methods

This section introduces a smooth recommender system, which incorporates user and/or item specific covariates and the user-item specific network information to improve prediction accuracy. The proposed method utilizes informative observations for each user-item pair for personalized prediction and resolving the "cold-start" problem.

3.1. Proposed method

One key strategy of the proposed method is to pool information across user-item pairs to improve prediction accuracy through increasing effective sample size. In contrast to grouping approaches based on user-item-specific information (Bi et al., 2017; Masthoff, 2011), the proposed recommender system integrates similar observed pairs $(u', i') \in \Omega$ for each (u, i) through a weight function, regardless of whether (u, i) is observed or not. This allows for nonlinear or nonparametric modeling of the relation between user-item preferences and latent factors, which is more flexible than the linear modeling in (2).

The proposed weight function is constructed based on the closeness between continuous covariates in addition to a user-item specific network. To precisely describe the network structure, we define an indicator $S^{ui}_{u'i'} = S^u_{u'} S^i_{i'} = 1$ if there exist edges connecting u and u' in a user-specific social network and i and i' in an item-specific network, and $S^{ui}_{u'i'} = 0$ otherwise. Moreover, we integrate the above user-item specific network with existing user social network, item network, as well as information from discrete covariates. For example,

in the *Last.fm* dataset, r_{ui} is the listening count of a user-artist pair (u, i), and $\boldsymbol{x}_{ui}$ is the pattern of listening counts for user u and artist i. The available network information is given as $S^{ui}_{u'i'} = S^u_{u'} S^i_{i'}$, where $S^u_{u'}$ represents a user's friendship network, and $S^i_{i'}$ is constructed from artists' tag information with $S^i_{i'} = 1$ indicating artists i and i' share the same tags, and $S^i_{i'} = 0$ otherwise.

Using local likelihood via smoothing weights (Tibshirani and Hastie, 1987; Fan and Gijbels, 1996), we propose the following cost function for a smooth neighborhood recommender system:

$$\mathcal{L}(\boldsymbol{P}, \boldsymbol{Q}) = \frac{1}{nm} \sum_{u=1}^{n} \sum_{i=1}^{m} \Big(\sum_{(u',i') \in \Omega} \omega_{ui,u'i'} (r_{u'i'} - \boldsymbol{p}_u^T \boldsymbol{q}_i)^2 \Big) + \lambda_1 \sum_{u=1}^{n} J(\boldsymbol{p}_u) + \lambda_2 \sum_{i=1}^{m} J(\boldsymbol{q}_i), \quad (3)$$

where $J(\cdot)$ is a general penalty, and λ_1 and λ_2 are two nonnegative tuning parameters. It is also assumed that $\sum_{(u',i') \in \Omega} \omega_{ui,u'i'} = 1$. Lemma 1 below gives an equivalent form of (3).

Lemma 1 *The cost function* $\mathcal{L}(\boldsymbol{P}, \boldsymbol{Q})$ *in* (3) *is equivalent to*

$$\frac{1}{nm} \sum_{u=1}^{n} \sum_{i=1}^{m} \Big(\sum_{(u',i') \in \Omega} \omega_{ui,u'i'} r_{u'i'} - \boldsymbol{p}_u^T \boldsymbol{q}_i \Big)^2 + \lambda_1 \sum_{u=1}^{n} J(\boldsymbol{p}_u) + \lambda_2 \sum_{i=1}^{m} J(\boldsymbol{q}_i). \quad (4)$$

The choice of $\omega_{ui,u'i'}$ is critical to properly measure the similarity between (u, i) and (u', i'). We introduce a weight function to define the smooth neighborhood of (u, i):

$$\omega_{ui,u'i'} = \frac{\mathcal{K}_h(\boldsymbol{x}_{ui}, \boldsymbol{x}_{u'i'}) S^{ui}_{u'i'}}{\sum_{(u',i') \in \Omega} \mathcal{K}_h(\boldsymbol{x}_{ui}, \boldsymbol{x}_{u'i'}) S^{ui}_{u'i'}}, \quad (5)$$

where $\omega_{ui,u'i'}$ involves only observed pairs (u', i')'s with $S^{ui}_{u'i'} = 1$. In (5), if $\boldsymbol{x}_{ui}$ or $S^{ui}_{u'i'}$ is absent, $\mathcal{K}_h(\boldsymbol{x}_{ui}, \boldsymbol{x}_{u'i'})$ or $S^{ui}_{u'i'}$ can be set as 1 correspondingly. The kernel function is set as $\mathcal{K}_h(\boldsymbol{x}_{ui}, \boldsymbol{x}_{u'i'}) = \mathcal{K}(h^{-1} \|\boldsymbol{x}_{ui} - \boldsymbol{x}_{u'i'}\|_2)$ measuring the closeness between $\boldsymbol{x}_{ui}$ and $\boldsymbol{x}_{u'i'}$, where the choice of the L_2-norm reduces the dimension of the covariate space and other choices of distance may be also considered. Here $h > 0$ is the window size and $\mathcal{K}(\cdot)$ is a kernel whose degree of smoothness reflects prior knowledge about how the true preference varies in terms of $\|\boldsymbol{x}_{ui} - \boldsymbol{x}_{u'i'}\|_2$. Note that this is different from standard kernel smoothing (Fan and Gijbels, 1996; Delaigle and Hall, 2010), in that the smooth neighborhood is constructed based on continuous and discrete covariates as well as user-item specific networks, whereas standard kernel smoothing focuses primarily on smoothing over continuous covariates.

The proposed framework has the following advantages. First, the user-item specific covariates and network structures are integrated in constructing the neighborhood for (u, i) pairs. Thus the effective sample size for (u, i) increases when pooling information from its neighborhood through $\sum_{(u',i') \in \Omega} \omega_{ui,u'i'} r_{u'i'}$ in (4). Second, it solves the "cold-start" problem and yields more accurate estimators of $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$ for all u's and i's by leveraging dependencies among users and items, expressed in terms of user-specific social networks and items' tagging information. This is evident from (4), since even for an unobserved (u, i), $\sum_{(u',i') \in \Omega} \omega_{ui,u'i'} r_{u'i'}$ is a weighted average of preference scores over its neighborhood. On this ground, a recommendation can be made by estimating $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$ for any new user-item

pair (u,i). Third, the non-ignorable missing can be addressed through a covariate-adjusted neighborhood associated with missingness and users' preferences. For instance, each user's and item's missing percentages and the percentiles of their observed ratings can be modeled nonparametrically as covariates in defining $\omega_{ui,u'i'}$.

3.2. Scalable computation

To solve large-scale optimization in (3), we employ a "divide-and-conquer" type of alternating least squares (ALS) algorithm, with a principle of solving many small penalized regression problems iteratively. This permits parallel and efficient computation. The ALS method has been extensively investigated in the literature (Carroll and Chang, 1970; Friedman and Stuetzle, 1981), and the divide-and-conquer strategy is also employed in (Zhu et al., 2016) for the parallelization of (2).

The computational strategy of ALS is to break large-scale optimization into multiple small subproblems by alternatively fixing either $\boldsymbol{p}_u$ or $\boldsymbol{q}_i$, where each subproblem is a simple penalized least squares regression and can be solved analytically with $J(\cdot) = \|\cdot\|_2^2$. Note that this strategy is applicable as long as $J(\cdot)$ is separable for $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$.

For illustration, consider $J(\cdot) = \|\cdot\|_2^2$. At iteration k, $\widehat{\boldsymbol{Q}}^{(k)}$ is fixed and the latent factor $\boldsymbol{p}_u$ is updated as $\hat{\boldsymbol{p}}_u^{(k+1)} = \text{argmin}_{\boldsymbol{p}_u} \sum_i \Big(\sum_{(u',i')\in\Omega} \omega_{ui,u'i'} (r_{u'i'} - \boldsymbol{p}_u^T \hat{\boldsymbol{q}}_i^{(k)})^2 \Big) + \lambda_1 \|\boldsymbol{p}_u\|_2^2$. Similarly, with fixed $\widehat{\boldsymbol{P}}^{(k+1)}$, $\boldsymbol{q}_i$ is updated as $\hat{\boldsymbol{q}}_i^{(k+1)} = \text{argmin}_{\boldsymbol{q}_i} \sum_u \Big(\sum_{(u',i')\in\Omega} \omega_{ui,u'i'} (r_{u'i'} - (\hat{\boldsymbol{p}}_u^{(k+1)})^T \boldsymbol{q}_i)^2 \Big) + \lambda_2 \|\boldsymbol{q}_i\|_2^2$. Then each subproblem is solved analytically,

$$\hat{\boldsymbol{p}}_u^{(k+1)} = \Big(\sum_i \hat{\boldsymbol{q}}_i^{(k)} (\hat{\boldsymbol{q}}_i^{(k)})^T + \lambda_1 \boldsymbol{I}_K \Big)^{-1} \Big(\sum_i \bar{r}_{ui}^{\Omega} \hat{\boldsymbol{q}}_i^{(k)} \Big), \tag{6}$$

$$\hat{\boldsymbol{q}}_i^{(k+1)} = \Big(\sum_u \hat{\boldsymbol{p}}_u^{(k+1)} (\hat{\boldsymbol{p}}_u^{(k+1)})^T + \lambda_2 \boldsymbol{I}_K \Big)^{-1} \Big(\sum_u \bar{r}_{ui}^{\Omega} \hat{\boldsymbol{p}}_u^{(k+1)} \Big), \tag{7}$$

where $\bar{r}_{ui}^{\Omega} = \sum_{(u',i')\in\Omega} \omega_{ui,u'i'} r_{u'i'}$ is a weighted rating for (u,i) over the neighborhood, and $\boldsymbol{I}_K$ is a $K \times K$ identity matrix. The iterative updating is continued until a termination criterion is reached. Once the solution $\{\hat{\boldsymbol{p}}_u, \hat{\boldsymbol{q}}_i\}_{1\le u\le n; 1\le i\le m}$ is obtained, the final predicted preference is $\hat{r}_{ui} = \hat{\boldsymbol{p}}_u^T \hat{\boldsymbol{q}}_i$.

It follows from Chen et al. (2012) that the algorithm converges to a stationary point $(\bar{\boldsymbol{P}}, \bar{\boldsymbol{Q}})$ of $\mathcal{L}(\boldsymbol{P}, \boldsymbol{Q})$ in (3), where $\bar{\boldsymbol{P}} = \text{argmin}_{\boldsymbol{P}} \mathcal{L}(\boldsymbol{P}, \bar{\boldsymbol{Q}})$, and $\bar{\boldsymbol{Q}} = \text{argmin}_{\boldsymbol{Q}} \mathcal{L}(\bar{\boldsymbol{P}}, \boldsymbol{Q})$. This is due to the nonconvex minimization in addition to having many missing observations. Moreover, each update of $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$ in (6) and (7) can be computed in a parallel fashion. This can substantially speed up the computation, particularly when K is small but m and n are large.

The overall computational complexity of the algorithm is no greater than $O\big((nmK^2 + (n+m)K^3) I_{ALS}\big)$, where I_{ALS} is the number of iterations in the ALS algorithm. In our implementation, the algorithm is coded through PyMP, which is a Python version of OpenMP in C, and can handle a dataset with a size up to the order of 10^8 on a quad-core computer with one 3.40GHz CPU and 8G memory.

4. Theory

This section presents theoretical properties to quantify the asymptotic behavior of the proposed recommender system. Particularly, we show the convergence rate in prediction error of the proposed system.

Let the true and estimated parameters be $\theta_{ui}^0 = \boldsymbol{p}_u^{0\,T} \boldsymbol{q}_i^0$ and $\hat{\theta}_{ui} = \hat{\boldsymbol{p}}_u^T \hat{\boldsymbol{q}}_i$, where $\hat{\boldsymbol{p}}_u$ and $\hat{\boldsymbol{q}}_i$ are estimated latent factors; $u = 1, \cdots, n$; $i = 1, \cdots, m$. Note that the representation is unique with respect to θ_{ui}, although the decomposition of $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$ may not be unique. Let $\hat{\boldsymbol{\theta}} = (\hat{\theta}_{ui}) \in \mathbb{R}^{n\times m}$ and $\boldsymbol{\theta}^0 = (\theta_{ui}^0) \in \mathbb{R}^{n\times m}$, the prediction accuracy of $\hat{\boldsymbol{\theta}}$ is defined by the root mean square error:

$$\text{RMSE}(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^0) = \Big(\frac{1}{nm} \sum_{u=1}^{n} \sum_{i=1}^{m} \big(\hat{\theta}_{ui} - \theta_{ui}^0\big)^2 \Big)^{1/2}.$$

We require the following technical assumptions.

Assumption A. There exist constants $c_1 > 0$ and $\alpha > 0$ such that for any (u,i) and (u',i') , $\big|\theta_{ui}^0 - \theta_{u'i'}^0\big| \le c_1 \sqrt{K} \max\{\|\boldsymbol{x}_{ui} - \boldsymbol{x}_{u'i'}\|_2^{\alpha}, I(S_{u'i'}^{ui} = 0)\}$, where the corresponding expression in the maximum operator is set as 0 if $\boldsymbol{x}_{ui}$ or $S_{u'i'}^{ui}$ is absent.

Assumption A defines the smoothness of θ_{ui}^0 in terms of the continuous covariate $\boldsymbol{x}_{ui}$ in the presence of connected user-item pairs in the network. As a special case, if $\boldsymbol{x}_{ui}$ is absent, Assumption A degenerates to $\theta_{ui}^0 = \theta_{u'i'}^0$ for pairs with $S_{u'i'}^{ui} = 1$. This assumption is mild when all covariates are available, and is relatively more restrictive when $\boldsymbol{x}_{ui}$ is absent as it pushes the model to be a parametric one with only a finite number of unknown θ's. It is necessary to capture the smooth property of θ over the network structure, and similar assumptions are widely used in nonparametric regression (Vieu, 1991; Wassermann, 2006) and kernel density estimation (Stone, 1984; Marron and Padgett, 1987).

Assumption B. The continuous covariate $\boldsymbol{x}$ has a bounded support $\mathcal{X}$, and the error term ε_{ui} has a sub-Gaussian distribution with variance σ^2.

Assumption B is a regularity condition for the underlying probability distribution, and widely used in literature (Ma and Huang, 2017; Lin et al., 2017). Further, assume that $\{\boldsymbol{x}_{ui}, \Delta_{ui}\}_{1\le u\le n, 1\le i\le m}$ are independent and identically distributed, but the distribution of Δ_{ui} may depend on $\boldsymbol{x}_{ui}$. We denote the parameter space $\boldsymbol{\Gamma}(L)$ to be $\{\boldsymbol{\theta} : \|\boldsymbol{\theta}\|_\infty \le L\}$, where $\|\cdot\|_\infty$ is the uniform norm, and L is chosen so that $\boldsymbol{\theta}^0 \in \boldsymbol{\Gamma}(L)$. To accurately utilize the network information, we set $\omega_{ui,u'i'} = 0$ if $S_{u'i'}^{ui} = 0$. Theorem 1 establishes an upper bound for the estimation error of the proposed recommender system, where the convergence rate is determined by the size of the preference matrix nm, the number of parameters K, the size of the observed ratings $|\Omega|$, the tuning parameters λ_1 and λ_2, and the window size h.

Theorem 1 *Suppose that Assumptions A and B are satisfied. For some constant $c_2 > 0$, let $\kappa_1 = \max_{u,i} \sum_{(u',i')\in\Omega} \omega_{ui,u'i'} \|\boldsymbol{x}_{ui} - \boldsymbol{x}_{u'i'}\|_2^{\alpha}$, and $\kappa_2 = \max_{u,i} \sum_{(u',i')\in\Omega} \omega_{ui,u'i'}^2$, then*

$$P\big(RMSE(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^0) \ge \eta\big) \le \exp\big\{ -\frac{c_2 \eta^2}{\kappa_2} + \log(nm) \big\},$$

provided that $\eta \ge \max\big\{\sqrt{K}\kappa_1, \sqrt{\kappa_2}\big\} \log(nm)$ and $\lambda_1 \sum_{u=1}^n J(\boldsymbol{p}_u^0) + \lambda_2 \sum_{i=1}^m J(\boldsymbol{q}_i^0) \le \eta^2/4$. The convergence rate then becomes $RMSE(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^0) = O_p(\max\big\{\sqrt{K}\kappa_1, \kappa_2^{1/2}\big\} \log(nm))$, where κ_1 and κ_2 tend to zero and can be computed for some specific weights.

Theorem 1 provides a general upper bound of RMSE$(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^0)$, which may vary by the choice of weights $\omega_{ui,u'i'}$ and the two quantities κ_1 and κ_2. In the following, an explicit convergence rate is given for $\omega_{ui,u'i'}$ defined in (5) under some additional assumptions.

Let $\Delta_{ui} \in \{0, 1\}$ be a binary variable, with $\Delta_{ui} = 1$ indicating that r_{ui} is observed and 0 otherwise. Assume that $\{\boldsymbol{x}_{ui}, \Delta_{ui}\}_{1\le u\le n, 1\le i\le m}$ are independent and identically distributed, but the distribution of Δ_{ui} may depend on $\boldsymbol{x}_{ui}$.

Assumption C. For any (u, i) and (u', i'), $P(S^{ui}_{u'i'} = 1|\Delta_{u'i'} = 1)$ is bounded away from zero, and the conditional density $f_{U^{ui}_{u'i'}|S^{ui}_{u'i'}=1,\Delta_{u'i'}=1}$ is continuous and bounded away from zero, where $U^{ui}_{u'i'} = \|\boldsymbol{x}_{ui} - \boldsymbol{x}_{u'i'}\|_2$.

Assumption C is necessary for tackling the "cold-start" problem, and similar assumptions are also widely used in the local smoothing technique (Chen et al., 2014; Scott, 2015). It ensures that for any pair (u, i), the probability of $\Delta_{ui} = 1$ may depend on covariates $\boldsymbol{x}_{ui}$ and $S^{ui}_{u'i'}$, and that the corresponding neighboring pairs are observed with positive probability. In fact, it suffices to assume that $P(S^{ui}_{u'i'} = 1|\Delta_{u'i'} = 1)$ is bounded away from zero with certain order, and similar results can be obtained with more involved derivation.

Assumption D. There exists a constant c_2 such that the nonnegative kernel $\mathcal{K}(\cdot)$ satisfies

$$\max\Big\{\int_0^\infty \mathcal{K}^2(u)du, \int_0^\infty \mathcal{K}(u)u^\alpha du\Big\} \le c_3.$$

Assumption D is a standard assumption for smoothing kernels. Notably, the choice of kernel should match up the smoothness at an order α of $\boldsymbol{\theta}^0$. Alternatively, to effectively employ the smoothing pattern of θ, the decay rate of the chosen kernel may be larger than α, to filter out more distant user-item pairs and preserve more reliable local neighborhood structure. For example, the Gaussian kernel has an exponential decay rate, and always satisfies the inequality conditions in Assumption D.

Corollary 1 *Suppose that Assumptions A-D are satisfied. For $\omega_{ui,u'i'}$ defined in* (5)*, then the convergence rate of RMSE$(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^0)$ becomes RMSE$(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^0) = O_p\big(|\Omega|^{-\frac{\alpha}{2\alpha+1}} K^{\frac{1}{2(2\alpha+1)}} \log(nm)\big)$.*

Note that this convergence rate is intriguing compared with some existing results. Particularly, when $K = O(1)$, we have RMSE$(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^0) = O_P\big(|\Omega|^{-\frac{\alpha}{2\alpha+1}} \log(nm)\big)$. For $\alpha > 1/2$, since $|\Omega| \le nm < (n + m)^2$, it leads to a tighter bound than $O_P\Big(\big(\frac{n+m}{|\Omega|}\log(\frac{\sqrt{nm}}{|\Omega|})\big)^{\frac{1}{2}}\Big)$ and $O_P\Big(\big(\frac{n+m}{|\Omega|}\log(m)\log(\frac{|\Omega|}{n+m})\big)^{\frac{1}{2}}\Big)$ established in Bi et al. (2017) and Srebro et al. (2005), respectively. In addition, Theorem 1 still guarantees the convergence of RMSE$(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^0)$, if K goes to infinity at a rate slower than $|\Omega|^{2\alpha}(\log(nm))^{-2(2\alpha+1)}$. In practice, the size of Ω is often much less than nm, and only proportional to $(n + m)K$. For example, in the MovieLens 1M dataset with $K = 10$, $nm = 0.2 \times 10^8, |\Omega| = 10^6$, and $(n + m)K = 10^6$; and in the *Last.fm* dataset, $nm = 0.3 \times 10^8, |\Omega| = 10^6$ and $(n + m)K = 0.2 \times 10^6$. In such cases with $K = O(1)$, the theoretical results in Bi et al. (2017) and Srebro et al. (2005) fail to give a reasonable convergence rate. However, Theorem 1 still yields that RMSE$(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}^0) = O_P\big(|\Omega|^{-\frac{\alpha}{2\alpha+1}} \log(nm)\big)$. Interestingly, if the continuous covariates are absent, then Assumption A is satisfied with $\alpha = \infty$, and there is only a finite number of θ's to be estimated for different discrete covariates. In such cases, Theorem 1 implies that the recommender system can be estimated with a rate $\eta \sim O_P\big(|\Omega|^{-1/2} \log(nm)\big)$.

5. Numerical results

This section examines the performance of the proposed method, denoted as sSVD, in simulations and with a real *Last.fm* dataset. We compare sSVD to several strong competitors, including restricted Boltzmann machines (RBM; Salakhutdinov et al. 2007) or a continuous version of restricted Boltzmann machines (CRBM; Chen and Murray 2003); an iterative soft-threshold matrix completion method (SoftImpute; Hastie et al. 2015); the regularized SVD (rSVD; Paterek 2007); the self-recovered side regression (SSR; Zhao et al. (2016)) and a group-specific SVD approach (gSVD; Bi et al. 2017). Note that the Python codes for RBM and CRBM are publicly available (`https://github.com/yusugomori/DeepLearning/tree/master/python`), the code for SoftImpute is available in the Python package "fancyimpute", the Matlab code for SSR is provided by Zhao et al. (2016), and the R code for gSVD is provided by Bi et al. (2017). Note that RBM and CRBM are essentially the same but differ in implementation only for discrete and continuous responses. Although RBM, SoftImpute and rSVD are not designed to incorporate covariates, we include them in comparison to illustrate the importance of utilizing covariates for personalized prediction.

For tuning parameter selection, we set the learning rate, the momentum rate and the number of hidden units for RBM and CRBM as 0.005, 0.9, and 100, respectively. For rSVD, gSVD and sSVD, we set the tuning parameter K to be the true one, and the optimal λ is determined by a grid search over $\{10^{(\nu-31)/10}; \nu = 1, \cdots, 61\}$. For the proposed sSVD, a Gaussian kernel is used with the window size h being the median distance among all user-item pairs. The predictive performance of all methods is measured by the root mean squared error (RMSE).

5.1. Simulated examples

This subsection investigates the "cold-start" problem and the utility of a user-item specific network on prediction performance. We generate the simulation setting as follows. The dimensions of a rating matrix $\{r_{ui}\}_{1\le u\le n; 1\le i\le m}$ are $n = 501$ and $m = 201$. Let $\boldsymbol{p}_u = (1 + 0.002u)\mathbf{1}_K + N(\mathbf{0}_K, \xi\boldsymbol{I}_K)$ and $\boldsymbol{q}_i = (1 + 0.0075i)\mathbf{1}_K + N(\mathbf{0}_K, \xi\boldsymbol{I}_K)$ for $u = 1, \cdots, n$ and $i = 1, \cdots, m$, where $\mathbf{1}_K$ and $\mathbf{0}_K$ being K-dimensional vectors of ones and zeros, respectively. Here we sample observed ratings at random, where π is the missing rate, and the total number of observed ratings is $|\Omega| = (1 - \pi)nm$. For each user-item pair $(u, i) \in \Omega$, we let u and i be uniformly sampled from $1, \cdots, n$ and $1, \cdots, m$, respectively, and r_{ui} be generated from a truncated normal distribution on $[1, 5]$ with mean $\boldsymbol{p}_u^T\boldsymbol{q}_i$ and standard deviation 0.5. Further, r_{ui} is rounded to the closest integer in $\{1, \cdots, 5\}$ to mimic discrete ratings in practice.

To generate a similarity network for each user-item (u, i) pair, we connect (u, i) with its nearest first-order neighbor indices, say $u - 1$ and $u + 1$ for u and $i - 1$ and $i + 1$ for i. This mimics a user's friendship network. In addition, we introduce a "cold-start" rate ρ to examine the prediction accuracy of the proposed method on new users and items. Specifically, we randomly select ρ-proportion of users and items, and retain all of their corresponding ratings in a testing set to resemble the "cold-start" phenomenon. The remaining ratings are randomly split into training, tuning and testing sets with 60%, 15%, and 25% of the observations, respectively. In simulations, we choose $\xi = 0.1$, $K = 3$ or 6, the missing rate $\pi = .8, .9, .95$, and the "cold-start" rate $\rho = 0, 0.1, 0.2$. For the proposed sSVD, the

weight function is set as $\omega_{ui,u'i'} = S^{ui}_{u'i'}$, which is constructed based on the adjacency of each user-item pairs. For gSVD, we use the ranking percentages of each user and item as covariates according to Bi et al. (2017), but the user-item specific network information is not incorporated.

Table 1: *Averaged RMSEs of various methods and their estimated standard deviations in parentheses on the simulated examples over 50 simulations. Here RBM, SoftImput, rSVD, gSVD, sSVD denote: restricted Boltzmann machines (Salakhutdinov et al., 2007), SoftImput method (Hastie et al., 2015), regularized SVD method (Paterek, 2007), group-specific SVD method (Bi et al., 2017) and the proposed method, respectively. The best performer in each setting is bold-faced.*

	RBM	SoftImpute	SSR	rSVD	gSVD	sSVD
Use of covariate	No	No	Yes	No	Yes	Yes
K = 3, π = 0.8						
$\rho = 0.0$	0.842(0.001)	1.082(.003)	2.471(.002)	0.323(.000)	**0.302(.000)**	0.364(.001)
$\rho = 0.1$	0.911(0.003)	2.100(.007)	2.570(.004)	1.112(.009)	0.472(.003)	**0.371(.001)**
$\rho = 0.2$	0.943(0.002)	2.404(.007)	2.650(.014)	1.331(.007)	0.578(.009)	**0.375(.001)**
K = 3, π = 0.9						
$\rho = 0.0$	0.848(0.002)	2.146(0.003)	2.617(.002)	0.498(0.002)	**0.351(0.002)**	0.368(0.001)
$\rho = 0.1$	0.915(0.002)	2.486(0.006)	2.674(.003)	1.170(0.010)	0.525(0.007)	**0.378(0.001)**
$\rho = 0.2$	0.948(0.003)	2.261(0.005)	2.697(.005)	1.418(0.007)	0.622(0.006)	**0.388(0.001)**
K = 3, π = 0.95						
$\rho = 0.0$	0.861(0.002)	2.554(0.005)	2.693(.004)	1.002(0.004)	0.442(0.002)	**0.379(0.001)**
$\rho = 0.1$	0.923(0.003)	2.668(0.006)	2.716(.005)	1.445(0.009)	0.707(0.017)	**0.407(0.002)**
$\rho = 0.2$	0.956(0.003)	2.722(0.005)	2.733(.004)	1.636(0.007)	0.849(0.014)	**0.440(0.002)**
K = 6, π = 0.8						
$\rho = 0.0$	0.843(0.001)	1.086(0.003)	2.472(.002)	0.340(0.001)	**0.302(0.001)**	0.367(0.001)
$\rho = 0.1$	0.914(0.003)	2.113(0.006)	2.584(.004)	0.823(0.008)	0.472(0.005)	**0.374(0.001)**
$\rho = 0.2$	0.943(0.003)	2.414(0.006)	2.634(.004)	0.984(0.007)	0.568(0.005)	**0.382(0.001)**
K = 6, π = 0.9						
$\rho = 0.0$	0.851(0.002)	2.156(0.004)	2.616(.003)	0.451(0.001)	**0.350(0.001)**	0.373(0.001)
$\rho = 0.1$	0.908(0.002)	2.492(0.005)	2.667(.005)	0.872(0.007)	0.529(0.005)	**0.387(0.001)**
$\rho = 0.2$	0.946(0.002)	2.625(0.004)	2.697(.003)	1.030(0.006)	0.619(0.006)	**0.403(0.001)**
K = 6, π = 0.95						
$\rho = 0.0$	0.857(0.003)	2.558(0.005)	2.697(.003)	0.713(0.003)	0.442(0.002)	**0.396(0.002)**
$\rho = 0.1$	0.925(0.002)	2.679(0.006)	2.726(.005)	1.015(0.009)	0.696(0.019)	**0.422(0.002)**
$\rho = 0.2$	0.955(0.003)	2.717(0.005)	2.733(.004)	1.154(0.006)	0.866(0.012)	**0.460(0.002)**

Table 1 indicates that the proposed sSVD achieves the highest predictive accuracy compared against its competitors in 14 out of 18 different settings, whereas its accuracy is close to the best performer gSVD in the remaining four settings. Note that these four settings correspond to no "cold-start" user-item pairs in a linear situation. This simulation result is anticipated as only sSVD is capable of utilizing user-specific social networks. Interestingly, gSVD, without using the user-specific networks, performs well as it is able to capture user-user dependencies with respect to ranking due to strong dependency from the neighboring user-specific friendship networks. By comparison, SSR has a comparable performance as SoftImput, which is much worse than rSVD, gSVD, and sSVD, partly because it fails to account for non-ignorable missing.

Compared with the methods without using covariates such as rSVD and SoftImpute, sSVD yields superior performance in most scenarios, and the percentages of improvement can be as high as 83.1%. In contrast to RBM, SoftImpute and SSR, the percentages of improvement from the sSVD range from 51.8% to 83.2%. In addition, sSVD is robust

against high missing rates and high "cold-start" rates as measured by the missing rate π and the cold-start rate ρ. As π and ρ increase, the performance of sSVD is stable, whereas the performances of competitors deteriorate rapidly. In other words, the improvement of sSVD over its competitors is more significant when π and ρ are large. For example, the largest amount of improvement of sSVD over the second best performer gSVD is 46.9% when $K = 6$, $\pi = .95$, and $\rho = .2$.

Moreover, we examine the robust aspect of the proposed sSVD with respect to the degree of smoothness of ratings in a neighborhood and the degree of sparseness of a user-specific network. Specifically, given $K = 6$ and $\pi = 0.95$, we contaminate the neighborhood structure by increasing the noise level $\xi = 0.1, 0.2, 0.4, 0.6$, and allow the sparsity ratio $\pi_{network} = 0.0, 0.2, 0.5, 1.0$ for user-item specific networks, where the sparseness ratio is the probability that the edges connecting each pair of users or items are removed from the original network. Thus as $\pi_{network}$ increases, the resultant network becomes sparser, and the proposed sSVD degenerates to rSVD when $\pi_{network} = 1.0$. As suggested by Table 2, the performance of sSVD deteriorates as the contamination level $\pi_{network}$ escalates, yet still outperforming rSVD when $\xi = 0.1, 0.2$. On the other hand, rSVD outperforms the proposed sSVD when $\xi \geq 0.4$, which is anticipated because the proposed sSVD relies on the smoothness structure of the ratings. Moreover, it is evident that the proposed sSVD is less affected by the "cold-start" users and items when $\pi_{network}$ decreases.

Table 2: *Averaged RMSEs and estimated standard deviations over 50 simulations for the proposed method when the error standard deviation is 0.1, 0.2, 0.4, or 0.6, and the "cold-start" rate ρ is 0.0, 0.1, or 0.2, and the network structure missing rate $\pi_{network}$ is 0.0, 0.2, 0.5, or 1.0.*

	$\xi = 0.1$	$\xi = 0.2$	$\xi = 0.4$	$\xi = 0.6$
$\pi_{network} = 0.0$				
$\rho = 0.0$	0.396(0.002)	0.589(0.003)	0.984(0.005)	1.547(0.010)
$\rho = 0.1$	0.422(0.002)	0.665(0.004)	1.032(0.014)	1.596(0.015)
$\rho = 0.2$	0.460(0.002)	0.716(0.005)	1.094(0.009)	1.683(0.016)
$\pi_{network} = 0.2$				
$\rho = 0.0$	0.512(0.002)	0.608(0.003)	0.987(0.007)	1.518(0.010)
$\rho = 0.1$	0.599(0.004)	0.694(0.005)	1.062(0.010)	1.622(0.016)
$\rho = 0.2$	0.648(0.006)	0.746(0.004)	1.129(0.009)	1.660(0.013)
$\pi_{network} = 0.5$				
$\rho = 0.0$	0.569(0.003)	0.648(0.004)	1.000(0.009)	1.516(0.014)
$\rho = 0.1$	0.660(0.004)	0.743(0.007)	1.092(0.008)	1.629(0.011)
$\rho = 0.2$	0.711(0.007)	0.796(0.006)	1.127(0.010)	1.670(0.012)
$\pi_{network} = 1.0$				
$\rho = 0.0$	0.713(0.003)	0.745(0.006)	0.952(0.009)	1.397(0.008)
$\rho = 0.1$	1.015(0.009)	1.046(0.011)	1.289(0.014)	1.686(0.015)
$\rho = 0.2$	1.154(0.006)	1.193(0.009)	1.415(0.013)	1.790(0.020)

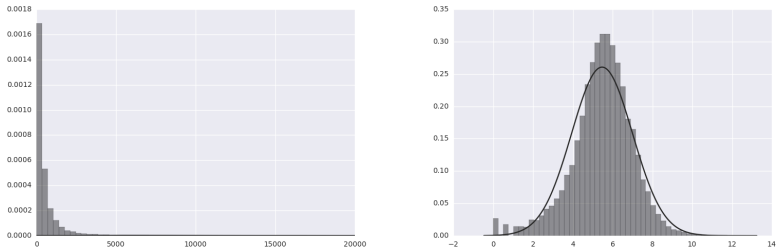
In summary, the simulation studies illustrate that integration of user-item specific networks into latent factor modeling can enhance the accuracy of prediction and tackle the "cold-start" issue. In this regard, sSVD is a more effective recommender system than the existing methods in the literature.

5.2. Music data from *Last.fm*

In this section, we analyze a online music dataset from the *Last.fm* (`http://www.last.fm`), which was released in the second International Workshop HetRec 2011 (`http://ir.ii.uam.es/hetrec2011`). This dataset contains user-artist listening counts, social networks among 1,892 users, and tagging information for 17,632 artists. The user-artist listening counts contain 92,834 tuples defined by `[user, artist, listeningCount]`, where each user listens to an average of about 49 artists, and each artist has been listened to by about 5 users on average. Note that the observation rate in this dataset is below 0.28%, leading to a severe "cold-start" problem.

For this application, we focus on utilizing user-item specific covariates and networks to solve the "cold-start" issue. Specifically, in the *Last.fm* dataset, a user-specific social network is available based on 12,717 bi-directional user friendships with an average of about 13 friends per user. Moreover, there are 186,479 tags given by users to artists, with an average of about 99 tags per user, and about 15 tags per artist. A typical tag of an artist gives a short description of the artist, such as "rock", "electronic", "jazz" and "80s". The artist-specific tags are converted to a l-length 0/1 covariate vector to indicate whether music tags are assigned to an artist by users, where l is the total number of different tags.

Figure 2 illustrates the listening pattern, showing that a log-transformation of the listening counts appears to be normally distributed. In addition, Figure 3 indicates that only a small portion of artists are popular and frequently listened to by many users. However, the majority of the artists are categorized in more specialized genres, which make them less popular over all, but still having their own followers. This is illustrated by the horizontal and diagonal stripes in Figure 3 respectively.



(a) Original listening counts (b) log-transformed listening counts

Figure 2: The original and log-transformed listening counts in the *Last.fm* dataset.

Figure 4 shows that popular artists tend to be listened to more by each follower; that is, the number of users listening to each artist is positively related with its averaged listening counts, indicating that missing is non-ignorable and that the missing pattern shall be incorporated in the recommender system.

For comparison, we investigate the performance of five methods: CRBM, SoftImpute, SSR, rSVD, and gSVD, where the residuals of these methods are obtained after adjusting for

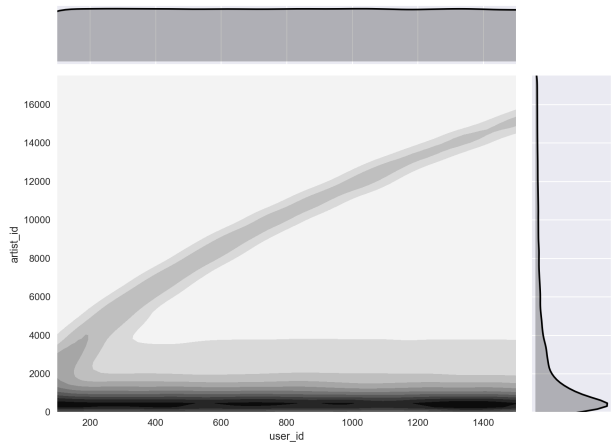


Figure 3: Two-dimensional histogram plot of the joint distribution of the observed user-artist pairs from the *Last.fm* dataset. The grey level represents the concentration, with darker color indicating more densely populated. Together with the joint distribution, the marginal distributions for user and artist are displayed along the horizontal and vertical axes.

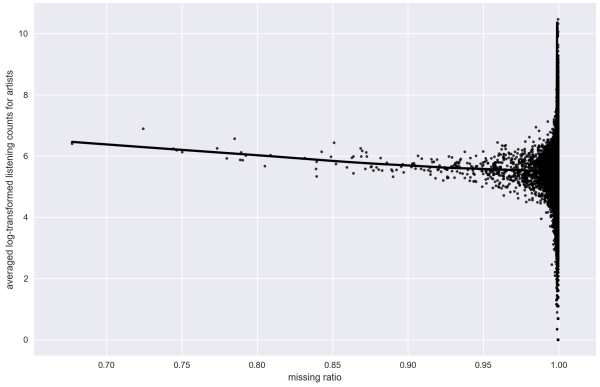


Figure 4: Missing pattern analysis for artists in the *Last.fm* dataset. It is clear that artists attracting fewer users tend to have smaller averaged listening counts.

covariate effects from user-specific social networks and artist-specific tags. Furthermore, to incorporate the non-ignorable missing pattern, the missing percentage is used to generate 12 homogeneous subgroups of users and 10 homogeneous subgroups of artists for gSVD, and covariates $(\boldsymbol{x}_u, \boldsymbol{x}_i)$ for sSVD are obtained from 5%, 25%, 50%, 75%, and 95% quantiles of the log observed listening counts in the training set for each user and artist. Also, we slightly modify the weight $\omega_{ui,u'i'}$ defined in (5) using thresholding. Specifically, $S^{ui}_{u'i'} = S^u_{u'} S^i_{i'}$, where $S^u_{u'} = 0.8$ if user u and u' are adjacent in the user social network, and $S^u_{u'} = 0.5$ otherwise; and $S^i_{i'}$ is computed based on the cosine similarity between the tag information of artists i and i'. For new users or artists, only user social network and artist tags information are available, therefore the weight function for covariate is automatically set as one. Furthermore, we apply a Gaussian kernel with window size h as the median distance among all user-artist pairs. For each user-artist pair (u, i), we compute the weights as in (5), and truncate them to keep only the five most similar user-artist pairs for each (u, i) to facilitate computation.

For evaluation, we apply 5-fold cross-validation over a random partition of the original dataset, and calculate the RMSE as in Koyejo and Ghosh (2011). For SVD-based methods, we set $K = 5$, and select the optimal λ from $\{1, \cdots, 25\}$ through the 5-fold cross-validation. For CRBM, the parameters are set as suggested in Nguyen and Lauw (2016); and for SSR, the parameters are determined through cross-validation as suggested in Zhao et al. (2016).

Table 3: *RMSEs of various methods and their estimated standard deviations in parentheses for observed, cold-start, and entire pairs on the Last.fm dataset. Here RBM, SoftImput, SSR, rSVD, gSVD, sSVD denote: restricted Boltzmann machines (Salakhutdinov et al., 2007), SoftImput method (Hastie et al., 2015), Self-recovered side regression (Zhao et al., 2016), regularized SVD method (Paterek, 2007), group-specific SVD method (Bi et al., 2017) and the proposed method, respectively. The second column indicates what type of covariates are used by each method, where N, T and M denote user-specific social networks, artist tags and non-ignorable missing pattern, respectively. The best performance in each setting is bold-faced.*

	Covariate	Observed pair	"Cold-start" pair	Entire pair
Regression	N, T	1.507(.005)	1.832(.023)	1.552(.005)
CRBM	N, T	1.507(.005)	1.834(.023)	1.552(.004)
SoftImpute	N, T	1.308(.005)	1.832(.023)	1.386(.004)
SSR	N, T	1.436(.006)	1.841(.035)	1.493(.009)
rSVD	N, T	1.124(.006)	1.832(.023)	1.237(.004)
gSVD	M, T	0.997(.013)	1.202(.012)	1.026(.011)
sSVD	N, M, T	**0.880(.004)**	**0.706(.003)**	**0.860(.003)**

Table 3 shows that sSVD significantly outperforms its competitors with a RMSE of 0.860 for the entire dataset, whereas gSVD is the second best performer with a RMSE of 1.020. As a reference, these two RMSEs are both smaller than 1.071 for the weighted interaction method under the same setting, which is reported as the best performer in analyzing the *Last.fm* dataset (Koyejo and Ghosh, 2011). The amounts of improvement of sSVD over gSVD, rSVD, SoftImput, SSR and CRBM are 15.6%, 24.6%, 38.0%, 42.4%, and 44.6%,

respectively. For the "cold-start" pairs, sSVD yields a more than 40% improvement over the second-best competitor gSVD. Note that SoftImput and rSVD do not perform well in handling the "cold-start" problem, as their penalization leads to the same performance as the regression approach.

6. Summary

This article proposes a smooth collaborative recommender system which integrates the network structure of user-item pairs to improve prediction accuracy. The proposed method provides a flexible framework to exploit the covariate information, such as user demographics, item contents, and social network information for users and/or items. The network structure allows us to increase the effective sample size for higher prediction accuracy, in addition to producing a more accurate recommender system for "cold-start" pairs. In addition, we implement a "divide-and-conquer" type of alternating least square algorithm. We also establish the asymptotic properties of the proposed method, which provide the theoretical foundation of its superior performance over other state-of-the-art methods. Although the proposed method is formulated based on the latent factor model, the framework can be extended to other models, such as Koren (2008) and Bi et al. (2017).

Acknowledgments

We would like to acknowledge support for this project from the National Science Foundation grants DMS-1415500, DMS-1712564, DMS-1721216, DMS-1613190, DMS-1415308, DMS-1821198, the National Institute of Health grants 1R01GM126002, R01HL105397, and Hong Kong Research Grant Council grants GRF-11302615, GRF-11303918 and GRF-11331016. The authors are grateful to reviewers and the Action Editor for their insightful comments and suggestions which have improved the manuscript significantly.

Appendix: technical proofs

Proof of Lemma 1. By the definition of $(\hat{\boldsymbol{P}}, \hat{\boldsymbol{Q}})$ as a minimizer of (3), we have

$$
\begin{aligned}
(\hat{\boldsymbol{P}}, \hat{\boldsymbol{Q}}) &= \underset{\boldsymbol{P},\boldsymbol{Q}}{\operatorname{argmin}} \ \frac{1}{nm}\sum_{u=1}^{n}\sum_{i=1}^{m}\Big(\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}(r_{u'i'}-\boldsymbol{p}_u^T\boldsymbol{q}_i)^2\Big)+\lambda_1\sum_{u=1}^{n}J(\boldsymbol{p}_u)+\lambda_2\sum_{i=1}^{m}J(\boldsymbol{q}_i)\\
&= \underset{\boldsymbol{P},\boldsymbol{Q}}{\operatorname{argmin}} \ \frac{1}{nm}\sum_{u=1}^{n}\sum_{i=1}^{m}\Big(\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}r_{u'i'}^2-\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}r_{u'i'}\boldsymbol{p}_u^T\boldsymbol{q}_i\\
&\qquad\qquad +\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}(\boldsymbol{p}_u^T\boldsymbol{q}_i)^2\Big)+\lambda_1\sum_{u=1}^{n}J(\boldsymbol{p}_u)+\lambda_2\sum_{i=1}^{m}J(\boldsymbol{q}_i)\\
&= \underset{\boldsymbol{P},\boldsymbol{Q}}{\operatorname{argmin}} \ \frac{1}{nm}\sum_{u=1}^{n}\sum_{i=1}^{m}\Big(\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}r_{u'i'}^2-\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}r_{u'i'}\boldsymbol{p}_u^T\boldsymbol{q}_i+(\boldsymbol{p}_u^T\boldsymbol{q}_i)^2\\
&\quad -\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}r_{u'i'}^2+\big(\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}r_{u'i'}\big)^2\Big)+\lambda_1\sum_{u=1}^{n}J(\boldsymbol{p}_u)+\lambda_2\sum_{i=1}^{m}J(\boldsymbol{q}_i)\\
&= \underset{\boldsymbol{P},\boldsymbol{Q}}{\operatorname{argmin}} \ \frac{1}{nm}\sum_{u=1}^{n}\sum_{i=1}^{m}\Big(\big(\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}r_{u'i'}\big)^2-\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}r_{u'i'}\boldsymbol{p}_u^T\boldsymbol{q}_i+(\boldsymbol{p}_u^T\boldsymbol{q}_i)^2\Big)\\
&\qquad\qquad +\lambda_1\sum_{u=1}^{n}J(\boldsymbol{p}_u)+\lambda_2\sum_{i=1}^{m}J(\boldsymbol{q}_i)\\
&= \underset{\boldsymbol{P},\boldsymbol{Q}}{\operatorname{argmin}} \ \frac{1}{nm}\sum_{u=1}^{n}\sum_{i=1}^{m}\Big(\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}r_{u'i'}-\boldsymbol{p}_u^T\boldsymbol{q}_i\Big)^2+\lambda_1\sum_{u=1}^{n}J(\boldsymbol{p}_u)+\lambda_2\sum_{i=1}^{m}J(\boldsymbol{q}_i),
\end{aligned}
$$

where the third equality follows that $\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}=1$ and the fact that adding constants to the cost function does not impact minimization. This completes the proof.

Proof of Theorem 1. Our treatment for bounding $P\big(\mathrm{RMSE}(\hat{\boldsymbol{\theta}},\boldsymbol{\theta})\geq\eta\big)$ is to bound empirical processes induced by $\mathrm{RMSE}(\cdot,\cdot)$ by a chaining argument as in (Wong and Shen, 1995; Shen et al., 2003; Liu and Shen, 2006).

Let $\boldsymbol{\Gamma}(L,\eta)=\{\boldsymbol{\theta}\in\boldsymbol{\Gamma}(L):\mathrm{RMSE}(\boldsymbol{\theta},\boldsymbol{\theta}^0)\geq\eta\}$ be a parameter subset of the parameter space $\boldsymbol{\Gamma}(L)$. Let $\lambda J(\boldsymbol{\theta})=\lambda_1\sum_{u=1}^{n}J(\boldsymbol{p}_u)+\lambda_2\sum_{i=1}^{m}J(\boldsymbol{q}_i)$ be the regularizer.

Note that $\hat{\boldsymbol{\theta}}$ is a minimizer of $\mathcal{L}(\boldsymbol{P},\boldsymbol{Q})$ in $\boldsymbol{\Gamma}(L)$, we have that $P\big(\mathrm{RMSE}(\hat{\boldsymbol{\theta}},\boldsymbol{\theta}^0))\geq\eta\big)\leq P^*\Big(\sup_{\boldsymbol{\theta}\in\boldsymbol{\Gamma}(L,\eta)}(\mathcal{L}(\boldsymbol{P}^0,\boldsymbol{Q}^0)-\mathcal{L}(\boldsymbol{P},\boldsymbol{Q}))\geq 0\Big)$, where P^* is the outer probability (Billingsley, 2013). Using the expression of $\mathcal{L}(\boldsymbol{P},\boldsymbol{Q})$, we obtain an upper bound of the latter as follows.

$$
\begin{aligned}
&P^*\Big(\sup_{\boldsymbol{\theta}\in\boldsymbol{\Gamma}(L,\eta)}(nm)^{-1}\sum_{u,i}\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}(\theta_{ui}-\theta_{ui}^0)(2r_{u'i'}-\theta_{ui}^0-\theta_{ui})+\lambda J(\boldsymbol{\theta}^0)-\lambda J(\boldsymbol{\theta})\geq 0\Big)\\
&\leq P^*\Big(\sup_{\boldsymbol{\theta}\in\boldsymbol{\Gamma}(L,\eta)}(nm)^{-1}\sum_{u,i}\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}(\theta_{ui}-\theta_{ui}^0)(2r_{u'i'}-\theta_{ui}^0-\theta_{ui})\geq-\lambda J(\boldsymbol{\theta}^0)\Big),
\end{aligned}
$$

where the fact that $\lambda J(\boldsymbol{\theta})\geq 0$ has been used. Now, let $A_j=\big\{\boldsymbol{\theta}\in\boldsymbol{\Gamma}(L):2^{j-1}\eta\leq\mathrm{RMSE}(\boldsymbol{\theta},\boldsymbol{\theta}^0)\leq 2^j\eta\big\}$; $j=1,\cdots,\infty$ be a partition in that $\boldsymbol{\Gamma}(L,\eta)=\bigcup_{j=1}^{\infty}A_j$. Combining

the above inequalities, we have that $P\big(\mathrm{RMSE}(\hat{\boldsymbol{\theta}},\boldsymbol{\theta}^0)\geq\eta\big)$ is bounded by

$$
\sum_{j=1}^{\infty}P^*\Big(\sup_{A_j}(nm)^{-1}\sum_{u,i}\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}(\theta_{ui}-\theta_{ui}^0)(2r_{u'i'}-\theta_{ui}^0-\theta_{ui})\geq-\lambda J(\boldsymbol{\theta}^0)\Big)\equiv\sum_{j=1}^{\infty}I_j.
$$

Next, we bound each I_j separately. Since $\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}=1$,

$$
\begin{aligned}
&(nm)^{-1}\sum_{u,i}\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}(\theta_{ui}-\theta_{ui}^0)(2r_{u'i'}-\theta_{ui}^0-\theta_{ui})\\
=\ &(nm)^{-1}\sum_{u,i}\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}(\theta_{ui}-\theta_{ui}^0)(\theta_{u'i'}^0-\theta_{ui}^0)\\
&+(nm)^{-1}\sum_{u,i}\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}(\theta_{ui}-\theta_{ui}^0)\epsilon_{u'i'}-(nm)^{-1}\sum_{u,i}(\theta_{ui}-\theta_{ui}^0)^2,
\end{aligned}
$$

which, by Assumption A and the fact that $\omega_{ui,u'i'}=0$ when $S_{u'i'}^{ui}=0$, is upper-bounded by

$$
\begin{aligned}
&c_1\sqrt{K}(nm)^{-1}\sum_{u,i}|\theta_{ui}-\theta_{ui}^0|\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}\|\boldsymbol{x}_{u'i'}-\boldsymbol{x}_{ui}\|_2^{\alpha}\\
&\qquad+(nm)^{-1}\sum_{u,i}\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}(\theta_{ui}-\theta_{ui}^0)\epsilon_{u'i'}-(nm)^{-1}\sum_{u,i}(\theta_{ui}-\theta_{ui}^0)^2.
\end{aligned}
$$

Then for any $\boldsymbol{\theta}\in A_j$, $(2^j\eta)^2\geq(nm)^{-1}\sum_{u,i}(\theta_{ui}-\theta_{ui}^0)^2\geq(2^{j-1}\eta)^2$, by the Cauchy-Schwarz inequality,

$$
\begin{aligned}
&\sup_{A_j}c_1\sqrt{K}(nm)^{-1}\sum_{u,i}|\theta_{ui}-\theta_{ui}^0|\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}\|\boldsymbol{x}_{u'i'}-\boldsymbol{x}_{ui}\|_2^{\alpha}\\
&\leq\sup_{A_j}c_1\sqrt{K}\Big((nm)^{-1}\sum_{u,i}(\theta_{ui}-\theta_{ui}^0)^2\Big)^{1/2}\Big((nm)^{-1}\sum_{u,i}\big(\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}\|\boldsymbol{x}_{u'i'}-\boldsymbol{x}_{ui}\|_2^{\alpha}\big)^2\Big)^{1/2}\\
&\leq 2c_1\sqrt{K}\kappa_1 2^j\eta.
\end{aligned}
$$

Thus, $I_j\leq P^*\Big(\sup_{A_j}(nm)^{-1}\sum_{u,i}\sum_{(u',i')\in\Omega}2\omega_{ui,u'i'}(\theta_{ui}-\theta_{ui}^0)\epsilon_{u'i'}\geq(2^{j-1}\eta)^2-2c_1\sqrt{K}\kappa_1 2^j\eta-\lambda J(\boldsymbol{\theta}^0)\Big)$. Note that $\lambda J(\boldsymbol{\theta}^0)\leq\eta^2/4$ and $\mathrm{RMSE}(\boldsymbol{\theta},\boldsymbol{\theta}^0)=\Big(\sum_{u,i}(\theta_{ui}-\theta_{ui}^0)^2\Big)^{1/2}\geq 2^{j-1}\eta$ for

$\boldsymbol{\theta} \in A_j$. For any $\eta \geq 2^4 c_1\sqrt{K}\kappa_1$, $(2^{j-1}\eta)^2 - 2c_1\sqrt{K}\kappa_1 2^j\eta - \lambda J(\boldsymbol{\theta}^0) \geq 2^{2j-4}\eta^2$, implying that

$$
\begin{aligned}
I_j &\leq P^*\Big(\sup_{A_j}(nm)^{-1}\sum_{u,i}(\theta_{ui}-\theta^0_{ui})\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}\epsilon_{u'i'} \geq 2^{2j-4}\eta^2\Big)\\
&\leq P^*\Big(\sup_{A_j}(nm)^{-1}\Big(\sum_{u,i}(\theta_{ui}-\theta^0_{ui})^2\Big)^{1/2}\Big(\sum_{u,i}\big(\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}\epsilon_{u'i'}\big)^2\Big)^{1/2} \geq 2^{2j-4}\eta^2\Big)\\
&\leq P^*\Big(\Big((nm)^{-1}\sum_{u,i}\big(\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}\epsilon_{u'i'}\big)^2\Big)^{1/2} \geq 2^{j-4}\eta\Big)\\
&\leq P^*\Big(\max_{u,i}\big|\sum_{(u',i')\in\Omega}\omega_{ui,u'i'}\epsilon_{u'i'}\big| \geq 2^{j-4}\eta\Big)\\
&\leq \sum_{u,i} 2\exp\Big(-\frac{2^{(2j-8)}\eta^2}{2\sigma^2\sum_{(u',i')\in\Omega}\omega^2(\boldsymbol{x}_{ui},\boldsymbol{x}_{u'i'})}\Big) \leq 2nm\exp\Big(-\frac{2^{(2j-8)}\eta^2}{2\sigma^2\kappa_2}\Big),
\end{aligned}
$$

where the second to the last inequalities follow from the Chernoff inequality of a weighted sub-Gaussian distribution (Chung and Lu, 2006) and Assumption B. Hence, there exist some positive constants a_2 and a_3, for $\eta \geq 2^4 c_1\sqrt{K}\kappa_1$ such that

$$
\begin{aligned}
&P\big(\mathrm{RMSE}(\hat{\boldsymbol{\theta}},\boldsymbol{\theta}^0) \geq \eta\big) \leq 4nm\sum_{j=1}^{\infty}\exp\Big\{-\frac{2^{(2j-8)}\eta^2}{2\sigma^2\kappa_2}\Big\}\\
&\leq 4nm\exp\{-a_2\frac{\eta^2}{\sigma^2\kappa_2}\}/(1-\exp\{-a_2\frac{\eta^2}{\sigma^2\kappa_2}\}) \leq a_3\exp\big\{-a_2\frac{\eta^2}{\sigma^2\kappa_2}+\log(nm)\big\}.
\end{aligned}
$$

The desired result then follows immediately. □

Proof of Corollary 1. It suffices to compute κ_1 and κ_2. First, we will show that for any $\boldsymbol{x}_{ui} \in \mathcal{X}$, when $|\Omega|$ is sufficiently large, there exist positive constants a_4–a_7 such that

$$
\begin{aligned}
\sum_{(u',i')\in\Omega}\mathcal{K}_h(\|\boldsymbol{x}_{u'i'}-\boldsymbol{x}_{ui}\|_2)S^{ui}_{u'i'} &\geq \frac{|\Omega|}{2}E\big(\mathcal{K}_h(\|\boldsymbol{x}-\boldsymbol{x}_{ui}\|_2)S^{ui}\big|\Delta=1\big)\\
&= \frac{|\Omega|}{2}P(S^{ui}=1|\Delta=1)E\big(\mathcal{K}_h(\|\boldsymbol{x}-\boldsymbol{x}_{ui}\|_2)\big|S^{ui}=1,\Delta=1\big)\\
&\geq a_4|\Omega|E\big(\mathcal{K}_h(\|\boldsymbol{x}-\boldsymbol{x}_{ui}\|_2)\big|S^{ui}=1,\Delta=1\big)\\
&\geq a_5|\Omega|\int\mathcal{K}_h(u)f_{U^{ui}|S^{ui}=1,\Delta=1}(u)du \geq a_6|\Omega|h\int\mathcal{K}(u)du\\
&\geq a_7|\Omega|h,
\end{aligned}
$$

where the $f_{U^{ui}|S^{ui}=1,\Delta=1}$ is the conditional density for U^{ui}, the first inequality follows from the law of large numbers, and the third inequality and the third to the last inequalities follow from Assumption C. Similarly, for some positive constants a_8 and a_9,

$$
\begin{aligned}
\sum_{(u',i')\in\Omega}\mathcal{K}_h(\|\boldsymbol{x}_{ui}-\boldsymbol{x}_{u'i'}\|_2)\|\boldsymbol{x}_{ui}-\boldsymbol{x}_{u'i'}\|^\alpha S^{ui}_{u'i'} &\leq 2|\Omega|E\big(\mathcal{K}_h(U^{ui})(U^{ui})^\alpha|S^{ui}=1,\Delta=1\big)\\
&= 2|\Omega|\int\mathcal{K}_h(u)u^\alpha f_{U^{ui}|S^{ui}=1,\Delta=1}(u)du \leq a_8|\Omega|h^{\alpha+1},
\end{aligned}
$$

where the last inequality follows from Assumptions C and D, and

$$
\sum_{(u',i')\in\Omega}\mathcal{K}_h^2(\|\boldsymbol{x}_{ui}-\boldsymbol{x}_{u'i'}\|_2)S^{ui}_{u'i'} \leq 2|\Omega|E\big(\mathcal{K}_h^2(U^{ui})|S^{ui}=1,\Delta=1\big) \leq a_9|\Omega|h.
$$

Combing the above inequalities yields that

$$
\sum_{(u',i')\in\Omega}\omega(\boldsymbol{x}_0,\boldsymbol{x}_{u'i'})\|\boldsymbol{x}_0,\boldsymbol{x}_{u'i'}\|_2^\alpha = \frac{\sum_{(u',i')\in\Omega}\mathcal{K}_h(\|\boldsymbol{x}_0-\boldsymbol{x}_{u'i'}\|_2)\|\boldsymbol{x}_0-\boldsymbol{x}_{u'i'}\|_2^\alpha S^{ui}_{u'i'}}{\sum_{(u',i')\in\Omega}\mathcal{K}_h(\|\boldsymbol{x}-\boldsymbol{x}_{u'i'}\|_2)S^{ui}_{u'i'}} \leq a_8h^\alpha.
$$

Furthermore,

$$
\sum_{(u',i')\in\Omega}\omega^2(\boldsymbol{x}_0,\boldsymbol{x}_{u'i'}) \leq \frac{\sum_{(u',i')\in\Omega}\mathcal{K}_h^2(\|\boldsymbol{x}_0-\boldsymbol{x}_{u'i'}\|_2)S^{ui}_{u'i'}}{\big(\sum_{(u',i')\in\Omega}\mathcal{K}_h(\|\boldsymbol{x}-\boldsymbol{x}_{u'i'}\|_2)S^{ui}_{u'i'}\big)^2} \leq \frac{a_9}{a_7^2|\Omega|h}.
$$

Consequently, $\kappa_1 = h^\alpha$ and $\kappa_2 = (|\Omega|h)^{-1}$, then the desired result follows immediately. □

References

Deepak Agarwal, Liang Zhang, and Rahul Mazumder. Modeling item-item similarities for personalized recommendations on Yahoo! front page. *Annals of Applied Statistics*, 5(3): 1839–1875, 2011.

Stephen H Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research*, 18 (109):1–67, 2017.

Robert M Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *7th IEEE International Conference on Data Mining (ICDM 2007)*, pages 43–52. IEEE, 2007.

Xuan Bi, Annie Qu, Junhui Wang, and Xiaotong Shen. A group-specific recommender system. *Journal of the American Statistical Association*, 112(519):1344–1353, 2017.

Patrick Billingsley. *Convergence of probability measures*. John Wiley & Sons, 2013.

Daniel Billsus and Michael J Pazzani. User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, 10(2-3):147–180, 2000.

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

Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35 (3):283–319, 1970.

Bilian Chen, Simai He, Zhening Li, and Shuzhong Zhang. Maximum block improvement and polynomial optimization. *SIAM Journal on Optimization*, 22(1):87–107, 2012.

Hsin Chen and Alan F Murray. Continuous restricted Boltzmann machine with an implementable training algorithm. *IEE Proceedings-Vision, Image and Signal Processing*, 150 (3):153–158, 2003.

Tianle Chen, Yuanjia Wang, Huaihou Chen, Karen Marder, and Donglin Zeng. Targeted local support vector machine for age-dependent classification. *Journal of the American Statistical Association*, 109(507):1174–1187, 2014.

Fan Chung and Linyuan Lu. Concentration inequalities and martingale inequalities: A survey. *Internet Mathematics*, 3(1):79–127, 2006.

Aurore Delaigle and Peter Hall. Defining probability density for a distribution of random functions. *Annals of Statistics*, 38(2):1171–1193, 2010.

Özgür Demir, Alexey Rodriguez Yakushev, Rany Keddo, and Ursula Kallio. Item-item music recommendations with side information. *CoRR*, abs/1706.00218, 2017. URL `http://arxiv.org/abs/1706.00218`.

Jianqing Fan and Irene Gijbels. *Local polynomial modelling and its applications: monographs on statistics and applied probability*, volume 66. CRC Press, 1996.

Andrey Feuerverger, Yu He, and Shashi Khatri. Statistical significance of the Netflix challenge. *Statistical Science*, 27(2):202–231, 2012.

Peter Forbes and Mu Zhu. Content-boosted matrix factorization for recommender systems: Experiments with recipe recommendation. In *Proceedings of the 5th ACM conference on Recommender Systems*, pages 261–264. ACM, 2011.

Jerome H Friedman and Werner Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, 76(376):817–823, 1981.

Quanquan Gu, Jie Zhou, and Chris Ding. Collaborative filtering: Weighted nonnegative matrix factorization incorporating user and item graphs. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 199–210. SIAM, 2010.

Trevor Hastie, Rahul Mazumder, Jason D Lee, and Reza Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *Journal of Machine Learning Research*, 16(1):3367–3402, 2015.

Thomas Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1):89–115, 2004.

Yehuda Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 426–434. ACM, 2008.

Pigi Kouki, Shobeir Fakhraei, James Foulds, Magdalini Eirinaki, and Lise Getoor. Hyper: A flexible and extensible probabilistic framework for hybrid recommender systems. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 99–106. ACM, 2015.

Oluwasanmi Koyejo and Joydeep Ghosh. A kernel-based approach to exploiting interaction-networks in heterogeneous information sources for improved recommender systems. In *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, pages 9–16. ACM, 2011.

Ken Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339, 1995.

Huazhen Lin, Lixian Pan, Shaogao Lv, and Wenyang Zhang. Efficient estimation and computation for the generalised additive models with unknown link function. *Journal of Econometrics*, 2017.

Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

Yufeng Liu and Xiaotong Shen. Multicategory ψ-learning. *Journal of the American Statistical Association*, 101(474):500–509, 2006.

Shujie Ma and Jian Huang. A concave pairwise fusion approach to subgroup analysis. *Journal of the American Statistical Association*, 112(517):410–423, 2017.

JS Marron and WJ Padgett. Asymptotically optimal bandwidth selection for kernel density estimators from randomly right-censored samples. *Annals of Statistics*, pages 1520–1535, 1987.

Judith Masthoff. Group recommender systems: Combining individual models. In *Recommender systems handbook*, pages 677–702. Springer, 2011.

Prem Melville, Raymond J Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. *AAAI/IAAI*, 23:187–192, 2002.

Stuart E Middleton, Nigel R Shadbolt, and David C De Roure. Ontological user profiling in recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):54–88, 2004.

Bradley N Miller, Istvan Albert, Shyong K Lam, Joseph A Konstan, and John Riedl. Movielens unplugged: Experiences with an occasionally connected recommender system. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*, pages 263–266. ACM, 2003.

A Mukherjee, K Chen, N Wang, and J Zhu. On the degrees of freedom of reduced-rank estimators in multivariate regression. *Biometrika*, 102(2):457–477, 2015.

Jennifer Nguyen and Mu Zhu. Content-boosted matrix factorization techniques for recommender systems. *Statistical Analysis and Data Mining*, 6(4):286–301, 2013.

Trong T Nguyen and Hady W Lauw. Representation learning for homophilic preferences. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 317–324. ACM, 2016.

Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD Cup and Workshop*, pages 5–8, 2007.

Michael J Pazzani, Jack Muramatsu, Daniel Billsus, et al. Syskill & Webert: Identifying interesting web sites. In *AAAI/IAAI*, volume 1, pages 54–61, 1996.

Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62 (1-2):107–136, 2006.

Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th International Conference on Machine Learning*, pages 880–887. ACM, 2008.

Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine learning*, pages 791–798. ACM, 2007.

David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

Xiaotong Shen, George C Tseng, Xuegong Zhang, and Wing Hung Wong. On ψ-learning. *Journal of the American Statistical Association*, 98(463):724–734, 2003.

Nathan Srebro, Noga Alon, and Tommi S Jaakkola. Generalization error bounds for collaborative prediction with low-rank matrices. In *Advances in Neural Information Processing Systems*, pages 1321–1328, 2005.

Charles J Stone. An asymptotically optimal window selection rule for kernel density estimates. *Annals of Statistics*, pages 1285–1297, 1984.

Robert Tibshirani and Trevor Hastie. Local likelihood estimation. *Journal of the American Statistical Association*, 82(398):559–567, 1987.

Philippe Vieu. Nonparametric regression: optimal local bandwidth choice. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 453–464, 1991.

Larry Wassermann. *All of nonparametric statistics*. New York: Springer, 2006.

Wing Hung Wong and Xiaotong Shen. Probability inequalities for likelihood ratios and convergence rates of sieve MLEs. *Annals of Statistics*, 23(2):339–362, 1995.

Shuang-Hong Yang, Bo Long, Alex Smola, Narayanan Sadagopan, Zhaohui Zheng, and Hongyuan Zha. Like like alike: joint friendship and interest propagation in social networks. In *Proceedings of the 20th International Conference on World Wide Web*, pages 537–546. ACM, 2011.

Feipeng Zhao and Yuhong Guo. Learning discriminative recommendation systems with side information. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 3469–3475, 2017.

Feipeng Zhao, Min Xiao, and Yuhong Guo. Predictive collaborative filtering with side information. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 2385–2391, 2016.

Tinghui Zhou, Hanhuai Shan, Arindam Banerjee, and Guillermo Sapiro. Kernelized probabilistic matrix factorization: Exploiting graphs and side information. In *Proceedings of the 2012 SIAM International Conference on Data Mining*, pages 403–414. SIAM, 2012.

Yunzhang Zhu, Xiaotong Shen, and Changqing Ye. Personalized prediction and sparsity pursuit in latent factor models. *Journal of the American Statistical Association*, 111(513): 241–252, 2016.

Journal of Machine Learning Research 20 (2019) 1-38 Submitted 10/17; Published 2/19

Delay and Cooperation in Nonstochastic Bandits

Nicolò Cesa-Bianchi nicolo.cesa-bianchi@unimi.it
Department of Computer Science & DSRC
Università degli Studi di Milano
20133 Milano, Italy

Claudio Gentile cla.gentile@gmail.com
Google Research
New York, NY, USA

Yishay Mansour mansour@tau.ac.il
Google Research and Tel-Aviv University
Tel-Aviv 6997801, Israel

Editor: Network Analysis Special Issue

Abstract

We study networks of communicating learning agents that cooperate to solve a common nonstochastic bandit problem. Agents use an underlying communication network to get messages about actions selected by other agents, and drop messages that took more than d hops to arrive, where d is a delay parameter. We introduce Exp3-Coop, a cooperative version of the Exp3 algorithm and prove that with K actions and N agents the average per-agent regret after T rounds is at most of order $\sqrt{\left(d+1+\frac{K}{N}\alpha_{\leq d}\right)(T\ln K)}$, where $\alpha_{\leq d}$ is the independence number of the d-th power of the communication graph G. We then show that for any connected graph, for $d=\sqrt{K}$ the regret bound is $K^{1/4}\sqrt{T}$, strictly better than the minimax regret $\sqrt{KT}$ for noncooperating agents. More informed choices of d lead to bounds which are arbitrarily close to the full information minimax regret $\sqrt{T\ln K}$ when G is dense. When G has sparse components, we show that a variant of Exp3-Coop, allowing agents to choose their parameters according to their centrality in G, strictly improves the regret. Finally, as a by-product of our analysis, we provide the first characterization of the minimax regret for bandit learning with delay.

Keywords: Multi-armed bandits, distributed learning, cooperative multi-agent systems, regret minimization, LOCAL communication

1. Introduction

Delayed feedback naturally arises in many sequential decision problems. For instance, a recommender system typically learns the utility of a recommendation by detecting the occurrence of certain events (e.g., a user conversion), which may happen with a variable delay after the recommendation was issued. Other examples are the communication delays experienced by interacting learning agents. Concretely, consider a network of geographically distributed ad servers using real-time bidding to sell their inventory. Each server sequentially learns how to set the auction parameters (e.g., reserve price) in order to maximize the network's overall revenue, and shares feedback information with other servers in order to speed up learning. However, the rate at which information is exchanged through the communication network is slower than the typical rate at which ads are served. This causes each learner to acquire feedback information from other servers with a delay that depends on the network's structure.

Motivated by the ad network example, we consider networks of learning agents that cooperate to solve the same nonstochastic bandit problem, and study the impact of delay on the global performance of these agents. We introduce the Exp3-Coop algorithm, a distributed and cooperative version of the Exp3 algorithm of Auer et al. (2002b). Exp3-Coop works within a distributed and synced model where each agent runs an instance of the same bandit algorithm (Exp3). All bandit instances are initialized in the same way irrespective to the agent's location in the network (that is, agents have no preliminary knowledge of the network), and we assume the information about an agent's actions is propagated through the network with a unit delay for each crossed edge. In each round t, each agent selects an action and incurs the corresponding loss (which is the same for all agents that pick that action in round t). Besides observing the loss of the selected action, each agent obtains the information previously broadcast by other agents with a delay equal to the shortest-path distance between the agents. Namely, at time t an agent learns what the agents at shortest-path distance s did at time $t-s$ for each $s=1,\dots,d$, where d is a delay parameter. In this scenario, we aim at controlling the growth of the regret averaged over all agents (the so-called average welfare regret).

In the noncooperative case, when agents ignore the information received from other agents, the average welfare regret grows like $\sqrt{KT}$ (the minimax rate for standard bandit setting), where K is the number of actions and T is the time horizon. We show that, using cooperation, N agents with communication graph G can achieve an average welfare regret of order $\sqrt{\left(d+1+\frac{K}{N}\alpha_{\leq d}\right)(T\ln K)}$. Here $\alpha_{\leq d}$ denotes the independence number of the d-th power of G (i.e., the graph G augmented with all edges between any two pair of nodes at shortest-path distance less than or equal to d). When $d=\sqrt{K}$ this bound is at most $K^{1/4}\sqrt{T\ln K}+\sqrt{K}(\ln T)$ for any connected graph —see Remark 8 in Section 4.1— which is asymptotically better than $\sqrt{KT}$.

Networks of nonstochastic bandits were also investigated by Awerbuch and Kleinberg (2008) in a setting where the distribution over actions is shared among the agents without delay. Awerbuch and Kleinberg (2008) prove a bound on the average welfare regret of order $\sqrt{\left(1+\frac{K}{N}\right)T}$ ignoring polylog factors.[1] We recover the same bound as a special case of our bound when G is a clique and $d=1$. In the clique case our bound is also similar to the bound $\sqrt{\frac{K}{N}(T\ln K)}$ achieved by Seldin et al. (2014) in a single-agent bandit setting where, at each time step, the agent can choose a subset of $N\leq K$ actions and observe their loss. In the case when $N=1$ (single agent), our analysis can be applied to the nonstochastic bandit problem where the player observes the loss of each played action with a delay of d steps. In this case we improve on the previous result of $\sqrt{(d+1)KT}$ by Neu et al. (2010, 2014), and give the first characterization (up to logarithmic factors) of the minimax regret, which is of order $\sqrt{(d+K)\,T}$.

1. The rate proven in (Awerbuch and Kleinberg, 2008, Theorem 2.1) has a worse dependence on T, but we believe this is due to the fact that their setting allows for dishonest agents and agent-specific loss vectors.

©2019 Nicolò Cesa-Bianchi, Claudio Gentile, and Yishay Mansour.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-631.html.

In principle, the problem of delays in online learning could be tackled by simple reductions. Yet, these reductions give rise to suboptimal results. In the single agent setting, where the delay is constant and equal to d, one can use the technique of Weinberger and Ordentlich (2002) and run $d+1$ instances of an online algorithm for the nondelayed case, where each instance is used every $d+1$ steps. This delivers a suboptimal regret bound of $\sqrt{(d+1)KT}$. In the case of multiple delays, like in our multi-agent setting, one can repeat the same action for $d+1$ steps while accumulating information from the other agents, and then perform an update on scaled-up losses. The resulting (suboptimal) bound on the average welfare regret would be of the form $\sqrt{(d+1)\big(1+\frac{K}{N}\alpha_{\leq d}\big)(T\ln K)}$.

Rather than using reductions, the analysis of Exp3-Coop rests on quantifying the performance of suitable importance weighted estimates. In fact, in the single-agent setting with delay parameter d, using Exp3-Coop reduces to running the standard Exp3 algorithm performing an update as soon a new loss becomes available. This implies that at any round $t > d$, Exp3 selects an action without knowing the losses incurred during the last d rounds. The resulting regret is bounded by relating the standard analysis of Exp3 to a detailed quantification of the extent to which the distribution maintained by Exp3 can drift in d steps.

In the multi-agent case, the importance weighted estimate of Exp3-Coop is designed in such a way that at each time $t > d$ the instance of the algorithm run by an agent v updates all actions that were played at time $t-d$ by agent v or by other agents not further away than d from v. Compared to the single agent case, here each agent can exploit the information circulated by the other agents. However, in order to compute the importance weighted estimates used locally by each agent, the probabilities maintained by the agents must be propagated together with the observed losses. Here, further concerns may show up, like the amount of communication, and the location of each agent within the network. In particular, when G has sparse components, we show that a variant of Exp3-Coop, allowing agents to choose their parameters according to their centrality within G, strictly improves on the regret of Exp3-Coop.

Finally, we propose a second variant of Exp3-Coop where each agent is able to use loss information as soon as it becomes available. This implies that the updates performed at time t now involve losses with different delays. For this reason, the new variant of Exp3-Coop combines many loss estimators, each defined for a different level of delay, through a fixed distribution D. In the analysis we show how the introduction of combined loss estimators affects the average welfare regret.

A preliminary version of this work appeared as an extended abstract in (Cesa-Bianchi et al., 2016).

2. Additional Related Work

Many important ideas in delayed online learning, including the observation that the effect of delays can be limited by controlling the amount of change in the agent strategy, were introduced by Mesterharm (2005) —see also (Mesterharm, 2007, Chapter 8). A more recent investigation on delayed online learning is due to Neu et al. (2010, 2014), who analyzed exponential weights with delayed feedbacks. Furher progress is made by Joulani et al. (2013), who also study delays in the general partial monitoring setting. Additional works (Joulani

et al., 2016; Quanrud and Khashabi, 2015) prove regret bounds for the full-information case of the form $\sqrt{(D+T)\ln K}$, where D is the total delay experienced over the T rounds. In the stochastic case, bandit learning with delayed feedback was considered by Dudík et al. (2011); Joulani et al. (2013) and in a harder anonymized model by Pike-Burke et al. (2018).

To the best of our knowledge, the first paper about nonstochastic cooperative bandit networks is (Awerbuch and Kleinberg, 2008). More papers analyze the stochastic setting, and the closest one to our work is perhaps (Szorenyi et al., 2013). In that paper, delayed loss estimates in a network of cooperating stochastic bandits are analyzed using a dynamic P2P random networks as communication model. A more recent paper is (Landgren et al., 2015), where the communication network is a fixed graph and a cooperative version of the UCB algorithm is introduced which uses a distributed consensus algorithm to estimate the mean rewards of the arms. The main result is an individual (per-agent) regret bound that depends on the network structure without taking delays into account.

Another interesting paper about cooperating bandits in a stochastic setting is (Kar et al., 2011). Similar to our model, agents sit on the nodes of a communication network. However, only one designated agent observes the rewards of actions he selects, whereas the others remain in the dark. This designated agent broadcasts his sampled actions through the networks to the other agents, who must learn their policies relying only on this indirect feedback. The paper shows that in any connected network this information is sufficient to achieve asymptotically optimal regret. Cooperative bandits with asymmetric feedback are also studied by Barrett and Stone (2011). In their model, an agent must teach the reward distribution to another agent while keeping the discounted regret under control. Tekin and van der Schaar (2015) investigate a stochastic contextual bandit model where each agent can either privately select an action or have another agent select an action on his behalf. In a related paper, Tekin et al. (2014) look at a stochastic bandit model with combinatorial actions in a distributed recommender system setting, and study incentives among agents who can now recommend items taken from other agents' inventories. A more recent paper (Kolla et al., 2016) studies the performance of the UCB algorithm by Auer et al. (2002a) in a multi-agent setting, where at each time step each agent observes also the losses of actions chosen by any agent located in the same neighborhood of the communication network.

A further line of relevant work involves problems of decentralized bandit coordination. For example, Stranders et al. (2012) consider a bandit coordination problem where the reward function is global and can be represented as a factor graph in which each agent controls a subset of the variables.

A parallel thread of research concerns networks of bandits that compete for shared resources. A paradigmatic application domain is that of cognitive radio networks, in which a number of channels are shared among many users and any two or more users interfere whenever they simultaneously try to use the same channel. The resulting bandit problem is one of coordination in a competitive environment, because every time two or more agents select the same action at the same time step they both get a zero reward due to the interference —see (Rosenski et al., 2015) for recent work on stochastic competitive bandits and (Kleinberg et al., 2009) for a study of more general congestion games in a game-theoretic setting.

Finally, there exists an extensive literature on the adaptation of gradient descent and related algorithms to distributed computing settings, where asynchronous processors natu-

rally introduce delays —see, e.g., (Zinkevich et al., 2009; Agarwal and Duchi, 2011; Li et al., 2013; McMahan and Streeter, 2014; Quanrud and Khashabi, 2015; Liu et al., 2015; Duchi et al., 2015). However, none of these works considers bandit settings, which are an essential ingredient for our analysis.

3. Preliminaries

We now establish our notation, along with basic assumptions and preliminary facts related to our algorithms. Notation and setting here both refer to the single agent case. The cooperative setting with multiple agents (and notation thereof) will be introduced in Section 4.

Let $A = \{1, \dots, K\}$ be the action set. A learning agent runs an exponentially-weighted algorithm with weights $w_t(i)$, and learning rate $\eta > 0$. Initially, $w_1(i) = 1$ for all $i \in A$. At each time step $t = 1, 2, \dots$, the agent draws action I_t with probability $\mathbb{P}(I_t = i) = p_t(i) = w_t(i)/W_t$, where $W_t = \sum_{j \in A} w_t(j)$. After observing the loss $\ell_t(I_t) \in [0, 1]$ associated with the chosen action I_t, and possibly some additional information, the agent computes, for each $i \in A$, nonnegative loss estimates $\widehat{\ell}_t(i)$, and performs the exponential update

$$w_{t+1}(i) = p_t(i) \exp\bigl(-\eta\, \widehat{\ell}_t(i)\bigr) \tag{1}$$

to these weights. The following two lemmas are general results that control the evolution of the probability distributions in the exponentially-weighted algorithm. As we said in the introduction, bounding the extent to which the distribution used by our algorithms can drift in d steps is key to controlling regret in a delayed setting. The first result bounds the *additive* change in the probability of any action, and it holds no matter how $\widehat{\ell}_t(i)$ is defined.

Lemma 1 *Under the update rule (1), for all $t \geq 1$ and for all $i \in A$,*

$$-\eta\, p_t(i)\widehat{\ell}_t(i) \leq p_{t+1}(i) - p_t(i) \leq \eta\, p_{t+1}(i) \sum_{j \in A} p_t(j)\widehat{\ell}_t(j)$$

holds deterministically with respect to the agent's randomization.

Proof Directly from the definition of the update (1), $w_{t+1}(i) \leq p_t(i)$ for all $i \in A$, so that $W_{t+1} \leq 1$, which in turn implies $w_{t+1}(i) \leq w_{t+1}(i)/W_{t+1} = p_{t+1}(i)$. Therefore

$$p_{t+1}(i) - p_t(i) \geq w_{t+1}(i) - p_t(i) = p_t(i)\left(e^{-\eta\,\widehat{\ell}_t(i)} - 1\right) \geq -\eta\, p_t(i)\widehat{\ell}_t(i)\ ,$$

the last inequality using $1 - e^{-x} \leq x$, for $x \geq 0$. Similarly,

$$\begin{aligned}
p_{t+1}(i) - p_t(i) &\leq p_{t+1}(i) - w_{t+1}(i) \\
&= p_{t+1}(i) - p_{t+1}(i)W_{t+1} \\
&= p_{t+1}(i) \sum_{j \in A} \bigl(p_t(j) - w_{t+1}(j)\bigr) \\
&= p_{t+1}(i) \sum_{j \in A} p_t(j)\left(1 - e^{-\eta\,\widehat{\ell}_t(j)}\right) \\
&\leq \eta\, p_{t+1}(i) \sum_{j \in A} p_t(j)\widehat{\ell}_t(j)\ ,
\end{aligned}$$

hence concluding the proof. ∎

The second result delivers a *multiplicative* bound on the change in the probability of any action when the loss estimates $\widehat{\ell}_t(i)$ are of the following form:

$$\widehat{\ell}_t(i) = \begin{cases} \dfrac{\ell_{t-d}(i)}{q_{t-d}(i)}\, B_{t-d}(i) & \text{if } t > d, \\ 0 & \text{otherwise}\ , \end{cases} \tag{2}$$

where $d \geq 0$ is a delay parameter, $B_{t-d}(i) \in \{0, 1\}$, for $i \in A$, are indicator functions, and $q_{t-d}(i) \geq p_{t-d}(i)$ for all i and $t > d$. In all later sections, $B_{t-d}(i)$ will be instantiated to the indicator function of the event that action i has been played at time $t - d$ by some agent, and $q_{t-d}(i)$ will be the (conditional) probability of this event.

Lemma 2 *Let $\widehat{\ell}_t(i)$ be of the form (2) for each $t \geq 1$ and $i \in A$. If $\eta \leq \frac{1}{Ke(d+1)}$ in the update rule (1), then*

$$p_{t+1}(i) \leq \left(1 + \frac{1}{d}\right) p_t(i)$$

holds for all $t \geq 1$ and $i \in A$, deterministically with respect to the agent's randomization.

Proof We proceed by induction over t. For all $t \leq d$, $\widehat{\ell}_t(\cdot) = 0$. Hence $p_t(\cdot) = 1/K$, and the lemma trivially holds. For $t > d$ we can write

$$\begin{aligned}
\sum_{i \in A} p_t(i)\widehat{\ell}_t(i) &= \sum_{i \in A} p_t(i) \frac{\ell_{t-d}(i)}{q_{t-d}(i)} B_{t-d}(i) \\
&\leq \sum_{i \in A} \frac{p_t(i)}{q_{t-d}(i)} && \text{(because } B_{t-d}(i)\ell_{t-d}(i) \leq 1\text{)} \\
&\leq \sum_{i \in A} \left(1 + \frac{1}{d}\right)^d \frac{p_{t-d}(i)}{q_{t-d}(i)} && \text{(by the inductive hypothesis)} \\
&\leq \left(1 + \frac{1}{d}\right)^d K && \text{(because } q_{t-d}(i) \geq p_{t-d}(i)\text{)} \\
&\leq Ke\ .
\end{aligned}$$

Hence, using Lemma 1,

$$p_{t+1}(i)\bigl(1 - \eta Ke\bigr) \leq p_{t+1}(i)\left(1 - \eta \sum_{j \in A} p_t(j)\widehat{\ell}_t(j)\right) \leq p_t(i)$$

which implies $p_{t+1}(i) \leq \bigl(1 + \frac{1}{d}\bigr)\, p_t(i)$ whenever $\eta \leq \frac{1}{Ke(d+1)}$. ∎

As we said in Section 1, the idea of controlling the drift of the probabilities in order to bound the effects of delayed feedback is not new. In particular, variants of Lemma 1 were already derived in the work of Neu et al. (2010, 2014). However, Lemma 2 appears to be new, and this is the key result to achieving our improvements.

4. The Cooperative Setting on a Communication Network

In our multi-agent bandit setting, there are N agents sitting on the vertices of a connected and undirected communication graph $G = (V, E)$, with $V = \{1, \dots, N\}$. The agents cooperate to solve the same instance of a nonstochastic bandit problem while limiting the communication among them. Let $N_s(v)$ be the set of nodes $v' \in V$ whose shortest-path distance $\mathrm{dist}_G(v, v')$ from v in G is exactly s. At each time step $t = 1, 2, \dots$, each agent $v \in V$ draws an action $I_t(v)$ from the common action set A. Note that each action $i \in A$ delivers the same loss $\ell_t(i) \in [0, 1]$ to all agents v such that $I_t(v) = i$. At the end of round t, each agent v observes his own loss $\ell_t\big(I_t(v)\big)$, and sends to his neighbors in G the message

$$m_t(v) = \Big\langle t, v, I_t(v), \ell_t\big(I_t(v)\big), \boldsymbol{p}_t(v) \Big\rangle$$

where $\boldsymbol{p}_t(v) = \big(p_t(1, v), \dots, p_t(K, v)\big)$ is the distribution of $I_t(v)$. Moreover, v also receives from his neighbors a variable number of messages $m_{t-s}(v')$. Each message $m_{t-s}(v')$ that v receives from a neighbor is used to update $\boldsymbol{p}_t(v)$ and then forwarded to the other neighbors only if $s < d$, otherwise it is dropped.[2] Here d is the maximum delay, a parameter of the communication protocol. Therefore, at the end of round t, each agent v receives one message $m_{t-s}(v')$ from each agent v' such that $\mathrm{dist}_G(v, v') = s$, where $s \in \{1, \dots, d\}$. Graph G can thus be seen as a synchronous multi-hop communication network where messages are broadcast, each hop causing a delay of one time step. Our learning protocol is summarized in Figure 1, while Figure 2 contains a pictorial example.

Our model is similar to the LOCAL communication model in distributed computing (Linial, 1992; Suomela, 2013), where the output of a node depends only on the inputs of other nodes in a constant-size neighborhood of it, and the goal is to derive algorithms whose running time is independent of the network size. (The main difference is that the task here has no completion time, however, also in our model influence on a node is only through a constant-size neighborhood of it.)

One aspect deserving attention is that, apart from the common delay parameter d, the agents need not share further information. In particular, the agents need not know neither the topology of the graph G nor the total number of agents N. In Section 5, we show that our distributed algorithm can also be analyzed when each agent v uses a personalized delay $d(v)$, thus doing away with the need of a common delay parameter, and guaranteeing a generally better performance.

Further graph notation is needed at this point. Given G as above, let us denote by $G_{\le d}$ the graph $(V, E_{\le d})$ where $(u, v) \in E_{\le d}$ if and only if the shortest-path distance between agents u and v in G is *at most* d (hence $G_{\le 1} = G$). Graph $G_{\le d}$ is sometimes called the d-th power of G. We also use G_0 to denote the graph $(V, \emptyset)$. Recall that an independent set of G is any subset $T \subseteq V$ such that no two $i, j \in T$ are connected by an edge in E. The largest size of an independent set is the *independence number* of G, denoted by $\alpha(G)$. Let d_G be the *diameter* of G (maximal length over all possible shortest paths between all pairs of nodes); then $G_{\le d_G}$ is a clique, and one can easily see that $N = \alpha(G_0) > \alpha(G) \ge \alpha(G_{\le 2}) \ge \cdots \ge \alpha(G_{\le d_G}) = 1$. We show in Section 4.1 that the collective performance of our algorithms depends on $\alpha(G_{\le d})$. If the graph G under consideration is directed (see Section 5), then $\alpha(G)$ is the independence number of the undirected graph obtained from G by disregarding edge orientation.

2. Dropping messages older than d rounds is clearly immaterial with respect to proving bandit regret bounds. We added this feature just to prove a point about the message complexity of the protocol. See Remark 10 in Section 5 for further discussion.

The cooperative bandit protocol
Parameters: Undirected communication graph $G = (V, E)$, hidden loss vectors $\boldsymbol{\ell}_t = \big(\ell_t(1), \dots, \ell_t(K)\big) \in [0, 1]^K$ for $t \ge 1$, delay d.
For $t = 1, 2, \dots$

1. Each agent $v \in V$ plays action $I_t(v) \in A$ drawn according to distribution $\boldsymbol{p}_t(v)$;
2. Each agent $v \in V$ observes loss $\ell_t\big(I_t(v)\big)$, sends to his neighbors the message $m_t(v)$, and receives from his neighbors messages $m_{t-s}(v')$;
3. Each agent $v \in V$ drops any message $m_{t-s}(v')$ received from some neighbor such that $s \ge d$, and forwards to the other neighbors the remaining messages.

Figure 1: The cooperative bandit protocol where all agents share the same delay parameter d.

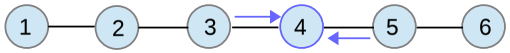


Figure 2: In this example, G is a line graph with $N = 6$ agents, and delay $d = 2$. At the end of time step t, Agent 4 sends to his neighbors 3 and 5 message $m_t(4)$, receives from Agent 3 messages $m_{t-1}(3)$ and $m_{t-2}(2)$, and from Agent 5 messages $m_{t-1}(5)$ and $m_{t-2}(6)$. Finally, Agent 4 forwards to Agent 5 message $m_{t-1}(3)$ and forwards to Agent 3 message $m_{t-1}(5)$. Any message older than $t - 1$ received by Agent 4 at the end of round t will not be forwarded to his neighbors.

The adversary generating losses is oblivious: loss vectors $\boldsymbol{\ell}_t = \big(\ell_t(1), \dots, \ell_t(K)\big) \in [0, 1]^K$ do not depend on the agents' internal randomization. The agents' goal is to control the *average welfare* regret R_T^{coop}, defined as

$$R_T^{\mathrm{coop}} = \left(\frac{1}{N} \sum_{v \in V} \mathbb{E}\left[\sum_{t=1}^{T} \ell_t\big(I_t(v)\big) \right] - \min_{i \in A} \sum_{t=1}^{T} \ell_t(i) \right) ,$$

the expectation being with respect to the internal randomization of each agent's algorithm. In the sequel, we write $\mathbb{E}_t[\cdot]$ to denote the expectation w.r.t. the product distribution $\prod_{v \in V} \boldsymbol{p}_t(v)$, conditioned on $I_1(v), \dots, I_{t-1}(v)$, $v \in V$.

4.1. The Exp3-Coop algorithm

Our first algorithm, called EXP3-COOP (Cooperative Exp3) is described in Figure 3. The algorithm works in the learning protocol of Figure 1. Each agent $v \in V$ runs the exponentially-weighted algorithm (1), combined with a "delayed" importance-weighted loss estimate

The Exp3-Coop Algorithm
Parameters: Undirected communication graph $G = (V, E)$; delay d; learning rate η.
Init: Each agent $v \in V$ sets weights $w_1(i, v) = 1$ for all $i \in A$.
For $t = 1, 2, \ldots$

1. Each agent $v \in V$ plays action $I_t(v) \in A$ drawn according to distribution $\boldsymbol{p}_t(v) = (p_t(1, v), \ldots, p_t(K, v))$, where
$$p_t(i, v) = \frac{w_t(i, v)}{W_t(v)},\ i = 1, \ldots, K, \qquad \text{and} \qquad W_t(v) = \sum_{j \in A} w_t(j, v)\ ;$$
2. Each agent $v \in V$ observes loss $\ell_t\big(I_t(v)\big)$ and exchanges messages with his neighbors (Steps 2 and 3 of the protocol in Figure 1);
3. Each agent $v \in V$ performs the update $w_{t+1}(i, v) = p_t(i, v)\, \exp\big(-\eta\, \widehat{\ell}_t(i, v)\big)$ for all $i \in A$, where
$$\widehat{\ell}_t(i, v) = \begin{cases} \dfrac{\ell_{t-d}(i)}{q_{d,t-d}(i, v)}\, B_{d,t-d}(i, v) & \text{if } t > d, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$
and $B_{d,t-d}(i, v) = \mathbb{I}\{\exists v' \in N_{\le d}(v) \,:\, I_{t-d}(v') = i\}$ with
$$q_{d,t-d}(i, v) = 1 - \prod_{v' \in N_{\le d}(v)} \big(1 - p_{t-d}(i, v')\big)\ .$$

Figure 3: The Exp3-Coop algorithm where all agents share the same delay parameter d.

$\widehat{\ell}_t(i, v)$ that incorporates the delayed information sent by the other agents. Specifically, denote by $N_{\le d}(v) = \bigcup_{s \le d} N_s(v)$ the set of nodes in G whose shortest-path distance from v is at most d, and note that, for all v, $\{v\} = N_{\le 0}(v) \subseteq N_{\le 1}(v) \subseteq N_{\le 2}(v) \subseteq \cdots$. If any of the agents in $N_{\le d}(v)$ has played at time $t - d$ action i (that is, $B_{d,t-d}(i, v) = 1$ in (3)), then the corresponding loss $\ell_{t-d}(i)$ is incorporated by v into $\widehat{\ell}_t(i, v)$. The denominator $q_{d,t-d}(i, v)$ is simply, conditioned on the history, the probability of $B_{d,t-d}(i, v) = 1$, i.e., $q_{d,t-d}(i, v) = \mathbb{E}_t[B_{d,t-d}(i, v)]$. Observe that $\{v\} \subseteq N_{\le d}(v)$ for all $d \ge 0$ implies $q_{d,t-d}(i, v) \ge p_{t-d}(i, v)$, as required by (2) so as to make Lemma 2 hold. It is also worth mentioning that, despite this is not strictly needed by our learning protocol, each agent v actually exploits the loss information gathered from playing action $I_t(v)$ only d time steps later. An extension of Exp3-Coop, where each loss is exploited as soon as it is made available to an agent, is analyzed in Section 6. Section 7, instead, studies an important special case of this setting where there is a single bandit agent receiving delayed feedback.

By their very definition, the loss estimates $\widehat{\ell}_t(\cdot, \cdot)$ at time t are determined by the realizations of $I_s(\cdot)$, for $s = 1, \ldots, t - d$. This implies that the numbers $p_t(\cdot, \cdot)$ defining $q_{d,t-d}(\cdot, \cdot)$, are determined by the realizations of $I_s(\cdot)$ for $s = 1, \ldots, t - d - 1$ (because the probabilities $\boldsymbol{p}_t(v)$ at time t are determined by the loss estimates up to time $t - 1$, see (1)). We have, for

all $t > d$, $i \in A$, and $v \in V$,
$$\mathbb{E}_{t-d}\Big[\widehat{\ell}_t(i, v)\Big] = \ell_{t-d}(i)\ . \tag{4}$$

Further, because of what we just said about $p_t(\cdot, \cdot)$ and $q_{d,t-d}(\cdot, \cdot)$ being determined by $I_1(\cdot), \ldots, I_{t-d-1}(\cdot)$, we also have
$$\mathbb{E}_{t-d}\Big[p_t(i, v)\widehat{\ell}_t(i, v)\Big] = p_t(i, v)\ell_{t-d}(i)\ , \quad \mathbb{E}_{t-d}\Big[p_t(i, v)\widehat{\ell}_t(i, v)^2\Big] = p_t(i, v)\frac{\ell_{t-d}(i)^2}{q_{d,t-d}(i, v)}\ . \tag{5}$$

The next lemma relates the variance of the estimates (3) to the structure of the communication graph G. The lemma is stated for a generic undirected communication graph G, but our application of it actually involves graph $G_{\le d}$.

Lemma 3 *Let $G = (V, E)$ be an undirected graph with independence number $\alpha(G)$. For each $v \in V$, let $N_{\le 1}(v)$ be the neighborhood of node v (including v itself), and $\boldsymbol{p}(v) = \big(p(1, v), \ldots, p(K, v)\big)$ be a probability distribution over $A = \{1, \ldots, K\}$. Then, for all $i \in A$,*
$$\sum_{v \in V} \frac{p(i, v)}{q(i, v)} \le \frac{1}{1 - e^{-1}} \left(\alpha(G) + \sum_{v \in V} p(i, v)\right) \quad \textit{where} \quad q(i, v) = 1 - \prod_{v' \in N_{\le 1}(v)} \big(1 - p(i, v')\big)\ .$$

Proof Fix $i \in A$ and set for brevity $P(i, v) = \sum_{v' \in N_{\le 1}(v)} p(i, v')$. We can write
$$\sum_{v \in V} \frac{p(i, v)}{q(i, v)} = \underbrace{\sum_{v \in V \,:\, P(i,v) \ge 1} \frac{p(i, v)}{q(i, v)}}_{\text{(I)}} + \underbrace{\sum_{v \in V \,:\, P(i,v) < 1} \frac{p(i, v)}{q(i, v)}}_{\text{(II)}}\ ,$$
and proceed by upper bounding the two terms (I) and (II) separately. Let $r(v)$ be the cardinality of $N_{\le 1}(v)$. We have, for any given $v \in V$,
$$\min\left\{q(i, v) \,:\, \sum_{v' \in N_{\le 1}(v)} p(i, v') \ge 1\right\} = 1 - \left(1 - \frac{1}{r(v)}\right)^{r(v)} \ge 1 - e^{-1}\ .$$
The equality is due to the fact that the minimum is achieved when $p(i, v') = \frac{1}{r(v)}$ for all $v' \in N_{\le 1}(v)$, and the inequality comes from $r(v) \ge 1$ (for, $v \in N_{\le 1}(v)$). Hence
$$\text{(I)} \le \sum_{v \in V \,:\, P(i,v) \ge 1} \frac{p(i, v)}{1 - e^{-1}} \le \sum_{v \in V} \frac{p(i, v)}{1 - e^{-1}}\ .$$
As for (II), using the inequality $1 - x \le e^{-x}, x \in [0, 1]$, with $x = p(i, v')$, we can write
$$q(i, v) \ge 1 - \exp\left(-\sum_{v' \in N_{\le 1}(v)} p(i, v')\right) = 1 - \exp\left(-P(i, v)\right)\ .$$

In turn, because $P(i,v) < 1$ in terms (II), we can use the inequality $1 - e^{-x} \geq (1 - e^{-1})\, x$, holding when $x \in [0,1]$, with $x = P(i,v)$, thereby concluding that $q(i,v) \geq (1-e^{-1})P(i,v)$. Thus

$$\text{(II)} \leq \sum_{v \in V\,:\, P(i,v)<1} \frac{p(i,v)}{(1-e^{-1})P(i,v)} \leq \frac{1}{1-e^{-1}} \sum_{v \in V} \frac{p(i,v)}{P(i,v)} \leq \frac{\alpha(G)}{1-e^{-1}}$$

where in the last step we used (Alon et al., 2017, Lemma 10). Note that despite the statement of this lemma refers to a directed graph and its maximum acyclic subgraph, in the special case of undirected graphs, the size of the maximum acyclic subgraph coincides with the independence number. Moreover, observe that $p(i,1), \ldots, p(i,N) \geq 0$ need not sum to one in order for this lemma to hold. ■

The following theorem quantifies the behavior of Exp3-Coop in terms of a free parameter γ in the learning rate, the tuning of which will be addressed in the subsequent Theorem 5.

Theorem 4 *The regret of* Exp3-Coop *run over a network $G = (V,E)$ of N agents, each using delay d and learning rate $\eta = \gamma/\big(Ke(d+1)\big)$, for $\gamma \in (0,1]$, satisfies*

$$R_T^{\text{coop}} \leq 2d + \frac{Ke(d+1)\ln K}{\gamma} + \gamma \left(\frac{\alpha(G_{\leq d})}{2(1-e^{-1})(d+1)N} + \frac{3}{Ke} \right) T \ .$$

Proof The standard analysis of the exponentially-weighted algorithm with importance-sampling estimates —see, e.g., the proof of (Alon et al., 2017, Lemma 1)— gives for each agent v and each action k the deterministic bound

$$\sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v) \leq \sum_{t=1}^{T} \widehat{\ell}_t(k,v) + \frac{\ln K}{\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v)^2 \ . \tag{6}$$

We take expectations of the three (double) sums in (6) separately. As for the first sum, note that an iterative application of Lemma 1 gives, for $t > d$,

$$p_t(i,v) \geq p_{t-d}(i,v) - \eta \sum_{s=1}^{d} p_{t-s}(i,v) \widehat{\ell}_{t-s}(i,v) \ ,$$

so that, setting for brevity $A_t(i,v) = \sum_{s=1}^{d} p_{t-s}(i,v) \widehat{\ell}_{t-s}(i,v)$, we have

$$\begin{aligned} \sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v) &\geq \sum_{t=2d+1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v) \\ &\geq \sum_{t=2d+1}^{T} \sum_{i=1}^{K} p_{t-d}(i,v) \widehat{\ell}_t(i,v) - \eta \sum_{t=2d+1}^{T} \sum_{i=1}^{K} A_t(i,v)\, \widehat{\ell}_t(i,v) \ . \end{aligned}$$

Hence

$$\begin{aligned} &\mathbb{E}\left[\sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v) \right] \geq \mathbb{E}\left[\sum_{t=2d+1}^{T} \sum_{i=1}^{K} p_{t-d}(i,v) \widehat{\ell}_t(i,v) \right] - \eta\, \mathbb{E}\left[\sum_{t=2d+1}^{T} \sum_{i=1}^{K} A_t(i,v)\, \widehat{\ell}_t(i,v) \right] \\ &= \mathbb{E}\left[\sum_{t=2d+1}^{T} \sum_{i=1}^{K} p_{t-d}(i,v)\, \mathbb{E}_{t-d}\left[\widehat{\ell}_t(i,v) \right] \right] - \eta\, \mathbb{E}\left[\sum_{t=2d+1}^{T} \sum_{i=1}^{K} A_t(i,v)\, \mathbb{E}_{t-d}\left[\widehat{\ell}_t(i,v) \right] \right] \\ &\text{(since } p_t(i,v) \text{ is determined by } I_1(\cdot), \ldots, I_{t-d-1}(\cdot)\text{)} \\ &= \mathbb{E}\left[\sum_{t=2d+1}^{T} \sum_{i=1}^{K} p_{t-d}(i,v)\, \ell_{t-d}(i) \right] - \eta\, \mathbb{E}\left[\sum_{t=2d+1}^{T} \sum_{i=1}^{K} A_t(i,v)\, \ell_{t-d}(i) \right] \qquad \text{(using (4))} \\ &\geq \mathbb{E}\left[\sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v)\, \ell_t(i) \right] - 2d - \eta\, T\, d \ . \end{aligned}$$

The last step uses

$$\begin{aligned} \mathbb{E}\left[\sum_{i=1}^{K} A_t(i,v)\, \ell_{t-d}(i) \right] &\leq \mathbb{E}\left[\sum_{i=1}^{K} A_t(i,v) \right] = \mathbb{E}\left[\sum_{i=1}^{K} \sum_{s=1}^{d} p_{t-s}(i,v) \widehat{\ell}_{t-s}(i,v) \right] \\ &= \mathbb{E}\left[\sum_{i=1}^{K} \sum_{s=1}^{d} p_{t-s}(i,v) \ell_{t-s-d}(i) \right] \leq \mathbb{E}\left[\sum_{i=1}^{K} \sum_{s=1}^{d} p_{t-s}(i,v) \right] = d \end{aligned}$$

holding for $t \geq 2d+1$. Similarly, for the second sum in (6), we have

$$\mathbb{E}\left[\sum_{t=1}^{T} \widehat{\ell}_t(k,v) \right] = \sum_{t=d+1}^{T} \ell_{t-d}(k) \leq \sum_{t=1}^{T} \ell_t(k) \ .$$

Finally, for the third sum in (6), an iterative application of Lemma 2 yields, for $t > d$,

$$p_t(i,v) \leq \left(1 + \frac{1}{d}\right)^d p_{t-d}(i,v) \leq e\, p_{t-d}(i,v) \ ,$$

so that we can write

$$\begin{aligned} \mathbb{E}\left[\sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v)^2 \right] &= \mathbb{E}\left[\sum_{t=d+1}^{T} \sum_{i=1}^{K} \mathbb{E}_{t-d}\left[p_t(i,v) \widehat{\ell}_t(i,v)^2 \right] \right] \\ &\leq \mathbb{E}\left[\sum_{t=d+1}^{T} \sum_{i=1}^{K} \frac{p_t(i,v)}{q_{d,t-d}(i,v)} \right] \qquad \text{(using (5) and } \ell_t(\cdot) \leq 1\text{)} \\ &\leq e\, \mathbb{E}\left[\sum_{t=d+1}^{T} \sum_{i=1}^{K} \frac{p_{t-d}(i,v)}{q_{d,t-d}(i,v)} \right] \ . \end{aligned}$$

The last inequality comes from an iterative application of Lemma 2, and the observation that $\left(1 + \frac{1}{d}\right)^d \leq e$.

Hence, summing over all agents v, dividing by N, and using Lemma 3 on $G_{\leq d}$ gives

$$\begin{aligned}\frac{1}{N}\,\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{K}\sum_{v\in V} p_t(i,v)\widehat{\ell}_t(i,v)^2\right] &\leq \frac{e}{N}\,\mathbb{E}\left[\sum_{t=d+1}^{T}\sum_{i=1}^{K}\sum_{v\in V}\frac{p_{t-d}(i,v)}{q_{d,t-d}(i,v)}\right]\\ &\leq \frac{e}{(1-e^{-1})\,N}\,\mathbb{E}\left[\sum_{t=d+1}^{T}\sum_{i=1}^{K}\left(\alpha(G_{\leq d})+\sum_{v\in V} p_{t-d}(i,v)\right)\right]\\ &\leq \frac{e}{1-e^{-1}}\,T\left(\frac{K}{N}\,\alpha(G_{\leq d})+1\right)\ .\end{aligned}$$

Finally, putting together as in (6), setting $\eta=\gamma/\big(Ke(d+1)\big)$, and overapproximating, we obtain the desired bound. ■

We might be tempted to optimize the bound of Theorem 4 for γ. However, this is not a legal learning rate setting in a distributed scenario, for the optimized value of γ would depend on the global quantities N and $\alpha(G_{\leq d})$. Thus, instead of this global tuning, we let each agent set its own learning rate γ through a "doubling trick" played locally. The doubling trick[3] works as follows. For each $v\in V$, we let $\gamma_r(v)=Ke(d+1)\sqrt{(\ln K)/2^r}$ for each $r=r_0,r_0+1,\ldots$, where $r_0=\big\lceil \log_2\ln K+2\log_2(Ke(d+1))\big\rceil$ is chosen in such a way that $\gamma_r(v)\leq 1$ for all $r\geq r_0$. Let T_r be the random set of consecutive time steps where the same $\gamma_r(v)$ was used. Whenever the local algorithm at v is running with $\gamma_r(v)$ and detects $\sum_{s\in T_r} Q_s(v)>2^r$, where $Q_s(v)$ is the quantity defined in (9) —see the proof of Theorem 5 in the appendix— then we restart this algorithm with $\gamma(v)=\gamma_{r+1}(v)$.

We have the following result (proof in the appendix).

Theorem 5 *The regret of* Exp3-Coop *run over a network $G=(V,E)$ of N agents, each using delay d, and an individual learning rate $\eta(v)=\gamma(v)/\big(Ke(d+1)\big)$, where $\gamma(v)\in(0,1]$ is adaptively selected by each agent through the above doubling trick, satisfies, when T grows large,*[4]

$$R_T^{\mathrm{coop}}=\mathcal{O}\left(\sqrt{(\ln K)\left(d+1+\frac{K}{N}\,\alpha(G_{\leq d})\right)T}+d\,\log T\right)\ .$$

Remark 6 *Theorem 5 shows a natural trade-off between delay and information. To make it clear, suppose $N\approx K$. In this case, the regret bound becomes of order $\sqrt{\big(d+\alpha(G_{\leq d})\big)T\ln K}+d\ln T$. Now, if d is as big as the diameter d_G of G, then $\alpha(G_{\leq d})=1$. This means that at every time step all $N\approx K$ agents observe (with some delay) the losses of each other's actions. This is very much reminiscent of a full information scenario, and in fact our bound becomes of order $\sqrt{(d_G+1)T\ln K}+d_G\ln T$, which is close to the full information minimax rate $\sqrt{(d+1)T\ln K}$ when feedback has a constant delay d (Weinberger and Ordentlich, 2002). When G is sparse (i.e., d_G is likely to be large, say $d_G\approx N$), then agents have no advantage in taking $d=d_G$ since $d_G\approx N\approx K$. In this case, agents may even give up cooperation (choosing $d=0$ in Figure 3), and fall back on the standard bandit bound $\sqrt{TK\ln K}$, which corresponds to running* Exp3-Coop *on the edgeless graph G_0. (No doubling trick is needed in this case, hence no extra $\log T$ term appears.)*

Remark 7 *When $d=d_G$, each neighborhood $N_{\leq d}(v)$ used in the loss estimate (3) is equal to V, hence all agents receive the same feedback. Because they all start off from the same initial weights, the agents end up computing the same updates. This in turn implies that: (1) the individual regret incurred by each agent is the same as the average welfare regret R_T^{coop}; (2) the messages exchanged by the agents (see Figure 1) may be shortened by dropping the distribution part $\boldsymbol{p}_{t-s}(v')$.*

Remark 8 *An interesting question is whether the agents can come up with a reasonable choice for the value of d even when they lack any information whatsoever about the global structure of G. A partial answer to this question follows. It is easy to show that the choice $d=\sqrt{K}$ in Theorem 5 yields a bound on the average welfare regret of the form $K^{1/4}\sqrt{T\ln K}+\sqrt{K}(\ln T)$ for all G (and irrespective to the value of $N=|V|$), provided G is connected. This holds because, for any connected graph G, the independence number $\alpha(G_{\leq d})$ is always bounded by*[5] $\big\lceil 2N/(d+2)\big\rceil$. *To see why this latter statement is true, observe that the neighborhood $N_{\leq d/2}(v)$ of any node v in $G_{\leq d/2}$ contains at least $d/2+1$ nodes (including v), and any pair of nodes $v',v''\in N_{\leq d/2}(v)$ are adjacent in $G_{\leq d}$. Therefore, no independent set of $G_{\leq d}$ can have size bigger than $\big\lceil 2N/(d+2)\big\rceil$. A more detailed bound is contained, e.g., in (Firby and Haviland, 1997).*

Remark 9 *The choice of d minimizing the bound $d+\frac{K}{N}\,\alpha(G_{\leq d})$ requires prior knowledge of the independence numbers $\alpha(G_{\leq d})$ for different values of d. Computing these quantities is NP-hard in general, even when G is fully known.*

5. Extension I: Cooperation with Individual Parameters

In this section, we analyze a modification of Exp3-Coop that allows each agent v in the network to use a delay parameter $d(v)$ different from that of the other agents. We then show how such individual delays may improve the average welfare regret of the agents. In the previous setting, where all agents use the same delay parameter d, messages have an implicit time-to-live equal to d. In this setting, however, agents may not have a detailed knowledge of the delay parameters used by the other agents. For this reason we allow an agent v to generate messages with a time-to-live $ttl(v)$ possibly different from the delay parameter $d(v)$. Note that the role of the two parameters $d(v)$ and $ttl(v)$ is inherently different. Whereas $d(v)$ rules the extent to which v uses the messages received from the other agents, $ttl(v)$ limits the number of times a message from v is forwarded to the other agents, thereby limiting the message complexity of the algorithm. In order to accomodate this additional parameter, we are required to modify the cooperative bandit protocol of

3. There has been some recent work on adaptive learning rate tuning applied to nonstochastic bandit algorithms (Kocák et al., 2014; Neu, 2015). One might wonder whether the same techniques may apply here as well. Unfortunately, the specific form of our update (1) makes this adaptation nontrivial, and this is why we resorted to a more traditional "doubling trick".

4. The big-oh notation here hides additive terms that are independent of T and do depend polynomially on the other parameters.

5. Because it holds for a worst-case (connected) G, this upper bound on $\alpha(G_{\leq d})$ can be made tighter when specific graph topologies are considered.

Figure 1. As in Section 4, we have an undirected communication network $G = (V, E)$ over the agents. However, in this new protocol the message that at the end of round t each agent v sends to his neighbors in G has the format

$$m_t(v) = \Big\langle t, v, ttl(v), I_t(v), \ell_t\big(I_t(v)\big), \boldsymbol{p}_t(v) \Big\rangle$$

where $ttl(v)$ is the time-to-live parameter of agent v. Each message $m_{t-s}(v')$, which v receives from a neighbor, first has its time-to-leave decremented by one. If the resulting value is positive, the message is forwarded to the other neighbors, otherwise it is dropped. Moreover, v uses this message to update $\boldsymbol{p}_t(v)$ only if $s \le d(v)$. Hence, at time t an agent v uses the message sent at time $t - s$ by v' if and only if $\mathrm{dist}_G(v', v) = s$ with $s \le \min\{d(v), ttl(v')\}$, where $\mathrm{dist}_G(v, v')$ is the shortest-path distance from v' to v in G.

Based on the collection $\mathcal{P} = \{d(v), ttl(v)\}_{v \in V}$ of individual parameters, we define the *directed* graph $G_{\mathcal{P}} = (V, E_{\mathcal{P}})$ as follows: arc $(v', v) \in E_{\mathcal{P}}$ if and only if $\mathrm{dist}_G(v, v') \le \min\{d(v), ttl(v')\}$. The in-neighborhood $N^-_{\mathcal{P}}(v)$ of v thus contains the set of all $v' \in V$ whose distance from v is not larger than $\min\{d(v), ttl(v')\}$. Notice that, with this definition, $v \in N^-_{\mathcal{P}}(v)$, so that $(V, E_{\mathcal{P}})$ includes all self-loops (v, v). Figure 4(a) illustrates these concepts through a simple pictorial example.

Remark 10 *It is important to remark that the communication structure encoded by $\mathcal{P}$ is an exogenous parameter of the regret minimization problem, and so our algorithms cannot trade it off against regret. In addition to that, the parameterization $\mathcal{P} = \{d(v), ttl(v)\}_{v \in V}$ defines a simple and static communication graph which makes it relatively easy to express regret as a function of the amount of available communication. This would not be possible if we had each individual node v decide whether to forward a message based, say, on its own local delay parameter $d(v)$. To see why, consider the situation where nodes v and v' are along the route of a message that is reaching v before v'. The decision of v to drop the message may clash with the willingness of v' to receive it, and this may clearly happen when $d(v) < d(v')$. The structure of the communication graph resulting from this individual behavior of the nodes would be rather complicated. On the contrary, the time-to-live-based parametrization, which is commonly used in communication networks to control communication complexity, does not have this issue.*

Figure 5 contains our algorithm (called Exp3-Coop2) for this setting. Exp3-Coop2 is a strict generalization of Exp3-Coop, and so is its analysis. The main difference between the two algorithms is that Exp3-Coop2 deals with directed graphs. This fact prevents us from using the same techniques of Section 4.1 in order to control the regret. Intuitively, adding orientations to the edges reduces the information available to the agents and thus increases the variance of their loss estimates. Thus, in order to control this variance, we need a lower bound[6] on the probabilities $p_t(i, v)$. From Figure 5, one can easily see that

$$1 = \sum_{i \in A} \frac{w_t(i,v)}{W_t(v)} \le \widetilde{P}_t(v) \le \sum_{i \in A} \left(\frac{w_t(i,v)}{W_t(v)} + \frac{\delta}{K} \right) = 1 + \delta \tag{7}$$

6. We find it convenient to derive this lower bound without mixing with the uniform distribution over A —see, e.g., (Auer et al., 2002b)— but in a slightly different manner. This facilitates our delayed feedback analysis.

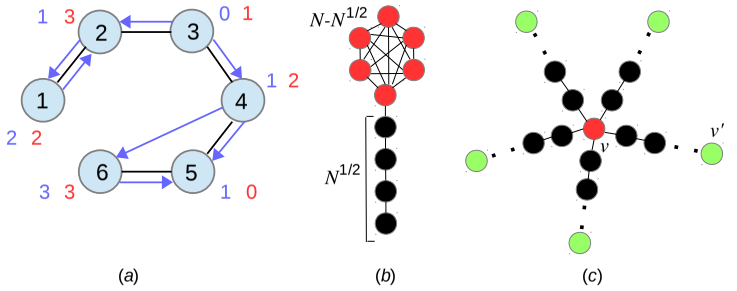


Figure 4: **(a)** In this example, the communication network G is an undirected line graph with $N = 6$ agents, whose edges are depicted in black. Close to each node $v = 1, \ldots, 6$ is the individual delay $d(v)$ (in blue), and the individual time-to-leave $ttl(v)$ (in red). The arcs (a.k.a., directed edges) of the induced directed graph $G_{\mathcal{P}}$ are also depicted in blue. Self-loops are not depicted. For instance, we have $N^-_{\mathcal{P}}(5) = \{4, 5, 6\}$ and $N^-_{\mathcal{P}}(3) = \{3\}$. **(b)** A communication network having a dense (red nodes) and a sparse (black nodes) region. The black region has $N^{1/2}$ agents, the red one has $N - N^{1/2}$ agents. **(c)** A star graph with long rays. The center v (in red) sets a small $d(v)$ and a large $ttl(v)$. The peripheral nodes v' (in green) set a large $d(v')$ and a small $ttl(v')$.

implying the lower bound $p_t(i, v) \ge \frac{\delta}{K(1+\delta)}$, holding for all i, t, and v.

The following theorem (proof in the appendix) is the main result of this section.

Theorem 11 *The regret of* Exp3-Coop2 *run over a network $G = (V, E)$ of N agents, each agent v using individual delay $d(v)$, individual time-to-leave $ttl(v)$, exploration parameter $\delta = 1/T$, and learning rate η such that $\eta \to 0$ as $T \to \infty$ satisfies, when T grows large,*

$$R_T^{\mathrm{coop}} = \mathcal{O}\left(\frac{\ln K}{\eta} + \eta\Big(\bar{d}_V + \frac{K}{N}\, \alpha\left(G_{\mathcal{P}}\right) \ln(TNK) \Big) T \right) \quad , \quad \text{where} \quad \bar{d}_V = \frac{1}{N} \sum_{v \in V} d(v) \ .$$

Using a doubling trick in much the same way we used it to prove Theorem 5, we can state the following result (proof in the appendix).

Corollary 12 *The regret of* Exp3-Coop2 *run over a network $G = (V, E)$ of N agents, each agent v using individual delay $d(v)$, individual time-to-leave $ttl(v)$, exploration parameter $\delta = 1/T$, and individual learning rate $\eta(v)$ adaptively selected by each agent through a doubling trick, satisfies, when T grows large*

$$R_T^{\mathrm{coop}} = \mathcal{O}\left(\sqrt{(\ln K)\left(\bar{d}_V + 1 + \frac{K}{N}\, \alpha(G_{\mathcal{P}}) \ln(TNK) \right) T} + \bar{d}_V \left(\ln T + \ln\ln(TNK) \right) \right) \ .$$

To illustrate the advantage of having individual delays as opposed to sharing the same delay value, it suffices to consider a communication network including regions of different density.

The Exp3-Coop2 Algorithm
Parameters: Undirected graph $G = (V, E)$; learning rate η; exploration parameter $\delta > 0$.
Init: Each $v \in V$ sets weights $w_1(i, v) = 1$, for all $i \in A$, delay $d(v)$, and time-to-live $ttl(v)$.
For $t = 1, 2, \ldots$

1. Each agent $v \in V$ plays action $I_t(v) \in A$ drawn according to distribution $\boldsymbol{p}_t(v) = (p_t(1, v), \ldots, p_t(K, v))$, where

$$p_t(i, v) = \frac{\widetilde{p}_t(i, v)}{\widetilde{P}_t(v)}, \qquad \widetilde{P}_t(v) = \sum_{j \in A} \widetilde{p}_t(j, v),$$

and

$$\widetilde{p}_t(i, v) = \max\left\{ \frac{w_t(i, v)}{W_t(v)}, \frac{\delta}{K} \right\}, \qquad W_t(v) = \sum_{j \in A} w_t(j, v)\,;$$

2. Each agent $v \in V$ observes loss $\ell_t\big(I_t(v)\big)$ and exchanges messages with his neighbors (see main text for an explanation);
3. Each agent $v \in V$ performs the update $w_{t+1}(i, v) = p_t(i, v)\, \exp\big(-\eta\, \widehat{\ell}_t(i, v)\big)$ for all $i \in A$, where

$$\widehat{\ell}_t(i, v) = \begin{cases} \dfrac{\ell_{t-d(v)}(i)}{q_{\mathcal{P}, t-d(v)}(i, v)}\, B_{\mathcal{P}, t-d(v)}(i, v) & \text{if } t > d(v), \\ 0 & \text{otherwise,} \end{cases}$$

and $B_{\mathcal{P}, t-d(v)}(i, v) = \mathbb{I}\{\exists v' \in N^-_{\mathcal{P}}(v) \,:\, I_{t-d(v)}(v') = i\}$, with

$$q_{\mathcal{P}, t-d(v)}(i, v) = 1 - \prod_{v' \in N^-_{\mathcal{P}}(v)} \big(1 - p_{t-d(v)}(i, v')\big)\,.$$

Figure 5: The Exp3-Coop2 algorithm with individual delay and time-to-live parameters.

Concretely, consider the graph in Figure 4(b) with a large densely connected region (red agents) and a small sparsely connected region black agents). In this example, the black agents prefer a large value of their individual delay so as to receive more information from nearby agents, but this comes at the price of a larger bias for their estimators $\widehat{\ell}_t(i, v)$. On the contrary, information from nearby agents is readily available to the red agents, so that they do not gain any regret improvement from a large delay parameter. A similar argument applies here to the individual time-to-live values: red agents v will set a small $ttl(v)$ to reduce communication. Black agents v' may decide to set $ttl(v')$ depending on their intention to reach the red nodes. But because the red agents have set a small $d(v)$, any effort made by v' trying to reach them would be a communication waste. Hence, it is reasonable for a black

agent v' to set a moderately large value for $ttl(v')$, but perhaps not so large as to reach the red agents. One can read this off the bounds in both Theorem 11 and Corollary 12, as explained next. Suppose for simplicity that $K \approx N$ so that, disregarding log factors, these bounds depend on parameters $\mathcal{P}$ only through the quantity $H = \bar{d}_V + \alpha\,(G_{\mathcal{P}})$. Now, in the case of a common delay parameter d (Section 4.1), it is not hard to see that the best setting for d in order to minimize H is of the form $d = N^{1/4}$, resulting in $H = \Theta(N^{1/4})$. On the other hand, the best setting for the individual delays is $d(v) = 1$ when v is red, and $d(v) = \sqrt{N}$ when v is black, resulting in $H = \Theta(1)$.

The time-to-live parameters $ttl(v)$ affect the regret bound only through $\alpha\,(G_{\mathcal{P}})$, but they clearly play the additional role of bounding the message complexity of the algorithm. In our example of Figure 4(b), we essentially have $d(v) \approx ttl(v)$ for all v. A typical scenario where agents may have $d(v) \neq ttl(v)$ is illustrated in Figure 4(c). In this case, we have star-like graph where a central agent is connected through long rays to all others agents. The center v prefers to set a small $d(v)$, since it has a large degree, but also a large $ttl(v)$ in order to reach the green peripheral nodes. The green nodes v' are reasonably doing the opposite: a large $d(v')$ in order to gather information from other nodes, but also a smaller time-to-live than the center, for the information transmitted by v' is comparatively less valuable to the whole network than the one transmitted by the center.

Agents can set their individual parameters in a topology-dependent manner using any algorithm for assessing the centrality of nodes in a distributed fashion —e.g., (Wehmuth and Ziviani, 2013), and references therein. This can be done at the beginning in a number of rounds which only depends on the network topology (but not on T). Hence, this initial phase would affect the regret bound only by an additive constant.

6. Extension II: Cooperation with Mixed Delays

The two algorithms we designed so far do not use the loss information in the most effective way, as they both postpone the update step by d (Figure 3) or $d(v)$ (Figure 5) time steps. The advantage of postponing updates is that loss estimates are simple to design, because the updates at time t all involve losses generated at the same time $t - d$ or $t - d(v)$. In this section, instead, we study generalized versions of Exp3-coop and Exp3-coop2 where all losses $\ell_{t-s}(i)$ sent from agents at distance s to any given agent v are used by v at time t; that is, they are used as soon as they become available to v. Unlike before, the updates performed at time t now involve losses with different delays, and for this reason these generalized algorithms combine many loss estimators, each defined for a different level of delay, through a fixed distribution D (e.g., a distribution emphasizing recent losses). As we show, in the resulting regret bounds both delays and independence numbers end up correspondingly mixed according to the distribution D. In the rest of this section, we present Exp3-Coop-Mix, which generalizes Exp3-Coop to mixed delayed estimators. The learning protocol remains the same (Figure 1). A similar extension exists for Exp3-Coop2, where instead of having individual delays $d(v)$, we have individual distributions over delay values. We decided to omit this further extension from the paper because it does not add any extra value to the overall discussion.

Exp3-Coop-Mix is described in Figure 6, where we simply replaced estimator (3) by its mixed version (8). The next theorem (whose proof is in the appendix) and the subsequent

The Exp3-Coop-Mix Algorithm
Parameters: Undirected communication graph $G = (V, E)$; maximal delay d; delay distribution D over $\{0, 1, \ldots, d-1\}$; learning rate $\eta > 0$.
Init: Each agent $v \in V$ sets weights $w_1(i, v) = 1$ for all $i \in A$.
For $t = 1, 2, \ldots$

1. Each agent $v \in V$ plays action $I_t(v) \in A$ drawn according to distribution $\boldsymbol{p}_t(v) = (p_t(1, v), \ldots, p_t(K, v))$, where

$$p_t(i, v) = \frac{w_t(i, v)}{W_t(v)},\ i = 1, \ldots, K \qquad \text{and} \qquad W_t(v) = \sum_{j \in A} w_t(j, v)\ ;$$

2. Each agent $v \in V$ observes loss $\ell_t\big(I_t(v)\big)$ and exchanges messages with his neighbors (Steps 2 and 3 of the protocol in Figure 1);
3. Each agent $v \in V$ performs the update $w_{t+1}(i, v) = p_t(i, v)\, \exp\big(-\eta\, \widehat{\ell}_t(i, v)\big)$ for all $i \in A$, where

$$\widehat{\ell}_t(i, v) = \begin{cases} \displaystyle\sum_{s=0}^{d-1} D(s)\, \frac{\ell_{t-s}(i)}{q_{s,t-s}(i, v)}\, B_{s,t-s}(i, v) & \text{if } t > d, \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

and $B_{s,t-s}(i, v) = \mathbb{I}\{\exists v' \in N_{\le s}(v) \,:\, I_{t-s}(v') = i\}$ with

$$q_{s,t-s}(i, v) = 1 - \prod_{v' \in N_{\le s}(v)} \big(1 - p_{t-s}(i, v')\big)\ .$$

Figure 6: The Exp3-Coop-Mix algorithm where all agents share the same delay distribution D.

discussion investigate the impact of using a distribution over delay levels on the average welfare regret bound.

Theorem 13 *The regret of* Exp3-Coop-Mix *run over a network $G = (V, E)$ of N agents, each using the same delay distribution D, and learning rate $\eta \le 1/(Ke(d+1))$, satisfies*

$$R_T^{\text{coop}} \le 3d + \frac{\ln K}{\eta} + \frac{e^2 \eta\, T}{1 - e^{-1}} \left(1 + \mu_D + \frac{2K}{N}\, \bar{\alpha}_D\right)\ ,$$

where $\bar{\alpha}_D = \sum_{s=0}^{d-1} D(s) \alpha(G_{\le s})$ is the expected independence number of G's power as determined by mixture D, and μ_D is the expectation of distribution D.

Through a doubling trick played locally by each agent, which is utterly similar to the one described before the statement of Theorem 5, we can state (proof in the appendix) the following corollary.

Corollary 14 *The regret of* Exp3-Coop-Mix *run over a network $G = (V, E)$ of N agents, each using the same delay distribution D, and individual learning rate $\eta(v)$ adaptively selected by each agent through a doubling trick, satisfies, when T grows large*

$$R_T^{\text{coop}} = \mathcal{O}\left(\sqrt{(\ln K)\left(\mu_D + 1 + \frac{K}{N}\, \bar{\alpha}_D)\right) T} + \mu_D \log T\right)\ .$$

7. Delayed Losses (for a Single Agent)

Exp3-Coop can be specialized to the setting where a single agent is facing a bandit problem in which the loss of the chosen action is observed with a fixed delay d. In this setting, at the end of each round t the agent incurs loss $\ell_t(I_t)$ and observes $\ell_{t-d}(I_{t-d})$, if $t > d$, and nothing otherwise. The regret is defined in the usual way,

$$R_T = \mathbb{E}\left[\sum_{t=1}^{T} \ell_t(I_t)\right] - \min_{i=1,\ldots,K} \sum_{t=1}^{T} \ell_t(i)\ .$$

This problem was studied by Weinberger and Ordentlich (2002) in the full information case, for which they proved that $\sqrt{(d+1)T \ln K}$ is the optimal order for the minimax regret. The result was extended to the bandit case by Neu et al. (2010, 2014) —see also Joulani et al. (2013)— whose techniques can be used to obtain a regret bound of order $\sqrt{(d+1)KT}$. Yet, no matching lower bound was available for the bandit case.

As a matter of fact, the upper bound $\sqrt{(d+1)KT}$ for the bandit case is easily obtained: just run in parallel $d+1$ instances of the minimax optimal bandit algorithm for the standard (no delay) setting, achieving $R_T \le \sqrt{KT}$ (ignoring constant factors). At each time step $t = (d+1)r + s$ (for $r = 0, 1, \ldots$ and $s = 0, \ldots, d$), use instance $s+1$ for the current play. Hence, the no-delay bound applies to every instance and, assuming $d+1$ divides T, we immediately obtain $R_T \le \sum_{s=1}^{d+1} \sqrt{K \frac{T}{d+1}} \le \sqrt{(d+1)KT}$, again, ignoring constant factors.

Next, we show that the machinery we developed in Section 4.1 delivers an improved upper bound on the regret for the bandit problem with delayed losses, and then we complement this result by providing a lower bound matching the upper bound up to log factors, thereby characterizing (up to log factors) the minimax regret for this problem.

Corollary 15 *In the nonstochastic bandit setting with $K \ge 2$ actions and delay $d \ge 0$, where at the end of each round t the predictor has access to the losses $\ell_1(I_1), \ldots, \ell_s(I_s) \in [0, 1]^K$ for $s = \max\{1, t-d\}$, the minimax regret is of order $\sqrt{(K+d)T}$, ignoring logarithmic factors.*

Proof In order to prove the upper bound, we use the exponentially-weighted algorithm with estimate (3) specialized to the case of one agent only, namely $B_{t-d}(i) = \mathbb{I}\{I_{t-d} = i\}$ and $q_{d,t-d}(i) = p_{t-d}(i)$. Notice that this amounts to running the standard Exp3 algorithm performing an update as soon a new loss becomes available. In this case, because $N = \alpha(G_{\le d}) = 1$, the bound of Theorem 4, with a suitable choice of γ (which depends on T, K, and d) reduces to

$$R_T = \mathcal{O}\left(d + \sqrt{(K+d)\, T \ln K}\right)\ .$$

We now prove a lower bound matching our upper bound up to logarithmic factors. The proof hinges on combining the known lower bound $\Omega\big(\sqrt{KT}\big)$ for bandits without delay of Auer et al. (2002b) with the following argument by Weinberger and Ordentlich (2002) that provides a lower bound for the full information case with delay. The proof of the latter bound is by contradiction: we show that a low-regret full information algorithm for delay $d > 0$ can be used to design a low-regret full information algorithm for the $d = 0$ (no delay) setting. We then apply the known lower bound for the minimax regret in the no-delay setting to derive a lower bound for the setting with delay.

Fix $d > 0$ and let $\mathcal{A}$ be a predictor for the full-information online prediction problem with delay d. Let $\boldsymbol{p}_t$ be the probability distribution used by $\mathcal{A}$ at time t. We now apply algorithm $\mathcal{A}$ to design a new algorithm $\mathcal{A}'$ for a full information online prediction problem with arbitrary loss vectors $\boldsymbol{\ell}'_1, \dots, \boldsymbol{\ell}'_B \in [0,1]^K$ and no delay. More specifically, we create a sequence $\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_T \in [0,1]^K$ of loss vectors such that $T = (d+1)B$ and $\boldsymbol{\ell}_t = \boldsymbol{\ell}'_b$ where $b = \big\lceil t/(d+1) \big\rceil$. At each time $b = 1, \dots, B$ algorithm $\mathcal{A}'$ uses the distribution

$$\boldsymbol{p}'_b = \frac{1}{d+1} \sum_{s=1}^{d+1} \boldsymbol{p}_{(d+1)(b-1)+s}$$

where $\boldsymbol{p}_t = \big(\frac{1}{K}, \dots, \frac{1}{K}\big)$ for all $t \le 1$. Note that $\boldsymbol{p}'_b$ is defined using $\boldsymbol{p}_{(d+1)(b-1)+1}, \dots, \boldsymbol{p}_{(d+1)b}$. These are in turn defined using the same loss vectors $\boldsymbol{\ell}'_1, \dots, \boldsymbol{\ell}'_{b-1}$ since, by definition, each $\boldsymbol{p}_{t+1}$ uses $\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_{t-d}$, and $\big\lceil (t-d)/(d+1) \big\rceil = b-1$ for all $t = (d+1)(b-1), \dots, (d+1)b-1$. So $\mathcal{A}'$ is a legitimate full-information online algorithm for the problem $\boldsymbol{\ell}'_1, \dots, \boldsymbol{\ell}'_B$ with no delay. As a consequence,

$$\begin{aligned}
\sum_{t=1}^{T} \sum_{i=1}^{K} \ell_t(i) p_t(i) &= \sum_{b=1}^{B} \sum_{s=1}^{d+1} \sum_{i=1}^{K} \ell'_b(i) p_{(d+1)(b-1)+s}(i) \\
&= (d+1) \sum_{b=1}^{B} \sum_{i=1}^{K} \frac{1}{d+1} \sum_{s=1}^{d+1} \ell'_b(i) p_{(d+1)(b-1)+s}(i) \\
&= (d+1) \sum_{b=1}^{B} \sum_{i=1}^{K} \ell'_b(i) p'_b(i) \ .
\end{aligned}$$

Moreover,

$$\min_{k \in A} \sum_{t=1}^{T} \ell_t(k) = (d+1) \min_{k \in A} \sum_{b=1}^{B} \ell'_b(k) \ .$$

Since we know that for any predictor $\mathcal{A}'$ there exists a loss sequence $\boldsymbol{\ell}'_1, \boldsymbol{\ell}'_2, \dots$ such that the regret of $\mathcal{A}'$ is at least $\big(1 - o(1)\big)\sqrt{(T/2)\ln K}$, where $o(1) \to 0$ for $K, B \to \infty$, we have that the regret of $\mathcal{A}$ is at least

$$(d+1) R_{T/(d+1)}(\mathcal{A}') = \big(1 - o(1)\big)(d+1)\sqrt{\frac{T}{2(d+1)} \ln K} = \big(1 - o(1)\big)\sqrt{(d+1)\frac{T}{2} \ln K} \ ,$$

where $R_{T/(d+1)}(\mathcal{A}')$ is the regret of $\mathcal{A}'$ over $T/(d+1)$ time steps. The proof is completed by observing that that the regret of any predictor in the bandit setting with delay d cannot be

smaller than the regret of the predictor in the bandit setting with no delay or smaller than the regret of the predictor in the full information setting with delay d. Hence, the minimax regret in the bandit setting with delay d must be at least of order

$$\max\Big\{\sqrt{KT}, \sqrt{(d+1)T \ln K}\Big\} = \Omega\left(\sqrt{(K+d)\,T}\right) \ ,$$

ignoring log factors. This concludes the proof. ■

Recent results by Joulani et al. (2016)— see also (Quanrud and Khashabi, 2015)— consider a full information setting with variable delays $d_t > 0$. At the end of each round t, the agent simultaneously observes all loss vectors $\boldsymbol{\ell}_s$ such that $s + d_s = t$. For this setting, they prove a regret bound of order $\sqrt{(D+T)\ln K}$, where D is the sum of the delays d_t. The proof is based on a generic algorithm (SOLID) which simply feeds the loss vectors, as soon as they become available, to any deterministic predictor BASE designed to operate in a setting without delays. At the beginning of each round t, SOLID predicts using the current distribution $\boldsymbol{p}_t$ of BASE. A general argument shows that the regret of SOLID is equal to the regret of BASE on the sequence of loss vectors permuted according to the delays, plus a term that accounts for the drift of the distributions $\boldsymbol{p}_t$. For the bandit setting, we conjecture an upper bound on the regret of order $\sqrt{(D+KT)\ln K}$. However, we have not been able to prove this result via a direct application of our techniques.

8. Conclusions and Open Questions

We have investigated a cooperative and nonstochastic bandit scenario where cooperation comes at the price of delayed information. We have proven average welfare regret bounds that exhibit a natural tradeoff between amount cooperation and delay, the tradeoff being ruled by the underlying communication network topology. As a by-product of our analysis, we have also provided the first characterization to date of the regret of learning with (constant) delayed feedback in an adversarial bandit setting. There are a number of possible extensions of this work:

1. Our analysis only delivers average welfare regret bounds. It would be interesting to show simultaneous regret bounds that hold for each agent individually. We conjecture that the individual regret bound of an agent v is of the form

$$\sqrt{(\ln K)\left(d + \frac{K}{|N_{\le d}(v)|}\right) T}$$

   where $|N_{\le d}(v)|$ is the degree of v in $G_{\le d}$ (plus one). Such bound would in fact imply, e.g., the one in Theorem 5. A possible line of attack to solve this problem could be the use of graph sparsity along the lines of (Pan et al., 2015; Duchi et al., 2013; Mania et al., 2015; McMahan and Streeter, 2014).

2. It would be nice to characherize the average welfare regret by complementing our upper bounds with suitable *lower* bounds. For example, is the upper bound of Theorem 5 optimal in the communication model considered here?

3. It is natural to think of ways to adaptively tune our algorithms so as to automatically determine the best local parameters, e.g., the delay parameter d. For instance, disregarding message complexity, is there a way for each agent to adaptively tune d locally so to minimize the bound in Theorem 5?

4. Our messages $m_t(v)$ contain both action/loss information and distribution information. Is it possible to drop the distribution information and still achieve average welfare regret bounds similar to those in Theorems 4, 11, and 13?

5. Besides settling the conjecture advanced at the end of Section 7, we generally think that the study of learning on a communication network with time-varying delays, and its impact on the regret rates, is a topic worth of attention.

Acknowledgments

Parts of this work were done while NCB and CG were visiting YM at Tel-Aviv University. NCB gratefully acknowledges partial support from EU through the COST Action CA16228 (GAMENET). YM was supported in part by a grant from the Israel Science Foundation (ISF), and the Israeli Centers of Research Excellence (I-CORE) program (Center No. 4/11).

Proofs from Section 4.1

Proof of Theorem 5. We start off from first part of the proof of Theorem 4 which, after rearranging terms, gives the following bound for each agent v:

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{K} p_t(i,v)\ell_t(i)\right] - \sum_{t=1}^{T}\ell_t(k) \\
&\le 2d + \mathbb{E}\left[\frac{\ln K}{\eta(v)} + \eta(v)\,d^2 + \eta(v)\sum_{t=d+1}^{T}\left(d + \frac{e}{2}\sum_{i=1}^{K}\frac{p_{t-d}(i,v)}{q_{d,t-d}(i,v)}\right)\right] \\
&\le 3d + \mathbb{E}\left[\frac{Ke(d+1)\ln K}{\gamma(v)} + \frac{\gamma(v)}{Ke(d+1)}\sum_{t=1}^{T}\underbrace{\left(\mathbb{I}\{t>d\}\,d + \frac{e}{2}\left(\sum_{i=1}^{K}\frac{p_{t-d}(i,v)}{q_{d,t-d}(i,v)}\right)\mathbb{I}\{t>d\}\right)}_{Q_t(v)}\right] .
\end{aligned} \tag{9}$$

Note that the optimal tuning of $\gamma(v)$ depends on the random quantity

$$\overline{Q}_T(v) = \sum_{t=1}^{T} Q_t(v) \ .$$

We now apply the doubling trick to each instance of Exp3-Coop. Recall that, for each $v \in V$, we let $\gamma_r(v) = Ke(d+1)\sqrt{(\ln K)/2^r}$ for each $r = r_0, r_0+1, \ldots$, where $r_0 = \left\lceil \log_2 \ln K + 2\log_2(Ke(d+1))\right\rceil$ is chosen in a way that $\gamma_r(v) \le 1$ for all $r \ge r_0$. Let T_r be the random set of consecutive time steps where the same $\gamma_r(v)$ was used. Whenever the algorithm is running with $\gamma_r(v)$ and detects $\sum_{s\in T_r} Q_s(v) > 2^r$, then we restart the algorithm with $\gamma(v) = \gamma_{r+1}(v)$. The largest $r = r(v)$ we need is $\left\lceil \log_2 \overline{Q}_T(v)\right\rceil$ and

$$\sum_{r=r_0}^{\left\lceil \log_2 \overline{Q}_T(v)\right\rceil} 2^{r/2} < 5\sqrt{\overline{Q}_T(v)} \ .$$

Because of (9), the regret agent v suffers when using $\gamma_r(v)$ within T_r is at most $3d + 2\sqrt{(\ln K)2^r}$. Now, since we pay at most regret d at each restart, we have

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i} p_t(i,v)\ell_t(i)\right] - \sum_{t=1}^{T}\ell_t(k) &\le 3d + 4Ke(d+1)\ln K \\
&\quad + \mathbb{E}\left[10\sqrt{(\ln K)\overline{Q}_T(v)} + 3d\left\lceil \log_2 \overline{Q}_T(v)\right\rceil\right] .
\end{aligned}$$

The term $3d + 4Ke(d+1)\ln K$ bounds the regret when the algorithm is never restarted implying that only $\gamma_{r_0}(v)$ is used.

Taking averages with respect to v, using Jensen's inequality multiple times, and applying the deterministic bound

$$\frac{1}{N}\sum_{v\in V}\overline{Q}_T(v) \le \left(d + \frac{e}{2(1-e^{-1})}\,\frac{K\,(\alpha(G_{\le d})+1)}{N}\right) T$$

derived with the aid of Lemma 3 at the end of the proof of Theorem 4, gives

$$\begin{aligned} R_T^{\text{coop}} &\le 3d + 4Ke(d+1)\ln K \\ &\quad + 10\sqrt{(\ln K)\mathbb{E}\left[\frac{1}{N}\sum_{v\in V}\overline{Q}_T(v)\right]} + 3d\log_2\left(\mathbb{E}\left[\frac{1}{N}\sum_{v\in V}\overline{Q}_T(v)\right]\right) \\ &\le 10\sqrt{(\ln K)\left(d + \frac{e}{2(1-e^{-1})}\,\frac{K\left(\alpha(G_{\le d})+1\right)}{N}\right)T} + 3d\log_2 T + C\,, \end{aligned}$$

where C is independent of T and depends polynomially on the other parameters. Hence, as T grows large,

$$R_T^{\text{coop}} = \mathcal{O}\left(\sqrt{(\ln K)\left(d+1+\frac{K}{N}\,\alpha(G_{\le d})\right)T} + d\,\log T\right)$$

as claimed. ■

Proofs from Section 5

We first need to adapt the preliminary Lemmas 1 and 2 to the new update rule of Exp3-Coop2 contained in Figure 5.

Lemma 16 *Under the update rule contained in Figure 5, for all $t \ge 1$, for all $i \in A$, and for all $v \in V$*

$$\begin{aligned} -p_t(i,v)\left(\eta\widehat{\ell}_t(i,v)+\delta\right) &\le p_{t+1}(i,v) - p_t(i,v) \\ &\le p_{t+1}(i,v)\sum_{j=1}^{K} p_t(j,v)\left(1-\mathbb{I}\{\widetilde{p}_{t+1}(i,v)>\delta/K\}\big(1-\eta\,\widehat{\ell}_t(i,v)\big)\right) \end{aligned}$$

holds deterministically with respect to the agents' randomization.

Proof For the lower bound, we have

$$p_{t+1}(i,v) - p_t(i,v) = \frac{\widetilde{p}_{t+1}(i,v)}{\widetilde{P}_{t+1}(v)} - p_t(i,v) \ge \frac{w_{t+1}(i,v)}{W_{t+1}(v)\,\widetilde{P}_{t+1}(v)} - p_t(i,v)\,.$$

Since $W_{t+1}(v) = \sum_{i\in A} p_t(i,v)e^{-\eta\widehat{\ell}_t(i,v)} \le \sum_{i\in A} p_t(i,v) = 1$, and $\widetilde{P}_{t+1}(v) \le 1+\delta$ by (7), we can write

$$\begin{aligned} p_{t+1}(i,v) - p_t(i,v) &\ge \frac{w_{t+1}(i,v)}{1+\delta} - p_t(i,v) \\ &= p_t(i,v)\left(\frac{e^{-\eta\widehat{\ell}_t(i,v)}}{1+\delta} - 1\right) \\ &\ge p_t(i,v)\left(\frac{1-\eta\widehat{\ell}_t(i,v)}{1+\delta} - 1\right) \qquad (\text{using } e^{-x}\ge 1-x) \\ &\ge p_t(i,v)\left(-\delta - \eta\widehat{\ell}_t(i,v)\right) \end{aligned}$$

as claimed. As for the upper bound, we first claim that

$$\frac{w_{t+1}(i,v)}{W_{t+1}(v)} \ge p_{t+1}(i,v)\mathbb{I}\{\widetilde{p}_{t+1}(i,v) > \delta/K\}\,. \tag{10}$$

To prove (10), we recall that $\widetilde{p}_{t+1}(i,v) = \max\left\{\frac{w_{t+1}(i,v)}{W_{t+1}(v)}, \frac{\delta}{K}\right\}$. Then we distinguish two cases:

1. If $\frac{w_{t+1}(i,v)}{W_{t+1}(v)} \le \frac{\delta}{K}$, then $\widetilde{p}_{t+1}(i,v) = \delta/K$, and $w_{t+1}(i,v)/W_{t+1}(v) > 0$ by definition, hence (10) holds;
2. If $\frac{w_{t+1}(i,v)}{W_{t+1}(v)} > \frac{\delta}{K}$ then $\widetilde{p}_{t+1}(i,v) = \frac{w_{t+1}(i,v)}{W_{t+1}(v)}$, so that $p_{t+1}(i,v) \le p_{t+1}(i,v)\,\widetilde{P}_{t+1}(v) = \widetilde{p}_{t+1}(i,v)$ and (10) again holds.

Then, setting for brevity $C = \mathbb{I}\{\widetilde{p}_{t+1}(i,v) > \delta/K\}$, we can write

$$\begin{aligned} p_{t+1}(i,v) - p_t(i,v) &\le p_{t+1}(i,v) - w_{t+1}(i,v) && (\text{from the update (1)}) \\ &\le p_{t+1}(i,v) - W_{t+1}(v)p_{t+1}(i,v)\,C && (\text{using (10)}) \\ &= p_{t+1}(i,v)\big(1 - W_{t+1}(v)\,C\big) \\ &= p_{t+1}(i,v)\left(\sum_{j\in A}\big(p_t(j,v) - C\,w_{t+1}(j,v)\big)\right) \\ &= p_{t+1}(i,v)\sum_{j\in A} p_t(j,v)\left(1 - C\,e^{-\eta\widehat{\ell}_t(j,v)}\right) \\ &\le p_{t+1}(i,v)\sum_{j\in A} p_t(j,v)\left(1 - C(1-\eta\,\widehat{\ell}_t(j,v))\right) \end{aligned}$$

where in the last step we again used $e^{-x} \ge 1-x$. This concludes the proof. ■

Lemma 17 *Under the update rule contained in Figure 5, if $\delta \le 1/d(v)$ and $\eta \le \frac{1}{Ke(d(v)+1)}$, then*

$$p_{t+1}(i,v) \le \left(1 + \frac{1}{d(v)}\right) p_t(i,v) \tag{11}$$

holds for all $t \ge 1$ and $i \in A$, deterministically with respect to the agents' randomization.

Proof If $\widetilde{p}_{t+1}(i,v) = \delta/K$ then, from (7), we have $\delta/K = p_{t+1}(i,v)\widetilde{P}_{t+1}(v) \ge p_{t+1}(i,v)$, and $p_t(i,v) \ge \frac{\delta}{K(1+\delta)}$. Hence, $\frac{p_{t+1}(i,v)}{p_t(i,v)} \le \frac{\delta/K}{\delta/(K(1+\delta))} = 1+\delta$, so the claim follows from $\delta \le \frac{1}{d(v)}$. On the other hand, if $\widetilde{p}_{t+1}(i,v) > \delta/K$, then the proof is exactly the same as the proof of Lemma 2, for the second inequality in the statement of Lemma 16 turns out to be exactly the same as the corresponding inequality in the statement in Lemma 1. ■

Next, we generalize Lemma 3 to the case of directed graphs. This is where we need a lower bound on the probabilities $p_t(i,v)$. If $G = (V,E)$ is a directed graph, then for each $v \in V$ let $N_{\le 1}^{-}(v)$ be the in-neighborhood of node v (i.e., the set of $v' \in V$ such that arc $(v',v) \in E$), including v itself.

Lemma 18 *Let $G = (V, E)$ be a directed graph with independence number $\alpha(G)$. Let $\boldsymbol{p}(v) = \big(p(1,v), \dots, p(K,v)\big)$ be a probability distribution over $A = \{1, \dots, K\}$ such that $p(i,v) \geq \frac{\delta}{K(1+\delta)}$. Then, for all $i \in A$,*

$$\sum_{v \in V} \frac{p(i,v)}{q(i,v)} \leq \frac{1}{1-e^{-1}} \left(6\,\alpha(G) \ln\left(1 + \frac{N^2 K(1+\delta)}{\delta}\right) + \sum_{v \in V} p(i,v) \right) ,$$

where $q(i,v) = 1 - \prod_{v' \in N^-_{\leq 1}(v)} \big(1 - p(i,v')\big)$.

Proof We follow the notation and the proof of Lemma 3, where it is shown that

$$\sum_{v \in V} \frac{p(i,v)}{q(i,v)} \leq \frac{1}{1-e^{-1}} \sum_{v \in V} \left(\frac{p(i,v)}{P(i,v)} + p(i,v) \right) .$$

In order to bound from above the sum $\sum_{v \in V} \frac{p(i,v)}{P(i,v)}$, we combine (Alon et al., 2017, Lemma 14 and 16) and derive the upper bound

$$\sum_{v \in V} \frac{p(i,v)}{P(i,v)} \leq 6\,\alpha(G) \ln\left(1 + \frac{N^2 K(1+\delta)}{\delta}\right)$$

holding when $p(i,v) \geq \frac{\delta}{K(1+\delta)}$. Again, the probabilities $p(i,1), \dots, p(i,N) \geq 0$ need not sum to one in order for this lemma to apply. ■

With the above three lemmas handy, we are ready to prove Theorem 11.

Proof of Theorem 11. This proof is similar to the proof of Theorem 4, hence we only emphasize the differences between the two.

From the update rule in Figure 5, we have, for each $v \in V$,

$$\begin{aligned}
W_{T+1}(v) &= \sum_{i=1}^{K} \frac{\widetilde{p}_T(i)}{\widetilde{P}_T(v)} e^{-\eta \widehat{\ell}_T(i,v)} \\
&\geq \sum_{i=1}^{K} \frac{w_T(i,v)}{W_T(v) \widetilde{P}_T(v)} e^{-\eta \widehat{\ell}_T(i,v)} \qquad \text{(since } \widetilde{p}_T(i) \geq w_T(i,v)/W_T(v)\text{)} \\
&= \sum_{i=1}^{K} \frac{\widetilde{p}_{T-1}(i,v) e^{-\eta \widehat{\ell}_{T-1}(i,v)} e^{-\eta \widehat{\ell}_T(i,v)}}{W_T(v) \widetilde{P}_{T-1}(v) \widetilde{P}_T(v)} \\
&\vdots \\
&\geq \sum_{i=1}^{K} \frac{w_1(i,v)\, e^{-\eta \sum_{t=1}^{T} \widehat{\ell}_t(i,v)}}{W_1(v) \cdots W_T(v) \widetilde{P}_1(v) \cdots \widetilde{P}_T(v)} .
\end{aligned}$$

Now, because $w_1(i,v) = 1$, $W_1(v) = K$, and $\widetilde{P}_t(v) \leq 1 + \delta$ for all t, see (7), the above chain of inequalities implies that, for any fixed action $k \in A$,

$$(1+\delta)^T K \left(\prod_{t=1}^{T} W_{t+1}(v) \right) \geq e^{-\eta \sum_{t=1}^{T} \widehat{\ell}_t(k,v)} . \tag{12}$$

As usual, the quantity $W_{t+1}(v)$ can be upper bounded as

$$\begin{aligned}
W_{t+1}(v) &= \sum_{i=1}^{K} p_t(i,v) e^{-\eta \widehat{\ell}_t(i,v)} \\
&\leq \sum_{i=1}^{K} p_t(i,v) \left(1 - \eta \widehat{\ell}_t(i,v) + \frac{\eta^2}{2} \widehat{\ell}_t(i,v)^2 \right) \\
&\quad \text{(from } e^{-x} \leq 1 - x + x^2/2 \text{ for all } x \geq 0\text{)} \\
&= 1 - \eta \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v) + \frac{\eta^2}{2} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v)^2 .
\end{aligned}$$

Substituting into (12) and taking logs of both sides gives

$$\begin{aligned}
&T \ln(1+\delta) + \ln K + \sum_{t=1}^{T} \ln\left(1 - \eta \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v) + \frac{\eta^2}{2} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v)^2 \right) \\
&\qquad \geq -\eta \sum_{i=1}^{K} \widehat{\ell}_t(k,v) .
\end{aligned}$$

Finally, using $\ln(1+x) \leq x$, dividing by η, using $\delta = 1/T$, and rearranging yields

$$\sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v) \leq \frac{1 + \ln K}{\eta} + \sum_{t=1}^{T} \widehat{\ell}_t(k,v) + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v)^2 , \tag{13}$$

hence arriving at the counterpart of (6). From this point on, we proceed as in the proof of Theorem 4 by taking expectation on the three (double) sums in (13). Note that we do still have, for all $v \in V$, $t > d(v)$, and $i \in A$,

$$\begin{aligned}
\mathbb{E}_{t-d(v)}\Big[\widehat{\ell}_t(i,v)\Big] &= \ell_{t-d(v)}(i) \\
\mathbb{E}_{t-d(v)}\Big[p_t(i,v) \widehat{\ell}_t(i,v)\Big] &= p_t(i,v) \ell_{t-d(v)}(i) \\
\mathbb{E}_{t-d(v)}\Big[p_t(i,v) \widehat{\ell}_t(i,v)^2\Big] &= p_t(i,v) \frac{\ell_{t-d(v)}(i)^2}{q_{\mathcal{P},t-d(v)}(i,v)} .
\end{aligned}$$

We can write

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v) \right] &\geq \mathbb{E}\left[\sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v)\, \ell_t(i) \right] - 2d(v) - (\eta + \delta)\, T\, d(v) \\
\mathbb{E}\left[\sum_{t=1}^{T} \widehat{\ell}_t(k,v) \right] &\leq \sum_{t=1}^{T} \ell_t(k)
\end{aligned}$$

and, as in the proof of Theorem 4,

$$\mathbb{E}\left[\sum_{t=1}^{T} \sum_{i=1}^{K} p_t(i,v) \widehat{\ell}_t(i,v)^2 \right] \leq e\, \mathbb{E}\left[\sum_{t=d(v)+1}^{T} \sum_{i=1}^{K} \frac{p_{t-d(v)}(i,v)}{q_{\mathcal{P},t-d(v)}(i,v)} \right] .$$

Summing over all agents v, dividing by N, and applying Lemma 18 to the directed graph $G_{\mathcal{P}}$, the last inequality gives

$$\frac{1}{N}\,\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{K}\sum_{v\in V} p_t(i,v)\widehat{\ell}_t(i,v)^2\right] \leq \frac{e}{1-e^{-1}}\,T\left(\frac{6K}{N}\,\alpha\left(G_{\mathcal{P}}\right)\,\ln\left(1+2TN^2K\right)+1\right)\,.$$

Combining as in (13), recalling that $\delta = 1/T$, and setting for brevity $\bar{d}_V = \frac{1}{N}\sum_{v\in V} d(v)$, we have thus obtained that the average welfare regret of Exp3-Coop2 satisfies

$$\begin{aligned} R_T^{\text{coop}} &\leq 3\bar{d}_V + \eta\,T\,\bar{d}_V + \frac{1+\ln K}{\eta} + \frac{e\eta}{2(1-e^{-1})}\,T\left(\frac{6K}{N}\,\alpha\left(G_{\mathcal{P}}\right)\,\ln\left(1+2TN^2K\right)+1\right) \\ &= \mathcal{O}\left(\eta\,T\,\bar{d}_V + \frac{\ln K}{\eta} + \frac{\eta TK}{N}\,\alpha\left(G_{\mathcal{P}}\right)\,\ln\left(TNK\right)\right) \end{aligned}$$

as T grows large. This concludes the proof. ■

Proofs from Section 6

The following are further generalizations of Lemmas 1 and 2 that apply to mixed delay estimators.

Lemma 19 *Let $\widehat{\ell}_t(i,v)$ be of the form (8) for each $t \geq 1$, $i \in A$, and $v \in V$. If $\eta \leq \frac{1}{Ke(d+1)}$ in the update rule $w_{t+1}(i,v) = p_t(i,v)\,\exp\bigl(-\eta\,\widehat{\ell}_t(i,v)\bigr)$ then*

$$p_{t+1}(i,v) \leq \left(1+\frac{1}{d}\right) p_t(i,v)$$

holds for all $t \geq 1$, $i \in A$, and $v \in V$ deterministically with respect to the agent's randomization.

Proof The proof follows a similar inductive argument as in the proof of Lemma 2. We can write, for $t > d$,

$$\begin{aligned} \sum_{i\in A} p_t(i,v)\widehat{\ell}_t(i,v) &= \sum_{i\in A} p_t(i,v)\sum_{s=0}^{d-1} D(s)\,\frac{\ell_{t-s}(i,v)}{q_{s,t-s}(i,v)}B_{s,t-s}(i,v) \\ &\leq \sum_{i\in A} p_t(i,v)\sum_{s=0}^{d-1} D(s)\,\frac{1}{q_{s,t-s}(i,v)} \\ &\leq \sum_{i\in A}\sum_{s=0}^{d-1}\left(1+\frac{1}{d}\right)^s D(s)\,\frac{p_{t-s}(i,v)}{q_{s,t-s}(i,v)} \\ &\leq \sum_{i\in A}\sum_{s=0}^{d-1}\left(1+\frac{1}{d}\right)^s D(s) \\ &\leq Ke\,. \end{aligned}$$

Combining with Lemma 1 as in the proof of Lemma 2 concludes the proof. ■

Lemma 20 *Under the same assumptions and notation as in Lemma 19, we have that*

$$p_{t+1}(i,v) - p_t(i,v) \geq -e\eta\sum_{s=0}^{d-1} D(s)\,\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)}{q_{s,t-s}(i,v)}$$

holds for all $t \geq 1$, $i \in A$, and $v \in V$ deterministically with respect to the agent's randomization.

Proof When $t \leq d$ the statement is trivially verified, hence we continue by assuming $t > d$. From Lemma 1 we have $p_{t+1}(i,v) - p_t(i,v) \geq -\eta\,p_t(i,v)\widehat{\ell}_t(i,v)$. Then, proceeding as in the proof of Lemma 19, we have

$$\begin{aligned} p_t(i,v)\widehat{\ell}_t(i,v) &= p_t(i,v)\sum_{s=0}^{d-1} D(s)\,\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)} \\ &\leq \sum_{s=0}^{d-1} D(s)\,\frac{B_{s,t-s}(i,v)\,p_t(i,v)}{q_{s,t-s}(i,v)} \\ &\leq \sum_{s=0}^{d-1} D(s)\,\left(1+\frac{1}{d}\right)^s\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)}{q_{s,t-s}(i,v)} \\ &\leq e\sum_{s=0}^{d-1} D(s)\,\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)}{q_{s,t-s}(i,v)}\,. \end{aligned}$$

Putting together proves the claim. ■

Proof of Theorem 13. As in the proof of Theorem 4, we start off from (6). Using the definition of $\widehat{\ell}_t(i,v)$ in (8) this can be written as

$$\begin{aligned} \sum_{t=d+1}^{T}\sum_{i=1}^{K}\sum_{s=0}^{d-1} D(s)\,\frac{B_{s,t-s}(i,v)\,p_t(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)} &\leq \sum_{t=d+1}^{T}\sum_{s=0}^{d-1} D(s)\,\frac{B_{s,t-s}(k,v)\,\ell_{t-s}(k)}{q_{s,t-s}(k,v)} \\ &\quad + \frac{\ln K}{\eta} + \frac{\eta}{2}\sum_{t=d+1}^{T}\sum_{i=1}^{K} p_t(i,v)\left(\sum_{s=0}^{d-1} D(s)\,\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)}\right)^2\,. \quad (14) \end{aligned}$$

We now bound the first and the third sum in (14) separately. As for the first sum, a repeated application of Lemma 20 for $t \geq 2d+1$ and $s = 1,\ldots,d$ leads to

$$p_t(i,v) \geq p_{t-s}(i,v) - e\eta\sum_{h=1}^{s}\sum_{r=0}^{d-1} D(r)\,\frac{B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)}\,,$$

so that

$$\sum_{t=d+1}^{T}\sum_{i=1}^{K}\sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(i,v)\,p_t(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)} \geq \sum_{t=2d+1}^{T}\sum_{i=1}^{K}\sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(i,v)\,p_t(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)} \geq \sum_{t=2d+1}^{T}\sum_{i=1}^{K}\sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)} - e\eta\sum_{t=2d+1}^{T}\sum_{i=1}^{K} S_t(i,v)\,,$$

where the slack $S_t(i,v)$ satisfies

$$S_t(i,v) = \sum_{s=1}^{d-1} D(s)\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)}\sum_{h=1}^{s}\sum_{r=0}^{d-1} D(r)\frac{B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)}\,. \tag{15}$$

As for the third sum, Jensen's inequality and $\ell_{i,t-s}\in[0,1]$ jointly give

$$\left(\sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)}\right)^2 \leq \sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(i,v)}{q_{s,t-s}(i,v)^2}\,.$$

Moreover, recalling that $s\in\{1,\dots,d\}$, a repeated application of Lemma 19 yields

$$p_t(i,v) \leq \left(1+\frac{1}{d}\right)^s p_{t-s}(i,v) \leq \left(1+\frac{1}{d}\right)^d p_{t-s}(i,v) \leq e\,p_{t-s}(i,v)\;,$$

so that

$$\sum_{t=d+1}^{T}\sum_{i=1}^{K} p_t(i,v)\left(\sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)}\right)^2 \leq e\sum_{t=d+1}^{T}\sum_{i=1}^{K}\sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)}{q_{s,t-s}(i,v)^2}\;.$$

Putting together and summing over $v\in V$ we obtain

$$\underbrace{\sum_{t=2d+1}^{T}\sum_{i=1}^{K}\sum_{v\in V}\sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)}}_{\text{(I)}} \leq \underbrace{\sum_{t=d+1}^{T}\sum_{v\in V}\sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(k,v)\,\ell_{t-s}(k)}{q_{s,t-s}(k,v)}}_{\text{(II)}} + \frac{N\ln K}{\eta} + e\eta\underbrace{\sum_{t=2d+1}^{T}\sum_{i=1}^{K}\sum_{v\in V} S_t(i,v)}_{\text{(III)}} + \frac{e\eta}{2}\underbrace{\sum_{t=d+1}^{T}\sum_{i=1}^{K}\sum_{v\in V}\sum_{s=0}^{d-1} D(s)\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)}{q_{s,t-s}(i,v)^2}}_{\text{(IV)}}\,.$$

Similar to before, we have, for any $i\in A$, $v\in V$, $t=1,2,\dots$, and $s=0,\dots,d-1$,

$$\mathbb{E}_{t-s}\left[\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)}\right] = \frac{p_{t-s}(i,v)\,\ell_{t-s}(i)}{q_{s,t-s}(i,v)}\,\mathbb{E}_{t-s}\big[B_{s,t-s}(i,v)\big] = p_{t-s}(i,v)\,\ell_{t-s}(i)$$

$$\mathbb{E}_{t-s}\left[\frac{B_{s,t-s}(k,v)\,\ell_{t-s}(k)}{q_{s,t-s}(k,v)}\right] = \frac{\ell_{t-s}(k)}{q_{s,t-s}(k,v)}\,\mathbb{E}_{t-s}\big[B_{s,t-s}(k,v)\big] = \ell_{t-s}(k)$$

$$\mathbb{E}_{t-s}\left[\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)}{\big(q_{s,t-s}(i,v)\big)^2}\right] = \frac{p_{t-s}(i,v)}{\big(q_{s,t-s}(i,v)\big)^2}\,\mathbb{E}_{t-s}\big[B_{s,t-s}(i,v)\big] = \frac{p_{t-s}(i,v)}{q_{s,t-s}(i,v)}\,.$$

Hence

$$\begin{aligned}\mathbb{E}\left[(\mathrm{I})\right] &= \sum_{i=1}^{K}\sum_{v\in V}\sum_{t=2d+1}^{T}\sum_{s=0}^{d-1} D(s)p_{t-s}(i,v)\,\ell_{t-s}(i)\\ &\geq \sum_{i=1}^{K}\sum_{v\in V}\sum_{t=2d+1}^{T-d+1} p_t(i,v)\,\ell_t(i)\\ &\geq \sum_{i=1}^{K}\sum_{v\in V}\left(\sum_{t=1}^{T} p_t(i,v)\,\ell_t(i) - \sum_{t=1}^{2d} p_t(i,v) - \sum_{t=T-d+2}^{T} p_t(i,v)\right)\\ &\geq \sum_{v\in V}\sum_{t=1}^{T}\sum_{i=1}^{K} p_t(i,v)\,\ell_t(i) - 3dN\end{aligned}$$

and

$$\mathbb{E}\left[(\mathrm{II})\right] = \sum_{v\in V}\sum_{t=2d+1}^{T}\sum_{s=0}^{d-1} D(s)\ell_{t-s}(k) \leq \sum_{v\in V}\sum_{t=1}^{T}\ell_t(k)\,.$$

Moreover, from Lemma 3,

$$\begin{aligned}\mathbb{E}\left[(\mathrm{IV})\right] &= \sum_{i=1}^{K}\sum_{t=d+1}^{T}\sum_{s=0}^{d-1} D(s)\sum_{v\in V}\frac{p_{t-s}(i,v)}{q_{s,t-s}(i,v)}\\ &\leq \sum_{i=1}^{K}\sum_{t=d+1}^{T}\sum_{s=0}^{d-1} D(s)\left(\frac{1}{1-e^{-1}}\left(\alpha(G_{\leq s}) + \sum_{v\in V} p_{t-s}(i,v)\right)\right)\\ &= \frac{K}{1-e^{-1}}\sum_{t=d+1}^{T}\sum_{s=0}^{d-1} D(s)\alpha(G_{\leq s}) + \frac{1}{1-e^{-1}}\sum_{t=d+1}^{T}\sum_{s=0}^{d-1} D(s)\sum_{v\in V}\sum_{i=1}^{K} p_{t-s}(i,v)\\ &= \frac{K}{1-e^{-1}}\,(T-d)\,\bar{\alpha}_D + \frac{1}{1-e^{-1}}\,(TN - dN)\\ &< \frac{T}{1-e^{-1}}\,(K\bar{\alpha}_D + N)\;,\end{aligned}$$

where $\bar{\alpha}_D = \sum_{s=0}^{d-1} D(s)\alpha(G_{\leq s})$.

Finally, we are left with upper bounding $\mathbb{E}[(\mathrm{III})]$. Consider each slack term $S_t(i,v)$ in (15) We can then rewrite $S_t(i,v)$ as

$$S_t(i,v) = \sum_{s=1}^{d-1}\sum_{h=1}^{s}\sum_{r=0}^{d-1} D(s)D(r)\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)\,B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\;.$$

Now, we consider three cases, depending on the value of the indices s,h, and r in the triple sum.

Case 1: $t-s > t-h-r$ (i.e., $s < h+r$). We have

$$\begin{aligned}
&\mathbb{E}\left[\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)\,B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\right] \\
&= \mathbb{E}\left[\mathbb{E}_{t-s}\left[\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)\,B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\right]\right] \\
&= \mathbb{E}\left[\frac{\ell_{t-s}(i)\,B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\mathbb{E}_{t-s}\left[B_{s,t-s}(i,v)\right]\right] \\
&= \mathbb{E}\left[\frac{\ell_{t-s}(i)\,B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)}\right] \\
&\le \mathbb{E}\left[\frac{p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)}\,E_{t-h-r}[B_{r,t-h-r}(i,v)]\right] \\
&= \mathbb{E}\left[p_{t-h-r}(i,v)\right]\ .
\end{aligned}$$

Case 2: $t-s < t-h-r$ (i.e., $s > h+r$). We can write

$$\begin{aligned}
&\mathbb{E}\left[\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)\,B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\right] \\
&= \mathbb{E}\left[\mathbb{E}_{t-h-r}\left[\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)\,B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\right]\right] \\
&= \mathbb{E}\left[\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\mathbb{E}_{t-h-r}\left[B_{r,t-h-r}(i,v)\,\right]\right] \\
&\le \mathbb{E}\left[\frac{B_{s,t-s}(i,v)\,p_{t-h-r}(i,v)}{q_{s,t-s}(i,v)}\right] \\
&\le \left(1+\frac{1}{d}\right)^{s-h-r}\mathbb{E}\left[\frac{B_{s,t-s}(i,v)\,p_{t-s}(i,v)}{q_{s,t-s}(i,v)}\right] \qquad \text{(repeatedly using Lemma 19)} \\
&\le e\,\mathbb{E}\left[\frac{p_{t-s}(i,v)}{q_{s,t-s}(i,v)}\,\mathbb{E}_{t-s}\left[B_{s,t-s}(i,v)\right]\right] \\
&= e\,\mathbb{E}\left[p_{t-s}(i,v)\right]\ .
\end{aligned}$$

Case 3: $t-s = t-h-r$ (i.e., $s = h+r$). We have $p_{t-h-r}(i,v) = p_{t-s}(i,v)$. Moreover, since $h \ge 1$ we have $s > r$, so that $B_{s,t-s}(i,v)\,B_{r,t-h-r}(i,v) = B_{r,t-h-r}(i,v)$. We can write

$$\begin{aligned}
&\mathbb{E}\left[\frac{B_{s,t-s}(i,v)\,\ell_{t-s}(i)\,B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\right] \\
&= \mathbb{E}\left[\mathbb{E}_{t-h-r}\left[\frac{\ell_{t-h-r}(i)\,B_{r,t-h-r}(i,v)\,p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\right]\right] \\
&\le \mathbb{E}\left[\frac{p_{t-h-r}(i,v)}{q_{r,t-h-r}(i,v)\,q_{s,t-s}(i,v)}\,\mathbb{E}_{t-h-r}\left[B_{r,t-h-r}(i,v)\right]\right] \\
&= \mathbb{E}\left[\frac{p_{t-h-r}(i,v)}{q_{s,t-s}(i,v)}\right] \\
&= \mathbb{E}\left[\frac{p_{t-s}(i,v)}{q_{s,t-s}(i,v)}\right]\ .
\end{aligned}$$

Therefore, piecing together and overapproximating yields

$$\begin{aligned}
\mathbb{E}\left[S_t(i,v)\right] \le e\,\mathbb{E}\Bigg[&\sum_{s,h,r\,:\,s<h+r} D(s)D(r)\,p_{t-h-r}(i,v) + \sum_{s,h,r\,:\,s>h+r} D(s)D(r)\,p_{t-s}(i,v) \\
&+ \sum_{s,h,r\,:\,s=h+r} D(s)D(r)\,\frac{p_{t-s}(i,v)}{q_{s,t-s}(i,v)}\Bigg]\ .
\end{aligned}$$

This allows us to write

$$\begin{aligned}
\mathbb{E}[(\mathrm{III})] &\le e\sum_{t=2d+1}^{T}\sum_{i=1}^{K}\sum_{v\in V}\mathbb{E}\Bigg[\sum_{s,h,r\,:\,s<h+r} D(s)D(r)\,p_{t-h-r}(i,v) \\
&\qquad + \sum_{s,h,r\,:\,s>h+r} D(s)D(r)\,p_{t-s}(i,v) + \sum_{s,h,r\,:\,s=h+r} D(s)D(r)\,\frac{p_{t-s}(i,v)}{q_{s,t-s}(i,v)}\Bigg] \\
&= e\sum_{t=2d+1}^{T}\sum_{v\in V}\sum_{s,h,r\,:\,s<h+r} D(s)D(r) + e\sum_{t=2d+1}^{T}\sum_{v\in V}\sum_{s,h,r\,:\,s>h+r} D(s)D(r) \\
&\qquad + e\sum_{t=2d+1}^{T}\sum_{i=1}^{K}\sum_{s,h,r\,:\,s=h+r} D(s)D(r)\,\mathbb{E}\left[\sum_{v\in V}\frac{p_{t-s}(i,v)}{q_{s,t-s}(i,v)}\right] \\
&\le e\,T\,N\sum_{s,h,r\,:\,s\neq h+r} D(s)D(r) \\
&\qquad + \frac{e}{1-e^{-1}}\sum_{t=2d+1}^{T}\sum_{i=1}^{K}\sum_{s,h,r\,:\,s=h+r} D(s)D(r)\left(\alpha(G_{\le s}) + \sum_{v\in V} p_{t-s}(i,v)\right) \\
&= e\,T\,N\sum_{s,h,r\,:\,s\neq h+r} D(s)D(r) \\
&\qquad + \frac{T\,e}{1-e^{-1}}\left(K\sum_{s,h,r\,:\,s=h+r} D(s)D(r)\alpha(G_{\le s}) + N\sum_{s,h,r\,:\,s=h+r} D(s)D(r)\right) \\
&< \frac{T\,e}{1-e^{-1}}\left(K\sum_{s,h,r\,:\,s=h+r} D(s)D(r)\alpha(G_{\le s}) + N\sum_{s,h,r} D(s)D(r)\right)
\end{aligned}$$

Now,

$$\sum_{s,h,r} D(s)D(r) = \sum_{s=1}^{d-1} D(s)\sum_{h=1}^{s}\sum_{r=0}^{d-1} D(r) = \sum_{s=1}^{d-1} D(s)\,s = \mu_D$$

and

$$\sum_{s,h,r\,:\,s=h+r} D(s)D(r)\alpha(G_{\le s}) = \sum_{s=1}^{d-1} D(s)\alpha(G_{\le s})\sum_{h=1}^{s} D(s-h) \le \sum_{s=1}^{d-1} D(s)\alpha(G_{\le s}) = \bar{\alpha}_D$$

so that

$$\mathbb{E}[(\mathrm{III})] \leq \frac{T\,e}{1-e^{-1}}\left(N\mu_D + K\bar{\alpha}_D\right).$$

Putting pieces together, dividing by N, and further overapproximating gives

$$R_T^{\mathrm{coop}} \leq 3d + \frac{\ln K}{\eta} + \frac{e^2\eta\,T}{1-e^{-1}}\left(1+\mu_D+\frac{2K}{N}\,\bar{\alpha}_D\right),$$

thereby concluding the proof. ■

References

Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. In J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 873–881. Curran Associates, Inc., 2011.

Noga Alon, Nicolò Cesa-Bianchi, Claudio Gentile, Shie Mannor, Yishay Mansour, and Ohad Shamir. Nonstochastic multi-armed bandits with graph-structured feedback. *SIAM Journal of Computing*, 46(6):1785–1826, 2017.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002a.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002b.

Baruch Awerbuch and Robert Kleinberg. Competitive collaborative learning. *Journal of Computer and System Sciences*, 74(8):1271–1288, 2008.

Samuel Barrett and Peter Stone. Ad hoc teamwork modeled with multi-armed bandits: An extension to discounted infinite rewards. In *Proceedings of 2011 AAMAS Workshop on Adaptive and Learning Agents*, pages 9–14, 2011.

Nicolò Cesa-Bianchi, Claudio Gentile, Yishay Mansour, and Alberto Minora. Delay and cooperation in nonstochastic bandits. In *Conference on Learning Theory*, pages 605–622, 2016.

John Duchi, Michael I Jordan, and Brendan McMahan. Estimation, optimization, and parallelism when data is sparse. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2832–2840. Curran Associates, Inc., 2013.

John C Duchi, Sorathan Chaturapruek, and Christopher Ré. Asynchronous stochastic convex optimization. *arXiv preprint arXiv:1508.00882*, 2015.

Miroslav Dudík, Daniel J. Hsu, Satyen Kale, Nikos Karampatziakis, John Langford, Lev Reyzin, and Tong Zhang. Efficient optimal learning for contextual bandits. In *UAI 2011, Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence, Barcelona, Spain, July 14-17, 2011*, pages 169–178, 2011.

P. Firby and J. Haviland. Independence and average distance in graphs. *Discrete Applied Mathematics*, 75:27–37, 1997.

Pooria Joulani, András György, and Csaba Szepesvári. Online learning under delayed feedback. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1453–1461, 2013.

Pooria Joulani, András György, and Csaba Szepesvári. Delay-tolerant online convex optimization: Unified analysis and adaptive-gradient algorithms. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 1744–1750, 2016.

Soummya Kar, H Vincent Poor, and Shuguang Cui. Bandit problems in networks: Asymptotically efficient distributed allocation rules. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 1771–1778. IEEE, 2011.

Robert Kleinberg, Georgios Piliouras, and Éva Tardos. Multiplicative updates outperform generic no-regret learning in congestion games. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 533–542. ACM, 2009.

Tomáš Kocák, Gergely Neu, Michal Valko, and Remi Munos. Efficient learning by implicit exploration in bandit problems with side observations. In *Advances in Neural Information Processing Systems 27*, pages 613–621. 2014.

Ravi Kumar Kolla, Krishna Jagannathan, and Aditya Gopalan. Collaborative learning of stochastic bandits over a social network. In *54th Annual Allerton Conference on Communication, Control, and Computing*, pages 1228–1235. IEEE, 2016.

Peter Landgren, Vaibhav Srivastava, and Naomi Ehrich Leonard. On distributed cooperative decision-making in multiarmed bandits. *arXiv preprint arXiv:1512.06888*, 2015.

Mu Li, David G. Andersen, and Alexander Smola. Distributed delayed proximal gradient methods. In *NIPS Workshop on Optimization for Machine Learning*, 2013.

Nathan Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1):193–201, 1992.

Ji Liu, Stephen J Wright, Christopher Ré, Victor Bittorf, and Srikrishna Sridhar. An asynchronous parallel stochastic coordinate descent algorithm. *The Journal of Machine Learning Research*, 16(1):285–322, 2015.

Horia Mania, Xinghao Pan, Dimitris Papailiopoulos, Benjamin Recht, Kannan Ramchandran, and Michael I Jordan. Perturbed iterate analysis for asynchronous stochastic optimization. *arXiv preprint arXiv:1507.06970*, 2015.

Brendan McMahan and Matthew Streeter. Delay-tolerant algorithms for asynchronous distributed online learning. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2915–2923. Curran Associates, Inc., 2014.

Chris Mesterharm. On-line learning with delayed label feedback. In *Algorithmic Learning Theory*, pages 399–413. Springer, 2005.

Chris Mesterharm. *Improving Online Learning*. PhD thesis, Department of Computer Science, Rutgers University, 2007.

Gergely Neu. Explore no more: Improved high-probability regret bounds for non-stochastic bandits. In *Advances in Neural Information Processing Systems 28 (NIPS)*, 2015.

Gergely Neu, Andras Antos, András György, and Csaba Szepesvári. Online Markov decision processes under bandit feedback. In *Advances in Neural Information Processing Systems 23*, pages 1804–1812. Curran Associates, Inc., 2010.

Gergely Neu, Andras Gyorgy, Csaba Szepesvari, and Andras Antos. Online markov decision processes under bandit feedback. *Automatic Control, IEEE Transactions on*, 59(3):676–691, 2014.

Xinghao Pan, Dimitris Papailiopoulos, Samet Oymak, Benjamin Recht, Kannan Ramchandran, and Michael I Jordan. Parallel correlation clustering on big graphs. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 82–90. Curran Associates, Inc., 2015.

Ciara Pike-Burke, Shipra Agrawal, Csaba Szepesvari, and Steffen Grunewalder. Bandits with delayed, aggregated anonymous feedback. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 4102–4110, 2018.

Kent Quanrud and Daniel Khashabi. Online learning with adversarial delays. In C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1270–1278. Curran Associates, Inc., 2015.

Jonathan Rosenski, Ohad Shamir, and Liran Szlak. Multi-player bandits – a musical chairs approach. *CoRR*, abs/1512.02866, 2015.

Yevgeny Seldin, Peter Bartlett, Koby Crammer, and Yasin Abbasi-Yadkori. Prediction with limited advice and multiarmed bandits with paid observations. In *Proceedings of The 31st International Conference on Machine Learning*, pages 280–287, 2014.

Ruben Stranders, Long Tran-Thanh, Francesco M Delle Fave, Alex Rogers, and Nicholas R Jennings. Dcops and bandits: Exploration and exploitation in decentralised coordination. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 289–296. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

Jukka Suomela. Survey of local algorithms. *ACM Computing Surveys*, 45(2):24, 2013.

Balazs Szorenyi, Róbert Busa-Fekete, István Hegedüs, Róbert Ormándi, Márk Jelasity, and Balázs Kégl. Gossip-based distributed stochastic bandit algorithms. In *30th International Conference on Machine Learning (ICML 2013)*, volume 28, pages 19–27. ACM Press, 2013.

Cem Tekin and Mihaela van der Schaar. Distributed online learning via cooperative contextual bandits. *IEEE Transactions on Signal Processing*, 63(14):3700–3714, 2015.

Cem Tekin, Simpson Z. Zhang, and Mihaela van der Schaar. Distributed online learning in social recommender systems. *J. Sel. Topics Signal Processing*, 8(4):638–652, 2014.

Klaus Wehmuth and Artur Ziviani. Daccer: Distributed assessment of the closeness centrality ranking in complex networks. *Computer Networks*, 57(13):2536–2548, 2013.

Marcelo J Weinberger and Erik Ordentlich. On delayed prediction of individual sequences. *IEEE Transactions on Information Theory*, 48(7):1959–1976, 2002.

Martin Zinkevich, John Langford, and Alex J. Smola. Slow learners are fast. In Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2331–2339. Curran Associates, Inc., 2009.

Journal of Machine Learning Research 20 (2019) 1-29 Submitted 11/17; Published 2/19

Multiplicative local linear hazard estimation and best one-sided cross-validation

María Luz Gámiz MGAMIZ@UGR.ES
Department of Statistics and Operations Research, University of Granada
Campus Fuentenueva s/n, 18071 Granada, Spain

María Dolores Martínez-Miranda MMIRANDA@UGR.ES
Department of Statistics and Operations Research, University of Granada
Campus Fuentenueva s/n, 18071 Granada, Spain

Jens Perch Nielsen JENS.NIELSEN.1@CITY.AC.UK
Cass Business School, City, University of London
106 Bunhill Row, London EC1Y8TZ, U.K.

Editor: Sara van de Geer

Abstract

This paper develops detailed mathematical statistical theory of a new class of cross-validation techniques of local linear kernel hazards and their multiplicative bias corrections. The new class of cross-validation combines principles of local information and recent advances in indirect cross-validation. A few applications of cross-validating multiplicative kernel hazard estimation do exist in the literature. However, detailed mathematical statistical theory and small sample performance are introduced via this paper and further upgraded to our new class of best one-sided cross-validation. Best one-sided cross-validation turns out to have excellent performance in its practical illustrations, in its small sample performance and in its mathematical statistical theoretical performance.

Keywords: Aalen's multiplicative model, multiplicative bias correction, bandwidth, indirect cross-validation

1. Introduction

There is a growing interest in validation techniques. While validation was always a crucial element of mathematical statistics, the use of validation techniques are growing rapidly at the moment under labels such as big data, machine learning or artificial intelligence. Many of these developments seem less patient with laborious mathematical statistical model formulation and estimation theory than what has been the trademark of the field of mathematical statistics. Instead inspiration seems to be taken from neighbouring fields such as engineering, computer science, public health or actuarial science, where specific knowledge is present on the problem at hand, allowing the development of clever and perhaps computationally challenging algorithms often replacing more labour intensive procedures of the past. These algorithms are often defined in such a way that they can change and learn over time via some optimization criteria and an efficient validation procedure. One example of such work relevant to the work of this paper is Muñoz and van der Laan (2012) where an impressive algorithm is developed to solve a complicated survival problem. The introduced

©2019 María Luz Gámiz, María Dolores Martínez-Miranda, and Jens Perch Nielsen .

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-663.html.

methodology is inspired by machine learning: the validation procedure is called a Super Learner. However, while the Super Learner is optimal in some sense, see van der Laan et al. (2007), then it is not optimal in the more detailed mathematical statistical sense that we consider in this paper. And this is not only because Muñoz and van der Laan (2012) consider piecewise constant hazard models that are less efficient than kernel smoothers. It is also because the validation theory provided by Muñoz and van der Laan (2012) does not provide the mathematical detail promoted in this paper, and therefore crucial insight of noisy second order components is not included in the theory. The approach of Muñoz and van der Laan (2012) is just one among many machine learning inspired survival analyses approaches. This paper considers one dimension only. Multidimensional cross-validation and one-dimensional cross-validation are closely related and mathematical definitions are similar. However, even in the one-dimensional case we face challenging theoretical, as well as practical issues, with cross-validation being too noisy and unstable and to such an extend that we cannot any longer recommend cross-validation in one dimension without some amendment for the noise involved. Our intention is that multidimensional big data type of problems, with further issues with data sparsity and noisy cross-validation, should benefit in the future from the insight on cross-validation analyses as provided in this paper. The mathematical point of view of this paper was perhaps initiated via the early contribution of Hall and Marron (1987) that provided a decision theoretical framework to distinguish between plug-in estimators, aiming at minimizing a mean integrated squared error, and cross-validation aiming at minimizing the infeasible stochastic integrated squared error. They concluded that plug-in did better from an asymptotic perspective even when the aim was the explicit aim of cross-validation: to get as close as possible to the infeasible minimization of the integrated squared error. One could view this as the foundation of a new decision theoretical framework to understand the quality of kernel bandwidth selection; a tractable place to start when understanding the complicated world of model selection. Hall and Johnstone (1992) pointed out that for any bandwidth selector there are two sources of noise for kernel density estimation, one that one can never get rid of and another one that seems to differ for different methods. The second source of noise can theoretically go as low as to zero such that one was left with the first noise component as a lower bound on noise. The plug-in type of methods had considerable lower second-component noise than cross-validation and plug-in was very popular in practice in the nineties, with Sheather and Jones (1991) being the perhaps most popular single method. However, plug-in methods depend on complicated underlying mathematical detail that does not easily generalize to new problems in the same straightforward way as cross-validation does. This is perhaps the single most important reason why cross-validation has regained its importance and is used for a wide variety of complicated problems in mathematical statistics, big data, machine learning and artificial intelligence. Hart and Yi (1998) introduced the concept of indirect cross-validation, formulated in nonparametric kernel regression, which simply meant that cross-validation was performed on an alternative kernel and the bandwidth was scaled back to the original kernel used for estimation. Hart and Yi (1998) suggested to use one-sided kernels as the alternative kernels because of their good practical performance and simple rescaling. In density estimation Savchuk et al. (2010) suggested a combination of a normal-bandwidth kernel and an oversmoothed kernel as alternative kernel to achieve the same mathematical statistical asymptotic performance as the plug-in estimator without the need

of a pilot. However, there was one catch with the elegant approach of Savchuk et al. (2010). Their approach needed to estimate some tuning parameters to decide the relative weight of the oversmoothed kernel that was contributing to the asymptotic noise via some term of lower order. So, even though Savchuk et al. (2010) in principle did pilot free estimation then there was still some tuning going on and some extra terms of just slightly lower order. And that was perhaps exactly the problem of the original plug-in methods as in Sheather and Jones (1991): that something with lower order noise had to be estimated, the pilot, and terms of slightly lower order had to be ignored in the asymptotic results. In this paper we define three dogmas for a modern kernel smoothing estimator:

1. It should be a direct estimation based on principles without complicated mathematical adjustments.

2. Extra terms of slightly lower order are not allowed in the expansions.

3. Further smoothing than those necessary for the original estimator is not allowed to be assumed while analysing the quality of the bandwidth selector.

The original cross-validation estimator and the approach of Hart and Yi (1998) lives up to all three dogma rules while the plug-in type estimators of e.g. Sheather and Jones (1991) and Savchuk et al. (2010) violate all three. We believe this to be the reason why Mammen et al. (2011,2014) concluded that their double one-sided kernel density bandwidth selector, directly inspired by Hart and Yi (1998), worked better in practice than the estimators of Sheather and Jones (1991) and Savchuk et al. (2010). The fundamental principles of this paper are therefore the three dogmas above and the decision theoretical framework of Hall and Marron (1987), and this has let us to explore double one-sided cross-validation and one-sided cross-validation even further because of their apparent practical superiority on the market of current kernel bandwidth selectors. A detailed investigation of both sides of local one-sided bandwidth selection showed us a perhaps surprising fact. While the left-sided and the right-sided cross-validation procedures have the same mathematical statistical behaviour, they do perform very differently in practice. Often one of the two sides breaks down completely. Double one-sided cross-validation works better than one-sided cross-validation in a wide variety of kernel smoothing problems, see for example Mammen et al. (2011,2014), Gámiz et al. (2013a,b, 2016). A closer investigation going through local features of individual simulation samples reveals that behind a good double one-sided cross-validation result often hides an average of a good one-sided estimator and a somehow random result from the other side. The suggestion of this paper is to improve the stability of one-sided cross-validation via a local information principle inspecting at every single local point whether to use the right side or the left side for cross-validation. This approach is stable in its practical performance, it obeys the three above dogmas and it provides the exact same asymptotic performance as the less stable one-sided and double-sided competitors mentioned above. We call the new approach best one-sided cross-validation. This paper furthermore introduces the mathematical statistical approach of Hall and Marron (1987) to multiplicative bias corrected local linear kernel hazard estimators and it introduces asymptotic theory and practical implementation of best one-sided-cross-validation for these multiplicative bias corrected hazard estimators. Multiplicative bias correction is known to improve the practical implementation of kernel hazard estimation, see Nielsen

(1998) and Nielsen and Tanggaard (2001). This parallels insights from the more researched world of kernel density estimation, see for example Jones et al. (1995) and Jones and Signorini (1997). The latter went through a series of small sample studies of kernel density estimation procedures to conclude that multiplicative bias correction seemed to be the best. The contribution of this paper is therefore also to update mathematical statistical theory and practice to the perhaps best practically performing kernel hazard estimator we have: the multiplicative bias corrected local linear kernel hazard estimator.

The rest of the paper is organized as follows. In Section 2 we describe the link between our proposal and methods in machine learning. In Section 3 we formulate the model we assume in the paper and present two hazard estimators, namely the local linear estimator and its multiplicative bias correction. Bandwidth selection for these estimators through cross-validation and the double one-sided cross-validation method of Gámiz et al. (2016) is described in Section 4, and our new best one-sided cross-validation method is suggested. The asymptotic properties of all presented validated bandwidths are analysed in Section 5. Assumptions are deferred to Appendix A and proofs are provided in the Supplementary Material. A further investigation of the theoretical properties of bandwidth selectors is described in Appendix B. Two case studies show the applicability of our proposals, which are described in Section 6. In Section 7 we describe simulation experiments to evaluate the finite sample properties of our proposal. The main findings from the simulations are discussed in Section 8 along with further insights about the asymptotic properties of bandwidth selectors. Final conclusions are drawn in Section 9. All numerical calculations have been performed with `R` and the methods proposed in this paper have been implemented in the `DOvalidation` package (Gámiz et al., 2017).

2. Training and learning versus cross-validation and adjusted cross-validation

To motivate our research beyond a wider crowd than experts in nonparametric hazard estimation, our point of view is formulated below via standard vocabulary from machine learning and artificial intelligence. Let us assume we observe n individuals over some time that could potentially be filtered via truncation and censoring, and let A be a training set and B be a learning set such that the two sets united equals the set $\{1, \ldots, n\}$. Let for the purpose of a discussion the number of elements of A be 80% of n and the number of elements in B be 20% of n. Then a standard approach to validation, see again Muñoz and van der Laan (2012), would be to estimate the hazard on the training set and evaluate it via the learning set. Under some standard independence assumptions this will lead to a decrease in efficiency of estimation itself corresponding to ignoring 20% of the data set and it will decrease the efficiency on the validation approach, compared to cross-validation and the theoretical approach considered in this paper, corresponding to ignoring 80% of the data set. One could of course consider all possible combinations of training and learning sets and average all these validations into one single validation principle or learning principle. This would correspond to a computationally inefficient cross-validation. In conclusion: even if all possible combinations of trainers and learners are calculated, we end up with standard cross-validation with the well known problems of data sparsity and noise. With the help of the theory originally developed by Hall and Marron (1987) in the kernel density context,

we will in this paper, in the kernel hazard context, consider more efficient use of data when estimators are validated or when trainers are learning. It turns out that this is indeed possible via relatively straightforward adjustments of standard cross-validation.

3. The counting process model and kernel hazard estimators

In this section we formulate events via counting processes. Counting processes are well designed when event data are filtered for example via truncation or censoring. An individual zero-one valued exposure process simply keeps tracks on whether an individual is under risk or not at any particular point in time. We assume that individuals are independent and that data filtering is non-informative. Formally, we observe n individuals, $i = 1, \ldots, n$. Let N_i count observed failures for the ith individual in the time interval $[0, T]$, N_i can take values 0 or 1. We assume that N_i is a one-dimensional counting process with respect to an increasing, right continuous, complete filtration $\mathcal{F}_t$, $t \in [0, T]$, i.e., it obeys the usual conditions (Andersen et al., 1993, p. 60). We assume Aalen's multiplicative model (Aalen, 1978) where the random intensity is written as $\lambda_i(t) = \alpha(t)Y_i(t)$, with no restriction on the functional form of the hazard function $\alpha(\cdot)$. Here Y_i is a predictable process taking values 0 or 1, indicating (by the value 1) when the ith individual is at risk and under observation. We assume that $(N_1, Y_1), \ldots, (N_n, Y_n)$ are independent and identically distributed for the n individuals. With these definitions λ_i is predictable and the processes $M_i(t) = N_i(t) - \Lambda_i(t)$, $i = 1, \ldots, n$, with $\Lambda_i(t) = \int_0^t \lambda_i(s)\, ds$, are square integrable local martingales.

As an example we illustrate how the above stochastic processes look like in the case of independent and non-informative left truncation and right censoring, where n tuples (L_i, Z_i, δ_i), $i = 1, \ldots, n$, are observed. Here L_i is the time the ith individual enters the study; Z_i is the time ith individual leaves the study either because an event has happened or because of right censoring; and δ_i is binary and equal to one if an event, for example death or an onset of a disease, is the reason for the ith individual to leave the study, and the value is zero when the reason for the ith individual to leave the study was uninformative right censoring. In this case, the process Y_i above would be $Y_i(t) = I(L_i \leq t < Z_i)$, and $N_i(t) = I(Z_i \leq t)\delta_i$, where $I(\cdot)$ is the indicator function. Hereafter we will work in the convenient and general stochastic process formulation only.

The local linear kernel hazard estimator in our general stochastic process formulation was introduced by Nielsen and Tanggaard (2001) and it is defined as

$$\widehat{\alpha}^{\mathrm{LL}}_{b,K}(t) = \sum_{i=1}^{n} \int_0^T \bar{K}_{t,b}(t-s) dN_i(s), \tag{1}$$

with the stochastic local linear kernel

$$\bar{K}_{t,b}(t-s) = \frac{a_{2,K}(t) - a_{1,K}(t)(t-s)}{a_{0,K}(t)a_{2,K}(t) - \{a_{1,K}(t)\}^2} K_b(t-s), \tag{2}$$

where $K_b(u) = b^{-1}K(u/b)$, $a_{j,K}(t) = \int_0^T K_b(t-s)(t-s)^j Y(s)ds$, for $j = 0, 1, 2$, and $Y(t) = \sum_{i=1}^n Y_i(t)$ is the aggregated risk process. Here K is a kernel function with support $[-1, 1]$ and $b > 0$ is the bandwidth parameter.

The local linear kernel $\bar{K}_{t,b}$ satisfies the properties: $\int_0^T \bar{K}_{t,b}(t-s)Y(s)ds = 1$, $\int_0^T \bar{K}_{t,b}(t-s)(t-s)Y(s)ds = 0$ and $\int_0^T \bar{K}_{t,b}(t-s)(t-s)^2 Y(s)ds > 0$. Thus, $\bar{K}_{t,b}$ can be interpreted as a

second order kernel with respect to the stochastic measure μ, where $d\mu(s) = Y(s)ds$. Defining the aggregated failure process, $N(t) = \sum_{i=1}^n N_i(t)$, we can write $\widehat{\alpha}^{\mathrm{LL}}_{b,K}(t) = \int_0^T \bar{K}_{t,b}(t-s)dN(s)$.

The multiplicative bias corrected estimator constructed from the local linear hazard estimator is defined as

$$\widehat{\alpha}^{\mathrm{MBC}}_{b,K}(t) = \sum_{i=1}^{n} \int \bar{K}^{\mathrm{MBC}}_{t,b}(t-s)\widehat{\alpha}^{\mathrm{LL}}_{b,K}(t)\{\widehat{\alpha}^{\mathrm{LL}}_{b,K}(s)\}^{-1} dN_i(s), \tag{3}$$

where the multiplicative kernel is

$$\bar{K}^{\mathrm{MBC}}_{t,b}(t-s) = \frac{a^{\mathrm{MBC}}_{2,K}(t) - a^{\mathrm{MBC}}_{1,K}(t)(t-s)}{a^{\mathrm{MBC}}_{0,K}(t)a^{\mathrm{MBC}}_{2,K}(t) - \{a^{\mathrm{MBC}}_{1,K}(t)\}^2} \left\{\widehat{\alpha}^{\mathrm{LL}}_{b,K}(s)\right\}^2 K_b(t-s), \tag{4}$$

with $a^{\mathrm{MBC}}_{j,K}(t) = \int_0^T K_b(t-s)(t-s)^j \left\{\widehat{\alpha}^{\mathrm{LL}}_{b,K}(s)\right\}^2 Y(s)ds$, for $j = 0, 1, 2$.

4. Cross-validation and best one-sided cross-validation of our two estimators

The two kernel hazards estimators considered in this paper depend on a bandwidth parameter that determines the smoothness degree of the resulting estimates. Choosing the bandwidth parameter is a crucial problem that starts by defining what the optimal bandwidth would be, so it can be estimated from data.

Let $\widehat{\alpha}_{b,K}$ denote a kernel hazard estimator with bandwidth b and kernel K, which can be any of the two defined in (1) or (3). Ideally we would like a bandwidth parameter b that minimizes the integrated squared error (ISE) defined as

$$\Delta_K(b) = n^{-1} \sum_{i=1}^{n} \int_0^T \{\widehat{\alpha}_{b,K}(s) - \alpha(s)\}^2 Y_i(s)w(s)ds,$$

where $w(\cdot)$ is some weight function. However, the minimizer of the integrated squared error, $\widehat{b}_{\mathrm{ISE},K}$, depends on the unknown hazard function and it is infeasible in practice. In this paper we consider $\widehat{b}_{\mathrm{ISE},K}$ as the optimal bandwidth and in this section we present estimates based on the cross-validation method. We refer the reader to Gámiz et al. (2016) for the history of cross-validation in kernel hazard estimation based on counting processes.

First notice that minimizing $\Delta_K(b)$ is equivalent to minimizing

$$n^{-1}\left[\sum_{i=1}^{n} \int_0^T \{\widehat{\alpha}_{b,K}(s)\}^2 Y_i(s)w(s)ds - 2\sum_{i=1}^{n} \int_0^T \widehat{\alpha}_{b,K}(s)\alpha(s)Y_i(s)w(s)ds\right],$$

and only the second term depends on the unknown hazard. The cross-validation approach estimates this second term from the data replacing $\alpha(s)ds$ by its empirical counterpart $dN_i(s)$. The cross-validated bandwidth, denoted by $\widehat{b}_{\mathrm{CV},K}$, is therefore the minimizer of

$$\widehat{Q}_K(b) = n^{-1}\left[\sum_{i=1}^{n} \int_0^T \{\widehat{\alpha}_{b,K}(s)\}^2 Y_i(s)w(s)ds - 2\sum_{i=1}^{n} \int_0^T \widehat{\alpha}^{[i]}_{b,K}(s)w(s)dN_i(s)\right], \tag{5}$$

where $\widehat{\alpha}^{[i]}_{b,K}(s)$ is the estimator arising when the data set is changed by setting the stochastic process $N_i(s)$ equal to 0 for all $s \in [0,T]$.

A practical and theoretical improvement of cross-validation was given in Gámiz et al. (2016) that developed double one-sided cross-validation (DO-validation), as a simple average of two indirect cross-validated bandwidths. Indirect cross-validation makes use of the fact that, under mild regularity conditions, asymptotically optimal bandwidths for two kernel estimators with different kernels K and L differ by a factor that only depends on the two kernels K and L. In indirect cross-validation one applies cross-validation to a kernel estimator with kernel L, and afterwards one multiplies the cross-validation bandwidth by the factor (depending on K and L) to get a bandwidth for the kernel estimator with kernel K. Such a construction makes sense if cross-validation for a kernel estimator with kernel L works better than cross-validation for a kernel estimator with kernel K. Double one-sided cross-validation averages the two indirect cross-validation bandwidths based on one-sided kernels: the left-sided $K_{\mathrm{L}}(u) = 2K(u)I(u<0)$, or the right-sided $K_{\mathrm{R}}(u) = 2K(u)I(u>0)$. More specifically, two one-sided cross-validation criteria, $\widehat{Q}_{K_{\mathrm{L}}}(b)$ and $\widehat{Q}_{K_{\mathrm{R}}}(b)$, are defined as in (5) but replacing K with K_{L} and K_{R}, respectively. Denoting by $\widehat{b}_{\mathrm{CV},K_{\mathrm{L}}}$ and $\widehat{b}_{\mathrm{CV},K_{\mathrm{R}}}$ their minimizers, the double one-sided cross-validation bandwidth estimate is the (conveniently) weighted average of these

$$\widehat{b}_{\mathrm{DO},K} = \rho \left\{ \frac{\widehat{b}_{\mathrm{CV},K_{\mathrm{L}}} + \widehat{b}_{\mathrm{CV},K_{\mathrm{R}}}}{2} \right\}.$$

For the local linear hazard estimator defined in (1), the factor ρ is

$$\rho^{\mathrm{LL}} = \left\{ \frac{R(K)}{R(\bar{K}^*_{\mathrm{L}})} \frac{\mu_2(\bar{K}^*_{\mathrm{L}})^2}{\mu_2(K)^2} \right\}^{1/5}. \tag{6}$$

Here, for a general kernel L, $\bar{L}^*$ denotes the equivalent local linear kernel defined as

$$\bar{L}^*(u) = \frac{\mu_2(L) - \mu_1(L)u}{\mu_2(L) - \mu_1(L)^2} L(u), \tag{7}$$

where $\mu_l(L) = \int u^l L(u)du$, for $l = 1, 2$, and $R(L) = \int L^2(u)du$. Notice that $\bar{L}^* = L$ if L is symmetric.

For the multiplicative bias corrected estimator, $\widehat{\alpha}^{\mathrm{MBC}}_{b,K}$, defined in (3), the factor ρ becomes

$$\rho^{\mathrm{MBC}} = \left\{ \frac{R(\Gamma_K)}{R(\Gamma_{\bar{K}^*_{\mathrm{L}}})} \frac{\mu_2(\bar{K}^*_{\mathrm{L}})^4}{\mu_2(K)^4} \right\}^{1/9}, \tag{8}$$

where $\Gamma_L(u) = 2L(u) - L(u) * L(u)$ is the kernel obtained by twicing the kernel L. Here $*$ denotes the convolution operator.

The asymptotic theory developed in Gámiz et al. (2016) for the local linear hazard estimator showed that left- and right-sided cross-validation have the same asymptotic properties, but different finite sample performance. There are situations where one of the two one-sided cross-validation methods breaks down so the averaging strategy of double one-sided cross-validation becomes inappropriate. The natural reaction in these cases would be

to take the side which is working fine. One common reason for one of the two one-sided cross-validated bandwidths to break down is the lack of occurrences (or exposures) in one of the two directions. Best one-sided cross-validation (BO-validation), introduced in this paper simply uses the one-sided version that, via local information, is predicted to work best at every single point t. There can therefore be both left-sided and right-sided kernels involved in best one-sided cross-validation. Imagine for example that the estimation interval is $(0,1)$, where two boundaries are present, then one would expect to use different sided kernels for a t close to the left boundary 0 and for a t close to the right boundary 1.

For the local linear hazard estimator we define the kernel estimator needed for best one-sided cross-validation as

$$\widehat{\alpha}^{\mathrm{BO,LL}}_{b,K}(t) = \int_0^T \left[\bar{K}_{t,b;\mathrm{L}}(t-s)\xi_b(t) + \bar{K}_{t,b;\mathrm{R}}(t-s)\left\{1-\xi_b(t)\right\} \right] dN(s), \tag{9}$$

where $\bar{K}_{t,b;\mathrm{L}}$ and $\bar{K}_{t,b;\mathrm{R}}$ are respectively the left and right versions of the local linear kernel $\bar{K}_{t,b}$ in (2), and $\xi_b(t)$ is a stochastic function, depending on the estimation time t and the bandwidth b, which takes the value 1 when the "best" side to consider is the indicated by the kernel K_{L}, and the value 0 otherwise. The combination of one-sided kernels that appears in the integrand of expression (9) is a kernel function which we denote as

$$\bar{K}^{\mathrm{BO,LL}}_{b,K}(t-s) = \bar{K}_{t,b;\mathrm{L}}(t-s)\xi_b(t) + \bar{K}_{t,b;\mathrm{R}}(t-s)\left\{1-\xi_b(t)\right\}. \tag{10}$$

Thus we write the estimator as $\widehat{\alpha}^{\mathrm{BO,LL}}_{b,K}(t) = \int_0^T \bar{K}^{\mathrm{BO,LL}}_{b,K}(t-s)dN(s)$.

For each time t, to designate which side is "best", $\xi_b(t)$ can be defined in terms of the occurrence process by

$$\xi^{\mathrm{O}}_b(t) = I\left(\int_{t-b}^t dN(s) < \int_t^{t+b} dN(s) \right),$$

or the exposure process by

$$\xi^{\mathrm{E}}_b(t) = I\left(\int_{t-b}^t Y(s)ds < \int_t^{t+b} Y(s)ds \right). \tag{11}$$

With any of these ξ^{O}_b or ξ^{E}_b, the best one-sided cross-validation bandwidth estimate is defined as

$$\widehat{b}^{\mathrm{LL}}_{\mathrm{BO},K} = \rho^{\mathrm{LL}} \arg\min_b \widehat{Q}^{\mathrm{BO,LL}}_K(b), \tag{12}$$

where $\widehat{Q}^{\mathrm{BO,LL}}_K$ is the cross-validation score in (5) calculated with the kernel estimator $\widehat{\alpha}^{\mathrm{BO,LL}}_{b,K}(t)$, defined in (9). In a similar way we define the best one-sided cross-validation bandwidth estimate for the multiplicative bias corrected estimator, $\widehat{b}^{MBC}_{\mathrm{BO},K}$, as in (12) but replacing the factor ρ^{LL} with ρ^{MBC}, given in (8), and defining the best one-sided cross-validation score, $\widehat{Q}^{\mathrm{BO,MBC}}_K$, with the hazard estimator

$$\widehat{\alpha}^{\mathrm{BO,MBC}}_{b,K}(t) = \int_0^T \left[\bar{K}^{\mathrm{MBC}}_{t,b;\mathrm{L}}(t-s)\frac{\widehat{\alpha}^{\mathrm{LL}}_{b,K_{\mathrm{L}}}(t)}{\widehat{\alpha}^{\mathrm{LL}}_{b,K_{\mathrm{L}}}(s)}\xi_b(t) + \bar{K}^{\mathrm{MBC}}_{t,b;\mathrm{R}}(t-s)\frac{\widehat{\alpha}^{\mathrm{LL}}_{b,K_{\mathrm{R}}}(t)}{\widehat{\alpha}^{\mathrm{LL}}_{b,K_{\mathrm{R}}}(s)}\left\{1-\xi_b(t)\right\} \right] dN(s). \tag{13}$$

5. Asymptotic theory

In this section we develop theory for the asymptotic behaviour of bandwidth selectors for the local linear hazard estimator in (1), and its multiplicative bias correction in (3). For each estimator we prove the asymptotic normality for bandwidths selectors based on cross-validation, the double one-sided cross-validation of Gámiz et al. (2016) and the new best one-sided cross-validation. Our theoretical results thus extend the results given in Gámiz et al. (2016), by including the new best one-sided cross-validation for local linear hazard estimator and considering its multiplicative bias correction.

Recall that the integrated squared error of a kernel hazard estimator, $\widehat{\alpha}_{b,L}$, with bandwidth b and kernel L, was defined as above as

$$\Delta_L(b) = n^{-1}\int_0^T \left\{\widehat{\alpha}_{b,L}(t)-\alpha(t)\right\}^2 w(t)Y(t)dt, \tag{14}$$

and its minimizer denoted as $\widehat{b}_{\mathrm{ISE},L}$. Hereafter we will make explicit reference to the considered hazard estimator using superscripts (LL for the local linear and MBC for the multiplicative bias correction). Besides a kernel denoted by K is assumed to be symmetric, while we use the notation L for a general kernel that can be asymmetric, as the one-sided kernels involved in double one-sided cross-validation and best one-sided cross-validation (see assumption A1 in Appendix A).

Let consider first the local linear hazard estimator, $\widehat{\alpha}^{\mathrm{LL}}_{b,L}$, given in (1). Following the same arguments described in Nielsen and Tanggaard (2001), the error $\widehat{\alpha}^{\mathrm{LL}}_{b,L}(t)-\alpha(t)$, can be decomposed as, $\widehat{\alpha}^{\mathrm{LL}}_{b,L}(t)-\alpha(t) = V^{\mathrm{LL}}_{b,L}(t)+B^{\mathrm{LL}}_{b,L}(t)$, where $B^{\mathrm{LL}}_{b,L}$ is a stable part converging in probability to zero,

$$B^{\mathrm{LL}}_{b,L} = \int_0^T \bar{L}_{t,b}(t-s)\left\{\alpha(s)-\alpha(t)\right\}Y(s)ds; \tag{15}$$

and $V^{\mathrm{LL}}_{b,L}$ is a variable part converging to a Normal distribution,

$$V^{\mathrm{LL}}_{b,L}(t) = \int_0^T \bar{L}_{t,b}(t-s)dM(s). \tag{16}$$

Using the above decomposition we can expand the integrated squared error for the local linear estimator, using standard martingale theory along with the approach of Mammen and Nielsen (2007). In Lemma 4 in the Supplementary Material we show that, under some regularity assumptions, $\Delta^{\mathrm{LL}}_L(b)$ in (14) is asymptotically equivalent to

$$M^{\mathrm{LL}}_L(b) = b^4\frac{\mu_2(\bar{L}^*)^2}{4}\int\left\{\alpha''(t)\right\}^2\gamma(t)w(t)dt + (nb)^{-1}R\left(\bar{L}^*\right)\int\alpha(t)w(t)dt,$$

where $\gamma(t) = n^{-1}E\left[Y(t)\right]$ is the expected exposure function. From this approximation a deterministic optimal bandwidth for the local linear estimator with kernel L is defined as

$$b^{\mathrm{LL}}_{\mathrm{MISE},L} = C^{\mathrm{LL}}_{0,L}n^{-1/5} \qquad \text{with} \qquad C^{\mathrm{LL}}_{0,L} = \left[\frac{R\left(\bar{L}^*\right)\int\alpha(t)w(t)dt}{\mu_2(\bar{L}^*)^2\int\left\{\alpha''(t)\right\}^2\gamma(t)w(t)dt}\right]^{1/5}. \tag{17}$$

Our main result in this section states the asymptotic normality of the three bandwidth estimates for the local linear hazard estimator, cross-validation, $\widehat{b}^{\mathrm{LL}}_{\mathrm{CV},K}$, double one-sided cross-validation, $\widehat{b}^{\mathrm{LL}}_{\mathrm{DO},K}$, and best one-sided cross-validation, $\widehat{b}^{\mathrm{LL}}_{\mathrm{BO},K}$; as well as the infeasible bandwidth $\widehat{b}^{\mathrm{LL}}_{\mathrm{MISE},K}$. Note that the latter is the optimal bandwidth targeted by plug-in bandwidth selection rules. The proof of the theorem is provided in the Supplementary Material.

Theorem 1 *Under assumptions A1–A3 in Appendix A, the bandwidth selectors, $\widehat{b}^{\mathrm{LL}}_{\mathrm{BO},K}$, $\widehat{b}^{\mathrm{LL}}_{\mathrm{DO},K}$,$\widehat{b}^{\mathrm{LL}}_{\mathrm{CV},K}$, and $\widehat{b}^{\mathrm{LL}}_{\mathrm{MISE},K}$, for the local linear estimator with kernel K satisfy*

$$\begin{aligned}
n^{3/10}\left(\widehat{b}^{\mathrm{LL}}_{\mathrm{BO},K}-\widehat{b}^{\mathrm{LL}}_{\mathrm{ISE},K}\right) &\longrightarrow N\left(0, S^{\mathrm{LL}}_2+S^{\mathrm{LL}}_1\Psi^{\mathrm{LL}}_{\mathrm{BO},K}\right)\\
n^{3/10}\left(\widehat{b}^{\mathrm{LL}}_{\mathrm{DO},K}-\widehat{b}^{\mathrm{LL}}_{\mathrm{ISE},K}\right) &\longrightarrow N\left(0, S^{\mathrm{LL}}_2+S^{\mathrm{LL}}_1\Psi^{\mathrm{LL}}_{\mathrm{DO},K}\right)\\
n^{3/10}\left(\widehat{b}^{\mathrm{LL}}_{\mathrm{CV},K}-\widehat{b}^{\mathrm{LL}}_{\mathrm{ISE},K}\right) &\longrightarrow N\left(0, S^{\mathrm{LL}}_2+S^{\mathrm{LL}}_1\Psi^{\mathrm{LL}}_{\mathrm{CV},K}\right)\\
n^{3/10}\left(\widehat{b}^{\mathrm{LL}}_{\mathrm{MISE},K}-\widehat{b}^{\mathrm{LL}}_{\mathrm{ISE},K}\right) &\longrightarrow N\left(0, S^{\mathrm{LL}}_2+S^{\mathrm{LL}}_1\Psi^{\mathrm{LL}}_{\mathrm{MISE},K}\right)
\end{aligned}$$

where

$$S^{\mathrm{LL}}_1 = \frac{1}{25}\frac{R(K)^{-7/5}\int\alpha^2(t)w^2(t)\,dt}{\mu_2(K)^{6/5}\left\{\int\alpha''(t)^2\gamma(t)w(t)\,dt\right\}^{3/5}\left\{\int\alpha(t)w(t)\,dt\right\}^{7/5}},$$

$$S^{\mathrm{LL}}_2 = \frac{4}{25}\frac{R(K)^{-2/5}\int\alpha''(t)^2\gamma(t)w^2(t)\alpha(t)\,dt}{\mu_2(K)^{6/5}\left\{\int\alpha(t)w(t)\,dt\right\}^{2/5}\left\{\int\alpha''(t)^2\gamma(t)w(t)\,dt\right\}^{8/5}},$$

and

$$\begin{aligned}
\Psi^{\mathrm{LL}}_{\mathrm{BO},K} &= \Psi^{\mathrm{LL}}_{\mathrm{DO},K} = \int\left\{\frac{R\left(K\right)}{R\left(\bar{L}^*\right)}\left(H_L-G_L\right)\left(\rho^{\mathrm{LL}}u\right)-H_K(u)\right\}^2 du,\\
\Psi^{\mathrm{LL}}_{\mathrm{CV},K} &= \int\left\{G_K(u)\right\}^2\,du,\\
\Psi^{\mathrm{LL}}_{\mathrm{MISE},K} &= \int\left\{H_K(u)\right\}^2\,du,
\end{aligned}$$

defining the functions $G_L(\cdot)$ and $H_L(\cdot)$ as

$$\begin{aligned}
G_L(w) &= I\left(w\neq 0\right)2\bar{L}^*_1(w),\\
H_L(w) &= I\left(w\neq 0\right)\int\bar{L}^*(u)\left\{\bar{L}^*_1(u+w)+\bar{L}^*_1(u-w)\right\}du,
\end{aligned}$$

where $\bar{L}^*_1(u) = -\bar{L}^*(u)-u\bar{L}^{*'}(u)$, with $L=K$ and $L=K_{\mathrm{L}}$.

Remark 2 *Gámiz et al. (2016) pointed out that all bandwidth estimates have similar asymptotics with the only difference of the factor $\Psi^{\mathrm{LL}}_{\cdot,K}$. These authors considered three*

Table 1: Comparison of asymptotic variances among bandwidth selectors. Factors $\Psi^{\bullet}_{\bullet,K}$ defined in Theorems 1 and 3, are shown for the local linear hazard estimator and its multiplicative bias correction, with three common symmetric kernels K: Epanechnikov, quartic and sextic.

	Local linear estimator			Multiplicative bias correction		
Method	Epanechnikov	Quartic	Sextic	Epanechnikov	Quartic	Sextic
BO-validation	1.09	0.95	1.18	4.41	2.44	2.05
DO-validation	1.09	0.95	1.18	4.41	2.44	2.05
Cross-validation	3.60	2.86	3.49	9.87	6.10	6.50
Plug-in	0.36	0.46	0.59	0.84	0.95	1.31

common choices of the kernel K (Epanechnikov, quartic and sextic kernels) and calculated the numerical value of this factor (multiplied by 2 for convenience in this former paper). It allows the comparison of the asymptotic performance of bandwidth selectors. These numerical values are reported in the first rows of Table 1, where now we have added the new best one-sided cross-validation. Notice that the actual values of S_1^{LL} and S_2^{LL} depend on the unknown hazard function α and its derivatives, but also on the exposure function γ, which are specific to the inference problem. In Appendix B.1 we have calculated the ratio of these two terms for one particular hazard model. The idea is to find out whether the discrepancy in the values of $\Psi^{\mathrm{LL}}_{\bullet,K}$ among bandwidth selectors shown in the table could be misleading in terms of the actual variances. For the considered model the term S_1^{LL} is almost a half of S_2^{LL}, both when exposure time goes zero and infinity. For the multiplicative bias corrected hazard estimator we will discuss in Remark 4 that the corresponding S_1 term can completely dominate the term S_2 in certain situations.

Consider now the multiplicative bias correction of the local linear hazard defined in (3), $\widehat{\alpha}^{\mathrm{MBC}}_{b,L}$, with bandwidth b and kernel L. As for the local linear estimator above, we define the corresponding integrated squared error for its multiplicative bias correction as in (14) and denote it as $\Delta_L^{\mathrm{MBC}}(b)$. We denote its minimizer as $\widehat{b}^{\mathrm{MBC}}_{\mathrm{ISE},L}$.

We consider the decomposition $\widehat{\alpha}^{\mathrm{MBC}}_{b,L}(t)-\alpha(t) = B^{\mathrm{MBC}}_{b,L}(t)+V^{\mathrm{MBC}}_{b,L}(t)$, where $B^{\mathrm{MBC}}_{b,L}(t)$ is a stable term converging in probability to zero, and $V^{\mathrm{MBC}}_{b,L}(t)$ is a variable term converging to a Normal distribution. These two terms are defined as follows:

$$V^{\mathrm{MBC}}_{b,L}(t) = \int f^{\mathrm{MBC}}_{t,b}(s) dM(s)$$

where

$$\begin{aligned} f^{\mathrm{MBC}}_{t,b}(s) &= \bar{L}^{\mathrm{MBC}}_{t,b}(t-s)\frac{\widehat{\alpha}^{\mathrm{LL}}_{b,L}(t)}{\widehat{\alpha}^{\mathrm{LL}}_{b,L}(s)} + \bar{L}_{t,b}(t-s) - \\ &\quad \int_0^T \bar{L}^{\mathrm{MBC}}_{t,b}(t-u)\frac{\widehat{\alpha}^{\mathrm{LL}}_{b,L}(t)}{\widehat{\alpha}^{\mathrm{LL}}_{b,L}(u)}\bar{L}_{u,b}(u-s)Y(u)du \end{aligned}$$

with $\bar{L}^{\mathrm{MBC}}_{t,b}(t-s)$ defined as in (4) for the kernel L,

and

$$\begin{aligned} B^{\mathrm{MBC}}_{b,L}(t) &= B^{\mathrm{LL}}_{b,L}(t) + \int \bar{L}^{\mathrm{MBC}}_{t,b}(t-s)\widehat{\alpha}^{\mathrm{LL}}_{b,L}(t)\left\{\widehat{\alpha}^{\mathrm{LL}}_{b,L}(s)\right\}^{-1} B^{\mathrm{LL}}_{b,L}(s)Y(s)ds \\ &= \int \bar{L}^{\mathrm{MBC}}_{t,b}(t-s)\widehat{\alpha}^{\mathrm{LL}}_{b,L}(t)\left\{\beta_{b,L}(t)-\beta_{b,L}(s)\right\}Y(s)ds, \end{aligned}$$

with $\beta_{b,L}(s) = \{\widehat{\alpha}^{\mathrm{LL}}_{b,L}(s)\}^{-1} B^{\mathrm{LL}}_{b,L}(s)$, where $B^{\mathrm{LL}}_{b,L}$ and $V^{\mathrm{LL}}_{b,L}$ are the stable and variable terms for the local linear estimator given in (15) and (16), respectively.

Using the above decomposition and derivations similar to the local linear case we can expand the integrated squared error of the multiplicative bias corrected estimator. In Lemma 7 in the Supplementary Material we show that, under some regularity assumptions, $\Delta_L^{\mathrm{MBC}}(b)$ is asymptotically equivalent to

$$M_L^{\mathrm{MBC}}(b) = b^8\frac{\mu_2(\bar{L}^*)^4}{16}\int h(t)^2\gamma(t)w(t)dt + (nb)^{-1}R\left(\Gamma_{\bar{L}^*}\right)\int \alpha(t)w(t)dt,$$

with $h(t) = \alpha(t)\left\{\alpha''(t)/\alpha(t)\right\}''$. From this approximation a deterministic optimal bandwidth for the multiplicative bias corrected estimator with kernel L is defined as

$$b^{\mathrm{MBC}}_{\mathrm{MISE,L}} = C^{\mathrm{MBC}}_{0,L}n^{-1/9}; \qquad C^{\mathrm{MBC}}_{0,L} = \left\{\frac{R\left(\Gamma_{\bar{L}^*}\right)\int \alpha(t)w(t)dt}{\frac{\mu_2(\bar{L}^*)^4}{2}\int h(t)^2\gamma(t)w(t)dt}\right\}^{1/9}, \tag{18}$$

where $\Gamma_{\bar{L}^*}(u) = 2\bar{L}^*(u) - \bar{L}^*(u) * \bar{L}^*(u)$ is the kernel obtained by twicing the equivalent kernel, $\bar{L}^*$, given in (7).

The following theorem states the asymptotic normality of the three bandwidth estimates, as well as the minimizer of the mean integrated squared error, for the multiplicative bias corrected hazard estimator with kernel K. The proof is provided in the Supplementary Material.

Theorem 3 *Under assumptions A1, A2' and A3', the bandwidth selectors $\widehat{b}^{\mathrm{MBC}}_{\mathrm{BO},K}$, $\widehat{b}^{\mathrm{MBC}}_{\mathrm{DO},K}$, $\widehat{b}^{\mathrm{MBC}}_{\mathrm{CV},K}$, and $\widehat{b}^{\mathrm{MBC}}_{\mathrm{MISE},K}$ satisfy*

$$\begin{aligned} n^{3/18}\left(\widehat{b}^{\mathrm{MBC}}_{\mathrm{BO},K} - \widehat{b}^{\mathrm{MBC}}_{\mathrm{ISE},K}\right) &\longrightarrow N\left(0, S_2^{\mathrm{MBC}} + S_1^{\mathrm{MBC}}\Psi^{\mathrm{MBC}}_{\mathrm{BO},K}\right) \\ n^{3/18}\left(\widehat{b}^{\mathrm{MBC}}_{\mathrm{DO},K} - \widehat{b}^{\mathrm{MBC}}_{\mathrm{ISE},K}\right) &\longrightarrow N\left(0, S_2^{\mathrm{MBC}} + S_1^{\mathrm{MBC}}\Psi^{\mathrm{MBC}}_{\mathrm{DO},K}\right) \\ n^{3/18}\left(\widehat{b}^{\mathrm{MBC}}_{\mathrm{CV},K} - \widehat{b}^{\mathrm{MBC}}_{\mathrm{ISE},K}\right) &\longrightarrow N\left(0, S_2^{\mathrm{MBC}} + S_1^{\mathrm{MBC}}\Psi^{\mathrm{MBC}}_{\mathrm{CV},K}\right) \\ n^{3/18}\left(\widehat{b}^{\mathrm{MBC}}_{\mathrm{MISE},K} - \widehat{b}^{\mathrm{MBC}}_{\mathrm{ISE},K}\right) &\longrightarrow N\left(0, S_2^{\mathrm{MBC}} + S_1^{\mathrm{MBC}}\Psi^{\mathrm{MBC}}_{\mathrm{MISE},K}\right) \end{aligned}$$

where

$$S_1^{\mathrm{MBC}} = \frac{2^{1/3}}{9^2}\frac{R\left(\Gamma_K\right)^{-5/6}\int \alpha(t)^2w(t)^2\ dt}{\mu_2(K)^{4/3}\left\{\int h(t)^2\gamma(t)w(t)\ dt\right\}^{1/3}\left\{\int \alpha(t)w(t)\ dt\right\}^{5/3}},$$

$$S_2^{\mathrm{MBC}} = \frac{2^{10/3}}{9^2}\frac{R\left(\Gamma_K\right)^{-2/3}\int h(t)^2\gamma(t)w(t)^2\alpha(t)\ dt}{\mu_2(K)^{4/3}\left\{\int \alpha(t)w(t)\ dt\right\}^{2/3}\left\{\int h(t)^2\gamma(t)w(t)\ dt\right\}^{4/3}},$$

with

$$\begin{aligned}\Psi^{\mathrm{MBC}}_{\mathrm{BO},K} &= \Psi^{\mathrm{MBC}}_{\mathrm{DO},K} = \int \left\{ \frac{R\left(\Gamma_K\right)}{R\left(\Gamma_{\tilde{L}^*}\right)} \left(H_{\Gamma_{\tilde{L}^*}} - G_{\Gamma_{\tilde{L}^*}}\right)\left(\rho^{\mathrm{MBC}} u\right) - H_{\Gamma_K}(u) \right\}^2 du, \\ \Psi^{\mathrm{MBC}}_{\mathrm{CV},K} &= \int \left\{G_{\Gamma_K}(u)\right\}^2 du, \\ \Psi^{\mathrm{MBC}}_{\mathrm{MISE},K} &= \int \left\{H_{\Gamma_K}(u)\right\}^2 du.\end{aligned}$$

where the functions $G_L(\cdot)$ and $H_L(\cdot)$ are defined as in Theorem 1, with $L = \Gamma_K$ and $L = \Gamma_{\tilde{L}^*}$, defined above.

Remark 4 *The result above shows that all bandwidth selectors have similar asymptotics with the only difference of the factor $\Psi^{\mathrm{MBC}}_{\cdot,K}$. A similar conclusion was derived for the local linear estimator. The three last columns of Table 1 show the value of this factor for three common choices of K. As for the local linear estimator we have calculated the ratio $S_1^{\mathrm{MBC}}/S_2^{\mathrm{MBC}}$ for a particular hazard model, the results are shown in Appendix B.2. In this case the term S_1^{MBC}, relative to the term S_2^{MBC}, is negligible when exposure time goes to zero, but dominates completely (infinite times bigger) when exposure time goes to infinity.*

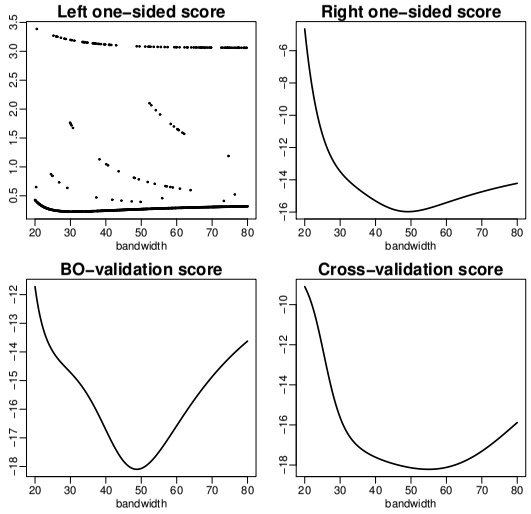
6. Applications

6.1. Old-age mortality

Our first application is on fitting hazard mortality curves for old-age population. We consider mortality data of women in Iceland in the calendar year 2006, with ages from 40 to 110. The same data were considered by Gámiz et al. (2016) and are available in the `DOvalidation` R-package (Gámiz et al., 2017). This package provides also functions implementing the hazard estimators and the bandwidth selection methods described above. The data were obtained from the Human Mortality Database and consist of aggregated yearly occurrences and exposures. Gámiz et al. (2016) showed that estimating the hazard from these data is challenge at the oldest ages. The lack of exposure at the right end and the few observed deaths induce a marked boundary effect precisely in the area of interest, the old ages. For these data we have calculated the two hazard estimators described in this paper, local linear and its multiplicative bias correction, using three bandwidth selectors: cross-validation, double one-sided cross-validation and the new best one-sided cross-validation. The cross-validation scores involved in these methods have been defined using a weighting function such that $w(s)Y(s) \equiv 1$, so all points in the time interval where the hazard function is estimated are evaluated with the same weight. This is different from Gámiz et al. (2016) where the weighting function was chosen so only areas where the exposure is significant contribute to the criteria. Notice that this makes an important difference in this data set where the end of the time interval comprises almost no exposure.

Before looking at the resulting hazard estimates we shall look at the cross-validation scores to be minimized for each bandwidth selection method. Figure 1 shows the cross-validated scores for each method considering the multiplicative bias corrected estimator. The local linear case looks quite similar and can be found in the Supplementary Material. From these plots we can see that the left one-sided score is not well behaved for both

Figure 1: Mortality data: bandwidth selection scores with multiplicative bias corrected hazard estimator.

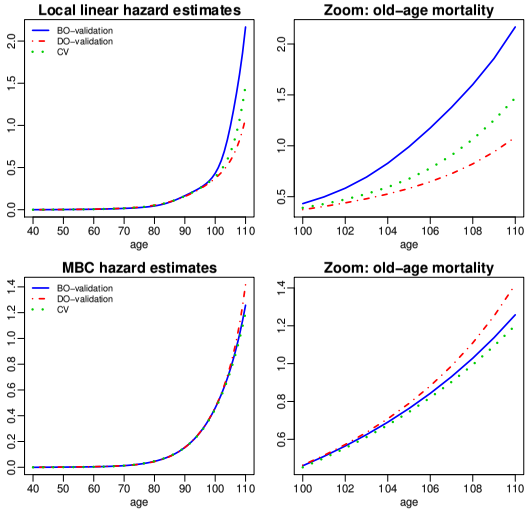


hazard estimators. Therefore the average DO-validated bandwidth becomes unreliable, even though the obtained values seem to be sensible ($\widehat{b}_{\mathrm{DO}} = 27.3$ for the local linear estimator and $\widehat{b}_{\mathrm{DO}} = 40$ for its multiplicative bias correction). On the other hand the best one-sided cross-validation method shows a clear minimum in both cases and, as expected, it moves close to the one-sided cross-validated bandwidth that is working fine (the right side in this case). Best one-sided cross-validation in this case has been calculated using the exposure process, that is, for each time t we use the function $\xi_b^E(t)$ given in (11). However the results are quite similar using the occurrence process instead. Figure 2 shows the resulting hazard estimates from each method and type of hazard estimate. Note from these plots that the multiplicative bias corrected hazard is more robust to the bandwidth choice than the local linear. Also the new best one-sided cross-validation method seems to provide a reasonable estimate for old-age mortality in both cases.

6.2. Prediction of outstanding liabilities in non-life insurance

We consider now a non-standard forecasting problem that arises in non-life insurance. The goal is to forecast the number of future claims from contracts underwritten in the past, which have not yet been reported. Typically actuaries are responsible of getting these forecasts, which represent perhaps the most important number in the accounts of the company (see

Figure 2: Comparison of hazard estimates from female mortality data in Iceland.



Martínez-Miranda et al. (2013) for a detailed background of this problem). Here we analyse a data set of reported and outstanding claims from a motor business in UK. The same data set was previously considered by Martínez-Miranda et al. (2013) and consists of $n = 1558$ large claims reported between January 1990 and March 2012. From a statistical perspective the data could be described as a sample $\{(X_1, Z_1), \ldots, (X_n, Z_n)\}$, where X_i denotes the underwriting date of the ith claim, and Z_i the corresponding reporting delay, this is, the time between the underwriting date and the reporting date of the claim. The sample is right truncated since it can be observed only those claims for which the underwriting time plus the reporting delay is not greater than the calendar time of data collection. Hence data exist on a triangle with $X_i + Z_i \leq 31$ March 2012, and $X_i + Z_i$ represents the calendar time. The aim is to forecast the mass of the unobserved, future triangle, where $X_i + Z_i > 31$ March 2012, which corresponds to the number of claims underwritten in the past which have not been reported yet. The problem is formulated assuming that the maximum reporting delay is 267 months, in the actuarial literature this assumption is described as the triangle is fully run off. Another challenge of the data set for this problem is that the data are only available in an aggregated way. This is a common feature of this kind of data in the reserving departments of the insurance companies. This means that the available observations are counts living in a triangle of dimension 267×267. Specifically for our data set the triangle has entries $\mathcal{N}_{x,z} = \sum_{i=1}^{n} I(X_i = x,\ Z_i = z)$, with $x, z \in \{1, \ldots, 267\}$, describing the number of claims underwritten in the xth month and reported in the zth month.

Martínez-Miranda et al. (2013) showed that a multiplicative structured density model, $f(x, z) = f_1(x)f_2(z)$, can be used to forecast the claims where the components f_1 and f_2 are the underwriting time density and the reporting time density, respectively. The assumption of a multiplicative density means that the reporting delay does not depend on the underwriting date. Using the counting process formulation considered in this paper, Hiabu et al. (2016) solved the forecasting problem estimating the two density components using a time-reversal approach. Data are transformed to the time reversed scale so the right-truncation problem is replaced by the more tractable left-truncation. Using the same time-reversal approach, we now use the hazard estimation methods presented in the previous sections to estimate the backward hazard functions corresponding to the two components, underwriting (α_1) and reporting delay (α_2). From these hazard estimates the density component estimates can be derived multiplying by respective estimators of the survival functions.

From the above description we solve the forecasting problem considering both local linear hazard estimator and its multiplicative bias correction. For each hazard component, the bandwidth parameters for these estimators have been estimated using cross-validation, double one-sided cross-validation and best one-sided cross-validation. In the three cases we use weighting functions for the involved cross-validation scores that are appropriate for the forecasting problem. Specifically, following the discussion in Hiabu et al. (2016), to estimate α_1 we consider weights $w_1(t) = \widehat{S}_1^2(t)\left\{1 - \widehat{S}_2(t)\right\}^2 / Y_1(t)$, where $\widehat{S}_1$ and $\widehat{S}_2$ are estimators of the survival functions of each component (underwriting time and the reporting time delay) on the reversed time scale; and $Y_1(t)$ is the risk process for the first component. In a similar way we define the weights to estimate α_2. As in the mortality study best one-sided cross-validation has been calculated using the exposure process.

Figure 3 shows the forecasts of the number of claims reported in the future calendar months. Table 2 shows these forecasts aggregated in years. The forecasts are given for each hazard estimator and bandwidth estimate. We have also included the forecasts derived from the chain ladder method, which involves histogram type estimators of the underwriting and reporting density components. The chain ladder method is the classical approach used in the insurance companies (see Martínez-Miranda et al. (2013) for more details about this approach). The plot of the forecasts shows that the classical insurance method chain ladder is overestimating the liabilities, while the kernel hazard methods provide lower forecasts. Previous empirical analyses with these data described in Martínez-Miranda et al. (2013) agree with this result and recommend multiplicative bias corrected local linear estimators for this kind of data. Looking at the results from the kernel estimators we can see that double one-sided cross-validation and best one-sided cross-validation provide similar forecasts when the local linear estimator is considered, but the results are quite different for the multiplicative bias corrected estimator. The predicted total number of claims using double one-sided cross-validated bandwidth is about 299, compared to 313 using best one-sided cross-validation. Our concern is that double one-sided cross-validation might not be behaving properly in this situation. A close inspection to the cross-validation scores that are minimized to derive these bandwidth estimates reveals what is happening. Figures 4 and 5 show these cross-validation scores when the multiplicative bias corrected estimator is considered for both underwriting and reporting delay components. From these plots we can see that the right one-sided score completely breaks down for the underwriting time compo-

Table 2: Forecasts of the number of claims to be reported in the future calendar years.

Year	CLM	LL-CV	LL-DO	LL-BO	MBC-CV	MBC-DO	MBC-BO
2012	99.95	76.85	77.98	77.95	80.55	81.75	81.76
2013	97.23	75.06	75.52	76.86	81.18	75.82	81.68
2014	74.32	58.75	59.05	60.04	62.23	58.89	62.88
2015	49.18	38.88	39.06	39.44	40.31	38.81	41.20
2016	24.52	19.42	19.50	19.66	20.01	19.34	20.44
2017	11.61	9.35	9.39	9.44	9.60	9.45	9.76
2018	6.21	5.07	5.06	5.09	5.15	4.99	5.27
2019	3.24	2.54	2.53	2.52	2.52	2.53	2.61
2020	1.36	1.25	1.23	1.23	1.23	1.18	1.22
2021	0.99	1.03	1.02	1.02	0.99	0.95	0.95
2022	1.11	0.85	0.84	0.85	0.87	0.83	0.88
2023	1.06	0.71	0.71	0.73	0.81	0.80	0.85
2024	1.20	0.81	0.81	0.84	0.93	0.90	0.94
2025	1.14	0.91	0.92	0.95	0.97	0.93	0.94
>2025	1.94	1.59	1.59	1.61	1.51	1.48	1.55
Total	375.07	293.07	295.20	298.23	308.86	298.69	312.92

nent, exhibiting several local minima. For the reporting delay component the score function continues decreasing as the value of the bandwidth increases, so it reaches the minimum at the upper limit of the search interval of bandwidths. The left one-sided score behaves more reasonably for the underwriting component but again breaks down for the reporting delay component. This means that one shouldn't trust the double one-sided cross-validation bandwidth derived from these two one-sided criteria, even though the derived estimates in this case turned to be reasonable values, $\widehat{b}_{\mathrm{DO}} = 55.8$ for the underwriting time, and $\widehat{b}_{\mathrm{DO}} = 31.6$ for the delay. On the contrary, the new best one-sided cross-validation method provides bandwidth estimates of $\widehat{b}_{\mathrm{BO}} = 43.4$ for the underwriting time and $\widehat{b}_{\mathrm{BO}} = 11.8$ for the delay, exhibiting well-behaved minimization scores as shown in Figure 4. Regarding to the cross-validation method it exhibits a rather flat score in the underwriting component leading to the large bandwidth estimate of $\widehat{b}_{\mathrm{CV}} = 63.5$, and a value of $\widehat{b}_{\mathrm{CV}} = 11.6$ for the delay that is close to the best one-sided cross-validated bandwidth. The impact of the cross-validated bandwidths on the forecasts is moderated, about 309 predicted claims compared to the 313 from best one-sided cross-validation. We have performed the same inspection with the local linear estimators. These plots can be seen in the Supplementary Material. The picture is again quite similar showing a poor performance of double one-sided cross-validation, however the impact on the forecasts in this case is not substantial. The total number predicted from cross-validation is about 293, compared to 295 from double one-sided cross-validation and 298 for best one-sided cross-validation.

7. Finite sample performance

In this section we investigate the finite sample performance of the new best one-sided cross-validation method for the local linear hazard estimator and its multiplicative bias correction. We have considered the same five hazard models described in Gámiz et al. (2016) (see also

Figure 3: Number of outstanding claims forecast using the local linear estimator and its multiplicative bias correction.

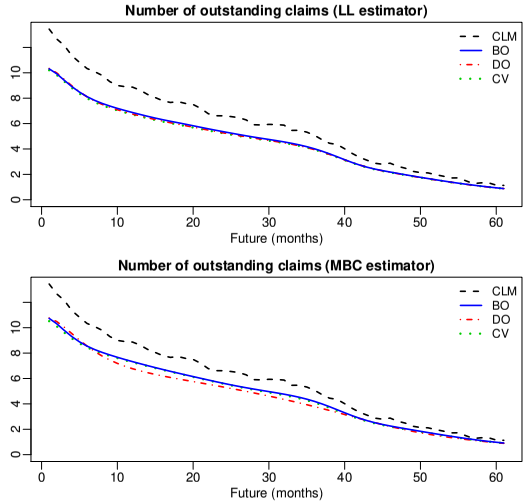


Figure 4: Underwriting component: bandwidth selection scores with multiplicative bias corrected hazard estimator.

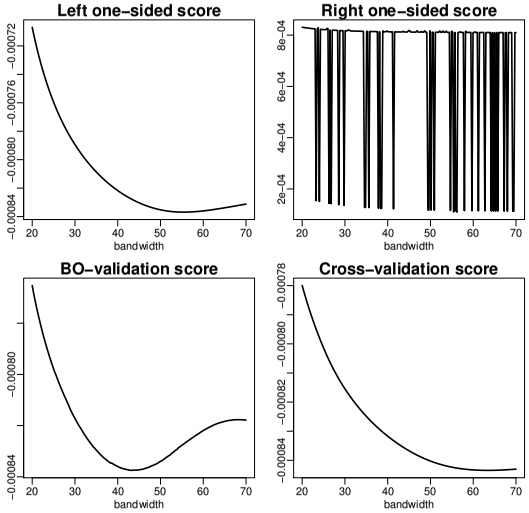
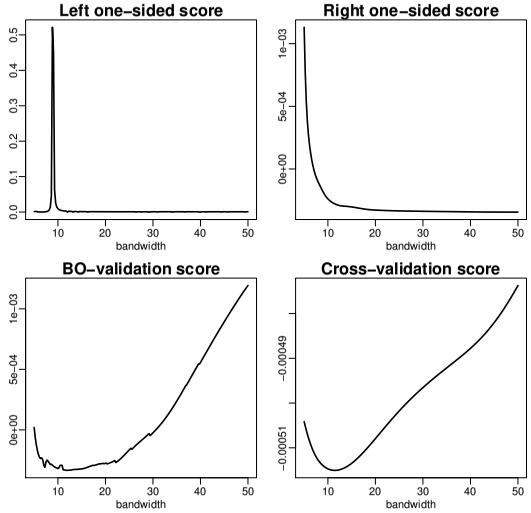


Figure 5: Reporting delay component: bandwidth selection scores with multiplicative bias corrected estimator.



Supplementary Material). The first four models consist of mixtures of Beta densities. Model 5 shows an exponential increase common in hazard mortality rates as those described in the first case study of Section 6. From each model we have simulated samples with three different sample sizes and two sampling schemes, right censoring with and without left truncation. For models 1 to 4 we have considered sample sizes $n = 100, 1000, 10000$, and for model 5, $n = 50000, 75000, 100000$. The number of Monte Carlo replications for each case has been always 500. We use the same mechanism to simulate data as in Gámiz et al. (2016). It generates data in aggregated form (number of occurrences and exposure) for an equally-spaced grid of size R defined on the time interval, and always produces right censored samples. For models 1 to 4 the time interval is $(0, 1)$ and we have defined the grid length with $\delta_R = 1/(R + 1)$. For model 5 time lies in the interval $(40, 110)$ and we have defined the grid length with $\delta_R = 70/(R + 1)$. The grid size has been chosen equal to $R = 500$ in both cases. We shall denote the grid points by t_r $(r = 1, \ldots, R)$. In the case of samples without left truncation, for a sample of n individuals, the number of occurrences at time t_r, denoted as O_r, have been generated from the binomial distribution $Bi\{Y_r, \alpha(t_r)\delta_R\}$, for $r = 1, \ldots, R$. Here Y_r denotes the size of the risk set at the beginning of the rth interval of the grid. The total number of simulated occurrences does not sum to n. Some of the simulated individuals are finally right censored, because they are still at risk at the end of the interval. Therefore our simulated sample are right censored and the censoring rates are around 20–30% for all models. When adding left truncation, independent truncation times are generated from the Uniform distribution.

From the simulated aggregated data we have calculated the local linear hazard estimator and its multiplicative bias correction using the sextic kernel: $K(x) = 3003/2048(1 - x^2)^6 I(-1 < x < 1)$, as in the two data analyses above. For each hazard estimator we have compared the best one-sided cross-validated bandwidth with cross-validation and double one-sided cross-validation. The performance of the bandwidth estimates have been analysed with respect to the (Monte Carlo approximated) mean integrated squared error of the resulting kernel hazard estimator. We shall refer to this performance measure as empirical MISE, denoted as $m_1(\widehat{b})$, for each bandwidth estimate $\widehat{b}$. As benchmarks in our analysis we have considered two infeasible optimal bandwidths: the bandwidth minimizing the integrated squared error criterion, $\widehat{b}_{\text{ISE}}$, and the bandwidth minimizing the empirical MISE. To compute all bandwidth estimates we have considered grids of 100 equally spaced bandwidth values chosen around $\widehat{b}_{\text{ISE}}$, for each model and sample size. All criteria have been defined using a weighting function such that $w(s)Y(s) \equiv 1$, so all points in the time interval where the hazard function is estimated are evaluated with the same weight. As we pointed out in our first case study this is different from Gámiz et al. (2016), and it makes an important difference in models such as Model 5 where the end of the time interval comprises almost no exposure.

Table 3 summarizes the simulation results in the case of samples with right censoring and left truncation. In this table bandwidth estimates are compared according to measure m_1. For convenience we report a relative measure to indicate when best one-sided cross-validation outperforms cross-validation. The relative measure is defined as:

$$Rerr(BO) = \left\{ m_1(\widehat{b}_{\text{CV}}) - m_1(\widehat{b}_{\text{ISE}}) \right\} / \left\{ m_1(\widehat{b}_{\text{BO}}) - m_1(\widehat{b}_{\text{ISE}}) \right\}. \tag{19}$$

With this definition values of $Rerr(BO)$ above 1 indicate that best one-sided cross-validation outperforms cross-validation. An analogous relative measure, $Rerr(DO)$, has been defined for double one-sided cross-validation. Notice that $Rerr(BO)$ greater than $Rerr(DO)$ indicates that best one-sided cross-validation outperforms double one-sided cross-validation. An overall view of the numbers in the table confirms that best one-sided cross-validation for the multiplicative hazard estimator always outperforms cross-validation, exhibiting $Rerr(BO)$ values above 1, and double one-sided cross-validation for all models except for few cases, where double one-sided cross-validation provides slightly lower empirical MISE values. The results for the local linear estimator show that double one-sided cross-validation and best one-sided cross-validation behave quite similarly, both outperforming in general cross-validation. The case of samples without left truncation is shown in Table 4. It brings similar conclusions though in this case best one-sided cross-validation is beaten by double one-sided cross-validation for Model 5. This case deserves a deeper analysis and it is shown in Table 5. In this table we have shown the empirical MISE defined above and denoted by $m_1(\widehat{b})$, for each bandwidth estimate $\widehat{b}$, as well as the average of the bandwidth estimates for all the samples $(avg(\widehat{b}))$, and we have included the left and right one-sided cross-validated bandwidths, from which double one-sided cross-validation is derived. From these results we can clearly see that the left one-sided bandwidth completely breaks down, for all sample sizes and both hazard estimators, while the right side behaves well (notice the large values of the empirical MISE for the left one-sided bandwidth in contrast with those values for the right one-sided bandwidth). The average of the left and right one-sided bandwidths (which double one-sided cross-validation performs) seems to be hiding the problem of the left side, and sometimes it even provides quite reasonable values. Notice that the double one-sided bandwidths are on average closer to the best ISE-optimal bandwidths than the best one-sided cross-validation for the multiplicative bias corrected estimator. However this happens because the double one-sided bandwidth is the average of a small left one-sided bandwidth and a large right one-sided bandwidth. On the other hand best one-sided cross-validation is behaving as the best of the two sides, as we would expect. A similar picture can be seen when analysing the behaviour of double one-sided cross-validation for Model 4 in the case of truncated samples (the full simulation results are provided in the Supplementary Material).

In summary, the simulation results indicate that best-one sided cross-validation and double one-sided cross-validation do better than one-sided cross-validation (that sometimes breaks down) and standard cross-validation. However, it is not always which one is the better of best one-sided cross-validation or double one-sided cross-validation. We suggest to try out both best one-sided and double one-sided cross-validation in any empirical study.

8. Discussion

The overall conclusion of our finite sample section is that double one-sided cross-validation and best one-sided cross-validation are better than standard cross-validation, however, there is no clear winner between double one-sided cross-validation and best one-sided cross-validation. In practice we would suggest to consider both. It is also concluded that the multiplicative bias corrected local linear hazard estimator is more often than not better than the simpler local linear hazard estimator. There is a tendency that double one-sided cross validation is better than best one-sided cross-validation for the simple local linear case,

Table 3: Simulation results for datasets with right censoring and left truncation. Hazard estimators and bandwidth selectors are compared by the relative measure $Rerr(\cdot)$ defined in (19).

Model	n	LL-DO	LL-BO	MBC-DO	MBC-BO
1	100	1.55	1.25	1.47	1.79
	1000	2.32	2.00	0.97	2.88
	10000	1.90	1.71	1.82	3.30
2	100	2.28	2.04	0.46	2.47
	1000	2.42	1.99	0.15	3.66
	10000	2.18	1.84	0.34	3.81
3	100	1.86	1.74	1.47	1.27
	1000	0.96	0.99	0.82	1.19
	10000	2.20	2.07	2.12	3.50
4	100	0.08	1.12	2.13	0.92
	1000	2.51	1.91	2.30	1.08
	10000	2.17	1.83	3.76	2.62
5	50000	1.62	1.70	1.77	2.09
	75000	2.04	2.18	1.41	2.31
	10^5	1.68	1.73	1.07	1.90

Table 4: Simulation results for datasets without left truncation. Hazard estimators and bandwidth selectors are compared by the relative measure $Rerr(\cdot)$ defined in (19).

Model	n	LL-DO	LL-BO	MBC-DO	MBC-BO
1	100	2.58	2.05	0.89	2.51
	1000	2.62	2.27	1.24	4.60
	10000	2.75	2.47	1.62	8.57
2	100	2.55	1.81	0.22	2.92
	1000	2.70	2.29	0.10	3.51
	10000	2.63	2.40	0.26	4.71
3	100	1.50	1.40	0.99	0.70
	1000	2.72	2.33	0.74	3.40
	10000	1.81	2.10	0.65	3.52
4	100	2.03	1.89	2.19	1.13
	1000	2.09	2.03	1.28	0.90
	10000	1.24	1.28	1.03	1.65
5	50000	0.80	6.45	5.33	1.60
	75000	0.63	5.47	4.63	1.96
	10^5	0.56	4.32	4.16	2.28

Table 5: Performance of double one-sided cross-validation in simulations. The empirical MISE (multiplied by 10^6), $m_1(\widehat{b})$, and the average bandwidth estimates, $avg(\widehat{b})$, are shown for samples generated from Model 5 without left truncation.

	n		ISE	MISE	CV	OSCV-l	OSCV-r	DO	BO
LL	50000	$m_1(\widehat{b})$	1.14	2.16	13.31	13424.45	3.04	16.29	3.03
		$avg(\widehat{b})$	9.71	22.48	18.09	2.82	10.99	6.91	10.98
	75000	$m_1(\widehat{b})$	0.82	1.32	8.82	4835.00	1.92	10.28	1.92
		$avg(\widehat{b})$	9.37	20.91	16.18	3.57	9.77	6.67	9.76
	10^5	$m_1(\widehat{b})$	0.53	0.86	4.44	1894.00	1.43	7.55	1.43
		$avg(\widehat{b})$	9.04	20.12	14.34	3.98	8.98	6.48	8.96
MBC	50000	$m_1(\widehat{b})$	0.33	0.72	11.84	203897.80	7.01	2.49	7.54
		$avg(\widehat{b})$	18.93	21.70	45.62	17.69	41.43	29.56	40.78
	75000	$m_1(\widehat{b})$	0.23	0.43	5.74	92670.27	3.01	1.42	3.04
		$avg(\widehat{b})$	18.45	20.91	38.88	18.02	35.20	26.61	32.31
	10^5	$m_1(\widehat{b})$	0.15	0.27	3.51	16302.04	1.65	0.96	1.63
		$avg(\widehat{b})$	17.87	20.12	32.85	19.01	30.79	24.90	27.20

while the performance is vice versa when using the multiplicative kernel hazard estimation. The exception from this rule seems to be our finite sample study in Table 5 inspired by our real-life mortality data.

In Appendix B we have studied how big the second noise component (the one that is improved by our new bandwidth selectors) is compared to the first noise component in the bandwidth selection (the one that is same across all the bandwidth selectors). Using our real-life mortality data as inspiration, we investigate a simple version of a Gomperz-Makeham shaped hazard and conclude that for the local linear hazard the second variance component is almost a half of the first component, both when exposure time goes zero and infinity. This picture is very different for the multiplicative hazard estimator, where the second variance component is negligible when exposure time goes to zero, but dominates completely (infinite times bigger) when exposure time goes to infinity. One can in other words get all kinds of relationships between the second variance component and first variance component. This calls for more research, but it seems already clear that there will be many situations, where we cannot rely on simple cross-validation (not to mention less efficient training-and-learning algorithms), because cross-validation has a very high second variance component and that second variance component will occasionally dominate completely the smoothing selection problem. We believe our findings to be relevant for machine learning, big data and artificial intelligence in general, where over-reliance of simple training-and-learning or cross-validation might lead to false-discoveries because of the noise involved. Methods such as double one-sided cross-validation, best one-sided cross-validation, or other improved smoothing procedures dampening the second noise component, therefore seem to be a very important element of future research in machine learning and related fields.

9. Conclusion

We have proposed a new bandwidth selection method for local linear hazard estimation and its multiplicative bias correction. Our proposal is called best one-sided cross-validation and consists of an improvement of the double one-sided cross-validation of Gámiz et al. (2016). Best one-sided cross-validation solves the lack of stability of double one-sided cross-validation in practice via a local information principle.

Our empirical studies show that best one-sided cross-validation provides a good strategy for bandwidth selection for both local linear and multiplicative bias corrected hazard estimators. Best one-sided cross-validation inherits the good properties of one-sided cross-validation while avoiding the stability problems that double one-sided cross-validation sometimes faces. The current algorithm is only about optimisation of statistical inference. However, it could be also interesting to consider computational performance, see for example Kapotufe and Verma (2017).

Detailed mathematical theory at the level of Hall and Marron (1987) and Gámiz et al. (2016) is included. This type of theory is completely novel for the multiplicative bias corrected hazard estimators. Theory on best one-sided cross-validation introduced in this paper is of course also new for the local linear hazard estimator.

Acknowledgments

The authors are grateful for constructive comments from an anonymous Reviewer and the Associate Editor. This work has been partially supported by the Spanish Ministry of Economy and Competitiveness, through grant number MTM2016-76969P, which include support from the European Regional Development Fund (ERDF). The authors thank Centro de Servicios de Informática y Redes de Comunicaciones (CSIRC), University of Granada, for providing the computing time.

Appendix A. Assumptions for asymptotic theory

Assumption A1. The kernels K and L are compactly supported (i.e. the support is contained in $[-C_K, C_K]$ for some constants $C_K > 0$). The kernels are continuous on $\mathbb{R}\backslash\{0\}$ and have one-sided derivatives that are Hölder continuous on $\mathbb{R}^- = \{x : x < 0\}$ and $\mathbb{R}^+ = \{x : x > 0\}$, that is there exist constants c and d such that $|\phi(x) - \phi(y)| \leq c|x - y|^d$ for $x, y < 0$ or $x, y > 0$ with ϕ equal to K' or L'. The left- and right-sided derivatives differ at most on a finite set. The kernel K is symmetric.

Assumption A2. For the expected exposure function $\gamma(t) = n^{-1}E\{Y(t)\}$ it holds that $\gamma \in C_2([0,T])$, that it is strictly positive for $t \in [0,T]$, and that

$$\begin{aligned}\sup_{s\in[0,T]} |Y(s)/n - \gamma(s)| &= o_P\left\{(\log n)^{-1}\right\},\\ \sup_{s,t\in[0,T],|t-s|\leq C_K b} |\{Y(t) - Y(s)\}/n - \{\gamma(t) - \gamma(s)\}| &= o_P\left\{(nb\log n)^{-1/2}\right\},\end{aligned}$$

where the constant C_K is defined in assumption A1.

Assumption A2'. Same conditions as in assumption A2 but replacing $o_P\left\{(nb\log n)^{-1/2}\right\}$ with $o_P\left\{(nb)^{-1}\right\}$

Assumption A3. It holds that $\alpha \in C_2([0,T])$, $w \in C_1([0,T])$. The second derivative of α is Hölder continuous with exponent $d > 0$.

Assumption A3'. Same conditions as in assumption A3 but $\alpha \in C_4([0,T])$ and its fourth derivative is Hölder continuous.

Appendix B. Evaluation of the common variance terms

In Section 5 we compare bandwidth selectors by their asymptotic variances which are of the form $S_2 + S_1\Psi$, where Ψ is a factor that differs among bandwidth selectors, while the terms S_1 and S_2 are common for all of them. For both the local linear and the multiplicative biased corrected hazards, the factor Ψ only depends on the chosen kernels so we have evaluated it for some common choices in Table 1. The terms terms S_1 and S_2 however depend on the hazard function α, the exposure function γ and the weighting function w. Here we evaluate the ratio S_1/S_2 for the two hazard estimators, considering a specific choice for these functions. We consider a hazard function of the form $\alpha(t) = \lambda + c\exp(\beta t)$, where λ, c and β are constants. This hazard specification characterizes the Gompertz-Makeham law of mortality, where the empirical magnitudes for the parameters β and c are about 0.085 and 3×10^{31}, respectively. For the weighting function we consider the case $w(t) \equiv 1$, and for the exposure function the case $\gamma(t) = 1_{\{0\leq t\leq T\}}$, for $T > 0$.

B.1. Local linear estimator

For the local linear hazard estimators the terms S_1 and S_2 are given by

$$S_1^{\mathrm{LL}} = \frac{1}{25}\frac{R(K)^{-7/5}\int \alpha^2(t)w^2(t)\,dt}{\mu_2(K)^{6/5}\left\{\int \alpha''(t)^2\gamma(t)w(t)\,dt\right\}^{3/5}\left\{\int \alpha(t)w(t)\,dt\right\}^{7/5}},$$

$$S_2^{\mathrm{LL}} = \frac{4}{25}\frac{R(K)^{-2/5}\int \alpha''(t)^2\gamma(t)w^2(t)\alpha(t)\,dt}{\mu_2(K)^{6/5}\left\{\int \alpha(t)w(t)\,dt\right\}^{2/5}\left\{\int \alpha''(t)^2\gamma(t)w(t)\,dt\right\}^{8/5}},$$

The ratio $R^{\mathrm{LL}} = S_1^{\mathrm{LL}}/S_2^{\mathrm{LL}}$, for $\gamma(t) = 1_{\{0\leq t\leq T\}}$ and $w(t) \equiv 1$, is given by

$$R^{\mathrm{LL}} = \frac{S_1^{\mathrm{LL}}}{S_2^{\mathrm{LL}}} = \frac{1}{4R(K)}\frac{\int_0^T \alpha^2(t)\,dt\int_0^T \alpha''(t)^2\,dt}{\int_0^T \alpha(t)\,dt\int_0^T \alpha''(t)^2\alpha(t)\,dt}$$

For the choice $\alpha(t) = \lambda + c\exp(\beta t)$ the above integrals become

$$\begin{aligned}\int_0^T \alpha(t)\,dt &= \frac{c}{\beta}\left(\exp(\beta T) - 1\right) + \lambda T\\ \int_0^T \alpha(t)^2\,dt &= \frac{c^2}{2\beta}\left(\exp(2\beta T) - 1\right) + \lambda^2 T + \frac{2\lambda c}{\beta}\left(\exp(\beta T) - 1\right)\end{aligned}$$

$$\int_0^T \alpha''(t)^2\ dt = \frac{c^2\beta^3}{2}\left(\exp(2\beta T)-1\right)$$
$$\int_0^T \alpha''(t)^2\alpha(t)\ dt = \frac{c^3\beta^3}{3}\left(\exp(3\beta T)-1\right)+\frac{\lambda c^2\beta^3}{2}\left(\exp(2\beta T)-1\right)$$

We substitute these results in the expression of R^{LL} and take limits for $T\to\infty$. We only look at the leading terms in the numerator and the denominator (that is $\exp(4\beta T)$) and we get that $R_{LL}\to 3/(16R(K))$, as $T\to\infty$. For the Epanechnikov kernels the limit is $5/16$. Notice that the limit at zero is $R_{LL}\to(4R(K))^{-1}$, which takes the value $5/12$ for the Epanechnikov kernel.

B.2. Multiplicative bias corrected estimator

For the multiplicative bias corrected estimator the terms are

$$S_1^{\mathrm{MBC}} = \frac{2^{1/3}}{9^2}\frac{R\left(\Gamma_K\right)^{-5/6}\int\alpha(t)^2w(t)^2\ dt}{\mu_2(K)^{4/3}\left\{\int h(t)^2\gamma(t)w(t)\ dt\right\}^{1/3}\left\{\int\alpha(t)w(t)\ dt\right\}^{5/3}},$$

$$S_2^{\mathrm{MBC}} = \frac{2^{10/3}}{9^2}\frac{R\left(\Gamma_K\right)^{-2/3}\int h(t)^2\gamma(t)w(t)^2\alpha(t)\ dt}{\mu_2(K)^{4/3}\left\{\int\alpha(t)w(t)\ dt\right\}^{2/3}\left\{\int h(t)^2\gamma(t)w(t)\ dt\right\}^{4/3}},$$

where $h(t)=\alpha(t)(\alpha''(t)/\alpha(t))''$.

We compute the ratio $R^{\mathrm{MBC}}=S_1^{\mathrm{MBC}}/S_2^{\mathrm{MBC}}$ for the same choice of γ and w as before. It yields to the following expression

$$R^{\mathrm{MBC}}=\frac{S_1^{\mathrm{MBC}}}{S_2^{\mathrm{MBC}}} = \frac{1}{8R(\Gamma_K)^{1/6}}\frac{\int_0^T\alpha^2(t)\ dt\int_0^T h(t)^2\ dt}{\int_0^T\alpha(t)\ dt\int_0^T h(t)^2\alpha(t)\ dt}$$

For the choice $\alpha(t)=\lambda+c\exp(\beta t)$ the calculations are as follows:

$$\int_0^T\alpha(t)\ dt = \frac{c}{\beta}\left(\exp(\beta T)-1\right)+\lambda T$$
$$\int_0^T\alpha(t)^2\ dt = \frac{c^2}{2\beta}\left(\exp(2\beta T)-1\right)+\lambda^2T+\frac{2\lambda c}{\beta}\left(\exp(\beta T)-1\right)$$

$$\begin{aligned}\int_0^T h^2(t)dt &= \int_{\lambda+c}^{\lambda+c\exp(\beta T)} c\beta^7\lambda^2\frac{e^{\beta t}}{y^4}(2\lambda-y)^2dy=\int_{\lambda+c}^{\lambda+c\exp(\beta T)}\beta^7\lambda\frac{y-\lambda}{y^4}(2\lambda-y)^2dy\\ &= \int_{\lambda+c}^{\lambda+c\exp(\beta T)}\beta^7\lambda^2\left(\frac{8}{y^3}\lambda^2-\frac{4}{y^4}\lambda^3-\frac{5}{y^2}\lambda+\frac{1}{y}\right)dy\\ &= \beta^7\lambda^2\left[\frac{4\lambda^3}{3y^3}-\frac{4\lambda^2}{y^2}+5\frac{\lambda}{y}+\ln y\right]_{y=\lambda+c}^{y=\lambda+c\exp(\beta T)}.\end{aligned}$$

Here

$$h(t) = \alpha(t)\left(\frac{d^2}{dt^2}\left(\alpha''(t)/\alpha(t)\right)\right)=c\beta^4\lambda\frac{e^{t\beta}}{\left(\lambda+ce^{t\beta}\right)^2}\left(\lambda-ce^{t\beta}\right),$$

and we have made the change of variable $y=\alpha(t)$, $dy=c\beta e^{\beta t}dt=\beta(y-\lambda)\ dt$. Similarly

$$\begin{aligned}\int_0^T h^2(t)\alpha(t)\ dt &= \int_{\lambda+c}^{\lambda+c\exp(\beta T)} c\beta^7\lambda^2\frac{e^{\beta t}}{y^3}(2\lambda-y)^2dy=\int_{\lambda+c}^{\lambda+c\exp(\beta T)}\beta^7\lambda\frac{y-\lambda}{y^3}(2\lambda-y)^2dy\\ &= \int_{\lambda+c}^{\lambda+c\exp(\beta T)}\beta^7\lambda^2\left(\frac{8}{y^2}\lambda^2-\frac{4}{y^3}\lambda^3-\frac{5}{y}\lambda+1\right)dy\\ &= \beta^7\lambda^2\left[\frac{2\lambda^3}{y^2}-\frac{8\lambda^2}{y}-5\lambda\ln(y)+y\right]_{y=\lambda+c}^{y=\lambda+c\exp(\beta T)}\end{aligned}$$

We then substitute the above results in the expression of R^{MBC} and take limits for $T\to\infty$. To this goal we only look at the leading terms in the numerator and the denominator and we get that $R_{MBC}\to\infty$ as $T\to\infty$. And the ratio increases to ∞ as $\log(\lambda+c\exp(\beta T))$, this is, at the linear rate βT. The limit for $T\to 0$ is $(8R(\Gamma_K))^{-1/6}$, which for the Epanechnikov kernel is about 0.13.

References

Aalen, O. O. (1978). Non-parametric inference for a family of counting processes. *Ann. Statist.*, **6**, 701–726.

Andersen, P., Borgan, O., Gill, R. and Keiding, N. (1993). *Statistical Models Based on Counting Processes*. New York: Springer.

Gámiz, M. L., Mammen, E., Martínez-Miranda, M. D. and Nielsen, J. P. (2016). Double one-sided cross-validation of local linear hazards. *J. Royal Statist. Soc. B*, **78**, 755–779.

Gámiz, M. L., Mammen, E., Martínez-Miranda, M. D. and Nielsen, J. P. (2017). `DOvalidation`: *Kernel Hazard Estimation with Best One-Sided and Double One-Sided Cross-Validation.* R package version 1.1.0.

Gámiz, M. L., Martínez-Miranda, M. D. and Nielsen, J. P. (2013a). Smoothing survival densities in practice. *Comput. Statist. Data Anal.*, **58**, 368–382.

Gámiz, M. L., Janys, L., Martínez-Miranda, M. D. and Nielsen, J. P. (2013b). Bandwidth selection in marker dependent kernel hazard estimation. *Comput. Statist. Data Anal.*, **68**, 155–169.

Hall, P. and Johnstone, I. (1992). Empirical Functionals and Efficient Smoothing Parameter Selection. *J. Royal Statist. Soc. B*, **54**(2), 475–530.

Hall, P. and Marron, J. S. (1987). Extent to which Least-Squares Cross-Validation Minimises Integrated Square Error in Nonparametric Density Estimation. *Probab. Theory Rel.*, **74**, 567–581.

Hart, J. D. and Yi, S. (1998) One-Sided Cross-Validation. *J. Am. Statist. Ass.*, **93**, 620–631.

Hiabu, M., Mammen, E., Martínez-Miranda, M. D. and Nielsen, J. P. (2016) In-sample forecasting with local linear survival densities. *Biometrika*, **103**, 843–859.

Jones, M. C., Linton, O. B. and Nielsen, J. P. (1995). A simple bias reduction method for density estimation. *Biometrika*, **82**, 327–38.

Jones, M. C. and Signorini, D. F. (1997). A comparison of higher-order bias kernel density estimators. *J. Am. Statist. Ass.*, **92**, 1063–1073.

Kapotufe, S. and Verma, N. (2017). Time-Accuracy Tradeoffs in Kernel Prediction: Controlling Prediction Quality. *J. Mach. Learn. Res.*, **18**, 1–29.

Mammen, E., Martínez-Miranda, M. D., Nielsen, J. P. and Sperlich, S. (2011). Do-validation for kernel density estimation. *J. Am. Statist. Ass.*, **106**, 651–660.

Mammen, E., Martínez-Miranda, M. D., Nielsen, J. P. and Sperlich, S. (2014). Further theoretical and practical insight to the do-validated bandwidth selector. *J. Korean Statist. Soc.*, **43**, 355–365.

Mammen, E. and Nielsen, J. P. (2007). A general approach to the predictability issue in survival analysis with applications. *Biometrika*, **94**, 873–892.

Martínez-Miranda, M. D., Nielsen, J. P., Sperlich, S. and Verrall, R. (2013). Continuous Chain Ladder: Reformulating and generalizing a classical insurance problem. *Expert Syst. Appl.*, **40**(14), 5588–5603.

Muñoz, I. D. and van der Laan, M. J. (2012). Super Learner Based Conditional Density Estimation with Application to Marginal Structural Models. *Int. J. Biostat.*, **7**(1), article 38.

Nielsen, J. P. (1998). Multiplicative bias correction in kernel hazard estimation. *Scand. J. Statist.*, **25**, 541–553.

Nielsen, J. P. and Tanggaard, C. (2001). Boundary and bias correction in kernel hazard estimation. *Scand. J. Statist.*, **28**, 675–698.

Savchuk, O. Y., Hart, J. D., Sheather S. J. (2010). Indirect crossvalidation for density estimation. *J. Am. Statist. Ass.*, **105**, 415–423.

Sheather, S. J. and Jones, M. C. (1991). A reliable data-based bandwidth selection method for kernel density estimation. *J. Royal Statist. Soc. B*, **53**, 683–690.

van der Laan, M. J., Polley, E. C. and Hubbard, A. E. (2007). Super Learner. Statistical Applications of Genetics and Molecular Biology, **6**(25), article 25.

Journal of Machine Learning Research 20 (2019) 1-5 Submitted 12/17; Revised 11/18; Published 2/19

spark-crowd: A Spark Package for Learning from Crowdsourced Big Data

Enrique G. Rodrigo ENRIQUE.GRODRIGO@UCLM.ES

Juan A. Aledo JUANANGEL.ALEDO@UCLM.ES

José A. Gámez JOSE.GAMEZ@UCLM.ES
University of Castilla-La Mancha
Escuela Superior de Ingeniería Informática de Albacete
Avenida de España s/n
Albacete (02071), Spain

Editor: Geoff Holmes

Abstract

As the data sets increase in size, the process of manually labeling data becomes unfeasible by small groups of experts. Thus, it is common to rely on crowdsourcing platforms which provide inexpensive, but noisy, labels. Although implementations of algorithms to tackle this problem exist, none of them focus on scalability, limiting the area of application to relatively small data sets. In this paper, we present `spark-crowd`, an Apache Spark package for learning from crowdsourced data with scalability in mind.

Keywords: crowdsourcing, crowdsourced data, learning from crowds, big data, multiple noisy labeling, spark

1. Introduction

With the recent appearance of crowdsourcing platforms such as Amazon Mechanical Turk, several researchers have expressed their interest in taking advantage of them to increase the efficiency and scope of their research (Snow et al. 2008). Thanks to these platforms, problems that would be too expensive to tackle using traditional methods have now become easier and, at the same time, tasks which were not feasible become tractable.

However, the use of crowdsourcing in the field of Machine Learning is not without problems. The most important of them is the considerable uncertainty associated with members of the group providing the annotations in the process of data acquisition. Some of them may be better qualified for the task than others; some may give useless information or even try to ruin the results. After this process, the practitioner receives a data set of tuples $(example, annotator, annotation)$. With these, the problem usually consists in estimating the ground truth for each instance using the annotations obtained. As the annotator's quality is unknown, it is also common to want an estimation of the quality of each annotator. Thus, to obtain good results with crowdsourced data, the machine learning process needs to take into account all these difficulties (Zhang et al. 2016).

©2019 Enrique G. Rodrigo, Juan A. Aledo and Jose A. Gamez.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v20/17-743.html`.

These problems are exacerbated with large data sets as the cost constraints for the project compels to use as few annotations as possible so that the majority of examples can be labeled. For this reason, the practitioner usually receives a reduced number of annotations per instance, invalidating the use of simple aggregation methods and making the estimation of which annotator to trust crucial. Moreover, as the data increases, some other problems for data practitioners also appear, as processing the data sets in a limited amount of time becomes much more difficult (Ouyang et al. 2016). Recently, platforms such as Apache Hadoop and Apache Spark can tackle these problems, but they are not used in the available implementations of the algorithms capable of learning accurately from crowdsourced data.

In this paper, we present `spark-crowd`[1], an *Apache Spark* package for tackling the problem of learning from crowds in domains where scalability is essential. The following sections will give an overview of the project information and methods, compare it with other packages implementing similar techniques, and comment the future plans of the developers of this package.

2. Project Management

This project makes heavy use of `git` and `Github` to make the collaboration easier, as well as a means for issue tracking, code integration, and idea discussions. To assure the quality of the package, we use a continuous integration system (`CircleCI`) so that the contributions of any developer can be checked automatically to facilitate the integration of new code. Apart from this, we also ensure code quality through a set of unit tests (testing the functionality of all the methods), which provides a coverage of 99% of the package in its version 0.2.1.

The documentation for the project is provided using `sphinx` through Github Pages[2]. This documentation includes installation instructions as well as examples to start using the package. The users can also find a description of the methods as well as a comparison with similar packages. We also offer a Docker Image with the package preinstalled, so that testing the algorithms is as comfortable as possible in the platforms Docker is supported (Windows, Mac and Linux based systems, among others). Moreover, all the dependencies, as well as the package itself, are available under open source licenses so that they can be used in a wide variety of scenarios.

The package is also published in `Maven Central` and `Spark Packages` which makes adding the library as a dependency easier (both as a project dependency or directly in the Spark Shell). It is also published in the `mloss` site, to facilitate dissemination.

3. Implementation Description

This package implements 12 methods for dealing with crowdsourced data, as shown in Table 1. We provide algorithms for both discrete class (binary and multiclass) and continuous target variables. Except for the `MajorityVoting` methods (that includes the mode for discrete classes and the mean for continuous target variables), all of them return an estimation of the annotator's quality. Only the Raykar's algorithms (divided into the methods

1. The package is available in `https://github.com/enriquegrodrigo/spark-crowd`.
2. You can find the documentation of the package at https://enriquegrodrigo.github.io/spark-crowd/.

Method	Binary	Multiclass	Continuous	Reference
Majority Voting	X	X	X	Sheng (2011)
DawidSkene	X	X		Dawid and Skene (1979)
IBCC	X	X		Kim and Ghahramani (2012)
GLAD	X			Whitehill et al. (2009)
CGLAD	X			Rodrigo et al. (2018)
Raykar	X	X	X	Raykar et al. (2010)
CATD			X	Li et al. (2014a)
PM			X	Li et al. (2014b)
PMTI			X	Zheng et al. (2017)

Table 1: Methods implemented in `spark-crowd`

RaykarBinary, RaykarMulti, and RaykarCont) are able to take advantage of feature vectors while the others just use information about the annotations. For a more detailed description of the implemented algorithms, the reader may refer to the references in Table 1. In addition to the algorithms, the package also provides classes to manage annotation types as well as result objects.

An example of the library usage can be found in Listing 1. Although this code seems simple, it can be executed both locally or in a distributed environment (using the Apache Spark platform) without any modifications. The user may refer to the documentation in order to find more involved examples.

```
import com.enriquegrodrigo.spark.crowd.methods.DawidSkene
import com.enriquegrodrigo.spark.crowd.types._
//Loading file (any spark compatible format)
val data = spark.read.parquet("datafile.parquet").as[MulticlassAnnotation]
//Applying algorithm (data with columns [example, annotator, value])
val mode = DawidSkene(data.as[MulticlassAnnotation])
//Get MulticlassLabel with the class predictions
val pred = mode.getMu().as[MulticlassLabel]
//Annotator precision matrices
val annprec = mode.getAnnotatorPrecision()
```

Listing 1: Example of spark-crowd usage

4. Related Software

The most relevant software packages related to the one presented in this document are `Ceka` (Zhang et al. 2015) and the methods implemented in Zheng et al. 2017 (referred here as `Truth-Inf`). Both of them contain several algorithms for learning from crowdsourced data. The reader can find a full comparison study between these packages in the `spark-crowd` documentation, at the *Comparison with other packages* section. Here, however, we include a comparison for those methods considered in all three packages, i.e. MajorityVoting and DawidSkene[3]. We used the same environment for $\mathbf{t_1}$ (execution using only one core). For

3. The packages also implement GLAD and Raykar's algorithms. However, in `Ceka` these algorithms are implemented using wrappers to other libraries (the library for the GLAD algorithm is not available in our platform as it is given as an EXE Windows file, and the wrapper for Raykar's algorithms does not admit any configuration parameters).

	Majority Voting							DawidSkene						
	Ceka		Truth-Inf		spark-crowd			Ceka		Truth-Inf		spark-crowd		
Data set	**Acc**	$\mathbf{t_1}$	**Acc**	$\mathbf{t_1}$	**Acc**	$\mathbf{t_1}$	$\mathbf{t_c}$	**Acc**	$\mathbf{t_1}$	**Acc**	$\mathbf{t_1}$	**Acc**	$\mathbf{t_1}$	$\mathbf{t_c}$
binary1	0.931	21	0.931	1	0.931	11	7	0.994	57	0.994	12	0.994	31	32
binary2	0.936	15973	0.936	8	0.936	11	7	0.994	49259	0.994	161	0.994	60	51
binary3	X	X	0.936	112	0.936	21	8	X	X	0.994	1705	0.994	111	69
binary4	X	X	0.936	2908	0.936	57	37	X	X	M	M	0.904	703	426

Table 2: Results for accuracy and execution time on the test data sets

$\mathbf{t_c}$ we used an Apache Spark cluster with 3 executor nodes of 10 cores and 30Gb of memory each. The same code was executed on both platforms. This is, actually, one of the advantages of the library presented here. In Table 2, we show the accuracy and execution time (in seconds) obtained for four data sets of increasing size by the three libraries (the documentation of the package contains the details for these data sets). Regarding accuracy, all the packages achieved comparable results, as should be expected. Regarding the execution time, the results obtained by our package are significantly better, especially in the last two data sets. Against `Ceka`, our implementation obtained a slight speedup even in the smallest data set for both algorithms. For the second data set, `spark-crowd` obtained speedups of several orders of magnitude. It was not possible to obtain results for the last two instances in a reasonable amount of time with `Ceka`. However, `spark-crowd` solved these cases in few minutes. `Truth-Inf` obtained better results in terms of execution time in comparison to Ceka. As the algorithms chosen in this comparison are quite simple in terms of time complexity, the benefits of parallelization are less apparent, especially in the first two instances for the MajorityVoting algorithm, where the implementation in `Truth-Inf` obtained better results in terms of execution time. However, as the data and the complexity of the algorithm increases, the cost of parallelization is less noticeable. In the MajorityVoting algorithm one can see good results in the last two data sets. For the DawidSkene algorithm, the benefits of parallelization can be seen from the second data set, with a significant speedup especially in the third. `Truth-Inf` was not able to complete the execution for the last data set, due to a memory error.

5. Future Plans and Conclusions

As this is a field of recent interest, new algorithms may appear. Thus, it is in our plans to add the most interesting algorithms to the package. We also expect to develop new features and algorithms that scale even better for this problem using this library as the base of further developments. Finally, we hope that the community contributes with new implementations, which we will be delighted to integrate into the library.

Acknowledgments

This work has been partially funded by FEDER funds, the Spanish Research Agency (MINECO) and the Junta de Comunidades de Castilla-La Mancha (JCCM) through projects TIN2013-46638-C3-3-P, TIN2016-77902-C3-1-P and SBPLY/17/18050/000493. Enrique G. Rodrigo has also been funded by the FPU scholarship FPU15/02281 by MECD.

References

Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied statistics*, pages 20–28, 1979.

Hyun-Chul Kim and Zoubin Ghahramani. Bayesian classifier combination. In *AISTATS*, volume 22, pages 619–627, La Palma, Canary Islands, 2012.

Qi Li, Yaliang Li, Jing Gao, Lu Su, Bo Zhao, Murat Demirbas, Wei Fan, and Jiawei Han. A confidence-aware approach for truth discovery on long-tail data. *Proceedings of the VLDB Endowment*, 8(4):425–436, 2014a.

Qi Li, Yaliang Li, Jing Gao, Bo Zhao, Wei Fan, and Jiawei Han. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In *ACM SIGMOD*, pages 1187–1198, New York, NY, USA, 2014b.

Robin Wentao Ouyang, Mani Srivastava, Alice Toniolo, and Timothy J Norman. Truth discovery in crowdsourced detection of spatial events. *IEEE Transactions on Knowledge and Data Engineering*, 28(4):1047–1060, 2016.

Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of Machine Learning Research*, 11(Apr):1297–1322, 2010.

Enrique G. Rodrigo, Juan A. Aledo, and Jose A. Gamez. CGLAD: Using GLAD in crowdsourced large datasets. *In Lecture Notes in Computer Science, vol 11314 (IDEAL 2018)*, pages 783–791, 2018.

Victor S Sheng. Simple multiple noisy label utilization strategies. In *IEEE ICDM*, pages 635–644, 2011.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *EMNLP*, pages 254–263, 2008.

Jacob Whitehill, Ting fan Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *NIPS*, pages 2035–2043. 2009.

Jing Zhang, Victor S Sheng, Bryce A Nicholson, and Xindong Wu. Ceka: a tool for mining the wisdom of crowds. *Journal of Machine Learning Research*, 16(1):2853–2858, 2015.

Jing Zhang, Xindong Wu, and Victor S Sheng. Learning from crowdsourced labeled data: a survey. *Artificial Intelligence Review*, 46(4):543–576, 2016.

Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: is the problem solved? *Proceedings of the VLDB Endowment*, 10(5): 541–552, 2017.

Journal of Machine Learning Research 20 (2019) 1-33 Submitted 1/18; Revised 11/18; Published 2/19

Accelerated Alternating Projections for Robust Principal Component Analysis

HanQin Cai HQCAI@MATH.UCLA.EDU
Department of Mathematics
University of California, Los Angeles
Los Angeles, California, USA

Jian-Feng Cai JFCAI@UST.HK
Department of Mathematics
Hong Kong University of Science and Technology
Hong Kong SAR, China

Ke Wei KEWEI@FUDAN.EDU.CN
School of Data Science
Fudan University
Shanghai, China

Editor: Sujay Sanghavi

Abstract

We study robust PCA for the fully observed setting, which is about separating a low rank matrix $\boldsymbol{L}$ and a sparse matrix $\boldsymbol{S}$ from their sum $\boldsymbol{D} = \boldsymbol{L} + \boldsymbol{S}$. In this paper, a new algorithm, dubbed accelerated alternating projections, is introduced for robust PCA which significantly improves the computational efficiency of the existing alternating projections proposed in (Netrapalli et al., 2014) when updating the low rank factor. The acceleration is achieved by first projecting a matrix onto some low dimensional subspace before obtaining a new estimate of the low rank matrix via truncated SVD. Exact recovery guarantee has been established which shows linear convergence of the proposed algorithm. Empirical performance evaluations establish the advantage of our algorithm over other state-of-the-art algorithms for robust PCA.

Keywords: Robust PCA, Alternating Projections, Matrix Manifold, Tangent Space, Subspace Projection

1. Introduction

Robust principal component analysis (RPCA) appears in a wide range of applications, including video and voice background subtraction (Li et al., 2004; Huang et al., 2012), sparse graphs clustering (Chen et al., 2012), 3D reconstruction (Mobahi et al., 2011), and fault isolation (Tharrault et al., 2008). Suppose we are given a sum of a low rank matrix and a sparse matrix, denoted $\boldsymbol{D} = \boldsymbol{L} + \boldsymbol{S}$. The goal of RPCA is to reconstruct $\boldsymbol{L}$ and $\boldsymbol{S}$ simultaneously from $\boldsymbol{D}$. As a concrete example, for foreground-background separation in video processing, $\boldsymbol{L}$ represents static background through all the frames of a video which should be low rank while $\boldsymbol{S}$ represents moving objects which can be assumed to be sparse since typically they will not block a large portion of the screen for a long time.

©2019 HanQin Cai, Jian-Feng Cai and Ke Wei.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-022.html.

RPCA can be achieved by seeking a low rank matrix $\boldsymbol{L}'$ and a sparse matrix $\boldsymbol{S}'$ such that their sum fits the measurement matrix $\boldsymbol{D}$ as well as possible:

$$\min_{\boldsymbol{L}',\boldsymbol{S}'\in\mathbb{R}^{m\times n}} \|\boldsymbol{D}-\boldsymbol{L}'-\boldsymbol{S}'\|_F \quad \text{subject to } rank(\boldsymbol{L}')\le r \text{ and } \|\boldsymbol{S}'\|_0\le|\Omega|, \tag{1}$$

where r denotes the rank of the underlying low rank matrix, Ω denotes the support set of the underlying sparse matrix, and $\|\boldsymbol{S}'\|_0$ counts the number of non-zero entries in $\boldsymbol{S}'$. Compared to the traditional principal component analysis (PCA) which computes a low rank approximation of a data matrix, RPCA is less sensitive to outliers since it includes a sparse part in the formulation.

Since the seminal works of (Wright et al., 2009; Candès et al., 2011; Chandrasekaran et al., 2011), RPCA has received intensive investigations both from theoretical and algorithmic aspects. Noticing that (1) is a non-convex problem, some of the earlier works focus on the following convex relaxation of RPCA:

$$\min_{\boldsymbol{L}',\boldsymbol{S}'\in\mathbb{R}^{m\times n}} \|\boldsymbol{L}'\|_*+\lambda\|\boldsymbol{S}'\|_1 \quad \text{subject to } \boldsymbol{L}'+\boldsymbol{S}'=\boldsymbol{D}, \tag{2}$$

where $\|\cdot\|_*$ is the nuclear norm (*viz.* trace norm) of matrices, λ is the regularization parameter, and $\|\cdot\|_1$ denotes the ℓ_1-norm of the vectors obtained by stacking the columns of associated matrices. Under some mild conditions, it has been proven that the RPCA problem can be solved exactly by the aforementioned convex relaxation Candès et al. (2011); Chandrasekaran et al. (2011). However, a limitation of the convex relaxation based approach is that the resulting semidefinite programming is computationally rather expensive to solve, even for medium size matrices. Alternative to the convex relaxation, many non-convex algorithms have been designed to target (1) directly. This line of research will be reviewed in more detail in Section 2.3 after our approach has been introduced.

This paper targets the non-convex optimization for RPCA directly. The main contributions of this work are two-fold. Firstly, we propose a new algorithm, accelerated alternating projections (AccAltProj), for RPCA, which is substantially faster than other state-of-the-art algorithms. Secondly, exact recovery of accelerated alternating projections has been established for the fixed sparsity model, where we assume the ratio of the number of non-zero entries in each row and column of $\boldsymbol{S}$ is less than a threshold.

1.1. Assumptions

It is clear that the RPCA problem is ill-posed without any additional conditions. Common assumptions are that $\boldsymbol{L}$ cannot be too sparse and $\boldsymbol{S}$ cannot be locally too dense, which are formalized in A1 and A2, respectively.

A1 *The underlying low rank matrix $\boldsymbol{L}\in\mathbb{R}^{m\times n}$ is a rank-r matrix with μ-incoherence, that is*

$$\max_i \|\boldsymbol{e}_i^T\boldsymbol{U}\|_2\le\sqrt{\frac{\mu r}{m}}, \quad \text{and} \quad \max_j \|\boldsymbol{e}_j^T\boldsymbol{V}\|_2\le\sqrt{\frac{\mu r}{n}}$$

hold for a positive numerical constant $1\le\mu\le\frac{\min\{m,n\}}{r}$, where $\boldsymbol{L}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ is the SVD of $\boldsymbol{L}$.

Assumption A1 was first introduced in (Candès and Recht, 2009) for low rank matrix completion, and now it is a very standard assumption for related low rank reconstruction problems. It basically states that the left and right singular vectors of $\boldsymbol{L}$ are weakly correlated with the canonical basis, which implies $\boldsymbol{L}$ cannot be a very sparse matrix.

A2 *The underlying sparse matrix $\boldsymbol{S} \in \mathbb{R}^{m\times n}$ is α-sparse. That is, $\boldsymbol{S}$ has at most αn non-zero entries in each row, and at most αm non-zero entries in each column. In the other words, for all $1 \le i \le m, 1 \le j \le n$,*

$$\|\boldsymbol{e}_i^T \boldsymbol{S}\|_0 \le \alpha n \quad \textit{and} \quad \|\boldsymbol{S}\boldsymbol{e}_j\|_0 \le \alpha m. \tag{3}$$

In this paper, we assume[1]

$$\alpha \lesssim \min\left\{\frac{1}{\mu r^2\kappa^3}, \frac{1}{\mu^{1.5} r^2 \kappa}, \frac{1}{\mu^2 r^2}\right\}, \tag{4}$$

where κ is the condition number of $\boldsymbol{L}$.

Assumption A2 states that the non-zero entries of the sparse matrix $\boldsymbol{S}$ cannot concentrate in a few rows or columns, so there does not exist a low rank component in $\boldsymbol{S}$. If the indices of the support set Ω are sampled independently from the Bernoulli distribution with the associated parameter being slightly smaller than α, by the Chernoff inequality, one can easily show that (3) holds with high probability.

1.2. Organization and Notation of the Paper

The rest of the paper is organized as follows. In the remainder of this section, we introduce standard notation that is used throughout the paper. Section 2.1 presents the proposed algorithm and discusses how to implement it efficiently. The theoretical recovery guarantee of the proposed algorithm is presented in Section 2.2, followed by a review of prior art for RPCA. In Section 3, we present the numerical simulations of our algorithm. Section 4 contains all the mathematical proofs of our main theoretical result. We conclude this paper with future directions in Section 5.

In this paper, vectors are denoted by bold lowercase letters (e.g., $\boldsymbol{x}$), matrices are denoted by bold capital letters (e.g., $\boldsymbol{X}$), and operators are denoted by calligraphic letters (e.g., $\mathcal{H}$). In particular, $\boldsymbol{e}_i$ denotes the i^{th} canonical basis vector, $\boldsymbol{I}$ denotes the identity matrix, and $\mathcal{I}$ denotes the identity operator. For a vector $\boldsymbol{x}$, $\|\boldsymbol{x}\|_0$ counts the number of non-zero entries in $\boldsymbol{x}$, and $\|\boldsymbol{x}\|_2$ denotes the ℓ_2 norm of $\boldsymbol{x}$. For a matrix $\boldsymbol{X}$, $[\boldsymbol{X}]_{ij}$ denotes its $(i,j)^{th}$ entry, $\sigma_i(\boldsymbol{X})$ denotes its i^{th} singular value, $\|\boldsymbol{X}\|_\infty = \max_{ij} |[\boldsymbol{X}]_{ij}|$ denotes the maximum magnitude of its entries, $\|\boldsymbol{X}\|_2 = \sigma_1(\boldsymbol{X})$ denotes its spectral norm, $\|\boldsymbol{X}\|_F = \sqrt{\sum_i \sigma_i^2(\boldsymbol{X})}$ denotes its Frobenius norm, and $\|\boldsymbol{X}\|_* = \sum_i \sigma_i(\boldsymbol{X})$ denotes its nuclear norm. The inner product of two real valued vectors is defined as $\langle \boldsymbol{x}, \boldsymbol{y}\rangle = \boldsymbol{x}^T\boldsymbol{y}$, and the inner product of two real valued matrices is defined as $\langle \boldsymbol{X}, \boldsymbol{Y}\rangle = Trace(\boldsymbol{X}^T\boldsymbol{Y})$, where $(\cdot)^T$ represents the transpose of a vector or matrix.

Additionally, we sometimes use the shorthand σ_i^A to denote the i^{th} singular value of a matrix $\boldsymbol{A}$. Note that $\kappa = \sigma_1^L/\sigma_r^L$ always denotes the condition number of the underlying rank-r matrix $\boldsymbol{L}$, and $\Omega = supp(\boldsymbol{S})$ is always referred to as the support of the underlying sparse matrix $\boldsymbol{S}$. At the k^{th} iteration of the proposed algorithm, the estimates of the low rank matrix and the sparse matrix are denoted by $\boldsymbol{L}_k$ and $\boldsymbol{S}_k$, respectively.

1. The standard notion "$\lesssim$" in (4) means there exists an absolute numerical constant $C > 0$ such that α can be upper bounded by C times the right hand side.

2. Algorithm and Theoretical Results

In this section, we present the new algorithm and its recovery guarantee. For ease of exposition, we assume all matrices are square (i.e., $m = n$), but emphasize that nothing is special about this assumption and all the results can be easily extended to rectangular matrices.

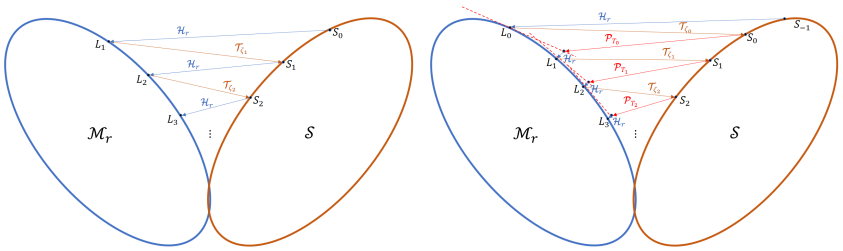
2.1. Proposed Algorithm

Alternating projections is a minimization approach that has been successfully used in many fields, including image processing (Wang et al., 2008; Chan and Wong, 2000; O'Sullivan and Benac, 2007), matrix completion (Keshavan et al., 2012; Jain et al., 2013; Hardt, 2013; Tanner and Wei, 2016), phase retrieval (Netrapalli et al., 2013; Cai et al., 2017; Zhang, 2017), and many others (Peters and Heath, 2009; Agarwal et al., 2014; Yu et al., 2016; Pu et al., 2017). A non-convex algorithm based on alternating projections, namely AltProj, is presented in (Netrapalli et al., 2014) for RPCA accompanied with a theoretical recovery guarantee. In each iteration, AltProj first updates $\boldsymbol{L}$ by projecting $\boldsymbol{D}-\boldsymbol{S}$ onto the space of rank-r matrices, denoted $\mathcal{M}_r$, and then updates $\boldsymbol{S}$ by projecting $\boldsymbol{D}-\boldsymbol{L}$ onto the space of sparse matrices, denoted $\mathcal{S}$; see the left plot of Figure 1 for an illustration. Regarding to the implementation of AltProj, the projection of a matrix onto the space of low rank matrices can be computed by the singular value decomposition (SVD) followed by truncating out small singular values, while the projection of a matrix onto the space of sparse matrices can be computed by the hard thresholding operator. As a non-convex algorithm which targets (1) directly, AltProj is computationally much more efficient than solving the convex relaxation problem (2) using semidefinite programming (SDP). However, when projecting $\boldsymbol{D}-\boldsymbol{S}$ onto the low rank matrix manifold, AltProj requires to compute the SVD of a full size matrix, which is computationally expensive. Inspired by the work in (Vandereycken, 2013; Wei et al., 2016a,b), we propose an accelerated algorithm for RPCA, coined accelerated alternating projections (AccAltProj), to circumvent the high computational cost of the SVD. The new algorithm is able to reduce the per-iteration computational cost of AltProj significantly, while a theoretical guarantee can be similarly established.

Our algorithm consists of two phases: initialization and projections onto $\mathcal{M}_r$ and $\mathcal{S}$ alternatively. We begin our discussion with the second phase, which is described in Algorithm 1. For geometric comparison between AltProj and AccAltProj, see Figure 1.

Let $(\boldsymbol{L}_k, \boldsymbol{S}_k)$ be a pair of current estimates. At the $(k+1)^{th}$ iteration, AccAltProj first trims $\boldsymbol{L}_k$ into an incoherent matrix $\widetilde{\boldsymbol{L}}_k$ using Algorithm 2. Noting that $\widetilde{\boldsymbol{L}}_k$ is still a rank-r matrix, so its left and right singular vectors define an $(2n-r)r$-dimensional subspace (Vandereycken, 2013),

$$\widetilde{T}_k = \{\widetilde{\boldsymbol{U}}_k\boldsymbol{A}^T + \boldsymbol{B}\widetilde{\boldsymbol{V}}_k^T \mid \boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times r}\}, \tag{5}$$



(i) Illustration of AltProj (ii) Illustration of AccAltProj

Figure 1: Visual comparison between AltProj and AccAltProj, where $\mathcal{M}_r$ denotes the manifold of rank-r matrices and $\mathcal{S}$ denotes the set of sparse matrices. The red dash line in (ii) represents the tangent space of $\mathcal{M}_r$ at $\boldsymbol{L}_k$. In fact, each circle represents a sum of a low rank matrix and a sparse matrix, but with the component on one circle fixed when projecting onto the other circle. For conciseness, the trim stage, i.e., $\widetilde{\boldsymbol{L}}_k$, is not included in the plot for AccAltProj.

Algorithm 1 Robust PCA by Accelerated Alternating Projections (AccAltProj)

Input: $\boldsymbol{D} = \boldsymbol{L} + \boldsymbol{S}$: matrix to be split; r: rank of $\boldsymbol{L}$; ϵ: target precision level; β: thresholding parameter; γ: target converge rate; μ: incoherence parameter of $\boldsymbol{L}$.
Initialization
$k = 0$
while $\|\boldsymbol{D} - \boldsymbol{L}_k - \boldsymbol{S}_k\|_F / \|\boldsymbol{D}\|_F \geq \epsilon$ **do**
  $\widetilde{\boldsymbol{L}}_k = \mathrm{Trim}(\boldsymbol{L}_k, \mu)$
  $\boldsymbol{L}_{k+1} = \mathcal{H}_r(\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D} - \boldsymbol{S}_k))$
  $\zeta_{k+1} = \beta\left(\sigma_{r+1}\left(\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D} - \boldsymbol{S}_k)\right) + \gamma^{k+1}\sigma_1\left(\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D} - \boldsymbol{S}_k)\right)\right)$
  $\boldsymbol{S}_{k+1} = \mathcal{T}_{\zeta_{k+1}}(\boldsymbol{D} - \boldsymbol{L}_{k+1})$
  $k = k + 1$
end while
Output: $\boldsymbol{L}_k$, $\boldsymbol{S}_k$

where $\widetilde{\boldsymbol{L}}_k = \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{\Sigma}}_k\widetilde{\boldsymbol{V}}_k^T$ is the SVD of $\widetilde{\boldsymbol{L}}_k$[2]. Given a matrix $\boldsymbol{Z} \in \mathbb{R}^{n\times n}$, it can be easily verified that the projections of $\boldsymbol{Z}$ onto the subspace $\widetilde{T}_k$ and its orthogonal complement are given by

$$\mathcal{P}_{\widetilde{T}_k}\boldsymbol{Z} = \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{Z} + \boldsymbol{Z}\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T - \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{Z}\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T \tag{6}$$

and

$$(\mathcal{I} - \mathcal{P}_{\widetilde{T}_k})\boldsymbol{Z} = (\boldsymbol{I} - \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)\boldsymbol{Z}(\boldsymbol{I} - \widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T). \tag{7}$$

2. In practice, we only need the trimmed orthogonal matrices $\widetilde{\boldsymbol{U}}_k$ and $\widetilde{\boldsymbol{V}}_k$ for the projection $\mathcal{P}_{\widetilde{T}_k}$, and they can be computed efficiently via a QR decomposition. The entire matrix $\widetilde{\boldsymbol{L}}_k$ should never be formed in an efficient implementation of AccAltProj.

Algorithm 2 Trim

Input: $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$: matrix to be trimmed; μ: target incoherence level.
$c_\mu = \sqrt{\frac{\mu r}{n}}$
for $i = 1$ `to` m **do**
  $\boldsymbol{A}^{(i)} = \min\{1, \frac{c_\mu}{\|\boldsymbol{U}^{(i)}\|}\}\boldsymbol{U}^{(i)}$
end for
for $j = 1$ `to` n **do**
  $\boldsymbol{B}^{(j)} = \min\{1, \frac{c_\mu}{\|\boldsymbol{V}^{(j)}\|}\}\boldsymbol{V}^{(i)}$
end for
Output: $\widetilde{\boldsymbol{L}} = \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{B}$

As stated previously, AltProj truncates the SVD of $\boldsymbol{D} - \boldsymbol{S}_k$ directly to get a new estimate of $\boldsymbol{L}$. In contrast, AccAltProj first projects $\boldsymbol{D} - \boldsymbol{S}_k$ onto the low dimensional subspace $\widetilde{T}_k$, and then projects the intermediate matrix onto the rank-r matrix manifold $\mathcal{M}_r$ using the truncated SVD. That is,

$$\boldsymbol{L}_{k+1} = \mathcal{H}_r(\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D} - \boldsymbol{S}_k)),$$

where $\mathcal{H}_r$ computes the best rank-r approximation of a matrix,

$$\mathcal{H}_r(\boldsymbol{Z}) := \boldsymbol{Q}\boldsymbol{\Lambda}_r\boldsymbol{P}^T \text{ where } \boldsymbol{Z} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{P}^T \text{ is its SVD and } [\boldsymbol{\Lambda}_r]_{ii} := \begin{cases} [\boldsymbol{\Lambda}]_{ii} & i \leq r \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Before proceeding, it is worth noting that the set of rank-r matrices $\mathcal{M}_r$ form a smooth manifold of dimension $(2n - r)r$, and $\widetilde{T}_k$ is indeed the tangent space of $\mathcal{M}_r$ at $\widetilde{\boldsymbol{L}}_k$ (Vandereycken, 2013). Matrix manifold algorithms based on the tangent space of low dimensional spaces have been widely studied in the literature, see for example (Ngo and Saad, 2012; Mishra et al., 2012; Vandereycken, 2013; Mishra and Sepulchre, 2014; Mishra et al., 2014; Wei et al., 2016a,b) and references therein. In particular, we invite readers to explore the book (Absil et al., 2009) for more details about the differential geometry ideas behind manifold algorithms.

One can see that a SVD is still needed to obtain the new estimate $\boldsymbol{L}_{k+1}$. Nevertheless, it can be computed in a very efficient way (Vandereycken, 2013; Wei et al., 2016a,b). Let $(\boldsymbol{I} - \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{V}}_k = \boldsymbol{Q}_1\boldsymbol{R}_1$ and $(\boldsymbol{I} - \widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{U}}_k = \boldsymbol{Q}_2\boldsymbol{R}_2$ be the QR decompositions of $(\boldsymbol{I} - \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{V}}_k$ and $(\boldsymbol{I} - \widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{U}}_k$, respectively. Note that $(\boldsymbol{I} - \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{V}}_k$ and $(\boldsymbol{I} - \widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{U}}_k$ can be computed by one matrix-matrix subtraction between an $n \times n$ matrix and an $n \times n$ matrix, two matrix-matrix multiplications between an $n \times n$ matrix and an $n \times r$ matrix, and a few matrix-matrix multiplications between a $r \times n$ and an $n \times r$ or between an $n \times r$ matrix and a $r \times r$ matrix. Moreover, A little algebra gives

$$\begin{aligned}
\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D} - \boldsymbol{S}_k) &= \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T(\boldsymbol{D} - \boldsymbol{S}_k) + (\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T - \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T \\
&= \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T(\boldsymbol{D} - \boldsymbol{S}_k)(\boldsymbol{I} - \widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T) + (\boldsymbol{I} - \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T + \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T \\
&= \widetilde{\boldsymbol{U}}_k\boldsymbol{R}_2^T\boldsymbol{Q}_2^T + \boldsymbol{Q}_1\boldsymbol{R}_1\widetilde{\boldsymbol{V}}_k^T + \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T(\boldsymbol{D} - \boldsymbol{S}_k)\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T
\end{aligned}$$

$$= \begin{bmatrix} \widetilde{\boldsymbol{U}}_k & \boldsymbol{Q}_1 \end{bmatrix} \begin{bmatrix} \widetilde{\boldsymbol{U}}_k^T(\boldsymbol{D}-\boldsymbol{S}_k)\widetilde{\boldsymbol{V}}_k & \boldsymbol{R}_2^T \\ \boldsymbol{R}_1 & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \widetilde{\boldsymbol{V}}_k^T \\ \boldsymbol{Q}_2^T \end{bmatrix}$$
$$:= \begin{bmatrix} \widetilde{\boldsymbol{U}}_k & \boldsymbol{Q}_1 \end{bmatrix} \boldsymbol{M}_k \begin{bmatrix} \widetilde{\boldsymbol{V}}_k^T \\ \boldsymbol{Q}_2^T \end{bmatrix},$$

where the fourth line follows from the fact $\widetilde{\boldsymbol{U}}_k^T\boldsymbol{Q}_1 = \widetilde{\boldsymbol{V}}_k^T\boldsymbol{Q}_2 = \boldsymbol{0}$. Let $\boldsymbol{M}_k = \boldsymbol{U}_{M_k}\boldsymbol{\Sigma}_{M_k}\boldsymbol{V}_{M_k}^T$ be the SVD of $\boldsymbol{M}_k$, which can be computed using $O(r^3)$ flops since $\boldsymbol{M}_k$ is a $2r \times 2r$ matrix. Then the SVD of $\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k) = \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{\Sigma}}_k\widetilde{\boldsymbol{V}}_k^T$ can be computed by

$$\widetilde{\boldsymbol{U}}_{k+1} = \begin{bmatrix} \widetilde{\boldsymbol{U}}_k & \boldsymbol{Q}_1 \end{bmatrix} \boldsymbol{U}_{M_k}, \quad \widetilde{\boldsymbol{\Sigma}}_{k+1} = \boldsymbol{\Sigma}_{M_k}, \quad \text{and} \quad \widetilde{\boldsymbol{V}}_{k+1} = \begin{bmatrix} \widetilde{\boldsymbol{V}}_k & \boldsymbol{Q}_2 \end{bmatrix} \boldsymbol{V}_{M_k} \tag{9}$$

since both the matrices $\begin{bmatrix} \widetilde{\boldsymbol{U}}_k & \boldsymbol{Q}_1 \end{bmatrix}$ and $\begin{bmatrix} \widetilde{\boldsymbol{V}}_k & \boldsymbol{Q}_2 \end{bmatrix}$ are orthogonal. In summary, the overall computational costs of $\mathcal{H}_r(\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k))$ lie in one matrix-matrix subtraction between an $n \times n$ matrix and an $n \times n$ matrix, two matrix-matrix multiplications between an $n \times n$ matrix and an $n \times r$ matrix, the QR decomposition of two $n \times r$ matrices, an SVD of a $2r \times 2r$ matrix, and a few matrix-matrix multiplications between a $r \times n$ matrix and an $n \times r$ matrix or between an $n \times r$ matrix and a $r \times r$ matrix, leading to a total of $4n^2r + n^2 + O(nr^2 + r^3)$ flops. Thus, the dominant per iteration computational complexity of AccAltProj for updating the estimate of $\boldsymbol{L}$ is the same as the novel gradient descent based approach introduced in (Yi et al., 2016). In contrast, computing the best rank-r approximation of a non-structured $n \times n$ matrix $\boldsymbol{D}-\boldsymbol{S}_k$ typically costs $O(n^2r) + n^2$ flops with a large hidden constant in front of n^2r.

After $\boldsymbol{L}_{k+1}$ is obtained, following the approach in (Netrapalli et al., 2014), we apply the hard thresholding operator to update the estimate of the sparse matrix,

$$\boldsymbol{S}_{k+1} = \mathcal{T}_{\zeta_{k+1}}(\boldsymbol{D}-\boldsymbol{L}_{k+1}),$$

where the thresholding operator $\mathcal{T}_{\zeta_{k+1}}$ is defined as

$$[\mathcal{T}_{\zeta_{k+1}}\boldsymbol{Z}]_{ij} = \begin{cases} [\boldsymbol{Z}]_{ij} & |[\boldsymbol{Z}]_{ij}| > \zeta_{k+1} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

for any matrix $\boldsymbol{Z} \in \mathbb{R}^{m\times n}$. Notice that the thresholding value of ζ_{k+1} in Algorithm 1 is chosen as

$$\zeta_{k+1} = \beta\left(\sigma_{r+1}\left(\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)\right) + \gamma^{k+1}\sigma_1\left(\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)\right)\right),$$

which relies on a tuning parameter $\beta > 0$, a convergence rate parameter $0 \le \gamma < 1$, and the singular values of $\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)$. Since we have already obtained all the singular values of $\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)$ when computing $\boldsymbol{L}_{k+1}$, the extra cost of computing ζ_{k+1} is very marginal. Therefore, the cost of updating the estimate of $\boldsymbol{S}$ is very low and insensitive to the sparsity of $\boldsymbol{S}$.

In this paper, a good initialization is achieved by two steps of modified AltProj when setting the input rank to r, see Algorithm 3. With this initialization scheme, we can construct an initial guess that is sufficiently close to the ground truth and is inside the "basin of attraction" as detailed in the next subsection. Note that the thresholding parameter β_{init} used in Algorithm 3 is different from that in Algorithm 1.

Algorithm 3 Initialization by Two Steps of AltProj

Input: $\boldsymbol{D} = \boldsymbol{L} + \boldsymbol{S}$: matrix to be split; r: rank of $\boldsymbol{L}$; β_{init}, β: thresholding parameters.
$\boldsymbol{L}_{-1} = \boldsymbol{0}$
$\zeta_{-1} = \beta_{init} \cdot \sigma_1^D$
$\boldsymbol{S}_{-1} = \mathcal{T}_{\zeta_{-1}}(\boldsymbol{D}-\boldsymbol{L}_{-1})$
$\boldsymbol{L}_0 = \mathcal{H}_r(\boldsymbol{D}-\boldsymbol{S}_{-1})$
$\zeta_0 = \beta \cdot \sigma_1(\boldsymbol{D}-\boldsymbol{S}_{-1})$
$\boldsymbol{S}_0 = \mathcal{T}_{\zeta_0}(\boldsymbol{D}-\boldsymbol{L}_0)$
Output: $\boldsymbol{L}_0$, $\boldsymbol{S}_0$

2.2. Theoretical Guarantee

In this subsection, we present the theoretical recovery guarantee of AccAltProj (Algorithm 1 together with Algorithm 3). The following theorem establishes the local convergence of AccAltProj.

Theorem 1 (Local Convergence of AccAltProj) *Let $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{S} \in \mathbb{R}^{n\times n}$ be two symmetric matrices satisfying Assumptions A1 and A2. If the initial guesses $\boldsymbol{L}_0$ and $\boldsymbol{S}_0$ obey the following conditions:*

$$\|\boldsymbol{L}-\boldsymbol{L}_0\|_2 \le 8\alpha\mu r\sigma_1^L, \quad \|\boldsymbol{S}-\boldsymbol{S}_0\|_\infty \le \frac{\mu r}{n}\sigma_1^L, \quad \text{and} \quad supp(\boldsymbol{S}_0) \subset \Omega,$$

then the iterates of Algorithm 1 with parameters $\beta = \frac{\mu r}{2n}$ and $\gamma \in \left(\frac{1}{\sqrt{12}}, 1\right)$ satisfy

$$\|\boldsymbol{L}-\boldsymbol{L}_k\|_2 \le 8\alpha\mu r\gamma^k\sigma_1^L, \quad \|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty \le \frac{\mu r}{n}\gamma^k\sigma_1^L, \quad \text{and} \quad supp(\boldsymbol{S}_k) \subset \Omega.$$

The next theorem states that the initial guesses obtained from Algorithm 3 fulfill the conditions required in Theorem 1.

Theorem 2 (Guaranteed Initialization) *Let $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{S} \in \mathbb{R}^{n\times n}$ be two symmetric matrices satisfying Assumptions A1 and A2, respectively. If the thresholding parameters obey $\frac{\mu r\sigma_1^L}{n\sigma_1^D} \le \beta_{init} \le \frac{3\mu r\sigma_1^L}{n\sigma_1^D}$ and $\beta = \frac{\mu r}{2n}$, then the outputs of Algorithm 3 satisfy*

$$\|\boldsymbol{L}-\boldsymbol{L}_0\|_2 \le 8\alpha\mu r\sigma_1^L, \quad \|\boldsymbol{S}-\boldsymbol{S}_0\|_\infty \le \frac{\mu r}{n}\sigma_1^L, \quad \text{and} \quad supp(\boldsymbol{S}_0) \subset \Omega.$$

The proofs of Theorems 1 and 2 are presented in Section 4. The convergence of AccAltProj follows immediately by combining the above two theorems together.

For conciseness, the main theorems are stated for symmetric matrices. However, similar results can be established for nonsymmetric matrix recovery problems as they can be cast as problems with respect to symmetric augmented matrices, as suggested in (Netrapalli et al., 2014). Without loss of generality, assume $dm \le n < (d+1)m$ for some $d \ge 1$ and construct

$\overline{\boldsymbol{L}}$ and $\overline{\boldsymbol{S}}$ as

$$\overline{\boldsymbol{L}} := \begin{bmatrix} \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{L} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{L} \\ \underbrace{\boldsymbol{L}^T \quad \cdots \quad \boldsymbol{L}^T}_{d \text{ times}} & & & \mathbf{0} \end{bmatrix} \Bigg\} d \text{ times}, \qquad \overline{\boldsymbol{S}} := \begin{bmatrix} \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{S} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{S} \\ \underbrace{\boldsymbol{S}^T \quad \cdots \quad \boldsymbol{S}^T}_{d \text{ times}} & & & \mathbf{0} \end{bmatrix} \Bigg\} d \text{ times}.$$

Then it is not hard to see that $\overline{\boldsymbol{L}}$ is $O(\mu)$-incoherent, and $\overline{\boldsymbol{S}}$ is $O(\alpha)$-sparse, with the hidden constants being independent of d. Moreover, based on the connection between the SVD of the augmented matrix and that of the original one, it can be easily verified that at the k^{th} iteration the estimates returned by AccAltProj with input $\overline{\boldsymbol{D}} = \overline{\boldsymbol{L}} + \overline{\boldsymbol{S}}$ have the form

$$\overline{\boldsymbol{L}}_k = \begin{bmatrix} \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{L}_k \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{L}_k \\ \underbrace{\boldsymbol{L}_k^T \quad \cdots \quad \boldsymbol{L}_k^T}_{d \text{ times}} & & & \mathbf{0} \end{bmatrix} \Bigg\} d \text{ times}, \qquad \overline{\boldsymbol{S}}_k = \begin{bmatrix} \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{S}_k \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{S}_k \\ \underbrace{\boldsymbol{S}_k^T \quad \cdots \quad \boldsymbol{S}_k^T}_{d \text{ times}} & & & \mathbf{0} \end{bmatrix} \Bigg\} d \text{ times},$$

where $\boldsymbol{L}_k, \boldsymbol{S}_k$ are the the k^{th} estimates returned by AccAltProj with input $\boldsymbol{D} = \boldsymbol{L} + \boldsymbol{S}$.

2.3. Related Work

As mentioned earlier, convex relaxation based methods for RPCA have higher computational complexity and slower convergence rate which are not applicable for high dimensional problems. In fact, the convergence rate of the algorithm for computing the solution to the SDP formulation of RPCA (Candès et al., 2011; Chandrasekaran et al., 2011; Xu et al., 2010) is sub-linear with the per iteration computational complexity being $O(n^3)$. By contrast, AccAltProj only requires $O(\log(1/\epsilon))$ iterations to achieve an accuracy of ϵ, and the dominant per iteration computational cost is $O(rn^2)$.

There have been many other algorithms which are designed to solve the non-convex RPCA problem directly. In (Wang et al., 2013), an alternating minimization algorithm was proposed for (1) based on the factorization model of low rank matrices. However, only convergence to fixed points was established there. In (Gu et al., 2016), the authors developed an alternating minimization algorithm for RPCA, which allows the sparsity level α to be $O(1/(\mu^{2/3}r^{2/3}n))$ for successful recovery, which is more stringent than our result when $r \ll n$. A projected gradient descent algorithm was proposed in Chen and Wainwright (2015) for the special case of positive semidefinite matrices based on the ℓ_1-norm of each row of the underlying sparse matrix, which is not very practical.

In Table 1, we compare AccAltProj with the other two competitive non-convex algorithms for RPCA: AltProj from (Netrapalli et al., 2014) and non-convex gradient descent

(GD) from (Yi et al., 2016). GD attempts to reconstruct the low rank matrix by minimizing an objective function which contains the prior knowledge of the sparse matrix. The table displays the computational complexity of each algorithm for updating the estimates of the low rank matrix and the sparse matrix, as well as the convergence rate and the theoretical tolerance for the number of non-zero entries in the sparse matrix.

From the table, we can see that AccAltProj achieves the same linear convergence rate as AltProj, which is faster than GD. Moreover, AccAltProj has the lowest per iteration computational complexity for updating both the estimates of $\boldsymbol{L}$ and $\boldsymbol{S}$ (ties with AltProj for updating the sparse part). It is worth emphasizing that the acceleration stage in AccAltProj which first projects $\boldsymbol{D} - \boldsymbol{S}_k$ onto a low dimensional subspace reduces the computational cost of the SVD in AltProj dramatically. Overall, AccAltProj will be substantially faster than AltProj and GD, as confirmed by our numerical simulations in next section. The table also shows that the theoretical sparsity level that can be tolerated by AccAltProj is lower than that of GD and AltProj. Our result looses an order in r because we have replaced the spectral norm by the Frobenius norm when considering the reduction of the reconstruction error in terms of the spectral norm. In addition, the condition number of the target matrix appears in the theoretical result because the current version of AccAltProj deals with the fixed rank case which requires the initial guess is sufficiently close to the target matrix for the theoretical analysis. Nevertheless, we note that the sufficient condition regarding to α to guarantee the exact recovery of AccAltProj is highly pessimistic when compared with its empirical performance. Numerical investigations in next section show that AccAltProj can tolerate as large α as AltProj does under different energy levels.

Table 1: Comparison of AccAltProj, AltProj and GD.

Algorithm	AccAltProj	AltProj	GD
Updating $\boldsymbol{S}$	$\boldsymbol{O}\left(\boldsymbol{n^2}\right)$	$O\left(rn^2\right)$	$O\left(n^2 + \alpha n^2 \log(\alpha n)\right)$
Updating $\boldsymbol{L}$	$\boldsymbol{O}\left(\boldsymbol{rn^2}\right)$	$O\left(r^2n^2\right)$	$\boldsymbol{O}\left(\boldsymbol{rn^2}\right)$
Tolerance of α	$O\left(\frac{1}{\max\{\mu r^2\kappa^3, \mu^{1.5}r^2\kappa, \mu^2r^2\}}\right)$	$\boldsymbol{O}\left(\frac{\mathbf{1}}{\boldsymbol{\mu r}}\right)$	$O\left(\frac{1}{\max\{\mu r^{1.5}\kappa^{1.5}, \mu r\kappa^2\}}\right)$
Iterations needed	$\boldsymbol{O}\left(\mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\epsilon}}\right)\right)$	$\boldsymbol{O}\left(\mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\epsilon}}\right)\right)$	$O\left(\kappa \log(\frac{1}{\epsilon})\right)$

3. Numerical Expierments

In this section, we present the empirical performance of our AccAltProj algorithm and compare it with the state-of-the-art AltProj algorithm from (Netrapalli et al., 2014) and the leading gradient descent based algorithm (GD) from (Yi et al., 2016). The tests are conducted on a laptop equipped with 64-bit Windows 7, Intel i7-4712HQ (4 Cores at 2.3 GHz) and 16GB DDR3L-1600 RAM, and executed from MATLAB R2017a. We implement AltProj by ourselves, while the codes for GD are downloaded from the author's website[3]. Hand tuned parameters are used for these algorithms to achieve the best performance in the numerical comparison. The codes for AccAltProj can be found online:

3. Website: `www.yixinyang.org/code/RPCA_GD.zip`.

https://github.com/caesarcai/AccAltProj_for_RPCA.

Notice that the computation of an initial guess by Algorithm 3 requires the truncated SVD on a full size matrix. As is typical in the literature, we used the PROPACK library[4] for this task when the size of $\boldsymbol{D}$ is large and r is relatively small. To reduce the dependence of the theoretical result on the condition number of the underlying low rank matrix, AltProj was originally designed to loop r stages for the input rank increasing from 1 to r and each stage contains a few number of iterations for a fixed rank. However, when the condition number is medium large which is the case in our tests, we have observed that AltProj achieves the best computational efficiency when fixing the rank to r. Thus, to make fair comparison, we test AltProj when input rank is fixed, the same as the other two algorithms.

Synthetic Datasets We follow the setup in (Netrapalli et al., 2014) and (Yi et al., 2016) for the random tests on synthetic data. An $n \times n$ rank r matrix $\boldsymbol{L}$ is formed via $\boldsymbol{L} = \boldsymbol{P}\boldsymbol{Q}^T$, where $\boldsymbol{P}, \boldsymbol{Q} \in \mathbb{R}^{n\times r}$ are two random matrices having their entries drawn i.i.d from the standard normal distribution. The locations of the non-zero entries of the underlying sparse matrix $\boldsymbol{S}$ are sampled uniformly and independently without replacement, while the values of the non-zero entries are drawn i.i.d from the uniform distribution over the interval $[-c \cdot \mathbb{E}(|[\boldsymbol{L}]_{ij}|), c \cdot \mathbb{E}(|[\boldsymbol{L}]_{ij}|)]$ for some constant $c > 0$. The relative computing error at the k^{th} iteration of a single test is defined as

$$err_k = \frac{\|\boldsymbol{D} - \boldsymbol{L}_k - \boldsymbol{S}_k\|_F}{\|\boldsymbol{D}\|_F}. \tag{11}$$

The test algorithms are terminated when either the relative computing error is smaller than a tolerance, $err_k < tol$, or a maximum number of 100 iterations is reached. Recall that μ is the incoherence parameter of the low rank matrix $\boldsymbol{L}$ and α is the sparsity parameter of the sparse matrix $\boldsymbol{S}$. In the random tests, we use 1.1μ in AltProj and AccAltProj, and use 1.1α in GD.

Though we are only able to provide a theoretical guarantee for AccAltProj with trim in this paper, it can be easily seen that AccAltProj can also be implemented without the trim step. Thus, both AccAltProj with and without trim are tested. The parameters β and β_{init} are set to be $\beta = \frac{1.1\mu r}{2\sqrt{mn}}$ and $\beta_{init} = \frac{1.1\mu r}{\sqrt{mn}}$ in our experiments, and $\gamma = 0.5$ is used when $\alpha < 0.55$ and $\gamma = 0.65$ is used when $\alpha \geq 0.55$.

We first test the performance of the algorithms under different values of α for fixed $n = 2500$ and $r = 5$. Three different values of c are investigated: $c \in \{0.2, 1, 5\}$, which represent three different signal levels of $\boldsymbol{S}$. For each value of c, 10 different values of α from 0.3 to 0.75 are tested. We set $tol = 10^{-6}$ in the stopping condition for all the test algorithms. The backtracking line search has been used in GD which can improve its recovery performance substantially in our tests. An algorithm is considered to have successfully reconstructed $\boldsymbol{L}$ (or equivalently, $\boldsymbol{S}$) if the low rank output of the algorithm $\boldsymbol{L}_k$ satisfies

$$\frac{\|\boldsymbol{L}_k - \boldsymbol{L}\|_F}{\|\boldsymbol{L}\|_F} \leq 10^{-4}.$$

4. Website: sun.stanford.edu/~rmunk/PROPACK.

Table 2: Rate of success for AccAltProj with and without trim, AltProj, and GD for different values of α.

$c = 0.2$	0.3	0.35	0.4	0.45	0.5	0.55	0.6	0.65	0.7	0.75
AccAltProj w/ trim	10	10	10	10	10	10	10	10	4	0
AccAltProj w/o trim	10	10	10	10	10	10	10	10	4	0
AltProj	10	10	10	10	10	10	10	10	0	0
GD	10	10	10	0	0	0	0	0	0	0

$c = 1$	0.3	0.35	0.4	0.45	0.5	0.55	0.6	0.65	0.7	0.75
AccAltProj w/ trim	10	10	10	10	10	10	10	9	0	0
AccAltProj w/o trim	10	10	10	10	10	10	10	9	0	0
AltProj	10	10	10	10	10	10	10	8	0	0
GD	10	10	10	10	9	0	0	0	0	0

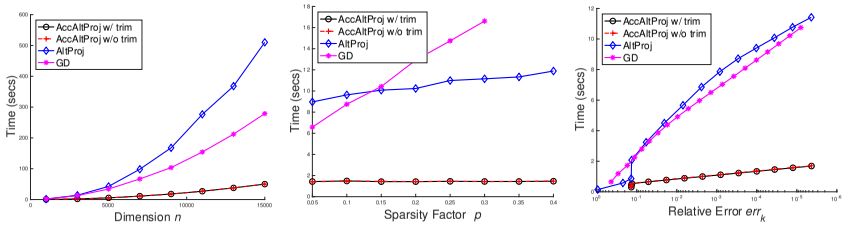
$c = 5$	0.3	0.35	0.4	0.45	0.5	0.55	0.6	0.65	0.7	0.75
AccAltProj w/ trim	10	10	10	10	10	10	10	5	0	0
AccAltProj w/o trim	10	10	10	10	10	10	10	5	0	0
AltProj	10	10	10	10	10	10	10	3	0	0
GD	10	10	10	10	10	10	7	0	0	0

The number of successful reconstructions for each algorithm out of 10 random tests are presented in Table 2. It is clear that AccAltProj (with and without trim) and AltProj exhibit similar behavior even though the theoretical requirement of AccAltProj with trim is a bit more stringent than that of AltProj, and they can tolerate larger values of α than GD when c is small.

Next, we evaluate the runtime of the test algorithms. The computational results are plotted in Figure 2 together with the setup corresponding to each plot. Figure 2(i) shows that AccAltProj is substantially faster than AltProj and GD. In particular, when n is large, it achieves about 10× speedup. Figure 2(ii) shows that AccAltProj and AltProj are less sensitive to the sparsity of $\boldsymbol{S}$. Notice that we have used a well-tuned fixed stepsize for GD here so that it can achieve the best computational efficiency. Thus, GD fails to converge when $\alpha \geq 0.35$ which is smaller than the largest value of α for successful recovery corresponding to $c = 1$ in Table 2. Lastly, Figure 2(iii) shows the lowest computational time of AccAltProj against the relative computing error.

Video Background Subtraction In this section, we compare the performance of AccAltProj with and without trim, AltProj and GD on video background subtraction, a real world benchmark problem for RPCA. The task in background subtraction is to separate moving foreground objects from a static background. The two videos we have used for this test are *Shoppingmall* and *Restaurant* which can be found online[5].The size of each frame of *Shoppingmall* is 256×320 and that of *Restaurant* is 120×160. The total number of frames are 1000 and 3055 in *Shoppingmall* and *Restaurant*, respectively. Each video can be represented by a matrix, where each column of the matrix is a vectorized frame of the

5. Website: perception.i2r.a-star.edu.sg/bk_model/bk_index.html.



(i) Varying Dimension vs Runtime (ii) Varying Sparsity vs Runtime (iii) Relative Error vs Runtime

Figure 2: Runtime for synthetic datasets: (i) Varying dimension n vs runtime, where $r = 5$, $\alpha = 0.1$, $c = 1$, and n varies from 1000 to 15000. The algorithms are terminated after $err_k < 10^{-4}$ is satisfied. (ii) Varying sparsity factor α vs runtime, where $r = 5$, $c = 1$ and $n = 2500$. The algorithms are terminated when either $err_k < 10^{-4}$ or 100 number of iterations is reached, whichever comes first. (iii) Relative error err_k vs runtime, where $r = 5$, $\alpha = 0.1$, $c = 1$, and $n = 2500$. The algorithms are terminated after $err_k < 10^{-5}$ is satisfied so that we can observe more iterations.

Table 3: Computational results for video background subtraction. Here "S" represents *Shoppingmall*, "R" represents *Restaurant*, and μ is the incoherence parameter of the output low rank matrices along the time axis (i.e., among different frames).

	AccAltProj w/ trim		AccAltProj w/o trim		AltProj		GD	
	runtime	μ	runtime	μ	runtime	μ	runtime	μ
S	38.98s	2.12	38.79s	2.26	82.97s	2.13	161.1s	2.85
R	28.09s	5.16	27.94s	5.25	69.12s	5.28	107.3s	6.07

video. Then, we apply each algorithm to decompose the matrix into a low rank part which represents the static background of the video and a sparse part which represents the moving objects in the video. Since there is no ground truth for the incoherence parameter and the sparsity parameter, their values are estimated by trial-and-error in the tests. We set $\gamma = 0.7$ and $r = 2$ in the decomposition of both videos, and *tol* is set to 10^{-4} in the stopping criteria. All the four algorithms can achieve desirable visual performance for the two tested videos and we only present the decomposition results of three selected frames for both AccAltProj with trim and without trim in Figure 3.

Table 3 contains the runtime of each algorithm. We can see that AccAltProj with and without trim are also faster than AltProj and GD for the background subtraction experiments conducted here. We also include the incoherence values of the output low rank matrices along the time axis. It is worth noting that the incoherence parameter value of the low rank output from AccAltProj with trim are smaller than that from AccAltProj without trim, which suggests the output backgrounds from AccAltProj with trim are more consistent through all the frames. Additionally, AccAltProj and AltProj have comparable output incoherence.



Figure 3: Video background subtraction: The top three rows correspond to three different frames from the video *Shoppingmall*, while the bottom three rows are frames from the video *Restaurant*. The first column contains the original frames, the middle two columns are the separated background and foreground outputs of AccAltProj with trim, and the right two columns are the separated background and foreground outputs of AccAltProj without trim.

4. Proofs

4.1. Proof of Theorem 1

The proof of Theorem 1 follows a route established in (Netrapalli et al., 2014). Despite this, the details of the proof itself are nevertheless quite involved because there are two more operations (i.e., projection onto a tangent space and trim) in AccAltProj than in AltProj. Overall, the proof consists of two steps:

- When $\|\boldsymbol{L}-\boldsymbol{L}_k\|_2$ and $\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty$ are sufficiently small, and $supp(\boldsymbol{S}_k)\subset\Omega$, then $\|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_2$ deceases in some sense by a constant factor (see Lemma 12) and $\|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_\infty$ is small (see Lemma 13).
- When $\|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_\infty$ is sufficiently small, we can choose ζ_{k+1} such that $supp(\boldsymbol{S}_{k+1})\subset\Omega$ and $\|\boldsymbol{S}-\boldsymbol{S}_{k+1}\|_\infty$ is small (see Lemma 14).

These results will be presented in a set of lemmas. For ease of notation we define $\tau:=4\alpha\mu r\kappa$ and $\upsilon:=\tau(48\sqrt{\mu r}\kappa+\mu r)$ in the sequel.

Lemma 3 ***(Weyl's inequality)*** *Let $\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}\in\mathbb{R}^{n\times n}$ be the symmetric matrices such that $\boldsymbol{A}=\boldsymbol{B}+\boldsymbol{C}$. Then the inequality*

$$|\sigma_i^A-\sigma_i^B|\le\|\boldsymbol{C}\|_2$$

holds for all i, where σ_i^A and σ_i^B represent the i^{th} singular values of $\boldsymbol{A}$ and $\boldsymbol{B}$ respectively.

Proof This is a well-known result and the proof can be found in many standard textbooks, see for example Bhatia (2013). ■

Lemma 4 *Let $\boldsymbol{S}\in\mathbb{R}^{n\times n}$ be a symmetric sparse matrix which satisfies Assumption A2. Then, the inequality*

$$\|\boldsymbol{S}\|_2\le\alpha n\|\boldsymbol{S}\|_\infty$$

holds, where α is the sparsity level of $\boldsymbol{S}$.

Proof The proof can be found in (Netrapalli et al., 2014, Lemma 4). ■

Lemma 5 *Let* Trim *be the algorithm defined by Algorithm 2. If $\boldsymbol{L}_k\in\mathbb{R}^{n\times n}$ is a rank-r matrix with*

$$\|\boldsymbol{L}_k-\boldsymbol{L}\|_2\le\frac{\sigma_r^L}{20\sqrt{r}},$$

then the trim output with the level $\sqrt{\frac{\mu r}{n}}$ satisfies

$$\|\widetilde{\boldsymbol{L}}_k-\boldsymbol{L}\|_F\le 8\kappa\|\boldsymbol{L}_k-\boldsymbol{L}\|_F, \tag{12}$$

$$\max_i\|\boldsymbol{e}_i^T\widetilde{\boldsymbol{U}}_k\|_2\le\frac{10}{9}\sqrt{\frac{\mu r}{n}},\quad\text{and}\quad\max_j\|\boldsymbol{e}_j^T\widetilde{\boldsymbol{V}}_k\|_2\le\frac{10}{9}\sqrt{\frac{\mu r}{n}}, \tag{13}$$

where $\widetilde{\boldsymbol{L}}_k=\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{\Sigma}}_k\widetilde{\boldsymbol{V}}_k^T$ is the SVD of $\widetilde{\boldsymbol{L}}_k$. Furthermore, it follows that

$$\|\widetilde{\boldsymbol{L}}_k-\boldsymbol{L}\|_2\le 8\sqrt{2r}\kappa\|\boldsymbol{L}_k-\boldsymbol{L}\|_2. \tag{14}$$

Proof Since both $\boldsymbol{L}$ and $\boldsymbol{L}_k$ are rank-r matrices, $\boldsymbol{L}_k-\boldsymbol{L}$ is rank at most $2r$. So

$$\|\boldsymbol{L}_k-\boldsymbol{L}\|_F\le\sqrt{2r}\|\boldsymbol{L}_k-\boldsymbol{L}\|_2\le\sqrt{2r}\frac{\sigma_r^L}{20\sqrt{r}}=\frac{\sigma_r^L}{10\sqrt{2}}.$$

Then, the first two parts of the lemma, i.e., (12) and (13), follow from (Wei et al., 2016a, Lemma 4.10). Noting that $\|\widetilde{\boldsymbol{L}}_k-\boldsymbol{L}\|_2\le\|\widetilde{\boldsymbol{L}}_k-\boldsymbol{L}\|_F$, (14) follows immediately. ■

Lemma 6 *Let $\boldsymbol{L}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ and $\widetilde{\boldsymbol{L}}_k=\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{\Sigma}}_k\widetilde{\boldsymbol{V}}_k^T$ be the SVD of two rank-r matrices, then*

$$\|\boldsymbol{U}\boldsymbol{U}^T-\widetilde{\boldsymbol{U}}\widetilde{\boldsymbol{U}}^T\|_2\le\frac{\|\widetilde{\boldsymbol{L}}_k-\boldsymbol{L}\|_2}{\sigma_r^L},\qquad\|\boldsymbol{V}\boldsymbol{V}^T-\widetilde{\boldsymbol{V}}\widetilde{\boldsymbol{V}}^T\|_2\le\frac{\|\widetilde{\boldsymbol{L}}_k-\boldsymbol{L}\|_2}{\sigma_r^L}, \tag{15}$$

and

$$\|(\mathcal{I}-\mathcal{P}_{\widetilde{T}_k})\boldsymbol{L}\|_2\le\frac{\|\widetilde{\boldsymbol{L}}_k-\boldsymbol{L}\|_2^2}{\sigma_r^L}. \tag{16}$$

Proof The proof of (15) can be found in (Wei et al., 2016b, Lemma 4.2). The Frobenius norm version of (16) can also be found in (Wei et al., 2016a,b). Here we only need to prove the spectral norm version, i.e., (16). Since $\boldsymbol{L}=\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{L}$ and $\widetilde{\boldsymbol{L}}_k(\boldsymbol{I}-\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)=\boldsymbol{0}$, we have

$$\begin{aligned}
\|(\mathcal{I}-\mathcal{P}_{\widetilde{T}_k})\boldsymbol{L}\|_2&=\|(\boldsymbol{I}-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)\boldsymbol{L}(\boldsymbol{I}-\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)\|_2\\
&=\|(\boldsymbol{I}-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{L}(\boldsymbol{I}-\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)\|_2\\
&=\|(\boldsymbol{U}\boldsymbol{U}^T-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{L}(\boldsymbol{I}-\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)\|_2\\
&=\|(\boldsymbol{U}\boldsymbol{U}^T-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)\boldsymbol{L}(\boldsymbol{I}-\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)\|_2\\
&=\|(\boldsymbol{U}\boldsymbol{U}^T-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)(\boldsymbol{L}-\widetilde{\boldsymbol{L}}_k)(\boldsymbol{I}-\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)\|_2\\
&\le\|(\boldsymbol{U}\boldsymbol{U}^T-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)\|_2\|(\boldsymbol{L}-\widetilde{\boldsymbol{L}}_k)\|_2\|(\boldsymbol{I}-\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T)\|_2\\
&\le\frac{\|\widetilde{\boldsymbol{L}}_k-\boldsymbol{L}\|_2^2}{\sigma_r^L},
\end{aligned}$$

where the last inequality follows from (15). ■

Lemma 7 *Let $\boldsymbol{S}\in\mathbb{R}^{n\times n}$ be a symmetric matrix satisfying Assumption A2. Let $\widetilde{\boldsymbol{L}}_k\in\mathbb{R}^{n\times n}$ be a rank-r matrix with $\frac{100}{81}\mu$-incoherence. That is,*

$$\max_i\|\boldsymbol{e}_i^T\widetilde{\boldsymbol{U}}_k\|_2\le\frac{10}{9}\sqrt{\frac{\mu r}{n}}\quad\text{and}\quad\max_j\|\boldsymbol{e}_j^T\widetilde{\boldsymbol{V}}_k\|_2\le\frac{10}{9}\sqrt{\frac{\mu r}{n}},$$

where $\widetilde{\boldsymbol{L}}=\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{\Sigma}}_k\widetilde{\boldsymbol{V}}_k^T$ is the SVD of $\widetilde{\boldsymbol{L}}_k$. If $supp(\boldsymbol{S}_k)\subset\Omega$, then

$$\|\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)\|_\infty\le 4\alpha\mu r\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty. \tag{17}$$

Proof By the incoherence assumption of $\widetilde{\boldsymbol{L}}_k$ and the sparsity assumption of $\boldsymbol{S}-\boldsymbol{S}_k$, we have

$$\begin{aligned}
[\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)]_{ab} &= \langle \mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k), \boldsymbol{e}_a\boldsymbol{e}_b^T\rangle \\
&= \langle \boldsymbol{S}-\boldsymbol{S}_k, \mathcal{P}_{\widetilde{T}_k}(\boldsymbol{e}_a\boldsymbol{e}_b^T)\rangle \\
&= \langle \boldsymbol{S}-\boldsymbol{S}_k, \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{e}_a\boldsymbol{e}_b^T + \boldsymbol{e}_a\boldsymbol{e}_b^T\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T - \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{e}_a\boldsymbol{e}_b^T\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T\rangle \\
&= \langle (\boldsymbol{S}-\boldsymbol{S}_k)\boldsymbol{e}_b, \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{e}_a\rangle + \langle \boldsymbol{e}_a^T(\boldsymbol{S}-\boldsymbol{S}_k), \boldsymbol{e}_b^T\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T\rangle - \langle \boldsymbol{S}-\boldsymbol{S}_k, \widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{e}_a\boldsymbol{e}_b^T\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T\rangle \\
&\le \|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty\left(\sum_{i|(i,b)\in\Omega}|\boldsymbol{e}_i^T\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{e}_a| + \sum_{j|(a,j)\in\Omega}|\boldsymbol{e}_b^T\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T\boldsymbol{e}_j|\right) \\
&\quad + \|\boldsymbol{S}-\boldsymbol{S}_k\|_2\,\|\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{e}_a\boldsymbol{e}_b^T\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T\|_* \\
&\le 2\alpha n\frac{100\mu r}{81n}\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty + \alpha n\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty\|\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{e}_a\boldsymbol{e}_b^T\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T\|_F \\
&\le \frac{200}{81}\alpha\mu r\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty + \alpha n\frac{\mu r}{n}\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty \\
&= 4\alpha\mu r\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty,
\end{aligned}$$

where the first inequality uses Hölder's inequality and the second inequality uses Lemma 4. We also use the fact $\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\boldsymbol{e}_a\boldsymbol{e}_b^T\widetilde{\boldsymbol{V}}_k\widetilde{\boldsymbol{V}}_k^T$ is a rank-1 matrix to bound its nuclear norm. ■

Lemma 8 *Under the symmetric setting, i.e., $\boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{V}\boldsymbol{V}^T$ where $\boldsymbol{U}\in\mathbb{R}^{n\times r}$ and $\boldsymbol{V}\in\mathbb{R}^{n\times r}$ are two orthogonal matrices, we have*

$$\|\mathcal{P}_T\boldsymbol{Z}\|_2 \le \sqrt{\frac{4}{3}}\|\boldsymbol{Z}\|_2$$

for any symmetric matrix $\boldsymbol{Z}\in\mathbb{R}^{n\times n}$. Moreover, the upper bound is tight.

Proof First notice that

$$\mathcal{P}_T\boldsymbol{Z} = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{Z} + \boldsymbol{Z}\boldsymbol{U}\boldsymbol{U}^T - \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{Z}\boldsymbol{U}\boldsymbol{U}^T$$

is symmetric. Let $\boldsymbol{y}\in\mathbb{R}^n$ be a unit vector such that $\|\mathcal{P}_T\boldsymbol{Z}\|_2 = |\boldsymbol{y}^T(\mathcal{P}_T\boldsymbol{Z})\boldsymbol{y}|$. Denote $\boldsymbol{y}_1 = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{y}$ and $\boldsymbol{y}_2 = (\boldsymbol{I}-\boldsymbol{U}\boldsymbol{U}^T)\boldsymbol{y}$. Then,

$$\begin{aligned}
\|\mathcal{P}_T\boldsymbol{Z}\|_2 &= |\boldsymbol{y}^T(\mathcal{P}_T\boldsymbol{Z})\boldsymbol{y}| \\
&= |\boldsymbol{y}_1^T\boldsymbol{Z}\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{Z}\boldsymbol{y}_1 - \boldsymbol{y}_1^T\boldsymbol{Z}\boldsymbol{y}_1| \\
&= |\boldsymbol{y}_1^T\boldsymbol{Z}\boldsymbol{y}_1 + \boldsymbol{y}_1^T\boldsymbol{Z}\boldsymbol{y}_2 + \boldsymbol{y}_2^T\boldsymbol{Z}\boldsymbol{y}_1| \\
&= |\langle \boldsymbol{y}_1\boldsymbol{y}_1^T + \boldsymbol{y}_1\boldsymbol{y}_2^T + \boldsymbol{y}_2\boldsymbol{y}_1^T, \boldsymbol{Z}\rangle| \\
&\le \|\boldsymbol{y}_1\boldsymbol{y}_1^T + \boldsymbol{y}_1\boldsymbol{y}_2^T + \boldsymbol{y}_2\boldsymbol{y}_1^T\|_*\|\boldsymbol{Z}\|_2.
\end{aligned}$$

Let $a = \|\boldsymbol{y}_1\|_2^2$. Since $\boldsymbol{y}_1\perp\boldsymbol{y}_2$, we have $\|\boldsymbol{y}_1\|_2^2 + \|\boldsymbol{y}_2\|_2^2 = 1$, which implies $\|\boldsymbol{y}_2\|_2^2 = 1-a$ and

$$\boldsymbol{y}_1\boldsymbol{y}_1^T + \boldsymbol{y}_1\boldsymbol{y}_2^T + \boldsymbol{y}_2\boldsymbol{y}_1^T = \begin{bmatrix}\boldsymbol{y}_1 & \boldsymbol{y}_2\end{bmatrix}\begin{bmatrix}1 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}\boldsymbol{y}_1 & \boldsymbol{y}_2\end{bmatrix}^T$$

$$= \begin{bmatrix}\frac{\boldsymbol{y}_1}{\sqrt{a}} & \frac{\boldsymbol{y}_2}{\sqrt{1-a}}\end{bmatrix}\begin{bmatrix}a & \sqrt{a(1-a)}\\ \sqrt{a(1-a)} & 0\end{bmatrix}\begin{bmatrix}\frac{\boldsymbol{y}_1}{\sqrt{a}} & \frac{\boldsymbol{y}_2}{\sqrt{1-a}}\end{bmatrix}^T.$$

Since $\begin{bmatrix}\frac{\boldsymbol{y}_1}{\sqrt{a}} & \frac{\boldsymbol{y}_2}{\sqrt{1-a}}\end{bmatrix}$ is an orthogonal matrix, one has

$$\begin{aligned}
\|\boldsymbol{y}_1\boldsymbol{y}_1^T + \boldsymbol{y}_1\boldsymbol{y}_2^T + \boldsymbol{y}_2\boldsymbol{y}_1^T\|_* &= \left\|\begin{bmatrix}a & \sqrt{a(1-a)}\\ \sqrt{a(1-a)} & 0\end{bmatrix}\right\|_* \\
&= \sqrt{a^2+4a(1-a)} \\
&= \sqrt{\frac{4}{3} - 3\left(a-\frac{2}{3}\right)^2} \\
&\le \sqrt{\frac{4}{3}},
\end{aligned}$$

which complete the proof for the upper bound.

To show the tightness of the bound, let $\boldsymbol{U} = \boldsymbol{V} = \begin{bmatrix}1\\0\end{bmatrix}$ and $\boldsymbol{Z} = \begin{bmatrix}1 & \sqrt{2}\\ \sqrt{2} & -1\end{bmatrix}$. It can be easily verified that $\|\mathcal{P}_T\boldsymbol{Z}\|_2 = \sqrt{\frac{4}{3}}\|\boldsymbol{Z}\|_2$. ■

Lemma 9 *Let $\boldsymbol{U}\in\mathbb{R}^{n\times r}$ be an orthogonal matrix with μ-incoherence, i.e., $\|\boldsymbol{e}_i^T\boldsymbol{U}\|_2 \le \sqrt{\frac{\mu r}{n}}$ for all i. Then, for any $\boldsymbol{Z}\in\mathbb{R}^{n\times n}$, the inequality*

$$\|\boldsymbol{e}_i^T\boldsymbol{Z}^a\boldsymbol{U}\|_2 \le \max_l\sqrt{\frac{\mu r}{n}}(\sqrt{n}\|\boldsymbol{e}_l^T\boldsymbol{Z}\|_2)^a$$

holds for all i and $a\ge 0$.

Proof This proof is done by mathematical induction.
Base case: When $a=0$, $\|\boldsymbol{e}_i^T\boldsymbol{U}\| \le \sqrt{\frac{\mu r}{n}}$ is satisfied following from the assumption.
Induction Hypothesis: $\|\boldsymbol{e}_i^T(\boldsymbol{Z})^a\boldsymbol{U}\|_2 \le \max_l\sqrt{\frac{\mu r}{n}}(\sqrt{n}\|\boldsymbol{e}_l^T\boldsymbol{Z}\|_2)^a$ for all i at the a^{th} power.
Induction Step: We have

$$\begin{aligned}
\|\boldsymbol{e}_i^T\boldsymbol{Z}^{a+1}\boldsymbol{U}\|_2^2 &= \|\boldsymbol{e}_i^T\boldsymbol{Z}\boldsymbol{Z}^a\boldsymbol{U}\|_2^2 \\
&= \sum_j\left(\sum_k[\boldsymbol{Z}]_{ik}[\boldsymbol{Z}^a\boldsymbol{U}]_{kj}\right)^2 \\
&= \sum_{k_1k_2}[\boldsymbol{Z}]_{ik_1}[\boldsymbol{Z}]_{ik_2}\sum_j[\boldsymbol{Z}^a\boldsymbol{U}]_{k_1j}[\boldsymbol{Z}^a\boldsymbol{U}]_{k_2j} \\
&= \sum_{k_1k_2}[\boldsymbol{Z}]_{ik_1}[\boldsymbol{Z}]_{ik_2}\langle \boldsymbol{e}_{k_1}^T\boldsymbol{Z}^a\boldsymbol{U}, \boldsymbol{e}_{k_2}^T\boldsymbol{Z}^a\boldsymbol{U}\rangle \\
&\le \sum_{k_1k_2}|[\boldsymbol{Z}]_{ik_1}[\boldsymbol{Z}]_{ik_2}|\|\boldsymbol{e}_{k_1}^T\boldsymbol{Z}^a\boldsymbol{U}\|_2\,\|\boldsymbol{e}_{k_2}^T\boldsymbol{Z}^a\boldsymbol{U}\|_2
\end{aligned}$$

$$\leq \max_l \frac{\mu r}{n}(\sqrt{n}\|\boldsymbol{e}_l^T\boldsymbol{Z}\|_2)^{2a}\sum_{k_1k_2}|[\boldsymbol{Z}]_{ik_1}[\boldsymbol{Z}]_{ik_2}|$$
$$\leq \max_l \frac{\mu r}{n}(\sqrt{n}\|\boldsymbol{e}_l^T\boldsymbol{Z}\|_2)^{2a}(\sqrt{n}\|\boldsymbol{e}_i^T\boldsymbol{Z}\|_2)^2$$
$$\leq \max_l \frac{\mu r}{n}(\sqrt{n}\|\boldsymbol{e}_l^T\boldsymbol{Z}\|_2)^{2a+2}.$$

The proof is complete by taking a square root from both sides. ■

Lemma 10 *Let $\boldsymbol{L}\in\mathbb{R}^{n\times n}$ and $\boldsymbol{S}\in\mathbb{R}^{n\times n}$ be two symmetric matrices satisfying Assumptions A1 and A2, respectively. Let $\widetilde{\boldsymbol{L}}_k\in\mathbb{R}^{n\times n}$ be the trim output of $\boldsymbol{L}_k$. If*

$$\|\boldsymbol{L}-\boldsymbol{L}_k\|_2\leq 8\alpha\mu r\gamma^k\sigma_1^L,\quad \|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty\leq\frac{\mu r}{n}\gamma^k\sigma_1^L,\text{ and } supp(\boldsymbol{S}_k)\subset\Omega,$$

then

$$\|(\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}+\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)\|_2\leq\tau\gamma^{k+1}\sigma_r^L \tag{18}$$

and

$$\max_l\sqrt{n}\|\boldsymbol{e}_l^T[(\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}+\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)]\|_2\leq\upsilon\gamma^k\sigma_r^L \tag{19}$$

hold for all $k\geq 0$, provided $1>\gamma\geq 512\tau r\kappa^2+\frac{1}{\sqrt{12}}$. Here recall that $\tau=4\alpha\mu r\kappa$ and $\upsilon=\tau(48\sqrt{\mu}r\kappa+\mu r)$.

Proof For all $k\geq 0$, we get

$$\begin{aligned}\|(\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}+\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)\|_2&\leq\|(\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}\|_2+\|\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)\|_2\\&\leq\frac{\|\boldsymbol{L}-\widetilde{\boldsymbol{L}}_k\|_2^2}{\sigma_r^L}+\sqrt{\frac{4}{3}}\|\boldsymbol{S}-\boldsymbol{S}_k\|_2\\&\leq\frac{(8\sqrt{2r}\kappa)^2\|\boldsymbol{L}-\boldsymbol{L}_k\|_2^2}{\sigma_r^L}+\sqrt{\frac{4}{3}}\alpha n\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty\\&\leq 128\cdot 8\alpha\mu r^2\kappa^3\|\boldsymbol{L}-\boldsymbol{L}_k\|_2+\sqrt{\frac{4}{3}}\alpha n\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty\\&\leq\left(512\tau r\kappa^2+\frac{1}{4}\sqrt{\frac{4}{3}}\right)4\alpha\mu r\gamma^k\sigma_1^L\\&\leq 4\alpha\mu r\gamma^{k+1}\sigma_1^L,\\&=\tau\gamma^{k+1}\sigma_r^L\end{aligned}$$

where the second inequality uses Lemma 6 and 8, the third inequality uses Lemma 4 and 5, the fourth inequality follows from $\frac{\|\boldsymbol{L}-\boldsymbol{L}_k\|_2}{\sigma_r^L}\leq 8\alpha\mu r\kappa$, and the last inequality uses the bound of γ.

To compute the bound of $\max_l\sqrt{n}\|\boldsymbol{e}_l^T[(\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}+\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)]\|_2$, first note that

$$\max_l\|\boldsymbol{e}_l^T(\mathcal{I}-\mathcal{P}_{\widetilde{T}_k})\boldsymbol{L}\|_2=\max_l\|\boldsymbol{e}_l^T(\boldsymbol{U}\boldsymbol{U}^T-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)(\boldsymbol{L}-\widetilde{\boldsymbol{L}}_k)(\boldsymbol{I}-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)\|_2$$

$$\begin{aligned}&\leq\max_l\|\boldsymbol{e}_l^T(\boldsymbol{U}\boldsymbol{U}^T-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T)\|_2\|\boldsymbol{L}-\widetilde{\boldsymbol{L}}_k\|_2\|\boldsymbol{I}-\widetilde{\boldsymbol{U}}_k\widetilde{\boldsymbol{U}}_k^T\|_2\\&\leq\left(\frac{19}{9}\sqrt{\frac{\mu r}{n}}\right)\|\boldsymbol{L}-\widetilde{\boldsymbol{L}}_k\|_2,\end{aligned}$$

where the last inequality follows from the fact $\boldsymbol{L}$ is μ-incoherent and $\widetilde{\boldsymbol{L}}_k$ is $\frac{100}{81}\mu$-incoherent. Hence, for all $k\geq 0$, we have

$$\begin{aligned}\max_l\sqrt{n}\|\boldsymbol{e}_l^T((\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}+\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k))\|_2&\leq\max_l\sqrt{n}\|\boldsymbol{e}_l^T(\mathcal{I}-\mathcal{P}_{\widetilde{T}_k})\boldsymbol{L}\|_2+\sqrt{n}\|\boldsymbol{e}_l^T\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)\|_2\\&\leq\frac{19\sqrt{n}}{9}\sqrt{\frac{\mu r}{n}}\|\boldsymbol{L}-\widetilde{\boldsymbol{L}}_k\|_2+n\|\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)\|_\infty\\&\leq\frac{19}{9}8\sqrt{2\mu}r\kappa\|\boldsymbol{L}-\boldsymbol{L}_k\|_2+4n\alpha\mu r\|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty\\&\leq 24\sqrt{\mu}r\kappa\cdot 8\alpha\mu r\gamma^k\sigma_1^L+4n\alpha\mu r\cdot\frac{\mu r}{n}\gamma^k\sigma_1^L\\&=\upsilon\gamma^k\sigma_r^L,\end{aligned}$$

where the third inequality uses Lemma 5 and 7. ■

Lemma 11 *Let $\boldsymbol{L}\in\mathbb{R}^{n\times n}$ and $\boldsymbol{S}\in\mathbb{R}^{n\times n}$ be two symmetric matrices satisfying Assumptions A1 and A2, respectively. Let $\widetilde{\boldsymbol{L}}_k\in\mathbb{R}^{n\times n}$ be the trim output of $\boldsymbol{L}_k$. If*

$$\|\boldsymbol{L}-\boldsymbol{L}_k\|_2\leq 8\alpha\mu r\gamma^k\sigma_1^L,\quad \|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty\leq\frac{\mu r}{n}\gamma^k\sigma_1^L,\text{ and } supp(\boldsymbol{S}_k)\subset\Omega,$$

then

$$|\sigma_i^L-|\lambda_i^{(k)}||\leq\tau\sigma_r^L \tag{20}$$

and

$$(1-2\tau)\gamma^j\sigma_1^L\leq|\lambda_{r+1}^{(k)}|+\gamma^j|\lambda_1^{(k)}|\leq(1+2\tau)\gamma^j\sigma_1^L \tag{21}$$

hold for all $k\geq 0$ and $j\leq k+1$, provided $1>\gamma\geq 512\tau r\kappa^2+\frac{1}{\sqrt{12}}$. Here $|\lambda_i^{(k)}|$ is the i^{th} singular value of $\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)$.

Proof Since $\boldsymbol{D}=\boldsymbol{L}+\boldsymbol{S}$, we have

$$\begin{aligned}\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)&=\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{L}+\boldsymbol{S}-\boldsymbol{S}_k)\\&=\boldsymbol{L}+(\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}+\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k).\end{aligned}$$

Hence, by Weyl's inequality and (18) in Lemma 10, we can see that

$$\begin{aligned}|\sigma_i^L-|\lambda_i^{(k)}||&\leq\|(\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}+\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)\|_2\\&\leq\tau\gamma^{k+1}\sigma_r^L\end{aligned}$$

hold for all i and $k\geq 0$. So the first claim is proved since $\gamma<1$.

Notice that $\boldsymbol{L}$ is a rank-r matrix, which implies $\sigma_{r+1}^L = 0$, so we have

$$\begin{aligned}||\lambda_{r+1}^{(k)}| + \gamma^j|\lambda_1^{(k)}| - \gamma^j\sigma_1^L| &= ||\lambda_{r+1}^{(k)}| - \sigma_{r+1}^L + \gamma^j|\lambda_1^{(k)}| - \gamma^j\sigma_1^L| \\ &\le \tau\gamma^{k+1}\sigma_r^L + \tau\gamma^{j+k+1}\sigma_r^L \\ &\le \left(1+\gamma^{k+1}\right)\tau\gamma^j\sigma_r^L \\ &\le 2\tau\gamma^j\sigma_1^L\end{aligned}$$

for all $j \le k+1$. This completes the proof of the second claim. ■

Lemma 12 *Let $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{S} \in \mathbb{R}^{n\times n}$ be two symmetric matrices satisfying Assumptions A1 and A2, respectively. Let $\widetilde{\boldsymbol{L}}_k \in \mathbb{R}^{n\times n}$ be the trim output of $\boldsymbol{L}_k$. If*

$$\|\boldsymbol{L}-\boldsymbol{L}_k\|_2 \le 8\alpha\mu r\gamma^k\sigma_1^L, \quad \|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty \le \frac{\mu r}{n}\gamma^k\sigma_1^L, \text{ and } supp(\boldsymbol{S}_k) \subset \Omega,$$

then we have

$$\|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_2 \le 8\alpha\mu r\gamma^{k+1}\sigma_1^L,$$

provided $1 > \gamma \ge 512\tau r\kappa^2 + \frac{1}{\sqrt{12}}$.

Proof A direct calculation yields

$$\begin{aligned}\|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_2 &\le \|\boldsymbol{L}-\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)\|_2 + \|\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)-\boldsymbol{L}_{k+1}\|_2 \\ &\le 2\|\boldsymbol{L}-\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)\|_2 \\ &= 2\|\boldsymbol{L}-\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{L}+\boldsymbol{S}-\boldsymbol{S}_k)\|_2 \\ &= 2\|(\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}+\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)\|_2 \\ &\le 2\cdot\tau\gamma^{k+1}\sigma_r^L \\ &= 8\alpha\mu r\gamma^{k+1}\sigma_1^L,\end{aligned}$$

where the second inequality follows from the fact $\boldsymbol{L}_{k+1} = \mathcal{H}_r(\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k))$ is the best rank-r approximation of $\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)$, and the last inequality uses (18) in Lemma 10. ■

Lemma 13 *Let $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{S} \in \mathbb{R}^{n\times n}$ be two symmetric matrices satisfying Assumptions A1 and A2, respectively. Let $\widetilde{\boldsymbol{L}}_k \in \mathbb{R}^{n\times n}$ be the trim output of $\boldsymbol{L}_k$. If*

$$\|\boldsymbol{L}-\boldsymbol{L}_k\|_2 \le 8\alpha\mu r\gamma^k\sigma_1^L, \quad \|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty \le \frac{\mu r}{n}\gamma^k\sigma_1^L, \text{ and } supp(\boldsymbol{S}_k) \subset \Omega,$$

then we have

$$\|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_\infty \le \left(\frac{1}{2}-\tau\right)\frac{\mu r}{n}\gamma^{k+1}\sigma_1^L,$$

provided $1 > \gamma \ge \max\{512\tau r\kappa^2 + \frac{1}{\sqrt{12}}, \frac{2\upsilon}{(1-12\tau)(1-\tau-\upsilon)^2}\}$ *and* $\tau < \frac{1}{12}$.

Proof Let $\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k) = \begin{bmatrix}\boldsymbol{U}_{k+1} & \ddot{\boldsymbol{U}}_{k+1}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Lambda} & \boldsymbol{0}\\ \boldsymbol{0} & \ddot{\boldsymbol{\Lambda}}\end{bmatrix}\begin{bmatrix}\boldsymbol{U}_{k+1}^T\\ \ddot{\boldsymbol{U}}_{k+1}^T\end{bmatrix} = \boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T + \ddot{\boldsymbol{U}}_{k+1}\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_{k+1}^T$ be its eigenvalue decomposition. We use the lighter notation λ_i $(1 \le i \le n)$ for the eigenvalues of $\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)$ at the k-th iteration and assume they are ordered by $|\lambda_1| \ge |\lambda_2| \ge \cdots \ge |\lambda_n|$. Moreover, $\boldsymbol{\Lambda}$ has its r largest eigenvalues in magnitude, $\boldsymbol{U}_{k+1}$ contains the first r eigenvectors, and $\ddot{\boldsymbol{U}}_{k+1}$ has the rest. It follows that $\boldsymbol{L}_{k+1} = \mathcal{H}_r(\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)) = \boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T$.

Denote $\boldsymbol{Z} = \mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)-\boldsymbol{L} = (\mathcal{P}_{\widetilde{T}_k}-\mathcal{I})\boldsymbol{L}+\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{S}-\boldsymbol{S}_k)$. Let $\boldsymbol{u}_i$ be the i^{th} eigenvector of $\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)$. Noting that $(\lambda_i\boldsymbol{I}-\boldsymbol{Z})\boldsymbol{u}_i = \boldsymbol{L}\boldsymbol{u}_i$, we have

$$\boldsymbol{u}_i = \left(\boldsymbol{I}-\frac{\boldsymbol{Z}}{\lambda_i}\right)^{-1}\frac{\boldsymbol{L}}{\lambda_i}\boldsymbol{u}_i = \left(\boldsymbol{I}+\frac{\boldsymbol{Z}}{\lambda_i}+\left(\frac{\boldsymbol{Z}}{\lambda_i}\right)^2+\cdots\right)\frac{\boldsymbol{L}}{\lambda_i}\boldsymbol{u}_i$$

for all $\boldsymbol{u}_i$ with $1 \le i \le r$, where the expansion is valid because

$$\frac{\|\boldsymbol{Z}\|_2}{\lambda_i} \le \frac{\|\boldsymbol{Z}\|_2}{\lambda_r} \le \frac{\tau}{1-\tau} < 1$$

following from (18) in Lemma 10 and (20) in Lemma 11. This implies

$$\begin{aligned}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T &= \sum_{i=1}^r \boldsymbol{u}_i\lambda_i\boldsymbol{u}_i^T \\ &= \sum_{i=1}^r\left(\sum_{a\ge0}\left(\frac{\boldsymbol{Z}}{\lambda_i}\right)^a\frac{\boldsymbol{L}}{\lambda_i}\right)\boldsymbol{u}_i\lambda_i\boldsymbol{u}_i^T\left(\sum_{b\ge0}\left(\frac{\boldsymbol{Z}}{\lambda_i}\right)^b\frac{\boldsymbol{L}}{\lambda_i}\right)^T \\ &= \sum_{a\ge0}\boldsymbol{Z}^a\boldsymbol{L}\sum_{i=1}^r\left(\boldsymbol{u}_i\frac{1}{\lambda_i^{a+b+1}}\boldsymbol{u}_i^T\right)\boldsymbol{L}\sum_{b\ge0}\boldsymbol{Z}^b \\ &= \sum_{a,b\ge0}\boldsymbol{Z}^a\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T\boldsymbol{L}\boldsymbol{Z}^b.\end{aligned}$$

Thus, we have

$$\begin{aligned}\|\boldsymbol{L}_{k+1}-\boldsymbol{L}\|_\infty &= \|\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T-\boldsymbol{L}\|_\infty \\ &= \|\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_{k+1}^T\boldsymbol{L}-\boldsymbol{L}+\sum_{a+b>0}\boldsymbol{Z}^a\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T\boldsymbol{L}\boldsymbol{Z}^b\|_\infty \\ &\le \|\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_{k+1}^T\boldsymbol{L}-\boldsymbol{L}\|_\infty+\sum_{a+b>0}\|\boldsymbol{Z}^a\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T\boldsymbol{L}\boldsymbol{Z}^b\|_\infty \\ &:= \boldsymbol{Y}_0+\sum_{a+b>0}\boldsymbol{Y}_{ab}.\end{aligned}$$

We will handle $\boldsymbol{Y}_0$ first. Recall that $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ is the SVD of the symmetric matrix $\boldsymbol{L}$ which obeys μ-incoherence, i.e., $\boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{V}\boldsymbol{V}^T$ and $\|\boldsymbol{e}_i^T\boldsymbol{U}\boldsymbol{U}^T\|_2 \le \sqrt{\frac{\mu r}{n}}$ for all i. So, for each (i,j) entry of $\boldsymbol{Y}_0$, one has

$$\boldsymbol{Y}_0 = \max_{ij}|\boldsymbol{e}_i^T(\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_{k+1}^T\boldsymbol{L}-\boldsymbol{L})\boldsymbol{e}_j|$$

$$
\begin{aligned}
&= \max_{ij} |\boldsymbol{e}_i^T \boldsymbol{U}\boldsymbol{U}^T(\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_{k+1}^T\boldsymbol{L} - \boldsymbol{L})\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{e}_j| \\
&\le \max_{ij} \|\boldsymbol{e}_i^T\boldsymbol{U}\boldsymbol{U}^T\|_2 \, \|\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_{k+1}^T\boldsymbol{L} - \boldsymbol{L}\|_2 \, \|\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{e}_j\|_2 \\
&\le \frac{\mu r}{n}\|\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_{k+1}^T\boldsymbol{L} - \boldsymbol{L}\|_2,
\end{aligned}
$$

where the second equation follows from the fact $\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{L} = \boldsymbol{L}\boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{L}$. Since $\boldsymbol{L} = \boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T + \ddot{\boldsymbol{U}}_{k+1}\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_{k+1}^T - \boldsymbol{Z}$, there hold

$$
\begin{aligned}
&\|\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_{k+1}^T\boldsymbol{L} - \boldsymbol{L}\|_2 \\
&= \|(\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T + \ddot{\boldsymbol{U}}_{k+1}\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_{k+1}^T - \boldsymbol{Z})\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_{k+1}^T(\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T + \ddot{\boldsymbol{U}}_{k+1}\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_{k+1}^T - \boldsymbol{Z}) - \boldsymbol{L}\|_2 \\
&= \|\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T - \boldsymbol{L} - \boldsymbol{U}_{k+1}\boldsymbol{U}_{k+1}^T\boldsymbol{Z} - \boldsymbol{Z}\boldsymbol{U}_{k+1}\boldsymbol{U}_{k+1}^T - \boldsymbol{Z}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_{k+1}^T\boldsymbol{Z}\|_2 \\
&\le \|\boldsymbol{Z} - \ddot{\boldsymbol{U}}_{k+1}\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_{k+1}^T\|_2 + 2\|\boldsymbol{Z}\|_2 + \frac{\|\boldsymbol{Z}\|_2^2}{|\lambda_r|} \\
&\le \|\ddot{\boldsymbol{U}}_{k+1}\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_{k+1}^T\|_2 + 4\|\boldsymbol{Z}\|_2 \\
&\le |\lambda_{r+1}| + 4\|\boldsymbol{Z}\|_2 \\
&\le 5\|\boldsymbol{Z}\|_2 \\
&\le 5\tau\gamma^{k+1}\sigma_1^L,
\end{aligned}
$$

where the fifth inequality uses (18) in Lemma 10, and notice that $\frac{\|\boldsymbol{Z}\|_2}{|\lambda_r|} \le \frac{\tau}{1-\tau} < 1$ since $\tau < \frac{1}{2}$ and $|\lambda_{r+1}| \le \|\boldsymbol{Z}\|_2$ since $\boldsymbol{L}$ is a rank-r matrix. Thus, we have

$$
\boldsymbol{Y}_0 \le \frac{\mu r}{n} 5\tau\gamma^{k+1}\sigma_1^L. \tag{22}
$$

Next, we derive an upper bound for the rest part. Note that

$$
\begin{aligned}
\boldsymbol{Y}_{ab} &= \max_{ij} |\boldsymbol{e}_i^T\boldsymbol{Z}^a\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T\boldsymbol{L}\boldsymbol{Z}^b\boldsymbol{e}_j| \\
&= \max_{ij} |(\boldsymbol{e}_i^T\boldsymbol{Z}^a\boldsymbol{U}\boldsymbol{U}^T)\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T\boldsymbol{L}(\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{Z}^b\boldsymbol{e}_j)| \\
&\le \max_{ij} \|\boldsymbol{e}_i^T\boldsymbol{Z}^a\boldsymbol{U}\|_2 \, \|\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T\boldsymbol{L}\|_2 \, \|\boldsymbol{U}^T\boldsymbol{Z}^b\boldsymbol{e}_j\|_2 \\
&\le \max_{l} \frac{\mu r}{n}(\sqrt{n}\|\boldsymbol{e}_l^T\boldsymbol{Z}\|_2)^{a+b}\|\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T\boldsymbol{L}\|_2,
\end{aligned}
$$

where the last inequality uses Lemma 9. Furthermore, by using $\boldsymbol{L} = \boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T + \ddot{\boldsymbol{U}}_{k+1}\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_{k+1}^T - \boldsymbol{Z}$ again, we get

$$
\begin{aligned}
&\|\boldsymbol{L}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T\boldsymbol{L}\|_2 \\
&= \|(\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T + \ddot{\boldsymbol{U}}_{k+1}\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_{k+1}^T - \boldsymbol{Z})\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T(\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}\boldsymbol{U}_{k+1}^T + \ddot{\boldsymbol{U}}_{k+1}\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_{k+1}^T - \boldsymbol{Z})\|_2 \\
&= \|\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b-1)}\boldsymbol{U}_{k+1}^T - \boldsymbol{Z}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b)}\boldsymbol{U}_{k+1}^T - \boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b)}\boldsymbol{U}_{k+1}^T\boldsymbol{Z} + \boldsymbol{Z}\boldsymbol{U}_{k+1}\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_{k+1}^T\boldsymbol{Z}\|_2 \\
&\le |\lambda_r|^{-(a+b-1)} + |\lambda_r|^{-(a+b)}\|\boldsymbol{Z}\|_2 + |\lambda_r|^{-(a+b)}\|\boldsymbol{Z}\|_2 + |\lambda_r|^{-(a+b+1)}\|\boldsymbol{Z}\|_2^2 \\
&= |\lambda_r|^{-(a+b-1)}\left(1 + \frac{2\|\boldsymbol{Z}\|_2}{|\lambda_r|} + \left(\frac{\|\boldsymbol{Z}\|_2}{|\lambda_r|}\right)^2\right)
\end{aligned}
$$

$$
\begin{aligned}
&= |\lambda_r|^{-(a+b-1)}\left(1 + \frac{\|\boldsymbol{Z}\|_2}{|\lambda_r|}\right)^2 \\
&\le |\lambda_r|^{-(a+b-1)}\left(\frac{1}{1-\tau}\right)^2 \\
&\le \left(\frac{1}{1-\tau}\right)^2\left((1-\tau)\sigma_r^L\right)^{-(a+b-1)},
\end{aligned}
$$

where the second inequality follows from $\frac{\|\boldsymbol{Z}\|_2}{|\lambda_r|} \le \frac{\tau}{1-\tau}$, and the last inequality follows from Lemma 11. Together with (19) in Lemma 10, we have

$$
\begin{aligned}
\sum_{a+b>0}\boldsymbol{Y}_{ab} &\le \sum_{a+b>0}\frac{\mu r}{n}\left(\frac{1}{1-\tau}\right)^2\upsilon\gamma^k\sigma_r^L\left(\frac{\upsilon\gamma^k\sigma_r^L}{(1-\tau)\sigma_r^L}\right)^{a+b-1} \\
&\le \frac{\mu r}{n}\left(\frac{1}{1-\tau}\right)^2\upsilon\gamma^k\sigma_1^L\sum_{a+b>0}\left(\frac{\upsilon}{1-\tau}\right)^{a+b-1} \\
&\le \frac{\mu r}{n}\left(\frac{1}{1-\tau}\right)^2\upsilon\gamma^k\sigma_1^L\left(\frac{1}{1-\frac{\upsilon}{1-\tau}}\right)^2 \\
&\le \frac{\mu r}{n}\left(\frac{1}{1-\tau-\upsilon}\right)^2\upsilon\gamma^k\sigma_1^L.
\end{aligned} \tag{23}
$$

Finally, combining (22) and (23) together gives

$$
\begin{aligned}
\|\boldsymbol{L}_{k+1} - \boldsymbol{L}\|_\infty &= \boldsymbol{Y}_0 + \sum_{a+b>0}\boldsymbol{Y}_{ab} \\
&\le \frac{\mu r}{n}5\tau\gamma^{k+1}\sigma_1^L + \frac{\mu r}{n}\left(\frac{1}{1-\tau-\upsilon}\right)^2\upsilon\gamma^k\sigma_1^L \\
&\le \left(\frac{1}{2} - \tau\right)\frac{\mu r}{n}\gamma^{k+1}\sigma_1^L,
\end{aligned}
$$

where the last inequality follows from $\gamma \ge \frac{2\upsilon}{(1-12\tau)(1-\tau-\upsilon)^2}$. ■

Lemma 14 *Let $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{S} \in \mathbb{R}^{n\times n}$ be two symmetric matrices satisfying Assumptions A1 and A2, respectively. Let $\widetilde{\boldsymbol{L}}_k \in \mathbb{R}^{n\times n}$ be the trim output of $\boldsymbol{L}_k$. Recall that $\beta = \frac{\mu r}{2n}$. If*

$$
\|\boldsymbol{L} - \boldsymbol{L}_k\|_2 \le 8\alpha\mu r\gamma^k\sigma_1^L, \quad \|\boldsymbol{S} - \boldsymbol{S}_k\|_\infty \le \frac{\mu r}{n}\gamma^k\sigma_1^L, \text{ and } supp(\boldsymbol{S}_k) \subset \Omega
$$

then we have

$$
supp(\boldsymbol{S}_{k+1}) \subset \Omega \quad \textit{and} \quad \|\boldsymbol{S} - \boldsymbol{S}_{k+1}\|_\infty \le \frac{\mu r}{n}\gamma^{k+1}\sigma_1^L,
$$

provided $1 > \gamma \ge \max\{512\tau r\kappa^2 + \frac{1}{\sqrt{12}}, \frac{2\upsilon}{(1-12\tau)(1-\tau-\upsilon)^2}\}$ *and* $\tau < \frac{1}{12}$.

Proof We first notice that

$$[\boldsymbol{S}_{k+1}]_{ij} = [\mathcal{T}_{\zeta_{k+1}}(\boldsymbol{D}-\boldsymbol{L}_{k+1})]_{ij} = [\mathcal{T}_{\zeta_{k+1}}(\boldsymbol{S}+\boldsymbol{L}-\boldsymbol{L}_{k+1})]_{ij} = \begin{cases} \mathcal{T}_{\zeta_{k+1}}([\boldsymbol{S}+\boldsymbol{L}-\boldsymbol{L}_{k+1}]_{ij}) & (i,j)\in\Omega \\ \mathcal{T}_{\zeta_{k+1}}([\boldsymbol{L}-\boldsymbol{L}_{k+1}]_{ij}) & (i,j)\in\Omega^c \end{cases}.$$

Let $|\lambda_i^{(k)}|$ denote i^{th} singular value of $\mathcal{P}_{\widetilde{T}_k}(\boldsymbol{D}-\boldsymbol{S}_k)$. By Lemmas 11 and 13, we have

$$\begin{aligned} |[\boldsymbol{L}-\boldsymbol{L}_{k+1}]_{ij}| &\le \|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_\infty \le \left(\frac{1}{2}-\tau\right)\frac{\mu r}{n}\gamma^{k+1}\sigma_1^L \\ &\le \left(\frac{1}{2}-\tau\right)\frac{\mu r}{n}\frac{1}{1-2\tau}\left(|\lambda_{r+1}^{(k)}|+\gamma^{k+1}|\lambda_1^{(k)}|\right) \\ &= \zeta_{k+1} \end{aligned}$$

for any entry of $\boldsymbol{L}-\boldsymbol{L}_{k+1}$. Hence, $[\boldsymbol{S}_{k+1}]_{ij}=0$ for all $(i,j)\in\Omega^c$, i.e., $supp(\boldsymbol{S}_{k+1})\subset\Omega$.

Denote $\Omega_{k+1} := supp(\boldsymbol{S}_{k+1}) = \{(i,j) \mid [(\boldsymbol{D}-\boldsymbol{L}_{k+1})]_{ij} > \zeta_k\}$. Then, for any entry of $\boldsymbol{S}-\boldsymbol{S}_{k+1}$, there hold

$$[\boldsymbol{S}-\boldsymbol{S}_{k+1}]_{ij} = \begin{cases} 0 \\ [\boldsymbol{L}_{k+1}-\boldsymbol{L}]_{ij} \\ [\boldsymbol{S}]_{ij} \end{cases} \le \begin{cases} 0 \\ \|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_\infty \\ \|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_\infty+\zeta_{k+1} \end{cases} \le \begin{cases} 0 & (i,j)\in\Omega^c \\ \left(\frac{1}{2}-\tau\right)\frac{\mu r}{n}\gamma^{k+1}\sigma_1^L & (i,j)\in\Omega_{k+1} \\ \frac{\mu r}{n}\gamma^{k+1}\sigma_1^L & (i,j)\in\Omega\backslash\Omega_{k+1}. \end{cases}$$

Here the last step follows from Lemma 11 which implies $\zeta_{k+1} = \frac{\mu r}{2n}(|\lambda_{r+1}^{(k)}| + \gamma^{k+1}|\lambda_1^{(k)}|) \le \left(\frac{1}{2}+\tau\right)\frac{\mu r}{n}\gamma^{k+1}\sigma_1^L$. Therefore, $\|\boldsymbol{S}-\boldsymbol{S}_{k+1}\|_\infty \le \frac{\mu r}{n}\gamma^{k+1}\sigma_1^L$. ■

Now, we have all the ingredients for the proof of Theorem 1.

Proof [Proof of Theorem 1] This theorem will be proved by mathematical induction.
Base Case: When $k=0$, the base case is satisfied by the assumption on the intialization.
Induction Step: Assume we have

$$\|\boldsymbol{L}-\boldsymbol{L}_k\|_2 \le 8\alpha\mu r\gamma^k\sigma_1^L, \quad \|\boldsymbol{S}-\boldsymbol{S}_k\|_\infty \le \frac{\mu r}{n}\gamma^k\sigma_1^L, \quad \text{and} \quad supp(\boldsymbol{S}_k)\subset\Omega$$

at the k^{th} iteration. At the $(k+1)^{th}$ iteration. If follows directly from Lemmas 12 and 14 that

$$\|\boldsymbol{L}-\boldsymbol{L}_{k+1}\|_2 \le 8\alpha\mu r\gamma^{k+1}\sigma_1^L, \quad \|\boldsymbol{S}-\boldsymbol{S}_{k+1}\|_\infty \le \frac{\mu r}{n}\gamma^{k+1}\sigma_1^L \quad \text{and} \quad supp(\boldsymbol{S}_{k+1})\subset\Omega,$$

which completes the proof.

Additionally, notice that we overall require $1>\gamma\ge\max\{512\tau r\kappa^2+\frac{1}{\sqrt{12}}, \frac{2\upsilon}{(1-12\tau)(1-\tau-\upsilon)^2}\}$. By the definition of τ and υ, one can easily see that the lower bound approaches $\frac{1}{\sqrt{12}}$ when the constant hidden in (4) is sufficiently large. Therefore, the theorem can be proved for any $\gamma\in\left(\frac{1}{\sqrt{12}},1\right)$. ■

4.2. Proof of Theorem 2

We first present a lemma which is a variant of Lemma 9 and also appears in (Netrapalli et al., 2014, Lemma 5). The lemma can be similarly proved by mathematical induction.

Lemma 15 *Let $\boldsymbol{S}\in\mathbb{R}^{n\times n}$ be a sparse matrix satisfying Assumption A2. Let $\boldsymbol{U}\in\mathbb{R}^{n\times r}$ be an orthogonal matrix with μ-incoherence, i.e., $\|\boldsymbol{e}_i^T\boldsymbol{U}\|_2\le\sqrt{\frac{\mu r}{n}}$ for all i. Then*

$$\|\boldsymbol{e}_i^T\boldsymbol{S}^a\boldsymbol{U}\|_2 \le \sqrt{\frac{\mu r}{n}}(\alpha n\|\boldsymbol{S}\|_\infty)^a$$

for all i and $a\ge 0$.

Though the proposed initialization scheme (i.e., Algorithms 3) basically consists of two steps of AltProj (Netrapalli et al., 2014), we provide an independent proof for Theorem 2 here because we bound the approximation errors of the low rank matrices using the spectral norm rather than the infinity norm. The proof of Theorem 2 follows a similar structure to that of Theorem 1, but without the projection onto a low dimensional tangent space. Instead of first presenting several auxiliary lemmas, we give a single proof by putting all the elements together.

Proof [Proof of Theorem 2] The proof can be partitioned into several parts.

(i) Note that $\boldsymbol{L}_{-1}=0$ and

$$\|\boldsymbol{L}-\boldsymbol{L}_{-1}\|_\infty = \|\boldsymbol{L}\|_\infty = \max_{ij}\left|\boldsymbol{e}_i^T\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T\boldsymbol{e}_j\right| \le \max_{ij}\|\boldsymbol{e}_i^T\boldsymbol{U}\|_2\|\boldsymbol{\Sigma}\|_2\|\boldsymbol{U}^T\boldsymbol{e}_j\|_2 \le \frac{\mu r}{n}\sigma_1^L,$$

where the last inequality follows from Assumption A1, i.e., $\boldsymbol{L}$ is μ-incoherent. Thus, with the choice of $\beta_{init}\ge\frac{\mu r\sigma_1^L}{n\sigma_1^D}$, we have

$$\|\boldsymbol{L}-\boldsymbol{L}_{-1}\|_\infty \le \beta_{init}\sigma_1^D = \zeta_{-1}. \tag{24}$$

Since

$$[\boldsymbol{S}_{-1}]_{ij} = [\mathcal{T}_{\zeta_{-1}}(\boldsymbol{S}+\boldsymbol{L}-\boldsymbol{L}_{-1})]_{ij} = \begin{cases} \mathcal{T}_{\zeta_{-1}}([\boldsymbol{S}+\boldsymbol{L}-\boldsymbol{L}_{-1}]_{ij}) & (i,j)\in\Omega \\ \mathcal{T}_{\zeta_{-1}}([\boldsymbol{L}-\boldsymbol{L}_{-1}]_{ij}) & (i,j)\in\Omega^c, \end{cases}$$

it follows that $[\boldsymbol{S}_{-1}]_{ij}=0$ for all $(i,j)\in\Omega^c$, i.e. $\Omega_{-1} := supp(\boldsymbol{S}_{-1})\subset\Omega$. Moreover, for any entries of $\boldsymbol{S}-\boldsymbol{S}_{-1}$, we have

$$[\boldsymbol{S}-\boldsymbol{S}_{-1}]_{ij} = \begin{cases} 0 \\ [\boldsymbol{L}_{-1}-\boldsymbol{L}]_{ij} \\ [\boldsymbol{S}]_{ij} \end{cases} \le \begin{cases} 0 \\ \|\boldsymbol{L}-\boldsymbol{L}_{-1}\|_\infty \\ \|\boldsymbol{L}-\boldsymbol{L}_{-1}\|_\infty+\zeta_{-1} \end{cases} \le \begin{cases} 0 & (i,j)\in\Omega^c \\ \frac{\mu r}{n}\sigma_1^L & (i,j)\in\Omega_{-1} \\ \frac{4\mu r}{n}\sigma_1^L & (i,j)\in\Omega\backslash\Omega_{-1} \end{cases},$$

where the last inequality follows from $\beta_{init}\le\frac{3\mu r\sigma_1^L}{n\sigma_1^D}$, so that $\zeta_{-1}\le\frac{3\mu r}{n}\sigma_1^L$. Therefore, if follows that

$$supp(\boldsymbol{S}_{-1})\subset\Omega \quad \text{and} \quad \|\boldsymbol{S}-\boldsymbol{S}_{-1}\|_\infty \le \frac{4\mu r}{n}\sigma_1^L. \tag{25}$$

By Lemma 4, we also have

$$\|\boldsymbol{S}-\boldsymbol{S}_{-1}\|_2 \le \alpha n\|\boldsymbol{S}-\boldsymbol{S}_{-1}\|_\infty \le 4\alpha\mu r\sigma_1^L.$$

(ii) To bound the approximation error of $\boldsymbol{L}_0$ to $\boldsymbol{L}$ in terms of the spectral norm, note that

$$\begin{aligned}\|\boldsymbol{L}-\boldsymbol{L}_0\|_2 &\le \|\boldsymbol{L}-(\boldsymbol{D}-\boldsymbol{S}_{-1})\|_2 + \|(\boldsymbol{D}-\boldsymbol{S}_{-1})-\boldsymbol{L}_0\|_2\\ &\le 2\|\boldsymbol{L}-(\boldsymbol{D}-\boldsymbol{S}_{-1})\|_2\\ &= 2\|\boldsymbol{L}-(\boldsymbol{L}+\boldsymbol{S}-\boldsymbol{S}_{-1})\|_2\\ &= 2\|\boldsymbol{S}-\boldsymbol{S}_{-1}\|_2,\end{aligned}$$

where the second inequality follows from the fact $\boldsymbol{L}_0 = \mathcal{H}_r(\boldsymbol{D}-\boldsymbol{S}_{-1})$ is the best rank-r approximation of $\boldsymbol{D}-\boldsymbol{S}_{-1}$. It follows immediately that

$$\|\boldsymbol{L}-\boldsymbol{L}_0\|_2 \le 8\alpha\mu r\sigma_1^L. \tag{26}$$

(iii) Since $\boldsymbol{D}=\boldsymbol{L}+\boldsymbol{S}$, we have $\boldsymbol{D}-\boldsymbol{S}_{-1} = \boldsymbol{L}+\boldsymbol{S}-\boldsymbol{S}_{-1}$. Let λ_i denotes the i^{th} eigenvalue of $\boldsymbol{D}-\boldsymbol{S}_{-1}$ ordered by $|\lambda_1|\ge|\lambda_2|\ge\cdots\ge|\lambda_n|$. The application of Weyl's inequality together with the bound of α in Assumption A2 implies that

$$|\sigma_i^L - |\lambda_i|| \le \|\boldsymbol{S}-\boldsymbol{S}_{-1}\|_2 \le \frac{\sigma_r^L}{8} \tag{27}$$

holds for all i. Consequently, we have

$$\frac{7}{8}\sigma_i^L \le |\lambda_i| \le \frac{9}{8}\sigma_i^L, \qquad \forall 1\le i\le r, \tag{28}$$

$$\frac{\|\boldsymbol{S}-\boldsymbol{S}_{-1}\|_2}{|\lambda_r|} \le \frac{\frac{\sigma_r^L}{8}}{\frac{7\sigma_r^L}{8}} = \frac{1}{7}. \tag{29}$$

Let $\boldsymbol{D}-\boldsymbol{S}_{-1} = [\boldsymbol{U}_0,\ddot{\boldsymbol{U}}_0]\begin{bmatrix}\boldsymbol{\Lambda} & \boldsymbol{0}\\ \boldsymbol{0} & \ddot{\boldsymbol{\Lambda}}\end{bmatrix}[\boldsymbol{U}_0,\ddot{\boldsymbol{U}}_0]^T = \boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T + \ddot{\boldsymbol{U}}_0\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_0^T$ be its eigenvalue decomposition, where $\boldsymbol{\Lambda}$ has the r largest eigenvalues in magnitude and $\ddot{\boldsymbol{\Lambda}}$ contains the rest eigenvalues. Also, $\boldsymbol{U}_0$ contains the first r eigenvectors, and $\ddot{\boldsymbol{U}}_0$ has the rest. Notice that $\boldsymbol{L}_0 = \mathcal{H}_r(\boldsymbol{D}-\boldsymbol{S}_{-1}) = \boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T$ due to the symmetric setting. Denote $\boldsymbol{E} = \boldsymbol{D}-\boldsymbol{S_{-1}}-\boldsymbol{L} = \boldsymbol{S}-\boldsymbol{S}_{-1}$. Let $\boldsymbol{u}_i$ be the i^{th} eigenvector of $\boldsymbol{D}-\boldsymbol{S}_{-1} = \boldsymbol{L}+\boldsymbol{E}$. For $1\le i\le r$, since $(\boldsymbol{L}+\boldsymbol{E})\boldsymbol{u}_i = \lambda_i\boldsymbol{u}_i$, we have

$$\boldsymbol{u}_i = \left(\boldsymbol{I}-\frac{\boldsymbol{E}}{\lambda_i}\right)^{-1}\frac{\boldsymbol{L}}{\lambda_i}\boldsymbol{u}_i = \left(\boldsymbol{I}+\frac{\boldsymbol{E}}{\lambda_i}+\left(\frac{\boldsymbol{E}}{\lambda_i}\right)^2+\cdots\right)\frac{\boldsymbol{L}}{\lambda_i}\boldsymbol{u}_i$$

for each $\boldsymbol{u}_i$, where the expansion in the last equality is valid because $\frac{\|\boldsymbol{E}\|_2}{|\lambda_i|} \le \frac{1}{7}$ for all $1\le i\le r$ following from (29). Therefore,

$$\begin{aligned}\|\boldsymbol{L}_0-\boldsymbol{L}\|_\infty &= \|\boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T-\boldsymbol{L}\|_\infty\\ &= \|\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_0^T\boldsymbol{L}-\boldsymbol{L}+\sum_{a+b>0}\boldsymbol{E}^a\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T\boldsymbol{L}\boldsymbol{E}^b\|_\infty\\ &\le \|\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_0^T\boldsymbol{L}-\boldsymbol{L}\|_\infty + \sum_{a+b>0}\|\boldsymbol{E}^a\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T\boldsymbol{L}\boldsymbol{E}^b\|_\infty\\ &:= \boldsymbol{Y}_0 + \sum_{a+b>0}\boldsymbol{Y}_{ab}.\end{aligned}$$

We will handle $\boldsymbol{Y}_0$ first. Recall that $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ is the SVD of the symmetric matrix $\boldsymbol{L}$ which is μ-incoherence, i.e., $\boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{V}\boldsymbol{V}^T$ and $\|\boldsymbol{e}_i^T\boldsymbol{U}\boldsymbol{U}^T\|_2 \le \sqrt{\frac{\mu r}{n}}$ for all i. For each (i,j) entry of $\boldsymbol{Y}_0$, we have

$$\begin{aligned}\boldsymbol{Y}_0 &= \max_{ij}|\boldsymbol{e}_i^T(\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_0^T\boldsymbol{L}-\boldsymbol{L})\boldsymbol{e}_j|\\ &= \max_{ij}|\boldsymbol{e}_i^T\boldsymbol{U}\boldsymbol{U}^T(\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_0^T\boldsymbol{L}-\boldsymbol{L})\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{e}_j|\\ &\le \max_{ij}\|\boldsymbol{e}_i^T\boldsymbol{U}\boldsymbol{U}^T\|_2\ \|\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_0^T\boldsymbol{L}-\boldsymbol{L}\|_2\ \|\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{e}_j\|_2\\ &\le \frac{\mu r}{n}\|\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_0^T\boldsymbol{L}-\boldsymbol{L}\|_2,\end{aligned}$$

where the second equation follows from the fact $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{L} = \boldsymbol{L}\boldsymbol{U}\boldsymbol{U}^T$. Since $\boldsymbol{L} = \boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T + \ddot{\boldsymbol{U}}_0\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_0^T - \boldsymbol{E}$,

$$\begin{aligned}&\|\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_0^T\boldsymbol{L}-\boldsymbol{L}\|_2\\ &= \|(\boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T+\ddot{\boldsymbol{U}}_0\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_0^T-\boldsymbol{E})\boldsymbol{U}_0\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_0^T(\boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T+\ddot{\boldsymbol{U}}_0\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_0^T-\boldsymbol{E})-\boldsymbol{L}\|_2\\ &= \|\boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T-\boldsymbol{L}-\boldsymbol{U}_0\boldsymbol{U}_0^T\boldsymbol{E}-\boldsymbol{E}\boldsymbol{U}_0\boldsymbol{U}_0^T-\boldsymbol{E}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-1}\boldsymbol{U}_0^T\boldsymbol{E}\|_2\\ &\le \|\boldsymbol{E}-\ddot{\boldsymbol{U}}_0\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_0^T\|_2 + 2\|\boldsymbol{E}\|_2 + \frac{\|\boldsymbol{E}\|_2^2}{|\lambda_r|}\\ &\le \|\ddot{\boldsymbol{U}}_0\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_0^T\|_2 + 4\|\boldsymbol{E}\|_2\\ &\le |\lambda_{r+1}| + 4\|\boldsymbol{E}\|_2\\ &\le 5\|\boldsymbol{E}\|_2,\end{aligned}$$

where the first and fourth inequality follow from (27) and (29), and $|\lambda_{r+1}| \le \|\boldsymbol{E}\|_2$ since $\sigma_{r+1}^L = 0$. Together, we have

$$\boldsymbol{Y}_0 \le \frac{5\mu r}{n}\|\boldsymbol{E}\|_2 \le 5\alpha\mu r\|\boldsymbol{E}\|_\infty, \tag{30}$$

where the last inequality follows from Lemma 4.

Next, we will find an upper bound for the rest part. Note that

$$\begin{aligned}\boldsymbol{Y}_{ab} &= \max_{ij}|\boldsymbol{e}_i^T\boldsymbol{E}^a\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T\boldsymbol{L}\boldsymbol{E}^b\boldsymbol{e}_j|\\ &= \max_{ij}|(\boldsymbol{e}_i^T\boldsymbol{E}^a\boldsymbol{U}\boldsymbol{U}^T)\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T\boldsymbol{L}(\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{E}^b\boldsymbol{e}_j)|\\ &\le \max_{ij}\|\boldsymbol{e}_i^T\boldsymbol{E}^a\boldsymbol{U}\|_2\ \|\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T\boldsymbol{L}\|_2\ \|\boldsymbol{U}^T\boldsymbol{E}^b\boldsymbol{e}_j\|_2\end{aligned}$$

$$\leq \frac{\mu r}{n}(\alpha n\|\boldsymbol{E}\|_\infty)^{a+b}\|\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T\boldsymbol{L}\|_2$$
$$\leq \alpha\mu r\|\boldsymbol{E}\|_\infty\left(\frac{\sigma_r^L}{8}\right)^{a+b-1}\|\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T\boldsymbol{L}\|_2,$$

where the second inequality uses Lemma 15. Furthermore, by using $\boldsymbol{L} = \boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T + \ddot{\boldsymbol{U}}_0\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_0^T - \boldsymbol{E}$ again, we have

$$\begin{aligned}
&\|\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T\boldsymbol{L}\|_2\\
&= \|(\boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T + \ddot{\boldsymbol{U}}_0\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_0^T - \boldsymbol{E})\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T(\boldsymbol{U}_0\boldsymbol{\Lambda}\boldsymbol{U}_0^T + \ddot{\boldsymbol{U}}_0\ddot{\boldsymbol{\Lambda}}\ddot{\boldsymbol{U}}_0^T - \boldsymbol{E})\|_2\\
&= \|\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b-1)}\boldsymbol{U}_0^T - \boldsymbol{E}\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b)}\boldsymbol{U}_0^T - \boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b)}\boldsymbol{U}_0^T\boldsymbol{E} + \boldsymbol{E}\boldsymbol{L}\boldsymbol{U}_0\boldsymbol{\Lambda}^{-(a+b+1)}\boldsymbol{U}_0^T\boldsymbol{E}\|_2\\
&\leq |\lambda_r|^{-(a+b-1)} + |\lambda_r|^{-(a+b)}\|\boldsymbol{E}\|_2 + |\lambda_r|^{-(a+b)}\|\boldsymbol{E}\|_2 + |\lambda_r|^{-(a+b+1)}\|\boldsymbol{E}\|_2^2\\
&= |\lambda_r|^{-(a+b-1)}\left(1 + \frac{2\|\boldsymbol{E}\|_2}{|\lambda_r|} + \left(\frac{\|\boldsymbol{E}\|_2}{|\lambda_r|}\right)^2\right)\\
&= |\lambda_r|^{-(a+b-1)}\left(1 + \frac{\|\boldsymbol{E}\|_2}{|\lambda_r|}\right)^2\\
&\leq 2|\lambda_r|^{-(a+b-1)}\\
&\leq 2\left(\frac{7}{8}\sigma_r^L\right)^{-(a+b-1)},
\end{aligned}$$

where the second inequality follows from (29) and the last inequality follows from (28). Together, we have

$$\begin{aligned}
\sum_{a+b>0}\boldsymbol{Y}_{ab} &\leq \sum_{a+b>0} 2\alpha\mu r\|\boldsymbol{E}\|_\infty\left(\frac{\frac{1}{8}\sigma_r^L}{\frac{7}{8}\sigma_r^L}\right)^{a+b-1}\\
&\leq 2\alpha\mu r\|\boldsymbol{E}\|_\infty\sum_{a+b>0}\left(\frac{1}{7}\right)^{a+b-1}\\
&\leq 2\alpha\mu r\|\boldsymbol{E}\|_\infty\left(\frac{1}{1-\frac{1}{7}}\right)^2\\
&\leq 3\alpha\mu r\|\boldsymbol{E}\|_\infty. \qquad (31)
\end{aligned}$$

Finally, combining (30)) and (31)) together yields

$$\begin{aligned}
\|\boldsymbol{L}_0 - \boldsymbol{L}\|_\infty &= \boldsymbol{Y}_0 + \sum_{a+b>0}\boldsymbol{Y}_{ab}\\
&\leq 5\alpha\mu r\|\boldsymbol{E}\|_\infty + 3\alpha\mu r\|\boldsymbol{E}\|_\infty\\
&\leq \frac{\mu r}{4n}\sigma_1^L, \qquad (32)
\end{aligned}$$

where the last step uses (25) and the bound of α in Assumption A2.

(iv) From the thresholding rule, we know that

$$[\boldsymbol{S}_0]_{ij} = [\mathcal{T}_{\zeta_0}(\boldsymbol{S}+\boldsymbol{L}-\boldsymbol{L}_0)]_{ij} = \begin{cases}\mathcal{T}_{\zeta_0}([\boldsymbol{S}+\boldsymbol{L}-\boldsymbol{L}_0]_{ij}) & (i,j)\in\Omega\\ \mathcal{T}_{\zeta_0}([\boldsymbol{L}-\boldsymbol{L}_0]_{ij}) & (i,j)\in\Omega^c\end{cases}.$$

So (28), (32) and $\zeta_0 = \frac{\mu r}{2n}\lambda_1$ imply $[\boldsymbol{S}_0]_{ij} = 0$ for all $(i,j)\in\Omega^c$, i.e., $supp(\boldsymbol{S}_0) := \Omega_0 \subset \Omega$. Also, for any entries of $\boldsymbol{S}-\boldsymbol{S}_0$, there hold

$$[\boldsymbol{S}-\boldsymbol{S}_0]_{ij} = \begin{cases}0\\ [\boldsymbol{L}_0-\boldsymbol{L}]_{ij}\\ [\boldsymbol{S}]_{ij}\end{cases} \leq \begin{cases}0\\ \|\boldsymbol{L}-\boldsymbol{L}_0\|_\infty\\ \|\boldsymbol{L}-\boldsymbol{L}_0\|_\infty+\zeta_0\end{cases} \leq \begin{cases}0 & (i,j)\in\Omega^c\\ \frac{\mu r}{4n}\sigma_1^L & (i,j)\in\Omega_0\\ \frac{\mu r}{n}\sigma_1^L & (i,j)\in\Omega\backslash\Omega_0.\end{cases}$$

Here the last inequality follows from (28) which implies $\zeta_0 = \frac{\mu r}{2n}\lambda_1 \leq \frac{3\mu r}{4n}\sigma_1^L$. Therefore, we have

$$supp(\boldsymbol{S}_0)\subset\Omega \quad\text{and}\quad \|\boldsymbol{S}-\boldsymbol{S}_0\|_\infty \leq \frac{\mu r}{n}\sigma_1^L.$$

The proof is compete by noting (26) and the above results. ■

5. Discussion and Future Direction

We have presented a highly efficient algorithm AccAltProj for robust principal component analysis. The algorithm is developed by introducing a novel subspace projection step before the SVD truncation, which reduces the per iteration computational complexity of the algorithm of alternating projections significantly. Theoretical recovery guarantee has been established for the new algorithm, while numerical simulations show that our algorithm is superior to other state-of-the-art algorithms.

There are three lines of research for future work. Firstly, the theoretical number of the non-zero entries in a sparse matrix below which AccAltProj can achieve successful recovery is highly pessimistic compared with our numerical findings. This suggests the possibility of improving the theoretical result. Secondly, recovery stability of the proposed algorithm to additive noise will be investigated in the future. Finally, this paper focuses on the fully observed setting. The proposed algorithm might be similarly extended to the partially observed setting where only partial entries of a matrix are observed. It is also interesting to study the recovery guarantee of the proposed algorithm under this partial observed setting.

Acknowledgments

This work was supported in part by grants HKRGC 16306317, NSFC 11801088, and Shanghai Sailing Program 18YF1401600.

References

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

Alekh Agarwal, Animashree Anandkumar, Prateek Jain, Praneeth Netrapalli, and Rashish Tandon. Learning sparsely used overcomplete dictionaries. In *Conference on Learning Theory*, pages 123–137, 2014.

Rajendra Bhatia. *Matrix analysis*, volume 169. Springer Science & Business Media, 2013.

Jian-Feng Cai, Haixia Liu, and Yang Wang. Fast rank one alternating minimization algorithm for phase retrieval. *arXiv preprint arXiv:1708.08751*, 2017.

Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717, 2009.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

Tony F Chan and Chiu-Kwong Wong. Convergence of the alternating minimization algorithm for blind deconvolution. *Linear Algebra and its Applications*, 316(1-3):259–285, 2000.

Venkat Chandrasekaran, Sujay Sanghavi, Pablo A Parrilo, and Alan S Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2): 572–596, 2011.

Yudong Chen and Martin J Wainwright. Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025*, 2015.

Yudong Chen, Sujay Sanghavi, and Huan Xu. Clustering sparse graphs. In *Advances in neural information processing systems*, pages 2204–2212, 2012.

Quanquan Gu, Zhaoran Wang, and Han Liu. Low-rank and sparse structure pursuit via alternating minimization. In *Artificial Intelligence and Statistics*, pages 600–609, 2016.

Moritz Hardt. On the provable convergence of alternating minimization for matrix completion. *arXiv preprint arXiv:1312.0925*, 2013.

Po-Sen Huang, Scott Deeann Chen, Paris Smaragdis, and Mark Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 57–60. IEEE, 2012.

Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674. ACM, 2013.

Raghunandan Hulikal Keshavan et al. *Efficient algorithms for collaborative filtering*. PhD thesis, Stanford University, 2012.

Liyuan Li, Weimin Huang, Irene Yu-Hua Gu, and Qi Tian. Statistical modeling of complex backgrounds for foreground object detection. *IEEE Transactions on Image Processing*, 13(11):1459–1472, 2004.

Bamdev Mishra and Rodolphe Sepulchre. R3MC: A Riemannian three-factor algorithm for low-rank matrix completion. In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pages 1137–1142. IEEE, 2014.

Bamdev Mishra, K Adithya Apuroop, and Rodolphe Sepulchre. A Riemannian geometry for low-rank matrix completion. *arXiv preprint arXiv:1211.1550*, 2012.

Bamdev Mishra, Gilles Meyer, Silvère Bonnabel, and Rodolphe Sepulchre. Fixed-rank matrix factorizations and Riemannian low-rank optimization. *Computational Statistics*, 29(3-4):591–621, 2014.

Hossein Mobahi, Zihan Zhou, Allen Y Yang, and Yi Ma. Holistic 3D reconstruction of urban structures from low-rank textures. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 593–600. IEEE, 2011.

Praneeth Netrapalli, Prateek Jain, and Sujay Sanghavi. Phase retrieval using alternating minimization. In *Advances in Neural Information Processing Systems*, pages 2796–2804, 2013.

Praneeth Netrapalli, UN Niranjan, Sujay Sanghavi, Animashree Anandkumar, and Prateek Jain. Non-convex robust PCA. In *Advances in Neural Information Processing Systems*, pages 1107–1115, 2014.

Thanh Ngo and Yousef Saad. Scaled gradients on Grassmann manifolds for matrix completion. In *Advances in Neural Information Processing Systems*, pages 1412–1420, 2012.

Joseph A O'Sullivan and Jasenka Benac. Alternating minimization algorithms for transmission tomography. *IEEE Transactions on Medical Imaging*, 26(3):283–297, 2007.

Steven W Peters and Robert W Heath. Interference alignment via alternating minimization. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 2445–2448. IEEE, 2009.

Ye Pu, Melanie N Zeilinger, and Colin N Jones. Complexity certification of the fast alternating minimization algorithm for linear MPC. *IEEE Transactions on Automatic Control*, 62(2):888–893, 2017.

Jared Tanner and Ke Wei. Low rank matrix completion by alternating steepest descent methods. *Applied and Computational Harmonic Analysis*, 40(2):417–429, 2016.

Yvon Tharrault, Gilles Mourot, José Ragot, and Didier Maquin. Fault detection and isolation with robust principal component analysis. *International Journal of Applied Mathematics and Computer Science*, 18(4):429–442, 2008.

Bart Vandereycken. Low-rank matrix completion by Riemannian optimization. *SIAM Journal on Optimization*, 23(2):1214–1236, 2013.

Yi Wang, Arthur Szlam, and Gilad Lerman. Robust locally linear analysis with applications to image denosing and blind inpainting. *SIAM Journal on Imaging Sciences*, 6(1):526–562, 2013.

Yilun Wang, Junfeng Yang, Wotao Yin, and Yin Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

Ke Wei, Jian-Feng Cai, Tony F Chan, and Shingyu Leung. Guarantees of Riemannian optimization for low rank matrix completion. *arXiv preprint arXiv:1603.06610*, 2016a.

Ke Wei, Jian-Feng Cai, Tony F Chan, and Shingyu Leung. Guarantees of Riemannian optimization for low rank matrix recovery. *SIAM Journal on Matrix Analysis and Applications*, 37(3):1198–1222, 2016b.

John Wright, Arvind Ganesh, Shankar Rao, Yigang Peng, and Yi Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in neural information processing systems*, pages 2080–2088, 2009.

Huan Xu, Constantine Caramanis, and Sujay Sanghavi. Robust PCA via outlier pursuit. In *Advances in Neural Information Processing Systems*, pages 2496–2504, 2010.

Xinyang Yi, Dohyung Park, Yudong Chen, and Constantine Caramanis. Fast algorithms for robust PCA via gradient descent. In *Advances in neural information processing systems*, pages 4152–4160, 2016.

Xianghao Yu, Juei-Chin Shen, Jun Zhang, and Khaled B Letaief. Alternating minimization algorithms for hybrid precoding in millimeter wave MIMO systems. *IEEE Journal of Selected Topics in Signal Processing*, 10(3):485–500, 2016.

Teng Zhang. Phase retrieval using alternating minimization in a batch setting. *arXiv preprint arXiv:1706.08167*, 2017.

Journal of Machine Learning Research 20 (2019) 1-55 Submitted 1/18; Revised 10/18; Published 1/19

Spectrum Estimation from a Few Entries

Ashish Khetan khetan2@illinois.edu
Sewoong Oh swoh@illinois.edu
Department of Industrial and Enterprise Systems Engineering
University of Illinois Urbana-Champaign
Urbana, IL 61801, USA

Editor: Moritz Hardt

Abstract

Singular values of a data in a matrix form provide insights on the structure of the data, the effective dimensionality, and the choice of hyper-parameters on higher-level data analysis tools. However, in many practical applications such as collaborative filtering and network analysis, we only get a partial observation. Under such scenarios, we consider the fundamental problem of recovering spectral properties of the underlying matrix from a sampling of its entries. In this paper, we address the problem of directly recovering the spectrum, which is the set of singular values, and also in sample-efficient approaches for recovering a spectral sum function, which is an aggregate sum of a fixed function applied to each of the singular values. Our approach is to first estimate the Schatten k-norms of a matrix for a small set of values of k, and then apply Chebyshev approximation when estimating a spectral sum function or apply moment matching in Wasserstein distance when estimating the singular values directly. The main technical challenge is in accurately estimating the Schatten norms from a sampling of a matrix. We introduce a novel unbiased estimator based on counting small structures called network motifs in a graph and provide guarantees that match its empirical performance. Our theoretical analysis shows that Schatten norms can be recovered accurately from strictly smaller number of samples compared to what is needed to recover the underlying low-rank matrix. Numerical experiments suggest that we significantly improve upon a competing approach of using matrix completion methods, below the matrix completion threshold, above which matrix completion algorithms recover the underlying low-rank matrix exactly.

Keywords: spectrum estimation, matrix completion, counting subgraphs.

1. Introduction

Computing and analyzing the set of singular values of a data in a matrix form, which is called the spectrum, provide insights into the geometry and topology of the data. Such a spectral analysis is routinely a first step in general data analysis with the goal of checking if there exists a lower dimensional subspace explaining the important aspects of the data, which itself might be high dimensional. Concretely, it is a first step in dimensionality reduction methods such as principal component analysis or canonical correlation analysis.

However, spectral analysis becomes challenging in practical scenarios where the data is only partially observed. We commonly observe pairwise relations of randomly chosen pairs: each user only rates a few movies in recommendation systems, each player/team only plays against a few opponents in sports leagues, each word appears in the same sentence with a small number of other words in word count matrices, and each worker answers a few questions in crowdsourcing. In other applications, we have more structured samples. For example, in a network analysis we might be interested in the spectrum of a large network, but only get to see the connections within a small subset of nodes corresponding to sampling a sub-matrix of the adjacency matrix. Whatever the sampling pattern is, typical number of paired relations we observe is significantly smaller than the dimension of the data matrix.

We study all such variations in sampling patterns for partially observed data matrices, and ask the following fundamental question: *can we estimate spectral properties of a data matrix from partial observations?* We propose a novel approach that allows us to estimate the spectrum, i.e. the singular values. A crucial building block in our approach is that spectral properties can be accurately approximated from the first few moments of the spectrum known as the Schatten k-norms defined as

$$\|M\|_k \;=\; \Big(\sum_{i=1}^{d} \sigma_i(M)^k\Big)^{1/k} ,$$

where $\sigma_1(M) \geq \sigma_2(M) \geq \cdots \geq \sigma_d(M) \geq 0$ are the singular values of the data matrix $M \in \mathbb{R}^{d\times d}$. Once we obtain accurate estimates of Schatten k-norms, these estimates, as well as corresponding performance guarantees, can readily be translated into accurate estimates of the spectrum of the matrix. Further, if we are interested in estimating a class of functions known as spectral sum functions of the form (4), our estimates of the Schatten norms can be used to estimate any spectral sum function using Chebyshev expansions.

1.1. Setup

We want to estimate the Schatten k-norm of a positive semidefinite matrix $M \in \mathbb{R}^{d\times d}$ from a subset of its entries. The restriction to positive semidefinite matrices is primarily for notational convenience, and our analyses, the estimator, and the efficient algorithms naturally generalize to any non-square matrices. Namely, we can extend our framework to bipartite graphs and estimate Schatten k-norm of any matrix for any even k. Let Ω denote the set of indices of samples we are given and let $\mathcal{P}_\Omega(M) = \{(i,j,M_{ij})\}_{(i,j)\in\Omega}$ denote the samples. With a slight abuse of notation, we used $\mathcal{P}_\Omega(M)$ to also denote the $d\times d$ sampled matrix:

$$\mathcal{P}_\Omega(M)_{ij} = \begin{cases} M_{ij} & \text{if } (i,j)\in\Omega\,, \\ 0 & \text{otherwise}\,, \end{cases}$$

and it should be clear from the context which one we refer to. Although we propose a framework that generally applies to any probabilistic sampling, it is necessary to propose specific sampling scenarios to provide tight analyses on the performance. Hence, we focus on *Erdös-Rényi sampling*.

There is an extensive line of research in low-rank matrix completion problems (Candès and Recht, 2009; Keshavan et al., 2010a), which addresses a fundamental question of how many samples are required to *complete* a matrix (i.e. estimate all the missing entries) from a small subset of sampled entries. It is typically assumed that each entry of the matrix is sampled independently with a probability $p \in (0,1]$. We refer to this scenario as *Erdös-Rényi sampling*, as the resulting pattern of the samples encoded as a graph is distributed

©2019 Ashish Khetan and Sewoong Oh.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-027.html.

as an Erdös-Rényi random graph. The spectral properties of such an sampled matrix have been well studied in the literature (Friedman et al., 1989; Achlioptas and McSherry, 2001; Feige and Ofek, 2005; Keshavan et al., 2010a; Le et al., 2015). In particular, it is known that the original matrix is close in spectral norm to the sampled one where the missing entries are filled in with zeros and properly rescaled under certain incoherence assumptions. This suggests using the singular values of the sampled and rescaled matrix $(d^2/|\Omega|)\mathcal{P}(M)$ directly for estimating the Schatten norms. However, in the sub-linear regime in which the number of samples $|\Omega| = d^2p$ is comparable to or significantly smaller than the degrees of freedom in representing a symmetric rank-r matrix, which is $dr - r^2$, the spectrum of the sampled matrix is significantly different from the spectrum of the original matrix as shown in Figure 1. We need to design novel estimators that are more sample efficient in the sub-linear regime where $d^2p \ll dr$.

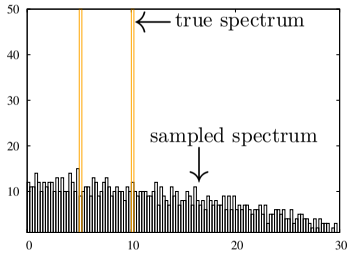


Figure 1: In yellow, we show the histogram of the singular values of a positive semi-definite matrix $M \in \mathbb{R}^{d\times d}$ of size $d = 1000$ with rank $r = 100$, with $\sigma_1 = \cdots = \sigma_{50} = 10$, $\sigma_{51} = \cdots = \sigma_{100} = 5$, and the rest at zero (we omit zero singular values in the plot for illustration). In comparison, we show in black the histogram of the singular values of the sampled matrix where each entry of M is sampled with probability $p = (1/d)r^{1-2/7}$ (properly rescaled by $1/p$ to best match the original spectrum).

1.2. Summary of the approach and preview of results

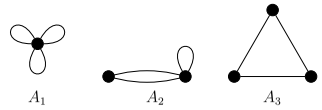
We propose first estimating one or a few Scahtten norms, which can be accurately estimated from samples, and using these estimated Schatten norms to approximate the spectral properties of interest: spectral sum functions and the spectrum. We use an alternative expression of the Schatten k-norm for positive semidefinite matrices as the trace of the k-th power of M, i.e. $(\|M\|_k)^k = \mathrm{Tr}(M^k)$. This sum of the entries along the diagonal of M^k is the sum of total weights of all the closed walks of length k. Consider the entries of M as weights on a complete graph K_d over d nodes (with self-loops). A closed walk of length k is defined as a sequence of nodes $w = (w_1, w_2, \ldots, w_{k+1})$ with $w_1 = w_{k+1}$, where we allow repeated nodes and repeated edges. The *weight* of a closed walk $w = (w_1, \ldots, w_k, w_1)$ is defined as

$\omega_M(w) \equiv \prod_{i=1}^{k} M_{w_i w_{i+1}}$, which is the product of the weights along the walk. It follows that

$$\|M\|_k^k \;=\; \sum_{w:\text{ all length } k \text{ closed walks}} \omega_M(w)\,. \tag{1}$$

Following the notations from enumeration of small simple cycles in a graph by Alon et al. (1997), we partition this summation into those with the same pattern H that we call a *k-cyclic pseudograph*. Let $C_k = (V_k, E_k)$ denote the undirected simple cycle graph with k nodes, e.g. A_3 in Figure 2 is C_3. We expand the standard notion of simple k-cyclic graphs to include multi-edges and loops, hence the name *pseudograph*.

Definition 1 We define an unlabelled and undirected pseudograph $H = (V_H, E_H)$ to be a *k-cyclic pseudograph* for $k \geq 3$ if there exists an onto node-mapping from $C_k = (V_k, E_k)$, i.e. $f : V_k \to V_H$, and a one-to-one edge-mapping $g : E_k \to E_H$ such that $g(e) = (f(u_e), f(v_e))$ for all $e = (u_e, v_e) \in E_k$. We use $\mathcal{H}_k$ to denote the set of all k-cyclic pseudographs. We use $c(H)$ to the number of different node mappings f from C_k to a k-cyclic pseudograph H.



$c(A_1) = 1 \quad c(A_2) = 3 \quad c(A_3) = 6$

Figure 2: The 3-cyclic pseudographs $\mathcal{H}_3 = \{A_1, A_2, A_3\}$.

In the above example, each member of $\mathcal{H}_3$ is a distinct pattern that can be mapped from C_3. For A_1, it is clear that there is only one mapping from C_3 to A_1 (i.e. $c(A_1) = 1$). For A_2, one can map any of the three nodes to the left-node of A_2, hence $c(A_2) = 3$. For A_3, any of the three nodes can be mapped to the bottom-left-node of A_3 and also one can map the rest of the nodes clockwise or counter-clockwise, resulting in $c(A_3) = 6$. For $k \leq 7$, all the k-cyclic pseudo graphs are given in the Section 9 (See Figures 10–15).

Each closed walk w of length k is associated with one of the graphs in $\mathcal{H}_k$, as there is a unique H that the walk is an Eulerian cycle of (under a one-to-one mapping of the nodes). We denote this graph by $H(w) \in \mathcal{H}_k$. Considering the weight of a walk $\omega_M(w)$, there are multiple distinct walks with the same weight. For example, a length-3 walk $w = (v_1, v_2, v_2, v_1)$ has $H(w) = A_2$ and there are 3 walks with the same weight $\omega(w) = (M_{v_1v_2})^2 M_{v_2v_2}$, i.e. (v_1, v_2, v_2, v_1), (v_2, v_2, v_1, v_2), and (v_2, v_1, v_2, v_2). This multiplicity of the weight depends only on the structure $H(w)$ of a walk, and it is exactly $c(H(w))$ the number of mappings from C_k to $H(w)$ in Definition 1. The total sum of the weights of closed walks of length k can be partitioned into their respective pattern, which will make computation of such terms more efficient (see Section 2) and also de-biasing straight forward (see Equation (3)):

$$\|M\|_k^k \;=\; \sum_{H\in\mathcal{H}_k} \omega_M(H)\, c(H)\,, \tag{2}$$

where with a slight abuse of a notation, we let $\omega_M(H)$ for $H \in \mathcal{H}_k$ be the sum of all *distinct* weights of walks w with $H(w) = H$, and $c(H)$ is the multiplicity of each of those distinct weights. This gives an alternative tool for computing the Schatten k-norm without explicitly computing the singular values.

Given only the access to a subset of sampled entries, one might be tempted to apply the above formula to the sampled matrix with an appropriate scaling, i.e.

$$\left\| \frac{d^2}{|\Omega|} \mathcal{P}_\Omega(M) \right\|_k^k = \frac{d^2}{|\Omega|} \sum_{H \in \mathcal{H}_k} \omega_{\mathcal{P}_\Omega(M)}(H)\, c(H)\,,$$

to estimate $\|M\|_k^k$. However, this is significantly biased. To eliminate the bias, we propose rescaling each term in (1) by the inverse of the probability of sampling that particular walk w (i.e. the probability that all edges in w are sampled). A crucial observation is that, for any sampling model that is invariant under a relabelling of the nodes, this probability only depends on the pattern $H(w)$. In particular, this is true for the Erdös-Rényi sampling. Based on this observation, we introduce a novel estimator that de-biases each group separately:

$$\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) = \sum_{H \in \mathcal{H}_k} \frac{1}{p(H)}\, \omega_{\mathcal{P}_\Omega(M)}(H)\, c(H)\,, \tag{3}$$

where $p(H)$ is the probability a pattern H is sampled, i.e. all edges traversed in a walk w with $H(w) = H$ is sampled. It immediately follows that this estimator is unbiased, i.e. $\mathbb{E}_\Omega[\widehat{\Theta}_k(\mathcal{P}_\Omega(M))] = \|M\|_k^k$, where the randomness is in Ω. However, computing this estimate can be challenging. Naive enumeration over all closed walks of length k takes time scaling as $O(d\,\Delta^{k-1})$, where Δ is the maximum degree of the graph. Except for extremely sparse graphs, this is impractical. Inspired by the work of Alon et al. (1997) in counting short cycles in a graph, we introduce a novel and efficient method for computing the proposed estimate for small values of k.

Proposition 2 *For a positive semidefinite matrix M and any sampling pattern Ω, the proposed estimate $\widehat{\Theta}_k(\mathcal{P}_\Omega(M))$ in (3) can be computed in time $O(d^\alpha)$ for $k \in \{3, 4, 5, 6, 7\}$, where $\alpha < 2.373$ is the exponent of matrix multiplication. For $k = 1$ or 2, $\widehat{\Theta}_k(\mathcal{P}_\Omega(M))$ can be computed in time $O(d)$ and $O(d^2)$, respectively.*

This bound holds regardless of the degree, and the complexity can be even smaller for sparse graphs as matrix multiplications are more efficient. We give a constructive proof by introducing a novel algorithm achieving this complexity in Section 2. For $k \geq 8$, our approach can potentially be extended, but the complexity of the problem fundamentally changes as it is at least as hard as counting K_4 in a graph, for which the best known run time is $O(d^{\alpha+1})$ for general graphs (Kloks et al., 2000).

We make the following contributions in this paper:

- We give in (3) an unbiased estimator of the Schatten k-norm of a positive semidefinite matrix M, from a random sampling of its entries. In general, the complexity of computing the estimate scales as $O(d\Delta^{k-1})$ where Δ is the maximum degree (number of sampled entries in a column) in the sampled matrix. We propose a novel efficient algorithm for computing the estimate in (3) exactly for small $k \leq 7$, which involves only matrix operations. This algorithm is significantly more efficient and has run-time scaling as $O(d^\alpha)$ independent of the degree and for all $k \leq 7$ (see Proposition 2) .

- Under the typical Erdös-Rényi sampling, we show that the Schatten k-norm of an incoherent rank-r matrix can be approximated within any constant multiplicative error, with number of samples scaling as $O(dr^{1-2/k})$ (see Theorem 3). In particular, this is strictly smaller than the number of samples necessary to complete the matrix, which scales as $O(dr \log d)$. Below this matrix completion threshold, numerical experiments confirm that the proposed estimator significantly outperforms simple heuristics of using singular values of the sampled matrices directly or applying state-of-the-art matrix completion methods (see Figure 4).

- Given the estimation of first K Schatten norms, it is straight forward to approximate spectral sum functions of the form (4) using Chebyshev's expansion, and also estimate the spectrum itself using moment matching in Wasserstein distance. We apply our Schatten norm estimates to the applications of estimating the generalized rank studied in Zhang et al. (2015) and estimating the spectrum studied in Kong and Valiant (2016). We provide performance guarantees for both applications and provide experimental results suggesting we improve upon other competing methods.

- We propose a new sampling model, which we call *graph sampling*, that preserves the structural properties of the pattern of the samples. We identify a fundamental property of the structure of the pattern ($\lambda^*_{G,r}$ in Eq.(14)) that captures the difficulty of estimating the Schatten k-norm from such graph sampling (see Theorem 11). Under this graph sampling, we show that there are sampling patterns that are significantly more efficient, for estimating the spectral properties, than Erdös-Rényi sampling.

In the remainder of this section, we review existing works in Schatten norm approximation, and provide an efficient implementation of the estimator (3) for small k in Section 2. In Section 3, we provide a theoretical analysis of our estimator under the Erdös-Rényi sampling scenario. In Section 4, we provide a theoretical analysis under the graph sampling scenario. We conclude with a discussion on interesting observations and remaining challenges in Section 5.

1.3. Related work

We review the existing methods in approximating the Schatten norms, counting small structures in graphs, and various applications of Schatten norms.

Estimating k-Schatten norms of a data matrix. The proposed Schatten norm estimator can be used as a black box in various applications where we want to test the property of a data matrix or a network but limited to observe only a small portion of the data. These include, for example, network forensics, matrix spectral property testing, and testing for graph isospectral properties. Relatively little is known under the matrix completion setting studied in this paper. However, Schatten norm estimation under different resource constrained scenarios have been studied. Hutchinson (1990) propose a randomized algorithm

for approximating the trace of any large matrix, where the constraint is on the computational complexity. The goal is to design a random rank-one linear mapping such that the trace is preserved in expectation and the variance is small (Avron and Toledo, 2011; Roosta-Khorasani and Ascher, 2015). Li et al. (2014) propose an optimal bilinear sketching of a data matrix, where the constraint is on the memory, i.e. the size of the resulting sketch. The goal is to design a sketch of a data matrix M using minimal storage and a corresponding approximate reconstruction method for $\|M\|_k^k$. Li and Woodruff (2016) propose an optimal streaming algorithm where only one-pass on the data is allowed in a data stream model and the constraint is on the space complexity of the algorithm. The goal is to design a streaming algorithm using minimal space to estimate $\|M\|_k^k$. Zhang et al. (2015) propose an estimator under a distributed setting where columns of the data are store in distributed storage and the constraint is on the communication complexity. The goal is to design a distributed protocol minimizing the communication to estimate $\|M\|_k^k$. Given a random vector X, Kong and Valiant (2016) propose an optimal estimator for the Schatten k-norm of the covariance matrix, where the constraint is on the number of samples n. The goal is to design an estimator using minimum number of samples to estimate $\|\mathbb{E}[XX^T]\|_k^k$.

One of our contributions is that we propose an efficient algorithm for computing the weighted counts of small structures in Section 2, which can significantly improve upon less sample-efficient counterpart in, for example, (Kong and Valiant, 2016). Under the setting of (Kong and Valiant, 2016) (and also (Li et al., 2014)), the main idea of the estimator is that the weight of each length-k cycle in the observed empirical covariance matrix $(1/n)\sum_{i=1}^n X_i X_i^T$ provides an unbiased estimator of $\|\mathbb{E}[XX^T]\|_k^k$. One prefers to sum over the weights of as many cycles as computationally allowed in order to reduce the variance. As counting all cycles is in general computationally hard, they propose counting only increasing cycles (which only accounts for only $1/k!$ fraction of all the cycles), which can be computed in time $O(d^\alpha)$. If one has an efficient method to count all the (weighted) cycles, then the variance of the estimator could potentially decrease by an order of $k!$. For $k \le 7$, our proposed algorithm in Section 2 provides exactly such an estimator. We replace (Kong and Valiant, 2016, Algorithm 1) with ours, and run the same experiment to showcase the improvement in Figure 3, for dimension $d = 2048$ and various values of number of samples n comparing the multiplicative error in estimating $\|\mathbb{E}[XX^T]\|_k^k$, for $k = 7$. With the same run-time, significant gain is achieved by simply substituting our proposed algorithm for counting small structures, in the sub-routine. In general, the efficient algorithm we propose might be of independent interest to various applications, and can directly replace (and significantly improve upon) other popular but less efficient counterparts.

One of the main challenges under the sampling scenario considered in this paper is that existing counting methods like that of (Kong and Valiant, 2016) cannot be applied, regardless of how much computational power we have. Under the matrix completion scenario, we need to (a) sum over all small structures $H \in \mathcal{H}_k$ and not just C_k as in (Kong and Valiant, 2016); and (b) for each structure we need to sum over all subgraphs with the same structure and not just those walks whose labels form a monotonically increasing sequence as in (Kong and Valiant, 2016).

Algorithms for counting structures. An important problem in graph theory is to count the number of small structures, also called network motifs, in a given graph. This has many

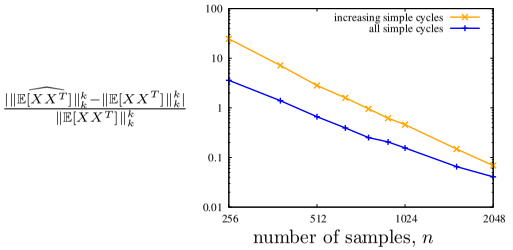


Figure 3: By replacing Algorithm 1 in (Kong and Valiant, 2016) that only counts increasing cycles with our proposed algorithm that counts all cycles, significant gain is acheived in estimating $\|\mathbb{E}[XX^T]\|_k^k$, for $k = 7$.

practical applications in designing good LDPC codes (Tian et al., 2004), understanding the properties of social networks (Ugander et al., 2013), and explaining gene regulation networks (Shen-Orr et al., 2002). Exact and approximate algorithms have been proposed in (Alon et al., 1997; Kloks et al., 2000; Liu and Wang, 2006; Halford and Chugg, 2006; Karimi and Banihashemi, 2013; Wang et al., 2014). The most relevant one is the work of Alon et al. (1997) on counting the number of cycles C_k, where counts of various small structures called k-cyclic graphs are used as sub-routines and efficient approaches are proposed for $k \le 7$. These are similar to k-cyclic pseudographs, but with multi-edges condensed to a single edge. When counting cycles in a simple (unweighted) graph, k-cyclic graphs are sufficient as all the edges have weight one. Hence, one does not need to track how many times an edge has been traversed; the weight of that walk is always one. In our setting, the weight of a walk depends on how many times an edge has been traversed, which we track using multiedges. It is therefore crucial to introduce the class of k-cyclic pseudographs in our estimator.

In a distributed environment, fast algorithms for counting small structures have been proposed by Elenberg et al. (2015) and Elenberg et al. (2016) for small values of $k \in \{3, 4\}$. However, the main strength of their approach is in distributed computing, and under the typical centralized setting we study, this approach can be slower by a factor exponential in k for, say $k \le 7$.

From Schatten norms to spectral sum functions. A dominant application of Schatten norms is in approximating a family of functions of a matrix, which are called *spectral sum functions* (Han et al., 2016) of the form

$$F(M; f) \equiv \sum_{i=1}^{d} f(\sigma_i(M)) \simeq \sum_{k=0}^{K} a_k \Big\{ \sum_{i=1}^{d} \sigma_i(M)^k \Big\} . \tag{4}$$

A typical approach is to compute the coefficients of a Chebyshev approximation of f, which immediately leads to an approximation of the spectral sum function of interest as the weighted sum of Schatten k-norms. This follows from the approximation of $f(x) \simeq$

$\sum_{k=0}^{K} a_k x^k$. This approach has been widely used in fast methods for approximating the log-determinant (Pace and LeSage, 2004; Zhang and Leithead, 2007; Boutsidis et al., 2015; Aune et al., 2014; Han et al., 2015), corresponding to $f(x) = \log x$. Practically, log-determinant computations are routinely (approximately) required in applications including Gaussian graphical models (Rue and Held, 2005), minimum-volume ellipsoids (Van Aelst and Rousseeuw, 2009), and metric learning (Davis et al., 2007). Fast methods for approximating trace of matrix inverse has been studied in (Wu et al., 2016; Chen, 2016), corresponding to $f(x) = x^{-1}$, motivated by applications in lattice quantum chromodynamics (Stathopoulos et al., 2013). Fast methods for approximating the Estarada index has been studied in (Han et al., 2016), corresponding to $f(x) = \exp(x)$. Practically, it is used in characterizing 3-dimensional molecular structure (Estrada, 2000) and measuring graph centrality (Estrada and Hatano, 2007), the entropy of a graph (Carbó-Dorca, 2008), and the bipartivity of a graph (Estrada and Rodriguez-Velázquez, 2005). Approximating the generalized rank under communication constraints has been studied in (Zhang et al., 2015), corresponding to $f(x; c_1) = \mathbb{I}(x \leq c_1)$. The generalized rank approximates a necessary tuning parameter in a number of problems where low-rank solutions are sought including robust PCA (Candès et al., 2011; Netrapalli et al., 2014) and matrix completion (Keshavan et al., 2010b,a; Jain et al., 2013), and also is required in sampling based methods in numerical analysis (Mahoney et al., 2011; Halko et al., 2011). Similarly, (Saade et al., 2015) studied the number of singular values in an interval, corresponding to $f(x; c_1, c_2) = \mathbb{I}(c_1 \leq x \leq c_2)$. In practice, a number of eigensolvers (Polizzi, 2009; Sakurai and Sugiura, 2003; Schofield et al., 2012) require the number of eigenvalues in a given interval. For more comprehensive list of references and applications of this framework, we refer to the related work section in (Han et al., 2016).

In a recent work, Kong and Valiant (2016) provide a novel approach to tackle the challenging problem of estimating the singular values themselves. Considering the histogram of the singular values as a one-dimensional distribution and the Schatten k-norm as the k-th moment of this distribution, the authors provide an innovative algorithm to estimate the histogram that best matches the moments in Wasserstein distance.

Matrix completion. Low-rank matrix completion addresses the problem of recovering a low-rank matrix from its sampled entries. Tight lower and upper bounds on the sample complexity is well studied in both cases where you want exact recovery when samples are noiseless (Candès and Recht, 2009; Keshavan et al., 2010a; Bhojanapalli and Jain, 2014), and also when samples are noisy and where you want approximate recovery (Keshavan et al., 2010b; Negahban and Wainwright, 2012). In practical applications, one might not have enough samples to estimate all the missing entries with sufficient accuracy. However, one might still be able to infer important spectral properties of the data, such as the singular values or the rank. Such spectral properties can also assist in making decisions on how many more samples to collect in order to make accurate inference on the quantity of interest. In this paper, one of the fundamental question we ask and answer affirmatively is: Can we accurately recover the spectral properties of a low-rank matrix from sampling of its entries, below the matrix completion threshold?

2. Efficient Algorithm

In this section we give a constructive proof of Proposition 2, inspired by the seminal work of Alon et al. (1997) and generalize their counting algorithm for k-cyclic graphs to counting (weighted) k-cyclic pseudographs. In computing the estimate in (3), $c(H)$ can be computed in time $O(k!)$. Suppose $p(H)$ has been computed. We will explain how to compute $p(H)$ for Erös-Rényi sampling and graph sampling in Sections 3 and 4. The bottleneck then is computing the weights $\omega_{\mathcal{P}_\Omega(M)}(H)$ for each $H \in \mathcal{H}_k$. Let $\gamma_M(H) \equiv \omega_M(H)c(H)$. We give matrix-multiplication-based equations to compute $\gamma_M(H)$ for every $H \in \mathcal{H}_k$ for $k \in \{3, 4, 5, 6, 7\}$. This establishes that $\gamma_M(H)$, and hence $\omega_M(H)$, can be computed in time $O(d^\alpha)$, proving Proposition 2.

For any matrix $A \in \mathbb{R}^{d\times d}$, let $\mathrm{diag}(A)$ to be a diagonal matrix such that $(\mathrm{diag}(A))_{ii} = A_{ii}$, for all $i \in [d]$ and $(\mathrm{diag}(A))_{i,j} = 0$, for all $i \neq j \in [d]$. For a given matrix $M \in \mathbb{R}^{d\times d}$, define O_M to be matrix of off-diagonal entries of M that is $O_M \equiv M - \mathrm{diag}(M)$ and $D_M \equiv \mathrm{diag}(M)$. Let $\mathrm{tr}(A)$ denote the trace of A, that is, $\mathrm{tr}(A) = \sum_{i\in[d]} A_{ii}$. Let $A*B$ denote the standard matrix multiplication of two matrices A and B. Consider computing $\gamma_M(H)$ for $H \in \mathcal{H}_3$ as labeled in Figure 2:

$$\gamma_M(A_1) = \mathrm{tr}(D_M{*}D_M{*}D_M) \quad (5)$$

$$\gamma_M(A_2) = 3\,\mathrm{tr}(D_M{*}O_M{*}O_M) \quad (6)$$

$$\gamma_M(A_3) = \mathrm{tr}(O_M{*}O_M{*}O_M) \quad (7)$$

The first weighted sum $\gamma_M(A_1)$ is the sum of all weights of walks of length 3 that consist of three self-loops. One can show that $\gamma_M(A_1) = \sum_{i\in[d]} M_{ii}^3$, which is (5) in our matrix operation notations. Similarly, $\gamma_M(A_3)$ is the sum of weights of length 3 walks with no self-loop, which leads to (7) and, $\gamma_M(A_2)$ is the sum of weights of length 3 walks with a single self-loop, which leads to (6). The factor 3 accounts for the fact that the self loop could have been placed at the first, second or the third in the walk.

Similarly, for each k-cyclic pseudographs in $\mathcal{H}_k$ for $k \leq 7$, computing $\gamma_M(H)$ involves a few matrix operations with run-time $O(d^\alpha)$. We provide the complete set of explicit expressions in Section 10. A MATLAB implementation of the estimator (3), that includes as its sub-routines the computation of the weights of all k-cyclic pseudographs, is available for download at `https://github.com/khetan2/Schatten_norm_estimation`. The explicit formulae in Section 10 together with the implementation in the above url might be of interest to other problems involving counting small structures in graphs.

For $k = 1$, the estimator simplifies to $\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) = (1/p)\sum_i \mathcal{P}_\Omega(M)_{ii}$, which can be computed in time $O(d)$. For $k = 2$, the estimator simplifies to $\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) = (1/p)\sum_{i,j} \mathcal{P}_\Omega(M)_{ij}^2$, which can be computed in time $O(|\Omega|)$. However, for $k \geq 8$, there exist walks over K_4, a clique over 4 nodes, that cannot be decomposed into simple computations involving matrix operations. The best known algorithm for a simpler task of counting K_4 has run-time scaling as $O(d^{\alpha+1})$, which is fundamentally different. We refer to Section 5 for further discussions on the computational complexity beyond $k = 7$.

Algorithm 1 Schatten k-norm estimator

Require: $\mathcal{P}_\Omega(M)$, k, $\mathcal{H}_k$, $p(H)$ for all $H \in \mathcal{H}_k$
Ensure: $\widehat{\Theta}_k(\mathcal{P}_\Omega(M))$
1: **if** $k \leq 7$ **then**
2: For each $H \in \mathcal{H}_k$, compute $\gamma_{\mathcal{P}_\Omega(M)}(H)$ using the formula from Eq. (5)–(7) for $k = 3$ and Eq. (60) – (203) for $k \in \{4, 5, 6, 7\}$
3: $\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) \leftarrow \sum_{H \in \mathcal{H}_k} \frac{1}{p(H)} \gamma_{\mathcal{P}_\Omega(M)}(H)$
4: **else**
5: $\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) \leftarrow$ Algorithm 4$[\mathcal{P}_\Omega(M)$, k, $\mathcal{H}_k$, $p(H)$ for all $H \in \mathcal{H}_k]$ [Section 6]
6: **end if**

3. Erdös-Rényi sampling

Under the stylized but canonical Erdös-Rényi sampling, notice that the probability $p(H)$ that we observe all edges in a walk with pattern H is

$$p(H) \;=\; p^{m(H)}\,,$$

where p is the probability an edge is sampled and $m(H)$ is the number of distinct edges in a k-cyclic pseudograph H. Plugging in this value of $p(H)$, which can be computed in time linear in k, into the estimator (3), we get an estimate customized for Erdös-Rényi sampling.

Given a rank-r matrix M, the difficulty of estimating properties of M from sampled entries is captured by the *incoherence* of the original matrix M, which we denote by $\mu(M) \in \mathbb{R}$ (Candès and Recht, 2009). Formally, let $M \equiv U\Sigma U^\top$ be the singular value decomposition of a positive definite matrix where U is a $d \times r$ orthonormal matrix and $\Sigma \equiv \mathrm{diag}(\sigma_1, \cdots, \sigma_r)$ with singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$. Let $U_{i,r}$ denote the i-th row and j-th column entry of matrix U. The incoherence $\mu(M)$ is defined as the smallest positive value μ such that the following holds:

A1. For all $i \in [d]$, we have $\sum_{a=1}^r U_{ia}^2(\sigma_a/\sigma_1) \leq \mu r/d$.

A2. For all $i \neq j \in [d]$, we have $|\sum_{a=1}^r U_{ia}U_{ja}(\sigma_a/\sigma_1)| \leq \mu\sqrt{r}/d$.

The incoherence measures how well the matrix is spread out and is a common measure of difficulty in completing a matrix from random samples (Candès and Recht, 2009; Keshavan et al., 2010a). We note that our incoherence condition depends upon singular values unlike the one given in (Candès and Recht, 2009) which only depends upon row and column spaces. The lower the incoherence, the more spread out the entries are, and estimation is easier. On the other hand, if there are a few entries that are much larger than the rest, estimating a property of the matrix (such as the Schatten k-norm) from uniformly sampled entries can be extremely challenging.

3.1. Performance guarantee

For any $d \times d$ positive semidefinite matrix M of rank r with incoherence $\mu(M) = \mu$ and the effective condition number $\kappa = \sigma_{\max}(M)/\sigma_{\min}(M)$, we define

$$\rho^2 \;\equiv\; (\kappa\mu)^{2k} g(k) \max\left\{1, \frac{(dp)^{k-1}}{d}, \frac{r^k p^{k-1}}{d^{k-1}}\right\},$$

such that the variance of our estimator is bounded by $\mathrm{Var}(\widehat{\Theta}(\mathcal{P}_\Omega(M))/\|M\|_k^k) \leq \rho^2(r^{1-2/k}/dp)^k$ as we show in the proof of Theorem 3 in Section 8.1. Here, $g(k) = O(k!)$ is a function depending only on k. Note that for $k \geq 3$, ρ^2 simplifies to $\rho^2 = (\kappa\mu)^{2k} g(k) \max\left\{1, \frac{(dp)^{k-1}}{d}\right\}$.

Theorem 3 (Upper bound under the Erdös-Rényi sampling) *For any integer $k \in [3, \infty)$, any $\delta > 0$, any rank-r positive semidefinite matrix $M \in \mathbb{R}^{d\times d}$, and given i.i.d. samples of the entries of M with probability p, the proposed estimate of* (3) *achieves normalized error δ with probability bounded by*

$$\mathbb{P}\left(\frac{|\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) - \|M\|_k^k|}{\|M\|_k^k} \;\geq\; \delta\right) \;\leq\; \frac{\rho^2}{\delta^2}\Big(\frac{r^{1-2/k}}{dp}\Big)^k. \tag{8}$$

Consider a typical scenario where μ, κ, and k are finite with respect to d and r, and rank is sufficiently small, $r = O(d^{k/((k-1)(k-2))})$. Then the Chebyshev's bound in (8) implies that the sample complexity of $d^2p = O(dr^{1-2/k})$ is sufficient to recover $\|M\|_k^k$ up to arbitrarily small multiplicative error and arbitrarily small (but strictly positive) error probability. This is strictly less than the known minimax sample complexity for recovering the entire low-rank matrix, which scales is $\Theta(rd\log d)$. As we seek to estimate only a property of the matrix (i.e. the Schatten k-norm) and not the whole matrix itself, we can be more efficient on the sample complexity by a factor of $r^{2/k}$ in rank and a factor of $\log d$ in the dimension. We emphasize here that such a gain can only be established using the proposed estimator based on the structure of the k-cyclic pseudographs. We will show empirically that the standard matrix completion approaches fail in the critical regime of samples below the recovery threshold of $O(rd\log d)$.

Figure 4 is a scatter plot of the absolute relative error in estimated Schatten k-norm, $\big|\|M\|_k^k - \widehat{\|M\|_k^k}\big|/\|M\|_k^k$, for $k = 5$, for three approaches: the proposed estimator, Schatten norm of the scaled sampled matrix (after rank-r projection), and Schatten norm of the completed matrix, using state-of-the-art alternating minimization algorithm (Jain et al., 2013). All the three estimators are evaluated 20 times for each value of p. M is a symmetric positive semi-definite matrix of size $d = 500$, and rank $r = 100$ (left panel) and $r = 500$ (right panel). Singular vectors U of $M = U\Sigma U^\top$, are generated by QR decomposition of $\mathcal{N}(0, \mathbb{I}_{d\times d})$ and $\Sigma_{i,i}$ is uniformly distributed over $[1, 2]$. For a low rank matrix on the left, there is a clear critical value of $p \simeq 0.45$, above which matrix completion is exact with high probability. However, this algorithm knows the underlying rank and crucially exploits the fact that the underlying matrix is exactly low-rank. In comparison, our approach is agnostic to the low-rank assumption but finds the accurate estimate that is adaptive to the actual rank in a data-driven manner. Using the first r singular values of the (rescaled) sampled matrix fails miserably for all regimes (we truncate the error at one for illustration purposes). In this paper, we are interested in the regime where exact matrix completion is impossible as we do not have enough samples to exactly recover the underlying matrix: $p \leq 0.45$ in the left panel and all regimes in the right panel.

The sufficient condition of $d^2p = O(dr^{1-2/k})$ in Theorem 3 holds for a broad range of parameters where the rank is sufficiently small $r = O(d^{k/((k-1)(k-2))})$ (to ensure that

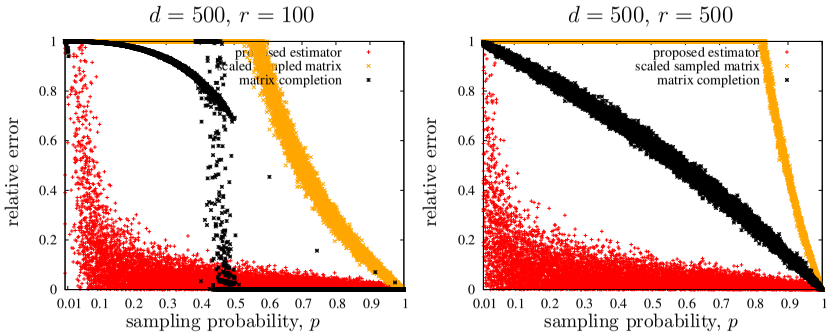


Figure 4: The proposed estimator outperforms both baseline approaches below the matrix completion threshold. For $k = 5$, comparison of the absolute relative error in estimated Schatten norm that is $\big|\|M\|_k^k - \widehat{\|M\|_k^k}\big|/\|M\|_k^k$ for the three algorithms: (1) the proposed estimator, $\widehat{\|M\|_k^k} = \widehat{\Theta}_k(\mathcal{P}_\Omega(M))$, (2) Schatten norm of the scaled sampled matrix, $\widehat{\|M\|_k^k} = \|(1/p)\mathcal{P}_r(\mathcal{P}_\Omega(M))\|_k^k$, (3) Schatten norm of the completed matrix, $\widetilde{M} = \text{AltMin}(\mathcal{P}_\Omega(M))$ from (Jain et al., 2013), $\widehat{\|M\|_k^k} = \|\widetilde{M}\|_k^k$, where $\mathcal{P}_r(\cdot)$ is the standard best rank-r projection of a matrix. Ω is generated by Erdös-Rényi sampling of matrix M with probability p.

the first term in ρ^2 dominates). However, the following results in Figure 5 on numerical experiments suggest that our analysis holds more generally for all regimes of the rank r, even those close to d. M is generated using settings similar to that of Figure 4. Empirical probabilities are computed by averaging over 100 instances.

One might hope to tighten the Chebyshev bound by exploiting the fact that the correlation among the summands in our estimator (3) is weak. This can be made precise using the recent result from Schudy and Sviridenko (2011), where a Bernstein-type bound was proved for the sum of polynomials of independent random variables that are weakly correlated. The first term in the bound (9) is the natural Bernstein-type bound corresponding to the Chebyshev's bound in (8). However, under the regime where k is large or p is large, the correlation among the summands becomes stronger, and the second and third term in the bound (9) start to dominate. In the typical regime of interest where μ, κ, k are finite, rank is sufficiently small $r = O(d^{k/((k-1)(k-2))})$, and sample complexity $d^2p = O(dr^{1-2/k})$, the error probability is dominated by the first term on the right-hand side of (9). Neither one of the two bounds in (8) and (9) dominates the other, and depending on the values of the problem parameters, we might want to apply the one that is tighter. We provide a proof in Section 8.2.

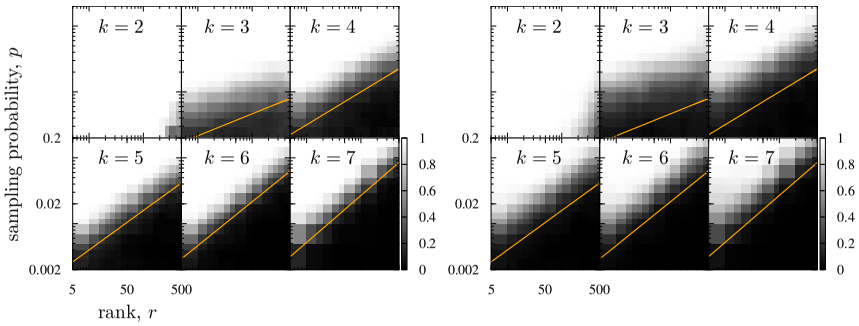


Figure 5: Each colormap in each block for $k \in \{2, 3, 4, 5, 6, 7\}$ show empirical probability of the event $\big\{\big|\|M\|_k^k - \widehat{\Theta}_k(\mathcal{P}_\Omega(M))\big|/\|M\|_k^k \le \delta\big\}$, for $\delta = 0.5$ (left panel) and $\delta = 0.2$ (right panel). Ω is generated by Erdös-Rényi sampling of matrix M with probability p (vertical axis). M is a symmetric positive semi-definite matrix of size $d = 1000$. The solid lines correspond to our theoretical prediction $p = (1/d)r^{1-2/k}$.

Theorem 4 *Under the hypotheses of Theorem 3, the error probability is upper bounded by*

$$\mathbb{P}\left(\frac{\big|\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) - \|M\|_k^k\big|}{\|M\|_k^k} \ge \delta\right) \le$$
$$e^2 \max\left\{e^{-\frac{\delta^2}{\rho^2}\left(\frac{dp}{r^{1-2/k}}\right)^k}, e^{-(dp)\left(\frac{\delta d}{\rho r^{k-1}}\right)^{1/k}}, e^{-(dp)\left(\frac{\delta d}{\rho r^{k-1}}\right)}, e^{-\frac{\delta dp}{\rho}}\right\}. \quad (9)$$

These two results show that the sample size of $d^2p = O(dr^{1-2/k})$ is sufficient to estimate a Schatten k-norm accurately when μ, κ, k are finite and rank is sufficiently small $r = O(d^{k/((k-1)(k-2))})$. In general, we do not expect to get a universal upper bound that is significantly tighter for all r, because for a special case of $r = d$, the following corollary of (Li et al., 2014, Theorem 3.2) provides a lower bound; it is necessary to have sample size $d^2p = O(d^{2-4/k})$ when $r = d$. Hence, the gap is at most a factor of $r^{2/k}$ in the sample complexity.

Corollary 5 *Consider any linear observation $X \in \mathbb{R}^n$ of a matrix $M \in \mathbb{R}^{d\times d}$ and any estimate $\theta(X)$ satisfying $(1-\delta_k)\|M\|_k^k \le \theta(X) \le (1+\delta_k)\|M\|_k^k$ for any M with probability at least $3/4$, where $\delta_k = (1.2^k - 1)/(1.2^k + 1)$. Then, $n = \Omega(d^{2-4/k})$.*

For $k \in \{1, 2\}$, precise bounds can be obtained with simpler analyses. In particular, we have the following remarks, whose proof follows immediately by applying Chebyshev's inequality and Bernstien's inequality along with the incoherence assumptions.

Remark 6 *For $k = 1$, the probability of error in* (8) *is upper bounded by* $\min\{\nu_1, \nu_2\}$, *where*

$$\nu_1 \equiv \frac{1}{\delta^2}\frac{(\kappa\mu)^2}{dp}\,, \quad \textit{and} \quad \nu_2 \equiv 2\exp\Big(\frac{-\delta^2}{2}\Big(\frac{(\kappa\mu)^2}{dp} + \delta\frac{(\kappa\mu)}{3dp}\Big)^{-1}\Big)\,.$$

Remark 7 *For $k = 2$, the probability of error in* (8) *is upper bounded by* $\min\{\nu_1, \nu_2\}$, *where*

$$\nu_1 \equiv \frac{1}{\delta^2}\frac{(\kappa\mu)^4}{d^2p}\big(2 + \frac{r^2}{d}\big) \;\; \textit{, and} \quad \nu_2 \equiv 2\exp\Big(-\frac{\delta^2}{2}\Big(\frac{(\kappa\mu)^4}{d^2p}\big(2+\frac{r^2}{d}\big) + \delta\frac{(\kappa\mu)^2 r}{3d^2p}\Big)^{-1}\Big)\,.$$

When $k = 2$, for rank small $r \le C\sqrt{d}$, only we only need $d^2p = O(1)$ samples for recovery up to any arbitrary small multiplicative error. When rank r is large, our estimator requires $d^2p = O(d)$ for both $k \in \{1, 2\}$.

3.2. From Schatten norms to spectrum and spectral sum functions

Schatten norms by themselves are rarely of practical interest in real applications, but they provide a popular means to approximate functions of singular values, which are often of great practical interest (Di Napoli et al., 2016; Zhang et al., 2015; Kong and Valiant, 2016). In this section, we consider two such applications using the first few Schatten norms explicitly: estimating the generalized rank in Section 3.2.1 and estimating the singular values in Section 3.2.2.

3.2.1. Estimating the generalized rank

For a matrix $M \in \mathbb{R}^{d\times d}$ and a given constant $c \ge 0$, its *generalized rank* of order c is given by

$$\operatorname{rank}(M, c) = \sum_{i=1}^{d} \mathbb{I}\big[\sigma_i(M) > c\big]\,.$$

This recovers the standard rank as a special case when $c = 0$. Without loss of generality, we assume that $\sigma_{\max}(M) \le 1$. For any given $0 \le c_2 < c_1 \le 1$, and $\delta \in [0, 1)$, our goal is to get an estimate $\widehat{r}(\mathcal{P}_\Omega(M))$ from sampled entries $\mathcal{P}_\Omega(M)$ such that

$$(1-\delta)\operatorname{rank}(M, c_1) \;\le\; \widehat{r}(\mathcal{P}_\Omega(M)) \;\le\; (1+\delta)\operatorname{rank}(M, c_2)\,.$$

The reason we take two different constants c_1, c_2 is to handle the ambiguous case when the matrix M has many eigenvalues smaller than but very close to c_1. If we were to set $c_2 = c_1$, then any estimator $\widehat{r}(M)$ would be strictly prohibited from counting these eigenvalues. However, since these eigenvalues are so close to the threshold, distinguishing them from other eigenvalues just above the threshold is difficult. Setting $c_2 < c_1$ allows us to avoid this difficulty and focus on the more fundamental challenges of the problem.

Consider the function $H_{c_1,c_2} : \mathbb{R} \to [0, 1]$ given by

$$H_{c_1,c_2}(x) = \begin{cases} 1 & \text{if } x > c_1 \\ 0 & \text{if } x < c_2 \\ \frac{x-c_2}{c_1-c_2} & \text{otherwise.} \end{cases}$$

It is a piecewise linear approximation of a step function and satisfies the following:

$$\operatorname{rank}(M, c_1) \;\le\; \textstyle\sum_{i=1}^{d} H_{c_1,c_2}(\sigma_i(M)) \;\le\; \operatorname{rank}(M, c_2)\,. \tag{10}$$

We exploit this sandwich relation and estimate the generalized rank. Given a polynomial function $f : \mathbb{R} \to \mathbb{R}$ of finite degree m such that $f(x) \approx H_{c_1,c_2}(x)$ for all x, such that $f(x) = a_0 + a_1x + \cdots + a_mx^m$, we immediately have the following relation, which extends to a function on the cone of PSD matrices in the standard way:

$$\sum_{i=1}^{d} f(\sigma_i(M)) \;=\; a_0 d + \sum_{k=1}^{m} a_k \|M\|_k^k\,. \tag{11}$$

Using this equality, we propose the estimator:

$$\widehat{r}(\mathcal{P}_\Omega(M); c_1, c_2) \;\equiv\; a_0 d + \sum_{k=1}^{m} a_k \widehat{\Theta}_k(\mathcal{P}_\Omega(M))\,, \tag{12}$$

where we use the first several $\widehat{\Theta}_k(\mathcal{P}_\Omega(M))$'s obtained by the estimator (3). Note that function f depends upon c_1, c_2. The remaining task is to obtain the coefficients of the polynomials in f that is a suitable approximation of the function H_{c_1,c_2}. In a similar context of estimating the generalized rank from approximate Schatten norms, Zhang et al. (2015) propose to use a composite function $f = q_s \circ q$ where q is a finite-degree Chebyshev polynomial of the first kind such that $\sup_{x\in[0,1]} |q(x) - H_{c_1,c_2}(x)| \le 0.1$, and q_s is a polynomial of degree $2s + 1$ given by

$$q_s(x) \;=\; \frac{1}{B(s+1, s+1)}\int_0^x t^s(1-t)^s dt\,, \qquad \text{where } B(\cdot,\cdot) \text{ is the Beta function.}$$

Note that, since H_{c_1,c_2} is a continuous function with bounded variation, classical theory in (Mason and Handscomb, 2002, Theorem 5.7), guarantees existence of the Chebyshev polynomial q of a finite constant degree, say C_b, that depends upon c_1 and c_2. Specifically, for a given choice of thresholds $0 \le c_1 < c_2 \le 1$ and the degree of the beta approximation s, the estimator $\widehat{r}(\mathcal{P}_\Omega(M); c_1, c_2)$ in (12) can be computed as follows.

The approximation of H_{c_1,c_2} with $f = q_s \circ q$ and our upper bound on estimated Schatten norms $\widehat{\Theta}_k(\mathcal{P}_\Omega(M))$ translate into the following guarantee on generalized rank estimator $\widehat{r}(\mathcal{P}_\Omega(M); c_1, c_2)$ given in (12).

Corollary 8 *Suppose $\|M\|_2 \le 1$. Under the hypotheses of Theorem 3, for any given $1 \ge c_1 > c_2 \ge 0$, there exists a constant C_b, such that for any $s \ge 0$ and any $\gamma > 0$, the estimate in* (12) *with the choice of $f = q_s \circ q$ satisfies*

$$(1-\delta)(\operatorname{rank}(M, c_1) - 2^{-s}d) \;\le\; \widehat{r}(\mathcal{P}_\Omega(M); c_1, c_2) \;\le\; (1+\delta)(\operatorname{rank}(M, c_2) + 2^{-s}d)\,,$$

with probability at least $1 - \gamma C_b(2s+1)$, where $\delta \equiv \max_{1\le k\le C_b(2s+1)}\Big\{\sqrt{\frac{\rho^2}{\gamma}(\frac{\max\{1, r^{1-2/k}\}}{dp})^k}\Big\}$.

Algorithm 2 Generalized rank estimator (a variation of Zhang et al. (2015))

Require: $\mathcal{P}_\Omega(M)$, c_1, c_2, s
Ensure: $\widehat{r}(\mathcal{P}_\Omega(M); c_1, c_2)$
1: For given c_1 and c_2, find a Chebyshev polynomial of the first kind $q(x)$ satisfying

$$\sup_{x\in[0,1]} |q(x) - H_{c_1,c_2}(x)| < 0.1$$

[Algorithm 7]
2: Let C_b denote the degree of $q(x)$
3: Find the degree $(2s+1)C_b$ polynomial expansion of $q_s \circ q(x) = \sum_{k=0}^{(2s+1)C_b} a_k x^k$
4: $\widehat{r}(\mathcal{P}_\Omega(M); c_1, c_2) \leftarrow a_0 d + \sum_{k=1}^{(2s+1)C_b} a_k \widehat{\Theta}_k(\mathcal{P}_\Omega(M))$ [Algorithm 1]

The proof follows immediately from Theorem 3 and the following lemma which gives a uniform bound on the approximation error between H_{c_1,c_2} and $f = q_s \circ q$. Lemma 9, together with Equations (10) and (11), provides a (deterministic) functional approximation guarantee of

$$\operatorname{rank}(M, c_1) - d\,2^{-s} \;\le\; \sum_{i=1}^{d} f(\sigma_i(M)) \;\le\; \operatorname{rank}(M, c_1) + d\,2^{-s}\,,$$

for any $c_1 < c_2$ and any choice of s, as long as C_b is large enough to guarantee a 0.1 uniform error bound on the Chebyshev polynomial approximation. Since we can achieve $1 \pm \delta$ approximation on each polynomial in $f(\sigma_i(x))$, Theorem 3 implies the desired Corollary 8. Note that using Remarks 6 and 7, the bounds in (9) hold for $k \in [1, \infty)$ with $r^{1-2/k}$ replaced by $\max\{1, r^{1-2/k}\}$.

Lemma 9 (Zhang et al. (2015), Lemma 1) *Consider the composite polynomial $f(x) = q_s(q(x))$. Then $f(x) \in [0,1]$ for all $x \in [0,1]$, and moreover*

$$|f(x) - H_{c_1,c_2}(x)| \le 2^{-s}\,, \qquad \text{for all } x \in [0, c_2] \cup [c_1, 1]\,.$$

In Figure 6, we evaluate the performance of estimator (12) numerically. We construct a symmetric matrix M of size $d = 1000$ and rank $r = 200$, $\sigma_i \sim \text{Uni}(0, 0.4)$ for $1 \le i \le r/2$, and $\sigma_i \sim \text{Uni}(0.6, 1)$ for $r/2 + 1 \le i \le r$. We estimate $\widehat{r}(\mathcal{P}_\Omega(M); c_1, c_2)$ for Erdös-Rényi sampling Ω, and a choice of $c_2 = 0.5$ and $c_1 = 0.6$, which is motivated by the distribution of σ_i. We use Chebyshev polynomial of degree $C_b = 2$, and $s = 1$ for q_s. That is function f is of degree 6. Accuracy of the estimator can be improved by increasing C_b and s, however that would require estimating higher Schatten norms.

3.2.2. Estimating the spectrum

Given accurate estimates of first K Schatten norms of a matrix M, we can estimate singular values of M using a linear programming based algorithm given in (Kong and Valiant, 2016). In particular, we get the following guarantees on the estimated singular values, whose proof follows directly using the analysis techniques in the proof of (Kong and Valiant, 2016,

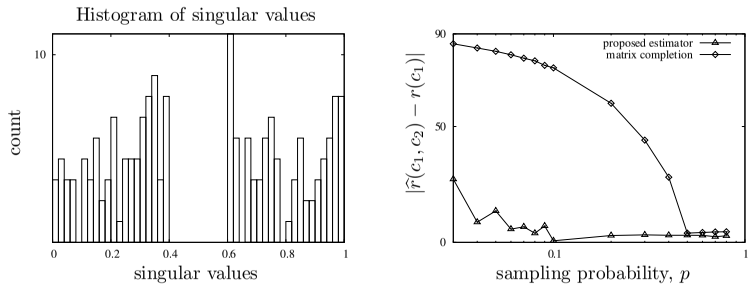


Figure 6: The left panel shows a histogram of singular values of M chosen for the experiment. The right panel compares absolute error in estimation $\widehat{r}(\mathcal{P}_\Omega(M); c_1 = 0.5, c_2 = 0.6)$ for two choices of the Schatten norm estimates $\widehat{\|M\|_k^k}$: first the proposed estimator $\widehat{\Theta}_k(\mathcal{P}_\Omega(M))$ in (3), and second the Schatten norm of the completed matrix, $\widetilde{M} = \text{AltMin}(\mathcal{P}_\Omega(M))$ from (Jain et al., 2013).

Theorem 2). The main idea is that given the rank, the maximum support size of the true spectrum, and an estimate of its first K moments, one can find r singular values whose K first moments are close to the estimated Schatten norms.

Algorithm 3 Spectrum estimator (a variation of Kong and Valiant (2016))

Require: $\mathcal{P}_\Omega(M)$, K, ϵ, target rank r, lower bound a and upper bound b on the positive singular values
Ensure: estimated singular values $(\widehat{\sigma}_1, \widehat{\sigma}_2, \ldots, \widehat{\sigma}_r)$
1: $L \in \mathbb{R}^K : L_k = \widehat{\Theta}_k(\mathcal{P}_\Omega(M))$ for $k \in [K]$ [Algorithm 1]
2: $t = \lceil (b-a)/\epsilon \rceil + 1$, $x \in \mathbb{R}^t$: $x_i = a + \epsilon(i-1)$, for $i \in [t]$,
3: $V \in \mathbb{R}^{K \times t} : V_{ij} = x_j^i$ for $i \in [K], j \in [t]$
4: $p^* \equiv \{\min_{p \in \mathbb{R}^t} |Vp - L|_1 : \mathbb{1}_t^\top p = 1, p \ge 0\}$
5: $\widehat{\sigma}_i = \min\{x_j : \sum_{\ell \le j} p_\ell^* \ge \frac{i}{r+1}\}$, ith $(r+1)$st-quantile of distribution corresponding to p^*

Further, our upper bound on the first K moments can be translated into an upper bound on the Wasserstein distance between those two distributions, which in turn gives the following bound on the singular values. With small enough ϵ and large enough K and r, we need sample size $d^2 p > C_{r,K,\epsilon,\gamma} d r^{1-2/k}$ to achieve arbitrary small error.

Corollary 10 *Under the hypotheses of Theorem 3, given rank r, constants $0 \le a < b$ such that $\sigma_{\min} \ge a$, $\sigma_{\max} \le b$, and estimates of the first K Schatten norms of M, $\{\widehat{\Theta}_k(\mathcal{P}_\Omega(M))\}_{k\in[K]}$ obtained by the estimator (3), for any $0 < \epsilon \ll (b-a)$, and $\gamma > 0$, Algo-*

rithm 3 runs in time $\mathrm{poly}(r, K, (b-a)/\epsilon)$ *and returns* $\{\widehat{\sigma}_i\}_{i\in[r]}$ *an estimate of* $\{\sigma_i(M)\}_{i\in[r]}$ *such that*

$$\frac{1}{r}\sum_{i=1}^{r}|\widehat{\sigma}_i - \sigma_i| \quad \leq$$
$$\frac{C(b-a)}{K} + \frac{b-a}{r} + g(K)(b-a)\left(\epsilon K b^{K-1} + \sum_{k=1}^{K}\sigma_{\max}^k\sqrt{\frac{\rho^2}{\gamma}\left(\frac{\max\{1, r^{1-2/k}\}}{dp}\right)^k}\right),$$

with probability at least $1-\gamma K$*, where* C *is an absolute constant and* $g(K)$ *only depends on* K*.*

In Figure 7, we evaluate the performance of the proposed estimator (3), in recovering the true spectrum using Algorithm 3. We compare the results with the case when Schatten norms are estimated using matrix completion. We consider two distributions on singular values, one peak and two peaks. More general distributions of spectrum can be recovered accurately, however that would require estimating higher Schatten norms. For both cases, the proposed estimator outperforms matrix completion approaches, and achieves better accuracy as sample size increases with α. In each graph, the black solid line depicts the empirical Cumulative Distribution Function (CDF) of the ground truths $\{\sigma_i\}_{i\in[r]}$ for those r strictly positive singular values. In the first experiment (the top panel), there are r singular values at one peak $\sigma_i = 1$, and in the second experiment (the bottom pannel) there are $r/2$ singular values at each of the two peaks at $\sigma_i = 1$ and $\sigma_i = 2$. Each cell shows the result of a choice of rank $r \in \{50, 200, 500\}$ and a parameter $\alpha \in \{3, 5, 8, 10\}$, where Ω is generated using Erdös-Rényi sampling with probability $p = (\alpha/d)r^{1-2/7}$. Matrix M is a symmetric matrix of size $d = 1000$ and rank r with singular values $\{\sigma_i\}_{i\in[d]}$. In each cell, there are one black line, three blue lines, and three orange lines. Each blue line corresponds to the empirical CDF of $\{\widehat{\sigma}_i\}_{i\in[d]}$ for each trial, over three independent trials. Each orange line corresponds to the empirical CDF of $\{\widetilde{\sigma}_i\}_{i\in[d]}$. Here, $\widehat{\sigma}_i$'s are estimated using $\{\widehat{\Theta}_k(\mathcal{P}_\Omega(M))\}_{k\in[K]}$ obtained by the estimator (3), and $\widetilde{\sigma}_i$'s are estimated using $\{\|\widetilde{M}\|_k^k\}_{k\in[K]}$ where $\widetilde{M} = \mathrm{AltMin}(\mathcal{P}_\Omega(M))$, along with Algorithm 2 in (Kong and Valiant, 2016), for $K = 7$.

4. Graph sampling

Our framework for estimating the Schatten k-norms can be applied more generally to any random sampling, as long as the distribution is permutation invariant. In practice, we typically observe one instance of a sampled matrix and do not know how the samples were generated. Under a mild assumption that the probability of sampling an entry is independent of the value of that entry, the only information about the sampling model that we have is the *pattern*, i.e. an unlabelled graph $G = (V, E)$ capturing the pattern of sampled indices by the edges. This naturally suggests a novel sampling scenario that we call *graph sampling*.

The Erdös-Rényi sampling has been criticized as being too strict for explaining how real-world datasets are sampled. When working with natural data, we typically only get

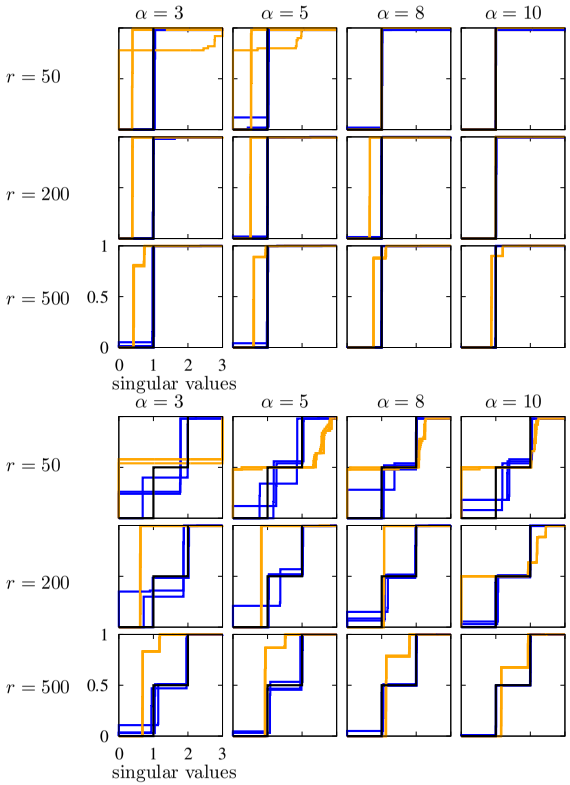


Figure 7: The proposed estimator (in blue solid lines) outperforms matrix completion approaches (in orange solid lines) in estimating the ground truths empirical cumulative distribution function of the r strictly positive singular values (in black solid line) for two examples: one peak at $\sigma_i = 1$ on the top and two peaks at $\sigma_i = 1$ or $\sigma_i = 2$ on the bottom.

one instance of a sampled matrix without the knowledge of how those entries are sampled. In this section, we propose a new sampling model that we call *graph sampling* that makes minimal assumptions about how the data was sampled. We assume that the *pattern* has been determined a priori, which is represented by a deterministic graph $G = (V, E)$ with d nodes denoted by V and undirected edges denoted by E. The random sampling Ω is chosen uniformly at random over all relabeling of the nodes in G. Formally, for a given $G = (V, E)$, a permutation $\pi : [d] \to V$ is drawn uniformly at random and samples are drawn according

to

$$\mathcal{P}_\Omega(M) \;=\; \{(i,j,M_{ij})\}_{(\pi(i),\pi(j))\in E}\,.$$

As the sampling pattern G is completely known to the statistician who only has one instance of a random sampling, we are only imposing that the samples are drawn uniformly at random from all instances that share the same pattern. Further, understanding this graph sampling model has a potential to reveal the subtle dependence of the estimation problem on the underlying pattern, which is known to be hard even for an established area of matrix completion.

In this section, we provide an estimator under graph sampling, and characterize the fundamental limit on the achievable error. This crucially depends on the original pattern G via a fundamental property $\lambda^*_{G,r}$, which is generally challenging to compute. However, we provide a bound on $\lambda^*_{G,r}$ for two extreme cases of varying difficulty: a clique sampling that requires only $O(r^{2-4/k})$ samples and a clique-star sampling that requires as many as $O(dr^{1-4/k})$ samples. This is made formal by showing a lower bound on the minimax sample complexity. Comparing the two necessary conditions on sample complexity, $O(r^{2-4/k})$ for clique sampling and $O(dr^{1-4/k})$ for clique-star sampling, it follows that depending on the pattern of the samples, the sample complexity can vary drastically, especially for low-rank matrices where $r \ll d$.

Under the *graph sampling*, the probability $p(H)$ that we observe all edges in a walk with pattern H is

$$p(H) \;=\; \frac{\omega_{\mathcal{P}_\Omega(\mathbb{1}_d\mathbb{1}_d^T)}(H)}{\omega_{\mathbb{1}_d\mathbb{1}_d^T}(H)}\,,$$

where $\mathbb{1}_d\mathbb{1}_d^T$ is the all ones matrix, and by permutation invariance, the probability is the ratio between total (unweighted) number of walks with $H(w)=H$ in the original pattern Ω and that of the complete graph K_d. Note that although Ω is a random quantity, $\omega_{\mathcal{P}_\Omega(\mathbb{1}\mathbb{1}^T)}(H)$ only depends on the structure and not the labelling of the nodes and hence is a deterministic quantity. Plugging in this value of $p(H)$, which can be computed in time $O(d^\alpha)$ for $k \le 7$ as shown in Proposition 2 (and in general only increases the computational complexity of the estimate by a factor of two), into the estimator (3), we get an estimate customized for graph sampling.

4.1. Performance Guarantees

Recall the graph sampling defined in Section 1.1, where we relabel the nodes of a pattern graph $G(V,E)$ according to a random uniform permutation, and sample the entries of the matrix M on the edges. We prove a fundamental lower bound on the sample complexity that crucially depends on the following property of the pattern G. Let $G_\pi(\widetilde{V},\Omega)$ denote the graph after relabeling the nodes of $G=(V,E)$ with permutation $\pi:[d]\to[d]$. For independent Rademacher variables u_i for $i\in[r]$

$$f_{G,r}(\lambda) \;\equiv\; \max_{\pi}\left\{\mathbb{E}_u\left[\exp\left((5/d)^2\lambda^2\sum_{(i,j)\in\mathcal{P}^{(r)}(G_\pi)}u_iu_j\right)\right]\right\}, \tag{13}$$

where $\mathcal{P}^{(r)}(G_\pi)\subseteq[r]\times[r]$ is a projection of the edges Ω over d nodes to a set of edges over r nodes by mapping a node $i\in[d]$ to a node $1+(i-1 \bmod r)\in[r]$. Precisely, $(i,j)\in\mathcal{P}^{(r)}(G_\pi)$ if there exists an edge $(i',j')\in\Omega$ such that $i=1+(i'-1 \bmod r)$ and $j=1+(j'-1 \bmod r)$. Observe that $f_{G,r}(\lambda)$ is a non-decreasing function of λ. It follows from the fact that for any positive λ and random variable x and any $\epsilon>0$, we have $\mathbb{E}[e^{\lambda(1+\epsilon)x}]\ge\mathbb{E}[e^{\lambda x}](\mathbb{E}[e^{\lambda x}])^\epsilon\ge\mathbb{E}[e^{\lambda x}]e^{\epsilon\lambda\mathbb{E}[x]}\ge\mathbb{E}[e^{\lambda x}]$. The first and the second inequalities use Jensen's inequality and the third one holds when $\mathbb{E}[x]\ge 0$. Note that $\mathbb{E}_u[\sum_{(i,j)\in\mathcal{P}^{(r)}(G_\pi)}u_iu_j]\ge 0$, since u_i's are i.i.d. Rademacher variables.

This function measures the distance between a particular low-rank matrix with Gaussian entries and its rank one perturbation, which is used in our constructive lower bound (see Eq. (35)). Intuitively, smaller $f_{G,r}(\lambda)$ implies that two rank-r matrices with separated Schatten norms look similar after graph sampling w.r.t. G. Hence, when this function is small, say less than $26/25$, then it is hard to distinguish which of the two (distributions of) matrices we are observing. This is captured by the largest value of λ that still maintains $f_{G,r}(\lambda)$ sufficiently small:

$$\lambda^*_{G,r} \;\equiv\; \max_{\{\lambda>0:f_{G,r}(\lambda)\le 26/25\}}\lambda\,. \tag{14}$$

One can choose any number not necessarily $26/25$ as long as it is strictly larger than one and strictly smaller than two, and this will only change the probability upper bound in (15). If we sample from a graph G with large $\lambda^*_{G,r}$, then we cannot distinguish two distributions even if they have a large Schatten norm separation. We do not have enough samples and/or our pattern is not sample efficient. The dependence of the fundamental lower bound on the graph G is captured by this quantity $\lambda^*_{G,r}$, which is made precise in the following theorem. We provide a lower bound that captures how sample complexity depends on the pattern G and also on the underlying matrix, by providing analysis customized for each family of matrices $\mathcal{M}_{r,\mu}$ parametrized by its rank and incoherence:

$$\mathcal{M}_{r,\mu}\equiv\big\{M\in\mathbb{R}^{d\times d}: M=M^\top, \mathrm{rank}(M)\le r\,,\ \mu(M)\le\mu\big\}.$$

Theorem 11 (General lower bound under graph sampling) *For any $k\in[3,\infty)$ suppose we observe samples under the graph sampling defined in Section 1.1 with respect to a pattern graph $G=(V,E)$. Then there exist universal constants $C>0$, $C'>0$ and $C''>0$ such that for any $r\ge e^{C''k}$ and $\mu\ge C'\sqrt{\log r}$, if $\lambda^*_{G,r}\ge Cdr^{1/k-1/2}$ then*

$$\inf_{M\in\mathcal{M}_{r,\mu}}\sup_{\widetilde{\Theta}}\;\mathbb{P}\left(\frac{1}{2}\|M\|_k\le\widetilde{\Theta}(\mathcal{P}_{\Omega(M)})\le 2\|M\|_k\right)\;\le\;\frac{3}{4}\,, \tag{15}$$

where the supremum is over any measurable function of $\mathcal{P}_{\Omega(M)}$ and the probability is with respect to the random sampling Ω.

A proof of Theorem 11 is given in Section 8.3. It is in general challenging to evaluate $\lambda^*_{G,r}$ for a given graph. For a special case of *clique sampling*, where the pattern $G(V,E)$ is a clique over a subset of ℓ nodes among d, we provide a sharp upper bound on $\lambda^*_{G,r}$.

Lemma 12 (Lower bound for clique sampling) *If the pattern graph $G(V,E)$ is a clique over a subset of ℓ nodes, then $\lambda^*_{G,r}\ge 2^{-4}d(\min\{\ell,r\})^{-1/2}$.*

Therefore, if $\ell \le 2^{-8}C^{-2}r^{1-2/k}$ then $\lambda^*_{G,r} \ge Cdr^{1/k-1/2}$, which together with Theorem 11 implies that with probability at least $1/4$ any estimator makes a multiplicative error larger than two. Hence, sample size of $\ell(\ell+1)/2 = O(r^{2-4/k})$ is necessary to achieve multiplicative error less than two with high probability. We show that our estimator is optimal for $k = 3$, by providing a matching upper bound on the sample complexity. For any positive semidefinite matrix $M \in \mathbb{R}^{d\times d}$ of rank r with incoherence $\mu(M) = \mu$, $\kappa = \sigma_{\max}(M)/\sigma_{\min}(M)$, and some function $g(k) = O(k!)$, we define

$$\tilde{\rho}^2 \;\equiv\; (\kappa\mu)^{2k} g(k) \max\left\{1\,,\, \frac{\ell^{k-1}}{r^{k-2}}\,,\, \frac{\ell}{r}\,,\, \frac{r^{1/2}\ell^k}{d}\right\},$$

such that the variance of our estimator is bounded by $\mathrm{Var}(\widehat{\Theta}(\mathcal{P}_\Omega(M))/\|M\|_k^k) \le \rho^2(r^{1-2/k}/\ell)^k$ as we show for $k = 3$ in the proof of Theorem 13 in Section 8.6. Here, $g(k) = O(k!)$ is a function of k only.

Theorem 13 (Upper bound for clique sampling) *For $k = 3$, any $\delta > 0$, and any rank-r matrix $M \succeq 0$, the proposed estimator* (3) *achieves a multiplicative error δ with probability of error bounded by*

$$\mathbb{P}\left(\frac{\big|\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) - \|M\|_k^k\big|}{\|M\|_k^k} \;\ge\; \delta\right) \;\le\; \frac{\tilde{\rho}^2}{\delta^2}\Big(\frac{r^{1-2/k}}{\ell}\Big)^k,$$

under the graph sampling with the pattern graph G that is a clique over ℓ nodes.

For a typical scenario with finite μ and κ, this upper bound shows that sample size of $\ell(\ell+1)/2 = O(r^{2-4/k})$ is sufficient to achieve any arbitrarily small multiplicative error for $k = 3$ and sufficiently small rank $r \le d^{2k/(3k-2)}$ and $\ell \le r^{(k-2)/(k-1)}$, to ensure that the first term dominates in $\tilde{\rho}^2$. However, the numerical experiments suggest that our analysis holds more generally for all regimes of the rank r. This matches the previous lower bound, proving optimality of the proposed estimator for $k = 3$. Although the current analysis holds only for $k = 3$, we are intentionally writing the guarantee in general form as we expect the bound to hold more generally. In particular, we believe that Lemma 19 holds for all $k \ge 3$, and thereby Theorem 13 holds for any fixed integer $k \in [3, \infty)$. In the numerical experiments in Figure 8, M is generated using settings similar to that of Figure 4. Empirical probabilities are computed by averaging over 100 instances.

Although our analysis does not give a tight lower bound for Erdös-Rényi sampling, there exists graph patterns such that the sample complexity is large, i.e. scales linearly in d. Consider a *clique-star sampling* where the pattern graph $G(V, E)$ has a clique on a small subset of nodes V_1, $|V_1| = \ell$, and the remaining nodes $V \setminus V_1$ are disconnected among themselves and are fully connected with the clique in V_1. Precisely, $G = (V, E)$ with $(i, j) \in E$ if $i \in V_1$ or $j \in V_1$.

Lemma 14 (Lower bound for clique-star sampling) *Under the clique-star sampling over a clique of size ℓ, there exists an absolute constant c such that $\lambda^*_{G,r} \ge cd(r(\min\{\ell, r\}))^{-1/4}$.*

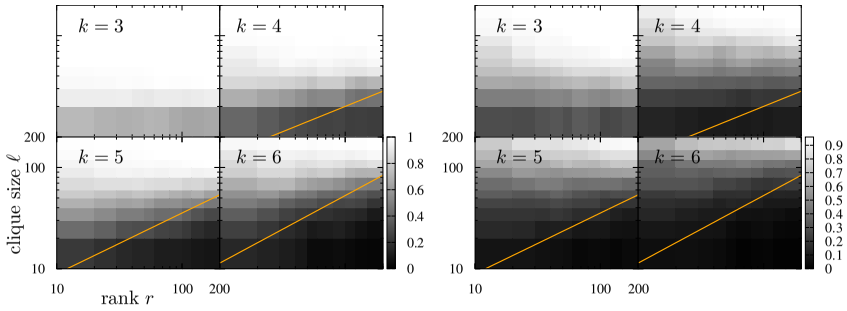


Figure 8: Each colormap in each block for $k \in \{3, 4, 5, 6\}$ show empirical probability of the event $\big\{\big|\|M\|_k^k - \widehat{\Theta}_k(\mathcal{P}_\Omega(M))\big|/\|M\|_k^k \le \delta\big\}$, for $\delta = 0.5$ (left panel) and $\delta = 0.2$ (right panel). Ω is generated by clique sampling of matrix M with a clique of size ℓ (vertical axis). M is a positive semi-definite matrix of size $d = 1000$. The solid lines correspond to our theoretical prediction $\ell = \sqrt{k}r^{1-2/k}$.

Together with Theorem 11, this implies that if $\ell \le c^4C^{-4}r^{1-4/k}$, then with probability at least $1/4$, any estimator makes an multiplicative error larger than two. This implies that the total number of edges in the pattern graph should be $O(dr^{1-4/k})$ for accurate estimation. Together with the upper bound on clique sampling in Theorem 13, this shows that the sample complexity can drastically change based on the pattern of your sampling model. Clique sampling requires only $O(r^{2-4/k})$ samples (for $k = 3$) whereas clique-star sampling requires at least $O(dr^{1-4/k})$. A proof of Lemma 12 and Lemma 14 is given in Section 8.4 and 8.5 respectively.

5. Discussion

We list some observations and future research directions.

Complexity of the estimator beyond $k = 7$. For $k \ge 8$, our approach of using matrix operations to count (the weights of) walks for each pattern $H \in \mathcal{H}_k$ can potentially be extended. However, the complexity of the problem fundamentally changes for $k \ge 8$. As our estimator is at least as hard as counting small structures in a simple (unweighted) graph, we can borrow known complexity results to get a lower bound. For instance, for $k \ge 8$, we need to count K_4 in a graph. There is no known simple matrix computation to count K_4 in a general graph. The best known run time for counting all K_4 is $O(d^{\alpha+1})$ for general graphs (Kloks et al., 2000). For general k, under standard hardness assumptions, Flum and Grohe (2004) show that there is no algorithm with run time $O(f(k)d^c)$ for counting cycles of length k, for any function $f(k)$ and a constant c that does not depend on k. In comparison, finding *one cycle* of length k can be done in time $2^{O(k)}d^\alpha$ (Alon et al., 1997). This implies that the complexity should scale as $O(d^{f(k)})$, and we believe $f(k)$ should be larger than $(\alpha\sqrt{2k}/3)$. The reason is that for $k \ge \binom{\ell}{2}$ for an odd ℓ, our estimator needs to count the

number of cliques K_ℓ of size ℓ and, for $k \geq (1/2)\ell^2$ for an even ℓ, we require counting K_ℓ. The best known algorithm for counting K_ℓ takes time $O(\min\{d^{1+\alpha\lceil(\ell-1)/3\rceil}, d^{2+\alpha\lceil(\ell-2)/3\rceil}\})$ for general graphs (Alon et al., 1997, Theorem 6.4). Putting these bounds together, we believe that the estimator take time at least $d^{\alpha\sqrt{2k}/3}$.

Sufficiency of $k \leq 7$ for practical applications. There are many practical applications where $k \leq 7$ is sufficient for estimating spectral sum function of a matrix from its partial observations. In the two applications we discuss in this paper, generalized rank estimation and spectrum estimation, we show by numerical experiments that $k \leq 7$ is sufficient. In Figure 6, we show that the generalized rank of a matrix can be estimated within 0.05 multiplicative error, for the case when the eigenvalues of the matrix are uniformly distributed in an interval. In Figure 7, we show that $k = 7$ is sufficient for estimating the spectrum for one peak and two peak distributions of eigenvalues. Further, for uniform distribution of eigenvalues, Kong and Valiant (2016) use only first $k = 7$ Schatten norms for estimating the spectrum. They note that considering higher Schatten norms beyond $k = 7$ did not significantly improve the results.

Graph sampling. Typical guarantees known for matrix completion assumes the Erdös-Rényi sampling. One exception is the deterministic sampling studied by Bhojanapalli and Jain (2014), but such generalization in sampling comes at a price of requiring more strict assumptions on the matrix M. We propose graph sampling, which can potentially capture how estimation guarantees depend explicitly on the pattern G, and still remain analytically tractable. We give such examples for special graphs in Section 4, and graph sampling model can potentially be used to bridge the gap in sampling models between theory and practice.

(Standard) rank estimation. As several popular matrix completion approaches require the knowledge of the rank of the original matrix, it is of great practical interest to estimate the standard rank of a matrix from sampled entries. Our framework in Section 3.2.1 provides a way to estimate the standard rank from samples. However, there are a few parameters that need to be tuned, such as the thresholds c_1 and c_2, and the degree of the polynomial approximation and the order of the Schatten norm. For rank estimation, Keshavan and Oh (2009) give an estimator that is provably correct in the regime where matrix completion works, justifying the requirement that popular matrix completion algorithms (Keshavan et al., 2010a; Jain et al., 2013) need to know the underlying rank. However, in the regime of our interest, which is below the standard matrix completion threshold, the algorithm fails miserably and there are no guarantees. In a more recent work, Saade et al. (2015) propose a novel rank estimator of counting the negative eigenvalues of Bethe Hessian matrix. It is an interesting future direction to build upon our framework to provide a guideline for choosing the parameters for the standard rank estimation, and compare its performance to existing methods.

The effect of the effective rank. One property of the Schatten norm is that as k gets large and as the singular values have small effective rank (meaning that they decay fast), the summation is dominated by the largest few singular values. In such scenarios, in the estimation problem, any algorithm that tracks the first few singular values correctly would achieve small error. Hence, the gap get smaller as the effective rank gets smaller, between

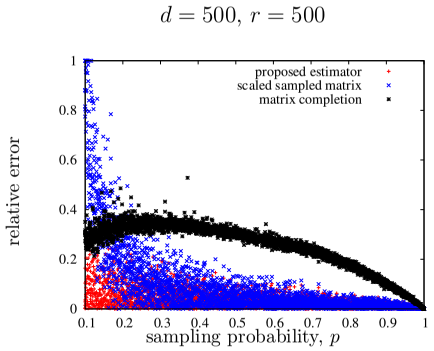


Figure 9: For a matrix with a very small effective rank, the gap between the proposed estimator and the simple scaled sampled matrix approach is smaller.

the proposed estimator and the simple Schatten k-norm of the rescaled sampled matrix, as depicted in Figure 9. We are using the same setting as those in Figure 4 with a full rank matrix M with $r = d = 500$, but the effective rank is relatively small as the singular values are decaying as $\sigma_i = 1/i^2$. For the current choice of $k = 5$, notice that the contribution in $\|M\|_k^k$ of the second singular value is a factor of 2^{10} smaller than the top singular value, making it effectively a rank one matrix.

Technical challenges. The technical challenge in proving bounds on the necessary number of samples needed to estimate Schatten k-norms lies in getting tight bounds on the variance of the estimator. Variance is a function of weighted counts of each pseudograph of $2k$-closed walks, in the complete matrix. As the weight of each walk can be positive or negative, significant cancellation occurs when we sum all the weights. However, this stochastic cancellation is hard to capture in the analysis and we assume the worst case where all the weights are positive, which cannot occur for incoherent and well-conditioned matrices. This weakness of the analysis leads to the requirement of the rank being sufficiently small in the case of Erdös-Rényi sampling and k small in the case of clique sampling. We believe these bounds can be tightened and the same is reflected in the numerical simulations which show the same scaling holds for all small values of k and rank close to the dimension of the matrix.

6. Algorithm for estimating Schatten k-norm for $k \geq 8$

The collection of pseudographs $\mathcal{H}_k$ is partitioned into sets $\{\mathcal{H}_{k,i}^{\text{iso}}\}_{1\leq i\leq r}$, for some $r \leq k!$. The partitions $\mathcal{H}_{k,i}^{\text{iso}}$ are defined such that the pseudographs in one partition are isomorphic to each other when multi-edges are condensed into one. This is useful since all the pseudographs in one partition are observed together in $G([d], \Omega)$ for any fixed subgraph in G. The underlying simple graph (including self loops) for each partition $\mathcal{H}_{k,i}^{\text{iso}}$ is denoted by $F_{k,i}$.

The main idea is to enumerate a list $\mathcal{L}_\ell$ of all connected ℓ-vertex induced subgraphs (possibly with loops) of the graph $G([d], \Omega)$, for each $1 \le \ell \le k$. The unbiased weighted count of all pseudographs $\mathcal{H}_k$ for each of these vertex-induced subgraphs $g \in \mathcal{L}_\ell$ is computed. This is achieved by further enumerating a list $\mathcal{S}_{g,\ell}$ of all ℓ-vertex subgraphs for each g. Then the unbiased weight of all pseudographs $H \in \mathcal{H}_k$ that exist in the subgraph h is computed and is summed over to get the estimate of the k-th Schatten norm. Recall the notation $\mathcal{P}_\Omega(M)$ which is used to denote the partially observed matrix corresponding to the index set Ω with the unobserved entries being replaced by zero. We abuse this notation and use $h(M)$ to represent the matrix M restricted to the subgraph h of the observed graph $G([d], \Omega)$.

Each connected induced subgraphs of size k in a graph can be enumerated in time polynomial in d and k (Elbassioni, 2015). The number of connected induced subgraphs of size k in a graph is upper bounded by $(e\Delta)^k/((\Delta-1)k)$, where Δ is the maximum degree of the graph (Uehara et al., 1999). Therefore, Algorithm 4 runs in time that is super exponential in k, polynomial in d and the number of k-connected induced subgraphs in the observed graph $G([d], \Omega)$.

Algorithm 4 Schatten k-norm estimator

Require: $\mathcal{P}_\Omega(M)$, k, $\mathcal{H}_k$, $p(H)$ for all $H \in \mathcal{H}_k$
Ensure: $\widehat{\Theta}_k(\mathcal{P}_\Omega(M))$
$\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) \leftarrow 0$
for $1 \le \ell \le k$ **do**
  Enumerate a list, $\mathcal{L}_\ell$, of all connected ℓ-vertex induced subgraphs (possibly with loops) of the graph $G([d], \Omega)$
  for all $g \in \mathcal{L}_\ell$ **do**
    Enumerate a list $\mathcal{S}_{g,\ell}$ of all connected ℓ-vertex subgraphs of the graph g by removing one or more edges
    for all $h \in \mathcal{S}_{g,\ell}$ **do**
      for $1 \le i \le r$ **do**
        if h is isomorphic to $F_{k,i}$ **then**
          $\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) \leftarrow \widehat{\Theta}_k(\mathcal{P}_\Omega(M)) + \sum_{H \in \mathcal{H}_{k,i}^{\text{iso}}} \frac{1}{p(H)} \omega_{h(M)}(H) c(H)$
        end if
      end for
    end for
  end for
end for

7. Algorithm for computing the Chebyshev polynomial

8. Proofs

We provide proofs for the main results and the technical lemmas.

Algorithm 5 Chebyshev polynomial of the first kind approximating $H_{c_1,c_2}(x)$

Require: H_{c_1,c_2}, c_1, c_2, and target accuracy $\delta = 0.1$
Ensure: Chebyshev polynomial $q(x)$ of first kind
$g(x) \equiv \frac{x-c_2}{c_1-c_2}$
$T_0(x) \equiv 1$, $T_1(x) \equiv x$
$q(x) \leftarrow \frac{1}{\pi}\int_{c_2}^{c_1}(1-x^2)^{-1/2} g(x) T_0(x) dx + \frac{1}{\pi}\int_{c_1}^{1}(1-x^2)^{-1/2} T_0(x) dx$
$i = 1$
while $\sup_{x\in[0,1]} |q(x) - H_{c_1,c_2}(x)| \ge \delta$ **do**
  $q(x) \leftarrow q(x) + \frac{2T_i(x)}{\pi}\int_{c_2}^{c_1}(1-x^2)^{-1/2} g(x) T_i(x) dx + \frac{2T_i(x)}{\pi}\int_{c_1}^{1}(1-x^2)^{-1/2} T_i(x) dx$
  $i \leftarrow i + 1$
  $T_i(x) \equiv 2xT_{i-1}(x) - T_{i-2}(x)$
end while

8.1. Proof of Theorem 3

Consider $\widetilde{W}$ to be the collection of all length k closed walks on a complete graph of d vertices. Here we slightly overload the notion of complete graph to refer to an undirected graph with not only all the $d(d-1)/2$ simple edges but also with d self loops as well. Construct the largest possible collection W from $\widetilde{W}$ wherein each walk has distinct weights that is $\omega(w) \neq \omega(w')$ for all $w, w' \in W$. We partition W according to the pattern among k-cyclic pseudographs, which are further partitioned into four groups. The estimator (3) can be re-written as

$$\begin{aligned}
\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) &= \sum_{w\in W} \frac{c(H(w))}{p(H(w))} \omega_{\mathcal{P}_\Omega(M)}(w) \\
&= \sum_{H\in\mathcal{H}_k} \Big\{ \frac{c(H)}{p(H)} \sum_{w: H(w)=H} \omega_M(w)\, \mathbb{I}(w \subseteq \Omega) \Big\} \\
&= \sum_{i=1}^{4} \sum_{H\in\mathcal{H}_{k,i}} \Big\{ \frac{c(H)}{p(H)} \sum_{w: H(w)=H} \omega_M(w)\, \mathbb{I}(w \subseteq \Omega) \Big\},
\end{aligned}$$

where we write $w \subseteq \Omega$ to denote the event that all the edges in the walk w are sampled, and we define

- $\mathcal{H}_{k,1} \equiv \{C_k\}$ is just a (set of a) simple cycle of length k and there are total $|\{w \in W : H(w) \in \mathcal{H}_{k,1}\}| = \binom{d}{k}(k!/2k) \le (d^k/2k)$ corresponding walks to this set, and $c(C_k) = 2k$.
- $\mathcal{H}_{k,2} \equiv \{H(V_H, E_H) \in \mathcal{H}_k : |V_H| \le k-1$ and no self loops$\}$, and there are total $|\{w \in W : H(w) \in \mathcal{H}_{k,2}| \le d^{k-1}$ corresponding walks to this set.
- $\mathcal{H}_{k,3} \equiv \bigcup_{s=1}^{k-1} \mathcal{H}_{k,3,s}$ where $\mathcal{H}_{k,3,s} = \{H \in \mathcal{H}_k$ with s self loops$\}$, and there are total $|\{w \in W : H(w) \in \mathcal{H}_{k,3}\}| \le d^{k-s}$ corresponding walks in this set.
- $\mathcal{H}_{k,4} \equiv \{H(V_H, E_H) \in \mathcal{H}_k : |V_H| = 1\}$ is a (set of a) graph with k self loops and there are total $|\{w \in W : H(w) \in \mathcal{H}_{k,4}\}| = d$ corresponding walks to this set.

Given this unbiased estimator, we provide an upper bound on the variance of each of the partitions to prove concentration with Chebyshev's inequality. For any walk $w \in W$, let $|w|$ denote the number of unique edges (including self loops) that the walk w traverses. Let $|w \cap w'|$ denote the number of unique overlapping edges (including self loops) of walks w and w'. We have,

$$\begin{aligned}
&\mathrm{Var}\big(\widehat{\Theta}_k(\mathcal{P}_\Omega(M))\big) \\
&= 2\sum_{\ell=1}^{k-1} \sum_{\substack{w\neq w'\in \widetilde{W} \\ |w\cap w'|=\ell}} \mathrm{cov}\left(\frac{\mathbb{I}(w\subseteq\Omega)\omega_M(w)c(H(w))}{p(H(w))}, \frac{\mathbb{I}(w'\subseteq\Omega)\omega_M(w')c(H(w'))}{p(H(w'))}\right) \\
&\quad + \sum_{i=1}^{4} \sum_{H\in\mathcal{H}_{k,i}} \left\{\frac{c(H)^2}{p(H)^2} \sum_{w:H(w)=H} \omega_M(w)^2 \mathrm{Var}\Big(\mathbb{I}(w\subseteq\Omega)\Big)\right\} \\
&< 4\sum_{\ell=1}^{k-1} \sum_{\substack{w\neq w'\in W \\ |w\cap w'|=\ell}} \mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\mathbb{I}(w'\subseteq\Omega)\Big]\left(\frac{\big|\omega_M(w)\,\omega_M(w')\big|c(H(w))c(H(w'))}{p(H(w))\,p(H(w'))}\right) \\
&\quad + \sum_{i=1}^{4} \sum_{H\in\mathcal{H}_{k,i}} \sum_{w:H(w)=H} \frac{c(H)^2\omega_M(w)^2}{p(H)^2} \mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\Big]. \qquad (16)
\end{aligned}$$

Recall from the definition of incoherence that $|M_{ii}| \le \sigma_1(M)\mu r/d$ and $|M_{ij}| = \sigma_1(M)\mu r^{1/2}/d$, and let $\alpha = \sigma_1(M)\mu r^{1/2}/d$ denote the maximum off-diagonal entry, such that $|M_{ij}| \le \alpha$ and $|M_{ii}| \le \alpha\sqrt{r}$ for all $i,j \in [d]$. Let $A_{p,k,\alpha,d} = d^k\alpha^{2k}/p^k$ denote the target scaling of the variance, then

$$\sum_{H\in\mathcal{H}_{k,i}} \sum_{w:H(w)=H} \frac{c(H)^2\,\omega_M(w)^2}{p(H)^2} \mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\Big] \;\le$$

$$\begin{cases}
\dfrac{d^k}{2k}\dfrac{(2k)^2\alpha^{2k}}{p^k} = 2kA_{p,k,\alpha,d}\,, & \text{for } i=1\ , \qquad (17)\\
d^{k-1}\dfrac{f(k)^2\alpha^{2k}}{p^k} = \dfrac{f(k)^2}{d}A_{p,k,\alpha,d}\,, & \text{for } i=2\ , \qquad (18)\\
d\dfrac{r^k\alpha^{2k}}{p} = \dfrac{r^kp^{k-1}}{d^{k-1}}A_{p,k,\alpha,d}\,, & \text{for } i=4\,, \qquad (19)
\end{cases}$$

and for $i=3$ and for $1\le s\le k-1$, we have

$$\sum_{H\in\mathcal{H}_{k,3,s}} \sum_{w:H(w)=H} \frac{c(H)^2\,\omega_M(w)^2}{p(H)^2} \mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\Big] \;\le\; d^{k-s}\frac{f(k)^2\alpha^{2k}r^s}{p^k} = \frac{f(k)^2r^s}{d^s}A_{p,k,\alpha,d}\,, \quad (20)$$

where $c(H)$ is defined as the multiplicity of walks with the same weight satisfying $c(H) \le f(k)$. For $w \neq w'$ and $|w\cap w'| = \ell$, where the range of ℓ varies across equations depending upon the set to which w, w' belongs, we have the following:

$$\sum_{\substack{w\neq w'\in W \\ |w\cap w'|=\ell, H(w)\in\mathcal{H}_{k,i,s}, H(w')\in\mathcal{H}_{k,i',s'}}} \mathbb{E}\Big[\mathbb{I}(w\in\Omega)\mathbb{I}(w'\in\Omega)\Big] \frac{\big|\omega_M(H(w))\omega_M(H(w'))\big|\,c(H(w))c(H(w'))}{p(H(w))p(H(w'))} \;\le$$

$$\begin{cases}
\dfrac{d^kd^{k-(\ell+1)}}{2k}\dfrac{\alpha^{2k}(2k)^2}{p^\ell} = \dfrac{(dp)^{k-\ell}}{d}\,2kA_{p,k,\alpha,d}, & \text{for } i=i'=1 \quad (21)\\
\dfrac{f(k)^2d^{k-1}d^{k-1-(\ell+1)}\alpha^{2k}}{p^\ell} \le \dfrac{f(k)^2(dp)^{k-\ell}}{d^3}A_{p,k,\alpha,d} & \text{for } i=i'=2 \quad (22)\\
\dfrac{f(k)^2d^{k-s}d^{k-s'-\ell}\alpha^{2k-s-s'}(\alpha\sqrt{r})^{s+s'}}{p^\ell} \le \dfrac{f(k)^2(dp)^{k-\ell}}{(d/\sqrt{r})^{s+s'}}A_{p,k,\alpha,d}\,, & \text{for } i=i'=3 \quad (23)\\
\dfrac{f(k)^2d^kd^{k-1-(\ell+1)}\alpha^{2k}}{p^\ell} \le \dfrac{f(k)^2(dp)^{k-\ell}}{d^2}A_{p,k,\alpha,d} & \text{for } i=1, i'=2 (24)\\
\dfrac{f(k)^2d^kd^{k-s-(\ell+1)}\alpha^{2k-s}(\alpha\sqrt{r})^s}{p^\ell} \le \dfrac{f(k)^2(dp)^{k-\ell}}{d(d/\sqrt{r})^s}A_{p,k,\alpha,d} & \text{for } i=1, i'=3 (25)\\
\dfrac{f(k)^2d^{k-1}d^{k-s-(\ell+1)}\alpha^{2k-s}(\alpha\sqrt{r})^s}{p^\ell} \le \dfrac{f(k)^2(dp)^{k-\ell}}{d^2(d/\sqrt{r})^s}A_{p,k,\alpha,d} & \text{for } i=2, i'=3 (26)\\
\dfrac{f(k)^2dd^{k-s-\ell}\alpha^{k-s}(\alpha\sqrt{r})^{k+s}}{p^\ell} \le \dfrac{f(k)^2(dp)^{k-\ell}}{d^{k-1}(d/\sqrt{r})^{k+s}}A_{p,k,\alpha,d} & \text{for } i=3, i'=4 (27)
\end{cases}$$

where (27) is valid only for $\ell = 1$. Note that for any w with $H(w) \in \mathcal{H}_{k,1}\bigcup\mathcal{H}_{k,2}$, it has no overlap with w' such that $H(w') \in \mathcal{H}_{k,4}$.

Observe that $\mathrm{Var}\big(\widehat{\Theta}_k(\mathcal{P}_\Omega(M))\big)$ as bounded in (16) is upper bounded by the sum of quantities in (17)-(27), summating over all possible values of $1\le\ell\le k-1$, and $1\le s,s'\le k-1$. Let $h(k) \equiv f(k)^2A_{p,k,\alpha,d}$. Observe that quantities in (17),(18), and (20) are upper bounded by $h(k)$. Quantities in (21)-(27) are upper bounded by $h_1(k) \equiv h(k)(dp)^{k-1}/d$. Quantity in (19) is upper bounded by $h_2(k) \equiv h(k)r^kp^{k-1}/d^{k-1}$.

Given $\|M\|_k^k \ge r(\sigma_{\min})^k$, recall a bound on off diagonals of matrix M by $|M_{ij}| \le \alpha = \mu\sigma_{\max}\sqrt{r}/d$ and $A_{p,k,\alpha,d} = d^k\alpha^{2k}/p^k$. This gives

$$\frac{A_{p,k,\alpha,d}}{\|M\|_k^{2k}} \;\le\; \frac{\kappa^{2k}\mu^{2k}r^{k-2}}{d^kp^k}\,.$$

Using Chebyshev's inequality and collecting all terms in the upper bound on the variance, we have for sufficiently large d the following bound:

$$\mathbb{P}\left(\frac{\big|\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) - \|M\|_k^k\big|}{\|M\|_k^k} \;\ge\; \delta\right) \;\le\; \frac{(\kappa\mu)^{2k}f(k)^2r^{k-2}}{\delta^2(dp)^k}\max\left\{1, \frac{(dp)^{k-1}}{d}, \frac{r^kp^{k-1}}{d^{k-1}}\right\},$$

where the second and the third term in the max expression follow from evaluating $h_1(k)$ and $h_2(k)$. If sampling probability p is small enough such that $dp \le Cd^{1/(k-1)}$ for some constant C, then the second and the third terms are smaller than the first term. Hence, the desired result in Theorem 3 follows.

8.2. Proof of Theorem 4

We can prove a Bernstein-type bound on accuracy of the estimator. The estimator (3) can be rewritten as a multi-linear polynomial function of $d(d+1)/2$ i.i.d. Bernoulli(p) random variables.

$$\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) \;=\; \sum_{w\in W}\left\{\frac{c(H(w))}{p(H(w))}\,\omega_M(w) \prod_{(i,j)\in\mathrm{unique}(w)} \mathbb{I}((i,j)\in\Omega)\right\},$$

where $\mathbb{I}((i,j) \subseteq \Omega)$ is a random variable that takes value 1 if the $(i,j)_{\text{th}}$ entry of the matrix M is sampled, and unique(w) denotes the set of the unique edges (and self loops) that the walk w traverses. Let q denote the power of the polynomial function that is the maximum number of unique edges in the walk w, that is $q = k$.

We use the following Bernstien-type concentration results of Schudy and Sviridenko (2011) for the polynomials of independent random variables.

Lemma 15 (Schudy and Sviridenko (2011),Theorem 1.3) *We are given $d(d+1)/2$ independent central moment bounded random variables $\{\mathbb{I}((i,j) \in \Omega)\}_{1 \le i \le j \le d}$ with the same parameter L. We are given a multilinear polynomial $\widehat{\Theta}_k(\mathcal{P}_\Omega(M))$ of power q, then*

$$\mathbb{P}\Big[\Big|\widehat{\Theta}_k(\mathcal{P}_\Omega(M)) - \mathbb{E}\big[\widehat{\Theta}_k(\mathcal{P}_\Omega(M))\big]\Big| \ge \lambda\Big] \;\le\; e^2 \max\left\{ e^{\frac{-\lambda^2}{\mathrm{Var}[\widehat{\Theta}_k(\mathcal{P}_\Omega(M))]R^q}}, \max_{t\in[q]} e^{-(\frac{\lambda}{\mu_t L^t R^q})^{1/t}} \right\},$$

where R is some absolute constant and μ_t is defined as follows:

$$\mu_t \;=\; \max_{\substack{S\subseteq\{(i,j):i,j\in[d]\}\\ |S|=t}} \left(\sum_{w\in W | w\supseteq S} \frac{c(H(w))}{p(H(w))} \, |\omega_M(w)| \prod_{(i,j)\in \text{unique}(w)\setminus S} \mathbb{E}[\mathbb{I}((i,j)\in\Omega)] \right) \quad (28)$$

where $w \supseteq S$ denotes that the walk w comprises edges(and self loops) contained in the set S. The parameter L is defined as follows: A random variable Z is called central moment bounded with real parameter $L > 0$, if for any integer $i \ge 1$ we have

$$E\big[|Z - \mathbb{E}[Z]|^i\big] \;\le\; i\,L\,\mathbb{E}[|Z-\mathbb{E}[Z]|^{i-1}]\,.$$

For each of the Bernoulli random variables $\{\mathbb{I}((i,j)\in\Omega)\}_{1\le i\le j\le d}$, L is contained in $[1/4, 1]$. In the following, we show that $\mu_t \le (\mu\sigma_{\max})^k g(k) r^k/(d(dp)^t)$, for $t \in [k]$. Using Lemma 15, along with $\|M\|_k^k \ge r(\sigma_{\min})^k$, the bound in (9) follows immediately.

To compute μ_t, define a set of walks $W_{\ell,s,\hat{s}}$ such that $w \in W_{\ell,s,\hat{s}}$ has $0 \le \ell \le k$ unique edges and $0 \le s \le k$ unique self loops, and $\hat{s}$ total self loops with $\ell + \hat{s} \le k$. For the set S as required in (28), let $S_{\tilde{\ell},\tilde{s}}$ be a set of $\tilde{\ell}$ unique edges and $\tilde{s}$ unique self loops, with $|S_{\tilde{\ell},\tilde{s}}| = \tilde{\ell}+\tilde{s}$

where $1 \le \tilde{\ell} + \tilde{s} \le k$. Therefore, we have

$$\begin{aligned}
\mu_t &= \max_{\substack{S_{\tilde{\ell},\tilde{s}}\\ :\tilde{\ell}+\tilde{s}=t}} \left(\sum_{\substack{0\le s\le\hat{s}\le k\\ \ell\in[k]:\ell+\hat{s}\le k}} \sum_{\substack{w\in W_{\ell,s,\hat{s}}\\ :w\supseteq S_{\tilde{\ell},\tilde{s}}}} \frac{c(H(w))}{p(H(w))} \, |\omega_M(w)| \prod_{(i,j)\in\text{unique}(w)\setminus S_{\tilde{\ell},\tilde{s}}} \mathbb{E}[\mathbb{I}((i,j)\subseteq\Omega)] \right) \\
&\le \max_{\substack{S_{\tilde{\ell},\tilde{s}}\\ :\tilde{\ell}+\tilde{s}=t}} \left(\sum_{\substack{0\le s\le\hat{s}\le k\\ \ell\in[k]:\ell+\hat{s}\le k}} \sum_{\substack{w\in W_{\ell,s,\hat{s}}\\ :w\supseteq S_{\tilde{\ell},\tilde{s}}}} \frac{f(k)}{p^{\ell+s}} \, \alpha^k r^{\hat{s}/2} p^{\ell+s-(\tilde{\ell}+\tilde{s})} \right) \\
&\le \max_{\substack{S_{\tilde{\ell},\tilde{s}}\\ :\tilde{\ell}+\tilde{s}=t}} \left(\sum_{\substack{0\le s\le\hat{s}\le k\\ \ell\in[k]:\ell+\hat{s}\le k,\tilde{s}\le s}} \frac{d^{\ell-(1+\tilde{\ell})} f(k)}{p^{\ell+s}} \, \frac{(\mu\sigma_{\max})^k r^{(k+\hat{s})/2}}{d^k} p^{\ell+s-(\tilde{\ell}+\tilde{s})} \right) \\
&= \max_{\substack{S_{\tilde{\ell},\tilde{s}}\\ :\tilde{\ell}+\tilde{s}=t}} \left(\sum_{\substack{0\le s\le\hat{s}\le k\\ \ell\in[k]:\ell+\hat{s}\le k,\tilde{s}\le s}} \frac{f(k)(\mu\sigma_{\max})^k r^{(k+\hat{s})/2}}{d d^{(k-\ell-\tilde{s})}(dp)^{(\tilde{\ell}+\tilde{s})}} \right) \\
&\le \max_{\substack{S_{\tilde{\ell},\tilde{s}}\\ :\tilde{\ell}+\tilde{s}=t}} \left(\frac{k^3 f(k)(\mu\sigma_{\max})^k r^{(k+\hat{s})/2}}{d d^{(k-\ell-\tilde{s})}(dp)^{(\tilde{\ell}+\tilde{s})}} \right) \;\le\; \frac{(\mu\sigma_{\max})^k g(k) r^k}{d(dp)^t}\,.
\end{aligned}$$

8.3. Proof of Theorem 11

The proof technique is a generalization to a rank-r symmetric matrix of the proof given by Li et al. (2014) for deriving lower bound on the size of a random bi-linear sketch needed for approximating Schatten norm of any matrix. It also draws on the techniques used in Andoni et al. (2013) for proving a lower bound on the size of the linear sketches of moments.

We prove Theorem 11 for an arbitrary fixed relabeling permutation π of the graph nodes. Indeed, by Yao's minimax principle, it suffices to give two distributions on matrix $M \in \mathcal{M}_r$ for which the $\|M\|_k$ values differ by a constant factor with high probability, but for any relabeling permutation π of the nodes of the pattern graph G, the induced distributions on the sampled entries $\mathcal{P}_\Omega(M)$ corresponding to the relabeled graph $G_\pi(\widetilde{V}, \Omega)$, have low total variation distance.

For positive $C > 0$ to be specified later, define $\lambda \equiv C d r^{1/k - 1/2}$. We construct distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ for $M \in \mathcal{M}_{r,\mu}$ with $\mu = C'\sqrt{\log r}$, for some absolute constant C', such that the following holds:

1. $\|M\|_k \le \lambda$ on the entire support of $\mathcal{D}_1$, and $\|M\|_k \ge 4\lambda$ on the entire support of $\mathcal{D}_2$.

2. Let $\mathcal{E}_1$ and $\mathcal{E}_2$ denote the distribution of the sampled matrix $\mathcal{P}_\Omega(M)$ when M is drawn from $\mathcal{D}_1$ and $\mathcal{D}_2$ respectively. Recall that Ω is the set of edges of the relabeled graph $G_\pi(\widetilde{V}, \Omega)$ as defined in Section 4.1. If $\lambda^*_{G,r} \ge \lambda$ then, the total variation distance between $\mathcal{E}_1$ and $\mathcal{E}_2$ is bounded by $\mathrm{TV}(\mathcal{E}_1, \mathcal{E}_2) \le 1/2$.

The desired result (15) follows from the above claims and the following relationship between statistical tests and estimators:

$$\begin{aligned}
&\mathop{\mathbb{P}}_{M\sim\frac{1}{2}(\mathcal{D}_1+\mathcal{D}_2)}\left(\frac{1}{2}\|M\|_k \le \widetilde{\Theta}(\mathcal{P}_{\Omega(M)}) \le 2\|M\|_k\right)\\
&\le \frac{1}{2}\mathop{\mathbb{P}}_{M\sim\mathcal{D}_2}\left(\widetilde{\Theta}(\mathcal{P}_{\Omega(M)}) \ge 2\lambda\right) + \frac{1}{2}\mathop{\mathbb{P}}_{M\sim\mathcal{D}_1}\left(\widetilde{\Theta}(\mathcal{P}_{\Omega(M)}) \le 2\lambda\right)\\
&\le \frac{1}{2}\big(1+\mathrm{TV}(\mathcal{E}_1,\mathcal{E}_2)\big) \;\le\; \frac{3}{4}\,,
\end{aligned}$$

where the last inequality follows from the following characterization of the total variation distance $\mathrm{TV}(\mathcal{E}_1,\mathcal{E}_2) \equiv \sup_A |\mathcal{E}_1(A) - \mathcal{E}_2(A)|$.

To prove the two claims, we construct one of the desired rank-r random matrix via tiling, i.e. covering the matrix with copies of a single $r\times r$ sub-matrix from the *Gaussian Wigner Ensemble*, where diagonals and off-diagonals(upper triangle) are both distributed as i.i.d. standard Gaussians. Another one is constructed by adding a rank one perturbation. Precisely, we define a random matrix drawn from $\mathcal{D}_1$ as follows.

A random $r\times r$ matrix Z chosen from Gaussian Wigner Ensemble, $\mathcal{G}(r,r)$, is a symmetric matrix whose entries $Z_{i,i}$ and $Z_{i,j}$ for $i<j$ are independent with $N(0,1)$ distribution. Define $B \equiv \mathbb{1}_{\lceil d/r\rceil}\mathbb{1}_{\lceil d/r\rceil}^\top$ to be an all-ones matrix of size $\lceil d/r\rceil \times \lceil d/r\rceil$. Let $\bar{\mathcal{D}}_1$ denote the distribution of $M_1 = Y\otimes B$ where $Y\sim\mathcal{G}(r,r)$, and $\otimes$ denotes the standard Kronecker product of two matrices. Note that the matrix norm of M_1 and Y are related by $\|M_1\|_k = \lceil d/r\rceil\|Y\|_k$. Since the Schatten norm of $Y\sim\mathcal{G}(r,r)$ takes value on the entire $\mathbb{R}_+$, we need to truncate it. We set $\mathcal{D}_1$ to be $\bar{\mathcal{D}}_1$ conditioned on the event $S_1 = \{M_1 : \|M_1\|_k \le \lambda, \mu(M_1) \le C'\sqrt{\log r}\}$, i.e. $\mathcal{D}_1(A) = \bar{\mathcal{D}}_1(A\cap S_1)/\bar{\mathcal{D}}_1(S_1)$.

We define $\bar{\mathcal{D}}_2$ by adding a rank one perturbation. Precisely, let $M_2 = M_1 + (5/d)\lambda U$, where $M_1\sim\bar{\mathcal{D}}_1$ and $U = uu^\top\otimes B$. Here a random vector $u\in\{\pm1\}^r$ is a vector of i.i.d. Rademacher random variables. Note that U is a rank one matrix and $\|U\|_k = \lceil d/r\rceil\|uu^\top\|_k = d$. We set $\mathcal{D}_2$ to be $\bar{\mathcal{D}}_2$ conditioned on the event $S_2 = \{M_2 : \|M_2\|_k \ge 4\lambda, \mu(M_2) \le C'\sqrt{\log r}\}$. Observe that $M_1\sim\bar{\mathcal{D}}_1$ and $M_2\sim\bar{\mathcal{D}}_2$ belong to $\mathbb{R}^{d\times d}$, are symmetric and both are rank at most $r+1$.

Let $\bar{\mathcal{E}}_1$ and $\bar{\mathcal{E}}_2$ denote the distribution of $\mathcal{P}_\Omega(M)$ when M is drawn from $\bar{\mathcal{D}}_1$ and $\bar{\mathcal{D}}_2$ respectively. We first show that their total variation distance is not too large. Using the triangle inequality, we have

$$\begin{aligned}
\mathrm{TV}(\mathcal{E}_1,\mathcal{E}_2) &\le \mathrm{TV}(\bar{\mathcal{E}}_1,\bar{\mathcal{E}}_2) + \mathrm{TV}(\bar{\mathcal{E}}_1,\mathcal{E}_1) + \mathrm{TV}(\bar{\mathcal{E}}_2,\mathcal{E}_2)\\
&\le \mathrm{TV}(\bar{\mathcal{E}}_1,\bar{\mathcal{E}}_2) + \mathrm{TV}(\bar{\mathcal{D}}_1,\mathcal{D}_1) + \mathrm{TV}(\bar{\mathcal{D}}_2,\mathcal{D}_2) \qquad (29)\\
&= \mathrm{TV}(\bar{\mathcal{E}}_1,\bar{\mathcal{E}}_2) + \mathop{\mathbb{P}}_{M_1\sim D_1}\big((\|M_1\|_k \ge \lambda)\cup(\mu(M_1)\ge C'\sqrt{\log r})\big)\\
&\quad + \mathop{\mathbb{P}}_{M_2\sim\mathcal{D}_2}\big((\|M_2\|_k \le 4\lambda)\cup(\mu(M_2)\ge C'\sqrt{\log r})\big)\,, \qquad (30)
\end{aligned}$$

where (29) follows from the data processing inequality and (30) follows from $\mathrm{TV}(\mathcal{E}_1,\mathcal{E}_2)\equiv\sup_A|\mathcal{E}_1(A)-\mathcal{E}_2(A)|$. We next show that the three terms in (30) are sufficiently small.

We first provide an upper bound on $\mathrm{TV}(\bar{\mathcal{E}}_1,\bar{\mathcal{E}}_2)$. As per our construction, only the upper triangular (including diagonals) of the upper-left submatrix of size $r\times r$ of $M_1\sim\mathcal{D}_1$ and $M_2\sim\mathcal{D}_2$ has unique entries and the rest are copies of these. Observe that the set of unique

entries of M_1(or M_2) corresponding to any pattern graph $G(V,E)$ are precisely the following entries of the projection graph $\mathcal{P}^{(r)}(G)$ that is defined in Section 4.1:

$$E(\mathcal{P}^{(r)}(G)) \;\equiv\; \Big\{(i,j) : i\le j\in[r], (i,j)\in\mathcal{P}^{(r)}(G(V,E))\Big\}\,. \qquad (31)$$

For the purpose of computing the total variation distance $\mathrm{TV}(\bar{\mathcal{E}}_1,\bar{\mathcal{E}}_2)$, it is sufficient to consider only $E(\mathcal{P}^{(r)}(G_\pi))$ entries of M_1 distributed as i.i.d. standard Gaussians $N(0,I_{\ell_1\times\ell_1})$, and the entries of M_2 distributed as $N(W,I_{\ell_1\times\ell_1})$), where $\ell_1 = |E(\mathcal{P}^{(r)}(G_\pi))|$. The random vector W represents the rank one perturbation and is distributed as

$$W_{i,j} = (5/d)\lambda\, u_i u_j\,, \qquad (i,j)\in E(\mathcal{P}^{(r)}(G_\pi))\,.$$

To bound total variation distance between $\bar{\mathcal{E}}_1$ and $\bar{\mathcal{E}}_2$, we use the following lemma and the fact that for any two distributions μ and ν, $\mathrm{TV}(\mu,\nu)\le\sqrt{\chi^2(\mu\,\|\,\nu)}$. Let $\mu*\nu$ denote the convolution of the density (or equivalently addition of the two random variables).

Lemma 16 (Ingster and Suslina (2012), p97) *It holds that $\chi^2(N(0,I_n)*\mu\,\|\,N(0,I_n)) \le \mathbb{E}\exp(\langle z,z'\rangle) - 1$, where $z,z'\sim\mu$ are independent.*

It follows that

$$\mathrm{TV}(\bar{\mathcal{E}}_1,\bar{\mathcal{E}}_2) \;\le\; \sqrt{\mathbb{E}e^{\langle W,W'\rangle}-1} \;\le\; 1/5\,,$$

for $\lambda^*_G\ge\lambda$ where the expectation is taken over independent W and W' which are identically distributed. We show that if $\lambda^*_G\ge\lambda$ the last inequality holds, as following:

$$\begin{aligned}
&\mathbb{E}_{W,W'}\exp\big(\langle W,W'\rangle\big)\\
&= \mathbb{E}_{u,u'}\exp\left((5/d)^2\lambda^2\sum_{(i,j)\in E(\mathcal{P}^{(r)}(G_\pi))} u_iu'_iu_ju'_j\right)\\
&= \mathbb{E}_u\exp\left((5/d)^2\lambda^2\sum_{(i,j)\in E(\mathcal{P}^{(r)}(G_\pi))} u_iu_j\right) \qquad (32)\\
&= \mathbb{E}_u\left[\exp\left((5/d)^2\lambda^2\sum_{\substack{(i,j)\in E(\mathcal{P}^{(r)}(G_\pi))\\ :i\ne j}} u_iu_j\right)\right]\exp\left((5/d)^2\lambda^2\sum_{\substack{(i,j)\in E(\mathcal{P}^{(r)}(G_\pi))\\ :i=j}} u_iu_j\right)\\
&\le \mathbb{E}_u\left[\exp\left((5/d)^2\lambda^2\sum_{\substack{(i,j)\in E(\mathcal{P}^{(r)}(G_\pi))\\ :i\ne j}} 2u_iu_j\right)\right]\exp\left((5/d)^2\lambda^2\sum_{\substack{(i,j)\in E(\mathcal{P}^{(r)}(G_\pi))\\ :i=j}} u_iu_j\right) \qquad (33)\\
&= \mathbb{E}_u\left[\exp\left((5/d)^2\lambda^2\sum_{\substack{(i,j)\in\mathcal{P}^{(r)}(G_\pi)\\ :i\ne j}} u_iu_j\right)\right]\exp\left((5/d)^2\lambda^2\sum_{\substack{(i,j)\in\mathcal{P}^{(r)}(G_\pi)\\ :i=j}} u_iu_j\right) \qquad (34)\\
&= \mathbb{E}_u\left[\exp\left((5/d)^2\lambda^2\sum_{(i,j)\in\mathcal{P}^{(r)}(G_\pi)} u_iu_j\right)\right]\\
&\le 1+1/25\,, \qquad (35)
\end{aligned}$$

where (32) follows from the fact that u, u' are i.i.d. Rademacher variables, (33) follows from the fact that $f_{G,r}(\lambda)$ defined in (13) is non-decreasing in λ, (34) follows from the definition of $E(\mathcal{P}^{(r)}(G_\pi))$ in (31),and (35) follows from the definition of λ^*_G in (14).

To bound the other two terms in (30), we use Wigner's semicircular law and its rate of convergence for Gaussian Wigner Ensemble, $\mathcal{G}(r,r)$ as defined above. Consider the empirical spectral distribution of $Z \in \mathbb{R}^{r\times r}$ as

$$F_Z(x) = \frac{1}{r}|\{i : \lambda_i(Z) \le x\}|.$$

Lemma 17 (Wigner (1955)) *Define $Z = (1/\sqrt{r})Y$ for $Y \sim \mathcal{G}(r,r)$. Then as $r \to \infty$ the empirical distribution $F_Z(x)$ of Z converges weakly to the distribution $G(x)$ with density*

$$g(t) = \frac{\sqrt{4-t^2}}{2\pi} \quad t \in [-2,2]\,.$$

Lemma 18 (Götze and Tikhomirov (2003)) *For any positive constant $\alpha > 0$, let $\ell_{r,\alpha} = \log r(\log\log r)^\alpha$. There exists an absolute positive constant C and c such that for r large enough,*

$$\mathbb{P}\Big\{ \sup_x \big|F_Z(x) - G(x)\big| \ge r^{-1}\log r \ell^6_{r,\alpha}\Big\} \;\le\; C\exp\big\{-c\ell_{r,\alpha}\big\}\,.$$

To bound the schatten norm of a matrix $Y \sim \mathcal{G}(r,r)$, along with Lemma 17 and Lemma 18 we use the following. If $F(x)$ and $G(x)$ are cumulative distribution functions of densities μ, ν then for any continuous and bounded function f, we have

$$\left|\int f d\mu - \int f d\nu\right| \;\le\; \|f\|_\infty \sup_x \big|F(x) - G(x)\big|\,.$$

Choosing $f(x) = x^k$ for $x \in [-2,2]$, we can see that for $k = O(\log r)$ there exists a constant $C > 2$ such that with probability $1 - 1/80$ it holds that

$$\|(1/\sqrt{r})Y\|_k^k = \left(\int_{-2}^{2} x^k \frac{\sqrt{4-x^2}}{2\pi}dx + o(1)\right)r \;\le\; (2^k + o(1))r \;\le\; C^k r\,. \tag{36}$$

Hence $\|Y\|_k \le Cr^{(1/k+1/2)}$. By construction of distribution $\bar{\mathcal{D}}_1$, for $M_1 \sim \bar{\mathcal{D}}_1$, $\|M_1\|_k = (d/r)\|Y\|_k \le Cdr^{(1/k-1/2)} = \lambda$. Also, by construction $M_2 \sim \bar{\mathcal{D}}_2$ is $M_2 = M_1 + (5/d)\lambda U$ where $\|U\|_k = d$. Using triangle inequality, we have

$$\begin{aligned}\|M_2\|_k &\ge \|(5/d)\lambda U\|_k - \|M_1\|_k \\ &\ge 5\lambda - Cdr^{1/k-1/2} \;=\; 4\lambda\,,\end{aligned}$$

Recall that, incoherence parameter $\mu(M)$ is defined as $\mu(M) = \max_{i\neq j\in[d]} M_{i,j}/(|\sigma_{\max}(M)|\sqrt{r}/d)$. From (36), there exists a constant $0 < C' < 1$ such that with probability $1 - 1/160$ it holds that $\|Y\|_2 \ge C'r$. The integral evaluates to 1 for $k = 2$. Therefore, the largest singular value of M_1 is lower bounded: $|\sigma_{\max}(M_1)| \ge C'd/\sqrt{r}$. Using the fact that there exists a constant C'' such that $\max_{i,j\in[r]}\{Y_{i,j}\} \le C''\sqrt{\log r}$ with probability at least $1 - 1/160$, we have, $\mu(M_1) \le (C''/C')\sqrt{\log r}$. The same $\mu(M_1)$ satisfies the upper bound on diagonals as well. Therefore, using union bound, the second and the third term in (30) are upper bounded by $1/40$.

8.4. Proof of Lemma 12

Observe that for any given permutation π, $\mathcal{P}^{(r)}(G_\pi)$ as defined in Section 4.1 is a clique over a subset of nodes $\widetilde{V}_\pi$, where $|\widetilde{V}_\pi| \le \min\{\ell, r\}$. From the definition of $f_{G,r}(\lambda)$, (13), we have the following:

$$\begin{aligned} & f_{G,r}(\lambda) \\ = \;& \max_\pi \left\{\mathbb{E}_u \exp\left((5/d)^2\lambda^2 \sum_{(i,j)\in\mathcal{P}^{(r)}(G_\pi)} u_i u_j\right)\right\} = \max_\pi \left\{\mathbb{E}_u \exp\left((5/d)^2\lambda^2\big(\sum_{i\in\widetilde{V}_\pi} u_i\big)^2\right)\right\} \\ = \;& \max_\pi \left\{\sum_{t=0}^{\infty} \frac{(5/d)^{2t}\lambda^{2t}\mathbb{E}_u\big[\big(\sum_{i\in\widetilde{V}_\pi} u_i\big)^{2t}\big]}{t!}\right\} \;\le\; \max_\pi \left\{\left(1 + 2\sum_{t=1}^{\infty}\big((5/d)^2\lambda^2|\widetilde{V}_\pi|\big)^t\right)\right\}, \end{aligned}$$

where the inequality follows from the bound in (37). Therefore, from the definition of $\lambda^*_{G,r}$, we have that $\lambda^*_{G,r}$ is lower bounded by $2^{-4}d(\min\{\ell, r\})^{-1/2}$.

To bound $\mathbb{E}(\sum_{i\in\widetilde{V}_\pi} u_i)^{2t}$, for $t \in [1,\infty)$, using Hoeffding bound we have that

$$\mathbb{E}\left|\sum_{i\in\widetilde{V}_\pi} u_i\right|^{2t} \;=\; \int_0^{|\widetilde{V}_\pi|^{2t}} \mathbb{P}\left(\left|\sum_{i\in\widetilde{V}_\pi} u_i\right|^{2t} \ge z\right)dz \;\le\; 2\int_0^{|\widetilde{V}_\pi|^{2t}} \exp\left(\frac{-z^{1/t}}{2|\widetilde{V}_\pi|}\right)dz \;\le\; 2(2|\widetilde{V}_\pi|)^t t!\,, \tag{37}$$

where the integral is evaluated by variable substitution.

8.5. Proof of Lemma 14

For the given pattern graph G and any given permutation π, let $\widetilde{A}_\pi \in \{0,1\}^{r\times r}$ be the adjacency matrix of the graph $\mathcal{P}^{(r)}(G_\pi)$ that is defined in Section 4.1. Observe that for a permutation π, ℓ_π rows of $\widetilde{A}_\pi$ are all-ones and the remaining are all-zeros, where $\ell_\pi \le \min\{\ell, r\}$. Let A_π be a copy of $\widetilde{A}_\pi$ where all the diagonal entries are replaced with zero. Note that $\mathbb{E}_u(u^\top A_\pi u)^{2t+1} = 0$ for all $t \ge 0$, where u_i's are i.i.d. Rademacher random variables. Define $C_\pi \equiv \exp((5/d)^2\lambda^2\ell_\pi)$.

From the definition of $f_{G,r}(\lambda)$, (13), we have the following:

$$\begin{aligned} & f_{G,r}(\lambda) \\ = \;& \max_\pi \left\{\mathbb{E}_u \exp\left((5/d)^2\lambda^2 \sum_{(i,j)\in\mathcal{P}^{(r)}(G_\pi)} u_i u_j\right)\right\} = \max_\pi \left\{C_\pi\mathbb{E}_u \exp\left((5/d)^2\lambda^2(u^\top A_\pi u)\right)\right\} \\ = \;& \max_\pi \left\{C_\pi\sum_{t=0}^{\infty} \frac{(5/d)^{4t}\lambda^{4t}\mathbb{E}_u\big[(u^\top A_\pi u)^{2t}\big]}{(2t)!}\right\} \;\le\; \max_\pi \left\{C_\pi\left(1 + 4\sum_{t=1}^{\infty}\big(2c(5/d)^2\lambda^2\sqrt{\ell_\pi r}\big)^{2t}\right)\right\}, \end{aligned}$$

where the inequality follows from the bound in (38), and c is some absolute constant. Therefore, from the definition of $\lambda^*_{G,r}$, we have that $\lambda^*_{G,r}$ is lower bounded by $cd((\min\{\ell, r\})r)^{-1/4}$.

To bound $\mathbb{E}_u\big[(u^\top A_\pi u)^{2t}\big]$, for $t \in [1,\infty)$, we use Hanson-Wright Inequality. Observe that $\|A_\pi\|_2 \le \sqrt{\ell_\pi r}$, and $\|A_\pi\|_{\rm F}^2 = (r-1)\ell_\pi < \ell_\pi r$.

$$\begin{aligned}
\mathbb{E}_u\big[(u^\top A_\pi u)^{2t}\big] &= \int_0^{(2\sqrt{r\ell_\pi})^{2t}} \mathbb{P}\big((u^\top A_\pi u)^{2t} \ge z\big)dz \;+\; \int_{(2\sqrt{r\ell_\pi})^{2t}}^{(\ell_\pi r)^{2t}} \mathbb{P}\big((u^\top A_\pi u)^{2t} \ge z\big)dz \\
&\le \int_0^{(2\sqrt{r\ell_\pi})^{2t}} \exp\left(\frac{-cz^{1/t}}{4\ell_\pi r}\right)dz + \int_{(2\sqrt{r\ell_\pi})^{2t}}^{(\ell_\pi r)^{2t}} \exp\left(\frac{-cz^{1/(2t)}}{2\sqrt{\ell_\pi r}}\right)dz \\
&\le 2(4\ell_\pi r/c)^t t! + 2(2\sqrt{\ell_\pi r}/c)^{2t}(2t)! \;\le\; 4(2\sqrt{\ell_\pi r}/c)^{2t}(2t)!\,, \qquad (38)
\end{aligned}$$

where the integral is evaluated by variable substitution.

8.6. Proof of Theorem 13

For a clique of size m selected uniformly at random, we derive an upper bound on the variance of our estimator. Following the notations defined in the proof of Theorem 3, we have the following bound on the variance.

$$\begin{aligned}
&\mathrm{Var}\big(\widehat{\Theta}_k(\mathcal{P}_\Omega(M))\big) \\
=\;& 2\sum_{\ell=0}^{k} \sum_{\substack{w\neq w'\in\widetilde{W}\\ |w\cap w'|=\ell}} \mathrm{Covar}\left(\frac{\mathbb{I}(w\subseteq\Omega)\omega_M(w)c(H(w))}{p(H(w))}, \frac{\mathbb{I}(w'\subseteq\Omega)\omega_M(w')c(H(w'))}{p(H(w'))}\right) \\
&+ \sum_{i=1}^{4}\sum_{H\in\mathcal{H}_{k,i}} \left\{\frac{c(H)^2}{p(H)^2} \sum_{w:H(w)=H} \omega_M(w)^2 \mathrm{Var}\Big(\mathbb{I}(w\subseteq\Omega)\Big)\right\} \\
<\;& 2\sum_{\ell=0}^{k} \sum_{\substack{w\neq w'\in W\\ |w\cap w'|=\ell}} \mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\mathbb{I}(w'\subseteq\Omega)\Big]\left(\frac{\omega_M(w)\,\omega_M(w')c(H(w))c(H(w'))}{p(H(w))\,p(H(w'))}\right) \\
&-2\sum_{\ell=0}^{k} \sum_{\substack{w\neq w'\in W\\ |w\cap w'|=\ell}} \mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\Big]\mathbb{E}\Big[\mathbb{I}(w'\subseteq\Omega)\Big]\left(\frac{\omega_M(w)\,\omega_M(w')c(H(w))c(H(w'))}{p(H(w))\,p(H(w'))}\right) \\
&+ \sum_{i=1}^{4}\sum_{H\in\mathcal{H}_{k,i}} \sum_{w:H(w)=H} \frac{c(H)^2\omega_M(w)^2}{p(H)^2}\mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\Big]\,. \qquad (39)
\end{aligned}$$

where we abuse the earlier defined notation $|w\cap w'|$ to denote the number of overlapping nodes in the two walks $w, w' \in W$ instead of number of overlapping edges. Note that in pattern sampling, covariance term for two walks that do not have any overlapping node is not zero. As earlier, we provide bound on each of the terms in (39).

Probability of any walk w being sampled is $\mathbb{P}[w\in\Omega] = \binom{m}{\ell}/\binom{d}{\ell} \le f(\ell)m^\ell/d^\ell$, where ℓ is the number of unique nodes that the walk traverses and $f(\ell)$ is an exponential function in ℓ. Recall that off diagonals of matrix M are bounded by $|M_{ij}| \le \alpha = \mu\sigma_{\max}\sqrt{r}/d$ and the diagonals are bounded by $|M_{ii}| \le \mu\sigma_{\max} r/d$. We have,

$$\sum_{H\in\mathcal{H}_{k,i}} \sum_{w:H(w)=H} \frac{c(H)^2\,\omega_M(w)^2}{p(H)^2}\mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\Big] \;\le$$

$$\begin{cases}
\dfrac{d^k}{2k}\dfrac{f(k)^2\alpha^{2k}d^k}{m^k} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^k}\,, & \text{for } i=1\;, \qquad (40)\\
\left(\dfrac{d^2}{m}\right)^{k-1} f(k)^2\alpha^{2k} = \dfrac{m}{d^2}\dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^k}\,, & \text{for } i=2\;, \qquad (41)\\
\dfrac{d^2}{m}r^k\alpha^{2k} = \dfrac{r^k m^{k-1}}{d^{2k-2}}\dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^k}\,, & \text{for } i=4\,, \qquad (42)
\end{cases}$$

and for $i=3$ and for $1\le s\le k-1$, we have

$$\begin{aligned}
&\sum_{H\in\mathcal{H}_{k,3,s}}\sum_{w:H(w)=H} \frac{c(H)^2\,\omega_M(w)^2}{p(H)^2}\mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\Big] \\
&\le \left(\frac{d^2}{m}\right)^{k-s} f(k)^2\alpha^{2k}r^s = \frac{m^s r^s}{d^{2s}}\frac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^k}\,, \qquad (43)
\end{aligned}$$

For any two walks w, w' with $\ell \ge 0$ overlapping nodes, $\mathbb{P}[w,w'\in\Omega]/(\mathbb{P}[w\in\Omega]\mathbb{P}[w'\in\Omega]) \le f(k)d^\ell/m^\ell$. For $w\neq w'$ and $|w\cap w'| = \ell$, where the range of ℓ varies across equations depending upon the set to which w, w' belongs, we have the following:

$$\sum_{\substack{w\neq w'\in W\\ |w\cap w'|=\ell\\ H(w)\in\mathcal{H}_{k,i,s}\\ H(w')\in\mathcal{H}_{k,i',s'}}} \Big(\mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\mathbb{I}(w'\subseteq\Omega)\Big] - \mathbb{E}\Big[\mathbb{I}(w\subseteq\Omega)\Big]\mathbb{E}\Big[\mathbb{I}(w'\subseteq\Omega)\Big]\Big)\left(\frac{\omega_M(w)\,\omega_M(w')c(H(w))c(H(w'))}{p(H(w))\,p(H(w'))}\right) \;\le$$

$$\begin{cases}
\dfrac{f(k)^2 d^\ell}{m^\ell}\dfrac{(\mu\sigma_{\max})^{2k}r^2}{d^\ell} = \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}\max\{r^2,r^\ell\}}{m^\ell}, & \text{for } i=i'=1, \ell\ge 1 \quad (44)\\
\dfrac{m^{2k-1}}{d^{2k}}\dfrac{d^{2k}f(k)^2(\mu\sigma_{\max})^{2k}r^2}{m^{2k}} = \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^2}{m}, & \text{for } i=i'=1, \ell=0 \quad (45)\\
\dfrac{f(k)^2 d^\ell d^{2k-2-\ell}\alpha^{2k}}{m^\ell} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^\ell d^2}, & \text{for } i=i'=2 \quad (46)\\
\dfrac{f(k)^2 d^\ell d^{2k-s-s'-\ell}\alpha^{2k}(\sqrt{r})^{s+s'}}{m^\ell} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^\ell d}, & \text{for } i=i'=3 \quad (47)\\
f(k)^2 d^2\alpha^{2k}(\sqrt{r})^{2k} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{d^{2k-2}/r^k}, & \text{for } i=i'=4 \quad (48)\\
\dfrac{f(k)^2 d^\ell d^{2k-1-\ell}\alpha^{2k}}{m^\ell} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^\ell d}, & \text{for } i=1, i'=2 \quad (49)\\
\dfrac{f(k)^2 d^\ell d^{2k-s-\ell}\alpha^{2k}(\sqrt{r})^s}{m^\ell} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^\ell d/\sqrt{r}}, & \text{for } i=1, i'=3 \quad (50)\\
\dfrac{f(k)^2 d^\ell d^{k+1-\ell}\alpha^{2k}(\sqrt{r})^k}{m^\ell} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^\ell d^{k-1}/(\sqrt{r})^k}, & \text{for } i=1, i'=4\,, \quad (51)\\
\dfrac{f(k)^2 d^\ell d^{2k-1-s-\ell}\alpha^{2k}(\sqrt{r})^s}{m^\ell} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^\ell d^2/\sqrt{r}}, & \text{for } i=2, i'=3 \quad (52)\\
\dfrac{f(k)^2 d^\ell d^{k-\ell}\alpha^{2k}(\sqrt{r})^k}{m^\ell} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^\ell d^k(\sqrt{r})^k}, & \text{for } i=2, i'=4\,, \quad (53)\\
\dfrac{f(k)^2 d^\ell d^{k+1-s-\ell}\alpha^{2k}(\sqrt{r})^{s+k}}{m^\ell} \le \dfrac{f(k)^2(\mu\sigma_{\max})^{2k}r^k}{m^\ell d^{k-1}/(\sqrt{r})^k}, & \text{for } i=3, i'=4\,, \quad (54)
\end{cases}$$

Where (44) and (45) both use (56), and (45) also uses (55). Note that ℓ is zero in (48). Collecting all the terms, and using Chebyshev's inequality, along with $\|M\|_k^k \geq r(\sigma_{\min})^k$, we get the desired result.

For any two disjoint simple cycles $w \neq w' \in \mathcal{H}_{k,1}$ with $|w \cap w'| = 0$, we have the following

$$\begin{aligned} \mathbb{P}\big[w \in \Omega\big] - \mathbb{P}\big[w \in \Omega \;\big|\; w' \in \Omega\big] &= \frac{\binom{m}{k}}{\binom{d}{k}} - \frac{\binom{m-k}{k}}{\binom{d-k}{k}} \\ &\leq \frac{m^k}{(d-k+1)^k} - \frac{(m-2k+1)^k}{(d-k)^k} \leq \frac{f(k)m^{k-1}}{d^k}, \end{aligned} \quad (55)$$

where the last inequality assumes that $k < d/2$.

Lemma 19 *For $k = 3$, and any $0 \leq \ell \leq k$*

$$\sum_{w \neq w' \in \mathcal{H}_{k,1} : |w \cap w'| = \ell} \omega_M(w)\omega_M(w') \;\leq\; \frac{f(k)(\mu\sigma_{\max})^{2k} \max\{r^2, r^\ell\}}{d^\ell}. \quad (56)$$

Although we give a proof for $k = 3$ only, we are intentionally writing the lemma for general k as we expect the lemma holds for all $k \geq 3$. The joint walk $w \neq w' \in \mathcal{H}_{k,1} : |w \cap w'| = \ell$ corresponds to $H(w) = D_{27}$, for $\ell = 1$; and $H(w) = D_{23}$, for $\ell = 2$ in Figure 12. Define $\tilde{M} \equiv M - \mathrm{diag}(M)$, and let $\odot$ denote the Hadamard product of two matrices. We have,

$$\sum_{w \neq w' \in \mathcal{H}_{k,1} : |w \cap w'| = 2} \omega_M(w)\omega_M(w') \;=\; (1/4) \sum_{i,j \in [d]} \Big(\big(\tilde{M}^2 \odot \tilde{M}^2 - (\tilde{M} \odot \tilde{M})^2\big) \odot (\tilde{M} \odot \tilde{M}) \Big)_{i,j} \quad (57)$$

Let's denote the quantity in (57) by C_1, we have,

$$\begin{aligned} &\sum_{w \neq w' \in \mathcal{H}_{k,1} : |w \cap w'| = 1} \omega_M(w)\omega_M(w') \\ &= \; (1/8) \sum_{i \in [d]} \Big(\mathrm{diag}(\tilde{M}^3) \odot \mathrm{diag}(\tilde{M}^3) - 2\mathrm{diag}((\tilde{M} \odot \tilde{M})^3) \Big)_i - 2C_1 \,. \end{aligned} \quad (58)$$

It is easy to verify Equation (56) for $k = 3$ and $\ell \in \{1, 2\}$ using the fact that M is a μ incoherent symmetric matrix with its off-diagonals bounded by $\mu\sigma_{\max}(\sqrt{r}/d)$. For $\ell = 0$, quantity in (56) is the sum of each pair of disjoint triangles. For sum of all triangles, we have,

$$\sum_{w \in \mathcal{H}_{k,1}} \omega_M(w) = (1/6) \sum_{i \in [d]} \Big(\mathrm{diag}(\tilde{M}^3) \Big)_i \;\leq\; (\mu\sigma_{\max})^3 r \,. \quad (59)$$

Using Equations (57), (58) and (59), bound for $\ell = 0$ follows immediately. Bound for $\ell = k$, follows by using the fact that $M_{i,j} \leq \mu\sigma_{\max}(\sqrt{r}/d)$ for $i \neq j \in [d]$.

9. k-cyclic pseudographs

We provide an enumeration of all k-cyclic psuedographs for $k \in \{4, 5, 6, 7\}$ in Figures (10–15).

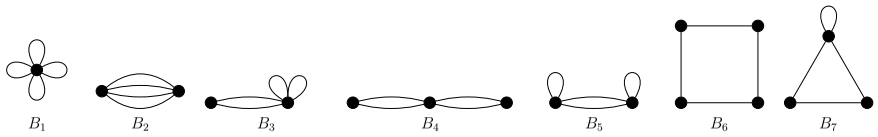


Figure 10: The 4-cyclic pseudographs $\mathcal{H}_4$.

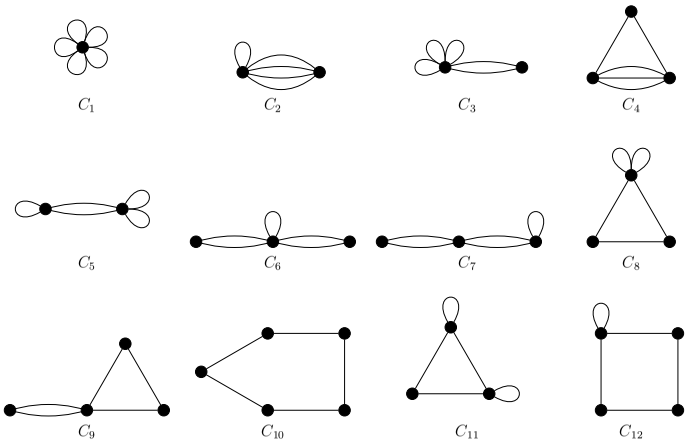


Figure 11: The 5-cyclic pseudographs $\mathcal{H}_5$.

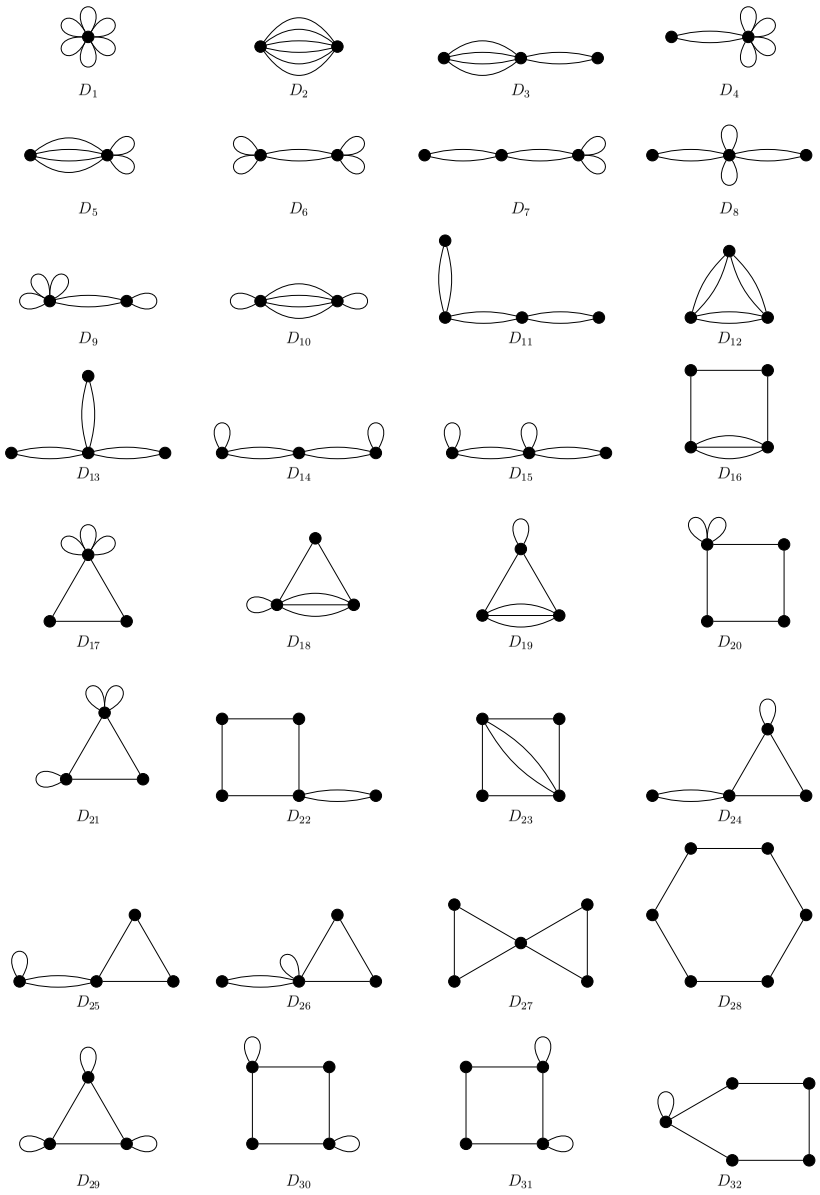


Figure 12: The 6-cyclic pseudographs $\mathcal{H}_6$.

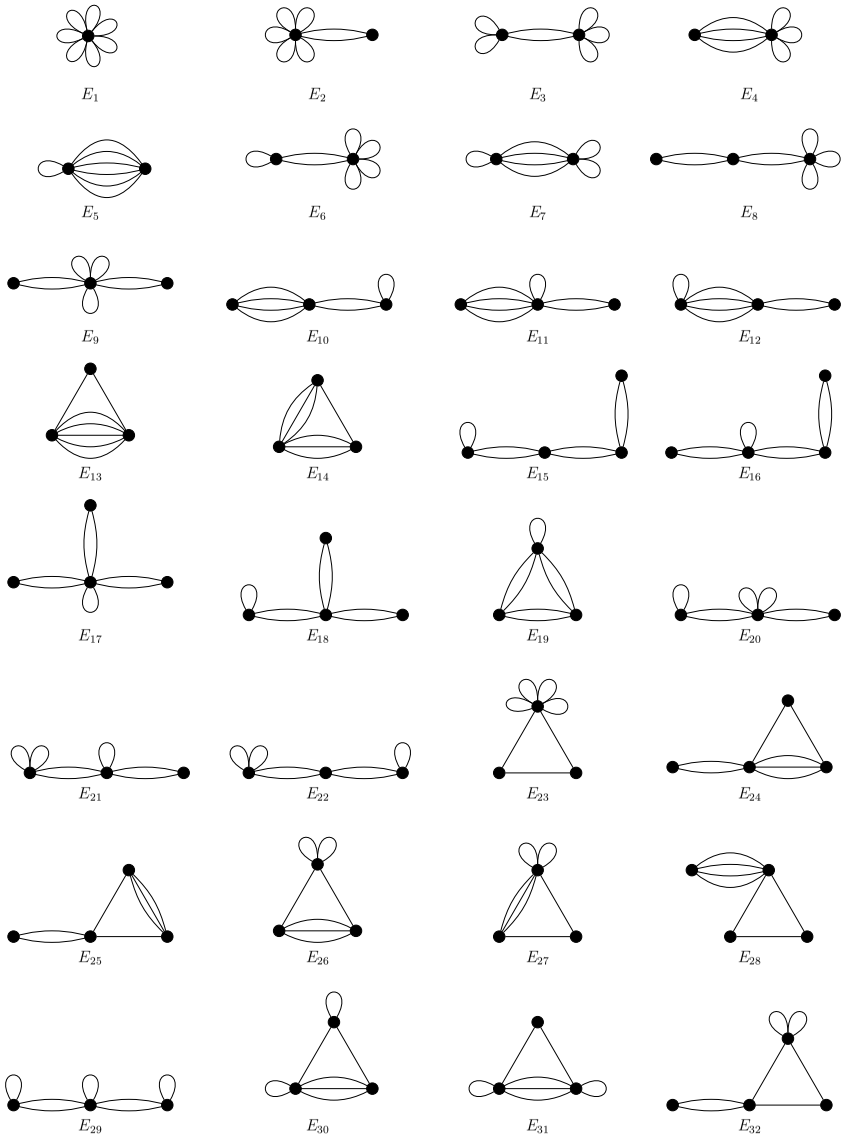


Figure 13: The 7-cyclic pseudographs $\mathcal{H}_7$

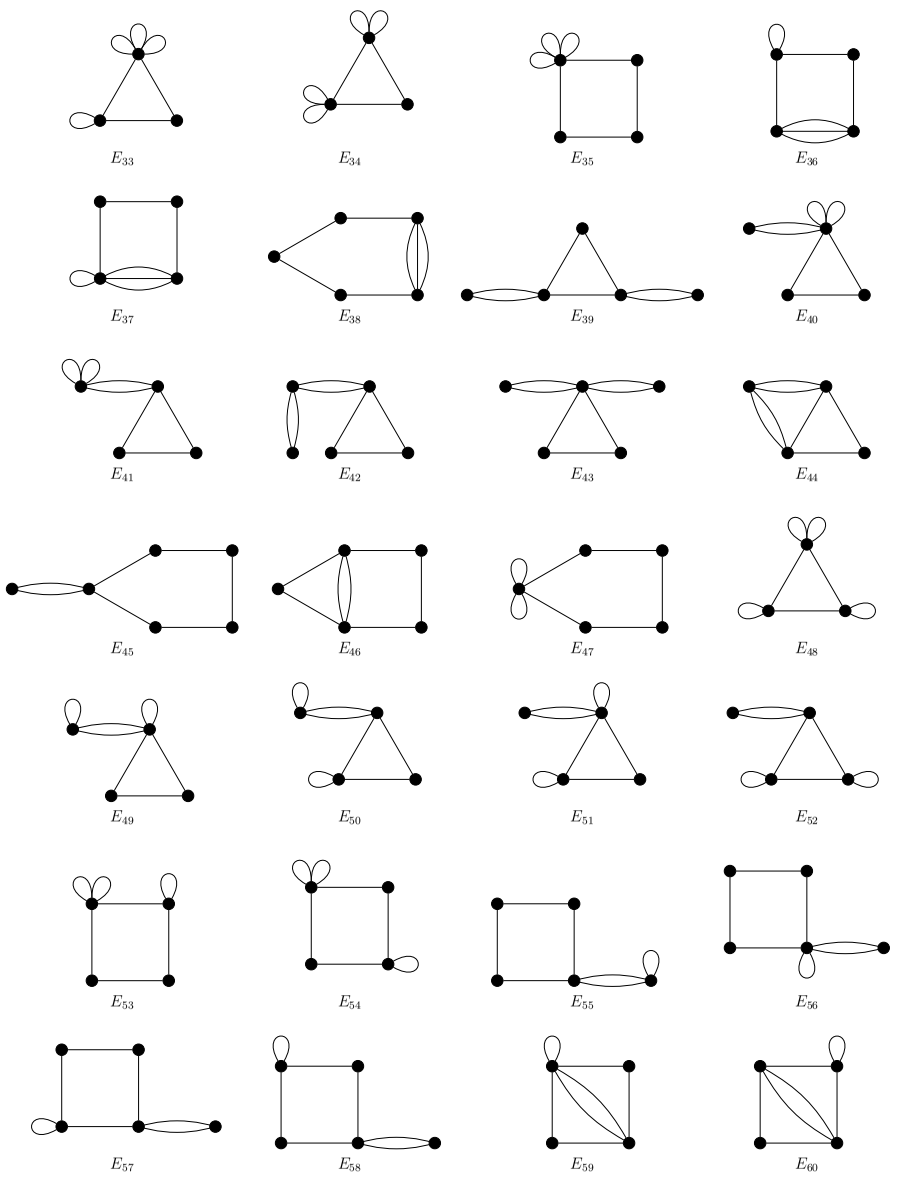


Figure 14: The 7-cyclic pseudographs $\mathcal{H}_7$

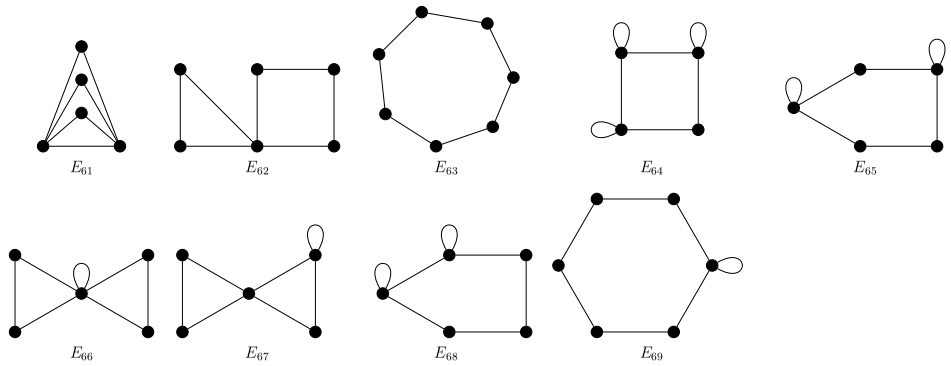


Figure 15: The 7-cyclic pseudographs $\mathcal{H}_7$.

10. Efficient computation of $\omega_M(H)$ for $k \in \{4,5,6,7\}$

In this section we provide the complete matrix oeprations for copmuting $\gamma_M(H)$'s. Equations (60) - (66) give expressions to compute $\gamma_M(H)$ for $H \in \mathcal{H}_4$ as labeled in Figure 10. Equations (67) - (78) give expressions to compute $\gamma_M(H)$ for $H \in \mathcal{H}_5$ as labeled in Figure 11. Equations (79) - (110) give expressions to compute $\gamma_M(H)$ for $H \in \mathcal{H}_6$ as labeled in Figure 12. Equations (111) - (203) give expressions to compute $\gamma_M(H)$ for $H \in \mathcal{H}_7$ as labeled in Figure 15.

For brevity of notations and readability, we define the following additional notations. Let $A \odot B$ denote the Hadamard product. For $A \in \mathbb{R}^{d\times d}$, let $\mathrm{sum}(A)$ denote a vector $v \in \mathbb{R}^d$ such that $v_i = \sum_{j\in[d]} A_{i,j}$. With a slight abuse of notation, for $v \in \mathbb{R}^d$, let $\mathrm{sum}(v)$ denote sum of all elements of v that is $\mathrm{sum}(v) = \sum_{i\in[d]} v_i$. Let $\mathrm{sum}(\gamma_M(H_i) : \gamma_M(H_j)) \equiv \sum_{i'=i}^{j} \gamma_M(H_{i'})$. Define $R \equiv \mathbb{1}_{d\times d} - \mathrm{diag}(\mathbb{1}_{d\times d})$, that is R is an all-ones matrix except on diagonals which are zeros. Further, for brevity, we omit the subscript M from the notations $\gamma_M(H), O_M$ and D_M.

$$\gamma(B_1) = \mathrm{sum}(\mathrm{sum}(D \odot D \odot D \odot D)) \tag{60}$$
$$\gamma(B_2) = \mathrm{sum}(\mathrm{sum}(O \odot O \odot O \odot O)) \tag{61}$$
$$\gamma(B_3) = 4\,\mathrm{tr}(O*O*D*D) \tag{62}$$
$$\gamma(B_4) = 2\,\mathrm{sum}(\mathrm{sum}((O \odot O)*(O \odot O) \odot R)) \tag{63}$$
$$\gamma(B_5) = 2\,\mathrm{tr}(O*D*O*D) \tag{64}$$
$$\gamma(B_6) = \mathrm{tr}(O*O*O*O) - \mathrm{sum}(\gamma(B_2) : \gamma(B_4)) \tag{65}$$
$$\gamma(B_7) = \mathrm{tr}(M*M*M*M) - \mathrm{sum}(\gamma(B_1) : \gamma(B_6)) \tag{66}$$

$$\gamma(C_1) = \operatorname{tr}(D \odot D \odot D \odot D \odot D) \tag{67}$$
$$\gamma(C_2) = 5\ \operatorname{sum}(\operatorname{sum}(D * O \odot O \odot O \odot O)) \tag{68}$$
$$\gamma(C_3) = 5\ \operatorname{sum}(\operatorname{sum}((D \odot D \odot D) * (O \odot O))) \tag{69}$$
$$\gamma(C_4) = 5\ \operatorname{tr}((O \odot O \odot O) * O * O) \tag{70}$$
$$\gamma(C_5) = 5\ \operatorname{sum}(\operatorname{sum}(D * (O \odot O) * (D \odot D))) \tag{71}$$
$$\gamma(C_6) = 5\ \operatorname{sum}(\operatorname{sum}(((O \odot O) * D * (O \odot O)) \odot R)) \tag{72}$$
$$\gamma(C_7) = 5\ \operatorname{sum}(\operatorname{sum}((D * (O \odot O) * (O \odot O)) \odot R)) \tag{73}$$
$$\gamma(C_8) = 5\ \operatorname{tr}(O * O * O * (D \odot D)) \tag{74}$$
$$\gamma(C_9) = 5\ \operatorname{sum}(\operatorname{diag}(O \odot O \odot O) \odot \operatorname{sum}(O \odot O)) - 10\ \operatorname{tr}((O \odot O \odot O) * O * O)) \tag{75}$$
$$\gamma(C_{10}) = \operatorname{tr}(O * O * O * O * O) - \gamma(C_4) - \gamma(C_9) \tag{76}$$
$$\gamma(C_{11}) = 5\ \operatorname{tr}(O * D * O * D * O) \tag{77}$$
$$\gamma(C_{12}) = \operatorname{tr}(M * M * M * M * M) - \operatorname{sum}(\gamma(C_1) : \gamma(C_{11})) \tag{78}$$

$$\gamma(D_1) = \operatorname{sum}(\operatorname{sum}(D \odot D \odot D \odot D \odot D \odot D)) \tag{79}$$
$$\gamma(D_2) = \operatorname{sum}(\operatorname{sum}(O \odot O \odot O \odot O \odot O \odot O)) \tag{80}$$
$$\gamma(D_3) = 6\ \operatorname{sum}(\operatorname{sum}(((O \odot O) * (O \odot O \odot O \odot O)) \odot R)) \tag{81}$$
$$\gamma(D_4) = 6\ \operatorname{sum}(\operatorname{sum}(((O \odot O) * (D \odot D \odot D \odot D)) \odot R)) \tag{82}$$
$$\gamma(D_5) = 9\ \operatorname{sum}(\operatorname{sum}(((D \odot D) * (O \odot O \odot O \odot O)) \odot R)) \tag{83}$$
$$\gamma(D_6) = 3\ \operatorname{sum}(\operatorname{sum}(((D \odot D) * (O \odot O) * (D \odot D)) \odot R)) \tag{84}$$
$$\gamma(D_7) = 6\ \operatorname{sum}(\operatorname{sum}(((D \odot D) * (O \odot O) * (O \odot O)) \odot R)) \tag{85}$$
$$\gamma(D_8) = 9\ \operatorname{sum}(\operatorname{sum}(((O \odot O) * (D \odot D) * (O \odot O)) \odot R)) \tag{86}$$
$$\gamma(D_9) = 6\ \operatorname{sum}(\operatorname{sum}(((D \odot D \odot D) * (O \odot O) * D) \odot R)) \tag{87}$$
$$\gamma(D_{10}) = 6\ \operatorname{sum}(\operatorname{sum}((D * (O \odot O \odot O \odot O) * D) \odot R)) \tag{88}$$
$$\gamma(D_{11}) = 3\ \operatorname{sum}\Big(\big(\operatorname{sum}(((O \odot O) * (O \odot O)) \odot R)\big) \odot \big(\operatorname{sum}(O \odot O)\big) - \operatorname{sum}\Big(\big(((O \odot O \odot O \odot O) * (O \odot O))\big) \odot R\Big) - \operatorname{diag}((O \odot O) * (O \odot O) * (O \odot O))\Big) \tag{89}$$
$$\gamma(D_{12}) = 4\ \operatorname{tr}((O \odot O) * (O \odot O) * (O \odot O)) \tag{90}$$
$$\gamma(D_{13}) = 2\ \operatorname{sum}\Big((\operatorname{sum}(O \odot O)) \odot (\operatorname{sum}(O \odot O)) \odot (\operatorname{sum}(O \odot O)) - \operatorname{sum}((O \odot O \odot O \odot O \odot O \odot O)) - 3\ \big((\operatorname{sum}(O \odot O \odot O \odot O)) \odot (\operatorname{sum}(O \odot O)) - (\operatorname{sum}(O \odot O \odot O \odot O \odot O \odot O))\big)\Big) \tag{91}$$
$$\gamma(D_{14}) = 3\ \operatorname{sum}(\operatorname{sum}((D * (O \odot O) * (O \odot O) * D) \odot R)) \tag{92}$$
$$\gamma(D_{15}) = 12\ \operatorname{sum}(\operatorname{sum}((D * (O \odot O) * D * (O \odot O)) \odot R)) \tag{93}$$
$$\gamma(D_{16}) = 6\ \operatorname{sum}\Big(\operatorname{sum}(((O \odot O \odot O) * O) \odot R \odot (O * O)) - \operatorname{sum}(((O \odot O \odot O \odot O) * (O \odot O)) \odot R)\Big) \tag{94}$$
$$\gamma(D_{17}) = 6\ \operatorname{tr}((D \odot D \odot D) * O * O * O) \tag{95}$$
$$\gamma(D_{18}) = 24\ \operatorname{tr}(D * (O \odot O \odot O) * O * O) \tag{96}$$
$$\gamma(D_{19}) = 6\ \operatorname{tr}(D * O * (O \odot O \odot O) * O) \tag{97}$$
$$\gamma(D_{20}) = 6\ \Big(\operatorname{sum}(\operatorname{sum}((O * O) \odot ((O * (D \odot D) * O) \odot R))) - \operatorname{sum}(\operatorname{sum}(((O \odot O) * (D \odot D) * (O \odot O)) \odot R))\Big) \tag{98}$$
$$\gamma(D_{21}) = 12\ \operatorname{tr}(O * (D \odot D) * O * D * O) \tag{99}$$
$$\gamma(D_{22}) = 6\ \Big(\operatorname{sum}\Big(\operatorname{sum}\big(((O * O) \odot R \odot (O * O) - ((O \odot O) * (O \odot O)) \odot R)\big) \odot \operatorname{sum}(O \odot O)\Big) - 2\ \operatorname{sum}\Big(\operatorname{sum}((((O \odot O \odot O) * O) \odot R \odot (O * O) - ((O \odot O \odot O \odot O) * (O \odot O)) \odot R))\Big) - \operatorname{sum}\Big(\operatorname{sum}\Big(\big((O * O) \odot R \odot (O * O) - ((O \odot O) * (O \odot O)) \odot R\big) \odot (O \odot O)\Big)\Big)\Big) \tag{100}$$
$$\gamma(D_{23}) = 9\ \operatorname{sum}(\operatorname{sum}(((O * O) \odot R \odot (O * O) - ((O \odot O) * (O \odot O)) \odot R) \odot ((O \odot O)))) \tag{101}$$
$$\gamma(D_{24}) = 12\ \operatorname{sum}(\operatorname{diag}(O * D * O * O) \odot \operatorname{sum}((O \odot O)) - \operatorname{diag}((O \odot O \odot O) * D * O * O) - \operatorname{diag}((O \odot O \odot O) * O * D * O)) \tag{102}$$
$$\gamma(D_{25}) = 6\ \operatorname{sum}(\operatorname{diag}(O * O * O) \odot \operatorname{sum}((O \odot O) * D) - 2\ \operatorname{diag}((O \odot O \odot O) * D * O * O)) \tag{103}$$
$$\gamma(D_{26}) = 12\ \operatorname{sum}(\operatorname{diag}(O * O * O) \odot \operatorname{diag}(D) \odot \operatorname{sum}((O \odot O)) - \operatorname{diag}((O \odot O \odot O) * O * O) \odot \operatorname{diag}(D)) \tag{104}$$
$$\gamma(D_{27}) = 3\ \operatorname{sum}\Big(\operatorname{diag}(O * O * O) \odot \operatorname{diag}(O * O * O) - 2\ \operatorname{diag}((O \odot O) * (O \odot O) * (O \odot O))\Big) - (4/3)\gamma(D_{23}) \tag{105}$$

$$\begin{aligned}
\gamma(D_{28}) &= \operatorname{tr}(O{*}O{*}O{*}O{*}O{*}O) - \gamma(D_2) - \gamma(D_3) - \gamma(D_{11}) - \gamma(D_{12}) - \gamma(D_{13}) \\
&\quad -\gamma(D_{16}) - \gamma(D_{22}) - \gamma(D_{23}) - \gamma(D_{27}) && (106)\\
\gamma(D_{29}) &= 2\operatorname{tr}(D{*}O{*}D{*}O{*}D{*}O) && (107)\\
\gamma(D_{30}) &= 3\operatorname{sum}(\operatorname{sum}((O{*}D{*}O)\odot R\odot(O{*}D{*}O)) - \operatorname{sum}(((O\odot O){*}(D\odot D){*}(O\odot O))\odot R)) && (108)\\
\gamma(D_{31}) &= 6\operatorname{sum}(\operatorname{sum}((O{*}D{*}O{*}D)\odot R\odot(O{*}O)) - \operatorname{sum}(((O\odot O){*}D{*}(O\odot O){*}D)\odot R)) && (109)\\
\gamma(D_{32}) &= \operatorname{tr}(M{*}M{*}M{*}M{*}M{*}M) - \operatorname{tr}(O{*}O{*}O{*}O{*}O{*}O) - \operatorname{sum}(\gamma(D_1) : \gamma(D_{26})) + \gamma(D_2) + \gamma(D_3) + \\
&\quad \gamma(D_{11}) + \gamma(D_{12}) + \gamma(D_{13}) + \gamma(D_{16}) + \gamma(D_{22}) + \gamma(D_{23}) - \gamma(D_{29}) - \gamma(D_{30}) - \gamma(D_{31}) && (110)
\end{aligned}$$

$$\begin{aligned}
\gamma(E_1) &= \operatorname{sum}(\operatorname{diag}((D\odot D\odot D\odot D\odot D\odot D\odot D))) && (111)\\
\gamma(E_2) &= 7\operatorname{sum}(\operatorname{sum}((O\odot O){*}(D\odot D\odot D\odot D\odot D))) && (112)\\
\gamma(E_3) &= 7\operatorname{sum}(\operatorname{sum}(((D\odot D){*}(O\odot O){*}(D\odot D\odot D))\odot R)) && (113)\\
\gamma(E_4) &= 14\operatorname{sum}(\operatorname{sum}((O\odot O\odot O\odot O){*}(D\odot D\odot D))) && (114)\\
\gamma(E_5) &= 7\operatorname{sum}(\operatorname{sum}((O\odot O\odot O\odot O\odot O\odot O){*}D)) && (115)\\
\gamma(E_6) &= 7\operatorname{sum}(\operatorname{sum}((D{*}(O\odot O){*}(D\odot D\odot D\odot D))\odot R)) && (116)\\
\gamma(E_7) &= 21\operatorname{sum}(\operatorname{sum}((D{*}(O\odot O\odot O\odot O){*}(D\odot D))\odot R)) && (117)\\
\gamma(E_8) &= 7\operatorname{sum}(\operatorname{sum}(((O\odot O){*}(O\odot O){*}(D\odot D\odot D))\odot R)) && (118)\\
\gamma(E_9) &= 14\operatorname{sum}(\operatorname{sum}(((O\odot O){*}(D\odot D\odot D){*}(O\odot O))\odot R)) && (119)\\
\gamma(E_{10}) &= 7\operatorname{sum}(\operatorname{sum}(((O\odot O\odot O\odot O){*}(O\odot O){*}D)\odot R)) && (120)\\
\gamma(E_{11}) &= 21\operatorname{sum}(\operatorname{sum}(((O\odot O\odot O\odot O){*}D{*}(O\odot O))\odot R)) && (121)\\
\gamma(E_{12}) &= 14\operatorname{sum}(\operatorname{sum}((D{*}(O\odot O\odot O\odot O){*}(O\odot O))\odot R)) && (122)\\
\gamma(E_{13}) &= 7\operatorname{tr}((O\odot O\odot O\odot O\odot O){*}O{*}O) && (123)\\
\gamma(E_{14}) &= 14\operatorname{tr}((O\odot O\odot O){*}O{*}(O\odot O\odot O)) && (124)\\
\gamma(E_{15}) &= 7\operatorname{sum}(\operatorname{sum}(((O\odot O){*}(O\odot O))\odot R)\odot\operatorname{sum}((O\odot O){*}D) - \operatorname{sum}(((O\odot O\odot O\odot O){*}D{*}(O\odot O))\odot R) \\
&\quad -\operatorname{diag}(((O\odot O){*}D{*}(O\odot O){*}(O\odot O)))) && (125)\\
\gamma(E_{16}) &= 14\operatorname{sum}((\operatorname{sum}(((O\odot O){*}(O\odot O))\odot R)\odot\operatorname{sum}((O\odot O)) - \operatorname{sum}(((O\odot O\odot O\odot O){*}(O\odot O))\odot R) \\
&\quad -\operatorname{diag}(((O\odot O){*}(O\odot O){*}(O\odot O))))\odot\operatorname{diag}(D)) && (126)\\
\gamma(E_{17}) &= 7\operatorname{sum}((\operatorname{sum}(O\odot O)\odot\operatorname{sum}(O\odot O)\odot\operatorname{sum}(O\odot O)) - \operatorname{sum}((O\odot O\odot O\odot O\odot O\odot O)) \\
&\quad -3\ (\operatorname{sum}((O\odot O\odot O\odot O))\odot\operatorname{sum}((O\odot O)) - \operatorname{sum}((O\odot O\odot O\odot O\odot O\odot O))))\odot\operatorname{diag}(D)) && (127)\\
Z_1 &\equiv 0.5\ ((\operatorname{sum}(O\odot O)\odot\operatorname{sum}(O\odot O)) - \operatorname{sum}((O\odot O\odot O\odot O))) \\
\gamma(E_{18}) &= 14\operatorname{sum}(\operatorname{sum}((O\odot O){*}D)\odot Z_1 - \operatorname{sum}((O\odot O\odot O\odot O){*}D)\odot\operatorname{sum}((O\odot O)) \\
&\quad +\operatorname{sum}((O\odot O\odot O\odot O\odot O\odot O){*}D)) && (128)\\
\gamma(E_{19}) &= 28\operatorname{sum}(\operatorname{diag}((O\odot O){*}(O\odot O){*}(O\odot O))\odot\operatorname{diag}(D)) && (129)\\
\gamma(E_{20}) &= 21\operatorname{sum}(\operatorname{sum}((D{*}(O\odot O){*}(D\odot D){*}(O\odot O))\odot R)) && (130)\\
\gamma(E_{21}) &= 14\operatorname{sum}(\operatorname{sum}(((D\odot D){*}(O\odot O){*}D{*}(O\odot O))\odot R)) && (131)\\
\gamma(E_{22}) &= 7\operatorname{sum}(\operatorname{sum}((D{*}(O\odot O){*}(O\odot O){*}(D\odot D))\odot R)) && (132)\\
\gamma(E_{23}) &= 7\operatorname{sum}(\operatorname{diag}(O{*}O{*}O)\odot\operatorname{diag}((D\odot D\odot D\odot D))) && (133)\\
\gamma(E_{24}) &= 28\operatorname{sum}(\operatorname{diag}((O\odot O\odot O){*}O{*}O)\odot\operatorname{sum}((O\odot O)) - \operatorname{diag}((O\odot O\odot O\odot O\odot O){*}O{*}O) \\
&\quad -\operatorname{diag}((O\odot O\odot O){*}O{*}(O\odot O\odot O))) && (134)\\
\gamma(E_{25}) &= 7\operatorname{sum}(\operatorname{diag}(O{*}(O\odot O\odot O){*}O)\odot\operatorname{sum}((O\odot O)) - 2\operatorname{diag}((O\odot O\odot O){*}(O\odot O\odot O){*}O)) && (135)\\
\gamma(E_{26}) &= 7\operatorname{sum}(\operatorname{diag}(O{*}(O\odot O\odot O){*}O)\odot\operatorname{diag}((D\odot D))) && (136)\\
\gamma(E_{27}) &= 42\operatorname{sum}(\operatorname{diag}((O\odot O\odot O){*}O{*}O)\odot\operatorname{diag}((D\odot D))) && (137)\\
\gamma(E_{28}) &= 7\operatorname{sum}(\operatorname{diag}(O{*}O{*}O)\odot\operatorname{sum}((O\odot O\odot O\odot O)) - 2\operatorname{diag}((O\odot O\odot O\odot O\odot O){*}O{*}O)) && (138)\\
\gamma(E_{29}) &= 7\operatorname{sum}(\operatorname{sum}((D{*}(O\odot O){*}D{*}(O\odot O){*}D)\odot R)) && (139)\\
\gamma(E_{30}) &= 28\operatorname{sum}(\operatorname{diag}(O{*}D{*}(O\odot O\odot O){*}O)\odot\operatorname{diag}(D)) && (140)
\end{aligned}$$

$$\gamma(E_{31}) = 28\,\mathrm{tr}(O{*}D{*}(O{\odot}O{\odot}O){*}D{*}O) \quad (141)$$
$$\gamma(E_{32}) = 14\,\mathrm{sum}(\mathrm{diag}(O{*}(D{\odot}D){*}O{*}O){\odot}\mathrm{sum}((O{\odot}O)) - \mathrm{diag}((O{\odot}O{\odot}O){*}O{*}(D{\odot}D){*}O) - \mathrm{diag}((O{\odot}O{\odot}O){*}(D{\odot}D){*}O{*}O)) \quad (142)$$
$$\gamma(E_{33}) = 14\,\mathrm{sum}(\mathrm{diag}(O{*}D{*}O{*}O){\odot}\mathrm{diag}((D{\odot}D{\odot}D))) \quad (143)$$
$$\gamma(E_{34}) = 7\mathrm{tr}(O{*}(D{\odot}D){*}O{*}(D{\odot}D){*}O) \quad (144)$$
$$\gamma(E_{35}) = 7(\mathrm{sum}(\mathrm{sum}((((O{*}O){\odot}R){\odot}((O{*}(D{\odot}D{\odot}D){*}O){\odot}R)))) - \mathrm{sum}(\mathrm{sum}(((O{\odot}O){*}(D{\odot}D{\odot}D){*}(O{\odot}O)){\odot}R))) \quad (145)$$
$$\gamma(E_{36}) = 14\,\mathrm{sum}(\mathrm{sum}(((O{\odot}O{\odot}O){*}O){\odot}R{\odot}(O{*}D{*}O)) - \mathrm{sum}(((O{\odot}O{\odot}O{\odot}O){*}D{*}(O{\odot}O)){\odot}R)) \quad (146)$$
$$\gamma(E_{37}) = 28\,\mathrm{sum}(\mathrm{sum}(((O{\odot}O{\odot}O){*}D{*}O){\odot}R{\odot}(O{*}O)) - \mathrm{sum}(((O{\odot}O{\odot}O{\odot}O){*}D{*}(O{\odot}O)){\odot}R)) \quad (147)$$
$$Z_2 \equiv (((O{*}O){\odot}R){*}O - O{\odot}(\mathbb{1}_{d\times 1}{*}(\mathrm{sum}((O{\odot}O)^\top))^\top - (O{\odot}O))){\odot}R \quad (148)$$
$$Z_3 \equiv (O{\odot}((O{*}O){\odot}R)){\odot}R \quad (149)$$
$$Z_4 \equiv (O{\odot}(((O{\odot}O{\odot}O{\odot}O{\odot}O){*}O){\odot}R)){\odot}R \quad (150)$$
$$Z_6 \equiv ((O{\odot}O{\odot}O){\odot}((O{*}O){\odot}R)){\odot}R \quad (151)$$
$$Z_7 \equiv (O{\odot}(((O{\odot}O{\odot}O){*}(O{\odot}O{\odot}O)){\odot}R)){\odot}R \quad (152)$$
$$\gamma(E_{38}) = 7\,\mathrm{sum}(\mathrm{sum}((((O{\odot}O{\odot}O){*}O){\odot}R{\odot}Z_2 - (((O{\odot}O{\odot}O{\odot}O){*}Z_3){\odot}R - Z_4) - ((Z_6{*}(O{\odot}O)){\odot}R - Z_7)))) \quad (153)$$
$$Z_7 \equiv 0.5\,\mathrm{sum}(\mathrm{sum}(O{\odot}(((O{\odot}O){*}(O{\odot}O)){\odot}R){\odot}((O{*}O){\odot}R) - O{\odot}(((O{\odot}O{\odot}O){*}(O{\odot}O{\odot}O)){\odot}R))) \quad (154)$$
$$\gamma(E_{39}) = 7\,(\mathrm{sum}(\mathrm{sum}((O{\odot}((O{*}O){\odot}R){\odot}(\mathrm{sum}((O{\odot}O)){*}\mathbb{1}_{1\times d} - (O{\odot}O)){\odot}(\mathbb{1}_{d\times 1}{*}(\mathrm{sum}((O{\odot}O)^\top))^\top - (O{\odot}O))))) - \mathrm{sum}(\mathrm{sum}((O{\odot}(((O{\odot}O{\odot}O){*}O){\odot}R){\odot}(\mathbb{1}_{d\times 1}{*}(\mathrm{sum}((O{\odot}O)^\top))^\top - (O{\odot}O))))) - \mathrm{sum}(\mathrm{sum}((O{\odot}((O{*}(O{\odot}O{\odot}O)){\odot}R){\odot}(\mathrm{sum}((O{\odot}O)){*}\mathbb{1}_{1\times d} - (O{\odot}O))))) + \mathrm{sum}(\mathrm{sum}((O{\odot}(((O{\odot}O{\odot}O){*}(O{\odot}O{\odot}O)){\odot}R))))) - 14\,Z_7 \quad (155)$$
$$\gamma(E_{40}) = 21\,\mathrm{sum}(\mathrm{diag}((D{\odot}D){*}O{*}O{*}O){\odot}\mathrm{sum}((O{\odot}O)) - 2\,\mathrm{diag}((D{\odot}D){*}(O{\odot}O{\odot}O){*}O{*}O)) \quad (156)$$
$$\gamma(E_{41}) = 7\,\mathrm{sum}(\mathrm{diag}(O{*}O{*}O){\odot}\mathrm{sum}((O{\odot}O){*}(D{\odot}D)) - 2\,\mathrm{diag}((O{\odot}O{\odot}O){*}(D{\odot}D){*}O{*}O)) \quad (157)$$
$$\gamma(E_{42}) = 7\,(\mathrm{sum}(\mathrm{diag}(O{*}O{*}O){\odot}\mathrm{sum}(((O{\odot}O){*}(O{\odot}O)){\odot}R) - 2\,\mathrm{diag}((O{\odot}O{\odot}O){*}(O{\odot}O{\odot}O){*}O)) - 2\,\mathrm{sum}(\mathrm{diag}((O{\odot}O{\odot}O){*}O{*}O){\odot}\mathrm{sum}((O{\odot}O)) - \mathrm{diag}((O{\odot}O{\odot}O{\odot}O{\odot}O){*}O{*}O) - \mathrm{diag}((O{\odot}O{\odot}O){*}O{*}(O{\odot}O{\odot}O)))) - 28\,Z_7 \quad (158)$$
$$\gamma(E_{43}) = 14\,\mathrm{sum}(\mathrm{diag}(O{*}O{*}O){\odot}Z_1 - 2\,(\mathrm{diag}((O{\odot}O{\odot}O){*}O{*}O){\odot}\mathrm{sum}((O{\odot}O)) - \mathrm{diag}((O{\odot}O{\odot}O{\odot}O{\odot}O){*}O{*}O) - 0.5\,\mathrm{diag}((O{\odot}O{\odot}O){*}O{*}(O{\odot}O{\odot}O)))) \quad (159)$$
$$\gamma(E_{44}) = 56\,Z_7 \quad (160)$$
$$Z_8 \equiv (O{\odot}(((O{\odot}O{\odot}O){*}O){\odot}R)){\odot}R \quad (161)$$
$$Z_9 \equiv (O{\odot}((O{*}O){\odot}R)){\odot}R \quad (162)$$
$$Z_{10} \equiv (O{\odot}((O{*}(O{\odot}O{\odot}O)){\odot}R)){\odot}R \quad (163)$$
$$Z_{11} \equiv ((O{*}O){\odot}R{\odot}Z_2 - (((O{\odot}O){*}Z_3){\odot}R - Z_8) - ((Z_9{*}(O{\odot}O)){\odot}R - Z_{10})) \quad (164)$$
$$\gamma(E_{45}) = 14\,(\mathrm{sum}(0.5\,\mathrm{sum}(Z_{11}){\odot}\mathrm{sum}((O{\odot}O))) - (1/7)\,\gamma(E_{38}) - \mathrm{sum}(\mathrm{sum}(((O{\odot}O)){\odot}Z_{11}))) \quad (165)$$

$$\gamma(E_{46}) = 21\,\mathrm{sum}(\mathrm{sum}(((O{\odot}O)){\odot}Z_{11})) \quad (166)$$
$$\gamma(E_{47}) = 7\,\mathrm{sum}(\mathrm{sum}(Z_{11}){\odot}\mathrm{diag}((D{\odot}D))) \quad (167)$$
$$\gamma(E_{48}) = 7\,\mathrm{tr}((D{\odot}D){*}O{*}D{*}O{*}D{*}O) \quad (168)$$
$$\gamma(E_{49}) = 14\,\mathrm{sum}(\mathrm{diag}(D{*}O{*}O{*}O){\odot}\mathrm{sum}((O{\odot}O){*}D) - 2\,\mathrm{diag}(D{*}(O{\odot}O{\odot}O){*}D{*}O)) \quad (169)$$
$$\gamma(E_{50}) = 14\,\mathrm{sum}(\mathrm{diag}(O{*}O{*}D{*}O){\odot}\mathrm{sum}((O{\odot}O){*}D) - \mathrm{diag}((O{\odot}O{\odot}O){*}D{*}O{*}D{*}O) - \mathrm{diag}((O{\odot}O{\odot}O){*}(D{\odot}D){*}O{*}O)) \quad (170)$$
$$\gamma(E_{51}) = 28\,\mathrm{sum}(\mathrm{diag}(D{*}O{*}D{*}O{*}O){\odot}\mathrm{sum}((O{\odot}O)) - \mathrm{diag}(D{*}(O{\odot}O{\odot}O){*}D{*}O{*}O) - \mathrm{diag}(D{*}(O{\odot}O{\odot}O){*}O{*}D{*}O)) \quad (171)$$
$$\gamma(E_{52}) = 7\,\mathrm{sum}(\mathrm{diag}(O{*}D{*}O{*}D{*}O){\odot}\mathrm{sum}((O{\odot}O)) - 2\,\mathrm{diag}((O{\odot}O{\odot}O){*}D{*}O{*}D{*}O)) \quad (172)$$
$$\gamma(E_{53}) = 14\,\mathrm{sum}((\mathrm{sum}(((((O{*}O){\odot}R){\odot}((O{*}D{*}O){\odot}R)) - ((O{\odot}O){*}D{*}(O{\odot}O)){\odot}R))){\odot}\mathrm{diag}((D{\odot}D))) \quad (173)$$
$$\gamma(E_{54}) = 7\,\mathrm{sum}(\mathrm{sum}(((((O{*}D{*}O){\odot}R){\odot}((O{*}(D{\odot}D){*}O){\odot}R)) - ((O{\odot}O){*}(D{\odot}D{\odot}D){*}(O{\odot}O)){\odot}R))) \quad (174)$$
$$Z_{12} \equiv \mathrm{sum}(0.5\,\mathrm{sum}(((((O{*}O){\odot}R){\odot}((O{*}O){\odot}R)) - (((O{\odot}O){*}(O{\odot}O)){\odot}R)){\odot}((O{\odot}O){*}D))) \quad (175)$$
$$Z_{13} \equiv \mathrm{sum}(\mathrm{sum}((((((O{\odot}O{\odot}O){*}D{*}O){\odot}R){\odot}((O{*}O){\odot}R)) - ((O{\odot}O{\odot}O{\odot}O){*}D{*}(O{\odot}O)){\odot}R))) \quad (176)$$
$$Z_{14} \equiv 0.5\,\mathrm{sum}(\mathrm{sum}(((((O{*}D{*}O){\odot}R){\odot}((O{*}O){\odot}R)) - (((O{\odot}O){*}D{*}(O{\odot}O)){\odot}R)){\odot}((O{\odot}O)))) \quad (177)$$
$$Z_{15} \equiv \mathrm{sum}(\mathrm{sum}((((((O{\odot}O{\odot}O){*}O){\odot}R){\odot}((O{*}D{*}O){\odot}R)) - ((O{\odot}O{\odot}O{\odot}O){*}D{*}(O{\odot}O)){\odot}R))) \quad (178)$$
$$\gamma(E_{55}) = 14\,(\mathrm{sum}(0.5\,(\mathrm{sum}(((((O{*}O){\odot}R){\odot}(((O{*}O)){\odot}R)) - ((O{\odot}O){*}(O{\odot}O)){\odot}R))){\odot}\mathrm{sum}((O{\odot}O){*}D)) - Z_{13} - Z_{12}) \quad (179)$$
$$\gamma(E_{56}) = 28\,(\mathrm{sum}(0.5\,(\mathrm{sum}(((((O{*}O){\odot}R){\odot}(((O{*}O)){\odot}R)) - ((O{\odot}O){*}(O{\odot}O)){\odot}R))){\odot}\mathrm{sum}(D{*}(O{\odot}O))) - Z_{13} - Z_{12}) \quad (180)$$
$$\gamma(E_{57}) = 14\,(\mathrm{sum}((\mathrm{sum}(((((O{*}D{*}O){\odot}R){\odot}(((O{*}O)){\odot}R)) - ((O{\odot}O){*}D{*}(O{\odot}O)){\odot}R))){\odot}\mathrm{sum}((O{\odot}O))) - Z_{13} - Z_{15} - 2\,Z_{14}) \quad (181)$$
$$\gamma(E_{58}) = 14\,(\mathrm{sum}(0.5\,\mathrm{sum}((((((O{*}O){\odot}R){\odot}((O{*}O){\odot}R)) - (((O{\odot}O){*}(O{\odot}O)){\odot}R)){*}D)){\odot}\mathrm{sum}((O{\odot}O))) - Z_{15} - Z_{12}) \quad (182)$$
$$\gamma(E_{59}) = 84\,Z_{12} \quad (183)$$
$$\gamma(E_{60}) = 42\,Z_{14} \quad (184)$$
$$Z_{25} = \mathrm{tr}(M{*}M{*}M{*}M{*}M{*}M{*}M{*}M) - \mathrm{sum}(\gamma(E_1) : \gamma(E_{60})) \quad (185)$$
$$Z_{26} = \mathrm{tr}(O{*}O{*}O{*}O{*}O{*}O{*}O{*}O) - \gamma(E_{13}) - \gamma(E_{14}) - \gamma(E_{24}) - \gamma(E_{25}) - \gamma(E_{28}) - \gamma(E_{38}) - \gamma(E_{39}) - \mathrm{sum}(\gamma(E_{42}) : \gamma(E_{46})) \quad (186)$$
$$Z_{16} \equiv (1/6)\,((O{*}O{\odot}R){\odot}(O{*}O{\odot}R){\odot}(O{*}O{\odot}R) - ((O{\odot}O{\odot}O){*}(O{\odot}O{\odot}O){\odot}R) - 3\,(((O{\odot}O){*}(O{\odot}O){\odot}R){\odot}(O{*}O{\odot}R) - ((O{\odot}O{\odot}O){*}(O{\odot}O{\odot}O){\odot}R))) \quad (187)$$

$$\gamma(E_{61}) = 42\,\mathrm{sum}(\mathrm{sum}(Z_{16} \odot O)) \quad (188)$$

$$Z_{17} \equiv \mathrm{sum}(\mathrm{sum}(0.5\ ((O*O\odot R)\odot(O*O\odot R) - ((O\odot O)*(O\odot O)\odot R)))\odot(0.5\ \mathrm{diag}(O*O*O))) \quad (189)$$

$$\gamma(E_{62}) = 28\,(Z_{17} - (6/84)\,\gamma(E_{61}) - (2/42)\,\gamma(E_{46}) - (3/56)\,\gamma(E_{44}) \quad (190)$$

$$\gamma(E_{63}) = Z_{26} - \gamma(E_{61}) - \gamma(E_{62}) \quad (191)$$

$$\gamma(E_{64}) = 7\,\mathrm{sum}(\mathrm{sum}((D*O*D*O*D\odot R)\odot(O*O\odot R))) - 7\,\mathrm{sum}(\mathrm{sum}(D*(O\odot O)*D*(O\odot O)*D\odot R)) \quad (192)$$

$$\gamma(E_{65}) = 7\,\mathrm{sum}(\mathrm{sum}(D*Z_{11}*D)) \quad (193)$$

$$Z_{18} \equiv \mathrm{sum}(((O*O)\odot R\odot(O*O) - ((O\odot O)*(O\odot O))\odot R)\odot((O\odot O))) \quad (194)$$

$$\gamma(E_{66}) = 7\,\mathrm{sum}(((\mathrm{diag}(O*O*O)\odot\mathrm{diag}(O*O*O)) - 2\,\mathrm{diag}((O\odot O)*(O\odot O)*(O\odot O)) - 4\,Z_{18})\odot\mathrm{diag}(D)) \quad (195)$$

$$Z_{20} \equiv 0.5\,\mathrm{sum}(\mathrm{sum}(((O*O\odot R)\odot(O*D*O\odot R) - ((O\odot O)*D*(O\odot O)))\odot(O\odot O))) \quad (196)$$

$$\gamma(E_{67}) = 14\,(\mathrm{sum}(\mathrm{diag}(O*O*O)\odot\mathrm{diag}(O*O*D*O) - 2\,\mathrm{diag}((O\odot O)*(O\odot O)*D*(O\odot O))) - 2\,\mathrm{sum}(Z_{18}\odot\mathrm{diag}(D)) - 4\,Z_{20}) \quad (197)$$

$$Z_{21} \equiv (((O*D*O*D)\odot R)*O - O\odot(\mathbb{1}_{d\times 1}*\mathrm{sum}(D*(O\odot O)*D, 1) - D*(O\odot O)*D))\odot R \quad (198)$$

$$Z_{22} \equiv (O\odot((O*D*O)\odot R))\odot R \quad (199)$$

$$Z_{23} \equiv (O\odot((D*(O\odot O\odot O)*D*O)\odot R))\odot R \quad (200)$$

$$Z_{24} \equiv (O\odot((O*D*(O\odot O\odot O)*D)\odot R))\odot R \quad (201)$$

$$\gamma(E_{68}) = 7\,\mathrm{sum}(\mathrm{sum}(((O*O)\odot R\odot Z_{21} - (((O\odot O)*D*Z_{22})\odot R - Z_{23}) - ((Z_{22}*D*(O\odot O))\odot R - Z_{24})))) \quad (202)$$

$$\gamma(E_{69}) = Z_{25} - Z_{26} - \mathrm{sum}(\gamma(E_{64}) : \gamma(E_{68})) \quad (203)$$

References

Dimitris Achlioptas and Frank McSherry. Fast computation of low rank matrix approximations. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 611–618. ACM, 2001.

N. Alon, R. Yuster, and U. Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997.

A. Andoni, H. L. Nguyên, Y. Polyanskiy, and Y. Wu. Tight lower bound for linear sketches of moments. In *International Colloquium on Automata, Languages, and Programming*, pages 25–32. Springer, 2013.

E. Aune, D. P. Simpson, and J. Eidsvik. Parameter estimation in high dimensional gaussian distributions. *Statistics and Computing*, 24(2):247–263, 2014.

H. Avron and S. Toledo. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *Journal of the ACM (JACM)*, 58(2):8, 2011.

S. Bhojanapalli and P. Jain. Universal matrix completion. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1881–1889, 2014.

C. Boutsidis, P. Drineas, P. Kambadur, E.-M. Kontopoulou, and A. Zouzias. A randomized algorithm for approximating the log determinant of a symmetric positive definite matrix. *arXiv preprint arXiv:1503.00374*, 2015.

E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

R. Carbó-Dorca. Smooth function topological structure descriptors based on graph-spectra. *Journal of Mathematical Chemistry*, 44(2):373–378, 2008.

J. Chen. How accurately should i compute implicit matrix-vector products when applying the hutchinson trace estimator? *SIAM Journal on Scientific Computing*, 38(6):A3515–A3539, 2016.

J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM, 2007.

E. Di Napoli, E. Polizzi, and Y. Saad. Efficient estimation of eigenvalue counts in an interval. *Numerical Linear Algebra with Applications*, 2016.

Khaled M Elbassioni. A polynomial delay algorithm for generating connected induced subgraphs of a given cardinality. *J. Graph Algorithms Appl.*, 19(1):273–280, 2015.

E. R. Elenberg, K. Shanmugam, M. Borokhovich, and A. G. Dimakis. Beyond triangles: A distributed framework for estimating 3-profiles of large graphs. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 229–238. ACM, 2015.

E. R. Elenberg, K. Shanmugam, M. Borokhovich, and A. G. Dimakis. Distributed estimation of graph 4-profiles. In *Proceedings of the 25th International Conference on World Wide Web*, pages 483–493. International World Wide Web Conferences Steering Committee, 2016.

E. Estrada. Characterization of 3d molecular structure. *Chemical Physics Letters*, 319(5): 713–718, 2000.

E. Estrada and N. Hatano. Statistical-mechanical approach to subgraph centrality in complex networks. *Chemical Physics Letters*, 439(1):247–251, 2007.

E. Estrada and J. A. Rodriguez-Velázquez. Spectral measures of bipartivity in complex networks. *Physical Review E*, 72(4):046105, 2005.

U. Feige and E. Ofek. Spectral techniques applied to sparse random graphs. *Random Struct. Algorithms*, 27(2):251–275, 2005.

J. Flum and M. Grohe. The parameterized complexity of counting problems. *SIAM Journal on Computing*, 33(4):892–922, 2004.

J. Friedman, J. Kahn, and E. Szemerédi. On the second eigenvalue in random regular graphs. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, pages 587–598, Seattle, Washington, USA, may 1989. ACM.

Friedrich Götze and Alexandre Tikhomirov. Rate of convergence to the semi-circular law. *Probability Theory and Related Fields*, 127(2):228–276, 2003.

T. R. Halford and K. M. Chugg. An algorithm for counting short cycles in bipartite graphs. *IEEE Transactions on Information Theory*, 52(1):287–292, 2006.

Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

I. Han, D. Malioutov, and J. Shin. Large-scale log-determinant computation through stochastic chebyshev expansions. In *ICML*, pages 908–917, 2015.

I. Han, D. Malioutov, H. Avron, and J. Shin. Approximating the spectral sums of large-scale matrices using chebyshev approximations. *arXiv preprint arXiv:1606.00942*, 2016.

Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 19(2):433–450, 1990.

Yuri Ingster and Irina A Suslina. *Nonparametric goodness-of-fit testing under Gaussian models*, volume 169. Springer Science & Business Media, 2012.

P. Jain, P. Netrapalli, and S. Sanghavi. Low-rank matrix completion using alternating minimization. In *STOC*, pages 665–674, 2013.

M. Karimi and A. H. Banihashemi. Message-passing algorithms for counting short cycles in a graph. *IEEE Transactions on Communications*, 61(2):485–495, 2013.

R. H. Keshavan and S. Oh. A gradient descent algorithm on the Grassman manifold for matrix completion. *arXiv preprint arXiv:0910.5260*, 2009.

R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from a few entries. *Information Theory, IEEE Transactions on*, 56(6):2980–2998, 2010a.

R. H Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *Journal of Machine Learning Research*, 11(2057-2078):1, 2010b.

T. Kloks, D. Kratsch, and H. Müller. Finding and counting small induced subgraphs efficiently. *Information Processing Letters*, 74(3):115–121, 2000.

W. Kong and G. Valiant. Spectrum estimation from samples. *arXiv preprint arXiv:1602.00061*, 2016.

C. M. Le, E. Levina, and R. Vershynin. Sparse random graphs: regularization and concentration of the laplacian. *arXiv preprint arXiv:1502.03049*, 2015.

Y. Li and D. P. Woodruff. On approximating functions of the singular values in a stream. *arXiv preprint arXiv:1604.08679*, 2016.

Y. Li, H. L. Nguyên, and D. P. Woodruff. On sketching matrix norms and the top singular vector. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1562–1581. Society for Industrial and Applied Mathematics, 2014.

H. Liu and J. Wang. A new way to enumerate cycles in graph. In *AICT/ICIW*, page 57, 2006.

Michael W Mahoney et al. Randomized algorithms for matrices and data. *Foundations and Trends® in Machine Learning*, 3(2):123–224, 2011.

J. C. Mason and D. C. Handscomb. *Chebyshev polynomials*. CRC Press, 2002.

S. Negahban and M. J. Wainwright. Restricted strong convexity and (weighted) matrix completion: Optimal bounds with noise. *Journal of Machine Learning Research*, 2012. To appear; posted at http://arxiv.org/abs/1009.2118.

Praneeth Netrapalli, UN Niranjan, Sujay Sanghavi, Animashree Anandkumar, and Prateek Jain. Non-convex robust pca. In *Advances in Neural Information Processing Systems*, pages 1107–1115, 2014.

R. K. Pace and J. P. LeSage. Chebyshev approximation of log-determinants of spatial weight matrices. *Computational Statistics & Data Analysis*, 45(2):179–196, 2004.

Eric Polizzi. Density-matrix-based algorithm for solving eigenvalue problems. *Physical Review B*, 79(11):115112, 2009.

F. Roosta-Khorasani and U. Ascher. Improved bounds on sample size for implicit matrix trace estimators. *Foundations of Computational Mathematics*, 15(5):1187–1212, 2015.

H. Rue and L. Held. *Gaussian Markov random fields: theory and applications*. CRC Press, 2005.

A. Saade, F. Krzakala, and L. Zdeborová. Matrix completion from fewer entries: Spectral detectability and rank estimation. In *Advances in Neural Information Processing Systems*, pages 1261–1269, 2015.

Tetsuya Sakurai and Hiroshi Sugiura. A projection method for generalized eigenvalue problems using numerical integration. *Journal of computational and applied mathematics*, 159 (1):119–128, 2003.

G. Schofield, J. R. Chelikowsky, and Y. Saad. A spectrum slicing method for the kohn–sham problem. *Computer Physics Communications*, 183(3):497–505, 2012.

W. Schudy and M. Sviridenko. Bernstein-like concentration and moment inequalities for polynomials of independent random variables: multilinear case. *arXiv preprint arXiv:1109.5193*, 2011.

S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of escherichia coli. *Nature genetics*, 31(1):64–68, 2002.

A. Stathopoulos, J. Laeuchli, and K. Orginos. Hierarchical probing for estimating the trace of the matrix inverse on toroidal lattices. *SIAM Journal on Scientific Computing*, 35(5): S299–S322, 2013.

T. Tian, C. R. Jones, J. D. Villasenor, and R. D. Wesel. Selective avoidance of cycles in irregular ldpc code construction. *IEEE Transactions on Communications*, 52(8):1242–1247, 2004.

Ryuhei Uehara et al. The number of connected components in graphs and its applications. *Manuscript. URL: http://citeseerx. ist. psu. edu/viewdoc/summary*, 1999.

J. Ugander, L. Backstrom, and J. Kleinberg. Subgraph frequencies: Mapping the empirical and extremal geography of large graph collections. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1307–1318. ACM, 2013.

S. Van Aelst and P. Rousseeuw. Minimum volume ellipsoid. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(1):71–82, 2009.

P. Wang, J. Lui, B. Ribeiro, D. Towsley, J. Zhao, and X. Guan. Efficiently estimating motif statistics of large networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 9(2):8, 2014.

E. P. Wigner. Characteristic vectors of bordered matrices with infinite dimensions. *Annals of Mathematics*, page 548564, 1955.

L. Wu, J. Laeuchli, V. Kalantzis, A. Stathopoulos, and E. Gallopoulos. Estimating the trace of the matrix inverse by interpolating from the diagonal of an approximate inverse. *Journal of Computational Physics*, 326:828–844, 2016.

Y. Zhang and W. E. Leithead. Approximate implementation of the logarithm of the matrix determinant in gaussian process regression. *Journal of Statistical Computation and Simulation*, 77(4):329–348, 2007.

Y. Zhang, M. J. Wainwright, and M. I. Jordan. Distributed estimation of generalized matrix rank: Efficient algorithms and lower bounds. *arXiv preprint arXiv:1502.01403*, 2015.

Journal of Machine Learning Research 20 (2019) 1-36 Submitted 1/18; Revised 11/18; Published 2/19

Random Feature-based Online Multi-kernel Learning in Environments with Unknown Dynamics

Yanning Shen SHENX513@UMN.EDU
Tianyi Chen CHEN3827@UMN.EDU
Georgios B. Giannakis GEORGIOS@UMN.EDU
Department of Electrical and Computer Engineering, University of Minnesota
Minneapolis, MN, 55455, USA

Editor: Karsten Borgwardt

Abstract

Kernel-based methods exhibit well-documented performance in various nonlinear learning tasks. Most of them rely on a preselected kernel, whose prudent choice presumes task-specific prior information. Especially when the latter is not available, multi-kernel learning has gained popularity thanks to its flexibility in choosing kernels from a prescribed kernel dictionary. Leveraging the random feature approximation and its recent orthogonality-promoting variant, the present contribution develops a scalable multi-kernel learning scheme (termed Raker) to obtain the sought nonlinear learning function 'on the fly,' first for static environments. To further boost performance in dynamic environments, an adaptive multi-kernel learning scheme (termed AdaRaker) is developed. AdaRaker accounts not only for data-driven learning of kernel combination, but also for the unknown dynamics. Performance is analyzed in terms of both static and dynamic regrets. AdaRaker is uniquely capable of tracking nonlinear learning functions in environments with unknown dynamics, and with with analytic performance guarantees. Tests with synthetic and real datasets are carried out to showcase the effectiveness of the novel algorithms.[1]

Keywords: Online learning, reproducing kernel Hilbert space, multi-kernel learning, random features, dynamic and adversarial environments.

1. Introduction

Function approximation emerges in various learning tasks such as regression, classification, clustering, dimensionality reduction, as well as reinforcement learning (Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004; Dai et al., 2017). Among them, the emphasis here is placed on supervised functional learning tasks: given samples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)\}_{t=1}^T$ with $\mathbf{x}_t \in \mathbb{R}^d$ and $y_t \in \mathbb{R}$, the goal is to find a function $f(\cdot)$ such that the discrepancy between each pair of y_t and $f(\mathbf{x}_t)$ is minimized. Typically, such discrepancy is measured by a cost function $\mathcal{C}(f(\mathbf{x}_t), y_t)$, which requires to find $f(\cdot)$ minimizing $\sum_{t=1}^T \mathcal{C}(f(\mathbf{x}_t), y_t)$. While this goal is too ambitious to achieve in general, the problem becomes tractable when $f(\cdot)$ is assumed to belong to a reproducing kernel Hilbert space (RKHS) induced by a kernel (Schölkopf and Smola, 2002). Comparable to deep neural networks, functions defined in RKHS can model highly nonlinear relationship, and thus kernel-based methods have well-documented merits for principled function approximation. Despite their popularity, most kernel methods rely on a single pre-selected kernel. Yet, multi-kernel learning (MKL) is more powerful, thanks to its data-driven kernel selection from a given dictionary; see e.g., (Shawe-Taylor and Cristianini, 2004; Rakotomamonjy et al., 2008; Cortes et al., 2009; Gönen and Alpaydın, 2011), and (Bazerque and Giannakis, 2013).

In addition to the attractive representation power that can be afforded by kernel methods, several learning tasks are also expected to be performed in an online fashion. Such a need naturally arises when the data arrive sequentially, such as those in online spam detection (Ma et al., 2009), and time series prediction (Richard et al., 2009); or, when the sheer volume of data makes it impossible to carry out data analytics in batch form (Kivinen et al., 2004). This motivates well online kernel-based learning methods that inherit the merits of their batch counterparts, while at the same time allowing efficient online implementation. Taking a step further, the optimal function may itself change over time in environments with *unknown dynamics*. This is the case when the function of interest e.g., represents the state in brain graphs, or, captures the temporal processes propagating over time-varying networks. Especially when variations are due to adversarial interventions, the underlying dynamics are unknown. Online kernel-based learning in such environments remains a largely uncharted territory (Kivinen et al., 2004; Hoi et al., 2013).

In accordance with these needs and desiderata, the *goal* of this paper is an algorithmic pursuit of scalable online MKL in environments with unknown dynamics, along with their associated performance guarantees. Major challenges come from two sources: i) the well-known "curse" of dimensionality in kernel-based learning; and, ii) the defiance of tracking unknown time-varying functions without future information. Regarding i), the representer theorem renders the size of kernel matrices to grow quadratically with the number of data (Wahba, 1990), thus the computational complexity to find even a *single* kernel-based predictor is cubic. Furthermore, storage of past data causes memory overflow in large-scale learning tasks such as those emerging in e.g., topology identification of social and brain networks (Shen et al., 2016, 2017; Shen and Giannakis, 2018), which makes kernel-based methods less scalable relative to their linear counterparts. For ii), most online learning settings presume time invariance or slow dynamics, where an algorithm achieving sub-linear regret incurs on average "no-regret" relative to the *best static* benchmark. Clearly, designing online schemes that are comparable to the *best dynamic* solution is appealing though formidably challenging without knowledge of the dynamics (Kivinen et al., 2004).

1.1. Related works

To put our work in context, we review prior art from the following two aspects.

Batch kernel methods. Kernel methods are known to suffer from the growing dimensionality in large-scale learning tasks (Shawe-Taylor and Cristianini, 2004). Major efforts have been devoted to scaling up kernel methods in batch settings. Those include approaches to approximating the kernel matrix using low-rank factorizations (Williams and Seeger, 2001; Sheikholeslami et al., 2018), whose performance was analyzed in (Cortes et al., 2010). Recently, random feature (RF) based function estimators have gained popularity since the work of (Rahimi and Recht, 2007) and (Dai et al., 2014), whose variance has been con-

1. Preliminary results in this paper were presented in part at the 2018 International Conference on Artificial Intelligence and Statistics (Shen et al., 2018).

©2019 Yanning Shen and Tianyi Chen and Georgios B. Giannakis.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-030.html.

siderably reduced through an *orthogonality promoting* RF modification (Yu et al., 2016). These approaches assume that the kernel is known, a choice crucially dependent on domain knowledge. Enabling kernel selection, several MKL-based approaches have emerged, see e.g., (Lanckriet et al., 2004; Rakotomamonjy et al., 2008; Bach, 2008; Cortes et al., 2009; Gönen and Alpaydm, 2011), and their performance gain has been documented relative to their single kernel counterparts. However, the aforementioned methods are designed for batch settings, and are either intractable or become less efficient in online setups. When the sought functions vary over time and especially when the dynamics are unknown (as in adversarial settings), batch schemes fall short in tracking the optimal function estimators.

Online (multi-)kernel learning. Tailored for streaming large-scale datasets, online kernel-based learning methods have gained due popularity. To deal with the growing complexity of online kernel learning, successful attempts have been made to design *budgeted* kernel learning algorithms, including techniques such as support vector removal (Kivinen et al., 2004; Dekel et al., 2008), and support vector merging (Wang et al., 2012). Maintaining an affordable budget, online multi-kernel learning (OMKL) methods have been reported for online classification (Jin et al., 2010; Hoi et al., 2013; Sahoo et al., 2016), and regression (Sahoo et al., 2014; Lu et al., 2018). Devoid of the need for budget maintenance, online kernel-based learning algorithms based on RF approximation (Rahimi and Recht, 2007) have been developed in (Lu et al., 2016; Bouboulis et al., 2018; Ding et al., 2017), but only with a single pre-selected kernel. More importantly, existing kernel-based learning approaches implicitly presume a *static* environment, where the benchmark is provided through the best static function (a.k.a. static regret) (Shalev-Shwartz, 2011). However, static regret is not a comprehensive metric for dynamic settings, where the optimal kernel also varies over time and the dynamics are generally unknown as with adversarial settings.

1.2. Our contributions

The present paper develops an adaptive online MKL algorithm, capable of learning a nonlinear function from sequentially arriving data samples. Relative to prior art, our contributions can be summarized as follows.

c1) For the first time, RFs are employed for scalable online MKL tackled by a weighted combination of advices from an ensemble of experts - an innovative cross-fertilization of online learning to MKL. Performance of the resultant algorithm (abbreviated as **Raker**) is benchmarked by the best time-invariant function approximant via static regret analysis.

c2) A novel adaptive approach (termed **AdaRaker**) is introduced for scalable online MKL in environments with unknown dynamics. AdaRaker is a hierarchical ensemble learner with scalable RF-based modules that provably yields sub-linear dynamic regret, so long as the accumulated variation grows sub-linearly with time.

c3) The novel algorithms are compared with competing alternatives for online nonlinear regression on both synthetic and real datasets. The tests corroborate that Raker and AdaRaker exhibit attractive performance in both accuracy and scalability.

Outline. Section 2 presents preliminaries, and states the problem. Section 3 develops the Raker for online MKL in static environments, and Section 4 develops its adaptive version for online MKL in environments with unknown dynamics. Section 5 reports numerical tests with both synthetic and real datasets, while conclusions are drawn in Section 6.

Notation. Bold uppercase (lowercase) letters will denote matrices (column vectors), while $(\cdot)^\top$ stands for vector and matrix transposition, and $\|\mathbf{x}\|$ denotes the ℓ_2-norm of a vector $\mathbf{x}$. Inequalities for vectors $\mathbf{x} > \mathbf{0}$, and the projection operator $[\mathbf{a}]^+ := \max\{\mathbf{a}, \mathbf{0}\}$ are defined entrywise. Symbol † represents the Hermitian operator, while the indicator function $\mathbb{1}_{\{A\}}$ takes value 1 when the event A happens, and 0 otherwise. $\mathbb{E}$ denotes the expectation, while $\langle\cdot,\cdot\rangle$ and $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ the vector inner product in Euclidian and Hilbert space respectively.

2. Preliminaries and Problem Statement

This section reviews briefly basics of kernel-based learning, to introduce notation and the needed background for our novel online MKL schemes.

Given samples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)\}_{t=1}^T$ with $\mathbf{x}_t \in \mathbb{R}^d$ and $y_t \in \mathbb{R}$, the function approximation task is to find a function $f(\cdot)$ such that $y_t = f(\mathbf{x}_t) + e_t$, where e_t denotes an error term representing noise or un-modeled dynamics. It is supposed that $f(\cdot)$ belongs to a reproducing kernel Hilbert space (RKHS), namely $\mathcal{H} := \{f | f(\mathbf{x}) = \sum_{t=1}^\infty \alpha_t \kappa(\mathbf{x}, \mathbf{x}_t)\}$, where $\kappa(\mathbf{x}, \mathbf{x}_t) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a symmetric positive semidefinite basis (so-termed kernel) function, which measures the similarity between $\mathbf{x}$ and $\mathbf{x}_t$. Among the choices of κ specifying different bases, a popular one is the Gaussian given by $\kappa(\mathbf{x}, \mathbf{x}_t) := \exp[-\|\mathbf{x}-\mathbf{x}_t\|^2/(2\sigma^2)]$. A kernel is reproducing if it satisfies $\langle \kappa(\mathbf{x}, \mathbf{x}_t), \kappa(\mathbf{x}, \mathbf{x}_{t'})\rangle_{\mathcal{H}} = \kappa(\mathbf{x}_t, \mathbf{x}_{t'})$, which in turn induces the RKHS norm $\|f\|_{\mathcal{H}}^2 := \sum_t \sum_{t'} \alpha_t \alpha_{t'} \kappa(\mathbf{x}_t, \mathbf{x}_{t'})$. Consider the optimization problem

$$\min_{f \in \mathcal{H}} \ \frac{1}{T} \sum_{t=1}^T \mathcal{C}(f(\mathbf{x}_t), y_t) + \lambda \Omega\left(\|f\|_{\mathcal{H}}^2\right) \tag{1}$$

where depending on the application, the cost function $\mathcal{C}(\cdot,\cdot)$ can be selected to be, e.g., the least-squares (LS), the logistic or the hinge loss; $\Omega(\cdot)$ is an increasing function; and, $\lambda > 0$ is a regularization parameter that controls overfitting. According to the representer theorem, the optimal solution of (1) admits the finite-dimensional form, given by (Wahba, 1990)

$$\hat{f}(\mathbf{x}) = \sum_{t=1}^T \alpha_t \kappa(\mathbf{x}, \mathbf{x}_t) := \boldsymbol{\alpha}^\top \mathbf{k}(\mathbf{x}) \tag{2}$$

where $\boldsymbol{\alpha} := [\alpha_1, \ldots, \alpha_T]^\top \in \mathbb{R}^T$ collects the combination coefficients, and the $T \times 1$ kernel vector is $\mathbf{k}(\mathbf{x}) := [\kappa(\mathbf{x}, \mathbf{x}_1), \ldots, \kappa(\mathbf{x}, \mathbf{x}_T)]^\top$. Substituting (2) into the RKHS norm, we find $\|f\|_{\mathcal{H}}^2 := \sum_t \sum_{t'} \alpha_t \alpha_{t'} \kappa(\mathbf{x}_t, \mathbf{x}_{t'}) = \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha}$, where the $T \times T$ kernel matrix $\mathbf{K}$ has entries $[\mathbf{K}]_{t,t'} := \kappa(\mathbf{x}_t, \mathbf{x}_{t'})$; thus, the functional problem (1) boils down to a T-dimensional optimization over $\boldsymbol{\alpha}$, namely

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^T} \ \frac{1}{T} \sum_{t=1}^T \mathcal{C}(\boldsymbol{\alpha}^\top \mathbf{k}(\mathbf{x}_t), y_t) + \lambda \Omega\left(\boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha}\right) \tag{3}$$

where $\mathbf{k}^\top(\mathbf{x}_t)$ is the tth row of the matrix $\mathbf{K}$. While a scalar y_t is used here for brevity, coverage extends readily to vectors $\{\mathbf{y}_t\}$.

Note that (1) relies on: i) a known pre-selected kernel κ; and ii) having $\{\mathbf{x}_t, y_t\}_{t=1}^T$ available in batch form. A key observation here is that the dimension of the variable $\boldsymbol{\alpha}$ in

(3) grows with time T (or, the number of samples in the batch form), making it less scalable in online implementation. In the ensuing section, an online MKL method will be proposed to select κ as a superposition of multiple kernels, when the data become available online.

3. Online MKL in static environments

In this section, we develop an online learning approach that builds on the notion of **ra**ndom features (Rahimi and Recht, 2007; Yu et al., 2016), and leverages in a unique way multi-**ker**nel approximation – two tools justifying our acronym **Raker** used henceforth.

3.1. RF-based single kernel learning

To cope with the curse of dimensionality in optimizing (3), we will reformulate the functional optimization problem (1) as a parametric one with the dimension of optimization variables not growing with time. In this way, powerful toolboxes from convex optimization and online learning in vector spaces can be leveraged. We achieve this goal by judiciously using RFs. Although generalizations will follow, this subsection is devoted to RF-based single kernel learning, where basics of kernels, RFs, and online learning will be revisited.

As in (Rahimi and Recht, 2007), we will approximate κ in (2) using shift-invariant kernels that satisfy $\kappa(\mathbf{x}_t,\mathbf{x}_{t'})=\kappa(\mathbf{x}_t-\mathbf{x}_{t'})$. For $\kappa(\mathbf{x}_t-\mathbf{x}_{t'})$ absolutely integrable, its Fourier transform $\pi_\kappa(\mathbf{v})$ exists and represents the power spectral density, which upon normalizing to ensure $\kappa(\mathbf{0})=1$, can also be viewed as a probability density function (pdf); hence,

$$\kappa(\mathbf{x}_t-\mathbf{x}_{t'})=\int \pi_\kappa(\mathbf{v})e^{j\mathbf{v}^\top(\mathbf{x}_t-\mathbf{x}_{t'})}d\mathbf{v}:=\mathbb{E}_{\mathbf{v}}\left[e^{j\mathbf{v}^\top(\mathbf{x}_t-\mathbf{x}_{t'})}\right] \tag{4}$$

where the last equality is just the definition of the expected value. Drawing a sufficient number of D independent and identically distributed (i.i.d.) samples $\{\mathbf{v}_i\}_{i=1}^D$ from $\pi_\kappa(\mathbf{v})$, the ensemble mean in (4) can be approximated by the sample average

$$\hat{\kappa}_c(\mathbf{x}_t,\mathbf{x}_{t'}):=\frac{1}{D}\sum_{i=1}^{D}e^{j\mathbf{v}_i^\top(\mathbf{x}_t-\mathbf{x}_{t'})}:=\boldsymbol{\zeta}_{\mathbf{V}}^\dagger(\mathbf{x}_t)\boldsymbol{\zeta}_{\mathbf{V}}(\mathbf{x}_{t'}) \tag{5}$$

where $\mathbf{V}:=[\mathbf{v}_1,\ldots,\mathbf{v}_D]^\top\in\mathbb{R}^{D\times d}$, symbol $\dagger$ represents the Hermitian (conjugate-transpose) operator, and $\boldsymbol{\zeta}_{\mathbf{V}}(\mathbf{x})$ the complex RF vector

$$\boldsymbol{\zeta}_{\mathbf{V}}(\mathbf{x}):=\frac{1}{\sqrt{D}}\left[e^{j\mathbf{v}_1^\top\mathbf{x}},\ldots,e^{j\mathbf{v}_D^\top\mathbf{x}}\right]^\top. \tag{6}$$

Taking expected values on both sides of (5) and using (4) yields $\mathbb{E}_{\mathbf{v}}[\hat{\kappa}_c(\mathbf{x}_t,\mathbf{x}_{t'})]=\kappa(\mathbf{x}_t,\mathbf{x}_{t'})$, which means $\hat{\kappa}_c$ is unbiased. Likewise, $\hat{\kappa}_c$ can be shown consistent since $\mathbb{V}\text{ar}[\hat{\kappa}_c(\mathbf{x}_t,\mathbf{x}_{t'})]\propto D^{-1}$ vanishes as $D\to\infty$. Finding $\pi_\kappa(\mathbf{v})$ requires d-dimensional Fourier transform of κ, generally through numerical integration. For a number of popular kernels however, $\pi_\kappa(\mathbf{v})$ is available in closed form. Taking the Gaussian kernel as an example, where $\kappa_G(\mathbf{x}_t,\mathbf{x}_{t'})=\exp\left(\|\mathbf{x}_t-\mathbf{x}_{t'}\|_2^2/(2\sigma^2)\right)$, has Fourier transform corresponding to the pdf $\pi_G(\mathbf{v})=\mathcal{N}(0,\sigma^{-2}\mathbf{I})$.

Instead of the *complex* RFs $\{\boldsymbol{\zeta}_{\mathbf{V}}(\mathbf{x}_t)\}$ in (6) forming the linear kernel estimator $\hat{\kappa}_c$ in (5), one can consider its real part $\hat{\kappa}(\mathbf{x}_t,\mathbf{x}_{t'}):=\Re\{\hat{\kappa}_c(\mathbf{x}_t,\mathbf{x}_{t'})\}$ that is also an unbiased estimator

of κ. Defining the real RF vector $\mathbf{z}_{\mathbf{V}}(\mathbf{x}):=[\Re^\top\{\boldsymbol{\zeta}_{\mathbf{V}}(\mathbf{x}_t)\},\Im^\top\{\boldsymbol{\zeta}_{\mathbf{V}}(\mathbf{x}_t)\}]^\top$, this real kernel estimator becomes (cf. (5))

$$\hat{\kappa}(\mathbf{x}_t,\mathbf{x}_{t'})=\mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_t)\mathbf{z}_{\mathbf{V}}(\mathbf{x}_{t'}) \tag{7}$$

where the $2D\times 1$ *real* RF vector can be written as

$$\mathbf{z}_{\mathbf{V}}(\mathbf{x})=\frac{1}{\sqrt{D}}\left[\sin(\mathbf{v}_1^\top\mathbf{x}),\ldots,\sin(\mathbf{v}_D^\top\mathbf{x}),\cos(\mathbf{v}_1^\top\mathbf{x}),\ldots,\cos(\mathbf{v}_D^\top\mathbf{x})\right]^\top. \tag{8}$$

Hence, the nonlinear function that is optimal in the sense of (1) can be approximated by a linear one in the new $2D$-dimensional RF space, namely (cf. (2) and (7))

$$\hat{f}^{\text{RF}}(\mathbf{x})=\sum_{t=1}^{T}\alpha_t\mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_t)\mathbf{z}_{\mathbf{V}}(\mathbf{x}):=\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}) \tag{9}$$

where $\boldsymbol{\theta}^\top:=\sum_{\tau=1}^{T}\alpha_\tau\mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_\tau)$ is the new weight vector of size $2D$ whose dimension does not increase with number of data samples T.

While the solution $\hat{f}$ in (2) is the superposition of nonlinear functions κ, its RF approximant $\hat{f}^{\text{RF}}$ in (9) is a linear function of $\mathbf{z}_{\mathbf{V}}(\mathbf{x})$. As a result, the loss becomes

$$\mathcal{L}_t\big(f(\mathbf{x}_t)\big):=\mathcal{C}(f(\mathbf{x}_t),y_t)+\lambda\Omega\left(\|f\|_{\mathcal{H}}^2\right)=\mathcal{C}\left(\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t),y_t\right)+\lambda\Omega\left(\|\boldsymbol{\theta}\|^2\right) \tag{10}$$

where $\|\boldsymbol{\theta}\|^2:=\sum_t\sum_{t'}\alpha_t\alpha_{t'}\mathbf{z}_{\mathbf{V}}^\top(\mathbf{x}_t)\mathbf{z}_{\mathbf{V}}(\mathbf{x}_{t'}):=\|f\|_{\mathcal{H}}^2$; and the online learning task is

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^{2D}}\sum_{t=1}^{T}\mathcal{L}\left(\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t),y_t\right),\ \text{with}\ \mathcal{L}\big(\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t),y_t\big):=\mathcal{C}\big(\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t),y_t\big)+\lambda\Omega\big(\|\boldsymbol{\theta}\|^2\big). \tag{11}$$

Compared with the functional optimization in (1), the reformulated problem (11) is parametric, and more importantly it involves only optimization variables of fixed size $2D$. We can thus solve (11) using the online gradient descent iteration, e.g., (Hazan, 2016). Acquiring $\mathbf{x}_t$ per slot t, its RF $\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t)$ is formed as in (8), and $\boldsymbol{\theta}_{t+1}$ is updated online as

$$\boldsymbol{\theta}_{t+1}=\boldsymbol{\theta}_t-\eta_t\nabla\mathcal{L}(\boldsymbol{\theta}_t^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t),y_t) \tag{12}$$

where $\{\eta_t\}$ is the sequence of stepsizes that can tune learning rates, and $\nabla\mathcal{L}(\boldsymbol{\theta}_t^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t),y_t)$ the gradient at $\boldsymbol{\theta}=\boldsymbol{\theta}_t$. Iteration (12) provides *a functional update* since $\hat{f}_t^{\text{RF}}(\mathbf{x})=\boldsymbol{\theta}_t^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x})$, but the upshot of involving RFs is that this approximant is in the span of $\{\mathbf{z}_{\mathbf{V}}(\mathbf{x}),\forall\mathbf{x}\in\mathcal{X}\}$. Since $\mathbb{E}[\hat{\kappa}]=\kappa$, we find readily that $\mathbb{E}[\hat{f}^{\text{RF}}]=\hat{f}$; in words, unbiasedness of the kernel approximation ensures that the RF-based function approximant is also unbiased.

Variance-reduced RF. Besides unbiasedness, performance of the RF approximation is also influenced by the variance of RFs. Note that the variance of $\hat{\kappa}$ in (7) is of order $\mathcal{O}(D^{-1})$, but its scale can be reduced if $\mathbf{V}$ is formed to have orthogonal rows (Yu et al., 2016). Specifically for a Gaussian kernel with bandwidth σ^2, recall that $\mathbf{V}=\sigma^{-1}\mathbf{G}$ in (8), where each entry of $\mathbf{G}$ is drawn from $\mathcal{N}(0,1)$. For the variance-reduced orthogonal (O)RF

with $D = d$, one starts with Q-R factorization of $\mathbf{V} = \mathbf{QR}$, and uses the $d \times d$ factor $\mathbf{Q}$ along with a diagonal matrix $\mathbf{\Lambda}$, to form (Yu et al., 2016)

$$\mathbf{V}_{\text{ORF}} = \sigma^{-1}\,\mathbf{\Lambda Q} \tag{13}$$

where the diagonal entries of $\mathbf{\Lambda}$ are drawn i.i.d. from the χ distribution with d degrees of freedom, to ensure unbiasedness of the kernel approximant. For $D > d$, one selects $D = \nu d$ with $\nu > 1$ integer, and generates independently ν matrices each of size $d \times d$ as in (13). The final $\mathbf{V}_{\text{ORF}}$ is formed by concatenating these $d \times d$ sub-matrices. The upshot of ORF is that (Yu et al., 2016) $\mathbb{V}\text{ar}(\hat{\kappa}_{\text{ORF}}(\mathbf{x}_t, \mathbf{x}_{t'})) \leq \mathbb{V}\text{ar}(\hat{\kappa}(\mathbf{x}_t, \mathbf{x}_{t'}))$. As we have also confirmed via simulated tests, ORF-based function approximation can attain a prescribed accuracy with considerably less ORFs than what required by its RF-based counterpart.

The RF-based online single kernel learning scheme in this section presumes that κ is known a priori. Since this is not generally possible, it is prudent to adaptively select kernels by superimposing multiple kernel functions from a prescribed dictionary. This superposition will play a key role in the RF-based online MKL approach presented next.

3.2. Raker for online MKL

Specifying the kernel that "shapes" $\mathcal{H}$ is a critical choice for single kernel learning, since different kernels yield function estimates of variable accuracy. To deal with this, combinations of kernels from a prescribed and sufficiently rich dictionary $\{\kappa_p\}_{p=1}^P$ can be employed in (1). Each combination belongs to the convex hull $\bar{\mathcal{K}} := \{\bar{\kappa} = \sum_{p=1}^P \bar{\alpha}_p \kappa_p,\ \bar{\alpha}_p \geq 0,\ \sum_{p=1}^P \bar{\alpha}_p = 1\}$, and is itself a kernel (Schölkopf and Smola, 2002). With $\bar{\mathcal{H}}$ denoting the RKHS induced by $\bar{\kappa} \in \bar{\mathcal{K}}$, one then solves (1) with $\mathcal{H}$ replaced by $\bar{\mathcal{H}} := \mathcal{H}_1 \bigoplus \cdots \bigoplus \mathcal{H}_P$, where $\{\mathcal{H}_p\}_{p=1}^P$ represent the RKHSs corresponding to $\{\kappa_p\}_{p=1}^P$ (Micchelli and Pontil, 2005).

The candidate function $\bar{f} \in \bar{\mathcal{H}}$ is expressible in a separable form as $\bar{f}(\mathbf{x}) := \sum_{p=1}^P \bar{f}_p(\mathbf{x})$, where $\bar{f}_p(\mathbf{x})$ belongs to $\mathcal{H}_p$, for $p \in \mathcal{P} := \{1, \ldots, P\}$. To add flexibility per kernel in our ensuing online MKL scheme, we let wlog $\{\bar{f}_p = w_p f_p\}_{p=1}^P$, and seek functions of the form

$$f(\mathbf{x}) := \sum_{p=1}^P \bar{w}_p f_p(\mathbf{x}) \in \bar{\mathcal{H}} \tag{14}$$

where $f := \bar{f} / \sum_{p=1}^P w_p$, and the normalized weights $\{\bar{w}_p := w_p / \sum_{p=1}^P w_p\}_{p=1}^P$ satisfy $\bar{w}_p \geq 0$, and $\sum_{p=1}^P \bar{w}_p = 1$. Plugging (14) into (1), MKL solves the nonconvex problem

$$\min_{\{\bar{w}_p\},\{f_p\}} \ \frac{1}{T}\sum_{t=1}^T \mathcal{C}\left(\sum_{p=1}^P \bar{w}_p f_p(\mathbf{x}_t), y_t\right) + \lambda\Omega\left(\left\|\sum_{p=1}^P \bar{w}_p f_p\right\|_{\bar{\mathcal{H}}}^2\right) \tag{15a}$$

$$\text{s. to} \quad \sum_{p=1}^P \bar{w}_p = 1,\ \bar{w}_p \geq 0,\ p \in \mathcal{P} \tag{15b}$$

$$f_p \in \mathcal{H}_p,\ p \in \mathcal{P}. \tag{15c}$$

If Ω is convex over f, then (15a) is biconvex, meaning it is convex wrt $\{f_p\}$ ($\{\bar{w}_p\}$) when $\{\bar{w}_p\}$ ($\{f_p\}$) is given. Leveraging biconvexity, existing batch MKL schemes solve (15) via

alternating minimization that is known not to scale well with P and T (Micchelli and Pontil, 2005; Cortes et al., 2009; Gönen and Alpaydın, 2011).

To deal with scalability, our novel approach will leverage for the first time (O)RFs in a uniquely principled MKL formulation to end up with an efficient online learning approach. To this end, we will minimize a cost that upper bounds that in (15a), namely

$$\min_{\{\bar{w}_p\},\{f_p\}} \ \frac{1}{T}\sum_{t=1}^T\sum_{p=1}^P \bar{w}_p \mathcal{C}\left(f_p(\mathbf{x}_t), y_t\right) + \lambda\sum_{p=1}^P \bar{w}_p\,\Omega\left(\|f_p\|_{\mathcal{H}_p}^2\right) \quad \text{s. to (15b) and (15c)} \tag{16}$$

where Jensen's inequality confirms that under (15b) the cost in (16) upper bounds that of (15a). A key advantage of (16) is that its objective is separable across kernel 'atoms.'

We will exploit this separability jointly with the RF-based function approximation per kernel, to formulate our scalable online MKL task as

$$\min_{\{\bar{w}_p\},\{\hat{f}_p^{\text{RF}}\}} \sum_{t=1}^T\sum_{p=1}^P \bar{w}_p\,\mathcal{L}_t\left(\hat{f}_p^{\text{RF}}(\mathbf{x}_t)\right) \quad \text{s. to (15b) and } \hat{f}_p^{\text{RF}} \in \left\{\hat{f}_p(\mathbf{x}) = \boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}_p}(\mathbf{x})\right\} \tag{17}$$

where we interchangeably use $\mathcal{L}_t(\hat{f}(\mathbf{x}_t))$ as defined in (10) and $\mathcal{L}\big(\boldsymbol{\theta}^\top \mathbf{z}_{\mathbf{V}}(\mathbf{x}_t), y_t\big)$ as in (11). We will efficiently solve (17) 'on-the-fly' using our Raker algorithm, and what more, we will provide analytical performance guarantees. Our iterative solution will update separately each $\hat{f}_p^{\text{RF}}$ as in Section 3.1 using the scalable (O)RF-based function approximation scheme. Given $\mathbf{x}_t$, an RF vector $\mathbf{z}_p(\mathbf{x}_t)$ will be generated per p from pdf $\pi_{\kappa_p}(\mathbf{v})$ (cf. (8)), where we let $\mathbf{z}_p(\mathbf{x}_t) := \mathbf{z}_{\mathbf{V}_p}(\mathbf{x}_t)$ for notational brevity. Hence, for each p and slot t, we have

$$\hat{f}_{p,t}^{\text{RF}}(\mathbf{x}_t) = \boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t) \tag{18}$$

and as in (12), $\boldsymbol{\theta}_{p,t}$ is updated via

$$\boldsymbol{\theta}_{p,t+1} = \boldsymbol{\theta}_{p,t} - \eta\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top \mathbf{z}_p(\mathbf{x}_t), y_t). \tag{19}$$

As far as solving for $\bar{w}_{p,t}$, since it resides on a probability simplex (15b), our idea is to employ a multiplicative update (a.k.a. exponentiated gradient descent), e.g., (Hazan, 2016). Specifically, the un-normalized weights are found first as

$$w_{p,t+1} = \arg\min_{w_p}\ \eta\,\mathcal{L}_t\left(\hat{f}_{p,t}^{\text{RF}}(\mathbf{x}_t)\right)(w_p - w_{p,t}) + \mathcal{D}_{\text{KL}}(w_p \| w_{p,t}) \tag{20}$$

where $\mathcal{D}_{\text{KL}}(w_p \| w_{p,t}) := w_p \log(w_p / w_{p,t})$ is the KL-divergence. It can be readily verified that (20) admits the following closed-form update

$$w_{p,t+1} = w_{p,t} \exp\left(-\eta\mathcal{L}_t\left(\hat{f}_{p,t}^{\text{RF}}(\mathbf{x}_t)\right)\right) \tag{21}$$

where $\eta \in (0, 1)$ is a chosen constant that controls the adaptation rate of $\{w_{p,t}\}$. Having found $\{w_{p,t}\}$ as in (21), the normalized weights in (14) are obtained as $\bar{w}_{p,t} := w_{p,t} / \sum_{p=1}^P w_{p,t}$. Update (21) is intuitively pleasing because when $\hat{f}_{p,t}^{\text{RF}}$ contributes a larger loss relative to other $\hat{f}_{p',t}^{\text{RF}}$ with $p' \neq p$ at slot t, the corresponding $w_{p,t+1}$ decreases more than the other

Algorithm 1 Raker for online MKL in static environments

Input: Kernels $\kappa_p,\ p=1,\ldots,P$, step size $\eta>0$, and number of random features D.
Initialization: $\boldsymbol{\theta}_1=\mathbf{0}$.
for $t=1,2,\ldots,T$ **do**
  Receive a streaming datum $\mathbf{x}_t$.
  Construct $\mathbf{z}_p(\mathbf{x}_t)$ via (8) using κ_p for $p=1,\ldots,P$.
  Predict $\hat{f}_t^{\mathrm{RF}}(\mathbf{x}_t):=\sum_{p=1}^P \bar{w}_{p,t}\hat{f}_{p,t}^{\mathrm{RF}}(\mathbf{x}_t)$ with $\hat{f}_{p,t}^{\mathrm{RF}}(\mathbf{x}_t)$ in (18).
  Observe loss function $\mathcal{L}_t$, incur $\mathcal{L}_t(\hat{f}_t^{\mathrm{RF}}(\mathbf{x}_t))$.
  for $p=1,\ldots,P$ **do**
    Obtain loss $\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)$ or $\mathcal{L}_t(\hat{f}_{p,t}^{\mathrm{RF}}(\mathbf{x}_t))$.
    Update $\boldsymbol{\theta}_{p,t+1}$ via (19).
    Update $w_{p,t+1}$ via (21).
  end for
end for

weights in the next time slot. In other words, a more accurate RF-based approximant tends to play more important role in predicting the upcoming data.

Remark 1. The update (21) resembles the online learning paradigm, a.k.a. *online prediction with* (weighted) *expert advices* (Vovk, 1995; Cesa-Bianchi and Lugosi, 2006). Building on but going beyond OMKL in (Sahoo et al., 2014), the idea here is to view MKL with *RF-based function approximants* as a weighted combination of advices from an ensemble of P function approximants (experts). Besides permeating benefits from online learning to MKL, what is distinct here relative to (Vovk, 1995; Cesa-Bianchi and Lugosi, 2006) is that each function approximant also performs online learning for self improvement (cf. (19)).

In summary, our Raker for static (or slow-varying) dynamics is listed as Algorithm 1.

Memory requirement and computational complexity. At the t-th iteration, our Raker in Algorithm 1 needs to store a real $2D$ RF vector, and its corresponding weight vector per κ_p. Hence, the memory required is of order $\mathcal{O}(dDP)$. Regarding computational overhead, the per-iteration complexity (e.g., calculating inner products) is again of order $\mathcal{O}(dDP)$. Compared with the complexity of $\mathcal{O}(tdP)$ for OMKL by (Sahoo et al., 2014), or, $\mathcal{O}(t^3P)$ when matrix inversion required for the batch MKL, e.g., (Bazerque and Giannakis, 2013), the Raker is clearly more scalable, as t grows. Even when OMKL is confined to a budget of B past samples, the corresponding complexity of $\mathcal{O}(dBP)$ is comparable to that of Raker. This speaks for Raker's merits, whose performance guarantees will be proved analytically, and also demonstrated by numerical tests to outperform budgeted schemes.

Application examples: Online MKL regression and classification. To appreciate the usefulness of RF-based online MKL, consider first nonlinear regression, where given samples $\{\mathbf{x}_t\in\mathbb{R}^d, y_t\in\mathbb{R}\}_{t=1}^T$, the goal is to find a nonlinear function $f\in\mathcal{H}$, such that $y_t=f(\mathbf{x}_t)+e_t$. The criterion is to minimize the regularized prediction error of y_t, typically using the LS loss $\mathcal{L}(f(\mathbf{x}_t),y_t):=[y_t-f(\mathbf{x}_t)]^2+\lambda\|f\|_{\mathcal{H}}^2$, whose gradient is (cf. (19))

$$\nabla\mathcal{L}\left(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t\right)=2(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t)-y_t)\mathbf{z}_p(\mathbf{x}_t)+2\lambda\boldsymbol{\theta}_{p,t}. \tag{22}$$

It is clear that the per iteration complexity of Raker is only related to the dimension of $\mathbf{z}_p(\mathbf{x}_t)$, and does not increase over time.

For nonlinear classification, consider kernel-based perceptron and kernel-based logistic regression, which aim at learning a nonlinear classifier that best approximates either y_t or the pdf of y_t conditioned on $\mathbf{x}_t$. With binary labels $\{\pm1\}$, the perceptron solves (1) with $\mathcal{L}(f(\mathbf{x}_t),y_t)=\max(0,1-y_tf(\mathbf{x}_t))+\lambda\|f\|_{\mathcal{H}}^2$, which equals zero if $y_t=f(\mathbf{x}_t)$, otherwise it equals 1. Raker's gradient in this case is (cf. (19))

$$\nabla\mathcal{L}\left(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t\right)=-2y_t\mathcal{C}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\mathbf{z}_p(\mathbf{x}_t)+2\lambda\boldsymbol{\theta}_{p,t}. \tag{23}$$

Accordingly, given $\mathbf{x}_t$, logistic regression postulates that $\Pr(y_t=1|\mathbf{x}_t)=1/(1+\exp(f(\mathbf{x}_t)))$. Here the gradient of Raker takes the form (cf. (19))

$$\nabla\mathcal{L}\left(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t\right)=\frac{2y_t\exp(y_t\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t))}{1+\exp(y_t\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t))}\mathbf{z}_p(\mathbf{x}_t)+2\lambda\boldsymbol{\theta}_{p,t}. \tag{24}$$

To compare alternatives on equal footing, the numerical tests in Section 5 will deal with kernel-based regression and classification.

3.3. Static regret analysis of Raker

To analyze the performance of Raker, we assume that the following conditions are satisfied.

(as1) *Per slot t, the loss function $\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t),y_t)$ in (11) is convex w.r.t. $\boldsymbol{\theta}$.*

(as2) *For $\boldsymbol{\theta}$ belonging to a bounded set $\boldsymbol{\Theta}$ with $\|\boldsymbol{\theta}\|\leq C_\theta$, the loss is bounded; that is, $\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t),y_t)\in[-1,1]$, and has bounded gradient, meaning, $\|\nabla\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t),y_t)\|\leq L$.*

(as3) *Kernels $\{\kappa_p\}_{p=1}^P$ are shift-invariant, standardized, and bounded, that is, $\kappa_p(\mathbf{x}_i,\mathbf{x}_j)\leq 1,\ \forall\mathbf{x}_i,\mathbf{x}_j$; and w.l.o.g. they also have bounded entries, meaning $\|\mathbf{x}\|\leq 1$.*

Convexity of the loss under (as1) is satisfied by the popular loss functions including the square loss and the hinge loss. As far as (as2), it ensures that the losses, and their gradients are bounded, meaning they are L-Lipschitz continuous. While boundedness of the losses commonly holds since $\|\boldsymbol{\theta}\|$ is bounded, Lipschitz continuity is also not restrictive. Considering kernel-based regression as an example, the gradient is $(\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t)-y_t)\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t)+\lambda\boldsymbol{\theta}$. Since the loss is bounded, e.g., $\|\boldsymbol{\theta}^\top\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t)-y_t\|\leq 1$, and the RF vector in (8) can be bounded as $\|\mathbf{z}_{\mathbf{V}}(\mathbf{x}_t)\|\leq 1$, the constant is $L:=1+\lambda C_\theta$ using the Cauchy-Schwartz inequality. Kernels satisfying conditions in (as3) include Gaussian, Laplacian, and Cauchy (Rahimi and Recht, 2007). In general, (as1)-(as3) are standard in online convex optimization (OCO) (Shalev-Shwartz, 2011; Hazan, 2016), and in kernel-based learning (Micchelli and Pontil, 2005; Rahimi and Recht, 2007; Lu et al., 2016).

With regard to the performance of an online algorithm, static regret is commonly adopted as a metric by most OCO schemes to measure the difference between the aggregate loss of an OCO algorithm, and that of the best fixed function approximant in hindsight, e.g., (Shalev-Shwartz, 2011; Hazan, 2016). Specifically, for a generic sequence $\{\hat{f}_t\}$ generated by an RF-based kernel learning algorithm $\mathcal{A}$, its static regret is

$$\mathrm{Reg}_{\mathcal{A}}^{\mathrm{s}}(T):=\sum_{t=1}^T\mathcal{L}_t(\hat{f}_t(\mathbf{x}_t))-\sum_{t=1}^T\mathcal{L}_t(f^*(\mathbf{x}_t)) \tag{25}$$

where $\hat{f}_t$ will henceforth represent $\hat{f}_t^{\text{RF}}$ without the superscript for notational brevity; and, $f^*(\cdot)$ is obtained as the batch solution

$$f^*(\cdot) \in \arg \min_{\{f_p^*, p\in\mathcal{P}\}} \sum_{t=1}^{T} \mathcal{L}_t(f_p^*(\mathbf{x}_t)) \quad \text{with} \quad f_p^*(\cdot) \in \arg \min_{f\in\mathcal{F}_p} \sum_{t=1}^{T} \mathcal{L}_t(f(\mathbf{x}_t)) \tag{26}$$

with $\mathcal{F}_p := \mathcal{H}_p$, and $\mathcal{H}_p$ representing the RKHS induced by κ_p. Using (25) and (26), we first establish the static regret of our Raker approach in the following lemma.

Lemma 1 *Under (as1), (as2), and with $\hat{f}_p^*$ as in (26) with $\mathcal{F}_p := \{\hat{f}_p | \hat{f}_p(\mathbf{x}) = \boldsymbol{\theta}^\top \mathbf{z}_p(\mathbf{x}), \forall \boldsymbol{\theta} \in \mathbb{R}^{2D}\}$, the sequences $\{\hat{f}_{p,t}\}$ and $\{\bar{w}_{p,t}\}$ generated by Raker satisfy the following bound*

$$\sum_{t=1}^{T} \mathcal{L}_t\Big(\sum_{p=1}^{P} \bar{w}_{p,t}\hat{f}_{p,t}(\mathbf{x}_t)\Big) - \sum_{t=1}^{T} \mathcal{L}_t\left(\hat{f}_p^*(\mathbf{x}_t)\right) \leq \frac{\ln P}{\eta} + \frac{\|\boldsymbol{\theta}_p^*\|^2}{2\eta} + \frac{\eta L^2 T}{2} + \eta T \tag{27}$$

where $\boldsymbol{\theta}_p^*$ is associated with the best RF function approximant $\hat{f}_p^*(\mathbf{x}) = \left(\boldsymbol{\theta}_p^*\right)^\top \mathbf{z}_p(\mathbf{x})$.

Proof: See Appendix A. ■

Besides Raker's static regret bound, the next theorem compares the Raker loss relative to that of the best functional estimator in the original RKHS.

Theorem 1 *Under (as1)-(as3) and with f_p^* in (26) belonging to the RKHS $\mathcal{H}_p$, for a fixed $\epsilon > 0$, the following bound holds with probability at least $1 - 2^8\left(\frac{\sigma_p}{\epsilon}\right)^2 \exp\left(\frac{-D\epsilon^2}{4d+8}\right)$*

$$\sum_{t=1}^{T} \mathcal{L}_t\left(\sum_{p=1}^{P} \bar{w}_{p,t}\hat{f}_{p,t}(\mathbf{x}_t)\right) - \min_{p\in\{1,\ldots,P\}} \sum_{t=1}^{T} \mathcal{L}_t\left(f_p^*(\mathbf{x}_t)\right) \leq \frac{\ln P}{\eta} + \frac{(1+\epsilon)C^2}{2\eta} + \frac{\eta L^2 T}{2} + \eta T + \epsilon LTC \tag{28}$$

where C is a constant, while $\sigma_p^2 := \mathbb{E}_{\mathbf{v}}^{\pi_{\kappa_p}}[\|\mathbf{v}\|^2]$ is the second-order moment of the RF vector norm. Setting $\eta = \epsilon = \mathcal{O}(1/\sqrt{T})$ in (28), the static regret in (25) leads to

$$\text{Reg}_{\text{Raker}}^{\text{s}}(T) = \mathcal{O}(\sqrt{T}). \tag{29}$$

Proof: See Appendix B. ■

Observe that the probability of (28) to hold grows as D increases, and one can always find a D to ensure a positive probability for a given ϵ. Bearing this in mind, we will henceforth use "with high probability" (w.h.p.) to summarize the sense (28) and (29) hold. Theorem 1 establishes that with proper choice of parameters, the Raker achieves sub-linear regret relative to the best static function approximant in (26).

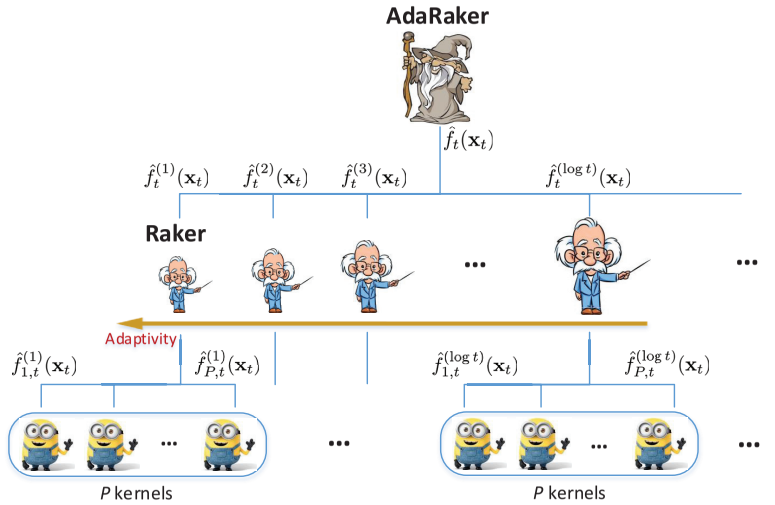


Figure 1: Hierarchical AdaRaker structure. Experienced experts in the middle layer present a Raker instance, where the size of expert cartoons is proportional to the interval length.

4. Online MKL in Environments with Unknown Dynamics

Our Raker in Section 3 combines an ensemble of kernel learners 'on the fly,' and performs on average as the "best" fixed function, thus fulfilling the learning objective in environments with zero (or slow) dynamics. To broaden its scope to environments with unknown dynamics, this section introduces an **ada**ptive **Raker** approach (termed **AdaRaker**).

4.1. AdaRaker with hierarchical ensembles

As with any online learning algorithm, the choice of η in (19) and (21) affects the performance critically. Especially in environments with unknown dynamics, a large η improves the tracking ability of fast-varying functions, while a smaller one allows improved estimation of slow-varying parameters $\{\boldsymbol{\theta}_t, w_{p,t}\}$. The optimal choice of η_t clearly depends on the variability of the optimal function estimator (Kivinen et al., 2004; Besbes et al., 2015). Selecting $\{\eta_t\}$ however, is formidably challenging if the environment dynamics are unknown.

Toward addressing this challenge, our idea here is to hedge between multiple Raker learners with different learning rates. Specifically, we view each Raker instance in Algorithm 1 as a black box algorithm $\mathcal{A}_I$, where the subscript I represents the algorithm running on interval $I := [\underline{I}, \bar{I}]$ starting from slot $\underline{I}$ to slot $\bar{I}$. Let a pre-selected set $\mathcal{I}$ collect all these intervals, the design of which will be specified later. At the beginning of each interval $I \in \mathcal{I}$, a new instance of the online Raker algorithm $\mathcal{A}_I$ is initialized with an interval-specific learning rate $\eta^{(I)} := \min\{1/2, \eta_0/\sqrt{|I|}\}$ with constant $\eta_0 > 0$. Allowing for overlap between intervals, multiple Raker instances $\{\mathcal{A}_I\}$ will be run in parallel. Consider now

Algorithm 2 AdaRaker for online MKL in dynamic environments

Initialization: learner weights $\{h_1^{(I)}\}$, and their learning rates $\{\eta^{(I)}\}$.
for $t = 1, 2, \ldots, T$ **do**
  Obtain $\hat{f}_t^{(I)}(\mathbf{x}_t)$ from each Raker instance $\mathcal{A}_I$, $I \in \mathcal{I}(t)$.
  Predict $\hat{f}_t(\mathbf{x}_t)$ via a weighted combination (33).
  Observe loss function $\mathcal{L}_t$, and incur $\mathcal{L}_t(\hat{f}_t(\mathbf{x}_t))$.
  for $I \in \mathcal{I}(t)$ **do**
    Incur loss $\mathcal{L}_t(\hat{f}_t^{(I)}(\mathbf{x}_t))$.
    Update $\hat{f}_t^{(I)}$ via Raker in Algorithm 1.
    Update weights $h_{t+1}^{(I)}$ via (31).
  end for
end for

collecting all active intervals at the current slot t in the set

$$\mathcal{I}(t) := \{I \in \mathcal{I} \,|\, t \in [\underline{I}, \bar{I}]\},\ \forall t \in \mathcal{T}. \tag{30}$$

For each Raker instance $\mathcal{A}_I$ with $I \in \mathcal{I}(t)$, let $\hat{f}_t^{(I)}(\cdot)$ denote its output at time t that combines multiple kernel-based function estimators, and $\mathcal{L}_t(\hat{f}_t^{(I)}(\mathbf{x}_t))$ represent the associated instantaneous loss. The output of the ensemble learner $\mathcal{A}$ at time t is the weighted combination of outputs from all learners, namely $\{\hat{f}_t^{(I)}, \forall I \in \mathcal{I}(t)\}$. With $h_t^{(I)}$ denoting the weight of the Raker instance $\mathcal{A}_I$, we will update it online via

$$h_{t+1}^{(I)} = \begin{cases} 0, & \text{if } t \notin I \\ \eta^{(I)}, & \text{if } t = \underline{I} \\ h_t^{(I)} \exp\left(-\eta^{(I)} r_t^{(I)}\right), & \text{else} \end{cases} \tag{31}$$

where $\underline{I}$ is the first time slot of interval I, and the loss of $\mathcal{A}_I$ *relative* to the overall loss is

$$r_t^{(I)} = \mathcal{L}_t(\hat{f}_t(\mathbf{x}_t)) - \mathcal{L}_t(\hat{f}_t^{(I)}(\mathbf{x}_t)),\ \forall I \in \mathcal{I}(t). \tag{32}$$

Intuitively thinking, one would wish to decrease (increase) the weights of those instances with small (large) losses in future rounds. Using update (31), and defining the normalized weight as $\bar{h}_t^{(I)} := h_t^{(I)} / \sum_{J \in \mathcal{I}(t)} h_t^{(J)}$, the overall output is given by

$$\hat{f}_t(\mathbf{x}) := \sum_{I \in \mathcal{I}} \bar{h}_t^{(I)} \hat{f}_t^{(I)}(\mathbf{x}) \quad \text{with} \quad \hat{f}_t^{(I)}(\mathbf{x}) := \sum_{p \in \mathcal{P}} \bar{w}_{p,t}^{(I)} \hat{f}_{p,t}^{(I)}(\mathbf{x}) \tag{33}$$

where $\{\bar{w}_{p,t}^{(I)}\}$ are the kernel combination weights generated by Raker $\mathcal{A}_I$ (cf. (21)).

The AdaRaker scheme is summarized in Algorithm 2, and depicted in Figure 1.

Selecting judiciously variable-length intervals in $\mathcal{I}$ can affect performance critically. Such a selection criterion for achieving interval regret has been reported in (Daniely et al., 2015). Instead, our pursuit is a hierarchical ensemble design for online MKL in environments with unknown dynamics using scalable RF-based function approximants. This hierarchical

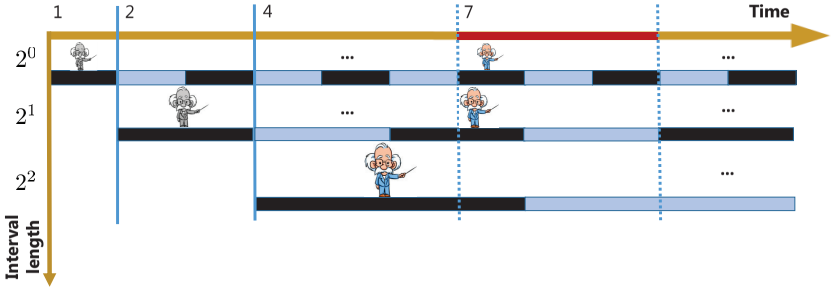


Figure 2: AdaRaker as an ensemble of Rakers with different learning rates: Each light/dark black interval initiates a Raker learner. At slot 7, colored experts are active, and gray ones are inactive.

design is well motivated because with long intervals, the Raker loss per interval is relatively low in slow-varying settings, but higher as the dynamics become more pronounced. On the other hand, a short interval can hedge against a possibly rapid change, but its performance on each interval could suffer if the objective stays nearly static. Bearing these tradeoffs in mind, we present next a simple yet efficient interval partitioning scheme.

Illustration of interval sets: *Consider partitioning the entire horizon into intervals of length* $2^0, 2^1, 2^2, \ldots$. *Intervals of length* 2^j *with a given* $j \in \mathbb{N}$ *are consecutively assigned without overlap starting from* $t = 2^j$. *In the non-overlapping case, define a set of intervals* $\mathcal{I}_j = [\underline{I}_j, \bar{I}_j]$ *such that each interval's length is* $|\mathcal{I}_j| = \bar{I}_j - \underline{I}_j + 1 = 2^j$, $j \in \mathbb{N}$. *For this selection of intervals, each time slot* t *is covered by a set of at most* $\lceil \log_2 t \rceil$ *intervals, which forms the active set of intervals* $\mathcal{I}(t)$ *at time* t. *See the diagram in Fig. 2.*

4.2. Dynamic regret analysis of AdaRaker

The static regret in Theorem 1 is with respect to a time-invariant optimal function estimator benchmark. In dynamic environments however, this optimal function benchmark may change over time - what justifies this subsection's performance analysis of AdaRaker.

Our analysis will rely on the *dynamic regret* that is related to tracking regret, and has been introduced in (Besbes et al., 2015; Jadbabaie et al., 2015) to quantify the performance of online algorithms. The dynamic regret is defined as (cf. (25))

$$\mathrm{Reg}_{\mathcal{A}}^{\mathrm{d}}(T) := \sum_{t=1}^{T} \mathcal{L}_t(\hat{f}_t(\mathbf{x}_t)) - \sum_{t=1}^{T} \mathcal{L}_t(f_t^*(\mathbf{x}_t)) \tag{34}$$

where the benchmark is the aggregate loss incurred by a sequence of the best dynamic functions$\{f_t^*\}$ from $\mathcal{F}$ formed by the union of function spaces $\mathcal{H}_p$ induced by $\{\kappa_p\}$, given by

$$f_t^*(\cdot) \in \arg \min_{\{f_{p,t}^* , p \in \mathcal{P}\}} \mathcal{L}_t(f_p^*(\mathbf{x}_t)) \quad \text{with} \quad f_{p,t}^*(\cdot) \in \arg \min_{f \in \mathcal{H}_p} \mathcal{L}_t(f(\mathbf{x}_t)) \tag{35}$$

Comparing (26) with (35) we deduce that the dynamic regret is always larger than the static regret in (25). Thus, a sub-linear dynamic regret implies a sub-linear static regret, but not vice versa. Given $\{\mathcal{L}_t\}$, AdaRaker generates functions $\{\hat{f}_t\}$ to minimize the dynamic regret.

To assess the AdaRaker performance, we will start with an *intermediate* result on the static regret associated with any sub-interval $I \subseteq \mathcal{T}$.

Lemma 2 *Under (as1)-(as3), the static regret on any interval $I \subseteq \mathcal{T}$ is given by*

$$\mathrm{Reg}^{\mathrm{s}}_{\mathcal{A}}(|I|) := \sum_{t\in I} \mathcal{L}_t(\hat{f}_t(\mathbf{x}_t)) - \sum_{t\in I} \mathcal{L}_t(f_I^*(\mathbf{x}_t)) \tag{36}$$

where $|I|$ denotes the length of interval I, and the best time-invariant function approximant is $f_I^* \in \arg\min_{f\in\bigcup_{p\in\mathcal{P}}\mathcal{H}_p} \sum_{t\in I}\mathcal{L}_t(f(\mathbf{x}_t))$, with $\mathcal{H}_p$ denoting the RKHS induced by κ_p. Then for any interval $I \subseteq \mathcal{T}$ and fixed positive constants C_0, C_1, the following bound holds

$$\mathrm{Reg}^{\mathrm{s}}_{\mathrm{AdaRaker}}(|I|) \leq C_0\sqrt{|I|} + C_1 \ln T\sqrt{|I|}, \text{ w.h.p.} \tag{37}$$

Proof: See Appendix C. ■

Lemma 2 establishes that by combining Raker learners with different learning rates, AdaRaker can achieve sub-linear static regret over *any* interval I with arbitrary interval length. This also holds for intervals overlapping with multiple intervals; see e.g., the red interval in Fig. 2. Clearly, the best fixed solution in (36) is interval specific, which can vary over different intervals. This is qualitatively why the function approximants generated by AdaRaker can cope with a *time-varying* benchmark. Such an intuition will in fact become quantitative in the next theorem, which establishes the dynamic regret for AdaRaker.

Theorem 2 *Suppose (as1)-(as3) are satisfied, and define the accumulated variation of online loss functions as*

$$\mathcal{V}(\{\mathcal{L}_t\}_{t=1}^T) := \sum_{t=1}^{T} \max_{f\in\mathcal{F}} \left|\mathcal{L}_{t+1}(f(\mathbf{x}_{t+1})) - \mathcal{L}_t(f(\mathbf{x}_t))\right| \tag{38}$$

where $\mathcal{F} := \bigcup_{p\in\mathcal{P}}\mathcal{H}_p$. Then AdaRaker can afford a dynamic regret in (34) *bounded by*

$$\begin{aligned}\mathrm{Reg}^{\mathrm{d}}_{\mathrm{AdaRaker}}(T) \leq& (2 + C_0 + C_1 \ln T) T^{\frac{2}{3}} \mathcal{V}^{\frac{1}{3}}(\{\mathcal{L}_t\}_{t=1}^T) \\ =& \tilde{\mathcal{O}}\left(T^{\frac{2}{3}}\mathcal{V}^{\frac{1}{3}}(\{\mathcal{L}_t\}_{t=1}^T)\right), \text{ w.h.p.}\end{aligned} \tag{39}$$

where $\tilde{\mathcal{O}}$ neglects the lower-order terms with a polynomial $\log T$ rate.

Proof: See Appendix D. ■

Theorem 2 asserts that AdaRaker's dynamic regret depends on the variation of loss functions in (38), and on the horizon T. Interesting enough, whenever the loss functions *do not vary on average*, meaning $\mathcal{V}(\{\mathcal{L}_t\}_{t=1}^T) = \mathbf{o}(T)$, AdaRaker achieves sub-linear dynamic regret. To this end, it is useful to present an example where this argument holds.

Intermittent switches: With $\mathcal{L}_t \neq \mathcal{L}_{t+1}$ defining a switch, consider that the number of switches is sub-linear over T; that is, $\sum_{t=1}^T \mathbb{1}(\mathcal{L}_t \neq \mathcal{L}_{t+1}) = T^\gamma$, $\forall\gamma \in [0,1)$. Then it follows that $\mathcal{V}(\{\mathcal{L}_t\}_{t=1}^T) = \mathcal{O}(T^\gamma)$, since the one-slot variation of the loss functions is bounded.

Other setups with sub-linear accumulated variation emerge, e.g., when the per-slot variation decreases as $\mathcal{V}(\mathcal{L}_t) = \mathcal{O}(t^{\gamma-1})$, $\forall\gamma \in [0,1)$. Besides dynamic losses, sub-linear dynamic regrets can be also effected by confining the variability of optimal function estimators.

Theorem 3 *Suppose the conditions of Theorem 2 hold, and define the regret relative to an m-switching dynamic benchmark as $\mathrm{Reg}^m_{\mathcal{A}}(T) := \sum_{t=1}^T \mathcal{L}_t(\hat{f}_t(\mathbf{x}_t)) - \sum_{t=1}^T \mathcal{L}_t(\check{f}_t^*(\mathbf{x}_t))$, where $\{\check{f}_t^*\}$ is any trajectory from*

$$\left\{\{\check{f}_t^*\}_{t=1}^T \in \bigcup\nolimits_{p\in\mathcal{P}} \mathcal{H}_p \middle| \sum\nolimits_{t=1}^T \mathbb{1}(\check{f}_t^* \neq \check{f}_{t-1}^*) \leq m\right\}. \tag{40}$$

With C_0 and C_1 denoting some universal constants, it then holds w.h.p. that

$$\mathrm{Reg}^m_{\mathrm{AdaRaker}}(T) \leq (C_0 + C_1 \ln T)\sqrt{Tm} = \tilde{\mathcal{O}}\left(\sqrt{Tm}\right). \tag{41}$$

Proof: See Appendix E. ■

Theorem 3 asserts that without prior knowledge of the environment dynamics, the dynamic regret of AdaRaker is sub-linearly growing with time, provided that the number of changes of the optimal function estimators is sub-linear in T; that is, $\mathrm{Reg}^m_{\mathrm{AdaRaker}}(T) = \mathbf{o}(T)$ given $m = \mathbf{o}(T)$. Therefore, our AdaRaker can track the optimal dynamic functions, if the optimal function *varies slowly* over time; e.g., it does not change in the long-term average sense. While the conditions to guarantee optimality in dynamic settings may appear restrictive, they are practically relevant, since abrupt changes or adversarial samples will likely not happen at each and every slot in practice.

5. Numerical Tests

This section evaluates the performance of our novel algorithms in online regression tasks using both synthetic and real-world datasets.

In the subsequent tests, we use the following benchmarks.

RBF: the online *single* kernel learning method using Gaussian kernels, a.k.a. radial basis functions (RBFs), with bandwidth $\sigma^2 = \{0.1, 1, 10\}$ (cf. RBF01, RBF1, RBF10);
POLY: the online *single* kernel method using polynomial kernels, with degree $d = \{2, 3\}$ (cf. POLY2, POLY3);
LINEAR: the online *single* kernel learning method using a linear kernel;
AvgMKL: the online *single* kernel learning method using the average of candidate kernels without updating the weights;
OMKL: the popular online (O)MKL algorithm without a budget (Sahoo et al., 2014);
OMKL-B: the OMKL algorithm on a budget for regression modified from its single kernel version (Kivinen et al., 2004), with the kernel combination weights updated as (21);
M-Forgetron: the online multi-kernel based Forgetron modified from its single kernel version (Dekel et al., 2008), with the kernel combination weights updated as in (21);

Time index	$[1, 200]$	$[201, 1000]$	$[1001, 2000]$	$[2001, 2300]$	$[2301, 3000]$
σ^2	0.01	1	10	0.01	1
Time index	$[3001, 3500]$	$[3501, 4300]$	$[4301, 5100]$	$[5101, 5900]$	$5901, 6500$
σ^2	10	0.01	1	0.01	0.1

Table 1: Intervals and $\{\sigma^2\}$ for synthetic dataset.

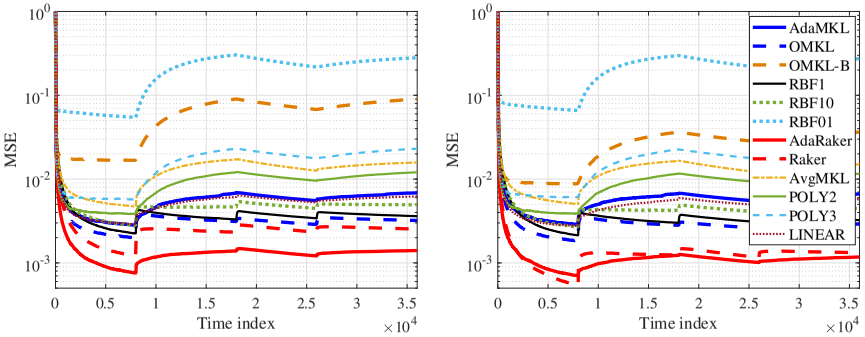
AdaMKL: the adaptive version of OMKL that operates in a similar fashion as Algorithm 2, but instead of using our Raker as an ensemble, it adopts OMKL as an instance $\mathcal{A}_I$.

Note that AdaMKL, OMKL-B, and M-Forgetron have not been formally proposed in existing works, but we introduced them here only for comparison purposes. All the considered MKL approaches use a dictionary of Gaussian kernels with $\sigma^2 = \{0.1, 1, 10\}$, and AvgMKL, OMKL, AdaMKL, OMKL-B, and M-Forgetron also include a linear, and a polynomial kernel with order of 2 into their kernel dictionary. For all MKL approaches, the stepsize for updating kernel combination weights in (21) is chosen as 0.5 uniformly, while the stepsize for updating per-kernel function estimators will be specified later in each test. The regularization parameter is set equal to $\lambda = 0.01$ for all approaches. Entries of $\{\mathbf{x}_t\}$ and $\{y_t\}$ are normalized to lie in $[0, 1]$. Regarding AdaMKL and AdaRaker, multiple instances are initialized on intervals with length $|I| := 2^0, 2^1, 2^2, \ldots$, along with the corresponding learning rate on the interval I as $\eta^{(I)} := \min\{1/2, 10/\sqrt{|I|}\}$; see the example in Figure 2. All the results in the tables were reported using the performance at the last time index.

5.1. Synthetic data tests for regression

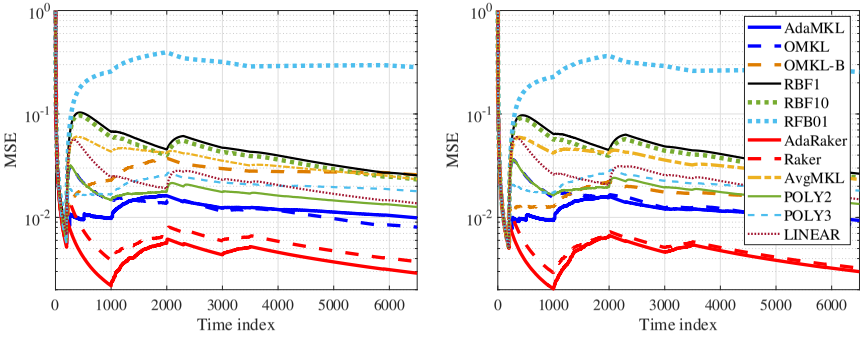
This subsection presents the synthetic data tests for regression.

Data generation. In this test, two synthetic datasets were generated as follows.
For *Dataset 1*, the feature vectors $\{\mathbf{x}_t \in \mathbb{R}^{10}\}_{t=1}^{14,000}$ are generated from the standardized Gaussian distribution, while y_t is generated as $y_t = \sum_{\tau=1}^{t} \alpha_\tau \kappa_\tau(\mathbf{x}_t, \mathbf{x}_\tau)$, where $\{\alpha_t\}$ is generated as $\alpha_t = 1 + e_t$ with $e_t \sim \mathcal{N}(0, \sigma_\alpha^2)$ and $\sigma_\alpha = 0.01$, while $\{\kappa_t\}$ are kernel functions that change overtime: for $t \in [1, 8000] \bigcup [18001, 26000]$, κ_t is a Gaussian kernel with $\sigma^2 = 1$, while for $t \in [8001, 18000] \bigcup [26001, 36000]$ the Gaussian kernel has $\sigma^2 = 10$. Therefore, the underlying nonlinear relationship between $\mathbf{x}_t$ and y_t undergoes intermittent changes, which come from corresponding changes in the optimal kernel combinations.
Dataset 2 is generated with more variance and switching points. Specifically, the feature vectors are generated from the standardized Gaussian distribution, while y_t is generated as $y_t = \sum_{\tau=1}^{t} \alpha_\tau \kappa_\tau(\mathbf{x}_t, \mathbf{x}_\tau)$, where $\{\kappa_t\}$ change over 10 intervals with different σ^2; see Table 1.
Testing performance. The performance of all schemes is tested in terms of the mean-square (prediction) error $\text{MSE}(t) := (1/t) \sum_{\tau=1}^{t} (y_\tau - \hat{y}_\tau)^2$ in Figure 3 and Figure 4, and their CPU time is listed in Table 2. For OMKL-B, $B = 20$ and 50 most recent data samples were kept in the budget; and for RF-based Raker and AdaRaker approaches, $D = 20$ and 50 orthogonal random features were used by default. The default stepsize is chosen as $1/\sqrt{T}$ for RBF, POLY, LINEAR, AvgMKL, OMKL, OMKL-B and Raker. In both tests, AdaRaker outperforms the alternatives in terms of MSE, especially when the true nonlinear relationship between $\mathbf{x}_t$ and y_t changes; e.g., compare the MSE of KL-RBF and Raker with that of AdaRaker at $t = 8000, 18000, 26000$ in Figure 3, and $t = 200, 2000, 3000, 3500$



(a) (b)

Figure 3: MSE performance on synthetic Dataset 1: a) $D = B = 20$; b) $D = B = 50$.



(a) (b)

Figure 4: MSE performance on synthetic Dataset 2: a) $D = B = 20$; b) $D = B = 50$.

in Figure 4. This corroborates the effectiveness of the novel AdaRacker method that can flexibly select learning rates according to the variability of the environments, and adaptively combine multiple kernels when the optimal underlying nonlinear relationship is varying over time. In addition, MKL approaches including our Raker approach enjoy lower MSE than that of the single-kernel approaches as well as the simple AvgMKL approach, which is also aligned with our design principle of developing MKL schemes that broaden generalizability of a kernel-based learner over a larger function space.

Table 2 records the CPU time of all benchmark algorithms running tests on two different datasets. It can be observed that leveraging the RF-based approximation, the proposed AdaRaker and Raker algorithms are much faster than AdaMKL and OMKL; hence, they are preferable especially for large-scale datasets. Although the CPU time of OMKL-B with

	Dataset 1		Dataset 2	
Setting	$D = B = 20$	$D = B = 50$	$D = B = 20$	$D = B = 50$
AdaMKL	318.52		27.29	
OMKL	157.10		5.47	
RBF	47.83		1.06	
POLY2	6.01		0.47	
POLY3	28.27		1.24	
LINEAR	4.80		0.35	
AvgMKL	144.85		5.02	
OMKL-B	3.75	4.05	0.72	0.77
Raker	1.39	1.53	0.18	0.20
AdaRaker	21.94	24.24	3.32	3.54

Table 2: CPU time (in seconds) on synthetic datasets. RBF, POLY represents all single-kernel methods using RBF and polynomial kernels, since they have the same CPU time.

Dataset	**# features** (d)	**# samples** (T)	**feature type**
Twitter	77	14,000	real & integer
Twitter (Large)	77	100,000	real & integer
Tom's hardware	96	10,000	real & integer
Energy	27	18,600	real
Air quality	13	9,358	real

Table 3: A summary of real datasets used in the tests.

a budget size $B = 20$ or $B = 50$ is relatively low, OMKL-B does not perform as well as AdaRaker and Raker algorithms. Therefore, the AdaRaker and Raker approaches attain a sweet-spot in the performance-complexity tradeoff.

5.2. Real data tests for online regression

To further evaluate our algorithms in real-world scenarios, the present subsection is devoted to testing and comparing on several popular real datasets.

Datasets description. Performance is tested on benchmark datasets from UCI machine learning repository (Lichman, 2013).

- **Twitter** dataset consists of $T = 14,000$ samples from a popular micro-blogging platform Twitter, where $\mathbf{x}_t \in \mathbb{R}^{77}$ include features such as the number of new interactive authors, and the length of discussion on a given topic, while y_t represents the average number of active discussion (popularity) on a certain topic (Kawala et al., 2013). A larger dataset with $T = 100,000$ is also included for testing only (Ada)Raker and OMKL-B, since other methods do not scale to such a large T.

- **Tom's hardware** dataset contains $T = 10,000$ samples from a worldwide new technology forum, where a 96-dimensional feature vector includes the number of discussions involving a certain topic, while y_t represents the average number of display about a certain topic on Tom's hardware (Kawala et al., 2013).

Algorithms/ Datasets	Twitter	Tom's	Energy	Air
RBF ($\sigma^2 = 0.1$)	27.0	14.4	28.9	26.3
RBF ($\sigma^2 = 1$)	13.5	17.0	28.8	12.7
RBF ($\sigma^2 = 10$)	23.3	18.8	28.8	15.5
POLY2	12.7	22.3	28.8	7.34
POLY3	20.4	22.7	28.9	5.91
LINEAR	8.57	19.5	28.8	10.7
AvgMKL	14.4	17.5	28.7	11.9
OMKL	8.55	14.3	28.1	6.4
AdaMKL	16.1	18.4	30.4	10.1
OMKL-B ($B = 50$)	27.0	22.1	73.3	35.9
Raker ($D = 50$)	3.0	3.4	19.3	2.0
AdaRaker ($D = 50$)	**2.6**	**1.9**	**13.8**	**1.3**

Table 4: MSE (10^{-3}) performance of different algorithms with stepsize $1/\sqrt{T}$.

Algorithms/ Datasets	Twitter	Tom's	Energy	Air
RBF ($\sigma^2 = 0.1$)	17.2	3.3	16.6	8.1
RBF ($\sigma^2 = 1$)	3.3	5.1	16.4	2.8
RBF ($\sigma^2 = 10$)	5.6	13.6	16.4	18.9
POLY2	8.1	15.9	16.2	3.3
POLY3	20.4	20.7	16.2	4.6
LINEAR	2.7	4.8	16.3	2.9
AvgMKL	7.1	6.2	16.3	2.8
OMKL	4.2	3.3	16.2	2.4
AdaMKL	16.1	18.4	30.4	10.1
OMKL-B ($B = 50$)	9.9	11.8	19	7.1
Raker ($D = 50$)	2.9	2.6	**13.8**	**1.3**
AdaRaker ($D = 50$)	**2.6**	**1.9**	**13.8**	**1.3**

Table 5: MSE (10^{-3}) performance of different algorithms with optimally chosen stepsizes.

- **energy** dataset consists of $T = 18,600$ samples, with each $\mathbf{x}_t \in \mathbb{R}^{27}$ describing the humidity and temperature indoors and outdoors, while y_t denotes the energy use of light fixtures in the house (Candanedo et al., 2017).

- **air quality** dataset collects $T = 9,358$ instances of hourly averaged responses from five chemical sensors located in a polluted area of Italy. The averaged sensor response $\mathbf{x}_t \in \mathbb{R}^{13}$ contains the hourly concentrations of e.g., CO, Non Metanic Hydrocarbons, and Nitrogen Dioxide (NO2), where the goal is to predict the concentration of polluting chemicals y_t in the air (De Vito et al., 2008).

To highlight the effectiveness of our approaches, the datasets mainly include time series data, where non-stationarity is more likely to happen; see Table 3 for a summary.

MSE performance. The MSE performance of each algorithm on the aforementioned datasets is presented in Table 4. By default, we use the complexity $B = D = 50$ for OMKL-B and (Ada)Raker, and the stepsize $1/\sqrt{T}$ for RBF, POLY, LINEAR, AvgMKL,

MSE	OMKL-B				Raker				AdaRaker
Stepsize	$1/\sqrt{T}$	$0.5/\sqrt{t}$	$0.1/\sqrt{t}$	Tuned	$1/\sqrt{T}$	$0.5/\sqrt{t}$	$0.1/\sqrt{t}$	Tuned	/
Twitter	27.0	27.1	29.6	9.9	3.0	17.9	4.3	2.9	**2.6**
Tom's	22.1	22.1	22.6	11.8	3.4	2.0	7.6	2.6	**1.9**
Energy	73.3	74.1	79.5	19.0	19.3	29.5	25.1	**13.8**	**13.8**
Air	35.9	35.9	40.1	7.1	2.0	29.1	4.0	**1.3**	**1.3**
Twitter (Large)	20.7	27.2	28.0	11.3	3.2	3.1	3.3	3.0	**2.7**

Table 6: MSE (10^{-3}) versus the choice of stepsizes with complexity $B = D = 50$.

OMKL, OMKL-B and Raker. To boost the performance of each algorithm, their MSE when using manually tuned stepsizes is also reported in Table 5, which selects the best stepsize on each dataset among $\{10^{-3}, 10^{-2}, \cdots, 10^{3}\}/\sqrt{T}$. A common observation is that leveraging the flexibility of multiple kernels, MKL methods in most cases outperform the algorithms using only a single kernel. By simply averaging over all the kernels, AvgMKL outperforms most of single kernel methods, but performs worse than the adaptive kernel combination methods. This confirms that relying on a pre-selected kernel function is not sufficient to guarantee low fitting loss, while allowing the MKL approaches to select the best kernel combinations in a data-driven fashion holds the key for improved performance.

In most tested datasets, Raker obtains function approximants with lower MSE relative to MKL alternatives without RF approximation. Furthermore, incorporating multiple Raker instances with variable learning rates, AdaRaker consistently yields the lowest MSE in all the tests. As it has been shown in the synthetic data test, the sizable performance gain of AdaRaker appears when the underlying nonlinear models change in the tested time-series datasets. This observation is aligned with our design principle of AdaRaker; that is, when the optimal function predictor varies slowly (fast), AdaRaker tends to select a Raker instance with small (large) learning rate. Interesting enough, even with adaptive learning rate, AdaMKL does not perform as well as OMKL in some tests. This is partially because unlike AdaRaker with fixed number of RFs, each instance in AdaMKL involves a different *number of support vectors* (samples). The instance operating on the longest interval contains at most $T/2$ support vectors, which may deteriorate performance relative to OMKL with T support vectors.

Table 6 further compares the MSE performance of AdaRaker with OMKL-B and Raker using different stepsizes. Clearly, the performance of OMKL-B and Raker is sensitive to the choice of stepsizes. While the optimal stepsize varies from dataset to dataset, selecting a constant stepsize $1/\sqrt{T}$ generally leads to better performance than a diminishing one of $\mathcal{O}(1/\sqrt{t})$. In the online scenarios however, the choice $1/\sqrt{T}$ may not be feasible if T is unknown ahead of time. In contrast, AdaRaker obtained the best MSE performance without knowing T, and without the need of stepsize selection, which confirms that AdaRaker is capable of adapting its stepsize to variable environments with unknown dynamics.

Computational complexity. The CPU time of all the considered schemes is recorded under all the tests; see Table 7. It is evident that in all tests, our RF-based MKL methods including Raker and AdaRaker are computationally more efficient than other MKL methods except that OMKL-B is faster than AdaRaker. Intuitively speaking, the per-slot complexity of Raker does not grow with time, since it requires computing only one inner product of two

Algorithms/ Datasets	Twitter	Tom's	Energy	Air
RBF	54.7	32.5	26.6	1.71
POLY2	2.25	1.16	2.42	0.58
POLY3	5.62	2.97	7.90	1.51
LINEAR	1.83	0.98	196	0.39
AvgMKL	148.4	81.6	82.4	5.29
OMKL	153.5	81.9	83.3	5.90
AdaMKL	164.1	102.7	117.9	35.3
OMKL-B ($B = 50$)	1.89	1.42	2.02	0.89
Raker ($D = 50$)	**0.51**	**0.38**	**0.65**	**0.28**
AdaRaker ($D = 50$)	8.64	6.03	10.94	5.28

Table 7: A summary of CPU time (second) on real datasets.

MSE	OMKL-B			Raker			AdaRaker		
Complexity	10	50	100	10	50	100	10	50	100
Twitter	28.9	27.0	26.1	5.9	3.0	3.0	3.8	**2.6**	**2.6**
Tom's	22.7	22.1	21.7	8.1	3.4	2.3	7.0	1.9	**1.8**
Energy	79.1	73.3	67.9	25.7	19.3	16.4	18.7	13.8	**13.3**
Air pollution	36.7	35.9	35.8	10.1	2.0	1.7	4.3	1.3	**1.2**
Twitter (Large)	25.0	20.7	19.0	3.9	3.2	3.0	3.3	**2.7**	**2.7**

Table 8: MSE (10^{-3}) versus complexity. For OMKL-B, the complexity measure is the data budget B; and for (Ada)Raker, the complexity measure is the number of RFs D.

$2D$-dimensional vectors per kernel learner, while the computational complexity of AdaMKL, OMKL, POLY, LINEAR, AvgMKL, and RBF increases with time at least linearly. With a fixed budget size, OMKL-B enjoys light-weight updates that leads to a lower CPU time than alternatives, but higher than Raker. However, given such a limited budget of data, OMKL-B exhibits higher MSE than AdaRaker and Raker; see MSE in Tables 4 and 6.

Running multiple instances of Raker in parallel, the complexity of AdaRaker is reasonably higher than Raker (roughly $\log T$ times higher), but its runtime is still only around 10% of that of AdaMKL, and significantly lower than other single-kernel alternatives especially when the actual feature dimension d is higher than the number of random features D. The computational advantage of our MKL algorithms in this test also corroborates the quantitative analysis at the end of Section 3.2. Regarding the tradeoff between learning accuracy and complexity, a delicate comparison among OMKL-B, Raker and AdaRaker follows next.

Accuracy versus complexity. To further understand the tradeoff between complexity and learning accuracy, the performance of three scalable methods AdaRaker, Raker and OMKL-B is tested under different parameter settings, e.g., D, the number of random features, and B, the number of budgeted data. The MSE performance is reported in Table 8 after one pass of all data in each dataset, while the corresponding CPU time is in Table 9.

Not surprising, all three algorithms require longer CPU time as the complexity (in terms of B or D) increases. For given complexity (same B and D), Raker requires the lowest CPU time, and its MSE is also markedly lower than that of OMKL-B in all tests. On the other hand, AdaRaker always attains the lowest MSE, and its performance gain is

Time	OMKL-B			Raker			AdaRaker		
Complexity	10	50	100	10	50	100	10	50	100
Twitter	1.42	1.89	2.84	0.42	0.51	0.80	7.58	8.64	11.65
Tom's	1.00	1.42	2.81	0.41	0.38	0.56	5.09	6.03	8.98
Energy	1.84	2.02	2.32	0.58	0.65	0.76	9.96	10.94	12.47
Air pollution	0.82	0.89	0.97	0.24	0.28	0.32	4.09	5.28	5.29
Twitter (Large)	12.90	16.34	23.6	6.07	6.63	8.42	67.10	78.10	109.40

Table 9: CPU time (second) versus complexity of B for OMKL-B, and D for (Ada)Raker.

	Classification error			CPU time		
Algorithms/Datasets	Movement	Devices	Activity	Movement	Devices	Activity
RBF ($\sigma^2 = 0.1$)	43.1	6.67	0.46	2.76	2.13	4.42
RBF ($\sigma^2 = 1$)	41.3	28.1	5.21	2.79	2.04	4.42
RBF ($\sigma^2 = 10$)	40.3	31.8	41.1	2.62	2.09	4.43
POLY2	43.5	14.3	3.13	1.63	0.31	0.81
POLY3	43.6	25.2	2.39	4.75	0.57	1.79
LINEAR	43.8	47.4	4.30	1.26	0.21	0.60
AvgMKL	41.7	23.9	3.16	10.2	6.72	14.67
OMKL	38.2	16.0	0.60	10.27	7.07	14.46
AdaMKL	3.5	0.86	0.98	33.77	10.51	21.46
M-Forgetron ($B = 50$)	1.64	0.53	1.14	0.92	0.27	0.53
Raker ($D = 50$)	9.74	2.54	0.58	**0.40**	**0.12**	**0.21**
AdaRaker ($D = 50$)	**1.10**	**0.36**	**0.34**	6.76	1.73	3.56

Table 10: Classification error (%) and runtime (second) of different algorithms with the default stepsize $1/\sqrt{T}$ for RBF, OMKL and Raker, and with complexity $B = D = 50$.

remarkable especially in the Energy and Air pollution datasets. For Twitter (Large) dataset, the performance of AdaRaker does not improve as RFs increase from $D = 50$ to $D = 100$, which implies that $D = 50$ is enough to provide reliable kernel approximation in this dataset. Considering that AdaRaker is embedded with concurrent $\log t$ Raker instances at time t, its CPU time is relatively higher. However, one would expect a major reduction in the number of concurrent instances and thus markedly lower CPU time, if a larger basic interval size (instead of base number 2 in Figure 2) is incorporated in AdaRaker real implementation.

At this point, one may wonder how many RFs are enough for Raker and AdaRaker to guarantee the same online learning accuracy as that of OMKL-B with B samples. While this intriguing question has been recently studied in the batch setting with an answer of $D = \mathcal{O}(\sqrt{B})$ RFs (Rudi and Rosasco, 2017), its thorough treatment in the online setting constitutes our future research.

5.3. Real data tests for online classification

In this section, the performance of Raker and AdaRaker is tested on real datasets for the online classification task. We use the logistic loss as the learning objective function with the regularization parameter $\lambda = 0.005$ for all considered approaches except for the perceptron-

	Classification error		
Algorithms/Datasets	Movement	Devices	Activity
RBF ($\sigma^2 = 0.1$)	28.9	5.16	0.26
RBF ($\sigma^2 = 1$)	1.27	0.42	0.53
RBF ($\sigma^2 = 10$)	**1.10**	0.36	1.14
POLY2	8.19	1.7	0.56
POLY3	15.2	17.3	0.45
LINEAR	7.46	30.7	0.60
AvgMKL	1.69	2.26	0.48
OMKL	**1.10**	0.36	0.29
AdaMKL	3.50	0.86	1.00
M-Forgetron ($B = 50$)	1.64	0.53	1.14
Raker ($D = 50$)	**1.10**	**0.28**	**0.24**
AdaRaker ($D = 50$)	**1.10**	0.36	0.34

Table 11: Classification error (%) of different algorithms with the dataset-specific optimally chosen stepsizes for RBF, OMKL and Raker, and with complexity $B = D = 50$.

based Forgetron algorithm. Kernels and all other parameters such as the default stepsizes, are chosen as those in the regression task.

Datasets description. We test classification performance on the following datasets.

- **Movement** dataset consists of $T = 13,197$ temporal streams of received signal strength (RSS) measured between the nodes of a wireless sensor network, with each $\mathbf{x}_t \in \mathbb{R}^4$ comprising 4 anchor nodes (Bacciu et al., 2014). Data has been collected during user movements at the frequency of 8 Hz (8 samples per second). The RSS samples in the dataset have been rescaled to lie in $[-1, 1]$. The binary label y_t indicates whether the user's trajectory will lead to a change in the spatial context (here a room change) or not.
- **Electronic Device** dataset consists of $T = 3,600$ samples collected as part of a government sponsored study called 'Powering the Nation,' where the feature vectors $\mathbf{x}_t \in \mathbb{R}^{60}$ represent electricity readings from different households over 15 mins, sampled within a month (Lines et al., 2011). Binary label y_t represents the type of electronic devices used at the certain interval of time time: dishwasher or kettle.
- **Human Activity** dataset consists of $T = 7,352$ samples collected from a group of 30 volunteers wearing a smartphone (Samsung Galaxy S II) on their waist to monitor activities (Anguita et al., 2013). Feature vectors $\{\mathbf{x}_t \in \mathbb{R}^{30}\}$ here measure e.g., triaxial acceleration and angular velocity, while binary label y_t represents the activity during a certain period: walking or not walking.

Classification performance. The classification error $(1/T)\sum_{t=1}^{T} \max\{0, \text{sign}(-y_t\hat{y}_t)\}$ and the CPU time of each algorithm on these datasets are summarized in Table 10 when a default stepsize $1/\sqrt{T}$ is used for POLY, LINEAR, RBF, AvgMKL, OMKL and Raker. The budget of M-Forgetron is set at $B = 50$ samples, while Raker and AdaRaker adopt $D = 50$ RFs. As with the regression tests, it is evident that AdaRaker attains the highest

	OMKL				Raker				AdaRaker
Stepsize	$1/\sqrt{T}$	$1/\sqrt{t}$	$10/\sqrt{t}$	Tuned	$1/\sqrt{T}$	$1/\sqrt{t}$	$10/\sqrt{t}$	Tuned	/
Movement	38.2	39.5	22.3	1.10	12.1	8.60	1.79	1.10	**1.10**
Devices	16.0	13.2	6.06	0.36	2.54	2.04	0.53	**0.28**	0.36
Activity	0.60	0.53	0.50	0.29	0.58	0.52	0.54	**0.24**	0.34

Table 12: Classification error (%) versus different choices of stepsizes with $B = D = 50$.

	Classification error			CPU time		
Algorithms/ Datasets	Movement	Devices	Activity	Movement	Devices	Activity
M-Forgetron ($B = 10$)	1.60	0.53	1.14	0.92	0.26	0.53
M-Forgetron ($B = 50$)	1.64	0.53	1.14	0.92	0.27	0.53
M-Forgetron ($B = 100$)	1.42	0.53	1.14	0.94	0.29	0.53
Raker ($D = 10$)	26.3	8.37	3.26	**0.35**	**0.10**	**0.18**
Raker ($D = 50$)	9.74	2.54	0.58	0.40	0.12	0.21
Raker ($D = 100$)	4.65	1.53	0.43	0.46	0.15	0.26
AdaRaker ($D = 10$)	2.46	0.66	0.68	6.13	0.65	3.22
AdaRaker ($D = 50$)	**1.10**	**0.36**	**0.34**	6.76	1.73	3.56
AdaRaker ($D = 100$)	**1.10**	**0.36**	**0.34**	7.55	2.04	4.22

Table 13: Classification error (%) and CPU time (second) versus complexity.

classification accuracy and the Raker has the lowest CPU time among all competing algorithms. Without having to tune stepsizes, the performance of AdaMKL and M-Forgetron is also competitive in this case. To explore the best performance of each algorithm, the classification performance under manually tuned stepsizes is reported in Table 11, where each algorithm uses the best stepsize among $\{10^{-3}, 10^{-2}, \cdots, 10^{3}\}/\sqrt{T}$ for each dataset. With the optimally chosen stepsizes, the performance of all algorithms improves, and Raker even achieves slightly lower classification error than AdaRaker in some datasets. This is reasonable since compared to Raker with the offline tuned stepsize, AdaRaker will incur some error due to the online adaptation to several (possibly suboptimal) learning rates.

To corroborate the effectiveness of our algorithms in adapting to unknown dynamics (e.g., unknown time horizon T and variability), Table 12 compares the performance of AdaRaker with OMKL and Raker using default, diminishing and optimally tuned stepsizes. Similar to regression tests, the performance of Raker and OMKL is sensitive to the stepsize choice, while AdaRaker achieves the desired performance by combining learners with different learning rates. By simply averaging over all the kernels, AvgMKL outperforms single kernel methods in most cases, but performs much worse than OMKL and (Ada)Raker methods. Note that the Raker also achieves competitive classification accuracy when the constant stepsize $1/\sqrt{T}$ is used. Such a choice is however not always feasible in practice, since it requires knowledge of how many data samples will be available ahead of time.

Accuracy versus complexity. In this experiment, we test classification performance in terms of both classification error and CPU time for different levels of complexity; see Table 13. We use the number of support vectors B for M-Forgetron, and the number of RFs D for Raker and AdaRaker to represent different levels of complexity, and compare

their performance using the default stepsize. It is expected that CPU time increases as the complexity increases, and the classification error decreases as the complexity grows. For all three datasets, the AdaRaker achieves the lowest classification error, and the Raker outperforms the M-Forgetron while at the same time it is more efficient computationally.

6. Concluding Remarks

This paper dealt with kernel-based learning in environments with unknown dynamics that also include static or slow variations. Uniquely combining advances in random feature based function approximation with online learning from an ensemble of experts, a scalable online multi-kernel learning approach termed Raker, was developed for static environments based on a dictionary of kernels. Endowing Raker with capability of tracking time-varying optimal function estimators, AdaRaker was introduced as an ensemble version of Raker with variable learning rates. The key modules of the novel learning approaches are: i) the random features are for scalability, as they reduce the per-iteration complexity; ii) the preselected kernel dictionary is for flexibility, that is to broaden generalizability of a kernel-based learner over a larger function space; iii) the weighted combination of kernels adjusted online accounts for the reliability of learners; and, iv) the adoption of multiple learning rates is for improved adaptivity to changing environments with unknown dynamics.

Complementing the principled algorithmic design, the performance of Raker is rigorously established using static regret analysis. Furthermore, without a-priori knowledge of dynamics, it is proved that AdaRaker achieves sub-linear dynamic regret, provided that either the loss or the optimal learning function does not change on average. Experiments on synthetic and real datasets validate the effectiveness of the novel methods.

Acknowledgments

This work is supported in part by the National Science Foundation under Grant 1500713 and 1711471, and NIH 1R01GM104975-01. Yanning Shen is also supported by the Doctoral Dissertation Fellowship from the University of Minnesota.

Appendix A. Proof of Lemma 1

To prove Lemma 1, we introduce two intermediate lemmata as follows.

Lemma 3 *Under (as1), (as2), and $\hat{f}_p^*$ as in* (26) *with $\mathcal{F}_p := \{\hat{f}_p | \hat{f}_p(\mathbf{x}) = \boldsymbol{\theta}^\top \mathbf{z}_p(\mathbf{x}), \forall \boldsymbol{\theta} \in \mathbb{R}^{2D}\}$, let $\{\hat{f}_{p,t}(\mathbf{x}_t)\}$ denote the sequence of estimates generated by Raker with a pre-selected kernel κ_p. Then the following bound holds true w.p.1*

$$\sum_{t=1}^{T} \mathcal{L}_t(\hat{f}_{p,t}(\mathbf{x}_t)) - \sum_{t=1}^{T} \mathcal{L}_t(\hat{f}_p^*(\mathbf{x}_t)) \leq \frac{\|\boldsymbol{\theta}_p^*\|^2}{2\eta} + \frac{\eta L^2 T}{2} \tag{42}$$

where η is the learning rate, L is the Lipschitz constant in (as2), and $\boldsymbol{\theta}_p^*$ is the corresponding parameter (or weight) vector supporting the best estimator $\hat{f}_p^*(\mathbf{x}) = (\boldsymbol{\theta}_p^*)^\top \mathbf{z}_p(\mathbf{x})$.

Proof: Similar to the regret analysis of online gradient descent (Shalev-Shwartz, 2011), using (12) for any fixed $\boldsymbol{\theta}$, we find

$$\begin{aligned}\|\boldsymbol{\theta}_{p,t+1}-\boldsymbol{\theta}\|^2 =&\|\boldsymbol{\theta}_{p,t}-\eta\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\boldsymbol{\theta}\|^2 \\ =&\|\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}\|^2+\eta^2\|\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2-2\eta\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}).\end{aligned} \tag{43}$$

Meanwhile, the convexity of the loss under (as1) implies that

$$\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\le\nabla^\top\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)(\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}). \tag{44}$$

Plugging (44) into (43) and rearranging terms yields

$$\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)-\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\le\frac{\|\boldsymbol{\theta}_{p,t}-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{p,t+1}-\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta}{2}\|\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2. \tag{45}$$

Summing (45) over $t=1,\ldots,T$, with $\hat{f}_{p,t}(\mathbf{x}_t)=\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t)$, we arrive at

$$\begin{aligned}&\sum_{t=1}^T\Big(\mathcal{L}(\hat{f}_{p,t}(\mathbf{x}_t),y_t)-\mathcal{L}(\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\Big)\\ \le&\frac{\|\boldsymbol{\theta}_{p,1}-\boldsymbol{\theta}\|^2-\|\boldsymbol{\theta}_{p,T+1}-\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta}{2}\sum_{t=1}^T\|\nabla\mathcal{L}(\boldsymbol{\theta}_{p,t}^\top\mathbf{z}_p(\mathbf{x}_t),y_t)\|^2\\ \overset{(a)}{\le}&\frac{\|\boldsymbol{\theta}\|^2}{2\eta}+\frac{\eta L^2T}{2}\end{aligned} \tag{46}$$

where (a) uses the Lipschitz constant in (as2), the non-negativity of $\|\boldsymbol{\theta}_{p,T+1}-\boldsymbol{\theta}\|^2$, and the initial value $\boldsymbol{\theta}_{p,1}=\mathbf{0}$. The proof of Lemma 3 is then complete by choosing $\boldsymbol{\theta}=\boldsymbol{\theta}_p^*=\sum_{t=1}^T\alpha_{p,t}^*\mathbf{z}_p(\mathbf{x}_t)$ such that $\hat{f}_p^*(\mathbf{x}_t)=\boldsymbol{\theta}^\top\mathbf{z}_p(\mathbf{x}_t)$ in (46). ■

Lemma 3 establishes that the static regret of the Raker is upper bounded by some constants, which mainly depend on the stepsize in (19) and the time horizon T.

In addition, we will bound the difference between the loss of the solution obtained from Algorithm 1 and the loss of the best single kernel-based online learning algorithm. Specifically the following lemma holds:

Lemma 4 *Under (as1) and (as2), with $\{\hat{f}_{p,t}\}$ generated from Raker, it holds that*

$$\sum_{t=1}^T\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t(\hat{f}_{p,t}(\mathbf{x}_t))-\sum_{t=1}^T\mathcal{L}_t(\hat{f}_{p,t}(\mathbf{x}_t))\le\eta T+\frac{\ln P}{\eta} \tag{47}$$

where η is the learning rate in (21)*, and P is the number of kernels in the dictionary.*

Proof: Letting $W_t:=\sum_{p=1}^P w_{p,t}$, the weight recursion in (21) implies that

$$\begin{aligned}W_{t+1}&=\sum_{p=1}^P w_{p,t+1}=\sum_{p=1}^P w_{p,t}\exp\Big(-\eta\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)\Big)\\ &\le\sum_{p=1}^P w_{p,t}\left(1-\eta\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)+\eta^2\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)^2\right)\end{aligned} \tag{48}$$

where the last inequality holds because $\exp(-\eta x)\le 1-\eta x+\eta^2x^2$, for $|\eta|\le 1$. Furthermore, substituting $\bar{w}_{p,t}:=w_{p,t}/\sum_{p=1}^P w_{p,t}=w_{p,t}/W_t$ into (48), it follows that

$$\begin{aligned}W_{t+1}&\le\sum_{p=1}^P W_t\bar{w}_{p,t}\left(1-\eta\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)+\eta^2\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)^2\right)\\ &=W_t\left(1-\eta\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)+\eta^2\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)^2\right).\end{aligned} \tag{49}$$

Using $1+x\le e^x$, $\forall x$, (49) leads to

$$W_{t+1}\le W_t\exp\left(-\eta\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)+\eta^2\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)^2\right). \tag{50}$$

Telescoping (50) from $t=1$ to T, we have ($W_1=1$)

$$W_{T+1}\le\exp\left(-\eta\sum_{t=1}^T\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)+\eta^2\sum_{t=1}^T\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)^2\right). \tag{51}$$

On the other hand, for any p, the following holds true

$$\begin{aligned}W_{T+1}\ge w_{p,T+1}&=w_{p,1}\prod_{t=1}^T\exp(-\eta\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big))\\ &=w_{p,1}\exp\left(-\eta\sum_{t=1}^T\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)\right).\end{aligned} \tag{52}$$

Combining (51) with (52), we arrive at

$$\begin{aligned}&\exp\left(-\eta\sum_{t=1}^T\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)+\eta^2\sum_{t=1}^T\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)^2\right)\\ \ge& w_{p,1}\exp\left(-\eta\sum_{t=1}^T\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)\right).\end{aligned} \tag{53}$$

Taking the logarithm on both sides of (53), we find that (cf. $w_{p,1}=1/P$)

$$-\eta\sum_{t=1}^T\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)+\eta^2\sum_{t=1}^T\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)^2\ge-\eta\sum_{t=1}^T\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)-\ln P \tag{54}$$

which leads to

$$\sum_{t=1}^T\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)\le\sum_{t=1}^T\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)+\eta\sum_{t=1}^T\sum_{p=1}^P\bar{w}_{p,t}\mathcal{L}_t\Big(\hat{f}_{p,t}(\mathbf{x}_t)\Big)^2+\frac{\ln P}{\eta} \tag{55}$$

and the proof is complete since $\mathcal{L}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right)^2 \leq 1$ and $\sum_{p=1}^P \bar{w}_{p,t} = 1$. ■

Moreover, since $\mathcal{L}_t(\cdot)$ is convex under (as1), Jensen's inequality implies that

$$\mathcal{L}_t\Big(\sum_{p=1}^P \bar{w}_{p,t}\hat{f}_{p,t}(\mathbf{x}_t)\Big) \leq \sum_{p=1}^P \bar{w}_{p,t}\mathcal{L}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right). \tag{56}$$

Plugging (56) into (47) in Lemma 4, we arrive at

$$\begin{aligned}\sum_{t=1}^T \mathcal{L}_t\Big(\sum_{p=1}^P \bar{w}_{p,t}\hat{f}_{p,t}(\mathbf{x}_t)\Big) &\leq \sum_{t=1}^T \mathcal{L}_t\left(\hat{f}_{p,t}(\mathbf{x}_t)\right) + \eta T + \frac{\ln P}{\eta} \\ &\overset{(a)}{\leq} \sum_{t=1}^T \mathcal{L}_t\left(\hat{f}_p^*(\mathbf{x}_t)\right) + \frac{\ln P}{\eta} + \frac{\|\boldsymbol{\theta}_p^*\|^2}{2\eta} + \frac{\eta L^2 T}{2} + \eta T \end{aligned} \tag{57}$$

where (a) follows because $\boldsymbol{\theta}_p^*$ is the optimal solution for any given kernel κ_p. This proves the claim in Lemma 1.

Appendix B. Proof of Theorem 1

To derive the performance bound relative to the best function estimator $f^*(\mathbf{x}_t)$ in the RKHS, the key step is to bound the error of approximation. For a given shift-invariant κ_p, the maximum point-wise error of the RF kernel approximant is uniformly bounded with probability at least $1 - 2^8\left(\frac{\sigma_p}{\epsilon}\right)^2 \exp\left(\frac{-D\epsilon^2}{4d+8}\right)$, by (Rahimi and Recht, 2007)

$$\sup_{\mathbf{x}_i,\mathbf{x}_j\in\mathcal{X}} \left|\mathbf{z}_p^\top(\mathbf{x}_i)\mathbf{z}_p(\mathbf{x}_j) - \kappa_p(\mathbf{x}_i,\mathbf{x}_j)\right| < \epsilon \tag{58}$$

where $\epsilon > 0$ is a given constant, D the number of features, while d represents the dimension of $\mathbf{x}$, and $\sigma_p^2 := \mathbb{E}_p[\|\mathbf{v}\|^2]$ is the second-order moments of the RF vector norm. Henceforth, for the optimal function estimator (26) in $\mathcal{H}_p$ denoted by $f_p^*(\mathbf{x}) := \sum_{t=1}^T \alpha_{p,t}^*\kappa_p(\mathbf{x},\mathbf{x}_t)$, and its RF-based approximant $\check{f}_p^* := \sum_{t=1}^T \alpha_{p,t}^*\mathbf{z}_p^\top(\mathbf{x})\mathbf{z}_p(\mathbf{x}_t) \in \mathcal{F}_p$, we have

$$\begin{aligned}\left|\sum_{t=1}^T \mathcal{L}_t\left(\check{f}_p^*(\mathbf{x}_t)\right) - \sum_{t=1}^T \mathcal{L}_t\left(f_p^*(\mathbf{x}_t)\right)\right| &\overset{(a)}{\leq} \sum_{t=1}^T \left|\mathcal{L}_t\left(\check{f}_p^*(\mathbf{x}_t)\right) - \mathcal{L}_t(f_p^*(\mathbf{x}_t))\right| \\ &\overset{(b)}{\leq} \sum_{t=1}^T L\left|\sum_{t'=1}^T \alpha_{p,t'}^*\mathbf{z}_p^\top(\mathbf{x}_{t'})\mathbf{z}_p(\mathbf{x}_t) - \sum_{t'=1}^T \alpha_{p,t'}^*\kappa_p(\mathbf{x}_{t'},\mathbf{x}_t)\right| \\ &\overset{(c)}{\leq} \sum_{t=1}^T L\sum_{t'=1}^T |\alpha_{p,t'}^*|\left|\mathbf{z}_p^\top(\mathbf{x}_{t'})\mathbf{z}_p(\mathbf{x}_t) - \kappa_p(\mathbf{x}_{t'},\mathbf{x}_t)\right| \end{aligned} \tag{59}$$

where (a) follows from the triangle inequality; (b) uses the Lipschitz continuity of the loss, and (c) is due to the Cauchy-Schwarz inequality. Combining with (58), yields

$$\left|\sum_{t=1}^T \mathcal{L}_t(\check{f}_p^*(\mathbf{x}_t)) - \sum_{t=1}^T \mathcal{L}_t(f_p^*(\mathbf{x}_t))\right| \leq \sum_{t=1}^T L\epsilon\sum_{t'=1}^T |\alpha_{p,t'}^*| \leq \epsilon LTC, \text{ w.h.p.} \tag{60}$$

where the equality follows from $C := \max_p \sum_{t=1}^T |\alpha_{p,t}^*|$. Under the kernel bounds in (as3), the uniform convergence in (58) implies that $\sup_{\mathbf{x}_t,\mathbf{x}_{t'}\in\mathcal{X}} \mathbf{z}_p^\top(\mathbf{x}_t)\mathbf{z}_p(\mathbf{x}_{t'}) \leq 1+\epsilon$, w.h.p., which in turn leads to

$$\left\|\boldsymbol{\theta}_p^*\right\|^2 := \left\|\sum_{t=1}^T \alpha_{p,t}^*\mathbf{z}_p(\mathbf{x}_t)\right\|^2 = \left|\sum_{t=1}^T\sum_{t'=1}^T \alpha_{p,t}^*\alpha_{p,t'}^*\mathbf{z}_p^\top(\mathbf{x}_t)\mathbf{z}_p(\mathbf{x}_{t'})\right| \leq (1+\epsilon)C^2 \tag{61}$$

where we again used the definition of C.

Lemma 1 together with (60) and (61) leads to the regret of the proposed Raker algorithm relative to the best static function in $\mathcal{H}_p$, that is given by

$$\begin{aligned}&\sum_{t=1}^T \mathcal{L}_t\Big(\sum_{p=1}^P w_{p,t}\hat{f}_{p,t}(\mathbf{x}_t)\Big) - \sum_{t=1}^T \mathcal{L}_t(f_p^*(\mathbf{x}_t)) \\ =&\sum_{t=1}^T \mathcal{L}_t\Big(\sum_{p=1}^P w_{p,t}\hat{f}_{p,t}(\mathbf{x}_t)\Big) - \sum_{t=1}^T \mathcal{L}_t\left(\check{f}_p^*(\mathbf{x}_t)\right) + \sum_{t=1}^T \mathcal{L}_t\left(\check{f}_p^*(\mathbf{x}_t)\right) - \sum_{t=1}^T \mathcal{L}_t(f_p^*(\mathbf{x}_t)) \\ \leq&\frac{\ln P}{\eta} + \frac{\eta L^2 T}{2} + \eta T + \frac{(1+\epsilon)C^2}{2\eta} + \epsilon LTC, \text{ w.h.p.} \end{aligned} \tag{62}$$

which completes the proof of Theorem 1.

Appendix C. Proof of Lemma 2

Using Theorem 1 with $\eta = \epsilon = \mathcal{O}(1/\sqrt{T})$, it holds w.h.p. that

$$\sum_{t=1}^T \mathcal{L}_t\Big(\sum_{p=1}^P w_{p,t}\hat{f}_{p,t}(\mathbf{x}_t)\Big) - \sum_{t=1}^T \mathcal{L}_t(f_{p^*}^*(\mathbf{x}_t)) \leq \left(\ln P + \frac{C^2}{2} + \frac{L^2}{2} + LC\right)\sqrt{T} := c_0\sqrt{T} \tag{63}$$

where $p^* := \arg\min_{p\in\mathcal{P}} \sum_{t=1}^T \mathcal{L}_t\left(\hat{f}_p^*(\mathbf{x}_t)\right)$. At the end of interval I, we then deduce that the static regret of the Raker learner $\mathcal{A}_I$ is (cf. (36))

$$\mathrm{Reg}_{\mathcal{A}_I}^{\mathrm{s}}(|I|) = \sum_{t\in I} \mathcal{L}_t\left(\hat{f}_t^{(I)}(\mathbf{x}_t)\right) - \sum_{t\in I} \mathcal{L}_t(f_I^*(\mathbf{x}_t)) \leq c_0\sqrt{|I|}, \text{ w.h.p.} \tag{64}$$

where $\hat{f}_t^{(I)}(\mathbf{x}_t)$ is defined in (33), and $f_I^* \in \arg\min_{f\in\bigcup_{p\in\mathcal{P}}\mathcal{H}_p} \sum_{t\in I} \mathcal{L}_t(f(\mathbf{x}_t))$. To this end, we sketch the main steps leading to Lemma 2 as follows.

For every interval I, the static regret of the AdaRaker can be decomposed as

$$\begin{aligned}\mathrm{Reg}_{\mathrm{AdaRaker}}^{\mathrm{s}}(|I|) &= \sum_{t\in I} \mathcal{L}_t\left(\hat{f}_t(\mathbf{x}_t)\right) - \sum_{t\in I} \mathcal{L}_t\left(\hat{f}_t^{(I)}(\mathbf{x}_t)\right) + \sum_{t\in I} \mathcal{L}_t\left(\hat{f}_t^{(I)}(\mathbf{x}_t)\right) - \sum_{t\in I} \mathcal{L}_t(f_I^*(\mathbf{x}_t)) \\ &:= \mathcal{R}_1 + \mathcal{R}_2 \end{aligned} \tag{65}$$

where the first two sums in (65) represented by $\mathcal{R}_1$ capture the regret of the Ada-Raker learner $\mathcal{A}$ relative to the Raker learner $\mathcal{A}_I$; while the last two sums in (65) forming $\mathcal{R}_2$ denote the static regret of $\mathcal{A}_I$ on this interval. Notice that $\mathcal{R}_2$ directly follows from (64), while $\mathcal{R}_1$

can be bounded following the same steps in Lemma 4. Different from the kernel selections however, the crux is that the number of Raker learners (experts) $|\mathcal{I}(t)|$ is time-varying.

A tight bound can be derived via the *Sleeping Experts* reformulation of (Luo and Schapire, 2015; Daniely et al., 2015), where an expert that has never appeared is thought of as being *asleep* for all previous rounds. For a looser bound, we assume the experts (instances $\{\mathcal{A}_I\}$) ever appeared until t are all active; that is, the total number of experts is upper bounded by $t\log t$, since at most $\log t$ experts are run during time t. Using (48)-(55), we have that

$$\mathcal{R}_1 \leq \eta^{(I)}|I| + \frac{\ln(t\log t)}{\eta^{(I)}} = \sqrt{|I|}\,(1+\ln t+\ln(\log t)) \leq \sqrt{|I|}\,(1+2\ln t) \tag{66}$$

where $\eta^{(I)} := 1/\sqrt{|I|}$, and $\ln(\log t) \leq \ln(t)$. With (64), for any interval $I \in \mathcal{I}$, we have

$$\mathrm{Reg}^{\mathrm{s}}_{\mathrm{AdaRaker}}(|I|) = \sqrt{|I|}\,(1+c_0+2\ln t) \leq \sqrt{|I|}\,(1+c_0+2\ln T)\,. \tag{67}$$

Since the static regret bound (65) holds only at the end of such interval, the bound (67) only holds for those intervals (collected in $\mathcal{I}$) (re)initializing Raker instance $\mathcal{A}_I$.

The next step is to show that (67) holds for any interval $I \subseteq \mathcal{T}$, possibly $I \notin \mathcal{I}$. This is possible whenever the interval set $\mathcal{I}$ is properly designed, e.g., the interval partition given in Section 4.1. For any interval I, define the set of subintervals covered by I as $\mathcal{I}\|I := \{I' \subseteq I, I' \in \mathcal{I}\}$. As argued in (Daniely et al., 2015, Lemma 5), interval I can be partitioned into two sequences of *non-overlapping* but *consecutive* intervals, given by $\{I_{-m},\ldots,I_0\} \subseteq \mathcal{I}\|I$ and $\{I_1,\ldots,I_n\} \subseteq \mathcal{I}\|I$, the lengths of which satisfy $|I_{k+1}|/|I_k| \leq 1/2$, $\forall k \in [1,n-1]$ and $|I_k|/|I_{k+1}| \leq 1/2$, $\forall k \in [-m,-1]$. Therefore, we have (using $\sum_{k=1}^{\infty}\sqrt{2^{-k}T_0} \leq 4\sqrt{T_0}$)

$$\mathrm{Reg}^{\mathrm{s}}_{\mathrm{AdaRaker}}(|I|) = \sum_{k=1}^{n-1}\mathrm{Reg}^{\mathrm{s}}_{\mathrm{Ada}}(|I_k|) + \sum_{k=-m}^{-1}\mathrm{Reg}^{\mathrm{s}}_{\mathrm{Ada}}(|I_k|) \leq C_0\sqrt{|I|} + C_1\ln T\sqrt{|I|} \tag{68}$$

where the inequality follows from (67) with $|I|$ replaced by $|I_k|$ ($\leq |I|$), and C_0, C_1 are constants depending on c_0 defined in (63). This completes the proof of Lemma 2.

Appendix D. Proof of Theorem 2

To start, the dynamic regret in (34) can be decomposed as

$$\mathrm{Reg}^{\mathrm{d}}_{\mathcal{A}}(T) := \sum_{t=1}^{T}\mathcal{L}_t(\hat{f}_t(\mathbf{x}_t)) - \sum_{t=1}^{T}\mathcal{L}_t(f^*(\mathbf{x}_t)) + \sum_{t=1}^{T}\mathcal{L}_t(f^*(\mathbf{x}_t)) - \sum_{t=1}^{T}\mathcal{L}_t(f_t^*(\mathbf{x}_t)) \tag{69}$$

where $f^*(\cdot)$ is the best fixed function in (26), and $f_t^*(\cdot)$ is the best dynamic function in (35), both of which belong to the union of spaces $\bigcup_{p\in\mathcal{P}}\mathcal{H}_p$. In (69), the first difference of sums is the static regret of AdaRaker, while the second difference of sums is the relative loss between the best fixed function and the best dynamic solution in the common space.

Intuitively, if the time horizon T is large, then the average static regret will become small, but the gap between the two benchmarks is large. With the insights gained from (Besbes et al., 2015; Luo et al., 2017), T essentially trades off the values of two terms. Thus,

splitting $\mathcal{T}$ into sub-horizons $\{\mathcal{T}_s\}$, $s = 1,\ldots,\lfloor T/\Delta T\rfloor$, each having length ΔT, the dynamic regret of AdaRaker can be bounded by

$$\begin{aligned}\mathrm{Reg}^{\mathrm{d}}_{\mathrm{AdaRaker}}(T) &= \sum_{s=1}^{\lfloor T/\Delta T\rfloor}\sum_{t\in\mathcal{T}_s}\Big(\mathcal{L}_t(\hat{f}_t(\mathbf{x}_t)) - \mathcal{L}_t(f^*_{\mathcal{T}_s}(\mathbf{x}_t))\Big) + \sum_{s=1}^{\lfloor T/\Delta T\rfloor}\sum_{t\in\mathcal{T}_s}\big(\mathcal{L}_t(f^*_{\mathcal{T}_s}(\mathbf{x}_t)) - \mathcal{L}_t(f_t^*(\mathbf{x}_t))\big)\\ &:= \sum_{s=1}^{\lfloor T/\Delta T\rfloor}\mathcal{R}_1 + \sum_{s=1}^{\lfloor T/\Delta T\rfloor}\mathcal{R}_2\end{aligned} \tag{70}$$

where the first sum over $\mathcal{T}_s$ we define as $\mathcal{R}_1$ can be bounded under AdaRaker from Lemma 2, while the second sum over $\mathcal{T}_s$ that we define as $\mathcal{R}_2$ depends on the variability of the environments $\mathcal{V}(\{\mathcal{L}_t\})$, can be bounded by (Besbes et al., 2015, Prop. 2)

$$\mathcal{R}_2 \leq 2\Delta T\mathcal{V}(\{\mathcal{L}_t\}_{t\in\mathcal{T}_s}). \tag{71}$$

Together with Lemma 2, it follows that

$$\begin{aligned}\mathrm{Reg}^{\mathrm{d}}_{\mathrm{AdaRaker}}(T) &\leq \sum_{s=1}^{\lfloor T/\Delta T\rfloor}\Big((C_0+C_1\ln T)\sqrt{\Delta T} + 2\Delta T\mathcal{V}(\{\mathcal{L}_t\}_{t\in\mathcal{T}_s})\Big)\\ &= (C_0+C_1\ln T)\frac{T}{\sqrt{|\Delta T|}} + 2|\Delta T|\mathcal{V}(\{\mathcal{L}_t\}_{t=1}^{T}),\ \text{w.h.p.}\end{aligned} \tag{72}$$

Since (37) in Lemma 2 holds for any interval $\Delta T \subseteq \mathcal{T}$, after selecting ΔT so that $|\Delta T| = \big(T/\mathcal{V}(\{\mathcal{L}_t\}_{t=1}^{T})\big)^{\frac{2}{3}}$, we arrive at

$$\mathrm{Reg}^{\mathrm{d}}_{\mathrm{AdaRaker}}(T) \leq (C_0+C_1\ln T)T^{\frac{2}{3}}\mathcal{V}^{\frac{1}{3}}(\{\mathcal{L}_t\}_{t=1}^{T}) + 2T^{\frac{2}{3}}\mathcal{V}^{\frac{1}{3}}(\{\mathcal{L}_t\}_{t=1}^{T}),\ \text{w.h.p.} \tag{73}$$

which completes the proof of Theorem 2.

Appendix E. Proof of Theorem 3

Suppose that the m-switching dynamic solution $\{\check{f}_t^*\}$ changes at slots $t_1 = 1,\ldots,t_m$, and define the m sub-intervals that partition $\mathcal{T} := \{1,\ldots,T\}$ as $\mathcal{T}_1 := [1,t_2-1]$, $\mathcal{T}_2 := [t_2,t_3-1],\ldots$, and $\mathcal{T}_m := [t_m,T]$. To use the bound in Lemma 2, we decompose the regret of AdaRaker relative to the m-switching dynamic solution $\{\check{f}_t^*\}$ by

$$\begin{aligned}\mathrm{Reg}^{m}_{\mathrm{AdaRaker}}(T) &\overset{(a)}{=} \sum_{s=1}^{m}\sum_{t\in\mathcal{T}_s}\Big(\mathcal{L}_t(\hat{f}_t(\mathbf{x}_t)) - \mathcal{L}_t(\check{f}^*_{t_s}(\mathbf{x}_t))\Big)\\ &\overset{(b)}{\leq} \sum_{s=1}^{m}\sum_{t\in\mathcal{T}_s}\Big(\mathcal{L}_t(\hat{f}_t(\mathbf{x}_t)) - \mathcal{L}_t(f^*_{\mathcal{T}_s}(\mathbf{x}_t))\Big)\end{aligned} \tag{74}$$

where (a) holds because the definition of $\{\check{f}_t^*\}$ in (40) implies that $\check{f}_t^* = \check{f}^*_{t_s}$, $\forall t \in \mathcal{T}_s$, and (b) because the best fixed function is given by $f^*_{\mathcal{T}_s} \in \arg\min_{f\in f\in\bigcup_{p\in\mathcal{P}}\mathcal{H}_p}\sum_{t\in\mathcal{T}_s}\mathcal{L}_t(f(\mathbf{x}_t))$. Therefore, using the regret bound of Lemma 2 in (37), we have

$$\mathrm{Reg}^{m}_{\mathrm{AdaRaker}}(T) \leq \sum_{s=1}^{m}\mathrm{Reg}^{\mathrm{s}}_{\mathcal{A}}(|\mathcal{T}_s|) \leq (C_0+C_1\ln T)\sum_{s=1}^{m}\sqrt{|\mathcal{T}_s|}. \tag{75}$$

Holder's inequality further implies that

$$\begin{aligned}\mathrm{Reg}^m_{\mathrm{AdaRaker}}(T) &\leq (C_0 + C_1 \ln T)\left(\sum_{s=1}^{m}(1)^2\right)^{\frac{1}{2}}\left(\sum_{s=1}^{m}\left(\sqrt{|\mathcal{T}_s|}\right)^2\right)^{\frac{1}{2}} \\ &\leq (C_0 + C_1 \ln T)\sqrt{Tm}, \text{ w.h.p.} \end{aligned} \tag{76}$$

which completes the proof of Theorem 3.

References

Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *Euro. Symp. on Artificial Neural Netw., Comp. Intell. and Mach. Learn.*, Bruges, Belgium, Apr. 2013.

Davide Bacciu, Paolo Barsocchi, Stefano Chessa, Claudio Gallicchio, and Alessio Micheli. An experimental characterization of reservoir computing in ambient assisted living applications. *Neural Computing and Applications*, 24(6):1451–1464, May 2014.

Francis R. Bach. Consistency of the group lasso and multiple kernel learning. *J. Machine Learning Res.*, 9:1179–1225, Jun. 2008.

Juan Andres Bazerque and Georgios B. Giannakis. Nonparametric basis pursuit via sparse kernel-based learning: A unifying view with advances in blind methods. *IEEE Signal Processing Magazine*, 30(4):112–125, Jul. 2013.

Omar Besbes, Yonatan Gur, and Assaf Zeevi. Non-stationary stochastic optimization. *Operations Research*, 63(5):1227–1244, Sep. 2015.

Pantelis Bouboulis, Symeon Chouvardas, and Sergios Theodoridis. Online distributed learning over networks in RKH spaces using random Fourier features. *IEEE Trans. Sig. Proc.*, to appear, 2018.

Luis M. Candanedo, Véronique Feldheim, and Dominique Deramaix. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and Buildings*, 140:81–97, 2017.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, Cambridge, United Kingdom, 2006.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. ℓ_2-regularization for learning kernels. In *Proc. Conf. on Uncertainty in Artificial Intelligence*, pages 109–116, Montreal, Canada, Jun. 2009.

Corinna Cortes, Mehryar Mohri, and Ameet Talwalkar. On the impact of kernel approximation on learning accuracy. In *Proc. Intl. Conf. on Artificial Intelligence and Statistics*, pages 113–120, Sardinia, Italy, May 2010.

Bo Dai, Bo Xie, Niao He, Yingyu Liang, Anant Raj, Maria-Florina F. Balcan, and Le Song. Scalable kernel methods via doubly stochastic gradients. In *Proc. Advances in Neural Info. Process. Syst.*, pages 3041–3049, Montreal, Canada, Dec. 2014.

Bo Dai, Niao He, Yunpeng Pan, Byron Boots, and Le Song. Learning from conditional distributions via dual embeddings. In *Proc. Intl. Conf. on Artificial Intelligence and Statistics*, pages 1458–1467, Fort Lauderdale, FL, Apr. 2017.

Amit Daniely, Alon Gonen, and Shai Shalev-Shwartz. Strongly adaptive online learning. In *Proc. Intl. Conf. on Machine Learning*, pages 1405–1411, Lille, France, Jun. 2015.

Saverio De Vito, Ettore Massera, M Piga, L Martinotto, and G Di Francia. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sensors and Actuators B: Chemical*, 129(2):750–757, Feb. 2008.

Ofer Dekel, Shai Shalev-Shwartz, and Yoram Singer. The forgetron: A kernel-based perceptron on a budget. *SIAM J. Computing*, 37(5):1342–1372, Jan. 2008.

Yi Ding, Chenghao Liu, Peilin Zhao, and Steven CH Hoi. Large scale kernel methods for online auc maximization. In *Proc. IEEE Intl. Conf. Data Mining*, pages 91–100, New Orleans, LO, November 2017.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *J. Machine Learning Res.*, 12:2211–2268, Jul. 2011.

Elad Hazan. Introduction to online convex optimization. *Found. and Trends in Mach. Learn.*, 2(3-4):157–325, 2016.

Steven CH. Hoi, Rong Jin, Peilin Zhao, and Tianbao Yang. Online multiple kernel classification. *Machine Learning*, 90(2):289–316, Feb. 2013.

Ali Jadbabaie, Alexander Rakhlin, Shahin Shahrampour, and Karthik Sridharan. Online optimization: Competing with dynamic comparators. In *Intl. Conf. Artificial Intell. and Stat.*, San Diego, CA, May 2015.

Rong Jin, Steven CH. Hoi, and Tianbao Yang. Online multiple kernel learning: Algorithms and mistake bounds. In *Proc. Intl. Conf. on Algorithmic Learning Theory*, pages 390–404, Canberra, Australia, Oct. 2010.

François Kawala, Ahlame Douzal-Chouakria, Eric Gaussier, and Eustache Dimert. Prédictions d'activité dans les réseaux sociaux en ligne. In *4ième Conférence sur les Modèles et l'Analyse des Réseaux: Approches Mathématiques et Informatiques*, 2013.

Jyrki Kivinen, Alexander J. Smola, and Robert C. Williamson. Online learning with kernels. *IEEE Trans. Sig. Proc.*, 52(8):2165–2176, Aug. 2004.

Gert R.G. Lanckriet, Nello Cristianini, Peter Bartlett, Laurent El Ghaoui, and Michael I. Jordan. Learning the kernel matrix with semidefinite programming. *J. Machine Learning Res.*, 5:27–72, Jan. 2004.

Moshe Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Jason Lines, Anthony Bagnall, Patrick Caiger-Smith, and Simon Anderson. Classification of household devices by electricity usage profiles. In *Intl. Conf. on Intelligent Data Engineering and Automated Learning*, pages 403–412, Norwich, United Kingdom, Sept. 2011.

Jing Lu, Steven CH. Hoi, Jialei Wang, Peilin Zhao, and Zhi-Yong Liu. Large scale online kernel learning. *J. Machine Learning Res.*, 17(47):1–43, Apr. 2016.

Jing Lu, Doyen Sahoo, Peilin Zhao, and Steven CH Hoi. Sparse passive-aggressive learning for bounded online kernel methods. *ACM Trans. Intell. Syst. Tech.*, 9(4):45, February 2018.

Haipeng Luo and Robert E. Schapire. Achieving all with no parameters: Adanormalhedge. In *Proc. Conf. on Learning Theory*, pages 1286–1304, Lille, France, Jul. 2015.

Haipeng Luo, Alekh Agarwal, and John Langford. Efficient contextual bandits in non-stationary worlds. *arXiv preprint:1708.01799*, Aug. 2017.

Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. Identifying suspicious URLs: An application of large-scale online learning. In *Proc. Intl. Conf. Mach. Learn.*, pages 681–688, Montreal, Canada, Jun. 2009.

Charles A. Micchelli and Massimiliano Pontil. Learning the kernel function via regularization. *J. Machine Learning Res.*, 6:1099–1125, Jul. 2005.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Proc. Advances in Neural Info. Process. Syst.*, pages 1177–1184, Vancouver, Canada, Dec. 2007.

Alain Rakotomamonjy, Francis R. Bach, Stéphane Canu, and Yves Grandvalet. SimpleMKL. *J. Machine Learning Res.*, 9:2491–2521, Nov. 2008.

Cédric Richard, José Carlos M. Bermudez, and Paul Honeine. Online prediction of time series data with kernels. *IEEE Trans. Sig. Proc.*, 57(3):1058–1067, Mar. 2009.

Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *Proc. Advances in Neural Info. Process. Syst.*, pages 3218–3228, Long Beach, CA, Dec. 2017.

Doyen Sahoo, Steven CH. Hoi, and Bin Li. Online multiple kernel regression. In *Proc. Intl. Conf. Knowledge Discovery and Data Mining*, pages 293–302, New York, NY, Aug. 2014.

Doyen Sahoo, Steven CH. Hoi, and Peilin Zhao. Cost sensitive online multiple kernel classification. In *Proc. Asian Conf. Machine Learning*, pages 65–80, Hamilton, New Zealand, Nov. 2016.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Found. and Trends in Mach. Learn.*, 4(2):107–194, 2011.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, United Kingdom, 2004.

Fateme Sheikholeslami, Dimitris Berberidis, and Georgios B. Giannakis. Large-scale kernel-based feature extraction via budgeted nonlinear subspace tracking. *IEEE Trans. Sig. Proc.*, 66(8):1967–1981, April 2018.

Yanning Shen and Georgios B Giannakis. Online identification of directional graph topologies capturing dynamic and nonlinear dependencies. In *Proc. of IEEE Data Science Workshop*, pages 195–199, Lausanne, Switzerland, June 2018.

Yanning Shen, Brian Baingana, and Georgios B. Giannakis. Nonlinear structural vector autoregressive models for inferring effective brain network connectivity. Oct. 2016. URL `https://arxiv.org/abs/1610.06551`.

Yanning Shen, Brian Baingana, and Georgios B. Giannakis. Kernel-based structural equation models for topology identification of directed networks. *IEEE Trans. Sig. Proc.*, 65 (10):2503–2516, May 2017.

Yanning Shen, Tianyi Chen, and Georgios B. Giannakis. Online ensemble multi-kernel learning adaptive to non-stationary and adversarial environments. In *Proc. of Intl. Conf. on Artificial Intelligence and Statistics*, Lanzarote, Canary Islands, April 2018.

Vladimir G. Vovk. A game of prediction with expert advice. In *Proc. Annual Conf. Computational Learning Theory*, pages 51–60, Santa Cruz, CA, Jul. 1995.

Grace Wahba. *Spline Models for Observational Data*. SIAM, Philadelphia, PA, 1990.

Zhuang Wang, Koby Crammer, and Slobodan Vucetic. Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale svm training. *J. Machine Learning Res.*, 13:3103–3131, Oct. 2012.

Christopher K.I. Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Proc. Advances in Neural Info. Process. Syst.*, pages 682–688, Vancouver, Canada, Dec. 2001.

Felix Yu, Ananda Theertha Suresh, Krzysztof Choromanski, Daniel Holtmann-Rice, and Sanjiv Kumar. Orthogonal random features. In *Proc. Advances in Neural Info. Process. Syst.*, pages 1975–1983, Barcelona, Spain, Dec. 2016.

Journal of Machine Learning Research 20 (2019) 1-38 Submitted 1/18; Revised 9/18; Published 2/19

Determining the Number of Latent Factors in Statistical Multi-Relational Learning

Chengchun Shi cshi4@ncsu.edu
Wenbin Lu lu@stat.ncsu.edu
Rui Song rsong@ncsu.edu
Department of Statistics
North Carolina State University
Raleigh, NC 27695, USA

Editor: Hui Zou

Abstract

Statistical relational learning is primarily concerned with learning and inferring relationships between entities in large-scale knowledge graphs. Nickel et al. (2011) proposed a RESCAL tensor factorization model for statistical relational learning, which achieves better or at least comparable results on common benchmark data sets when compared to other state-of-the-art methods. Given a positive integer s, RESCAL computes an s-dimensional latent vector for each entity. The latent factors can be further used for solving relational learning tasks, such as collective classification, collective entity resolution and link-based clustering.

The focus of this paper is to determine the number of latent factors in the RESCAL model. Due to the structure of the RESCAL model, its log-likelihood function is not concave. As a result, the corresponding maximum likelihood estimators (MLEs) may not be consistent. Nonetheless, we design a specific pseudometric, prove the consistency of the MLEs under this pseudometric and establish its rate of convergence. Based on these results, we propose a general class of information criteria and prove their model selection consistencies when the number of relations is either bounded or diverges at a proper rate of the number of entities. Simulations and real data examples show that our proposed information criteria have good finite sample properties.

Keywords: Information criteria; Knowledge graph; Model selection consistency; RESCAL model; Statistical relational learning; Tensor factorization.

1. Introduction

Relational data is becoming ubiquitous in artificial intelligence and social network analysis. These data sets are in the form of graphs, with nodes and edges representing entities and relationships, respectively. Recently, a number of companies have developed and released their knowledge graphs, including the Google Knowledge Graph, Microsoft Bing's Satori Knowledge Base, Yandex's Object Answer, the Linkedln Knowledge Graph, etc. These knowledge graphs are graph-structured knowledge bases that store factual information as relationships between entities. They are created via the automatic extraction of semantic relationships from semi-structured or unstructured text (see Section II.C in Nickel et al., 2016). The data may be incomplete, noisy and contain false information. It is therefore of great importance to infer the existence of a particular relationship to improve the quality of these extracted information.

Statistical relational learning is primarily concerned with learning from relational data sets, and solving tasks such as predicting whether two entities are related (link prediction), identifying equivalent entities (entity resolution), and grouping similar entities based on their relationships (link-based clustering). Statistical relational models can be roughly divided into three categories: the relational graphical models, the latent class models and the tensor factorization models. Relational graphical models include probabilistic relational models (Getoor and Mihalkova, 2011) and Markov logic networks (MLN, Richardson and Domingos, 2006). These models are constructed via Bayesian or Markov networks. In latent class models, each entity is assigned to one of the latent classes and the probability of a relationship between entities depends on their corresponding classes. Two important examples include the stochastic block model (SBM, Nowicki and Snijders, 2001) and the infinite relational model (IRM, Kemp et al., 2006). IRM can be viewed as a nonparametric extension of SBM where the total number of clusters is not prespecified. Both models have received considerable attentions in the statistics and machine learning literature for community detection in networks.

Tensors are multidimensional arrays. Tensor factorization methods such as CANDECOMP/PARAFAC (CP, Harshman and Lundy, 1994), Tucker (Tucker, 1966) and their extensions have found applications in a variety of fields. Kolda and Bader (2009) presented a thorough overview of tensor decompositions and their applications. Recently, tensor factorizations are being actively studied in the statistics literature and have becoming an emerging field of statistics. To name a few, Chi and Kolda (2012) developed a Poisson tensor factorization model for sparse count data. Yang and Dunson (2016) proposed a conditional tensor factorization model for high-dimensional classification with categorical predictors. Sun et al. (2017) proposed a sparse tensor decomposition method by incorporating a truncation step into the tensor power iteration step.

Relational data sets are typically expressed as (subject, predicate, object) triples and can be grouped as a third-order tensor. As a result, tensor factorization methods can be naturally applied to these data sets. Nickel (2013) proposed a RESCAL factorization model for statistical relational learning. Compared to other tensor factorization approaches such as CP and Tucker methods, RESCAL is more capable of detecting the correlations produced between multiple interconnected nodes. For relational data consisting of n entities, K types of relations, and a positive integer s, RESCAL computes an $n \times s$ factor matrix and an $s \times s \times K$ core tensor. The factor matrix and the core tensor can be further used for link prediction, entity resolution and link-based clustering. Nickel et al. (2011) showed that a linear RESCAL model achieved better or comparable results on common benchmark data sets when compared to other existing methods such as MLN, DEDICOM (Harshman, 1978), IRM, CP, MRC (Kok and Domingos, 2007), etc. It was shown in Nickel and Tresp (2013) that a logistic RESCAL model could further improve the link prediction results.

Central to the empirical validity of RESCAL is the correct specification of the number of latent factors. Nickel et al. (2011) proposed to select this parameter via cross-validation. As commonly known for cross-validation methods, there's no theoretical guarantee against overestimation. Besides, cross-validation can be computationally expensive, especially for large n and K. In the literature, model selection is less studied for tensor factorization

©2019 Chengchun Shi, Wenbin Lu, Rui Song.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-037.html.

methods. Allen (2012) and Sun et al. (2017) proposed to use Bayesian information criteria (BIC, Schwarz, 1978) for sparse CP decomposition. However, no theoretical results were provided for BIC. Indeed, we show in this paper that a BIC-type criterion may fail for the RESCAL model.

The contribution of this paper is twofold. First, we propose a general class of information criteria for the RESCAL model and prove their model selection consistency. Although we focus on the RESCAL model, our information criteria can be extended to select models for general tensor factorization methods with slight modification. The problem is nonstandard and challenging since both the factor matrix and the core tensor are not observed and need to be estimated. Besides, the model parameters are non-identifiable. Moreover, the derivation of model/tuning parameter selection consistency of information criteria usually relies on the (uniform) consistency of estimated parameters. For example, Fan and Tang (2013) derived the uniform consistency of the maximum likelihood estimators (MLEs) to prove the consistency of GIC (see Proposition 2 in that paper). Zhang et al. (2016) established the uniform consistency of the support vector machine solutions to prove the consistency of SVMIC$_H$ (see Lemma 2 in that paper). The consistency of these estimators are due to the concavity (convexity) of the likelihood (or the empirical loss) functions. In contrast, for most tensor decomposition models including RESCAL, the likelihood (or the empirical loss) function is usually non-concave (non-convex) and may have multiple local solutions. As a result, the corresponding global maximizer (minimizer) may not be consistent even with the identifiability constraints. It remains unknown how to establish the consistency of the information criterion without consistency of the estimator. A key innovation in our analysis is to design a "proper" pseudometric and show that the global optimum is consistent under this specific pseudometric. We further establish the rate of convergence of the global optimum under this pseudometric as a function of n and K. Based on these results, we establish the consistency of our information criteria when K is either bounded or diverges at a proper rate of n. No parametric assumptions are imposed on the latent factors. Second, we introduce a scalable algorithm for estimating the parameters in the logistic RESCAL model. Despite the fact that a linear RESCAL model can be conveniently solved by an alternating least square algorithm (Nickel et al., 2011), there are lack of optimization algorithms for solving general RESCAL models. The proposed algorithm is based on the alternating direction method of multipliers (ADMM, Boyd et al., 2011) and can be implemented in a parallelized fashion.

The rest of the paper is organized as follows. We formally introduce the RESCAL model and study the parameter identifiability in Section 2. Our information criteria are presented in Section 3 and their model selection properties are investigated. Numerical examples are presented in Section 4 to examine the finite sample performance of the proposed information criteria. Section 5 concludes with a summary and discussion of future extensions. All the proofs are given in the Appendix.

2. The RESCAL Model

This section is structured as follows. We introduce the RESCAL model in Section 2.1. In Section 2.2, we study the identifiability of parameters in the model.

2.1. Model Setup

In knowledge graphs, facts can be expressed in the form of (subject, predicate, object) triples, where subject and object are entities and predicate is the relation between entities. For example, consider the following sentence from Wikipedia:

Jon Snow is a fictional character in the A Song of Ice and Fire series of fantasy novels by American author George R. R. Martin, and its television adaptation Game of Thrones.

The information contained in this message can be summarized into the following set of (subject, predicate, object) triples:

Subject	*Predicate*	*Object*
Jon Snow	*character in*	*A Song of Ice and Fire*
Jon Snow	*character in*	*Game of Thrones*
A Song of Ice and Fire	*genre*	*novel*
Game of Thrones	*genre*	*television series*
George R.R. Martin	*author of*	*A Song of Ice and Fire*
George R.R. Martin	*profession*	*novelist*

In this example, we have a total of 7 entities, 4 types of relations and 6 triples. More generally, let $\mathcal{E} = \{e_1, \ldots, e_n\}$ denote the set of all entities and $\mathcal{R} = \{r_1, \ldots, r_K\}$ denote the set of all relation types. The number of relations K is either bounded or diverges with n. Assuming non-existing triples indicate false relationships, we can construct a third-order binary tensor

$$\boldsymbol{Y} = \{Y_{ijk}\}_{i,j\in\{1,\ldots,n\},k\in\{1,\ldots,K\}},$$

such that

$$Y_{ijk} = \begin{cases} 1, & \text{if a triple } (e_i, r_k, e_j) \text{ exists,} \\ 0, & \text{otherwise.} \end{cases}$$

The RESCAL model is defined as follows. For each entity e_i, a latent vector $\boldsymbol{a}_{i,0} \in \mathbb{R}^{s_0}$ is generated. The Y_{ijk}'s are assumed to be conditionally independent given all latent factors $\{\boldsymbol{a}_{i,0}\}_{i=1}^n$. Besides, it is assumed that

$$\Pr(Y_{ijk} = 1 | \{\boldsymbol{a}_{i,0}\}_{i=1}^n) = \mathrm{g}(\boldsymbol{a}_{i,0}^T \boldsymbol{R}_{k,0} \boldsymbol{a}_{j,0}), \tag{1}$$

for some strictly monotone link function g and $s_0 \times s_0$ matrices $\boldsymbol{R}_{1,0}, \ldots, \boldsymbol{R}_{K,0}$. In the above model, $\boldsymbol{a}_{i,0}$ corresponds to the latent representation of the ith entity and $\boldsymbol{R}_{k,0}$ specifies how these $\boldsymbol{a}_{i,0}$'s interact for the k-th relation. To account for asymmetric relations, we do not restrict $\boldsymbol{R}_{k,0}$'s to symmetric matrices. When the relations are symmetric, i.e.,

$$\Pr(Y_{ijk} = 1 | \{\boldsymbol{a}_{i,0}\}_{i=1}^n) = \Pr(Y_{jik} = 1 | \{\boldsymbol{a}_{i,0}\}_{i=1}^n), \quad \forall i, j, k,$$

one can impose the symmetry constraints and obtain a similar derivation.

For continuous Y_{ijk}, a related tensor factorization model is the TUCKER-2 decomposition, which decomposes the tensor into

$$Y_{ijk} = \boldsymbol{a}_{i,0}^T \boldsymbol{R}_{k,0} \boldsymbol{b}_{j,0} + e_{ijk}, \quad \forall i, j, k, \tag{2}$$

for some $\boldsymbol{a}_{1,0},\ldots,\boldsymbol{a}_{n,0} \in \mathbb{R}^{s_1}$, $\boldsymbol{b}_{1,0},\ldots,\boldsymbol{b}_{n,0} \in \mathbb{R}^{s_2}$, $\boldsymbol{R}_{1,0},\ldots,\boldsymbol{R}_{K,0} \in \mathbb{R}^{s_1\times s_2}$ and some (random) errors $\{e_{ijk}\}_{ijk}$. By Equation 1, RESCAL can be interpreted as a "nonlinear" TUCKER-2 model with the additional constraints that $s_1 = s_2 = s_0$ and $\boldsymbol{a}_{i,0} = \boldsymbol{b}_{i,0}, \forall i$.

CP decomposition is another important tensor factorization method that decomposes a tensor into a sum of rank-1 tensors. It assumes that

$$Y_{ijk} = \sum_{s=1}^{s_0} a_{i,s}b_{j,s}r_{k,s} + e_{ijk},$$

for some $\{a_{i,s}\}_{i,s}$, $\{b_{j,s}\}_{j,s}$, $\{r_{k,s}\}_{k,s}$ and $\{e_{ijk}\}_{ijk}$. Define $\boldsymbol{a}_{i,0} = (a_{i,1},\ldots,a_{i,s_0})^T$ and $\boldsymbol{b}_{i,0} = (b_{i,1},\ldots,b_{i,s_0})^T$. In view of Equation 2, CP is a special TUCKER-2 model with the constraints that $s_1 = s_2 = s_0$ and $\boldsymbol{R}_{k,0} = \mathrm{diag}(r_{k,1},\ldots,r_{k,s_0})$ where $\mathrm{diag}(r_{k,1},\ldots,r_{k,s_0})$ is a diagonal matrix with the sth diagonal elements being $r_{k,s}$.

In this paper, the proposed information criteria are designed in particular for the RESCAL model. However, they can be extended to estimate s_0 in a more general tensor factorization framework including CP and TUCKER-2 models. We discuss this further in Section 5.

2.2. Identifiability

The parameterization in Equation 1 is not identifiable. To see this, for any nonsingular matrix $\boldsymbol{C} \in \mathbb{R}^{s_0\times s_0}$, we define $\boldsymbol{a}_i = \boldsymbol{C}^{-1}\boldsymbol{a}_{i,0}$, $\boldsymbol{R}_k = \boldsymbol{C}^T\boldsymbol{R}_{k,0}\boldsymbol{C}$, $\forall i,k$. Observe that

$$\boldsymbol{a}_{i,0}^T\boldsymbol{R}_{k,0}\boldsymbol{a}_{j,0} = \boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j, \quad \forall i,j,k,$$

and hence we have

$$\Pr(Y_{ijk} = 1) = \mathrm{g}(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j).$$

Let $\boldsymbol{A}_0 = [\boldsymbol{a}_{1,0},\ldots,\boldsymbol{a}_{n,0}]^T$. We impose the following condition.

(A0) (i) Assume $\boldsymbol{A}_0$ has full column rank. (ii) Assume the matrix $[\boldsymbol{R}_{1,0}^T,\ldots,\boldsymbol{R}_{K,0}^T]$ has full row rank.

(A0)(i) requires the latent factors to be linearly independent. (A0)(ii) holds when at least one of the $\boldsymbol{R}_{k,0}$'s has full rank. Under Condition (A0), the following lemma states that the RESCAL model is identifiable up to a nonsingular linear transformation. In Section B.1 of the Appendix, we show (A0) is also necessary to guarantee such identifiability when $\boldsymbol{R}_{1,0},\ldots,\boldsymbol{R}_{K,0}$ are symmetric.

Lemma 1 (Identifiability). *Assume (A0) holds. Assume there exist some $\{\boldsymbol{a}_i\}_i$, $\{\boldsymbol{R}_k\}_k$ such that $\boldsymbol{a}_i \in \mathbb{R}^{s_0}$, $\boldsymbol{R}_k \in \mathbb{R}^{s_0\times s_0}$ and*

$$g(\boldsymbol{a}_{i,0}^T\boldsymbol{R}_{k,0}\boldsymbol{a}_{j,0}) = g(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j), \quad \forall i,j,k.$$

Then, there exists some invertible matrix $\boldsymbol{C} \in \mathbb{R}^{s_0\times s_0}$ such that

$$\boldsymbol{a}_i = \boldsymbol{C}^{-1}\boldsymbol{a}_{i,0} \quad \textit{and} \quad \boldsymbol{R}_k = \boldsymbol{C}^T\boldsymbol{R}_{k,0}\boldsymbol{C}.$$

To fix the nonsingular transformation indeterminacy, we adopt a specific constrained parameterization and focus on estimating $\{\boldsymbol{a}_i^*\}_i$ and $\{\boldsymbol{R}_k^*\}_k$ where

$$\boldsymbol{a}_i^* = (\boldsymbol{A}_{s_0,0}^{-1})^T\boldsymbol{a}_{i,0} \quad \text{and} \quad \boldsymbol{R}_k^* = \boldsymbol{A}_{s_0,0}\boldsymbol{R}_{k,0}\boldsymbol{A}_{s_0,0}^T,$$

where $\boldsymbol{A}_{s_0,0} = [\boldsymbol{a}_{1,0},\ldots,\boldsymbol{a}_{s_0,0}]^T$. Observe that

$$[\boldsymbol{a}_1^*,\ldots,\boldsymbol{a}_{s_0}^*] = (\boldsymbol{A}_{s_0,0}^{-1})^T[\boldsymbol{a}_{1,0},\ldots,\boldsymbol{a}_{s_0,0}] = (\boldsymbol{A}_{s_0,0}^{-1})^T\boldsymbol{A}_{s_0,0}^T = \boldsymbol{I}_{s_0},$$

where $\boldsymbol{I}_{s_0}$ stands for an $s_0\times s_0$ identity matrix. Therefore, the first s_0 $\boldsymbol{a}_i^*$'s are fixed as long as $\boldsymbol{A}_{s_0,0}$ is nonsingular. By Lemma 1, the parameters $\{\boldsymbol{a}_i^*\}_i$ and $\{\boldsymbol{R}_k^*\}_k$ are estimable.

From now on, we only consider the logistic link function for simplicity, i.e, $\mathrm{g}(x) = 1/\{1+\exp(-x)\}$. Results for other link functions can be similarly discussed.

3. Model Selection

Parameters $\{\boldsymbol{a}_i^*\}_{i=1}^n$ and $\{\boldsymbol{R}_k^*\}_{k=1}^K$ can be estimated by maximizing the (conditional) log-likelihood function. Since we use the logistic link function, the log-likelihood is equal to

$$\begin{aligned}
\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i\}_i,\{\boldsymbol{R}_k\}_k) &= \log\left(\prod_{ijk}\mathrm{g}(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j)^{Y_{ijk}}\{1-\mathrm{g}(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j)\}^{1-Y_{ijk}}\right)\\
&= \sum_{ijk}\left(Y_{ijk}\log\{\mathrm{g}(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j)\} + (1-Y_{ijk})\log\{1-\mathrm{g}(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j)\}\right)\\
&= \sum_{ijk}\left(Y_{ijk}\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j - \log\{1+\exp(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j)\}\right),
\end{aligned}$$

where the first equality is due to the conditional independence assumption.

We assume the number of latent factors s_0 is fixed. For any $s \in \{1,\ldots,s_{\max}\}$ where $s_{\max}$ is allowed to diverge with n and satisfies $s_{\max} \geq s_0$, we define the following constrained maximum likelihood estimator

$$(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) = \underset{\substack{\boldsymbol{a}_1^{(s)},\ldots,\boldsymbol{a}_n^{(s)}\in\Theta_a^{(s)}\\ \mathrm{vec}(\boldsymbol{R}_1^{(s)}),\ldots,\mathrm{vec}(\boldsymbol{R}_K^{(s)})\in\Theta_r^{(s)}}}{\arg\max} \ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k), \tag{3}$$

$$\text{subject to} \quad [\boldsymbol{a}_1^{(s)},\ldots,\boldsymbol{a}_s^{(s)}] = \boldsymbol{I}_s, \tag{4}$$

for some $\Theta_a^{(s)} \subseteq \mathbb{R}^s$, $\Theta_r^{(s)} \subseteq \mathbb{R}^{s^2}$, where the $\mathrm{vec}(\cdot)$ operator stacks the entries of a matrix into a column vector. To estimate the number of latent factors, we define the following likelihood-based information criteria

$$\mathrm{IC}(s) = 2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) - s\kappa(n,K),$$

for some penalty functions $\kappa(\cdot,\cdot)$. The estimated number of latent factors is given by

$$\hat{s} = \underset{s\in\{1,\ldots,s_{\max}\}}{\arg\max}\ \mathrm{IC}(s). \tag{5}$$

In addition to the constraint in Equation 4, there exist many other constraints that would make the estimators identifiable. The choice of the identifiability constraints might affect the value of IC. However, it wouldn't affect the value of $\hat{s}$. Detailed discussions can be found in Section A of the Appendix.

A major technical difficulty in establishing the consistency of IC is due to the nonconcavity of the objective function given in Equation 3. For any $\{\boldsymbol{a}_j\}_{j\in\{1,\ldots,n\}}$, $\{\boldsymbol{R}_k\}_{k\in\{1,\ldots,K\}}$, let

$$\boldsymbol{\beta} = (\boldsymbol{a}_1^T, \ldots, \boldsymbol{a}_n^T, \mathrm{vec}(\boldsymbol{R}_1)^T, \ldots, \mathrm{vec}(\boldsymbol{R}_K)^T)^T,$$

be the set of parameters.

For any $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^s$, $\boldsymbol{T}_1, \ldots, \boldsymbol{T}_K \in \mathbb{R}^{s\times s}$, we define

$$\boldsymbol{\zeta} = (\boldsymbol{b}_1^T, \ldots, \boldsymbol{b}_n^T, \mathrm{vec}(\boldsymbol{T}_1)^T, \ldots, \mathrm{vec}(\boldsymbol{T}_K)^T)^T.$$

With some calculations, we can show that

$$\begin{aligned} -\boldsymbol{\zeta}^T \frac{\partial^2 \ell_n}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^T}\boldsymbol{\zeta} &= \underbrace{\sum_{ijk} \pi_{ijk}(1-\pi_{ijk})(\boldsymbol{b}_i^T\boldsymbol{R}_k\boldsymbol{a}_j + \boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{b}_j + \boldsymbol{a}_i^T\boldsymbol{T}_k\boldsymbol{a}_j)^2}_{I_1} \\ &+ \underbrace{\sum_{ijk}(\pi_{ijk} - Y_{ijk})(2\boldsymbol{b}_i^T\boldsymbol{R}_k\boldsymbol{b}_j + \boldsymbol{b}_i^T\boldsymbol{T}_k\boldsymbol{a}_j + \boldsymbol{a}_i^T\boldsymbol{T}_k\boldsymbol{b}_j)}_{I_2}, \end{aligned}$$

where $\pi_{ijk} = \exp(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j)/\{1+\exp(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j)\}$. Here, I_1 is nonnegative. However, I_2 can be negative for some $\boldsymbol{\beta}$ and $\boldsymbol{\zeta}$. Therefore, the negative Hessian matrix is not positive semidefinite and the likelihood function is not concave. As a result, $\widehat{\boldsymbol{a}}_i^{s_0}$ and $\widehat{\boldsymbol{R}}_k^{s_0}$ may not be consistent to $\boldsymbol{a}_i^*$ and $\boldsymbol{R}_k^*$, even with the identifiability constraints in Equation 4. Here, the presence of I_2 is due to the bilinear formulation of the RESCAL model.

Let $\theta_{ijk} = \boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j$. Notice that ℓ_n is concave in θ_{ijk}, $\forall i,j,k$. This motivates us to consider the following pseudometric:

$$\begin{aligned} & d\left(\{\boldsymbol{a}_{i,1}^{(s_1)}\}_i, \{\boldsymbol{R}_{k,1}^{(s_1)}\}_{k_1}; \{\boldsymbol{a}_{i,2}^{(s_2)}\}_i, \{\boldsymbol{R}_{k,2}^{(s_2)}\}_{k_2}\right) \\ = & \left\{\frac{1}{n^2K}\sum_{ijk}\left((\boldsymbol{a}_{i,1}^{(s_1)})^T(\boldsymbol{R}_{k,1}^{(s_1)})^T\boldsymbol{a}_{j,1}^{(s_1)} - (\boldsymbol{a}_{i,2}^{(s_2)})^T(\boldsymbol{R}_{k,2}^{(s_2)})^T\boldsymbol{a}_{j,2}^{(s_2)}\right)^2\right\}^{1/2}, \end{aligned}$$

for any integers $s_1, s_2 > 0$ and $\boldsymbol{a}_{i,1}^{(s_1)} \in \mathbb{R}^{s_1}$, $\boldsymbol{R}_{k,1}^{(s_1)} \in \mathbb{R}^{s_1\times s_1}$, $\boldsymbol{a}_{i,2}^{(s_2)} \in \mathbb{R}^{s_2}$, $\boldsymbol{R}_{k,2}^{(s_2)} \in \mathbb{R}^{s_2\times s_2}$. Apparently, $d(\cdot,\cdot)$ is nonnegative, symmetric and satisfies the triangle inequality. Below, we establish the convergence rate of

$$d\left(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k; \{\boldsymbol{a}_i^*\}_i, \{\boldsymbol{R}_k^*\}_k\right).$$

We first introduce some notation. For any $s > s_0$, we define

$$\boldsymbol{a}_{i,0}^{(s)} = \begin{cases} ((\boldsymbol{a}_{i,0})^T, \boldsymbol{0}_{s-s_0}^T)^T, & i \notin \{s_0+1,\ldots,s\}, \\ ((\boldsymbol{a}_{i,0})^T, \underbrace{0,\ldots,0}_{i-s_0-1}, 1, \underbrace{0,\ldots,0}_{s-i})^T, & i \in \{s_0+1,\ldots,s\}, \end{cases}$$

and

$$\boldsymbol{R}_{k,0}^{(s)} = \begin{pmatrix} \boldsymbol{R}_{k,0} & \boldsymbol{O}_{r,s-r} \\ \boldsymbol{O}_{s-r,r} & \boldsymbol{O}_{s-r,s-r} \end{pmatrix},$$

where $\boldsymbol{0}_q$ denotes a q-dimensional zero vector and $\boldsymbol{O}_{p,q}$ is an $p\times q$ zero matrix. With a slight abuse of notation, we write $\boldsymbol{a}_{i,0}^{(s_0)} = \boldsymbol{a}_{i,0}$ and $\boldsymbol{R}_{k,0}^{(s_0)} = \boldsymbol{R}_{k,0}$. Clearly, for any $s \geq s_0$, we have

$$(\boldsymbol{a}_{i,0}^{(s)})^T\boldsymbol{R}_{k,0}^{(s)}\boldsymbol{a}_{j,0}^{(s)} = \boldsymbol{a}_{i,0}^T\boldsymbol{R}_{k,0}\boldsymbol{a}_{j,0}, \quad \forall i,j,k,$$

and hence

$$(\{\boldsymbol{a}_{i,0}^{(s)}\}_i, \{\boldsymbol{R}_{k,0}^{(s)}\}_k) = \underset{\{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k}{\arg\max} \mathrm{E}\ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k).$$

Let

$$\boldsymbol{a}_i^{(s)*} = (\boldsymbol{A}_{s,0}^{-1})^T\boldsymbol{a}_{i,0}^{(s)} \quad \text{and} \quad \boldsymbol{R}_k^{(s)*} = \boldsymbol{A}_{s,0}\boldsymbol{R}_{k,0}^{(s)}\boldsymbol{A}_{s,0}^T,$$

where $\boldsymbol{A}_{s,0} = [\boldsymbol{a}_{1,0}^{(s)}, \ldots, \boldsymbol{a}_{s,0}^{(s)}]^T$. When $\boldsymbol{A}_{s_0,0}$ is invertible, $\boldsymbol{A}_{s,0}$'s are invertible for all $s > s_0$. The defined $\{\boldsymbol{a}_i^{(s)*}\}$'s satisfy the identifiability constraints in Equation 4 for all $s \geq s_0$. We make the following assumption.

(A1) Assume $\boldsymbol{a}_i^{(s)*} \in \Theta_a^{(s)}$ and $\mathrm{vec}(\boldsymbol{R}_k^{(s)*}) \in \Theta_r^{(s)}$, $\forall i = 1,\ldots,n$, $k = 1,\ldots,K$ and $s_0 \leq s \leq s_{\max}$. In addition, assume $\sup_{\boldsymbol{x}\in\Theta_a^{(s)}}\|\boldsymbol{x}\|_2 \leq \omega_a$, $\sup_{\boldsymbol{y}\in\Theta_r^{(s)}}\|\boldsymbol{y}\|_2 \leq \omega_r$ for some $\omega_a, \omega_r > 0$.

Lemma 2. *Assume (A1) holds, $s_{\max} = o\{\sqrt{n/\log(nK)}\}$. Then there exists some constant $C_0 > 0$ such that the following event occurs with probability tending to 1,*

$$\max_{s\in\{s_0,\ldots,s_{\max}\}} \frac{1}{s^2}d^2\left(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k; \{\boldsymbol{a}_i^*\}_i, \{\boldsymbol{R}_k^*\}_k\right) \leq \frac{\exp(2\omega_a^2\omega_r)(n+K)(\log n + \log K)}{n^2K}.$$

Under the condition $s_{\max} = o\{\sqrt{n/\log(nK)}\}$, we have that

$$\frac{s_{\max}^2(n+K)(\log n + \log K)}{n^2K} \leq \frac{s_{\max}^2(2n\log n + 2K\log K)}{n^2K} \leq \frac{2s_{\max}^2\log n}{n} + \frac{2s_{\max}^2\log K}{n^2} = o(1).$$

When ω_a and ω_r are bounded, it follows that

$$\max_{s\in\{s_0,\ldots,s_{\max}\}} d^2\left(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k; \{\boldsymbol{a}_i^*\}_i, \{\boldsymbol{R}_k^*\}_k\right) \leq \frac{s_{\max}^2\exp(2\omega_a^2\omega_r)(n+K)(\log n + \log K)}{n^2K} = o(1).$$

Hence, $\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i$ and $\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k$ are consistent under the pseudometric d for all overfitted models. On the contrary, for underfitted models, we require the following conditions.

(A2) Assume there exists some constant $\bar{c} > 0$ such that $\lambda_{\min}(\boldsymbol{A}_0^T\boldsymbol{A}_0) \geq n\bar{c}$.
(A3) Let $\bar{K} = \lambda_{\min}(\sum_{k=1}^K \boldsymbol{R}_{k,0}^T\boldsymbol{R}_{k,0})$. Assume $\liminf_n \bar{K} > 0$.

Lemma 3. *Assume (A2) and (A3) hold. The for any $s \in \{1, 2, \ldots, s_0 - 1\}$, we have*

$$d^2\left(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k; \{\boldsymbol{a}_i^*\}_i, \{\boldsymbol{R}_k^*\}_k\right) \geq \frac{\bar{c}^2\bar{K}}{K},$$

where $\bar{c}$ and $\bar{K}$ are defined in (A2) and (A3), respectively.

Assumption (A3) holds if there exists some $k_0 \in \{1, \ldots, K\}$ such that

$$\liminf_n \lambda_{\min}(\boldsymbol{R}_{k_0,0}\boldsymbol{R}_{k_0,0}^T) > 0.$$

When $\bar{K} \geq c'K$ for some constant $c' > 0$, it follows from Lemma 3 that

$$\liminf_n d\left(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k; \{\boldsymbol{a}_i^*\}_i, \{\boldsymbol{R}_k^*\}_k\right) > 0.$$

Based on these results, we establish the consistency of $\hat{s}$ defined in Equation 5 below. For any sequences $\{a_n\}$ and $\{b_n\}$, we write $a_n \sim b_n$ if there exist some universal constants $c_1, c_2 > 0$ such that $c_1 a_n \leq b_n \leq c_2 a_n$.

Theorem 1. *Assume (A1)-(A3) hold, $K \sim n^{l_0}$ for some $0 \leq l_0 \leq 1$, $s_{\max} = o\{\sqrt{n/\log(nK)}\}$, $\liminf_n n^{(1-l_0)/2}\bar{K} \geq \exp(2\omega_a^2\omega_r)\sqrt{\log n}$. Assume $\kappa(n, K)$ satisfies*

$$s_{\max}\exp(\omega_a^2\omega_r)(n + K)(\log n + \log K) \ll \kappa(n, K) \ll \frac{n^2\bar{K}}{\exp(\omega_a^2\omega_r)}. \tag{6}$$

Then, we have $Pr(\hat{s} = s_0) \to 1$ where $\hat{s}$ is defined in Equation 5.

Let $c(n, K) = \kappa(n, K)(n + K)^{-1}(\log n + \log K)^{-1}$. When $s_{\max}, \omega_a, \omega_r$ are bounded, it follows from Theorem 1 that IC is consistent provided that $c(n, K) \to \infty$ and $c(n, K) = o(n\bar{K}/\log n)$. Define

$$\tau_\alpha(n, K) = \frac{(n + K)^\alpha}{\max^\alpha(n, K)},$$

for some $\alpha \geq 0$. Note that

$$1 \leq \tau_\alpha(n, K) \leq 2^\alpha. \tag{7}$$

Consider the following criteria:

$$\mathrm{IC}_\alpha(s) = 2\ell_n(\boldsymbol{Y}; \{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) - s\tau_\alpha(n, K)(n + K)(\log n + \log K)\log\{\log(nK)\}. \tag{8}$$

Note that the term $\log\{\log(nK)\}$ satisfies that $\log\{\log(nK)\} \to \infty$ and $\log\{\log(nK)\} = o(n/\log n)$. It follows from Equation 7 and Theorem 1 that IC_α is consistent for all $\alpha \geq 0$. When $\alpha > 0$, the term $\tau_\alpha(n, K)$ adjust the model complexity penalty upwards. We notice that Bai and Ng (2002) used a similar finite sample correction term in their proposed information criteria for approximate factor models. Our simulation studies show that such adjustment is essential to achieve selection consistency for large K.

Conditions (A1) and (A2) are directly imposed on the realizations of $\boldsymbol{a}_{1,0}, \ldots, \boldsymbol{a}_{n,0}$. In Section B.2 and B.3, we consider an asymptotic framework where $\boldsymbol{a}_{1,0}, \ldots, \boldsymbol{a}_{n,0}$ are i.i.d according to some distribution function and show (A1) and (A2) hold with probability tending to 1. Therefore, under this framework, we still have $\Pr(\hat{s} = s_0) \to 1$. The consistency of our information criterion remains unchanged.

Observe that we have a total of $n \times n \times K = n^2K$ observations. Consider the following BIC-type criterion:

$$\mathrm{BIC}(s) = 2\ell_n(\boldsymbol{Y}; \{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) - s\log(n^2K). \tag{9}$$

The model complexity penalty in BIC satisfies

$$\log(n^2K) = 2\log n + \log K \ll (n + K)(\log n + \log K).$$

Hence, it does not meet Condition (6) in Theorem 1. As a result, BIC may fail to identity the true model. As shown in our simulation studies, BIC will choose overfitted models and is not selection consistent.

4. Numerical Experiments

This section is organized as follows. In Section 4.1, we introduce our algorithm for computing the maximum likelihood estimators of a logistic RESCAL model. Simulation studies are presented in Section 4.2. In Section 4.3, we apply the proposed information criteria to a real dataset.

4.1. Implementation

In this section, we propose an algorithm for computing $\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i$ and $\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k$. The algorithm is based upon a 3-block alternating direction method of multipliers (ADMM). Set $\Theta_a^{(s)} = \mathbb{R}^s$, $\Theta_r^{(s)} = \mathbb{R}^{s\times s}$, $[\boldsymbol{a}_1^{(s)}, \ldots, \boldsymbol{a}_s^{(s)}] = \boldsymbol{I}_s$, $\widehat{\boldsymbol{a}}_i^{(s)}$'s and $\widehat{\boldsymbol{R}}_k^{(s)}$'s are defined by

$$(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_{i=(s+1)}^n, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) = \underset{\{\boldsymbol{a}_i^{(s)}\}_{i=s+1}^n, \{\boldsymbol{R}_k^{(s)}\}_k}{\arg\max} \ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k), \tag{10}$$

where

$$\ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k) = \sum_{ijk}\left(Y_{ijk}(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)} - \log[1 + \exp\{(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}\}]\right).$$

For any $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^s$, define

$$\bar{\ell}_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k, \{\boldsymbol{b}_i^{(s)}\}_i) = \sum_{ijk}\left(Y_{ijk}(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{b}_j^{(s)} - \log[1 + \exp\{(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{b}_j^{(s)}\}]\right).$$

Fix $[\boldsymbol{b}_1^{(s)}, \ldots, \boldsymbol{b}_s^{(s)}] = \boldsymbol{I}_s$, the optimization problem in Equation 10 is equivalent to

$$(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_{i=s+1}^n, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k, \{\widehat{\boldsymbol{b}}_i^{(s)}\}_{i=s+1}^n) = \underset{\substack{\{\boldsymbol{a}_i^{(s)}\}_{i=s+1}^n, \{\boldsymbol{R}_k^{(s)}\}_k \\ \{\boldsymbol{b}_i^{(s)}\}_{i=s+1}^n}}{\arg\max} \bar{\ell}_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k, \{\boldsymbol{b}_i^{(s)}\}_i),$$
$$\text{subject to} \qquad \boldsymbol{a}_i^{(s)} = \boldsymbol{b}_i^{(s)}, \forall i = s + 1, \ldots, n.$$

We then derive its augmented Lagrangian, which gives us

$$\begin{aligned}&L_\rho(\{\boldsymbol{a}_i^{(s)}\}_{i=s+1}^n, \{\boldsymbol{R}_k^{(s)}\}_k, \{\boldsymbol{b}_i^{(s)}\}_{i=s+1}^n, \{\boldsymbol{v}_i^{(s)}\}_{i=s+1}^n)\\ =\ & -\bar{\ell}_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k, \{\boldsymbol{b}_i^{(s)}\}_i) + \sum_{i=s+1}^n \rho(\boldsymbol{a}_i^{(s)} - \boldsymbol{b}_i^{(s)})^T \boldsymbol{v}_i^{(s)} + \sum_{i=s+1}^n \frac{\rho}{2}\|\boldsymbol{a}_i^{(s)} - \boldsymbol{b}_i^{(s)}\|_2^2,\end{aligned}$$

where $\rho > 0$ is a penalty parameter and $\boldsymbol{v}_{s+1}^{(s)}, \ldots, \boldsymbol{v}_n^{(s)} \in \mathbb{R}^s$.

Applying dual descent method yields the following steps, with l denotes the iteration number:

$$\{\boldsymbol{a}_{i,l+1}^{(s)}\}_{i=s+1}^n = \underset{\{\boldsymbol{a}_i^{(s)}\}_{i=s+1}^n}{\arg\min}\ L_\rho(\{\boldsymbol{a}_i^{(s)}\}_{i=s+1}^n, \{\boldsymbol{R}_{k,l}^{(s)}\}_k, \{\boldsymbol{b}_{i,l}^{(s)}\}_{i=s+1}^n, \{\boldsymbol{v}_{i,l}^{(s)}\}_{i=s+1}^n), \quad (11)$$

$$\{\boldsymbol{R}_{k,l+1}^{(s)}\}_{k=1}^K = \underset{\{\boldsymbol{R}_k^{(s)}\}_{k=1}^K}{\arg\min}\ L_\rho(\{\boldsymbol{a}_{i,l+1}^{(s)}\}_{i=s+1}^n, \{\boldsymbol{R}_k^{(s)}\}_k, \{\boldsymbol{b}_{i,l}^{(s)}\}_{i=s+1}^n, \{\boldsymbol{v}_{i,l}^{(s)}\}_{i=s+1}^n), \quad (12)$$

$$\{\boldsymbol{b}_{i,l+1}^{(s)}\}_{i=s+1}^n = \underset{\{\boldsymbol{b}_i^{(s)}\}_{i=s+1}^n}{\arg\min}\ L_\rho(\{\boldsymbol{a}_{i,l+1}^{(s)}\}_{i=s+1}^n, \{\boldsymbol{R}_{k,l+1}^{(s)}\}_k, \{\boldsymbol{b}_i^{(s)}\}_{i=s+1}^n, \{\boldsymbol{v}_{i,l}^{(s)}\}_{i=s+1}^n), \quad (13)$$

$$\boldsymbol{v}_{i,l+1}^{(s)} = \boldsymbol{v}_{i,l}^{(s)} + \boldsymbol{a}_{i,l}^{(s)} - \boldsymbol{b}_{i,l}^{(s)}, \qquad \forall i = s+1, \ldots, n.$$

Let us examine Equation 11-13 in more details. In Equation 11, we rewrite the objective function as

$$\begin{aligned}&L_\rho(\{\boldsymbol{a}_i^{(s)}\}_{i=s+1}^n, \{\boldsymbol{R}_{k,l}^{(s)}\}_k, \{\boldsymbol{b}_{i,l}^{(s)}\}_{i=s+1}^n, \{\boldsymbol{v}_{i,l}^{(s)}\}_{i=s+1}^n)\\ =\ & \sum_{i=s+1}^n \left\{ \sum_{j,k} \left(\log[1 + \exp\{(\boldsymbol{a}_i^{(s)})^T \boldsymbol{R}_{k,l}^{(s)} \boldsymbol{b}_{j,l}^{(s)}\}] - Y_{ijk} (\boldsymbol{a}_i^{(s)})^T \boldsymbol{R}_{k,l}^{(s)} \boldsymbol{b}_{j,l}^{(s)} \right) \right.\\ +\ & \left. \rho(\boldsymbol{a}_i^{(s)} - \boldsymbol{b}_{i,l}^{(s)})^T \boldsymbol{v}_{i,l}^{(s)} + \frac{\rho}{2}\|\boldsymbol{a}_i^{(s)} - \boldsymbol{b}_{i,l}^{(s)}\|_2^2 \right\}.\end{aligned}$$

Note that L_ρ can be represented as a separable sum of functions. As a result, $\boldsymbol{a}_{i,l+1}^{(s)}$'s can be solved in parallel. More specifically, we have

$$\begin{aligned}\boldsymbol{a}_{i,l+1}^{(s)} =\ & \underset{\boldsymbol{a}^{(s)}}{\arg\min} \left\{ \sum_{j,k} \left(\log[1 + \exp\{(\boldsymbol{a}_i^{(s)})^T \boldsymbol{R}_{k,l}^{(s)} \boldsymbol{b}_{j,l}^{(s)}\}] - Y_{ijk} (\boldsymbol{a}_i^{(s)})^T \boldsymbol{R}_{k,l}^{(s)} \boldsymbol{b}_{j,l}^{(s)} \right) \right.\\ +\ & \left. \rho(\boldsymbol{a}_i^{(s)} - \boldsymbol{b}_{i,l}^{(s)})^T \boldsymbol{v}_{i,l}^{(s)} + \frac{\rho}{2}\|\boldsymbol{a}_i^{(s)} - \boldsymbol{b}_{i,l}^{(s)}\|_2^2 \right\}.\end{aligned}$$

Hence, each $\boldsymbol{a}_{i,l+1}^{(s)}$ can be computed by solving a ridge type logistic regression with responses $\{Y_{ijk}\}_{j,k}$ and covariates $\{\boldsymbol{R}_{k,l}^{(s)} \boldsymbol{b}_{j,l}^{(s)}\}_{j,k}$.

In Equation 12, each $\boldsymbol{R}_{k,l+1}^{(s)}$ can be independently updated by solving a logistic regression with responses $\{Y_{ijk}\}_{i,j}$ and covariates $\boldsymbol{b}_{j,l}^{(s)} \otimes \boldsymbol{a}_{i,l+1}^{(s)}$, i.e,

$$\mathrm{vec}(\boldsymbol{R}_{k,l+1}^{(s)}) = \underset{\boldsymbol{r}_k^{(s)} \in \mathbb{R}^{s^2}}{\arg\min} \sum_{ij} \left\{ \log\left(1 + \exp[\{(\boldsymbol{b}_{j,l}^{(s)})^T \otimes (\boldsymbol{a}_{i,l+1}^{(s)})^T\} \boldsymbol{r}_k^{(s)}]\right) - Y_{ijk} \{(\boldsymbol{b}_{j,l}^{(s)})^T \otimes (\boldsymbol{a}_{i,l+1}^{(s)})^T\} \boldsymbol{r}_k^{(s)} \right\},$$

where $\otimes$ denotes the Kronecker product.

Similar to Equation 11, each $\boldsymbol{b}_{i,l+1}^{(s)}$ in Equation 13 can be independently computed by solving a ridge type regression with responses $\{Y_{ijk}\}_{j,k}$ and covariates $\{(\boldsymbol{R}_{k,l+1}^{(s)})^T \boldsymbol{a}_{j,l+1}^{(s)}\}_{j,k}$.

Using similar arguments in Theorem 2 in Wang et al. (2017), we can show that the proposed 3-block ADMM algorithm converges for any sufficiently large ρ. In our implementation, we set $\rho = nK/2$. To guarantee global convergence, we randomly generate multiple initial estimators and solve the optimization problem multiple times based on these initial values.

4.2. Simulations

We simulate $\{Y_{ijk}\}_{ijk}$ from the following model:

$$\begin{aligned}&\Pr(Y_{ijk} = 1 | \{\boldsymbol{a}_i\}_i, \{\boldsymbol{R}_k\}_k) = \frac{\exp(\boldsymbol{a}_i^T \boldsymbol{R}_k \boldsymbol{a}_j)}{1 + \exp(\boldsymbol{a}_i^T \boldsymbol{R}_k \boldsymbol{a}_j)},\\ &\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n \overset{iid}{\sim} N(0,1),\\ &\boldsymbol{R}_1 = \boldsymbol{R}_2 = \cdots = \boldsymbol{R}_K = \mathrm{diag}(\underbrace{1, -1, 1, -1, \ldots, 1, -1}_{s_0}),\end{aligned}$$

where $N(0,1)$ stands for a standard normal random variable and $\mathrm{diag}(v_1, \ldots, v_q)$ denotes a $q \times q$ diagonal matrix with the jth element equal to v_j.

We consider six simulation settings. In the first three settings, we fix $K = 3$ and set $n = 100, 150$ and 200, respectively. In the last three settings, we increase K to 10, 20, 50, and set $n = 50$. In each setting, we further consider three scenarios, by setting $s_0 = 2, 4$ and 8. Let $s_{\max} = 12$. The ADMM algorithm proposed in Section 4.1 is implemented in R. Some subroutines of the algorithm are written in C with the GNU Scientific Library (GSL, Galassi et al., 2015) to facilitate the computation. We compare the proposed IC_α (see Equation 8) with the BIC-type criterion (see Equation 9). In IC_α, we set $\alpha = 0, 0.5$ and 1. Note that when $\alpha = 0$, we have

$$\tau_\alpha(n, K) = \frac{(n+K)^\alpha}{\max^\alpha(n, K)} = 1.$$

Reported in Table 1 and 2 are the percentage of selecting the true models (TP) and the average of $\hat{s}$ selected by IC_0, $\mathrm{IC}_{0.5}$, IC_1 and BIC over 100 replications.

It can be seen from Table 1 and 2 that BIC fails in all settings. It always selects overfitted models. On the contrary, the proposed information criteria are consistent for most of the settings. For example, under settings where $s_0 = 2$ or 4, TPs of IC_0, $\mathrm{IC}_{0.5}$ and IC_1 are larger than or equal to 93%. When $s_0 = 8$, expect for the last setting, TPs of the proposed information criteria are no less than 60% for all cases.

IC_0, $\mathrm{IC}_{0.5}$ and IC_1 perform very similarly for small K. In the first three settings, TPs of these three information criteria are nearly the same for all cases. However, $\mathrm{IC}_{0.5}$ and IC_1 are more robust than IC_0 for large K. This can be seen in the last scenario of Setting 6, where the TP of IC_0 is no more than 20%. Besides, in the last two settings, TP of IC_0 is smaller than $\mathrm{IC}_{0.5}$ and IC_1 for all cases. These differences are due to the finite sample

	$s_0 = 2$		$s_0 = 4$		$s_0 = 8$	
$n = 100, K = 3$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	0.97 (0.02)	2.03 (0.02)	0.97 (0.02)	4.03 (0.02)	0.90(0.03)	7.90 (0.03)
$\text{IC}_{0.5}$	0.97 (0.02)	2.03 (0.02)	0.98 (0.01)	4.02 (0.01)	0.90(0.03)	7.90 (0.03)
IC_1	0.97 (0.02)	2.03 (0.02)	0.98 (0.01)	4.02 (0.01)	0.89(0.03)	7.89 (0.03)
BIC	0.00 (0.00)	11.99 (0.01)	0.00 (0.00)	12.00 (0.00)	0.00 (0.00)	11.99 (0.01)
$n = 150, K = 3$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	0.99 (0.01)	2.01 (0.01)	0.97 (0.02)	4.03 (0.02)	0.96(0.02)	8.04 (0.02)
$\text{IC}_{0.5}$	0.99 (0.01)	2.01 (0.01)	0.97 (0.02)	4.03 (0.02)	0.96(0.02)	8.04 (0.02)
IC_1	0.99 (0.01)	2.01 (0.01)	0.97 (0.02)	4.03 (0.02)	0.96(0.02)	8.04 (0.02)
BIC	0.00 (0.00)	12.00 (0.00)	0.00 (0.00)	12.00 (0.00)	0.00 (0.00)	11.98 (0.01)
$n = 200, K = 3$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	0.99 (0.01)	2.01 (0.01)	0.95 (0.02)	4.05 (0.02)	0.95(0.02)	8.05 (0.02)
$\text{IC}_{0.5}$	0.99 (0.01)	2.01 (0.01)	0.95 (0.02)	4.05 (0.02)	0.95(0.02)	8.05 (0.02)
IC_1	0.99 (0.01)	2.01 (0.01)	0.95 (0.02)	4.05 (0.02)	0.95(0.02)	8.05 (0.02)
BIC	0.00 (0.00)	12.00 (0.00)	0.00 (0.00)	11.99 (0.01)	0.00 (0.00)	11.98 (0.01)

Table 1: Simulation results for Setting I, II and III (standard errors in parenthesis)

	$s_0 = 2$		$s_0 = 4$		$s_0 = 8$	
$n = 50, K = 10$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	1.00 (0.00)	2.00 (0.00)	0.97 (0.02)	4.03 (0.02)	0.69(0.05)	7.91 (0.06)
$\text{IC}_{0.5}$	1.00 (0.00)	2.00 (0.00)	0.97 (0.02)	4.03 (0.02)	0.66(0.05)	7.75 (0.06)
IC_1	1.00 (0.00)	2.00 (0.00)	0.98 (0.01)	4.02 (0.01)	0.60(0.05)	7.62 (0.06)
BIC	0.00 (0.00)	11.81 (0.06)	0.00 (0.00)	11.60 (0.06)	0.01 (0.01)	11.67 (0.07)
$n = 50, K = 20$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	0.97 (0.02)	2.03 (0.02)	0.95 (0.02)	4.05 (0.02)	0.73(0.04)	8.46 (0.10)
$\text{IC}_{0.5}$	0.97 (0.02)	2.03 (0.02)	0.98 (0.01)	4.02 (0.01)	0.87(0.03)	8.09 (0.03)
IC_1	0.98 (0.01)	2.02 (0.02)	1.00 (0.00)	4.00 (0.00)	0.79(0.04)	7.99 (0.05)
BIC	0.00 (0.00)	12.00 (0.00)	0.00 (0.00)	11.92 (0.03)	0.00 (0.00)	11.99 (0.01)
$n = 50, K = 50$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	0.98 (0.01)	2.02 (0.01)	0.93 (0.03)	4.07 (0.03)	0.17(0.04)	11.24 (0.15)
$\text{IC}_{0.5}$	0.99 (0.01)	2.01 (0.01)	0.97 (0.02)	4.03 (0.02)	0.76(0.04)	8.24 (0.05)
IC_1	1.00 (0.00)	2.00 (0.00)	0.98 (0.01)	4.02 (0.01)	0.79(0.04)	7.99 (0.05)
BIC	0.00 (0.00)	12.00 (0.00)	0.00 (0.00)	12.00 (0.00)	0.00 (0.00)	11.99 (0.01)

Table 2: Simulation results for Setting IV, V and VI (standard errors in parenthesis)

s	1	2	3	4	5	6	7	8	9	10	11	12
AUC	0.7201	0.8341	0.8952	0.9095	0.9257	0.9364	0.9444	0.9486	0.9513	0.9518	0.9485	0.9467

Table 3: AUC scores

correction term $\tau_\alpha(n, K)$. As commented before, $\tau_{0.5}(n, K)$ and $\tau_1(n, K)$ increase the model complexity penalty term in $\text{IC}_{0.5}$ and IC_1 to avoid overfitting for large K.

In Section D of the Appendix, we examine the performance of our proposed information criteria under the scenario where $\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n \overset{iid}{\sim} N(0, \{0.5^{|i-j|}\}_{i,j=1,\ldots,s_0})$. Results are similar to those presented at Table 1 and 2.

4.3. Real Data Experiments

In this section, we apply the proposed information criteria to the "Social Evolution" dataset (Madan et al., 2012). This dataset comes from MIT's Human Dynamics Laboratory. It tracks everyday life of a whole undergraduate MIT dormitory from October 2008 to May 2009. We use the survey data, resulting in $n = 84$ participants and $K = 5$ binary relations. The five relations are: close relationship, political discussion, social interaction and two social media interaction.

We compute $\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i$ and $\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k$ for $s = \{1, \ldots, 12\}$ and select the number of latent factors using the proposed information criteria and BIC. It turns out that IC_0, $\text{IC}_{0.5}$ and IC_1 all suggest the presence of 9 factors. In contrast, BIC selects 12 factors. To further evaluate the number of latent factors selected by the proposed information criteria, we consider the following cross-validation procedure. For any $s \in [1, \ldots, 12]$, we randomly select 80% of the observations and estimate $\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i$ and $\{\widehat{\boldsymbol{R}}_k^{(s)}\}$ by maximizing the observed likelihood function based on these training samples. Then we compute

$$\widehat{\pi}_{ijk} = \frac{\exp\{(\widehat{\boldsymbol{a}}_i^{(s)})^T \widehat{\boldsymbol{R}}_k^{(s)} \widehat{\boldsymbol{a}}_j^{(s)}\}}{1 + \exp\{(\widehat{\boldsymbol{a}}_i^{(s)})^T \widehat{\boldsymbol{R}}_k^{(s)} \widehat{\boldsymbol{a}}_j^{(s)}\}}.$$

Based on these predicted probabilities, we calculate the area under the precision-recall curve (AUC) on the remaining 20% testing samples.

Reported in Table 3 are the AUC scores averaged over 100 replications. For any $s \in \{1, \ldots, 12\}$, we denoted by AUC_s the corresponding AUC score. It can be seen from Table 3 that AUC_s first increases and then decreases as s increases. The maximum AUC score is achieved at $s = 10$. Observe that AUC_9 is very close to AUC_{10}, and it is larger than the remaining AUC scores. This demonstrates that the proposed information criteria select less latent factors while achieve better or similar link prediction results when compared to BIC.

5. Discussion

In this paper, we propose information criteria for selecting the number of latent factors in the RESCAL tensor factorization model and prove their model selection consistency. Although we focus on the logistic RESCAL model, the proposed information criteria can be applied to general tensor factorization models. More specifically, consider the following

class of models:

$$Y_{ijk} = \mathrm{g}(\boldsymbol{a}_{i,0}^T \boldsymbol{R}_{k,0} \boldsymbol{b}_{j,0}) + e_{ijk}, \quad \forall i, j \in \{1, \ldots, n\}, k \in \{1, \ldots, K\}, \tag{14}$$

with any of (or without) the following constraints:
(C1) $\boldsymbol{R}_{k,0}$ is diagonal;
(C2) $\boldsymbol{a}_{i,0} = \boldsymbol{b}_{i,0}$ for $i \in \{1, \ldots, n\}$,
for some strictly increasing function g, $\boldsymbol{a}_{i,0}, \boldsymbol{b}_{i,0} \in \mathbb{R}^{s_0}$, $\boldsymbol{R}_{k,0} \in \mathbb{R}^{s_0 \times s_0}$ and some mean zero random errors $\{e_{ijk}\}_{ijk}$.

As commented in Section 2.1, such representation includes the RESCAL, CP and TUCKER-2 models. Specifically, it reduces to the TUCKER-2 model by setting g to be the identity function. If further (C1) holds, then the model in Equation 14 reduces to the CP model. When (C2) holds, it corresponds to the RESCAL model. Consider the following information criteria,

$$\mathrm{IC}(s) = \ell_n(\boldsymbol{Y}; \{\widehat{\boldsymbol{a}}_i\}_i, \{\widehat{\boldsymbol{R}}_k\}_k, \{\widehat{\boldsymbol{b}}_i\}_i) - s\kappa(n, K),$$

where ℓ_n stands for the likelihood function and $\widehat{\boldsymbol{a}}_i$, $\widehat{\boldsymbol{R}}_k$, $\widehat{\boldsymbol{b}}_i$ are the corresponding (constrained) MLEs. Similar to Theorem 1, we can show that with some properly chosen $\kappa(n, K)$, IC is consistent under this general setting.

Currently, we assume the tensor $\boldsymbol{Y}$ is completely observed. When some of the Y_{ijk}'s are missing, we can calculate $\widehat{\boldsymbol{a}}_i^{(s)}$'s and $\widehat{\boldsymbol{R}}_k^{(s)}$'s by maximizing the following observed likelihood function

$$\begin{aligned} \underset{\substack{\boldsymbol{a}_1^{(s)}, \ldots, \boldsymbol{a}_n^{(s)} \in \Theta_a^{(s)} \\ \mathrm{vec}(\boldsymbol{R}_1^{(s)}), \ldots, \mathrm{vec}(\boldsymbol{R}_K^{(s)}) \in \Theta_r^{(s)}}}{\arg\max} & \sum_{(i,j,k) \in N_{obs}} \left(Y_{ijk} (\boldsymbol{a}_i^{(s)})^T \boldsymbol{R}_k^{(s)} \boldsymbol{a}_j^{(s)} - \log[1 + \exp\{(\boldsymbol{a}_i^{(s)})^T \boldsymbol{R}_k^{(s)} \boldsymbol{a}_j^{(s)}\}] \right), \\ \text{subject to} \quad & [\boldsymbol{a}_1^{(s)}, \ldots, \boldsymbol{a}_s^{(s)}] = \boldsymbol{I}_s, \end{aligned}$$

where N_{obs} denotes the set of the observed responses. The above optimization problem can also be solved by a 3-block ADMM algorithm. Define the following class of information criteria,

$$\mathrm{IC}_{obs}(s) = \sum_{(i,j,k) \in N_{obs}} \left(Y_{ijk} (\widehat{\boldsymbol{a}}_i^{(s)})^T \widehat{\boldsymbol{R}}_k^{(s)} \widehat{\boldsymbol{a}}_j^{(s)} - \log[1 + \exp\{(\widehat{\boldsymbol{a}}_i^{(s)})^T \widehat{\boldsymbol{R}}_k^{(s)} \widehat{\boldsymbol{a}}_j^{(s)}\}] \right) - \hat{p} s \kappa(n, K),$$

where $\hat{p} = |N_{obs}|/(n^2 K)$ denotes the percentage of observed responses. Consistency of IC_{obs} can be similarly studied.

Acknowledgments

The authors wish to thank the Associate Editor and anonymous referees for their constructive comments, which lead to significant improvement of this work.

Appendix A. More on the Identifiability Constraint

Let $\pi(\cdot)$ be a permutation function of $\{1, \ldots, n\}$. Alternative to our estimator defined in Equation 3, one may consider

$$\begin{aligned} (\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\}_k) = & \underset{\substack{\boldsymbol{a}_1^{(s)}, \ldots, \boldsymbol{a}_n^{(s)} \in \Theta_a^{(s)} \\ \mathrm{vec}(\boldsymbol{R}_1^{(s)}), \ldots, \mathrm{vec}(\boldsymbol{R}_K^{(s)}) \in \Theta_r^{(s)}}}{\arg\max} \ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k), \\ & \text{subject to} \quad [\boldsymbol{a}_{\pi(1)}^{(s)}, \ldots, \boldsymbol{a}_{\pi(s)}^{(s)}] = \boldsymbol{I}_s, \end{aligned}$$

and the corresponding information criteria

$$\mathrm{IC}_\pi(s) = 2\ell_n(\boldsymbol{Y}; \{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\}_k) - s\kappa(n, K).$$

Since $\ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \boldsymbol{R}_k^{(s)}{}_k) = \ell_n(\boldsymbol{Y}; \{\boldsymbol{C}^{-1}\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{C}^T \boldsymbol{R}_k^{(s)} \boldsymbol{C}\}_k)$ for any invertible matrix $\boldsymbol{C} \in \mathbb{R}^{s \times s}$, $(\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\}_k)$ is also the maximizer of $\ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k)$ subject to the constraint that $[\boldsymbol{a}_{\pi(1)}^{(s)}, \ldots, \boldsymbol{a}_{\pi(s)}^{(s)}]$ is invertible. Similarly, the estimator $(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)$ is the maximizer of $\ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)}\}_i, \{\boldsymbol{R}_k^{(s)}\}_k)$ subject to the constraint that $[\boldsymbol{a}_1^{(s)}, \ldots, \boldsymbol{a}_s^{(s)}]$ is invertible. As a result, we have $\mathrm{IC}(s) = \mathrm{IC}_\pi(s)$ as long as

$$[\widehat{\boldsymbol{a}}_{\pi(1)}^{(s)}, \ldots, \widehat{\boldsymbol{a}}_{\pi(s)}^{(s)}] \text{ is invertible and } [\widehat{\boldsymbol{a}}_{1,\pi}^{(s)}, \ldots, \widehat{\boldsymbol{a}}_{s,\pi}^{(s)}] \text{ is invertible.} \tag{15}$$

However, it remains unknown whether Equation 15 holds or not. Hence, there's no guarantee that $\mathrm{IC}(s) = \mathrm{IC}_\pi(s)$. This means the choice of the identifiability constraint might affect the value of our proposed information criterion.

In the following, we prove

$$\Pr\left(\bigcap_{\substack{\pi: \{1,\ldots,n\} \to \{1,\ldots,n\} \\ |\{\pi(1), \ldots, \pi(n)\}| = n}} \left\{ \underset{s=1,\ldots,s_{\max}}{\arg\max} \{\mathrm{IC}_\pi(s)\} = s_0 \right\} \right) \to 1. \tag{16}$$

This means with probability tending to 1, all the information criteria with different identifiability constraints will select the true model. Therefore, the choice of the identifiability constraint will not affect the performance of our method. For any $s \in \{s_0, \ldots, s_{\max}\}$, let $\boldsymbol{A}_{s,0,\pi} = [\boldsymbol{a}_{\pi(1),0}^{(s)}, \ldots, \boldsymbol{a}_{\pi(s),0}^{(s)}]^T$,

$$\boldsymbol{a}_{i,\pi}^{(s)*} = (\boldsymbol{A}_{s,0,\pi}^{-1})^T \boldsymbol{a}_{i,0}^{(s)} \text{ and } \boldsymbol{R}_{k,\pi}^{(s)*} = \boldsymbol{A}_{s,0,\pi} \boldsymbol{R}_{k,0}^{(s)} \boldsymbol{A}_{s,0,\pi}^T.$$

We need the following condition.
(A4) Assume $\boldsymbol{a}_{i,\pi}^{(s)*} \in \Theta_a^{(s)}$ and $\mathrm{vec}(\boldsymbol{R}_{k,\pi}^{(s)*}) \in \Theta_r^{(s)}$, $\forall i = 1, \ldots, n$, $k = 1, \ldots, K$, $s_0 \le s \le s_{\max}$ and any permutation function $\pi(\cdot)$. In addition, assume $\sup_{\boldsymbol{x} \in \Theta_a^{(s)}} \|\boldsymbol{x}\|_2 \le \omega_a$, $\sup_{\boldsymbol{y} \in \Theta_r^{(s)}} \|\boldsymbol{y}\|_2 \le \omega_r$ for some $\omega_a, \omega_r > 0$.

Corollary 1. *Assume (A2)-(A4) hold,* $K \sim n^{l_0}$ *for some* $0 \le l_0 \le 1$, $s_{\max} = o\{\sqrt{n/\log(nK)}\}$, $\liminf_n n^{(1-l_0)/2} \bar{K} \ge \exp(2\omega_a^2 \omega_r)\sqrt{\log n}$. *Assume* $\kappa(n, K)$ *satisfies Condition* (6). *Then,* (16) *is satisfied.*

Appendix B. More on the Technical Conditions

B.1. Discussion of Condition (A0)

In this section, we show the necessity of (A0) when the matrices $\boldsymbol{R}_{1,0},\ldots,\boldsymbol{R}_{K,0}$ are symmetric. More specifically, when (A0) doesn't hold, we show there exist some $0 \le s < s_0$, $\overline{\boldsymbol{a}}_{1,0},\ldots,\overline{\boldsymbol{a}}_{n,0} \in \mathbb{R}^s$ and $\overline{\boldsymbol{R}}_{1,0},\ldots,\overline{\boldsymbol{R}}_{K,0} \in \mathbb{R}^{s\times s}$ such that

$$\overline{\boldsymbol{a}}_{i,0}^T\overline{\boldsymbol{R}}_{k,0}\overline{\boldsymbol{a}}_{j,0} = \boldsymbol{a}_{i,0}^T\boldsymbol{R}_{k,0}\boldsymbol{a}_{j,0}, \quad \forall 1 \le i,j \le n, 1 \le k \le K. \tag{17}$$

Let's first consider the case where $\mathrm{rank}(\boldsymbol{A}_0) = s$ for some $s < s_0$. Thus, it follows that

$$\boldsymbol{A}_0 = \overline{\boldsymbol{A}}_0\boldsymbol{C},$$

for some $\overline{\boldsymbol{A}}_0 \in \mathbb{R}^{n\times s}$ and $\boldsymbol{C} \in \mathbb{R}^{s\times s_0}$. Set $\overline{\boldsymbol{a}}_{i,0}$ to be the ith row of $\overline{\boldsymbol{A}}_0$ and $\overline{\boldsymbol{R}}_{k,0} = \boldsymbol{C}\boldsymbol{R}_{k,0}\boldsymbol{C}^T$, Equation 17 is thus satisfied. In addition, the new matrix $\overline{\boldsymbol{A}}_0$ shall have full column rank.

Let $\boldsymbol{R}_0 = (\boldsymbol{R}_{1,0}^T,\ldots,\boldsymbol{R}_{K,0}^T)^T$. Consider the case where $\mathrm{rank}(\boldsymbol{R}_0) = s$ for some $s < s_0$. It follows from the singular value decomposition that

$$\boldsymbol{R}_0 = \boldsymbol{U}_0\boldsymbol{\Lambda}_0\boldsymbol{V}_0^T, \tag{18}$$

for some diagonal matrix $\boldsymbol{\Lambda}_0 \in \mathbb{R}^{s\times s}$, and some matrices $\boldsymbol{U}_0 \in \mathbb{R}^{Ks_0\times s}$, $\boldsymbol{V}_0 \in \mathbb{R}^{s_0\times s}$ that satisfy $\boldsymbol{U}_0^T\boldsymbol{U}_0 = \boldsymbol{V}_0^T\boldsymbol{V}_0 = \boldsymbol{I}_s$. Denoted by $\boldsymbol{U}_{k,0}$ the submatrix of $\boldsymbol{U}_0$ formed by rows in $\{(k-1)s_0+1,(k-1)s_0+2,\cdots,ks_0\}$ and columns in $\{1,2,\ldots,s\}$. It follows from Equation 18 that

$$\boldsymbol{R}_{k,0} = \boldsymbol{U}_{k,0}\boldsymbol{\Lambda}_0\boldsymbol{V}_0^T, \quad \forall 1 \le k \le K.$$

Since $\boldsymbol{R}_{k,0}$ is symmetric, we have $\boldsymbol{U}_{k,0}\boldsymbol{\Lambda}_0\boldsymbol{V}_0^T = \boldsymbol{V}_0\boldsymbol{\Lambda}_0\boldsymbol{U}_{k,0}^T = \boldsymbol{R}_{k,0}$. Notice that $\boldsymbol{V}_0^T\boldsymbol{V}_0 = \boldsymbol{I}_s$. It follows that $\boldsymbol{U}_{k,0}\boldsymbol{\Lambda}_0 = \boldsymbol{V}_0\boldsymbol{\Lambda}_0\boldsymbol{U}_{k,0}^T\boldsymbol{V}_0$. Therefore, we have

$$\boldsymbol{R}_{k,0} = \boldsymbol{V}_0\boldsymbol{\Lambda}_0\boldsymbol{U}_{k,0}^T\boldsymbol{V}_0\boldsymbol{V}_0^T. \tag{19}$$

Define $\overline{\boldsymbol{a}}_{i,0} = \boldsymbol{V}_0^T\boldsymbol{a}_{i,0}, \forall 1 \le i \le n$ and $\overline{\boldsymbol{R}}_{k,0} = \boldsymbol{\Lambda}_0\boldsymbol{U}_{k,0}^T\boldsymbol{V}_0$. In view of Equation 19, it is immediate to see that Equation 17 holds. Since $\boldsymbol{V}_0^T\boldsymbol{V}_0 = \boldsymbol{I}_s$, we have $\overline{\boldsymbol{R}}_{k,0} = \boldsymbol{V}_0^T\boldsymbol{V}_0\boldsymbol{\Lambda}_0\boldsymbol{U}_{k,0}^T\boldsymbol{V}_0 = \boldsymbol{V}_0^T\boldsymbol{R}_{k,0}\boldsymbol{V}_0$. As a result, $\overline{\boldsymbol{R}}_{1,0},\ldots,\overline{\boldsymbol{R}}_{K,0}$ are also symmetric. Suppose $\overline{\boldsymbol{R}}_0 = (\overline{\boldsymbol{R}}_{1,0}^T,\cdots,\overline{\boldsymbol{R}}_{K,0}^T)^T$ doesn't have full column rank. Using the same arguments, we can find some $s' < s$, $\widetilde{\boldsymbol{a}}_{1,0},\cdots,\widetilde{\boldsymbol{a}}_{n,0} \in \mathbb{R}^{s'}$, $\widetilde{\boldsymbol{R}}_{1,0},\cdots,\widetilde{\boldsymbol{R}}_{K,0} \in \mathbb{R}^{s'\times s'}$ such that

$$\widetilde{\boldsymbol{a}}_{i,0}^T\widetilde{\boldsymbol{R}}_{k,0}\widetilde{\boldsymbol{a}}_{j,0} = \boldsymbol{a}_{i,0}^T\boldsymbol{R}_{k,0}\boldsymbol{a}_{j,0}, \quad \forall 1 \le i,j \le n, 1 \le k \le K. \tag{20}$$

We may repeat this procedure until we find some $\widetilde{\boldsymbol{a}}_{i,0}$'s, $\widetilde{\boldsymbol{R}}_{k,0}$'s satisfy Equation 20 and that the matrix $(\widetilde{\boldsymbol{R}}_{1,0}^T,\cdots,\widetilde{\boldsymbol{R}}_{K,0}^T)^T$ has full column rank.

B.2. Discussion of Condition (A1)

In this section, we consider an asymptotic framework where $\boldsymbol{a}_{1,0},\ldots,\boldsymbol{a}_{n,0}$ are i.i.d according to some distribution function and show (A1) holds with probability tending to 1. For any q-dimensional vector $\boldsymbol{q}$, let $\|\boldsymbol{q}\|_2$ denote its Euclidean norm. For any $m\times q$ matrix $\boldsymbol{Q}$, $\|\boldsymbol{Q}\|_2$ stands for the spectral norm of $\boldsymbol{Q}$ while $\|\boldsymbol{Q}\|_F$ denotes its Frobenius norm. For simplicity, we assume $\Theta_a = \{\boldsymbol{x} \in \mathbb{R}^{s_0} : \|\boldsymbol{x}\|_2 \le \omega_a\}$ and $\Theta_r = \{\boldsymbol{y} \in \mathbb{R}^{s_0^2} : \|\boldsymbol{y}\|_2 \le \omega_r\}$. Assume $\max_{k=1,\ldots,K}\|\boldsymbol{R}_{k,0}\|_F = O(1)$, $\|\boldsymbol{a}_{1,0}\|_2$ is bounded with probability 1. In addition, assume there exist some constants $c_0, t_0 > 0$ such that

$$\Pr\left(\|[\boldsymbol{a}_{1,0},\ldots,\boldsymbol{a}_{s_0,0}]^{-1}\|_F > t\right) \le c_0t^{-1}, \qquad \forall t \le t_0. \tag{21}$$

When $s_0 = 1$, the above condition is closely related to the margin assumption (Tsybakov, 2004; Audibert and Tsybakov, 2007) in the classification literature. It automatically holds when $a_{1,0}$ has a bounded probability density function.

In the following, we show with proper choice of ω_a and ω_r, (A1) holds with probability tending to 1. By the identifiability constraints, $\|\boldsymbol{a}_i^{(s)*}\|_2 = 1$, $\forall i = 1,\ldots,s$ and $s = s_0,\ldots,s_{\max}$. When $\omega_a,\omega_r \to \infty$, we have for sufficiently large n,

$$\Pr\left(\boldsymbol{a}_i^{(s)*} \in \Omega_a\right) = 1, \qquad \forall 1 \le i \le s, s_0 \le s \le s_{\max}. \tag{22}$$

By the definition of $\boldsymbol{A}_{s,0}$, we have

$$\boldsymbol{A}_{s,0}^{-1} = \begin{pmatrix} \boldsymbol{A}_{s_0,0}^{-1} & \boldsymbol{O}_{s_0\times(s-s_0)} \\ -\boldsymbol{A}_{(s_0+1):s,0}\boldsymbol{A}_{s_0,0}^{-1} & \boldsymbol{I}_{s-s_0} \end{pmatrix}, \tag{23}$$

where $\boldsymbol{A}_{(s_0+1):s,0} = [\boldsymbol{a}_{s_0+1,0},\ldots,\boldsymbol{a}_{s,0}]^T$. It follows that

$$\boldsymbol{a}_i^{(s)*} = \begin{pmatrix} \boldsymbol{A}_{s_0,0}^{-1}\boldsymbol{a}_{i,0} \\ (\underbrace{0,\ldots,0}_{i-s_0-1},1,\underbrace{0,\ldots,0}_{s-i})^T - \boldsymbol{A}_{(s_0+1):s,0}\boldsymbol{A}_{s_0,0}^{-1}\boldsymbol{a}_{i,0} \end{pmatrix}, \qquad \forall s < i \le n, s_0 \le s \le s_{\max}.$$

Under the given conditions, we have $\|\boldsymbol{a}_{i,0}\|_2 \le \omega_0$ with probability 1 for some constant $\omega_0 > 0$. Therefore, we have

$$\max_{\substack{s_0\le s\le s_{\max}\\ s<i\le n}}\|\boldsymbol{a}_i^{(s)*}\|_2 \le \omega_0\|\boldsymbol{A}_{s_0,0}^{-1}\|_2 + 1 + \sqrt{s_{\max}}\omega_0^2\|\boldsymbol{A}_{s_0,0}^{-1}\|_2 \le 1 + (1+\sqrt{s_{\max}}\omega_0)\omega_0\|\boldsymbol{A}_{s_0,0}^{-1}\|_F.$$

By (21), it is immediate to see that

$$\Pr\left(\max_{\substack{s_0\le s\le s_{\max}\\ s<i\le n}}\|\boldsymbol{a}_i^{(s)*}\|_2 > \omega_a\right) \to 0,$$

as long as $\omega_a \gg \sqrt{s_{\max}}$. This together with Equation 22 yields

$$\Pr\left(\max_{\substack{s_0\le s\le s_{\max}\\ 1\le i\le n}}\|\boldsymbol{a}_i^{(s)*}\|_2 > \omega_a\right) \to 0.$$

Combining Equation 23 with the definition of $\boldsymbol{R}_k^{(s)*}$ yields

$$\boldsymbol{R}_k^{(s)*} = \begin{pmatrix} \boldsymbol{A}_{s_0,0}^{-1}\boldsymbol{R}_{k,0}(\boldsymbol{A}_{s_0,0}^{-1})^T & -\boldsymbol{A}_{s_0,0}^{-1}\boldsymbol{R}_{k,0}(\boldsymbol{A}_{s_0,0}^{-1})^T\boldsymbol{A}_{(s_0+1):s,0}^T \\ -\boldsymbol{A}_{(s_0+1):s,0}\boldsymbol{A}_{s_0,0}^{-1}\boldsymbol{R}_{k,0}(\boldsymbol{A}_{s_0,0}^{-1})^T & \boldsymbol{A}_{(s_0+1):s,0}\boldsymbol{A}_{s_0,0}^{-1}\boldsymbol{R}_{k,0}(\boldsymbol{A}_{s_0,0}^{-1})^T\boldsymbol{A}_{(s_0+1):s,0}^T \end{pmatrix}$$

Since $\max_{1\le k\le K}\|\boldsymbol{R}_{k,0}\|_F = O(1)$, $\max_{1\le i\le n}\|\boldsymbol{a}_{i,0}\|_2 = O(1)$, we have

$$\max_{1\le k\le K}\|\boldsymbol{A}_{s_0,0}^{-1}\boldsymbol{R}_{k,0}(\boldsymbol{A}_{s_0,0}^{-1})^T\|_F \le \max_{1\le k\le K}\|\boldsymbol{A}_{s_0,0}^{-1}\|_F^2\|\boldsymbol{R}_{k,0}\|_F = O(\|\boldsymbol{A}_{s_0,0}^{-1}\|_F^2),$$

and

$$\begin{aligned} & \max_{\substack{1\le k\le K\\ s_0\le s\le s_{\max}}} \|\boldsymbol{A}_{(s_0+1):s,0}\boldsymbol{A}_{s_0,0}^{-1}\boldsymbol{R}_{k,0}(\boldsymbol{A}_{s_0,0}^{-1})^T\|_F \\ \le & \max_{\substack{1\le k\le K\\ s_0\le s\le s_{\max}}} \|\boldsymbol{A}_{s_0,0}^{-1}\|_F^2\|\boldsymbol{A}_{(s_0+1):s,0}\|_F\|\boldsymbol{R}_{k,0}\|_F = O(\sqrt{s_{\max}}\|\boldsymbol{A}_{s_0,0}^{-1}\|_F^2), \\ & \max_{\substack{1\le k\le K\\ s_0\le s\le s_{\max}}} \|\boldsymbol{A}_{(s_0+1):s,0}\boldsymbol{A}_{s_0,0}^{-1}\boldsymbol{R}_{k,0}(\boldsymbol{A}_{s_0,0}^{-1})^T\boldsymbol{A}_{(s_0+1):s,0}^T\|_F \\ \le & \max_{\substack{1\le k\le K\\ s_0\le s\le s_{\max}}} \|\boldsymbol{A}_{s_0,0}^{-1}\|_F^2\|\boldsymbol{A}_{(s_0+1):s,0}\|_F^2\|\boldsymbol{R}_{k,0}\|_F = O(s_{\max}\|\boldsymbol{A}_{s_0,0}^{-1}\|_F^2). \end{aligned}$$

By (21), we have $\max_{1\le k\le K, s_0\le s\le s_{\max}}\|\boldsymbol{R}_k^{(s)*}\|_F \le \omega_r$ with probability tending to 1, for any ω_r such that $\omega_r/s_{\max}\to\infty$.

B.3. Discussion of Condition (A2)

Assume the matrix $\mathrm{E}\boldsymbol{a}_{1,0}\boldsymbol{a}_{1,0}^T$ is positive definite. Since s_0 is fixed, it follows from the law of large numbers that

$$\frac{1}{n}\sum_{i=1}^n \boldsymbol{a}_{i,0}\boldsymbol{a}_{i,0}^T \xrightarrow{P} \mathrm{E}\boldsymbol{a}_{1,0}\boldsymbol{a}_{1,0}^T.$$

Therefore, (A2) holds with probability tending to 1.

Appendix C. Proofs

In the following, we provide proofs of Lemma 1, Lemma 2 and Theorem 1. We define

$$\ell_0(\{\boldsymbol{a}_i\}_i,\{\boldsymbol{R}_k\}_k) = \mathrm{E}\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i\}_i,\{\boldsymbol{R}_k\}_k),$$

for any $\boldsymbol{a}_1,\ldots,\boldsymbol{a}_n\in\mathbb{R}^s$, $\boldsymbol{R}_1,\ldots,\boldsymbol{R}_K\in\mathbb{R}^{s\times s}$ and any integer $s\ge 1$. Define $\theta_{ijk} = (\boldsymbol{a}_i^*)^T\boldsymbol{R}_k\boldsymbol{a}_j^*$ and $\hat\theta_{ijk} = \widehat{\boldsymbol{a}}_i^T\widehat{\boldsymbol{R}}_k\widehat{\boldsymbol{a}}_j$.

C.1. Proof of Lemma 1

Assume there exist some $\{\boldsymbol{a}_i\}_i$, $\{\boldsymbol{R}_k\}_k$ such that

$$\mathrm{g}(\boldsymbol{a}_{i,0}^T\boldsymbol{R}_{k,0}\boldsymbol{a}_{j,0}) = \mathrm{g}(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j),\quad \forall i,j,k.$$

Since $\mathrm{g}(\cdot)$ is strictly monotone, we have

$$\boldsymbol{a}_{i,0}^T\boldsymbol{R}_{k,0}\boldsymbol{a}_{j,0} = \boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j,\quad \forall i,j,k,$$

or equivalently,

$$\begin{pmatrix} \boldsymbol{A}\boldsymbol{R}_1 \\ \vdots \\ \boldsymbol{A}\boldsymbol{R}_K \end{pmatrix}\boldsymbol{A}^T = \begin{pmatrix} \boldsymbol{A}_0\boldsymbol{R}_{1,0} \\ \vdots \\ \boldsymbol{A}_0\boldsymbol{R}_{K,0} \end{pmatrix}\boldsymbol{A}_0^T.$$

where $\boldsymbol{A} = [\boldsymbol{a}_1,\boldsymbol{a}_2,\ldots,\boldsymbol{a}_n]^T$. Thus, it follows that

$$\begin{pmatrix} \boldsymbol{A}_0^T\boldsymbol{A}\boldsymbol{R}_1 \\ \vdots \\ \boldsymbol{A}_0^T\boldsymbol{A}\boldsymbol{R}_K \end{pmatrix}\boldsymbol{A}^T = \begin{pmatrix} \boldsymbol{A}_0^T\boldsymbol{A}_0\boldsymbol{R}_{1,0} \\ \vdots \\ \boldsymbol{A}_0^T\boldsymbol{A}_0\boldsymbol{R}_{K,0} \end{pmatrix}\boldsymbol{A}_0^T.$$

By (A0), the matrix $\boldsymbol{A}_0^T\boldsymbol{A}_0$ is invertible. As a result, we have

$$\begin{pmatrix} (\boldsymbol{A}_0^T\boldsymbol{A}_0)^{-1}\boldsymbol{A}_0^T\boldsymbol{A}\boldsymbol{R}_1 \\ \vdots \\ (\boldsymbol{A}_0^T\boldsymbol{A}_0)^{-1}\boldsymbol{A}_0^T\boldsymbol{A}\boldsymbol{R}_K \end{pmatrix}\boldsymbol{A}^T = \begin{pmatrix} \boldsymbol{R}_{1,0} \\ \vdots \\ \boldsymbol{R}_{K,0} \end{pmatrix}\boldsymbol{A}_0^T.$$

Therefore,

$$\left(\sum_{k=1}^K \boldsymbol{R}_{k,0}^T(\boldsymbol{A}_0^T\boldsymbol{A}_0)^{-1}\boldsymbol{A}_0^T\boldsymbol{A}\boldsymbol{R}_1\right)\boldsymbol{A}^T = \left(\sum_{k=1}^K \boldsymbol{R}_{k,0}^T\boldsymbol{R}_{k,0}\right)\boldsymbol{A}_0^T.$$

Notice that the matrix $\sum_{k=1}^K \boldsymbol{R}_{k,0}^T\boldsymbol{R}_{k,0}$ is invertible under Condition (A0). It follows that

$$\underbrace{\left(\sum_{k=1}^K \boldsymbol{R}_{k,0}^T\boldsymbol{R}_{k,0}\right)^{-1}\left(\sum_{k=1}^K \boldsymbol{R}_{k,0}^T(\boldsymbol{A}_0^T\boldsymbol{A}_0)^{-1}\boldsymbol{A}_0^T\boldsymbol{A}\boldsymbol{R}_k\right)}_{\boldsymbol{C}}\boldsymbol{A}^T = \boldsymbol{A}_0^T.$$

By Lemma 5.1 in Banerjee and Roy (2014), we have $\mathrm{rank}(\boldsymbol{C})\ge\mathrm{rank}(\boldsymbol{A}_0) = s_0$. Therefore, $\boldsymbol{C}$ is invertible. It follows that

$$\boldsymbol{A} = \boldsymbol{A}_0(\boldsymbol{C}^T)^{-1}, \tag{24}$$

or equivalently,

$$\boldsymbol{a}_i = \boldsymbol{C}^{-1}\boldsymbol{a}_{i,0},\quad \forall i = 1,\ldots,n.$$

By Equation 24, we obtain $\boldsymbol{A}_0(\boldsymbol{C}^T)^{-1}\boldsymbol{R}_k\boldsymbol{C}^{-1}\boldsymbol{A}_0^T = \boldsymbol{A}_0\boldsymbol{R}_{k,0}\boldsymbol{A}_0^T, \forall k$, and hence

$$\boldsymbol{A}_0^T\boldsymbol{A}_0(\boldsymbol{C}^T)^{-1}\boldsymbol{R}_k\boldsymbol{C}^{-1}\boldsymbol{A}_0^T\boldsymbol{A}_0 = \boldsymbol{A}_0^T\boldsymbol{A}_0\boldsymbol{R}_{k,0}\boldsymbol{A}_0^T\boldsymbol{A}_0,\quad \forall k = 1,\ldots,K.$$

Since $\boldsymbol{A}_0^T\boldsymbol{A}_0$ is invertible, this further implies $(\boldsymbol{C}^T)^{-1}\boldsymbol{R}_k\boldsymbol{C}^{-1} = \boldsymbol{R}_{k,0}, \forall k$, or equivalently,

$$\boldsymbol{R}_k = \boldsymbol{C}^T\boldsymbol{R}_{k,0}\boldsymbol{C},\quad \forall k = 1,\ldots,K.$$

The proof is hence completed.

C.2. Proof of Lemma 2

To prove Lemma 2, we need the following lemma.

Lemma 4 (Mendelson et al. (2008), Lemma 2.3). *Given $d \geq 1$, and $\varepsilon > 0$, we have*

$$N(\varepsilon, B_2^d, \|\cdot\|_2) \leq \left(1 + \frac{2}{\varepsilon}\right)^d,$$

where B_2^d is the unit ball in $\mathbb{R}^d$, and $N(\varepsilon, \cdot, \|\cdot\|_2)$ the covering number with respect to the Euclidean metric (see Definition 2.2.3 in van der Vaart and Wellner (1996) for details).

Under Condition (A1), we have $\ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^{(s)*}\}_i, \{\boldsymbol{R}_k^{(s)*}\}_k) \leq \ell_n(\boldsymbol{Y}; \{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)$ and hence

$$\ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i^*\}_i, \{\boldsymbol{R}_k^*\}_k) \leq \ell_n(\boldsymbol{Y}; \{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k). \tag{25}$$

Besides, we have

$$\max_i \|\widehat{\boldsymbol{a}}_i^{(s)}\|_2 \leq \omega_a, \quad \max_k \|\mathrm{vec}(\widehat{\boldsymbol{R}}_k^{(s)})\|_2 \leq \omega_r, \quad \forall s \in \{s_0, \ldots, s_{\max}\}, \tag{26}$$

and

$$\max_i \|\boldsymbol{a}_i^*\|_2 \leq \omega_a, \quad \max_k \|\mathrm{vec}(\boldsymbol{R}_k^*)\|_2 \leq \omega_r. \tag{27}$$

Therefore,

$$\begin{aligned} &\max_{i,j,k} |(\widehat{\boldsymbol{a}}_i^{(s)})^T \widehat{\boldsymbol{R}}_k^{(s)} \widehat{\boldsymbol{a}}_j^{(s)}| \leq \left(\max_i \|\widehat{\boldsymbol{a}}_i^{(s)}\|_2^2 \max_k \|\widehat{\boldsymbol{R}}_k^{(s)}\|_2\right) \leq \left(\max_i \|\widehat{\boldsymbol{a}}_i^{(s)}\|_2^2 \max_k \|\widehat{\boldsymbol{R}}_k^{(s)}\|_F\right) \\ \leq\ & \left(\max_i \|\widehat{\boldsymbol{a}}_i^{(s)}\|_2^2 \max_k \|\mathrm{vec}(\widehat{\boldsymbol{R}}_k^{(s)})\|_2\right) \leq \omega_a^2 \omega_r, \quad \forall s \in \{s_0, \ldots, s_{\max}\}. \end{aligned} \tag{28}$$

Similarly, we can show

$$\max_{i,j,k} \|(\boldsymbol{a}_i^*)^T \boldsymbol{R}_k^* \boldsymbol{a}_j^*\|_2 \leq \omega_a^2 \omega_r. \tag{29}$$

We define $\theta_{ijk}^* = (\boldsymbol{a}_i^*)^T \boldsymbol{R}_k^* \boldsymbol{a}_j^*$ and $\hat{\theta}_{ijk}^{(s)} = (\widehat{\boldsymbol{a}}_i^{(s)})^T \widehat{\boldsymbol{R}}_k^{(s)} \widehat{\boldsymbol{a}}_j^{(s)}$. It follows from a second-order Taylor expansion that

$$\mathrm{g}(\theta_{ijk}^*)(\theta_{ijk}^* - \hat{\theta}_{ijk}^{(s)}) - \log\{1 + \exp(\theta_{ijk}^*)\} + \log\{1 + \exp(\hat{\theta}_{ijk}^{(s)})\} = \frac{\exp(\tilde{\theta}_{ijk}^{(s)})(\theta_{ijk}^* - \hat{\theta}_{ijk}^{(s)})^2}{2\{1 + \exp(\tilde{\theta}_{ijk}^{(s)})\}^2}, \tag{30}$$

for some $\tilde{\theta}_{ijk}^{(s)}$ lying on the line segment joining θ_{ijk}^* and $\hat{\theta}_{ijk}^{(s)}$. By (28) and (29), we have for any i, j, k and $s \in \{s_0, \ldots, s_{\max}\}$, $|\tilde{\theta}_{ijk}^{(s)}| \leq \omega_a^2 \omega_r$. This together with Equation 30 gives that

$$\begin{aligned} &\mathrm{g}(\theta_{ijk}^*)\theta_{ijk}^* - \log\{1 + \exp(\theta_{ijk}^*)\} - \mathrm{g}(\theta_{ijk}^*)\hat{\theta}_{ijk}^{(s)} + \log\{1 + \exp(\hat{\theta}_{ijk}^{(s)})\} \\ \geq\ & \frac{(\theta_{ijk}^* - \hat{\theta}_{ijk}^{(s)})^2 \exp(\omega_a^2 \omega_r)}{2\{1 + \exp(\omega_a^2 \omega_r)\}^2} \geq \frac{(\theta_{ijk}^* - \hat{\theta}_{ijk}^{(s)})^2 \exp(\omega_a^2 \omega_r)}{8 \exp(2\omega_a^2 \omega_r)} = \frac{(\theta_{ijk}^* - \hat{\theta}_{ijk}^{(s)})^2}{8 \exp(\omega_a^2 \omega_r)}, \end{aligned}$$

and hence

$$\begin{aligned} &\ell_0(\{\boldsymbol{a}_i^*\}_i, \{\boldsymbol{R}_k^*\}_k) - \ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i, \{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) \\ =\ & \sum_{ijk} \left(\mathrm{g}(\theta_{ijk}^*)(\theta_{ijk}^* - \hat{\theta}_{ijk}^{(s)}) - \log \frac{\{1 + \exp(\theta_{ijk}^*)\}}{\{1 + \exp(\hat{\theta}_{ijk}^{(s)})\}}\right) \geq \frac{\sum_{ijk}(\theta_{ijk}^* - \hat{\theta}_{ijk}^{(s)})^2}{8 \exp(\omega_a^2 \omega_r)}. \end{aligned} \tag{31}$$

In the following, we provide an upper bound for

$$\begin{aligned} &\max_{s \in \{1, \ldots, s_{\max}\}} \sup_{\substack{\boldsymbol{a}_i \in \mathbb{R}^s, \boldsymbol{R}_k \in \mathbb{R}^{s \times s} \\ \max_i \|\boldsymbol{a}_i\|_2 \leq \omega_a \\ \max_k \|\mathrm{vec}(\boldsymbol{R}_k)\|_2 \leq \omega_r}} |\ell_0(\{\boldsymbol{a}_i\}_i, \{\boldsymbol{R}_k\}_k) - \ell_n(\boldsymbol{Y}; \{\boldsymbol{a}_i\}_i, \{\boldsymbol{R}_k\}_k)| \\ =\ & \max_{s \in \{1, \ldots, s_{\max}\}} \sup_{\substack{\boldsymbol{a}_i \in \mathbb{R}^s, \boldsymbol{R}_k \in \mathbb{R}^{s \times s} \\ \max_i \|\boldsymbol{a}_i\|_2 \leq \omega_a \\ \max_k \|\mathrm{vec}(\boldsymbol{R}_k)\|_2 \leq \omega_r}} \left|\sum_{ijk} (Y_{ijk} - \pi_{ijk}^*) \boldsymbol{a}_i^T \boldsymbol{R}_k \boldsymbol{a}_j\right|, \end{aligned}$$

where $\pi_{ijk}^* = \exp(\theta_{ijk}^*)/\{1 + \exp(\theta_{ijk}^*)\}$, $\forall i, j, k$.

Let $\varepsilon_a = \omega_a/(nK)^2$ and $\{\bar{\boldsymbol{a}}_1^{(s)}, \ldots, \bar{\boldsymbol{a}}_{N_{s,\varepsilon_a}}^{(s)}\}$ be a minimal ε_a-net of the vector space $(\{\boldsymbol{a} \in \mathbb{R}^s : \|\boldsymbol{a}\|_2 \leq \omega_a\}, \|\cdot\|_2)$. It follows from Lemma 4 that

$$\begin{aligned} N_{s,\varepsilon_a} &= N(\varepsilon_a, \{\boldsymbol{a} \in \mathbb{R}^s : \|\boldsymbol{a}\|_2 \leq \omega_a\}, \|\cdot\|_2) = N(1/(nK)^2, B_2^s, \|\cdot\|_2) \\ &\leq \left(1 + 2n^2K^2\right)^s \leq (3nK)^{2s}. \end{aligned} \tag{32}$$

Let $\varepsilon_r = \omega_r/(nK)^2$, and $\{\bar{\boldsymbol{R}}_1^{(s)}, \ldots, \bar{\boldsymbol{R}}_{N_{s^2,\varepsilon_r}}^{(s)}\}$ be a minimal ε_r-net of the vector space $(\{\boldsymbol{R} \in \mathbb{R}^{s \times s} : \|\mathrm{vec}(\boldsymbol{R})\|_2 \leq \omega\}, \|\cdot\|_F)$. For any $s \times s$ matrices $\boldsymbol{Q}$, we have $\|\boldsymbol{Q}\|_F = \|\mathrm{vec}(\boldsymbol{Q})\|_2$. Similar to (32), we can show that

$$N_{s^2,\varepsilon_r} \leq (3nK)^{2s^2}. \tag{33}$$

Hence, for any $\boldsymbol{a}_i, \boldsymbol{a}_j \in \mathbb{R}^s$ and $\boldsymbol{R}_k \in \mathbb{R}^{s \times s}$ satisfying $\|\boldsymbol{a}_i\|_2, \|\boldsymbol{a}_j\|_2 \leq \omega_a, \|\mathrm{vec}(\boldsymbol{R}_k)\|_2 \leq \omega_r$, there exist some $\bar{\boldsymbol{a}}_{l_i}^{(s)}$, $\bar{\boldsymbol{a}}_{l_j}^{(s)}$, $\bar{\boldsymbol{R}}_{t_k}^{(s)}$, such that

$$\|\boldsymbol{a}_i - \bar{\boldsymbol{a}}_{l_i}^{(s)}\|_2 \leq \frac{\omega_a}{(nK)^2}, \quad \|\boldsymbol{a}_j - \bar{\boldsymbol{a}}_{l_j}^{(s)}\|_2 \leq \frac{\omega_a}{(nK)^2}, \quad \|\boldsymbol{R}_k - \bar{\boldsymbol{R}}_{t_k}^{(s)}\|_F \leq \frac{\omega_r}{(nK)^2}. \tag{34}$$

This further implies

$$\begin{aligned} &|(\boldsymbol{a}_i)^T \boldsymbol{R}_k \boldsymbol{a}_j - (\bar{\boldsymbol{a}}_{l_i}^{(s)})^T \bar{\boldsymbol{R}}_{t_k}^{(s)} \bar{\boldsymbol{a}}_{l_j}^{(s)}| \leq |(\boldsymbol{a}_i)^T \boldsymbol{R}_k \boldsymbol{a}_j - (\bar{\boldsymbol{a}}_{l_i}^{(s)})^T \boldsymbol{R}_k \boldsymbol{a}_j| \\ +\ & |(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T \boldsymbol{R}_k \boldsymbol{a}_j - (\bar{\boldsymbol{a}}_{l_i}^{(s)})^T \bar{\boldsymbol{R}}_{t_k}^{(s)} \boldsymbol{a}_j| + |(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T \bar{\boldsymbol{R}}_{t_k}^{(s)} \boldsymbol{a}_j - (\bar{\boldsymbol{a}}_{l_i}^{(s)})^T \bar{\boldsymbol{R}}_{t_k}^{(s)} \bar{\boldsymbol{a}}_{l_j}^{(s)}| \\ \leq\ & \|\boldsymbol{a}_i - \bar{\boldsymbol{a}}_{l_i}^{(s)}\|_2 \|\boldsymbol{R}_k\|_2 \|\boldsymbol{a}_j\|_2 + \|\bar{\boldsymbol{a}}_{l_i}^{(s)}\|_2 \|\boldsymbol{R}_k - \bar{\boldsymbol{R}}_{t_k}^{(s)}\|_2 \|\boldsymbol{a}_j\|_2 + \|\bar{\boldsymbol{a}}_i^{(s)}\|_2 \|\bar{\boldsymbol{R}}_{t_k}^{(s)}\|_2 \|\boldsymbol{a}_j - \bar{\boldsymbol{a}}_{l_j}^{(s)}\|_2 \\ \leq\ & \|\boldsymbol{a}_i - \bar{\boldsymbol{a}}_{l_i}^{(s)}\|_2 \|\boldsymbol{R}_k\|_F \|\boldsymbol{a}_j\|_2 + \|\bar{\boldsymbol{a}}_{l_i}^{(s)}\|_2 \|\boldsymbol{R}_k - \bar{\boldsymbol{R}}_{t_k}^{(s)}\|_F \|\boldsymbol{a}_j\|_2 + \|\bar{\boldsymbol{a}}_i^{(s)}\|_2 \|\bar{\boldsymbol{R}}_{t_k}^{(s)}\|_F \|\boldsymbol{a}_j - \bar{\boldsymbol{a}}_{l_j}^{(s)}\|_2 \\ \leq\ & 2\frac{\omega_a}{(nK)^2} \omega_a \omega_r + \frac{\omega_r}{(nK)^2} \omega_a^2 = \frac{3\omega_a^2 \omega_r}{(nK)^2}. \end{aligned} \tag{35}$$

Therefore, we have

$$\max_{s\in\{1,\ldots,s_{\max}\}}\frac{1}{s}\sup_{\substack{\boldsymbol{a}_i\in\mathbb{R}^s,\boldsymbol{R}_k\in\mathbb{R}^{s\times s}\\ \max_i\|\boldsymbol{a}_i\|_2\le\omega_a\\ \max_k\|\mathrm{vec}(\boldsymbol{R}_k)\|_2\le\omega_r}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j\right| \tag{36}$$

$$\le \max_{s\in\{1,\ldots,s_{\max}\}}\frac{1}{s}\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\right|+\sum_{ijk}|Y_{ijk}-\pi^*_{ijk}|\frac{3\omega_a^2\omega_r}{n^2K^2}$$

$$\le \max_{s\in\{1,\ldots,s_{\max}\}}\frac{1}{s}\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\right|+\frac{3\omega_a^2\omega_r}{K}.$$

By Bernstein's inequality (van der Vaart and Wellner, 1996, Lemma 2.2.9), we obtain for any $t>0$,

$$\max_{\substack{s\in\{1,\ldots,s_{\max}\}\\ l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\Pr\left(\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\right|>t\right)\le 2\exp\left(-\frac{t^2}{2\sigma_0^2+2M_0t/3}\right), \tag{37}$$

where

$$M_0 = \max_{s\in\{1,\ldots,s_{\max}\}}\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\max_{ijk}|Y_{ijk}-\pi^*_{ijk}||(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}|,$$

$$\sigma_0^2 = \max_{s\in\{1,\ldots,s_{\max}\}}\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\sum_{ijk}\mathrm{E}|Y_{ijk}-\pi^*_{ijk}|^2|(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}|^2.$$

With some calculations, we can show that

$$M_0 \le \max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\|\bar{\boldsymbol{a}}_{l_i}^{(s)}\|_2\|\bar{\boldsymbol{R}}_{t_k}^{(s)}\|_2\|\bar{\boldsymbol{a}}_{l_j}^{(s)}\|_2\le\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\|\bar{\boldsymbol{a}}_{l_i}^{(s)}\|_2\|\bar{\boldsymbol{R}}_{t_k}^{(s)}\|_F\|\bar{\boldsymbol{a}}_{l_j}^{(s)}\|_2$$

$$\le \max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\|\bar{\boldsymbol{a}}_{l_i}^{(s)}\|_2\|\mathrm{vec}(\bar{\boldsymbol{R}}_{t_k}^{(s)})\|_2\|\bar{\boldsymbol{a}}_{l_j}^{(s)}\|_2\le\omega_a^2\omega_r, \tag{38}$$

and

$$\sigma_0^2 \le \max_{s\in\{1,\ldots,s_{\max}\}}\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\sum_{ijk}\mathrm{E}|Y_{ijk}-\pi^*_{ijk}|^2\|\bar{\boldsymbol{a}}_{l_i}^{(s)}\|_2^2\|\mathrm{vec}(\bar{\boldsymbol{R}}_{t_k}^{(s)})\|_2^2\|\bar{\boldsymbol{a}}_{l_j}^{(s)}\|_2^2$$

$$\le \omega_a^4\omega_r^2\sum_{ijk}\mathrm{E}|Y_{ijk}-\pi^*_{ijk}|^2\le\omega_a^4\omega_r^2n^2K. \tag{39}$$

Let $t_s=5\omega_a^2\omega_r sn\sqrt{K\max(n,K)\log(nK)}$, we have

$$\max_{s\in\{1,\ldots,s_{\max}\}}\Pr\left(\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\right|>t_s\right)$$

$$\le \max_{\substack{s\in\{1,\ldots,s_{\max}\}\\ l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}N_{s,\varepsilon_a}^nN_{s^2,\varepsilon_r}^K\Pr\left(\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\right|>t_s\right)$$

$$\le \max_{\substack{s\in\{1,\ldots,s_{\max}\}\\ l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}(3nK)^{2sn+2s^2K}\Pr\left(\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\right|>t_s\right)$$

$$\le \max_{s\in\{1,\ldots,s_{\max}\}}2\exp\left(-\frac{25\omega_a^4\omega_r^2s^2n^2K\max(n,K)\log(nK)}{2\omega_a^4\omega_r^2\{n^2K+sn\sqrt{K\max(n,K)\log(nK)}/3\}}+(2sn+2s^2K)\log(3nK)\right)$$

$$= \max_{s\in\{1,\ldots,s_{\max}\}}2\exp\left(-\frac{25s^2\max(n,K)\log(nK)}{2+2s\sqrt{\max(n,K)\log(nK)}/(3n\sqrt{K})}+(2sn+2s^2K)\log(3nK)\right), \tag{40}$$

where the first inequality is due to Bonferroni's inequality, the second inequality follows by (32) and (33), the third inequality is due to (37)-(39).

Under the given conditions, we have $s\le s_{\max}=o\{n/\log(nK)\}$ and hence

$$\frac{s\sqrt{\max(n,K)\log(nK)}}{n\sqrt{K}}\le\frac{s\sqrt{n\log(nK)}}{n\sqrt{K}}+\frac{s\sqrt{K\log(nK)}}{n\sqrt{K}}\le\frac{s_{\max}\sqrt{\log(nK)}}{\sqrt{n}}+\frac{s_{\max}\sqrt{\log(nK)}}{n}\ll 1.$$

It follows that for any $s\in\{1,\ldots,s_{\max}\}$,

$$\Pr\left(\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\right|>5\omega_a^2\omega_r sn\sqrt{K\max(n,K)\log(nK)}\right)$$

$$\le 2\exp\left(-\frac{25s^2\max(n,K)\log(nK)}{2+1}+4s^2\max(n,K)\log(3nK)\right)$$

$$\le 2\exp\{-4s^2\max(n,K)\log(nK)\}\le 2\exp\{-4s^2n\log(nK)\}.$$

By Bonferroni's inequality, we have

$$\Pr\left(\bigcup_{s\in\{1,\ldots,s_{\max}\}}\left\{\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\right|>t_s\right\}\right)$$

$$\le \sum_{s=1}^{s_{\max}}2\exp\{-4s^2n\log(nK)\}\le\sum_{s=1}^{+\infty}2\exp\{-4sn\log(nK)\}\le\frac{2\exp\{-4n\log(nK)\}}{1-\exp\{-4n\log(nK)\}}\to 0.$$

This together with (36) implies that

$$\max_{s\in\{1,\ldots,s_{\max}\}} \frac{1}{s} \sup_{\substack{\boldsymbol{a}_i\in\mathbb{R}^s,\boldsymbol{R}_k\in\mathbb{R}^{s\times s}\\ \max_i\|\boldsymbol{a}_i\|_2\le\omega_a\\ \max_k\|\mathrm{vec}(\boldsymbol{R}_k)\|_2\le\omega_r}} \left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j\right| \le 6\omega_a^2\omega_r n\sqrt{K\max(n,K)\log(nK)}, \quad (41)$$

with probability tending to 1. Combining this with (26) and (27), we obtain with probability tending to 1,

$$\begin{aligned} &\max\left(\max_s\left|\ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)-\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\right|/s,\right. \quad (42)\\ &\left.|\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)-\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)|/s_0\right) \le 6\omega_a^2\omega_r n\sqrt{K\max(n,K)\log(nK)}. \end{aligned}$$

Therefore, it follows from (31) that

$$\begin{aligned} &\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{1}{s}\sum_{ijk}(\theta^*_{ijk}-\hat\theta^{(s)}_{ijk})^2 \quad (43)\\ \le\ &\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{8\exp(\omega_a^2\omega_r)}{s}\left(\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)-\ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\right)\\ \le\ &\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{8\exp(\omega_a^2\omega_r)}{s}\left|\ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)-\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\right|\\ +\ &\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{8\exp(\omega_a^2\omega_r)}{s}\left|\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)-\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)\right|\\ +\ &\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{8\exp(\omega_a^2\omega_r)}{s}\left(\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)-\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\right)\\ \le\ &\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{8\exp(\omega_a^2\omega_r)}{s}\left|\ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)-\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\right|\\ +\ &\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{8\exp(\omega_a^2\omega_r)}{s}\left|\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)-\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)\right|\\ \le\ &96\omega_a^2\omega_r\exp(\omega_a^2\omega_r)n\sqrt{K\max(n,K)\log(nK)}, \end{aligned}$$

where the third inequality is due to that $\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\ge\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)$, for all $s\in\{s_0,\ldots,s_{\max}\}$.

Let $r_{n,K}=(n+K)^{-1/2}(\log n+\log K)^{-1/2}$. As $n\to\infty$, we have

$$r_{n,K}^2 n\sqrt{K\max(n,K)\log(nK)} \le r_{n,K}^2 n\sqrt{K(n+K)\log(nK)} = \frac{n\sqrt{K}}{\sqrt{(n+K)\log(nK)}} \ll \sqrt{nK}.$$

Since $\omega_a^2\omega_r\le\exp(\omega_a^2\omega_r)$, it follows from (43) that

$$\Pr\left(\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{r_{n,K}^2}{s^2\exp(2\omega_a^2\omega_r)}\sum_{ijk}(\theta^*_{ijk}-\hat\theta^{(s)}_{ijk})^2\ge\sqrt{nK}\right)\to 0. \quad (44)$$

For any integer $m\ge 1$, define

$$\mathbb{S}_m^{(s)} = \left\{(\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k): m-1<\frac{r_{n,K}}{s\exp(\omega_a^2\omega_r)}\sqrt{\sum_{ijk}\{\theta^*_{ijk}-(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}\}^2}\le m,\right.$$
$$\left.\boldsymbol{a}_i^{(s)}\in\mathbb{R}^s,\forall i,\boldsymbol{R}_k^{(s)}\in\mathbb{R}^{s\times s},\forall k,\max_i\|\boldsymbol{a}_i^{(s)}\|_2\le\omega_a,\max_k\|\mathrm{vec}(\boldsymbol{R}_k^{(s)})\|_2\le\omega_r\right\}.$$

For any $(\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k)\in\mathbb{S}_m^{(s)}$, similar to (31), we can show that

$$\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)-\ell_0(\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k)\ge\frac{\sum_{ijk}\{\theta^*_{ijk}-(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}\}^2}{8\exp(\omega_a^2\omega_r)}\ge\frac{(m-1)^2s^2r_{n,K}^{-2}}{8\exp(-\omega_a^2\omega_r)}. \quad (45)$$

The event $(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\in\mathbb{S}_m^{(s)}$ implies that

$$\sup_{(\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k)\in\mathbb{S}_m^{(s)}}\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k)\ge\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k).$$

It follows from (45) that

$$\sup_{(\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k)\in\mathbb{S}_m^{(s)}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\{\theta^*_{ijk}-(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}\}\right|\ge\frac{(m-1)^2s^2\exp(\omega_a^2\omega_r)}{8r_{n,K}^2}. \quad (46)$$

For any $\{l_i\}_i$ and $\{t_k\}_k$ satisfying (34), it follows from (35) that

$$\begin{aligned} &\sum_{ijk}\{\theta^*_{ijk}-(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\}^2\\ \le\ &\sum_{ijk}\{\theta^*_{ijk}-(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}+(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}-(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\}^2\\ \le\ &2\sum_{ijk}\left(\{\theta^*_{ijk}-(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}\}^2+\{(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}-(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\}^2\right)\\ \le\ &\frac{2m^2s^2\exp(2\omega_a^2\omega_r)}{r_{n,K}^2}+\frac{6\omega_a^2\omega_r}{K}\le\frac{3m^2s^2\exp(2\omega_a^2\omega_r)}{r_{n,K}^2}. \end{aligned} \quad (47)$$

Let $\Lambda_m^{(s)}=\{(\{l_i\}_i,\{t_k\}_k):\sum_{ijk}\{\theta^*_{ijk}-(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}\}^2\le 3m^2s^2\exp(2\omega_a^2\omega_r)/r_{n,K}^2\}$. Similar to (36), we can show

$$\begin{aligned} &\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{1}{s^2}\sup_{(\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k)\in\mathbb{S}_m^{(s)}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\{\theta^*_{ijk}-(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}\}\right|\\ \le\ &\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{1}{s^2}\max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}\\ (\{l_i\}_i,\{t_k\}_k)\in\Lambda_m^{(s)}}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\{(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}-\theta^*_{ijk}\}\right|+\frac{3\omega_a^2\omega_r}{K}. \end{aligned}$$

Under the event defined in (46), for any $m \geq 9$, we have

$$\begin{aligned}
&\max_{s\in\{s_0,\ldots,s_{\max}\}} \frac{1}{s^2} \max_{\substack{l_1,\ldots,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,\ldots,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}\\ (\{l_i\}_i,\{t_k\}_k)\in\Lambda_m^{(s)}}} \left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\{(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}-\theta^*_{ijk}\}\right| \\
\geq\ & \frac{(m-1)^2\exp(\omega_a^2\omega_r)}{8r_{n,K}^2}-\frac{3\omega_a^2\omega_r}{K} \geq \frac{(m-1)^2\exp(\omega_a^2\omega_r)}{16r_{n,K}^2}+\frac{4\exp(\omega_a^2\omega_r)}{r_{n,K}^2}-\frac{3\omega_a^2\omega_r}{K} \geq \frac{(m-1)^2\exp(\omega_a^2\omega_r)}{16r_{n,K}^2}.
\end{aligned}$$

Define

$$(\sigma_m^{(s)})^2 = \max_{\substack{l_1,..,l_n\in\{1,\ldots,N_{s,\varepsilon_{n,K}}\}\\ t_1,..,t_K\in\{1,\ldots,N_{s^2,\varepsilon_{n,K}}\}\\ (\{l_i\}_i,\{t_k\}_k)\in\Lambda_m^{(s)}}} \sum_{ijk}\mathrm{E}|Y_{ijk}-\pi^*_{ijk}|^2\{(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}-\theta^*_{ijk}\}^2.$$

By (47), it is immediate to see that

$$(\sigma_m^{(s)})^2 \leq 3m^2s^2\exp(2\omega_a^2\omega_r)/r_{n,K}^2. \tag{48}$$

Similar to (37) and (40), we can show there exist some constants $J_0>0$, $K_0>0$ such that for any $m\geq J_0$ and any s such that $s_0\leq s\leq s_{\max}$,

$$\begin{aligned}
&\Pr\left(\frac{1}{s^2}\max_{\substack{l_1,..,l_n\in\{1,\ldots,N_{s,\varepsilon_a}\}\\ t_1,..,t_K\in\{1,\ldots,N_{s^2,\varepsilon_r}\}\\ (\{l_i\}_i,\{t_k\}_k)\in\Lambda_m^{(s)}}}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\{(\bar{\boldsymbol{a}}_{l_i}^{(s)})^T\bar{\boldsymbol{R}}_{t_k}^{(s)}\bar{\boldsymbol{a}}_{l_j}^{(s)}-\theta^*_{ijk}\}\right| \geq \frac{(m-1)^2\exp(\omega_a^2\omega_r)}{16r_{n,K}^2}\right) \\
\leq\ & 2\exp\left(-\frac{(m-1)^4s^4\exp(2\omega_a^2\omega_r)/(256r_{n,K}^4)}{2(\sigma_m^{(s)})^2+(m-1)^2s^2\omega_a^2\omega_r\exp(\omega_a\omega_r)M_0/(24r_{n,K}^2)}+(2sn+2s^2K)\log(3nK)\right) \\
\leq\ & 2\exp\left(-\frac{(m-1)^4s^4/(256r_{n,K}^4)}{6m^2s^2/r_{n,K}^2+(m-1)^2s^2/(24r_{n,K}^2)}+(2sn+2s^2K)\log(3nK)\right) \leq 2\exp(-K_0m^2s^2r_{n,K}^{-2}),
\end{aligned}$$

where the second inequality is due to (38), (48) and the fact that $\omega_a^2\omega_r\leq\exp(\omega_a^2\omega_r)$. This yields

$$\Pr\left((\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\in\mathbb{S}_m^{(s)}\right)\leq 2\exp(-K_0m^2s^2r_{n,K}^{-2}). \tag{49}$$

Let $J_{n,K}$ be the integer such that $J_{n,K}-1\leq(nK)^{1/4}\leq J_{n,K}$. In view of (44), we have for any $m\geq J_0$,

$$\begin{aligned}
&\Pr\left(\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{r_{n,K}}{s\exp(\omega_a^2\omega_r)}\sqrt{\sum_{ijk}(\theta^*_{ijk}-\hat{\theta}^{(s)}_{ijk})^2}>m\right) \leq \sum_{s=s_0}^{s_{\max}}\sum_{l=m}^{J_{n,K}}2\exp(-K_0l^2s^2r_{n,K}^{-2})+o(1) \\
&\leq \sum_{s=1}^{+\infty}\sum_{l=1}^{+\infty}2\exp(-K_0lmss_0r_{n,K}^{-2})+o(1) \leq \frac{2\exp(-s_0mK_0r_{n,K}^{-2})}{1-2\exp(-s_0mK_0r_{n,K}^{-2})}+o(1)=o(1).
\end{aligned}$$

This implies there exists some constant $C_0>0$ such that the following event occurs with probability tending to 1,

$$\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{1}{s^2}\sum_{ijk}(\hat{\theta}^{(s)}_{ijk}-\theta^*_{ijk})^2\leq C_0\exp(\omega_a^2\omega_r)r_{n,K}^{-2}, \tag{50}$$

The proof is hence completed.

C.3. Proof of Lemma 3

For any $s<s_0$, we have

$$\begin{aligned}
&\sum_{ijk}\left((\widehat{\boldsymbol{a}}_i^{(s)})^T\widehat{\boldsymbol{R}}_k^{(s)}\widehat{\boldsymbol{a}}_j^{(s)}-(\boldsymbol{a}_i^*)^T\boldsymbol{R}_k^*\boldsymbol{a}_j^*\right)^2 \qquad (51)\\
\geq\ & \inf_{\substack{\boldsymbol{a}_1^{(s)},\ldots,\boldsymbol{a}_n^{(s)}\in\mathbb{R}^s\\ \boldsymbol{R}_1^{(s)},\ldots,\boldsymbol{R}_K^{(s)}\in\mathbb{R}^{s\times s}}}\sum_{ijk}\left((\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}-(\boldsymbol{a}_i^*)^T\boldsymbol{R}_k^*\boldsymbol{a}_j^*\right)^2 \\
=\ & \inf_{\substack{\boldsymbol{A}^{(s)}\in\mathbb{R}^{n\times s}\\ \boldsymbol{R}_1^{(s)},\ldots,\boldsymbol{R}_K^{(s)}\in\mathbb{R}^{s\times s}}}\sum_{k=1}^K\|\boldsymbol{A}^{(s)}\boldsymbol{R}_k^{(s)}(\boldsymbol{A}^{(s)})^T-\boldsymbol{A}_0\boldsymbol{R}_{k,0}\boldsymbol{A}_0^T\|_F^2 \\
=\ & \inf_{\substack{\boldsymbol{A}^{(s)}\in\mathbb{R}^{n\times s}\\ \boldsymbol{R}_1^{(s)},\ldots,\boldsymbol{R}_K^{(s)}\in\mathbb{R}^{s\times s}}}\left\|\begin{pmatrix}\boldsymbol{A}^{(s)}\boldsymbol{R}_1^{(s)}\\ \vdots\\ \boldsymbol{A}^{(s)}\boldsymbol{R}_K^{(s)}\end{pmatrix}(\boldsymbol{A}^{(s)})^T-\underbrace{\begin{pmatrix}\boldsymbol{A}_0\boldsymbol{R}_{1,0}\\ \vdots\\ \boldsymbol{A}_0\boldsymbol{R}_{K,0}\end{pmatrix}}_{\boldsymbol{B}_0}\boldsymbol{A}_0^T\right\|_F^2 \\
\geq\ & \inf_{\boldsymbol{A}^{(s)}\in\mathbb{R}^{n\times s},\boldsymbol{B}^{(s)}\in\mathbb{R}^{nK\times s}}\left\|\boldsymbol{B}^{(s)}(\boldsymbol{A}^{(s)})^T-\boldsymbol{B}_0\boldsymbol{A}_0^T\right\|_F^2.
\end{aligned}$$

Define

$$(\widehat{\boldsymbol{A}}^{(s)},\widehat{\boldsymbol{B}}^{(s)})=\arg\min_{\substack{\boldsymbol{A}^{(s)}\in\mathbb{R}^{n\times s}\\ \boldsymbol{B}^{(s)}\in\mathbb{R}^{nK\times s}}}\|\boldsymbol{B}^{(s)}(\boldsymbol{A}^{(s)})^T-\boldsymbol{B}_0\boldsymbol{A}_0^T\|_F^2.$$

The above minimizers are not unique. Notice that $\mathrm{rank}(\boldsymbol{B}_0\boldsymbol{A}_0^T)\leq\mathrm{rank}(\boldsymbol{A}_0^T)\leq s_0$. Assume $\boldsymbol{B}_0\boldsymbol{A}_0^T$ has the following singular value decomposition,

$$\boldsymbol{B}_0\boldsymbol{A}_0^T=n\boldsymbol{U}_n\boldsymbol{\Lambda}_n\boldsymbol{V}_n^T,$$

for some $\boldsymbol{U}_n\in\mathbb{R}^{nK\times s_0},\boldsymbol{V}_n\in\mathbb{R}^{n\times s_0}$ such that $\boldsymbol{U}_n^T\boldsymbol{U}_n=\boldsymbol{V}_n^T\boldsymbol{V}_n=\boldsymbol{I}_{s_0}$, and some diagonal matrix

$$\boldsymbol{\Lambda}_n=\mathrm{diag}(\lambda_n^{(1)},\lambda_n^{(2)},\ldots,\lambda_n^{(s_0)})$$

such that $|\lambda_n^{(1)}|\geq|\lambda_n^{(2)}|\geq\cdots\geq|\lambda_n^{(s_0)}|$. Then one solution is given by

$$\widehat{\boldsymbol{A}}^{(s)}=n\boldsymbol{V}_n\boldsymbol{\Lambda}_n\boldsymbol{U}_n^T\boldsymbol{U}_n^{(s)},\quad \widehat{\boldsymbol{B}}^{(s)}=\boldsymbol{U}_n^{(s)},$$

where $\boldsymbol{U}_n^{(s)}$ is the submatrix of $\boldsymbol{U}_n$ formed by its first s columns.

Since $\boldsymbol{U}_n^T\boldsymbol{U}_n = \boldsymbol{I}_{s_0}$, we have

$$\begin{aligned}\widehat{\boldsymbol{B}}^{(s)}(\widehat{\boldsymbol{A}}^{(s)})^T &= n\boldsymbol{U}_n^{(s)}(\boldsymbol{U}_n^{(s)})^T\boldsymbol{U}_n\boldsymbol{\Lambda}_n\boldsymbol{V}_n^T = n\boldsymbol{U}_n^{(s)}(\boldsymbol{I}_s, \boldsymbol{O}_{s,s_0-s})\boldsymbol{\Lambda}_n\boldsymbol{V}_n^T \\ &= n\boldsymbol{U}_n\begin{pmatrix}\boldsymbol{I}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{O}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{\Lambda}_n\boldsymbol{V}_n^T,\end{aligned}$$

and hence

$$\begin{aligned}\widehat{\boldsymbol{B}}^{(s)}(\widehat{\boldsymbol{A}}^{(s)})^T - \boldsymbol{B}_0\boldsymbol{A}_0^T &= n\boldsymbol{U}_n\begin{pmatrix}\boldsymbol{I}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{O}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{\Lambda}_n\boldsymbol{V}_n^T - n\boldsymbol{U}_n\boldsymbol{I}_{s_0}\boldsymbol{\Lambda}_n\boldsymbol{V}_n^T \\ &= -n\boldsymbol{U}_n\begin{pmatrix}\boldsymbol{O}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{I}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{\Lambda}_n\boldsymbol{V}_n^T.\end{aligned}$$

This together with (51) implies that

$$\begin{aligned}&\sum_{ijk}\left((\widehat{\boldsymbol{a}}_i^{(s)})^T\widehat{\boldsymbol{R}}_k^{(s)}\widehat{\boldsymbol{a}}_j^{(s)} - (\boldsymbol{a}_i^*)^T\boldsymbol{R}_k^*\boldsymbol{a}_j^*\right)^2 \geq \|\widehat{\boldsymbol{B}}^{(s)}(\widehat{\boldsymbol{A}}^{(s)})^T - \boldsymbol{B}_0\boldsymbol{A}_0^T\|_F^2 \\ \geq\ & n^2\text{trace}\left(\boldsymbol{U}_n\begin{pmatrix}\boldsymbol{O}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{I}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{\Lambda}_n\boldsymbol{V}_n^T\boldsymbol{V}_n\boldsymbol{\Lambda}_n\begin{pmatrix}\boldsymbol{O}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{I}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{U}_n^T\right) \\ =\ & n^2\text{trace}\left(\boldsymbol{U}_n\begin{pmatrix}\boldsymbol{O}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{I}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{\Lambda}_n^2\begin{pmatrix}\boldsymbol{O}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{I}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{U}_n^T\right) \qquad (52) \\ =\ & n^2\text{trace}\left(\boldsymbol{\Lambda}_n\begin{pmatrix}\boldsymbol{O}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{I}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{U}_n^T\boldsymbol{U}_n\begin{pmatrix}\boldsymbol{O}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{I}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{\Lambda}_n\right) \\ =\ & n^2\text{trace}\left(\boldsymbol{\Lambda}_n\begin{pmatrix}\boldsymbol{O}_s & \boldsymbol{O}_{s,s_0-s}\\ \boldsymbol{O}_{s_0-s,s} & \boldsymbol{I}_{s_0-s,s_0-s}\end{pmatrix}\boldsymbol{\Lambda}_n\right) \geq n^2(\lambda_n^{(s_0)})^2, \qquad (53)\end{aligned}$$

where (52) is due to that $\boldsymbol{V}_n^T\boldsymbol{V}_n = \boldsymbol{I}_{s_0}$ and the equality in (53) is due to that $\boldsymbol{U}_n^T\boldsymbol{U}_n = \boldsymbol{I}_{s_0}$.

To summarize, we've shown

$$\sum_{ijk}\left((\widehat{\boldsymbol{a}}_i^{(s)})^T\widehat{\boldsymbol{R}}_k^{(s)}\widehat{\boldsymbol{a}}_j^{(s)} - (\boldsymbol{a}_i^*)^T\boldsymbol{R}_k^*\boldsymbol{a}_j^*\right)^2 \geq n^2(\lambda_n^{(s_0)})^2. \qquad (54)$$

In the following, we provide a lower bound for $(\lambda_n^{(s_0)})^2$. By definition, $(\lambda_n^{(s_0)})^2$ is the s_0-th largest eigenvalue of

$$\frac{1}{n^2}\boldsymbol{A}_0\boldsymbol{B}_0^T\boldsymbol{B}_0\boldsymbol{A}_0^T = \frac{1}{n^2}\sum_{k=1}^K\boldsymbol{A}_0\boldsymbol{R}_{k,0}^T\boldsymbol{A}_0^T\boldsymbol{A}_0\boldsymbol{R}_{k,0}\boldsymbol{A}_0^T.$$

We first provide an lower bound for $\lambda_{\min}(\sum_{k=1}^K\boldsymbol{R}_{k,0}^T\boldsymbol{A}_0^T\boldsymbol{A}_0\boldsymbol{R}_{k,0}/n)$. Let $\boldsymbol{\Sigma}_A = \boldsymbol{A}_0^T\boldsymbol{A}_0/n$. Consider the following eigenvalue decomposition:

$$\boldsymbol{\Sigma}_A = \boldsymbol{U}_A\boldsymbol{\Lambda}_A\boldsymbol{U}_A^T,$$

for some orthogonal matrix $\boldsymbol{U}_A$ and some diagonal matrix $\boldsymbol{\Lambda}_A$. Under Assumption (A2), the matrix

$$\boldsymbol{\Sigma}_A - \bar{c}\boldsymbol{I}_{s_0}$$

is positive semidefinite. As a result, the matrix

$$\sum_{k=1}^K\boldsymbol{R}_{k,0}^T(\boldsymbol{\Sigma}_A - \bar{c}\boldsymbol{I}_{s_0})\boldsymbol{R}_{k,0}$$

is positive semidefinite. Therefore, we have

$$\begin{aligned}&\lambda_{\min}\left(\sum_{k=1}^K\boldsymbol{R}_{k,0}^T\boldsymbol{\Sigma}_A\boldsymbol{R}_{k,0}\right) = \inf_{\boldsymbol{a}_0\in\mathbb{R}^{s_0},\|\boldsymbol{a}_0\|_2=1}\boldsymbol{a}_0^T\left(\sum_{k=1}^K\boldsymbol{R}_{k,0}^T\boldsymbol{\Sigma}_A\boldsymbol{R}_{k,0}\right)\boldsymbol{a}_0 \qquad (55) \\ =\ & \inf_{\boldsymbol{a}_0\in\mathbb{R}^{s_0},\|\boldsymbol{a}_0\|_2=1}\left\{\boldsymbol{a}_0^T\left(\sum_{k=1}^K\boldsymbol{R}_{k,0}^T(\boldsymbol{\Sigma}_A - \bar{c}\boldsymbol{I}_{s_0})\boldsymbol{R}_{k,0}\right)\boldsymbol{a}_0 + \bar{c}\boldsymbol{a}_0^T\left(\sum_{k=1}^K\boldsymbol{R}_{k,0}^T\boldsymbol{R}_{k,0}\right)\boldsymbol{a}_0\right\} \\ \geq\ & \bar{c}\inf_{\boldsymbol{a}_0\in\mathbb{R}^{s_0},\|\boldsymbol{a}_0\|_2=1}\boldsymbol{a}_0^T\left(\sum_{k=1}^K\boldsymbol{R}_{k,0}^T\boldsymbol{R}_{k,0}\right)\boldsymbol{a}_0 = \bar{c}\lambda_{\min}\left(\sum_{k=1}^K\boldsymbol{R}_{k,0}^T\boldsymbol{R}_{k,0}\right) = \bar{c}\bar{K}.\end{aligned}$$

By the eigenvalue decomposition, we have

$$\sum_{k=1}^K\boldsymbol{R}_{k,0}^T\boldsymbol{\Sigma}_A\boldsymbol{R}_{k,0} = \boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}\boldsymbol{U}_{RA},$$

for some orthogonal matrix $\boldsymbol{U}_{RA}\in\mathbb{R}^{s_0\times s_0}$ and some diagonal matrix $\boldsymbol{\Lambda}_{RA}\in\mathbb{R}^{s_0\times s_0}$. It follows from (55) that all the diagonal elements in $\boldsymbol{\Lambda}_{RA}$ are positive. Let $\boldsymbol{\Lambda}_{RA}^{1/2}$ be the diagonal matrix such that $\boldsymbol{\Lambda}_{RA}^{1/2}\boldsymbol{\Lambda}_{RA}^{1/2} = \boldsymbol{\Lambda}_{RA}$. Apparently, the diagonal elements in $\boldsymbol{\Lambda}_{RA}^{1/2}$ are nonzero. Notice that

$$\frac{1}{n^2}\sum_{k=1}^K\boldsymbol{A}_0\boldsymbol{R}_{k,0}^T\boldsymbol{A}_0^T\boldsymbol{A}_0\boldsymbol{R}_{k,0}\boldsymbol{A}_0^T = \frac{1}{n}\boldsymbol{A}_0\boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}^{1/2}\boldsymbol{U}_{RA}\boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}^{1/2}\boldsymbol{U}_{RA}\boldsymbol{A}_0^T.$$

The s_0 largest eigenvalues in $\frac{1}{n^2}\sum_{k=1}^K\boldsymbol{A}_0\boldsymbol{R}_{k,0}^T\boldsymbol{A}_0^T\boldsymbol{A}_0\boldsymbol{R}_{k,0}\boldsymbol{A}_0^T$ corresponds to the smallest eigenvalue in

$$\frac{1}{n}\boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}^{1/2}\boldsymbol{U}_{RA}\boldsymbol{A}_0^T\boldsymbol{A}_0\boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}^{1/2}\boldsymbol{U}_{RA}.$$

Similar to (55), we can show that

$$\begin{aligned}&\lambda_{\min}\left(\frac{1}{n}\boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}^{1/2}\boldsymbol{U}_{RA}\boldsymbol{A}_0^T\boldsymbol{A}_0\boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}^{1/2}\boldsymbol{U}_{RA}\right) \geq \bar{c}\lambda_{\min}\left(\boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}^{1/2}\boldsymbol{U}_{RA}\boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}^{1/2}\boldsymbol{U}_{RA}\right) \\ =\ & \bar{c}\lambda_{\min}(\boldsymbol{U}_{RA}^T\boldsymbol{\Lambda}_{RA}\boldsymbol{U}_{RA}) \geq \bar{c}^2\lambda_{\min}\left(\sum_{k=1}^K\boldsymbol{R}_{k,0}^T\boldsymbol{\Sigma}_A\boldsymbol{R}_{k,0}\right).\end{aligned}$$

Combining this together with (55), we obtain that

$$(\lambda_{n,k}^{(s_0)})^2 \geq \bar{c}^2\bar{K}.$$

It follows from (54) that

$$\sum_{ijk}\left((\widehat{\boldsymbol{a}}_i^{(s)})^T\widehat{\boldsymbol{R}}_k^{(s)}\widehat{\boldsymbol{a}}_j^{(s)} - (\boldsymbol{a}_i^*)^T\boldsymbol{R}_k^*\boldsymbol{a}_j^*\right)^2 \geq n^2\bar{c}^2\bar{K}.$$

This completes the proof.

C.4. Proof of Theorem 1

It suffices to show

$$\Pr\left(\mathrm{IC}(s_0) > \max_{1\le s<s_0} \mathrm{IC}(s)\right) \to 1, \tag{56}$$

and

$$\Pr\left(\mathrm{IC}(s_0) > \max_{s_0\le s\le s_{\max}} \mathrm{IC}(s)\right) \to 1. \tag{57}$$

We first show (56). Combining Lemma 3 with (31), we obtain that

$$2\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k) - 2\ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) \ge \frac{\sum_{ijk}(\theta^*_{ijk}-\hat\theta^{(s)}_{ijk})^2}{4\exp(\omega_a^2\omega_r)} \ge \frac{n^2\bar c^2\bar K}{4\exp(\omega_a^2\omega_r)},$$

for any $s\in\{1,\ldots,s_0-1\}$. Combining this with (42), we have that

$$\begin{aligned}
&2\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k) - 2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) \ge 2\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k) - 2\ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\\
-\ & 2\left|\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k) - \ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)\right| - 2\left|\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) - \ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k)\right|\\
\ge\ & \frac{\bar c^2n^2\bar K}{4\exp(\omega_a^2\omega_r)} - 24\omega_a^2\omega_r n\sqrt{K\max(n,K)\log(nK)},
\end{aligned}\tag{58}$$

with probability tending to 1.

Under the given conditions, we have $K\sim n^{l_0}$ for some $0\le l_0\le 1$. This together with the condition $n^{(1-l_0)/2}\bar K \gg \exp(2\omega_a^2\omega_r)\sqrt{\log n}$ yields

$$\omega_a^2\omega_r n\sqrt{K\max(n,K)\log(nK)} = O\left(\exp(\omega_a^2\omega_r)n^{3/2+l_0/2}\sqrt{\log n}\right) \ll \frac{n^2\bar K}{\exp(\omega_a^2\omega_r)}. \tag{59}$$

By (58), we have with probability tending to 1 that

$$2\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k) - 2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) \ge \frac{\bar c^2n^2\bar K}{8\exp(\omega_a^2\omega_r)}, \tag{60}$$

for all $1\le s<s_0$. By definition, we have

$$\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s_0)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s_0)}\}_k) \ge \ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k).$$

This together with (60) gives that for all $1\le s<s_0$,

$$2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s_0)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s_0)}\}_k) - 2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) \ge \frac{\bar c^2n^2\bar K}{8\exp(\omega_a^2\omega_r)}, \tag{61}$$

with probability tending to 1.

Under the given conditions, we have $\kappa(n,K)\ll n^2\bar K/\exp(\omega_a^2\omega_r)$. Under the event defined in (61), we have that

$$\begin{aligned}
\mathrm{IC}(s_0)-\mathrm{IC}(s) &= 2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s_0)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s_0)}\}_k) - 2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) - (s_0-s)\kappa(n,K)\\
&\ge \frac{\bar c^2n^2\bar K}{8\exp(\omega_a^2\omega_r)} - s_0\kappa(n,K) \gg 0,
\end{aligned}$$

since s_0 is fixed. This proves (56).

Now we show (57). Similar to (37)-(42), we can show the following event occurs with probability tending to 1,

$$\sup_{\substack{s\in\{s_0,\ldots,s_{\max}\}\\ \boldsymbol{a}_1^{(s)},\ldots,\boldsymbol{a}_n^{(s)}\in\Omega_a,\boldsymbol{R}_1^{(s)},\ldots,\boldsymbol{R}_K^{(s)}\in\Omega_r\\ \frac{d(\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k;\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)}{s\exp(\omega_a^2\omega_r)}=O\left(\frac{r_{n,K}^{-1}}{n\sqrt{K}}\right)}} \frac{1}{s^2}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\{\theta^*_{ijk}-(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}\}\right| = O\left(\frac{\exp(\omega_a^2\omega_r)}{r_{n,K}^2}\right). \tag{62}$$

By Lemma 2, we obtain with probability tending to 1,

$$\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{1}{s^2}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\{\theta^*_{ijk}-(\widehat{\boldsymbol{a}}_i^{(s)})^T\widehat{\boldsymbol{R}}_k^{(s)}\widehat{\boldsymbol{a}}_j^{(s)}\}\right| = O\left(\frac{\exp(\omega_a^2\omega_r)}{r_{n,K}^2}\right),$$

and hence

$$\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{1}{s}\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\{\theta^*_{ijk}-(\widehat{\boldsymbol{a}}_i^{(s)})^T\widehat{\boldsymbol{R}}_k^{(s)}\widehat{\boldsymbol{a}}_j^{(s)}\}\right| = O\left(\frac{s_{\max}\exp(\omega_a^2\omega_r)}{r_{n,K}^2}\right). \tag{63}$$

Since $\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)\ge\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)$, under the event defined in (63), we have

$$\begin{aligned}
&2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) - 2\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k) \le 2\ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) - 2\ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k)\\
-\ & 2\left|\sum_{ijk}(Y_{ijk}-\pi^*_{ijk})\{\theta^*_{ijk}-(\widehat{\boldsymbol{a}}_i^{(s)})^T\widehat{\boldsymbol{R}}_k^{(s)}\widehat{\boldsymbol{a}}_j^{(s)}\}\right| = O\left(\frac{ss_{\max}\exp(\omega_a^2\omega_r)}{r_{n,K}^2}\right).
\end{aligned}$$

Notice that

$$\ell_0(\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) \le \ell_0(\{\boldsymbol{a}_i^*\}_i,\{\boldsymbol{R}_k^*\}_k),$$

we have with probability tending to 1 that

$$2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) - 2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s_0)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s_0)}\}_k) \le O\left(\frac{ss_{\max}\exp(\omega_a^2\omega_r)}{r_{n,K}^2}\right), \tag{64}$$

for all $s_0<s\le s_{\max}$. Under the event defined in (64), we have

$$\begin{aligned}
\mathrm{IC}(s_0)-\mathrm{IC}(s) &\ge 2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s_0)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s_0)}\}_k) - 2\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_i^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_k^{(s)}\}_k) + (s-s_0)\kappa(n,K)\\
&\ge (s-s_0)\kappa(n,K) - O\left(ss_{\max}\exp(\omega_a^2\omega_r)r_{n,K}^{-2}\right).
\end{aligned}$$

Under the condition $\kappa(n,K)\gg s_{\max}\exp(\omega_a^2\omega_r)(n+K)(\log n+\log K)$, we have that

$$\Pr\left(\mathrm{IC}(s_0) > \max_{s_0<s\le s_{\max}}\mathrm{IC}(s)\right)\to 1,$$

This proves (57). The proof is hence completed.

C.5. Proof of Corollary 1

Using similar arguments in Lemma 3, we can show for any permutation function $\pi : \{1,\ldots,n\} \to \{1,\ldots,n\}$,

$$\sum_{ijk}\left((\widehat{\boldsymbol{a}}_{i,\pi}^{(s)})^T\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\widehat{\boldsymbol{a}}_{j,\pi}^{(s)}-(\boldsymbol{a}_{i,\pi}^{*})^T\boldsymbol{R}_{k,\pi}^{*}\boldsymbol{a}_{j,\pi}^{*}\right)^2\geq n^2\bar{c}^2\bar{K}. \tag{65}$$

In addition, it follows from (41) and the condition $K=O(n^{l_0})$ that

$$\Pr\left(\max_{\substack{s\in\{1,\ldots,s_{\max}\}\\ \pi:\{1,\ldots,n\}\to\{1,\ldots,n\}}}\frac{1}{s}\left|\sum_{ijk}(Y_{ijk}-\pi_{ijk}^{*})(\widehat{\boldsymbol{a}}_{i,\pi}^{(s)})^T\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\widehat{\boldsymbol{a}}_{j,\pi}^{(s)}\right|\leq C_0\omega_a^2\omega_r n^{3/2+l_0/2}\sqrt{\log n}\right)\to 1,$$

and

$$\Pr\left(\frac{1}{s_0}\left|\sum_{ijk}(Y_{ijk}-\pi_{ijk}^{*})(\boldsymbol{a}_i^{*})^T\boldsymbol{R}_k^{*}\boldsymbol{a}_j^{*}\right|\leq C_0\omega_a^2\omega_r n^{3/2+l_0/2}\sqrt{\log n}\right)\to 1,$$

for some constant $C_0>0$. Hence, the following event occurs with probability tending to 1,

$$\begin{aligned}&\max\left(\max_{s,\pi}\left|\ell_0(\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\}_k)-\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\}_k)\right|/s,\right.\\ &\left.|\ell_0(\{\boldsymbol{a}_i^{*}\}_i,\{\boldsymbol{R}_k^{*}\}_k)-\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^{*}\}_i,\{\boldsymbol{R}_k^{*}\}_k)|/s_0\right)\leq C_0\omega_a^2\omega_r n\sqrt{K\max(n,K)\log(nK)}.\end{aligned} \tag{66}$$

Combining (65) with (31) yields

$$2\ell_0(\{\boldsymbol{a}_i^{*}\}_i,\{\boldsymbol{R}_k^{*}\}_k)-2\max_{\substack{s\in\{1,\ldots,s_0-1\}\\ \pi:\{1,\ldots,n\}\to\{1,\ldots,n\}}}\ell_0(\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\}_k)\geq\frac{n^2\bar{c}^2\bar{K}}{4\exp(\omega_a^2\omega_r)}. \tag{67}$$

Under the given conditions, we have

$$\frac{n^2\bar{c}^2\bar{K}}{4\exp(\omega_a^2\omega_r)}\gg\omega_a^2\omega_r n\sqrt{K\max(n,K)\log(nK)}.$$

This together with (66) and (67) gives

$$2\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^{*}\}_i,\{\boldsymbol{R}_k^{*}\}_k)-2\max_{\substack{s\in\{1,\ldots,s_0-1\}\\ \pi:\{1,\ldots,n\}\to\{1,\ldots,n\}}}\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\}_k)\geq\frac{n^2\bar{c}^2\bar{K}}{8\exp(\omega_a^2\omega_r)},$$

with probability tending to 1.

Under Condition (A4), we have $\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s_0)}\}_i,\{\widehat{\boldsymbol{R}}_{k,\pi}^{(s_0)}\}_k)\geq\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_{i,\pi}^{(s_0)*}\}_i,\{\boldsymbol{R}_{k,\pi}^{(s_0)*}\})=\ell_n(\boldsymbol{Y};\{\boldsymbol{a}_i^{*}\}_i,\{\boldsymbol{R}_k^{*}\}_k)$. Therefore, the following event occurs with probability tending to 1,

$$\min_{\pi:\{1,\ldots,n\}\to\{1,\ldots,n\}}2\left(\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s_0)}\}_i,\{\widehat{\boldsymbol{R}}_{k,\pi}^{(s_0)}\}_k)-\max_{s\in\{1,\ldots,s_0-1\}}\ell_n(\boldsymbol{Y};\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\}_k)\right)\geq\frac{n^2\bar{c}^2\bar{K}}{8\exp(\omega_a^2\omega_r)}.$$

Under the given conditions in Corollary 1, we obtain

$$\Pr\left(\bigcap_{\pi:\{1,\ldots,n\}\to\{1,\ldots,n\}}\left\{\mathrm{IC}_\pi(s_0)>\max_{1\leq s\leq s_0-1}\mathrm{IC}_\pi(s)\right\}\right)\to 1. \tag{68}$$

Similar to (46) and (49), we can show

$$\begin{aligned}&\Pr\left(\bigcup_{\pi:\{1,\ldots,n\}\to\{1,\ldots,n\}}\left\{(\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\},\widehat{\boldsymbol{R}}_{k,\pi}^{(s)})\in\mathbb{S}_m^{(s)}\right\}\right)\\ \leq\;&\Pr\left(\sup_{(\{\boldsymbol{a}_i^{(s)}\}_i,\{\boldsymbol{R}_k^{(s)}\}_k)\in\mathbb{S}_m^{(s)}}\left|\sum_{ijk}(Y_{ijk}-\pi_{ijk}^{*})\{\theta_{ijk}^{*}-(\boldsymbol{a}_i^{(s)})^T\boldsymbol{R}_k^{(s)}\boldsymbol{a}_j^{(s)}\}\right|\geq\frac{(m-1)^2s^2\exp(\omega_a^2\omega_r)}{8r_{n,K}^2}\right)\\ \leq\;&2\exp(-K_0m^2s^2r_{n,K}^{-2}),\end{aligned}$$

for some constant $K_0>0$. Using similar arguments in the proof of Lemma 2, this yields with probability tending to 1,

$$\max_{\substack{s\in\{s_0,\ldots,s_{\max}\}\\ \pi:\{1,\ldots,n\}\to\{1,\ldots,n\}}}\frac{1}{s^2}d^2\left(\{\widehat{\boldsymbol{a}}_{i,\pi}^{(s)}\}_i,\{\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\}_k;\{\boldsymbol{a}_i^{*}\}_i,\{\boldsymbol{R}_k^{*}\}_k\right)\leq\frac{\exp(2\omega_a^2\omega_r)(n+K)(\log n+\log K)}{n^2K}.$$

This together with (62) gives

$$\max_{s\in\{s_0,\ldots,s_{\max}\}}\frac{1}{s}\left|\sum_{ijk}(Y_{ijk}-\pi_{ijk}^{*})\{\theta_{ijk}^{*}-(\widehat{\boldsymbol{a}}_{i,\pi}^{(s)})^T\widehat{\boldsymbol{R}}_{k,\pi}^{(s)}\widehat{\boldsymbol{a}}_{j,\pi}^{(s)}\}\right|=O\left(\frac{s_{\max}\exp(\omega_a^2\omega_r)}{r_{n,K}^2}\right),$$

with probability tending to 1. Using similar arguments in the proof of Theorem 1, we can show

$$\Pr\left(\bigcap_{\pi:\{1,\ldots,n\}\to\{1,\ldots,n\}}\left\{\mathrm{IC}_\pi(s_0)>\max_{s_0<s\leq s_{\max}}\mathrm{IC}_\pi(s)\right\}\right)\to 1.$$

This together with (68) yields (16). The proof is hence completed.

Appendix D. Additional Simulation Results

We simulate the response $\{Y_{ijk}\}_{ijk}$ from the following model:

$$\begin{aligned}&\Pr(Y_{ijk}=1|\{\boldsymbol{a}_i\}_i,\{\boldsymbol{R}_k\}_k)=\frac{\exp(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j)}{1+\exp(\boldsymbol{a}_i^T\boldsymbol{R}_k\boldsymbol{a}_j)},\\ &\boldsymbol{a}_1,\boldsymbol{a}_2,\ldots,\boldsymbol{a}_n\overset{iid}{\sim}N(0,\{0.5^{|i-j|}\}_{i,j=1,\ldots,s_0}),\\ &\boldsymbol{R}_1=\boldsymbol{R}_2=\cdots=\boldsymbol{R}_K=\mathrm{diag}(\underbrace{1,-1,1,-1,\ldots,1,-1}_{s_0}).\end{aligned}$$

	$s_0 = 2$		$s_0 = 4$		$s_0 = 6$	
$n = 100, K = 3$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	1.00(0.00)	2.00(0.00)	0.96(0.02)	3.98(0.02)	0.88(0.03)	5.87(0.04)
$IC_{0.5}$	1.00(0.00)	2.00(0.00)	0.96(0.02)	3.98(0.02)	0.88(0.03)	5.87(0.04)
IC_1	1.00(0.00)	2.00(0.00)	0.96(0.02)	3.98(0.02)	0.85(0.04)	5.81(0.05)
BIC	0.00(0.00)	11.98(0.01)	0.00(0.00)	11.99(0.01)	0.00(0.00)	12.00(0.00)
$n = 150, K = 3$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	1.00(0.00)	2.00(0.00)	0.97(0.02)	4.03(0.02)	0.94(0.02)	6.04(0.02)
$IC_{0.5}$	1.00(0.00)	2.00(0.00)	0.97(0.02)	4.03(0.02)	0.94(0.02)	6.04(0.02)
IC_1	1.00(0.00)	2.00(0.00)	0.97(0.02)	4.03(0.02)	0.94(0.02)	6.04(0.02)
BIC	0.00(0.00)	12.00(0.00)	0.00(0.00)	12.00(0.00)	0.00(0.00)	11.99(0.01)
$n = 200, K = 3$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	1.00(0.00)	2.00(0.00)	0.97(0.02)	4.03(0.02)	0.98(0.01)	6.02(0.01)
$IC_{0.5}$	1.00(0.00)	2.00(0.00)	0.97(0.02)	4.03(0.02)	0.98(0.01)	6.02(0.01)
IC_1	1.00(0.00)	2.00(0.00)	0.97(0.02)	4.03(0.02)	0.98(0.01)	6.02(0.01)
BIC	0.00(0.00)	12.00(0.00)	0.00(0.00)	12.00(0.00)	0.00(0.00)	11.99(0.01)

Table 4: Simulation results for Setting I, II and III (standard errors in parenthesis)

	$s_0 = 2$		$s_0 = 4$		$s_0 = 6$	
$n = 50, K = 10$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	1.00(0.00)	2.00(0.00)	0.96(0.02)	3.98(0.02)	0.73(0.04)	5.83(0.06)
$IC_{0.5}$	1.00(0.00)	2.00(0.00)	0.95(0.02)	3.97(0.02)	0.69(0.05)	5.77(0.06)
IC_1	1.00(0.00)	2.00(0.00)	0.93(0.03)	3.93(0.03)	0.63(0.05)	5.57(0.07)
BIC	0.00(0.00)	11.83(0.05)	0.00(0.00)	11.82(0.04)	0.00(0.00)	11.86(0.04)
$n = 50, K = 20$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	0.98(0.01)	2.02(0.01)	0.90(0.03)	4.10(0.03)	0.76(0.04)	6.06(0.05)
$IC_{0.5}$	0.98(0.01)	2.02(0.01)	0.94(0.02)	3.98(0.02)	0.81(0.04)	5.99(0.04)
IC_1	0.98(0.01)	2.02(0.01)	0.94(0.02)	3.94(0.02)	0.74(0.04)	5.81(0.05)
BIC	0.00(0.00)	12.00(0.00)	0.00(0.00)	12.00(0.00)	0.00(0.00)	11.99(0.01)
$n = 50, K = 50$	TP	$\hat{s}$	TP	$\hat{s}$	TP	$\hat{s}$
IC_0	0.96(0.02)	2.04(0.02)	0.88(0.03)	4.12(0.03)	0.68(0.05)	6.57(0.13)
$IC_{0.5}$	0.98(0.01)	2.02(0.01)	0.94(0.02)	4.04(0.02)	0.82(0.04)	6.06(0.04)
IC_1	0.98(0.01)	2.02(0.01)	0.94(0.02)	4.02(0.02)	0.74(0.04)	5.75(0.05)
BIC	0.00(0.00)	12.00(0.00)	0.00(0.00)	12.00(0.00)	0.00(0.00)	12.00(0.00)

Table 5: Simulation results for Setting IV, V and VI (standard errors in parenthesis)

We use the same six settings as described in Section 4.2. In each setting, we further consider three scenarios, by setting $s_0 = 2, 4$ and 6. Reported in Table 3 and 4 are the percentage of selecting the true models (TP) and the average of $\hat{s}$ selected by IC_0, $IC_{0.5}$, IC_1 and BIC over 100 replications.

It can be seen from Table 4 and 5 that our proposed information criteria are consistent. In contrast, BIC fails in all settings. In addition, in the last two settings, TPs of $IC_{0.5}$ and IC_1 are larger than IC_0 for most of the cases. As commented before, these differences are due to the finite sample correction term $\tau_\alpha(n, K)$.

References

Genevera Allen. Sparse higher-order principal components analysis. In *International Conference on Artificial Intelligence and Statistics*, pages 27–36, 2012.

Jean-Yves Audibert and Alexandre B. Tsybakov. Fast learning rates for plug-in classifiers. *Ann. Statist.*, 35(2):608–633, 2007. ISSN 0090-5364. doi: 10.1214/009053606000001217. URL `http://dx.doi.org/10.1214/009053606000001217`.

Jushan Bai and Serena Ng. Determining the number of factors in approximate factor models. *Econometrica*, 70(1):191–221, 2002. ISSN 0012-9682. doi: 10.1111/1468-0262.00273. URL `http://dx.doi.org/10.1111/1468-0262.00273`.

Sudipto Banerjee and Anindya Roy. *Linear algebra and matrix analysis for statistics*. CRC Press, 2014.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Eric C. Chi and Tamara G. Kolda. On tensors, sparsity, and nonnegative factorizations. *SIAM J. Matrix Anal. Appl.*, 33(4):1272–1299, 2012. ISSN 0895-4798. URL `https://doi.org/10.1137/110859063`.

Yingying Fan and Cheng Yong Tang. Tuning parameter selection in high dimensional penalized likelihood. *J. R. Stat. Soc. Ser. B. Stat. Methodol.*, 75(3):531–552, 2013. ISSN 1369-7412. doi: 10.1111/rssb.12001. URL `http://dx.doi.org/10.1111/rssb.12001`.

Mark Galassi, Jim Davies, James Theiler, Brian Gough, Gerard Jungman, Patrick Alken, Michael Booth, Fabrice Rossi, and Rhys Ulerich. *GNU Scientific Library Reference Manual (Version 2.1)*, 2015. URL `http://www.gnu.org/software/gsl/`.

Lise Getoor and Lilyana Mihalkova. Learning statistical models from relational data. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 1195–1198. ACM, 2011.

Richard A Harshman. Models for analysis of asymmetrical relationships among n objects or stimuli. In *Paper presented at the First Joint Meeting of the Psychometric Society and the Society for Mathematical Psychology, Hamilton, Ontario, August*, 1978.

Richard A Harshman and Margaret E Lundy. Parafac: Parallel factor analysis. *Computational Statistics & Data Analysis*, 18(1):39–72, 1994.

Charles Kemp, Joshua B Tenenbaum, Thomas L Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *AAAI*, volume 3, page 5, 2006.

Stanley Kok and Pedro Domingos. Statistical predicate invention. In *Proceedings of the 24th international conference on Machine learning*, pages 433–440. ACM, 2007.

Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009. ISSN 0036-1445. URL https://doi.org/10.1137/07070111X.

Anmol Madan, Manuel Cebrian, Sai Moturu, Katayoun Farrahi, et al. Sensing the" health state" of a community. *IEEE Pervasive Computing*, 11(4):36–45, 2012.

Shahar Mendelson, Alain Pajor, and Nicole Tomczak-Jaegermann. Uniform uncertainty principle for bernoulli and subgaussian ensembles. *Constructive Approximation*, 28(3): 277–289, 2008.

Maximilian Nickel. *Tensor factorization for relational learning*. PhD thesis, the Ludwig-Maximilians-University of Munich, 2013.

Maximilian Nickel and Volker Tresp. Logistic tensor factorization for multi-relational data. *arXiv preprint arXiv:1306.2084*, 2013.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 809–816, 2011.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.

Krzysztof Nowicki and Tom A. B. Snijders. Estimation and prediction for stochastic blockstructures. *J. Amer. Statist. Assoc.*, 96(455):1077–1087, 2001. ISSN 0162-1459. doi: 10.1198/016214501753208735. URL http://dx.doi.org/10.1198/016214501753208735.

Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine learning*, 62 (1):107–136, 2006.

Gideon Schwarz. Estimating the dimension of a model. *Ann. Statist.*, 6(2):461–464, 1978. ISSN 0090-5364. URL http://links.jstor.org/sici?sici=0090-5364(197803)6:2<461:ETDOAM>2.0.CO;2-5&origin=MSN.

Will Wei Sun, Junwei Lu, Han Liu, and Guang Cheng. Provable sparse tensor decomposition. *J. R. Stat. Soc. Ser. B. Stat. Methodol.*, 79(3):899–916, 2017. ISSN 1369-7412. URL https://doi.org/10.1111/rssb.12190.

Alexandre B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Ann. Statist.*, 32(1):135–166, 2004. ISSN 0090-5364. doi: 10.1214/aos/1079120131. URL http://dx.doi.org/10.1214/aos/1079120131.

Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

Aad W. van der Vaart and Jon A. Wellner. *Weak convergence and empirical processes*. Springer Series in Statistics. Springer-Verlag, New York, 1996. ISBN 0-387-94640-3. doi: 10.1007/978-1-4757-2545-2. URL http://dx.doi.org/10.1007/978-1-4757-2545-2. With applications to statistics.

Yu Wang, Wotao Yin, and Jinshan Zeng. Global convergence of admm in nonconvex nonsmooth optimization. *arXiv preprint arXiv:1511.06324*, 2017.

Yun Yang and David B. Dunson. Bayesian conditional tensor factorizations for high-dimensional classification. *J. Amer. Statist. Assoc.*, 111(514):656–669, 2016. ISSN 0162-1459. URL https://doi.org/10.1080/01621459.2015.1029129.

Xiang Zhang, Yichao Wu, Lan Wang, and Runze Li. A consistent information criterion for support vector machines in diverging model spaces. *J. Mach. Learn. Res.*, 17(1):1–26, 2016. ISSN 1532-4435.

Journal of Machine Learning Research 20 (2019) 1-29 Submitted 3/18; Revised 11/18; Published 1/19

Joint PLDA for Simultaneous Modeling of Two Factors

Luciana Ferrer lferrer@dc.uba.ar
Instituto de Investigación en Ciencias de la Computación (ICC)
CONICET-Universidad de Buenos Aires
Pabellón I, Ciudad Universitaria, 1428, Ciudad Autónoma de Buenos Aires, Argentina

Mitchell McLaren mitchell.mclaren@sri.com
Speech Technology and Research Lab (StarLab)
SRI International
333 Ravenswood Ave, Menlo Park, 94025, United States

Editor: Barbara Engelhardt

Abstract

Probabilistic linear discriminant analysis (PLDA) is a method used for biometric problems like speaker or face recognition that models the variability of the samples using two latent variables, one that depends on the class of the sample and another one that is assumed independent across samples and models the within-class variability. In this work, we propose a generalization of PLDA that enables joint modeling of two sample-dependent factors: the class of interest and a nuisance condition. The approach does not change the basic form of PLDA but rather modifies the training procedure to consider the dependency across samples of the latent variable that models within-class variability. While the identity of the nuisance condition is needed during training, it is not needed during testing since we propose a scoring procedure that marginalizes over the corresponding latent variable. We show results on a multilingual speaker-verification task, where the language spoken is considered a nuisance condition. The proposed joint PLDA approach leads to significant performance gains in this task for two different data sets, in particular when the training data contains mostly or only monolingual speakers.

Keywords: Probabilistic linear discriminant analysis, speaker recognition, factor analysis, language variability, robustness to acoustic conditions

1. Introduction

PLDA was proposed by Prince (2007) for doing inferences about the identity of a person from an image of their face. A closely related model had been previously proposed by Ioffe (2006) and also tested on image processing tasks. The technique was later widely adopted by the speaker recognition community, becoming the state-of-the-art scoring technique for this task (Kenny, 2010; Burget et al., 2011; Brümmer, 2010a; Senoussaoui et al., 2011; Matejka et al., 2011). PLDA assumes that each sample[1] is represented by a feature vector of fixed dimension and that this vector is given by a sum of three terms: a term that depends on the class of the sample, a term that models the within-class variability and is assumed independent across samples, and a final term that models any remaining variability and is also independent across samples. These assumptions imply that all samples from the same class are independent of each other and also independent of samples from other classes once the class is known.

In contrast with the assumptions made by PLDA, many training data sets consist of samples that come from a small set of distinct conditions. For example, many speaker recognition data sets contain only a few acoustic conditions (different microphones or noise conditions), speech styles (conversational, monologue, read), and languages. Samples corresponding to the same condition will most likely be statistically dependent.

The literature proposes a few approaches that generalize PLDA to consider metadata about the samples during training. The main motivation for these approaches, though, is not to relax the conditional independence assumption but rather to enable a more flexible model that can adapt to each of the available conditions rather than assuming that samples from all conditions can be modeled with the same linear model. Yet, a side effect of the proposed generalizations is the introduction of a dependency between samples from the same condition. The simplest approach of this family is to train a separate PLDA model for each condition, as proposed by Garcia-Romero et al. (2012). Nevertheless, in this paper, the authors show that pooling the data from all conditions, as proposed by Lei et al. (2012), leads to better performance than training separate models. This result is reasonable, since training separate PLDA models does not allow the overall model to learn how samples from the same class vary across conditions; only within-condition variation is learned.

The tied PLDA model proposed by Li et al. (2012) is designed to tackle this problem. In this approach, one PLDA model is trained for each condition, but these models are tied by forcing the latent variable corresponding to each class to be the same across all conditions. The approach was shown to outperform standard PLDA with pooled training data when each class in the training data is seen under both considered conditions, frontal and profile, in a face recognition task. A similar approach is proposed by Mak et al. (2016), but in this case, the mixture component is not given during training. Instead, the PLDA mixture components depend on a continuous metadata value, which is modeled with a mixture of Gaussians. The approach is tested by adding noise to the training data at different SNR levels. The resulting training data then contains samples for each speaker at different SNR levels. Under these conditions, the authors show gains from the proposed approach compared to pooling all the data to train a single PLDA model.

In this paper, we consider a scenario where each speaker in the training data is seen only under a small subset of the conditions present in the training set (potentially, only one). Further, we expect some conditions to have much less training data than others. Under this scenario, the tied PLDA approach does not work well, since it requires training a PLDA model of the same dimensions for each condition, which may be impossible or suboptimal for the conditions with less data. Further, the tied PLDA model can only learn how the nuisance conditions affect the classes of interest if it is provided samples for each class under different conditions during training.

We propose a novel generalization of the PLDA model that relaxes the conditional independence assumption without increasing the size of the parameter space, keeping the same functional form of the original PLDA model but modifying the training and scoring procedures to consider the dependency across samples originating from the sample's condition. In the proposed approach, which we call Joint PLDA (JPLDA), the condition is assumed to be known during training but not during testing. An expectation-maximization (EM) training

1. Throughout this paper, the word sample is used to refer to an audio signal or recording containing speech.

©2019 Luciana Ferrer and Mitchell McLaren.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-134.html.

procedure is formulated that takes into account the condition of each sample. Scoring is performed, as in standard PLDA, by computing a likelihood ratio between the null hypothesis that the two sides of a trial belong to the same speaker versus the alternative hypothesis that the two sides belong to different speakers. The two likelihoods are computed by marginalizing over two hypotheses about the condition in both sides of a trial: that they are the same and that they are different. This way, we expect that the new model will be better at coping with same-condition versus different-condition trials than standard PLDA, since knowledge about the condition is used during training and implicitly considered during scoring. Further, we expect this model to behave better than tied PLDA under a training scenario where the number of samples is highly imbalanced across conditions and each speaker is seen only under one or a small subset of conditions.

The mathematical formulation for the proposed JPLDA approach was first published in arXiv (Ferrer, 2017), without results or analysis. A similar model with a significantly different formulation for the EM and scoring algorithms was later published, also in arXiv, by other authors (Shi et al., 2017). They show significant improvements from this approach on a text-dependent speaker verification task. In this paper, we show results and detailed analysis on two multilingual speaker recognition data sets, one composed of Mixer data (Cieri et al., 2007) from the speaker recognition evaluations organized by NIST and another that uses LASRS data (Beck et al., 2004). We evaluate two training scenarios, one using all available training data from the PRISM data set (Ferrer et al., 2011), which contains a small percentage of speakers speaking two different languages, and one where we subset the training data to contain only one language per speaker. We show that JPLDA significantly outperforms two standard PLDA approaches with different structures and tied PLDA, especially when the training data contains mostly or only a single language per speaker.

2. Standard PLDA Models

In this work, we adopt the nomenclature usually used by the speaker recognition community. Yet, the model proposed can be used for the original image processing task or any other task for which standard PLDA is used.

Standard PLDA (Prince, 2007) assumes that the vector m_i representing a certain sample from speaker s_i is given by

$$m_i = \mu + V y_{s_i} + U x_i + z_i, \tag{1}$$

where μ is a fixed bias; y_{s_i} is a vector of size R_y, the dimension of the speaker subspace; and x_i is a vector of size R_x, the dimension of the subspace corresponding to the nuisance condition or, as usually called in speaker recognition, the channel. The model assumes that

$$\begin{aligned} y_{s_i} &\sim N(0, I), \\ x_i &\sim N(0, I), \\ z_i &\sim N(0, D^{-1}), \end{aligned}$$

where the matrix D is assumed to be diagonal. All these latent variables are assumed independent: speaker variables are independent across speakers, and the nuisance variable x_i and noise variable z_i are independent across samples.

This model is equivalent (Sizov et al., 2014) to assuming the following distributions

$$m_i|\gamma_{s_i} \sim N(\gamma_{s_i}, UU^T + D^{-1}), \tag{2}$$

$$\gamma_{s_i} \sim N(\mu, VV^T). \tag{3}$$

where we can see that VV^T is the between-speaker and $UU^T + D^{-1}$ is the within-speaker covariance matrix of the m_i vectors. Note that, in the general case, the matrix VV^T could be singular. This fact, though, does not cause any theoretical or practical problems for the PLDA model.

The model described above corresponds to the original PLDA formulation, which we will call full PLDA (FPLDA). In speaker recognition, a simplified version of PLDA (SPLDA for the purpose of this paper) is more commonly used, where the nuisance term Ux_i is absorbed into the noise term, which is then assumed to have a full rather than diagonal covariance matrix. Sizov et al. (2014) gives a comprehensive explanation of the usual flavors of PLDA.

The training of PLDA parameters is done using an EM algorithm. The EM formulation for SPLDA and FPLDA can be found in two very detailed documents by Brümmer (2010a,b). For FPLDA, we use a minimum divergence (Kenny, 2010) step after every maximization step. This step is generally used to speed up convergence of the EM algorithm. For SPLDA, this step was not necessary for quick convergence since, as we will see, the smart initialization procedure described below already leads to an excellent model. We will not reproduce the EM formulas here, but we will describe the two initialization procedures we use, since they will be compared in the experimental section.

2.1. EM Initialization Procedure

The EM algorithm requires an initial model to start the iterations. This model can be generated randomly or, in the case of SPLDA, with a "smart" procedure that results in a much better initial model that, in turn, requires many fewer or no EM iterations to converge to the final parameters.

In our experiments, for random initialization, we set D to be an identity matrix, and V and U, when applicable, to be matrices with random elements drawn from a normal distribution with standard deviation 0.01 and mean 0.

For SPLDA, we also try a smart initialization approach that is directly motivated by Equations (2) and (3). Namely, we initialize V and D as follows

$$\begin{aligned} V &= Q\Lambda^{-1/2}, \\ D &= W^{-1}, \end{aligned}$$

where W is the empirical within-speaker covariance matrix of the training data, and Q is a matrix with the eigenvectors corresponding to the R_y largest eigenvalues of the between-speaker covariance matrix of the training data and $\Lambda^{-1/2}$ is a diagonal matrix containing the square roots of those eigenvalues. As will be shown in the experiments, this initialization procedure works quite well, leading to models that are as good as those trained with many iterations of EM.

2.2. Scoring

In this work, we consider a verification task. Two sets of samples, an enrollment set E and a test set T, each corresponding to one or more samples from the same speaker, are compared

to decide whether the two speakers are the same or different. This comparison is usually called a *trial* in speaker verification. In some applications, a hard decision is needed; in others, a soft score is preferable. The PLDA paper (Prince, 2007) showed how to use this model to compute the likelihood ratio (LR) between the two hypotheses, which can be used directly as soft score or thresholded to make hard decisions if required. The LR is given by

$$LR = \frac{p(E,T|H_{SS})}{p(E,T|H_{DS})},$$

where H_{SS} is the hypothesis that the speakers in both sets are the same, while H_{DS} is the hypothesis that the speakers are different. This value can be computed using a closed form using the PLDA model. In our code we use the formulation derived by Cumani et al. (2014), Equation (34). Note, though, that the last term in that equation should not be there (this mistake was confirmed by one coauthor of the paper).

3. Tied PLDA Model

The tied PLDA model was introduced by Li et al. (2012). The model is a mixture of PLDA models where the latent variable corresponding to the speaker is tied across components. The vector representing sample i from speaker s_i is modeled as

$$m_i = \mu_{d_i} + V_{d_i} y_{s_i} + U_{d_i} x_i + z_i, \tag{4}$$

where d_i indicates the mixture component corresponding to sample i, and where

$$\begin{aligned} y_{s_i} &\sim N(0, I), \\ x_i &\sim N(0, I), \\ z_i &\sim N(0, D_{d_i}^{-1}). \end{aligned}$$

Hence, once the mixture component is given, the model reduces to a standard PLDA model. In this work, we assume that the mixture component is known both during training and during testing, as in the original work (Li et al., 2012), though the authors note that this is not a necessary condition. In the simplest case we could take the mixture component to be the nuisance condition of the sample but, as we will see, this might not be feasible if some conditions have too few training samples in which case grouping of samples from different conditions into the same component might be necessary. Note that the latent variable y_{s_i} does not depend on the component. Rather, this variable is tied for all samples from the same speaker across components. This enables the model to properly represent cross-component variability.

As for the original PLDA model, a simple PLDA model can be used instead of the full PLDA model for each component in the mixture. Further, the covariance matrix for the noise term can be either full or diagonal. In this work, each component is described by a SPLDA model for simplicity of implementation, since the difference between SPLDA and FPLDA is very small in practice.

The TPLDA model described by Li et al. (2012) and used here coincides with what Mak et al. (2016) calls SNR-dependent mixture PLDA model if we assume the SNR to be discrete rather than continuous so that the posterior probability for each component is fixed to 1 for the component corresponding to the sample, and to 0 otherwise. The training and scoring procedures for TPLDA can be found in the supplementary material for Mak et al. (2016).

4. Joint PLDA Model

In this work, we propose a generalization of the original PLDA model where the nuisance variable is no longer considered independent across samples, but potentially shared (tied) across samples that correspond to the same nuisance condition. This makes the model symmetric in the two latent variables corresponding to the speaker and the nuisance condition. To represent this dependency, we introduce a condition label for each sample, called c_i. Given this label, and the speaker label s_i, we propose to model vector m_i of dimension R_m for sample i as:

$$m_i = \mu + V y_{s_i} + U x_{c_i} + z_i, \tag{5}$$

where, as before, y_{s_i} is a vector of size R_y and x_{c_i} is a vector of size R_x, and

$$\begin{aligned} y_{s_i} &\sim N(0, I), \\ x_{c_i} &\sim N(0, I), \\ z_i &\sim N(0, D^{-1}). \end{aligned}$$

The model's parameters to estimate are $\lambda = \{\mu, V, U, D\}$, as in the standard PLDA formulation, but the input data for the training algorithm is now required to have a second set of labels indicating the nuisance condition of each sample.

The expectation-maximization equations for training this new model are significantly more involved than for the original PLDA model. This is due to the fact that each speaker cannot be treated separately from the others since samples from one speaker might be dependent on samples from a different speaker. This creates a potential dependency between all training samples, which greatly complicates the formulation, increasing the computation time by orders of magnitude for each EM iteration. Nevertheless, as we will see in the experiments, initializing the model in a smart way basically makes EM unnecessary in our experiments, reducing the training time of the model to the same order of what is required to train standard PLDA on the same data. A detailed derivation of the EM algorithm and scoring procedure for JPLDA is given by Ferrer (2017). In Appendix A we give a summary of the formulation, including all the equations needed to implement these algorithms. We have not implemented a minimum divergence step for this model. We plan to add this step in the future. Yet, given that, as we will see, the smart initialization procedure described below makes EM unnecessary in our experiments, we believe minimum divergence is unlikely to result in better models for this approach, at least for the current experiments.

Note that the matrix D in the JPLDA model can be full or diagonal. If we want D to be diagonal, we simply set D to be the diagonal part of the estimated value for D in each maximization step of the EM algorithm, as done for the standard PLDA EM algorithm (Brümmer, 2010a).

4.1. EM Initialization Procedure

The JPLDA model can be randomly initialized using the same procedure as for standard PLDA described in Section 2.1. We propose an alternative procedure to get the initial values for the PLDA model, U_0, V_0 and D_0. The procedure first estimates the matrix U_0 by training an SPLDA model with condition labels as targets, implicitly absorbing the effect of the speaker term into the noise term. This PLDA model is used to estimate the effect

of the condition in the samples, which is then subtracted to obtain condition-compensated samples. The new training data is then used to estimate another SPLDA model which should now model the effect of the speaker. This second SPLDA model is used obtain V_0 and D_0. Algorithm 1 gives the pseudocode for the proposed initialization algorithm. Note that, as usually done when training PLDA models, we assume an initial step where μ is set to the global mean of the data and subtracted from all training samples. The initialization step, as well as the EM iterations, use the resulting centered samples as input. As we will see this "smart" initialization leads to such a good starting point that EM iterations are unnecessary in our experiments.

Algorithm 1 Smart EM initialization approach for JPLDA. The matrix M, of size $N \times R_m$ contains the centered training vectors. L_c and L_s are vectors of size N containing the condition and speaker labels for the training data, respectively. Function SPLDAtrain returns the parameters of an SPLDA model (the subspace matrix and the noise precision matrix) after running 20 EM iterations, while function SPLDAlatent returns the estimated latent variables for each of the training samples. Note that all samples with the same label have the same latent variable.

procedure JPLDAInitialization(M, L_c, L_s, R_x, R_y)
  U_0, D = SPLDAtrain(M, L_c, R_x)
  X = SPLDAlatent(M, L_c, U_0, D) ▷ X is a matrix of size $N \times R_x$
  $M_c = M - XU_0^T$
  V_0, D_0 = SPLDAtrain(M_c, L_s, R_y)
  return V_0, U_0, D_0

4.2. Scoring

As for standard PLDA, we define the score as the likelihood ratio between the two hypotheses: that the speakers are the same and that the speakers are different. Nevertheless, in this case we need to marginalize both likelihoods over two new hypotheses: that the nuisance conditions are the same and that they are different. This is because, in general, we cannot assume that the nuisance condition is known during testing. Hence, the LR is computed as follows:

$$LR = \frac{p(E,T|H_{SS},H_{SC})P(H_{SC}|H_{SS}) + p(E,T|H_{SS},H_{DC})P(H_{DC}|H_{SS})}{p(E,T|H_{DS},H_{SC})P(H_{SC}|H_{DS}) + p(E,T|H_{DS},H_{DC})P(H_{DC}|H_{DS})} \quad (6)$$

where, as before, H_{SS} is the hypothesis that the speakers for both sets are the same, and H_{DS} is the hypothesis that they are different, while H_{SC} is the hypothesis that the nuisance condition for both sets is the same, and H_{DC} is the hypothesis that they are different. This LR value can be computed using a closed form derived in Appendix A and, in more detail, in (Ferrer, 2017).

Note that here we assume that all samples from the enrollment set come from the same condition and all samples from the test set come from the same condition, which could be the same or different from the enrollment condition. This is trivially true when the sets are composed of a single sample, which is the case we consider in the experiments in this paper. The formulation would become more complex without this assumption since we would need to consider the possibility that each sample in each set could come from different conditions. In

(Ferrer, 2017), we also derive the scoring formula for a multi-enrollment single-test case where the enrollment conditions are known and different from the test condition. This formulation is used when applying JPLDA to language identification (LID). Experiments on LID will be the subject of another paper. Further, the generalization of the scoring formula to multiple enrollment or test samples with unknown conditions will be considered in future work.

Another implicit assumption made in Equation (6) is that the conditions in the test trials have not been seen during training. This assumption is also made with respect to the speakers, both in JPLDA and standard PLDA scoring. While, in many applications, the assumption that test speakers are unseen during training is appropriate, this may not always be the case for nuisance conditions. In particular, in our experiments, where the nuisance condition is given by the language spoken in the signal, this assumption does not hold for some trials, since some of the test languages are seen during training. In the future, we plan to relax this assumption, which may allow for further performance improvements.

The scoring formula above depends on two prior probabilities, the probability that the enrollment and test conditions are the same given that the speakers are the same, $P(H_{SC}|H_{SS})$, and the probability that the conditions are the same given that the speakers are different, $P(H_{SC}|H_{DS})$. The other two prior probabilities are dependent on these two since $P(H_{SC}|H_{SS}) + P(H_{DC}|H_{SS}) = 1$ and $P(H_{SC}|H_{DS}) + P(H_{DC}|H_{DS}) = 1$. These two independent prior probabilities are parameters that could be computed from the training data, tuned using a development set, or set to arbitrary values based on what is known about the test data. In some applications, the nuisance condition might be known also in testing. In that case, the same-condition priors can be set to 1.0 for same-condition trials and to 0 for different-condition trials.

5. Experimental Setup

In this section we describe the task, the performance metrics, the data used for the experiments and the procedure used to convert each audio sample to a fixed-length vector to be modeled by the different PLDA methods.

5.1. Multilanguage Speaker Verification

The task considered for our experiments is speaker verification, which consists of determining whether two sets of samples, an enrollment and a test set, belong to the same speaker or not. Here we consider the simplest case, where both enrollment and test sets each contain a single speech sample. A pair of enrollment and test samples is called a *trial*. A trial is a target trial if the enrollment and test speakers are the same and an impostor trial if the two speakers are different. In this paper we explore the problem of multilanguage speaker verification where test trials can be composed of two samples in the same language (same-language trials) or two samples in different languages (cross-language trials).

Most state-of-the-art speaker verification systems are inherently language-independent in the sense that they do not use information about the language spoken in order to generate the output score. Yet, this does not mean that they are robust to language variation. In fact, speaker verification performance is known to degrade significantly in cross-language trials as well as in same-language trials from languages not found in the training set (Auckenthaler et al., 2001; Misra and Hansen, 2014; Rozi et al., 2016).

Rozi et al. (2016) discusses a problem that occurs when training PLDA models with multilingual data: the distribution of the speaker factors becomes broader to cover the different languages that a speaker might speak, which could result in suboptimal performance on same-language trials. They propose to mitigate this problem by training a standard PLDA model using both language and speaker as targets (i.e., samples from the same speaker but different language are considered as different speakers). This language-aware PLDA model performs significantly better on same-language trials than the model trained with speaker targets, but degrades on cross-language trials, since it cannot model cross-language variation. JPLDA, on the other hand, can simultaneously model language and speaker factors, allowing the speaker factors to keep a sharper distribution, while still modeling the effect of language, resulting in improved performance both in same-language and cross-language trials with respect to standard PLDA.

5.2. Performance Metrics

We compute performance using the equal-error rate (EER) and detection error (DET) curves. The DET curves and the EER measure the performance of a system that uses the scores (in our case, the LRs) to make final decisions on the label of each sample by comparing these scores with a decision threshold. Samples whose scores are above the threshold are labeled as targets and samples whose scores are below the threshold are labeled as impostors. Two types of error are then possible: (1) misses, the true target trials that are labeled as impostors by the system, and (2) false alarms, the impostor trials that are labeled as targets by the system. The EER is given by the miss rate when the decision threshold is set such that the miss rate is equal to the false alarm rate.

DET curves (Martin et al., 1997) are a variation over the traditional receiver operating characteristic (ROC) curves that have been widely used for speaker verification for two decades. A DET curve is a plot of the false alarm rate versus the miss rate obtained while sweeping a decision threshold over a certain range where the axes are transformed to a probit scale. The probit transformation, the inverse of the cumulative distribution function of the standard normal distribution, converts the miss versus false alarm rate curve into a straight line if the score distribution for the two classes is Gaussian with the same standard deviation (Martin et al., 1997), which is a reasonable approximation for many speaker verification systems.

5.3. Training Data

We consider two training conditions, one that includes all our available training data (FULL) and a subset that keeps only one language for each speaker (MONOLING). The second condition is designed to help us analyze performance of the PLDA methods under this extremely challenging scenario where no explicit information is available in the training data of the effect that language has on the vectors representing the samples.

The FULL training set is composed of:

- Switchboard Cellular Part 1 (Graff et al., 2001) and Cellular Part 2 (Graff et al., 2004), consisting of English cellphone conversations
- Switchboard 2 Phase 2 (Graff et al., 1999) and Phase 3 (Graff et al., 2002) samples, consisting of English telephone conversations

- Mixer data (Cieri et al., 2007) from the 2004 to 2008 speaker recognition evaluations organized by the National Institute of Standards and Technology (NIST). This data contains English samples recorded both on telephone and microphone channels and non-English samples recorded on telephone channels. With very few exceptions, speakers that recorded non-English samples also recorded English samples. Only one speaker has data in two non-English languages and no data in English. Only a subset of this data is used for training, leaving some speakers out for testing (Section 5.4). We also discard data from languages for which only one or two speakers are available and samples where the language was unavailable or ambiguous (e.g., more than one language listed in the language key) in NIST's keys.

The MONOLING training set is created by randomly keeping the samples from only one of the languages spoken by each speaker from the FULL training set.

Finally, in Section 6.4, we analyze results when subsetting these two training sets to contain a more balanced representation of channels while keeping all data from bilingual speakers for the FULL training set. Specifically, we subset the FULL training set to discard all the telephone data from speakers that do not have data in both English and another language. This is data that is not adding much new information, since all non-English data is recorded over telephone line, and all speakers with non-English data also have telephone recordings in English. By subsetting the data this way, we achieve a more balanced representation of the telephone data with respect to the microphone data, while emphasizing the data from bilingual speakers, which is a very small minority on the original set including all the data. To create the subset for the MONOLING training set, we simply use the samples that appear in the subset from the FULL training set and also appear in the MONOLING set.

Table 1 shows statistics on the two training sets and their corresponding subsets. The languages included under "Other" are: Arabic (with 440 samples); Bengali (88); French (25); Chinese (868); Farsi (25); Hindi (143); Italian (11); Japanese (124); Korean (78); Russian (478); Spanish (170); Tagalog (26); Thai (185); Vietnamese (169); Chinese Wu (63); and Cantonese (216).

5.4. Test Data

We consider two testing conditions, one composed of Mixer data and used for development and one composed of LASRS data and used as held-out set for final evaluation of the selected methods.

The Mixer test data is composed of telephone samples from Mixer collections (Cieri et al., 2007) from the 2005 to 2010 NIST speaker recognition evaluations, from speakers not used for training. We include 119 samples in Arabic from 21 speakers; 200 samples in Russian from 47 speakers; 309 samples in Thai from 38 speakers; 827 samples in Chinese from 163 speakers; and 5755 samples in English from 701 speakers (including those that also speak one of the other languages). The trials are created by selecting the same number of target and impostor same-language and cross-language trials such that the final set of trials is a balanced union of both types of trials. Further, the same-language trials are created as a balanced union of English versus non-English trials. The final set of trials contains 11,522 target trials and 858,119 impostor trials.

The LASRS test data is composed of samples from a bilingual, multi-modal voice corpus (Beck et al., 2004). The corpus is composed of 100 bilingual speakers from each of

Name	Sel	Sample count				Speaker count			
		Eng Mic	Eng Phn	Other Phn	Total	MonoLing		BiLing	Total
						Eng	Other		
FULL	all	11017	38382	3109	52508	2764	34	495	3293
	subset	11017	3711	2733	17461	207	0	494	701
MONOLING	all	10797	36619	1688	49104	3026	267	0	3293
	subset	10797	1948	1332	14077	468	233	0	701

Table 1: Statistics for the four training sets considered in our experiments. "Eng" refers to English data while "Other" refers to any language other than English. "Phn" refers to telephone or cellphone data, and "Mic" refers to all other microphones in the training set. "Monoling" refers to speakers for which we only have samples in a single language, and "Biling" refers to speakers for which we have samples in two languages (English plus one other language in most cases).

three languages: Arabic, Korean and Spanish. In all cases, the other language of the speakers is English. Each speaker is asked to perform a series of tasks, including talking with a partner on the phone and reading various texts, in English and also in their native language. Each task is recorded using several recording devices and repeated in two separate sessions recorded on different days. The LASRS trials for this work are created by enrolling with data from the first recorded session and testing on the second recorded session in each of the two spoken languages. We use only the conversational data from each session. This results in the same number of same-language and cross-language trials for a total of 848 target trials and 100336 impostor trials for each of seven different microphones: a camcorder microphone (Cm); a Desktop microphone (Dm); a studio microphone (Sm); an omnidirectional microphone (Om); a local telephone microphone (Tm); a remote telephone microphone (Tk); and a telephone earpiece (Ts). For this study, we only consider same-microphone trials for simplicity of analysis. For more details on the collection protocol, see (Beck et al., 2004).

5.5. I-vector Extraction

For validation of the proposed approach, we use a traditional i-vector framework for speaker recognition (Dehak et al., 2011). I-vectors are fixed-dimensional vectors that attempt to represent, as fully as the assumptions allow, the characteristics of the speech in an audio recording. In this framework, each recording is first represented by a sequence of short-term feature vectors $x = \{x_1, \ldots, x_L\}$. The length L of this sequence is variable and depends on the duration of the recording. The i-vector approach then assumes each of these feature vectors x_j is independently drawn from an N-component Gaussian mixture model (GMM) with weights w_i, covariance matrices C_i and means $\mu_i + T_i\omega$, for $i \in \{1, \ldots, N\}$. The parameters of the i-vector model are the set of w_i, C_i, μ_i, and T_i. While weights and covariances are fixed for all recordings, the means vary in a subspace determined by the matrices T_i. Each recording is then modeled by a different GMM, determined by the vector ω. These vectors are assumed

to have a prior normal standard distribution. The mean of the posterior distribution of ω given the sequence of features x is the i-vector, used to represent the recording.

The posterior distribution of the i-vector model is intractable. It cannot be used in an EM algorithm to obtain the model's parameters or even to obtain the i-vectors for a new sample, given the model. A mean-field Variational Bayes (VB) EM approach can be used to estimate the model's parameters and the i-vectors, as described by Brümmer (2015). In this approach, the parameters are iteratively reestimated, along with the posterior distribution for ω and the responsibilities for each recording. The responsibilities are variational parameters that can be interpreted as soft assignments for the frames to each of the N components of the model.

In the classical i-vector approach, though, the parameters w_i, C_i and μ_i are fixed in an initial step. They are given by the weights, covariances and means of a GMM, called the universal background model (UBM), trained using recordings from many different speakers. An approximate posterior distribution of ω is then obtained using a simplifying assumption: the responsibilities are set to be the UBM state posteriors. With the responsibilities and the w_i, C_i and μ_i parameters fixed, the T_i matrices are estimated by maximizing the VB lower bound. Finally, once the parameters of the model have been estimated, i-vectors are extracted as the mean of the approximate posterior distribution of ω given the features for the recording, again fixing the responsibilities to be the UBM state posteriors.

In our experiments, the process for extracting an i-vector to represent a variable-length speech recording is as follows. The first 20 mel-frequency cepstral coefficients (MFCCs) are extracted from the audio signal using a 25ms window every 10ms. MFCCs are an acoustic feature vector that captures information regarding the amplitude of different frequencies in a similar manner to how sounds are perceived by the human ear (Davis and Mermelstein, 1980). The MFCCs are appended with deltas and double deltas to help capture the dynamics of speech over time (e.g., Gales and Young, 2008). This results in a feature vector of 60 dimensions, with 100 frames (of feature vector) per second.

Speech activity detection (SAD) is then applied to remove any frames that do not contain speech. For this purpose we use a deep neural network (DNN)-based model trained on telephone and microphone data from a combination of Fisher (Cieri et al., a,b), Switchboard (Graff et al., 2001, 2004, 1999, 2002) and Mixer data (Cieri et al., 2007), as well as a 30-minute long dual-tone multi-frequency (DTMF) signal without speech, and a set of 3740 signals where speech from the Fisher corpora was corrupted with non-vocal music at different SNR levels. The ground truth labels used to train the DNN were obtained using our previous SAD system which consisted of a speech/non-speech hidden Markov model (HMM) decoder and various duration constraints. This system performed very well but was slow, complex and hard to retrain given new data. The labels for the corrupted data were obtained from the clean signals. As input to the SAD system we use MFCC features, mean and variance normalized over each waveform, except for the C_0 coefficient, for which the maximum rather than the mean is subtracted before dividing by the standard deviation. The normalized features are concatenated over a window of 31 frames. The resulting 620-dimensional feature vector forms the input to a DNN that consists of two hidden layers of sizes 500 and 100. The output layer of the DNN consists of two nodes trained to predict the posteriors for the speech and non-speech classes. These posteriors are converted into likelihood ratios using Bayes rule (assuming a prior of 0.5), and a threshold of 0.5 is applied to obtain the final

speech regions. For a more detailed description of the DNN-based SAD approach, please see (Graciarena et al., 2016).

The UBM is then estimated with an EM algorithm using the speech frames from a random subset of 10,000 samples (full recordings) of the data used to estimate the T_i matrices described next. The subspaces T_i are then estimated using the FULL training data described in Section 5.3, except that samples from languages for which only one or two speakers are available or where the language was unavailable or ambiguous are not discarded for this purpose. Finally, once the model is trained, the i-vectors for any audio sample can be extracted using only the speech frames, as in training.

The i-vectors can then be used for determining speaker similarity between two utterances using PLDA. For the experiments, we process the i-vectors with multiclass linear discriminant analysis (LDA) trained on the same training data used for PLDA, after which we subtract the mean over the training data and perform length normalization (Garcia-Romero and Espy-Wilson, 2011). The length-normalization step serves to better satisfy the Gaussianity assumption behind PLDA.

6. Results

In this section we compare results for different EM initialization techniques, parameter settings and training data for the proposed and the baseline PLDA techniques described in previous sections.

The nuisance condition for JPLDA in these experiments is the language spoken in the sample. During training, this label is known; during scoring, the label is marginalized to compute the LR, unless otherwise indicated. For TPLDA, on the other hand, we cannot take the mixture component to be the language spoken in the sample. This is because we do not have enough training speakers for each language to train a good PLDA model for each component. Hence, we consider a two-component model with a component modeling all English data and another component modeling the non-English data. This, as we will see, turns out to be a good model when matched data is available for training both components. Note that our implementation of TPLDA assumes that the mixture component is given both in training and in scoring. This is possible in our experiments because we have the language spoken during testing.

For SPLDA, FPLDA and JPLDA, the LDA dimension is set to 400; no dimensionality reduction is done in these cases but the data is still transformed by the LDA matrix, centered and length normalized. For TPLDA, on the other hand, we use an LDA dimension of 200, because we found that this value gives significantly better performance than keeping the original dimension of 400.

The speaker and language ranks for all experiments in this section are fixed to 200 and 16, respectively. These values were chosen for being optimal or approximately optimal for all methods under study (FPLDA, SPLDA and JPLDA) when using all available training data. The language rank of 16 is the largest rank that can be used for JPLDA. This value turned out to be optimal for JPLDA. FPLDA is largely insensitive to this parameter, giving very similar performance for language ranks between 5 and 16. Unless otherwise stated, all JPLDA results are obtained using $P(H_{SC}|H_{SS}) = P(H_{SC}|H_{DS}) = 0.5$. For TPLDA we use a diagonal matrix for the covariance of the noise model which proved to be slightly better than a full covariance.

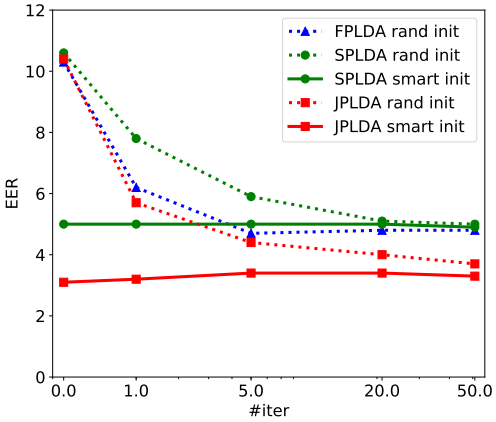


Figure 1: Comparison of performance as a function of the number of EM iterations on the Mixer test data using all available training data, for random and smart initialization for three different PLDA models. Note that a log scale is used for the x-axis.

All tuning decisions above were made based solely on the results on Mixer test data.

6.1. Initialization and Convergence of Training Procedure

We first show results for SPLDA, FPLDA and JPLDA as a function of the number of EM iterations run for the two initialization procedures, random and smart, explained in Sections 2.1 and 4.1. For FPLDA, no standard way exists of which we are aware to smartly initialize all parameters of the model. In this case, we only show results for random initialization. For this section, we use the FULL training data without subsetting and test on the Mixer data.

Results in Figure 1 show that EM iterations are essential when random initialization is used, leading to large gains over the initial random model as the iterations progress and converging to an approximately stable value when reaching 50 iterations. On the other hand, when smart initialization of SPLDA or JPLDA is used, EM iterations are not necessary on this data set. In fact, JPLDA performance with smart initialization slightly degrades for larger number of iterations, probably due to overfitting of the training data. For this reason, for the rest of the experiments we use only one iteration of EM for JPLDA, though zero iterations could also be safely used.

6.2. Prior Probability of Same Language in JPLDA

In this section we show JPLDA results on the Mixer development set when using all available training data as a function of the prior probabilities of same language, $P(H_{SC}|H_{SS})$ and $P(H_{SC}|H_{DS})$ (see Section 4.2). We fix these two parameters to the same value and sweep

this value between 0 and 1 at 0.1 steps. We show results on the full test data but also split the data into same-language and cross-language trials. We compare these results with those we would obtain by knowing the language of each sample a priori and using this knowledge during scoring to set the priors appropriately as explained in Section 4.2.

Figure 2 shows performance as a function of the probability of same language parameter. Values below 0.1 are optimal for the cross-language trials, while values above 0.1 and below 1.0 are optimal for same-language trials. The fact that a probability of same language of 1.0 is not optimal for same-language trials might be due to some samples including code-switching, making trials involving these samples not strictly same-language trials. Further, a value lower than 1.0 for same-language trials may better accommodate the variation in accent that takes place when people speak to different interlocutors. Once all trials are pooled together, values between 0.3 and 0.8 give almost identical performance. For this range of values, we can also see that performance is very close to what we would obtain if the language of the test files was known during scoring (the red dashed line in the plot). This performance is obtained by setting the probability of same language to 1.0 for same-language trials and to 0.0 for cross-language trials. For the remaining experiments, we use a probability of same language of 0.5.

Figure 2 shows that the same-language and different-language subsets have a significantly lower EER than the pooled set of trials. This is due to the fact that the scores for both sets of trials are misaligned with each other. That is, the EER threshold is different for both sets, leading to a larger EER than that for either set once the trials are pooled together. As we will see in Section 6.4, Figure 6, this effect is actually more salient in standard PLDA approaches, with JPLDA mitigating the problem, though not fully solving it.

6.3. Method Comparison

We now compare the performance of the four methods on all test sets from Mixer and LASRS divided by microphone type using the two training sets: FULL and MONOLING.

The top plot in Figure 3 shows that FPLDA gives slightly better performance than SPLDA for some channels (mostly the telephone ones) when the FULL training data is used. For this reason, for the remaining experiments in this paper, we use FPLDA as the baseline.

Comparing the two methods that consider language labels during training, TPLDA and JPLDA, on the top plot in Figure 3, we see that they both give significant gains over the baselines on Mixer data, where the channel is matched to the majority of the training data's channel. In this case, both approaches succeed in mitigating the effect of language variability. On the other hand, when the channel is not exactly the same as the one observed most in training, TPLDA fails to generalize, leading to consistently worse performance than JPLDA. This is reasonable: while alternative microphone data is observed for the English training data, only telephone data is observed for the non-English data. This implies that the PLDA mixture corresponding to non-English data in TPLDA was only learned with telephone data, resulting in the poorer performance observed on some of the LASRS channels. On the other hand, JPLDA can leverage the information about alternative microphones learned from English data for all languages, since the matrix that models this variability is shared across languages.

In the bottom plot in Figure 3, we see that when only a single language from each speaker is available for training (that is, the within speaker variation due to language is not

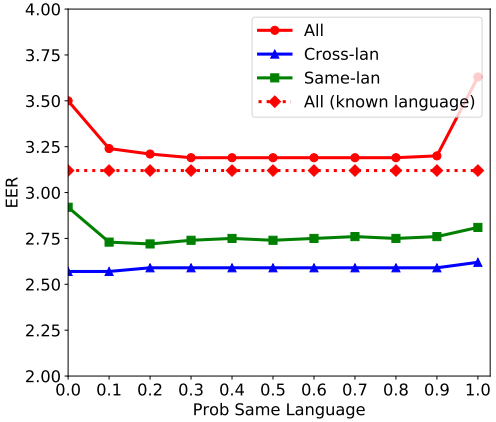


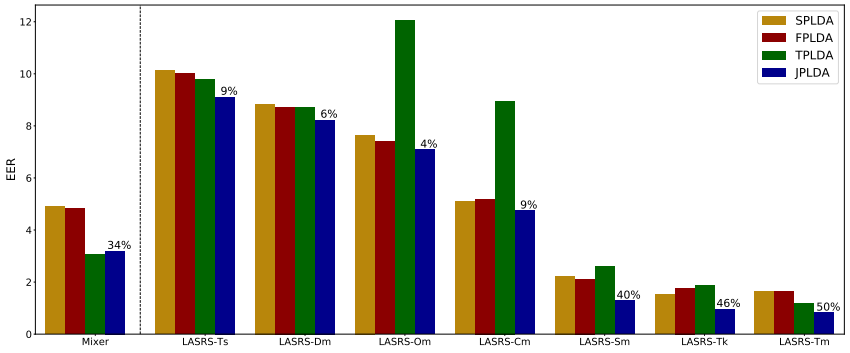
Figure 2: Comparison of JPLDA performance as a function of the prior probability of same language on the Mixer test data using the FULL training data. The known-language line corresponds to the performance on all trials when using the information about the test language during scoring.

observed in training), TPLDA leads to a large degradation over both baselines. Note that, as far as we know, TPLDA had not been tested under this challenging scenario. Rather, it was tested using training data where each class of interest (e.g., a face) was seen under all possible conditions (front and profile) (Li et al., 2012). When each class is seen under a single condition, the TPLDA model basically degenerates to separate (untied) PLDA models, each learned on the data from its own condition. This implies that the resulting mixture will be unable to model the cross-language variability, which results in extremely degraded performance on the cross-language trials. Indeed, our results indicate that the same-language trials get reasonable TPLDA performance (results not shown here), it is the degradation on the cross-language trials that affects the overall performance as observed in the plot.

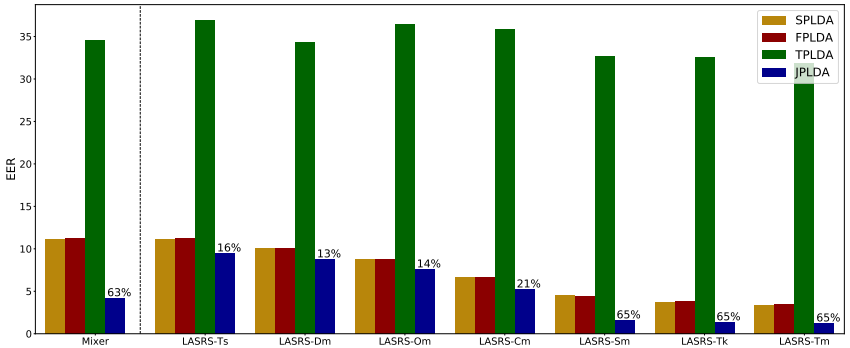
Finally, focusing on JPLDA, we see that significant gains are observed compared to both baselines using both training sets, with larger relative gains when the training data contains only a single language per speaker, in which case we find gains from 13% of up to 65% relative to the FPLDA baseline.

6.4. Training Data Comparison

Finally, in this section we compare the FPLDA baseline and JPLDA using the two training sets defined in Section 5.3 and their subsets, where we discard telephone samples from speakers that only have English samples in an attempt to achieve a better balance between English and non-English samples and telephone and microphone samples.



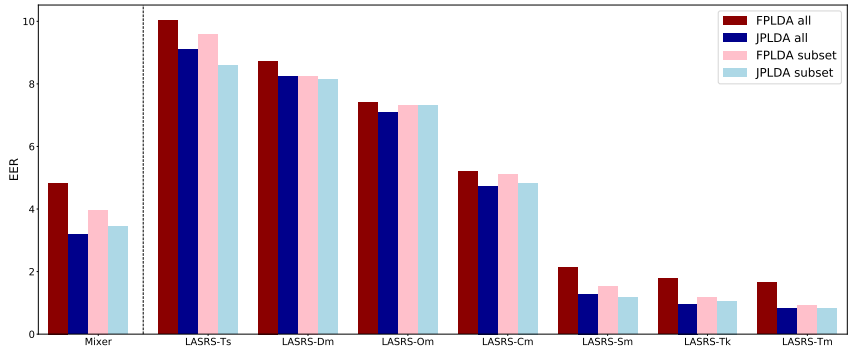
(a) Training data: FULL



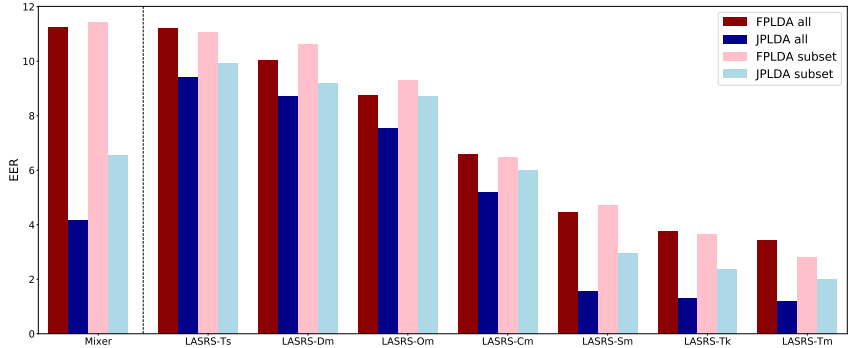
(b) Training data: MONOLING

Figure 3: Comparison of performance for four PLDA methods on all test sets using both training sets, FULL and MONOLING. The numbers on top of the JPLDA bars show the relative gain of JPLDA relative to FPLDA.

Figure 4 shows that, for FPLDA, using the subset is significantly better than using the full training set for both training conditions, FULL and MONOLING, for most test conditions. That is, FPLDA benefits from having a more balanced distribution of conditions within the training data. This is because, in standard PLDA, the samples from all speakers are assumed to follow the same distribution, regardless of whether these samples are all in English, or both in English and some other language. Hence, if a large proportion of speakers only have English samples, the parameters in the PLDA model will be mostly determined by what is optimal for these speakers, degrading the performance on non-English and cross-language trials.



(a) Training data: FULL



(b) Training data: MONOLING

Figure 4: Comparison of performance for FPLDA and JPLDA on all test sets using the two training sets, FULL and MONOLING. For each case, we compare using the full training set and a subset where we discard telephone samples from speakers that only have English samples in the FULL training set.

On the other hand, JPLDA does not seem to require subsetting the data[2]. In fact, for the FULL training condition, JPLDA leads to similar or better performance (using either the full training set or the subset) than FPLDA using the subset. For the MONOLING condition, the advantage of JPLDA over FPLDA is much larger than for the FULL training set, consistently showing significant gains over the best FPLDA result. Further, for this training condition we

2. Note that the EER on the better performing test sets (LASRS-Sm, LASRS-Tk and LASRS-Tm) corresponds to very few misses, making that metric somewhat unreliable on those sets. However, DET curves shown later in the section complement the EER results, supporting the overall conclusions made based on EERs.

see a consistent trend showing that JPLDA benefits from using the full training set, which indicates that, contrary to PLDA, JPLDA can handle the imbalance in the full set of data, successfully leveraging the additional samples missing from the subset.

To complement the EER results in the bar plots, Figure 5 shows the DET curves for all test sets. We show these curves for the more challenging training condition, MONOLING, where JPLDA gives the biggest advantage over FPLDA. The plots show that the gains are not specific to the EER operating point. Rather, JPLDA gives a significant gain over FPLDA over a very wide range of operating points corresponding to miss and false alarms rates between 0.01% to 40%. Further, we also see the advantage of using all the available training data rather than just the subset when using JPLDA, while the opposite is true for FPLDA, as already observed in the EER bar plots.

Finally, Figure 6 shows EER results on Mixer test data using the two training sets and their subsets for all trials (as in previous bar plots) as well as for same-language and cross-language trials. The performance on all trials is the same as in Figure 4. These plots show that: (1) Both same-language and cross-language trials benefit from using JPLDA, particularly for the MONOLING training conditions. (2) The JPLDA benefit from using the complete training sets holds for both same-language and cross-language subsets of trials. (3) The FPLDA benefit from using the subset only holds on the same-language trials; cross-language trial performance is degraded or unchanged by subsetting the training data. And (4) the relative gain from JPLDA is larger once same-language and cross-language trials are pooled together. This last observation indicates that JPLDA is not only improving discrimination for each type of trial (same-language and cross-language), but it is also aligning the distributions of these two types of trials such that when they are pooled together, the relative gain from using JPLDA is emphasized. Yet, as we can see, JPLDA does not appear to fully solve the problem, since the pooled performance is still somewhat worse than that of the subsets. We plan to study the source of the remaining misalignment in the near future.

7. Conclusions

We have proposed a generalization of PLDA where within-class variability factors are no longer considered independent across samples. The method assumes that the identity of a nuisance condition is known during training and ties the latent variable corresponding to the within-class variability across all samples with the same nuisance condition label. During scoring, a likelihood ratio is computed as for standard PLDA by marginalizing over the nuisance condition. Hence, the identity of the nuisance condition can be unknown during testing.

We show results on a multilingual speaker recognition task comparing the proposed method with two types of standard PLDA models as well as to a tied PLDA model where the nuisance condition is used to determine the component in a mixture of PLDA models. Our results show that large relative gains are obtained from using JPLDA when the training data contains few or no speakers with data in more than one language. That is, the JPLDA model is able to extrapolate the effect of language from a small proportion or even zero training speakers with data from more than one language. Standard PLDA models are only able to mitigate the effect of language when exposed to a significant proportion of training speakers with data in more than one language.

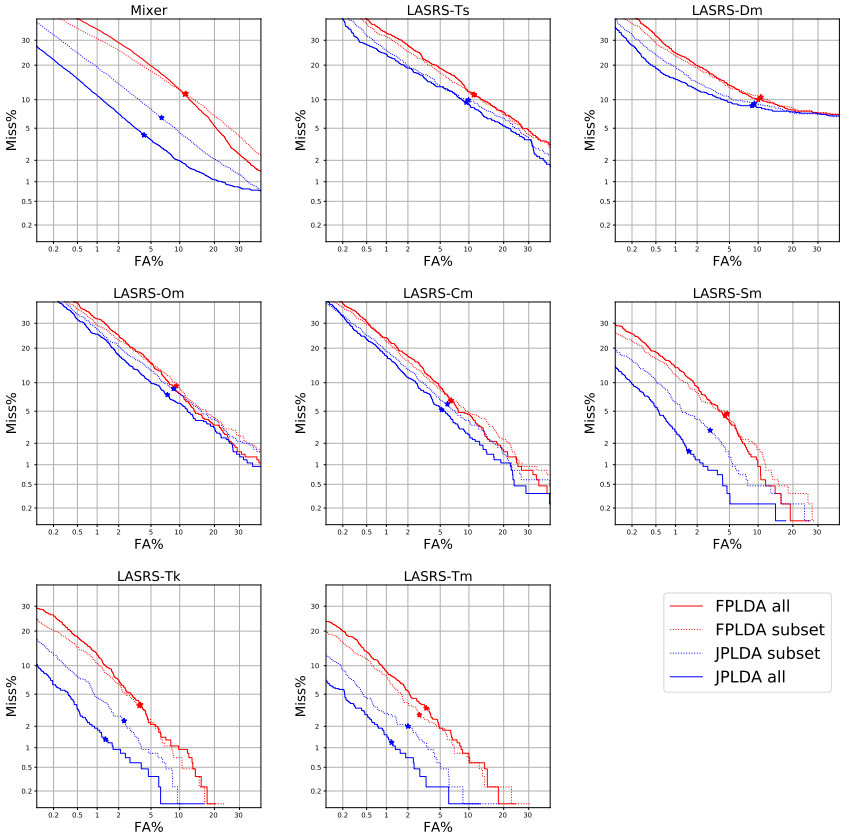
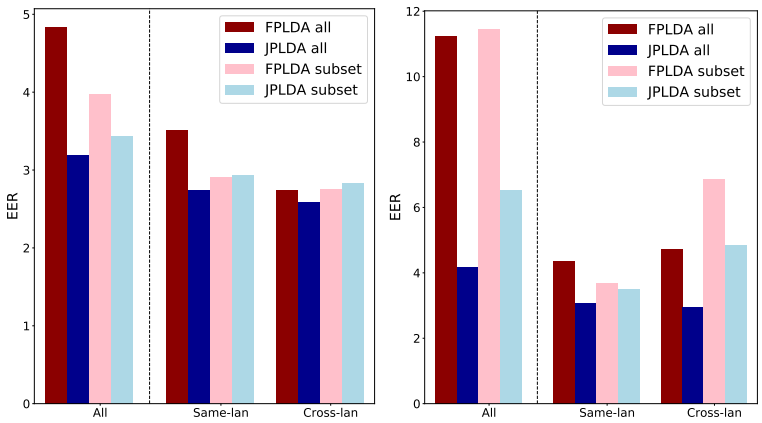


Figure 5: DET curves for FPLDA and JPLDA on all test sets using the MONOLING training set and its subset. The marker over each curve corresponds to the EER point for that system.

The proposed JPLDA method can be used for any task for which standard PLDA is used whenever a discrete nuisance condition is known during training. Examples include speaker recognition using channel, speaking style or language labels, among others, as the sample-dependent nuisance condition, and face recognition using pose as sample-dependent nuisance condition. The strength of JPLDA lies in its ability to extrapolate the effect that



(a) Training data: FULL (b) Training data: MONOLING

Figure 6: Comparison of performance for FPLDA and JPLDA on the Mixer test set on all trials as well as on same-language and cross-language subsets, using the two training sets, FULL and MONOLING, and their subsets.

the nuisance condition has on the samples based on few or even no classes (speakers or faces) seen under several nuisance conditions.

The proposed approach introduces the additional requirement with respect to the original PLDA approach that the identity of the nuisance condition be known during training. In future work, we will explore the possibility of automatically detecting the nuisance conditions, using classifiers trained on data for which the factors are known or using clustering with distance metrics designed to reflect the nuisance of interest. Finally, an interesting generalization of the proposed approach would be to allow for more than one sample-dependent nuisance condition. These are directions we plan to explore in the near future.

Acknowledgments

This material is based upon work supported partly by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-15-C-0037. The views, opinions, and/or findings contained in this article are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. Distribution Statement A: Approved for Public Release, Distribution Unlimited.

Appendix A. JPLDA Formulas

In this appendix we derive the probabilities that are needed during training with the EM algorithm and during scoring with the likelihood ratio for the proposed JPLDA method. The derivations closely follow the ones for standard PLDA in (Brümmer, 2010a) with one main difference: in the new model, most probabilities cannot be formulated by speaker and then multiplied to get the total probabilities, as is usually done for standard PLDA, since the condition introduces dependencies across samples from different speakers. Instead, we formulate all probabilities over all samples.

In the following, we take

$$\begin{aligned} Y &= \{y_1, \ldots, y_S\} \\ X &= \{x_1, \ldots, x_C\} \\ M &= \{m_1, \ldots, m_N\} \end{aligned}$$

where S, C and N are the total number of speakers, conditions, and samples, respectively. Further, we assume that $\mu = 0$. In the general case, as done for standard PLDA, this parameter is set to the global mean of the training data and subtracted from all samples before running EM or scoring.

A.1. Probability Distributions

The joint prior for the hidden variables for all the data is given by

$$p(Y, X) = p(X)p(Y) \propto \exp(-\frac{1}{2}\sum_s y_s^T y_s - \frac{1}{2}\sum_c x_c^T x_c), \tag{7}$$

and the full data likelihood is given by

$$\begin{aligned} p(M|Y, X, \lambda) &= \prod_i N(m_i|Vy_{s_i} + Ux_{c_i}, D^{-1}) \\ &\propto \exp\sum_i \left(-\frac{1}{2}(m_i - Vy_{s_i} - Ux_{c_i})^T D(m_i - Vy_{s_i} - Ux_{c_i}) + \frac{1}{2}\log|D|\right). \end{aligned} \tag{8}$$

The joint probability is proportional, as a function of M, X and Y, to the product of the likelihood and the prior,

$$\begin{aligned} p(M, Y, X|\lambda) \propto \exp\Bigg[&\sum_i \left(-\frac{1}{2}m_i^T Dm_i + m_i^T DVy_{s_i} + m_i^T DUx_{c_i} - x_{c_i}^T Jy_{s_i}\right) \\ &-\frac{1}{2}\sum_s y_s^T L_s y_s - \frac{1}{2}\sum_c x_c^T K_c x_c\Bigg] \end{aligned}$$

where

$$\begin{aligned} J &= U^T DV \\ K_c &= n_c U^T DU + I \\ L_s &= n_s V^T DV + I \end{aligned}$$

where n_c and n_s are the number of samples for condition c and speaker s, respectively.

We can now compute the posterior from two factors:

$$p(Y, X|M, \lambda) = p(Y|X, M, \lambda)p(X|M, \lambda)$$

The outer posterior is proportional, as a function of Y, to the joint probability. Keeping only the terms in the joint probability that depend on Y we get

$$p(Y|X, M, \lambda) \propto \exp\left[\sum_i \left(m_i^T DV y_{s_i} - x_{c_i}^T J y_{s_i}\right) - \frac{1}{2}\sum_s y_s^T L_s y_s\right] \propto \prod_s N(y_s|\hat{y}_s, L_s^{-1}) \quad (9)$$

where

$$\begin{aligned}
\hat{y}_s &= \tilde{y}_s - L_s^{-1} J^T \bar{x}_s \\
\tilde{y}_s &= L_s^{-1} V^T D f_s \\
\bar{x}_s &= \sum_{i|s_i=s} x_{c_i} \\
f_s &= \sum_{i|s_i=s} m_i
\end{aligned}$$

Marginalizing that distribution we can extract the posterior for a single latent variable y_s:

$$p(y_s|X, M, \lambda) = N(y_s|\hat{y}_s, L_s^{-1}) \quad (10)$$

The inner posterior is proportional to the joint probability of X and M:

$$\begin{aligned}
p(X|M, \lambda) &\propto p(M, X|\lambda) = \frac{p(Y, X, M|\lambda)}{p(Y|X, M, \lambda)}\Big|_{Y=0} \\
&\propto \exp\left(\sum_c g_c^T DU x_c - \frac{1}{2}\sum_c x_c^T K_c x_c - \sum_c x_c^T J\bar{\bar{y}}_c + \frac{1}{2}\sum_s \bar{x}_s^T J L_s^{-1} J^T \bar{x}_s\right)
\end{aligned}$$

where we have used the *candidate's formula* (Besag, 1989) to obtain the joint probability and where

$$\begin{aligned}
g_c &= \sum_{i|c_i=c} m_i \\
\bar{\bar{y}}_c &= \sum_{i|c_i=c} \tilde{y}_{s_i} = \sum_{i|c_i=c} L_{s_i}^{-1} V^T D f_{s_i}
\end{aligned}$$

We now define vectors which are the concatenation of all individual latent vectors:

$$\begin{aligned}
\boldsymbol{X} &= [x_1^T \dots x_C^T]^T \\
\boldsymbol{Y} &= [y_1^T \dots y_S^T]^T
\end{aligned}$$

and, similarly, for all other vectors. Converting the sums into matrix form, we get

$$p(\boldsymbol{X}|M, \lambda) \propto \exp\left(\boldsymbol{X}^T \Phi - \frac{1}{2}\boldsymbol{X}(T_2 - H^T T_4 H)\boldsymbol{X}\right) \propto N(\boldsymbol{X}|\hat{\boldsymbol{X}}, \Sigma) \quad (11)$$

where

$$\begin{aligned}
\Sigma &= (T_2 - H^T T_4 H)^{-1} \quad (12) \\
\hat{\boldsymbol{X}} &= \Sigma\Phi \\
\Phi &= T_1 \boldsymbol{G} - T_3 \bar{\bar{\boldsymbol{Y}}} \\
T_1 &= \text{diagn}(U^T D, C) \\
T_2 &= \text{diag}(K_1, \dots, K_C) \\
T_3 &= \text{diagn}(J, C) \\
T_4 &= \text{diag}(JL_1^{-1}J^T, \dots, JL_S^{-1}J^T)
\end{aligned}$$

where $\text{diagn}(M, N)$ is a block diagonal matrix with matrix M in each of N blocks and $\text{diag}(T_1, \dots, T_N)$ is a block diagonal matrix with blocks given by matrices T_i. The matrix H is of size SR_xxCR_x, where block $H_{s,c}$ (R_x rows and columns starting at position (sR_x, cR_x) in H) is given by:

$$H_{s,c} = n_{s,c} I$$

where I is the identity matrix of size R_x and $n_{s,c}$ is the number of times that condition c occurs for speaker s, which could be zero.

As for the outer posterior, we can marginalize the distribution in Equation (11) to get the distribution for an individual x_c

$$p(x_c|M, \lambda) = N(x_c|\hat{x}_c, \Sigma_c) \quad (13)$$

where Σ_c and $\hat{x}_c$ are the blocks corresponding to latent variable c in Σ and $\hat{X}$.

A.2. EM Algorithm

The EM auxiliary function is given by the expected value of the log-likelihood with respect to the posterior probability of the hidden variables given the data and the previously estimated model parameters, λ_{k-1}.

$$\begin{aligned}
Q(\lambda_k|\lambda_{k-1}) &= E_{X,Y|M,\lambda_{k-1}}\left[\log p(M, Y, X|\lambda_k)\right] \\
&= \frac{N}{2}\log|D| - \frac{1}{2}\text{tr}(SD) - \frac{1}{2}\text{tr}(RW^T DW) + \text{tr}(TDW) + \text{const}
\end{aligned}$$

where

$$\begin{aligned}
W &= [UV] \\
S &= \sum_i m_i m_i^T \\
R &= \sum_i \langle z_i z_i^T \rangle \\
T &= \sum_i \langle z_i \rangle m_i^T \\
z_i &= [x_{c_i}^T y_{s_i}^T]^T
\end{aligned}$$

where the $\langle$ and $\rangle$ symbols stand for the expectation with respect to the distribution of z_i given the data M and the previous parameters λ_{k-1}.

A.2.1. M-Step

Now, differentiating Q with respect to D and W and setting to zero, we get that

$$\begin{aligned} D^{-1} &= \frac{1}{N}(S - WT) \\ W^T &= R^{-1}T \end{aligned}$$

So, the matrices are estimated exactly the same way as in the standard PLDA approach (Brümmer, 2010a). The additional complexity of JPLDA lies in the forms that R and T take.

A.2.2. E-Step

To find T and R we use the equations we have obtained for the posterior distributions of x_c and y_s (Equations 10 and 13). The two components of T are given by:

$$\begin{aligned} T_x &= \sum_i \langle x_{c_i} \rangle m_i^T = \sum_c \hat{x}_c g_c^T \\ T_y &= \sum_s \langle y_{s_i} \rangle m_i^T = \sum_s L_s^{-1}(V^T D f_s - J^T \bar{\hat{x}}_s) f_s^T \end{aligned}$$

where we use the law of total expectations to get the expectation of y_s from its conditional expectation and where

$$\bar{\hat{x}}_s = \sum_{i|s_i=s} \hat{x}_{c_i}$$

Finally, we can get the components of R as follows:

$$\begin{aligned} R_{xx} &= \sum_i \langle x_{c_i} x_{c_i}^T \rangle = \sum_c n_c(\Sigma_c + \hat{x}_c \hat{x}_c^T) \\ R_{yx} &= \sum_i \langle y_{s_i} x_{c_i}^T \rangle = \sum_s \left[\tilde{y}_s \bar{\hat{x}}_s^T - \tilde{J}^T \sum_{i|s_i=s} \sum_{j|s_j=s} \left[\hat{x}_{c_i} \hat{x}_{c_j}^T + \Sigma_{c_j,c_i} \right] \right] \\ R_{yy} &= \sum_i \langle y_{s_i} y_{s_i}^T \rangle = \sum_s n_s \left[L_s^{-1} + \tilde{y}_s \tilde{y}_s^T - \tilde{y}_s \bar{\hat{x}}_s^T \tilde{J} - \tilde{J}^T \bar{\hat{x}}_s \tilde{y}_s^T + \tilde{J}^T \langle \bar{x}_s \bar{x}_s^T \rangle \tilde{J} \right] \end{aligned}$$

where $\tilde{J} = JL_s^{-1}$ and Σ_{c_j,c_i} is the block in matrix Σ (Equation 12) corresponding to latent variables c_i and c_j.

A.3. Scoring

In this paper we assume all trials are composed of a single enrollment and a single test sample. That is, the sets E and T in Equation (6) are each composed of a single vector, m_E and m_T, respectively. M is then given by $\{m_E, m_T\}$. We can now use the formulas derived above to obtain the LR in Equation (6) where we need to compute four probabilities for the data given different hypotheses. The probabilities can be obtained using the candidate's formula (Besag, 1989):

$$p(M|h_s, h_c) = \left. \frac{p(M|X_{h_c}, Y_{h_s}) p(X_{h_c}) p(Y_{h_s})}{p(Y_{h_s}|X_{h_c}, M) p(X_{h_c}|M, h_s)} \right|_{X_{h_c}=0, Y_{h_s}=0} \tag{14}$$

where $h_s \in \{H_{SS}, H_{DS}\}$ and $h_c \in \{H_{SC}, H_{DC}\}$, and where

$$X_{h_c} = \begin{cases} \{x\}, & \text{if } h_c = H_{SC} \\ \{x_E, x_T\}, & \text{if } h_c = H_{DC} \end{cases}$$

$$Y_{h_s} = \begin{cases} \{y\}, & \text{if } h_s = H_{SS} \\ \{y_E, y_T\}, & \text{if } h_s = H_{DS} \end{cases}$$

That is, the latent variables are two independent vectors for the different-condition and different-speaker hypotheses and a single vector for the same-condition and same-speaker hypotheses.

The likelihood in the numerator of Equation (14) is the same for all four combination of hypotheses since, regardless of whether the latent variables are tied or not, Equation (8) has the same form. Hence, that term cancels out in the computation of the LR. The priors, on the other hand, will have one factor for the same-speaker or same-condition case and two identical factors, once evaluated at 0, for the different-speaker or different-condition case. All that is left to do is compute the inner and outer posteriors in the denominator of Equation (14) and evaluate them at 0.

The outer posterior $p(Y_{h_s}|X_{h_c}, M)$, given by Equation (9), takes the same value for both values of h_c when the latent variables are set to 0. Its logarithm is given by

$$\log p(Y_{h_s}|X_{h_c}, M)|_0 = \begin{cases} \frac{1}{2}k + \frac{1}{2}\log|L_1| - \frac{1}{2}(\tilde{m}_E + \tilde{m}_T)^T L_1^{-1}(\tilde{m}_E + \tilde{m}_T), & \text{if } h_s = H_{SS} \\ k + \log|L_2| - \frac{1}{2}\tilde{m}_E^T L_2^{-1}\tilde{m}_E - \frac{1}{2}\tilde{m}_T^T L_2^{-1}\tilde{m}_T, & \text{if } h_s = H_{DS} \end{cases}$$

where $L_2 = V^T D V + I$, $L_1 = 2V^T D V + I$, $\tilde{m}_E = V^T D m_E$, $\tilde{m}_T = V^T D m_T$ and $k = -R_y \log(2\pi)$.

The inner posterior is given in Equation (11). For the scoring scenario, its logarithm is given by

$$\log p(X_{h_c}|M, h_s) = \begin{cases} -\frac{1}{2}R_x k - Q_{h_c,h_s}, & \text{if } h_c = H_{SC} \\ -R_x k - Q_{h_c,h_s}, & \text{if } h_c = H_{DC} \end{cases}$$

where $Q_{h_c,h_s} = \frac{1}{2}\log|\Sigma_{h_c,h_s}| + \frac{1}{2}\Phi_{h_c,h_s}^T \Sigma_{h_c,h_s} \Phi_{h_c,h_s}$ with

$$\begin{aligned} \Sigma_{H_{SC},H_{SS}} &= \left[2U^T D U + I - 4JL_S^{-1}J^T\right]^{-1} \\ \Sigma_{H_{SC},H_{DS}} &= \left[2U^T D U + I - 2JL_D^{-1}J^T\right]^{-1} \\ \Sigma_{H_{DC},H_{SS}} &= \left[\text{diagn}(K_D, 2) - [II]^T JL_S^{-1}J^T[II]\right]^{-1} \\ \Sigma_{H_{DC},H_{DS}} &= \left[\text{diagn}(K_D, 2) - \text{diagn}(JL_D^{-1}J^T, 2)\right]^{-1} \\ \Phi_{H_{SC},H_{SS}} &= \left((\hat{m}_E + \hat{m}_T) - 2JL_S^{-1}(\tilde{m}_E + \tilde{m}_T)\right) \\ \Phi_{H_{SC},H_{DS}} &= \left((\hat{m}_E + \hat{m}_T) - JL_D^{-1}(\tilde{m}_E + \tilde{m}_T)\right) \\ \Phi_{H_{DC},H_{SS}} &= \begin{bmatrix} \hat{m}_E - JL_S^{-1}(\tilde{m}_E + \tilde{m}_T) \\ \hat{m}_T - JL_S^{-1}(\tilde{m}_E + \tilde{m}_T) \end{bmatrix} \\ \Phi_{H_{DC},H_{DS}} &= \begin{bmatrix} \hat{m}_E - JL_D^{-1}\tilde{m}_E \\ \hat{m}_T - JL_D^{-1}\tilde{m}_T \end{bmatrix} \end{aligned}$$

where $\hat{m}_E = U^T D m_E$ and $\hat{m}_T = U^T D m_T$.

Finally, since the outer posterior is independent of the condition hypothesis, the logarithm of the LR (LLR) can be written as a sum of terms involving the outer posterior and the inner posteriors

$$\text{LLR} = \text{LLR}_o + \text{LLR}_i$$

where

$$\begin{aligned}
\text{LLR}_o &= \log \frac{p(Y_{H_{DS}}|X_{h_c}, M)}{p(y)p(Y_{H_{SS}}|X_{h_c}, M)}\Big|_0 \\
&= \log|L_2| - \frac{1}{2}\log|L_1| + \frac{1}{2}\tilde{m}_E^T(L_1^{-1} - L_2^{-1})\tilde{m}_E + \frac{1}{2}\tilde{m}_T^T(L_1^{-1} - L_2^{-1})\tilde{m}_T + \tilde{m}_T^T L_1^{-1}\tilde{m}_E \\
\text{LLR}_i &= \log \frac{p(x)p(X_{H_{SC}}|M, H_{SS})^{-1}P_{SS} + p(x)^2 p(X_{H_{DC}}|M, H_{SS})^{-1}P_{DS}}{p(x)p(X_{H_{SC}}|M, H_{DS})^{-1}P_{SD} + p(x)^2 p(X_{H_{DC}}|M, H_{DS})^{-1}P_{DD}}\Big|_0 \\
&= \log\left(\exp(Q_{H_{SC},H_{SS}})P_{SS} + \exp(Q_{H_{DC},H_{SS}})P_{DS}\right) - \\
&\quad \log\left(\exp(Q_{H_{SC},H_{DS}})P_{SD} + \exp(Q_{H_{DC},H_{DS}})P_{DD}\right)
\end{aligned}$$

where we use the fact that $\log p(x)|_0 = -\frac{1}{2}R_x k$ and $\log p(y)|_0 = -\frac{1}{2}R_y k$, and where $P_{SS} = P(H_{SC}|H_{SS})$, $P_{SD} = P(H_{SC}|H_{DS})$, $P_{DS} = P(H_{DC}|H_{SS})$, and $P_{DD} = P(H_{DC}|H_{DS})$.

References

R. Auckenthaler, M. J. Carey, and J. S. D. Mason. Language dependency in text-independent speaker verification. In *Proc. ICASSP*, Salt Lake City, May 2001.

S. D. Beck, R. Schwartz, and H. Nakasone. A bilingual multi-modal voice corpus for language and speaker recognition (LASR) services. In *Proc. Odyssey-04*, Toledo, Spain, May 2004.

J. Besag. A candidate's formula: A curious result in bayesian prediction. *Biometrika*, 76(1): 183–183, 1989.

N. Brümmer. EM for probabilistic LDA. Technical report, Available at https://sites.google.com/site/nikobrummer/EMforPLDA.pdf, 2010a.

N. Brümmer. EM for simplified PLDA. Technical report, Available at https://sites.google.com/site/nikobrummer/EMforSPLDA.pdf, 2010b.

N. Brümmer. Vb calibration to improve the interface between phone recognizer and i-vector extractor. *arXiv:1510.03203*, 2015.

L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matejka, and N. Brümmer. Discriminatively trained probabilistic linear discriminant analysis for speaker verification. In *Proc. ICASSP*, Prague, May 2011.

C. Cieri, D. Graff, O. Kimball, D. Miller, and K. Walker. Fisher english training speech part 1 speech ldc2004s13, a.

C. Cieri, D. Graff, O. Kimball, D. Miller, and K. Walker. Fisher english training speech part 2 speech ldc2005s13, b.

C. Cieri, L. Corson, D. Graff, and K. Walker. Resources for new research directions in speaker recognition: The Mixer 3, 4 and 5 corpora. In *Proc. Interspeech*, Antwerp, Belgium, August 2007.

S. Cumani, O. Plchot, and P. Laface. On the use of i-vector posterior distributions in probabilistic linear discriminant analysis. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 22(4):846–857, 2014.

S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980.

N. Dehak, P.J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *IEEE Trans. Audio, Speech, and Lang. Process.*, 19(4):788–798, May 2011.

L. Ferrer. Joint probabilistic linear discriminant analysis. *arXiv:1704.02346v2*, 2017.

L. Ferrer, H. Bratt, L. Burget, H. Cernocky, O. Glembek, M. Graciarena, A. Lawson, Y. Lei, P. Matejka, O. Plchot, and N. Scheffer. Promoting robustness for speaker modeling in the community: The PRISM evaluation set. In *Proceedings of SRE11 Analysis Workshop*, Atlanta, USA, December 2011.

M. Gales and S. Young. The application of hidden Markov models in speech recognition. *Foundations and trends in signal processing*, 1(3):195–304, 2008.

D. Garcia-Romero and C.Y. Espy-Wilson. Analysis of i-vector length normalization in speaker recognition systems. In *Proc. Interspeech*, Florence, Italy, August 2011.

D. Garcia-Romero, X. Zhou, and C. Y. Espy-Wilson. Multicondition training of gaussian PLDA models in i-vector space for noise and reverberation robust speaker recognition. In *Proc. ICASSP*, pages 4257–4260, Kyoto, March 2012. IEEE.

M. Graciarena, L. Ferrer, and V. Mitra. The SRI system for the NIST OpenSAD 2015 speech activity detection evaluation. In *Proc. Interspeech*, San Francisco, September 2016.

D. Graff, K. Walker, and A. Canavan. Switchboard-2 phase II LDC99S79, 1999. URL `https://catalog.ldc.upenn.edu/LDC99S79`.

D. Graff, K. Walker, and D. Miller. Switchboard cellular part 1 audio LDC2001S13, 2001. URL `https://catalog.ldc.upenn.edu/LDC2001S13`.

D. Graff, D. Miller, and K. Walker. Switchboard-2 phase III LDC2002S06, 2002. URL `https://catalog.ldc.upenn.edu/LDC2002S06`.

D. Graff, K. Walker, and D. Miller. Switchboard cellular part 2 audio LDC2004S07, 2004. URL `https://catalog.ldc.upenn.edu/LDC2004S07`.

S. Ioffe. Probabilistic linear discriminant analysis. In *Proc. of the 9th European Conference on Computer Vision*, Graz, Austria, 2006.

P. Kenny. Bayesian speaker verification with heavy-tailed priors. In *Proc. Odyssey-10*, Brno, Czech Republic, June 2010. Keynote presentation.

Y. Lei, L. Burget, L. Ferrer, M. Graciarena, and N. Scheffer. Towards noise robust speaker recognition using probabilistic linear discriminant analysis. In *Proc. ICASSP*, Kyoto, March 2012.

P. Li, Y. Fu, U. Mohammed, J. Elder, and S. Prince. Probabilistic models for inference about identity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):144–157, 2012.

M.-W. Mak, X. Pang, and J.-T. Chien. Mixture of plda for noise robust i-vector speaker verification. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(1):130–142, 2016.

A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET curve in assessment of detection task performance. In *Proc. Eurospeech*, Rhodes, Greece, September 1997.

P. Matejka, O. Glembek, F. Castaldo, J. Alam, O. Plchot, P. Kenny, L. Burget, and J. Cernocky. Full-covariance UBM and heavy-tailed PLDA in i-Vector speaker verification. In *Proc. Interspeech*, Florence, Italy, August 2011.

A. Misra and J. H. Hansen. Spoken language mismatch in speaker verification: An investigation with NIST-SRE and CRSS Bi-ling corpora. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pages 372–377. IEEE, 2014.

S. Prince. Probabilistic linear discriminant analysis for inferences about identity. In *Proceedings of the International Conference on Computer Vision*, 2007.

A. Rozi, D. Wang, L. Li, and T. F. Zheng. Language-aware plda for multilingual speaker recognition. In *Coordination and Standardization of Speech Databases and Assessment Techniques (O-COCOSDA), 2016 Conference of The Oriental Chapter of International Committee for*, pages 161–165. IEEE, 2016.

M. Senoussaoui, P. Kenny, N. Brümmer, N.mmer, E. De Villiers, and P. Dumouchel. Mixture of PLDA models in i-vector space for gender-independent speaker recognition. In *Proc. Interspeech*, pages 25–28, Florence, Italy, August 2011.

Z. Shi, L. Liu, and R. Liu. Multi-view (joint) probability linear discrimination analysis for multi-view feature verification. *arXiv:1704.06061*, 2017.

A. Sizov, K. A. Lee, and T. Kinnunen. Unifying probabilistic linear discriminant analysis variants in biometric authentication. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 464–475. Springer, 2014.

Journal of Machine Learning Research 20 (2019) 1-49 Submitted 4/18; Revised 10/18; Published 1/19

Group Invariance, Stability to Deformations, and Complexity of Deep Convolutional Representations

Alberto Bietti alberto.bietti@inria.fr
Julien Mairal julien.mairal@inria.fr
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France*

Editor: Lorenzo Rosasco

Abstract

The success of deep convolutional architectures is often attributed in part to their ability to learn multiscale and invariant representations of natural signals. However, a precise study of these properties and how they affect learning guarantees is still missing. In this paper, we consider deep convolutional representations of signals; we study their invariance to translations and to more general groups of transformations, their stability to the action of diffeomorphisms, and their ability to preserve signal information. This analysis is carried by introducing a multilayer kernel based on convolutional kernel networks and by studying the geometry induced by the kernel mapping. We then characterize the corresponding reproducing kernel Hilbert space (RKHS), showing that it contains a large class of convolutional neural networks with homogeneous activation functions. This analysis allows us to separate data representation from learning, and to provide a canonical measure of model complexity, the RKHS norm, which controls both stability and generalization of any learned model. In addition to models in the constructed RKHS, our stability analysis also applies to convolutional networks with generic activations such as rectified linear units, and we discuss its relationship with recent generalization bounds based on spectral norms.

Keywords: invariant representations, deep learning, stability, kernel methods

1. Introduction

The results achieved by deep neural networks for prediction tasks have been impressive in domains where data is structured and available in large amounts. In particular, convolutional neural networks (CNNs, LeCun et al., 1989) have shown to effectively leverage the local stationarity of natural images at multiple scales thanks to convolutional operations, while also providing some translation invariance through pooling operations. Yet, the exact nature of this invariance and the characteristics of functional spaces where convolutional neural networks live are poorly understood; overall, these models are sometimes seen as clever engineering black boxes that have been designed with a lot of insight collected since they were introduced.

Understanding the inductive bias of these models is nevertheless a fundamental question. For instance, a better grasp of the geometry induced by convolutional representations may bring new intuition about their success, and lead to improved measures of model complexity. In turn, the issue of regularization may be solved by providing ways to control the variations of prediction functions in a principled manner. One meaningful way to study such variations is to consider the stability of model predictions to naturally occuring changes of input signals, such as translations and deformations.

Small deformations of natural signals often preserve their main characteristics, such as class labels (*e.g.*, the same digit with different handwritings may correspond to the same images up to small deformations), and provide a much richer class of transformations than translations. The scattering transform (Mallat, 2012; Bruna and Mallat, 2013) is a recent attempt to characterize convolutional multilayer architectures based on wavelets. The theory provides an elegant characterization of invariance and stability properties of signals represented via the scattering operator, through a notion of Lipschitz stability to the action of diffeomorphisms. Nevertheless, these networks do not involve "learning" in the classical sense since the filters of the networks are pre-defined, and the resulting architecture differs significantly from the most used ones, which adapt filters to training data.

In this work, we study these theoretical properties for more standard convolutional architectures, from the point of view of positive definite kernels (Schölkopf and Smola, 2001). Specifically, we consider a functional space derived from a kernel for multi-dimensional signals that admits a multi-layer and convolutional structure based on the construction of convolutional kernel networks (CKNs) introduced by Mairal (2016); Mairal et al. (2014). The kernel representation follows standard convolutional architectures, with patch extraction, non-linear (kernel) mappings, and pooling operations. We show that our functional space contains a large class of CNNs with smooth homogeneous activation functions.

The main motivation for introducing a kernel framework is to study separately data representation and predictive models. On the one hand, we study the translation-invariance properties of the kernel representation and its stability to the action of diffeomorphisms, obtaining similar guarantees as the scattering transform (Mallat, 2012), while preserving signal information. When the kernel is appropriately designed, we also show how to obtain signal representations that are invariant to the action of any locally compact group of transformations, by modifying the construction of the kernel representation to become *equivariant* to the group action. On the other hand, we show that these stability results can be translated to predictive models by controlling their norm in the functional space, or simply the norm of the last layer in the case of CKNs (Mairal, 2016). With our kernel framework, the RKHS norm also acts as a measure of model complexity, thus controlling both stability and generalization, so that stability may lead to improved sample complexity. Finally, our work suggests that explicitly regularizing CNNs with the RKHS norm (or approximations thereof) can help obtain more stable models, a more practical question which we study in follow-up work (Bietti et al., 2019).

A short version of this paper was published at the Neural Information Processing Systems 2017 conference (Bietti and Mairal, 2017).

1.1. Summary of Main Results

Our work characterizes properties of deep convolutional models along two main directions.

- The first goal is to study *representation* properties of such models, independently of training data. Given a deep convolutional architecture, we study signal preservation as well as invariance and stability properties.

∗. Institute of Engineering Univ. Grenoble Alpes

©2019 Alberto Bietti and Julien Mairal.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-190.html.

- The second goal focuses on *learning* aspects, by studying the complexity of learned models based on our representation. In particular, our construction relies on kernel methods, allowing us to define a corresponding functional space (the RKHS). We show that this functional space contains a class of CNNs with smooth homogeneous activations, and study the complexity of such models by considering their RKHS norm. This directly leads to statements on the generalization of such models, as well as on the invariance and stability properties of their predictions.

- Finally, we show how some of our arguments extend to more traditional CNNs with generic and possibly non-smooth activations (such as ReLU or tanh).

Signal preservation, invariance and stability. We tackle this first goal by defining a deep convolutional representation based on hierarchical kernels. We show that the representation preserves signal information and guarantees near-invariance to translations and stability to deformations in the following sense, defined by Mallat (2012): for signals $x : \mathbb{R}^d \to \mathbb{R}^{p_0}$ defined on the continuous domain $\mathbb{R}^d$, we say that a representation $\Phi(x)$ is *stable* to the action of diffeomorphisms if

$$\|\Phi(L_\tau x) - \Phi(x)\| \leq (C_1\|\nabla\tau\|_\infty + C_2\|\tau\|_\infty)\|x\|,$$

where $\tau : \mathbb{R}^d \to \mathbb{R}^d$ is a C^1-diffeomorphism, $L_\tau x(u) = x(u-\tau(u))$ its action operator, and the norms $\|\tau\|_\infty$ and $\|\nabla\tau\|_\infty$ characterize how large the translation and deformation components are, respectively (see Section 3 for formal definitions). The Jacobian $\nabla\tau$ quantifies the size of local deformations, so that the first term controls the stability of the representation. In the case of translations, the first term vanishes ($\nabla\tau = 0$), hence a small value of C_2 is desirable for translation invariance. We show that such signal preservation and stability properties are valid for the multilayer kernel representation Φ defined in Section 2 by repeated application of patch extraction, kernel mapping, and pooling operators:

- The representation can be discretized with no loss of information, by subsampling at each layer with a factor smaller than the patch size;

- The translation invariance is controlled by a factor $C_2 = C_2'/\sigma_n$, where σ_n represents the "resolution" of the last layer, and typically increases exponentially with depth;

- The deformation stability is controlled by a factor C_1 which increases as κ^{d+1}, where κ corresponds to the patch size at a given layer, that is, the size of the "receptive field" of a patch relative to the resolution of the previous layer.

These results suggest that a good way to obtain a stable representation that preserves signal information is to use the smallest possible patches at each layer (*e.g.*, 3x3 for images) and perform pooling and downsampling at a factor smaller than the patch size, with as many layers as needed in order to reach a desired level of translation invariance σ_n. We show in Section 3.3 that the same invariance and stability guarantees hold when using kernel approximations as in CKNs, at the cost of losing signal information.

In Section 3.5, we show how to go beyond the translation group, by constructing similar representations that are invariant to the action of locally compact groups. This is achieved by modifying patch extraction and pooling operators so that they commute with the group action operator (this is known as *equivariance*).

Model complexity. Our second goal is to analyze the complexity of deep convolutional models by studying the functional space defined by our kernel representation, showing that certain classes of CNNs are contained in this space, and characterizing their norm.

The multi-layer kernel representation defined in Section 2 is constructed by using kernel mappings defined on local signal patches at each scale, which replace the linear mapping followed by a non-linearity in standard convolutional networks. Inspired by Zhang et al. (2017b), we show in Section 4.1 that when these kernel mappings come from a class of dot-product kernels, the corresponding RKHS contains functions of the form

$$z \mapsto \|z\|\sigma(\langle g, z\rangle/\|z\|),$$

for certain types of smooth activation functions σ, where g and z live in a particular Hilbert space. These behave like simple neural network functions on patches, up to homogeneization. Note that if σ was allowed to be homogeneous, such as for rectified linear units $\sigma(\alpha) = \max(\alpha, 0)$, homogeneization would disappear. By considering multiple such functions at each layer, we construct a CNN in the RKHS of the full multi-layer kernel in Section 4.2. Denoting such a CNN by f_σ, we show that its RKHS norm can be bounded as

$$\|f_\sigma\|^2 \leq \|w_{n+1}\|^2\, C_\sigma^2(\|W_n\|_2^2\, C_\sigma^2(\|W_{n-1}\|_2^2 \ldots C_\sigma^2(\|W_2\|_2^2\, C_\sigma^2(\|W_1\|_F^2))\ldots)),$$

where W_k are convolutional filter parameters at layer k, w_{n+1} carries the parameters of a final linear fully connected layer, C_σ^2 is a function quantifying the complexity of the simple functions defined above depending on the choice of activation σ, and $\|W_k\|_2$, $\|W_k\|_F$ denote spectral and Frobenius norms, respectively, (see Section 4.2 for details). This norm can then control generalization aspects through classical margin bounds, as well as the invariance and stability of model predictions. Indeed, by using the reproducing property $f(x) = \langle f, \Phi(x)\rangle$, this "linearization" lets us control stability properties of model predictions through $\|f\|$:

$$\text{for all signals } x \text{ and } x', \quad |f(x) - f(x')| \leq \|f\| \cdot \|\Phi(x) - \Phi(x')\|,$$

meaning that the prediction function f will inherit the stability of Φ when $\|f\|$ is small.

The case of standard CNNs with generic activations. When considering CNNs with generic, possibly non-smooth activations such as rectified linear units (ReLUs), the separation between a data-independent representation and a learned model is not always achievable in contrast to our kernel approach. In particular, the "representation" given by the last layer of a learned CNN is often considered by practitioners, but such a representation is data-dependent in that it is typically trained on a specific task and dataset, and does not preserve signal information.

Nevertheless, we obtain similar invariance and stability properties for the predictions of such models in Section 4.3, by considering a complexity measure given by the product of spectral norms of each linear convolutional mapping in a CNN. Unlike our study based on kernel methods, such results do not say anything about generalization; however, relevant generalization bounds based on similar quantities have been derived (though other quantities in addition to the product of spectral norms appear in the bounds, and these bounds do not directly apply to CNNs), *e.g.*, by Bartlett et al. (2017); Neyshabur et al. (2018), making the relationship between generalization and stability clear in this context as well.

1.2. Related Work

Our work relies on image representations introduced in the context of convolutional kernel networks (Mairal, 2016; Mairal et al., 2014), which yield a sequence of spatial maps similar to traditional CNNs, but where each point on the maps is possibly infinite-dimensional and lives in a reproducing kernel Hilbert space (RKHS). The extension to signals with d spatial dimensions is straightforward. Since computing the corresponding Gram matrix as in classical kernel machines is computationally impractical, CKNs provide an approximation scheme consisting of learning finite-dimensional subspaces of each RKHS's layer, where the data is projected. The resulting architecture of CKNs resembles traditional CNNs with a subspace learning interpretation and different unsupervised learning principles.

Another major source of inspiration is the study of group-invariance and stability to the action of diffeomorphisms of scattering networks (Mallat, 2012), which introduced the main formalism and several proof techniques that were keys to our results. Our main effort was to extend them to more general CNN architectures and to the kernel framework, allowing us to provide a clear relationship between stability properties of the representation and generalization of learned CNN models. We note that an extension of scattering networks results to more general convolutional networks was previously given by Wiatowski and Bölcskei (2018); however, their guarantees on deformations do not improve on the inherent stability properties of the considered signal, and their study does not consider learning or generalization, by treating a convolutional architecture with fixed weights as a feature extractor. In contrast, our stability analysis shows the benefits of deep representations with a clear dependence on the choice of network architecture through the size of convolutional patches and pooling layers, and we study the implications for learned CNNs through notions of model complexity.

Invariance to groups of transformations was also studied for more classical convolutional neural networks from methodological and empirical points of view (Bruna et al., 2013; Cohen and Welling, 2016), and for shallow learned representations (Anselmi et al., 2016) or kernel methods (Haasdonk and Burkhardt, 2007; Mroueh et al., 2015; Raj et al., 2017). Our work provides a similar group-equivariant construction to (Cohen and Welling, 2016), while additionally relating it to stability. In particular, we show that in order to achieve group invariance, pooling on the group is only needed at the final layer, while deep architectures with pooling at multiple scales are mainly beneficial for stability. For the specific example of the roto-translation group (Sifre and Mallat, 2013), we show that our construction achieves invariance to rotations while maintaining stability to deformations on the translation group.

Note also that other techniques combining deep neural networks and kernels have been introduced earlier. Multilayer kernel machines were for instance introduced by Cho and Saul (2009); Schölkopf et al. (1998). Shallow kernels for images modeling local regions were also proposed by Schölkopf (1997), and a multilayer construction was proposed by Bo et al. (2011). More recently, different models based on kernels have been introduced by Anselmi et al. (2015); Daniely et al. (2016); Montavon et al. (2011) to gain some theoretical insight about classical multilayer neural networks, while kernels are used by Zhang et al. (2017b) to define convex models for two-layer convolutional networks. Theoretical and practical concerns for learning with multilayer kernels have been studied in Daniely et al. (2017, 2016); Steinwart et al. (2016); Zhang et al. (2016) in addition to CKNs. In particular,

Daniely et al. (2017, 2016) study certain classes of dot-product kernels with random feature approximations, Steinwart et al. (2016) consider hierarchical Gaussian kernels with learned weights, and Zhang et al. (2016) study a convex formulation for learning a certain class of fully connected neural networks using a hierarchical kernel. In contrast to these works, our focus is on the kernel *representation* induced by the specific hierarchical kernel defined in CKNs and the geometry of the RKHS. Our characterization of CNNs and activation functions contained in the RKHS is similar to the work of Zhang et al. (2016, 2017b), but differs in several ways: we consider general *homogeneous* dot-product kernels, which yield desirable properties of kernel mappings for stability; we construct generic multi-layer CNNs with pooling in the RKHS, while Zhang et al. (2016) only considers fully-connected networks and Zhang et al. (2017b) is limited to two-layer convolutional networks with no pooling; we quantify the RKHS norm of a CNN depending on its parameters, in particular matrix norms, as a way to control stability and generalization, while Zhang et al. (2016, 2017b) consider models with constrained parameters, and focus on convex learning procedures.

1.3. Notation and Basic Mathematical Tools

A positive definite kernel K that operates on a set $\mathcal{X}$ implicitly defines a reproducing kernel Hilbert space $\mathcal{H}$ of functions from $\mathcal{X}$ to $\mathbb{R}$, along with a mapping $\varphi : \mathcal{X} \to \mathcal{H}$. A *predictive model* associates to every point z in $\mathcal{X}$ a label in $\mathbb{R}$; it consists of a linear function f in $\mathcal{H}$ such that $f(z) = \langle f, \varphi(z) \rangle_{\mathcal{H}}$, where $\varphi(z)$ is the *data representation*. Given now two points z, z' in $\mathcal{X}$, Cauchy-Schwarz's inequality allows us to control the variation of the predictive model f according to the geometry induced by the Hilbert norm $\|.\|_{\mathcal{H}}$:

$$|f(z) - f(z')| \leq \|f\|_{\mathcal{H}} \|\varphi(z) - \varphi(z')\|_{\mathcal{H}}. \tag{1}$$

This property implies that two points z and z' that are close to each other according to the RKHS norm should lead to similar predictions, when the model f has small norm in $\mathcal{H}$.

Then, we consider notation from signal processing similar to Mallat (2012). We call a signal x a function in $L^2(\mathbb{R}^d, \mathcal{H})$, where the domain $\mathbb{R}^d$ represents spatial coordinates, and $\mathcal{H}$ is a Hilbert space, when $\|x\|_{L^2}^2 := \int_{\mathbb{R}^d} \|x(u)\|_{\mathcal{H}}^2 du < \infty$, where du is the Lebesgue measure on $\mathbb{R}^d$. Given a linear operator $T : L^2(\mathbb{R}^d, \mathcal{H}) \to L^2(\mathbb{R}^d, \mathcal{H}')$, the operator norm is defined as $\|T\|_{L^2(\mathbb{R}^d,\mathcal{H}) \to L^2(\mathbb{R}^d,\mathcal{H}')} := \sup_{\|x\|_{L^2(\mathbb{R}^d,\mathcal{H})} \leq 1} \|Tx\|_{L^2(\mathbb{R}^d,\mathcal{H}')}$. For the sake of clarity, we drop norm subscripts, from now on, using the notation $\|\cdot\|$ for Hilbert space norms, L^2 norms, and $L^2 \to L^2$ operator norms, while $|\cdot|$ denotes the Euclidean norm on $\mathbb{R}^d$. We use cursive capital letters (*e.g.*, $\mathcal{H}, \mathcal{P}$) to denote Hilbert spaces, and non-cursive ones for operators (*e.g.*, P, M, A). Some useful mathematical tools are also presented in Appendix A.

1.4. Organization of the Paper

The rest of the paper is structured as follows:

- In Section 2, we introduce a multilayer convolutional kernel representation for continuous signals, based on a hierarchy of patch extraction, kernel mapping, and pooling operators. We present useful properties of this representation such as signal preservation, as well as ways to make it practical through discretization and kernel approximations in the context of CKNs.

- In Section 3, we present our main results regarding stability and invariance, namely that the kernel representation introduced in Section 2 is near translation-invariant and stable to the action of diffeomorphisms. We then show in Section 3.3 that the same stability results apply in the presence of kernel approximations such as those of CKNs (Mairal, 2016), and describe a generic way to modify the multilayer construction in order to guarantee invariance to the action of any locally compact group of transformations in Section 3.5.
- In Section 4, we study the functional spaces induced by our representation, showing that simple neural-network like functions with certain smooth activations are contained in the RKHS at intermediate layers, and that the RKHS of the full kernel induced by our representation contains a class of generic CNNs with smooth and homogeneous activations. We then present upper bounds on the RKHS norm of such CNNs, which serves as a measure of complexity, controlling both generalization and stability. Section 4.3 studies the stability for CNNs with generic activations such as rectified linear units, and discusses the link with generalization.
- Finally, we discuss in Section 5 how the obtained stability results apply to the practical setting of learning prediction functions. In particular, we explain why the regularization used in CKNs provides a natural way to control stability, while a similar control is harder to achieve with generic CNNs.

2. Construction of the Multilayer Convolutional Kernel

We now present the multilayer convolutional kernel, which operates on signals with d spatial dimensions. The construction follows closely that of convolutional kernel networks but is generalized to input signals defined on the continuous domain $\mathbb{R}^d$. Dealing with continuous signals is indeed useful to characterize the stability properties of signal representations to small deformations, as done by Mallat (2012) in the context of the scattering transform. The issue of discretization on a discrete grid is addressed in Section 2.1.

In what follows, we consider signals x_0 that live in $L^2(\mathbb{R}^d, \mathcal{H}_0)$, where typically $\mathcal{H}_0 = \mathbb{R}^{p_0}$ (*e.g.*, with $p_0 = 3$ and $d = 2$, the vector $x_0(u)$ in $\mathbb{R}^3$ may represent the RGB pixel value at location u in $\mathbb{R}^2$). Then, we build a sequence of reproducing kernel Hilbert spaces $\mathcal{H}_1, \mathcal{H}_2, \ldots$, and transform x_0 into a sequence of "feature maps", respectively denoted by x_1 in $L^2(\mathbb{R}^d, \mathcal{H}_1)$, x_2 in $L^2(\mathbb{R}^d, \mathcal{H}_2)$, *etc...* As depicted in Figure 1, a new map x_k is built from the previous one x_{k-1} by applying successively three operators that perform patch extraction (P_k), kernel mapping (M_k) to a new RKHS $\mathcal{H}_k$, and linear pooling (A_k), respectively. When going up in the hierarchy, the points $x_k(u)$ carry information from larger signal neighborhoods centered at u in $\mathbb{R}^d$ with more invariance, as we formally show in Section 3.

Patch extraction operator. Given the layer x_{k-1}, we consider a patch shape S_k, defined as a compact centered subset of $\mathbb{R}^d$, *e.g.*, a box, and we define the Hilbert space $\mathcal{P}_k := L^2(S_k, \mathcal{H}_{k-1})$ equipped with the norm $\|z\|^2 = \int_{S_k} \|z(u)\|^2 d\nu_k(u)$, where $d\nu_k$ is the normalized uniform measure on S_k for every z in $\mathcal{P}_k$. Specifically, we define the (linear) patch extraction operator $P_k : L^2(\mathbb{R}^d, \mathcal{H}_{k-1}) \to L^2(\mathbb{R}^d, \mathcal{P}_k)$ such that for all u in $\mathbb{R}^d$,

$$P_k x_{k-1}(u) = (v \mapsto x_{k-1}(u+v))_{v \in S_k} \in \mathcal{P}_k.$$

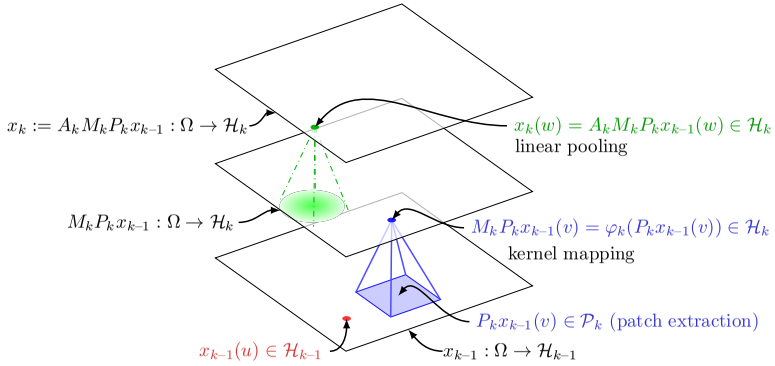


Figure 1: Construction of the k-th signal representation from the k–1-th one. Note that while the domain Ω is depicted as a box in $\mathbb{R}^2$ here, our construction is supported on $\Omega = \mathbb{R}^d$.

Note that by equipping $\mathcal{P}_k$ with a normalized measure, it is easy to show that the operator P_k preserves the norm—that is, $\|P_k x_{k-1}\| = \|x_{k-1}\|$ and hence $P_k x_{k-1}$ is in $L^2(\mathbb{R}^d, \mathcal{P}_k)$.

Kernel mapping operator. Then, we map each patch of x_{k-1} to a RKHS $\mathcal{H}_k$ thanks to the kernel mapping $\varphi_k : \mathcal{P}_k \to \mathcal{H}_k$ associated to a positive definite kernel K_k that operates on patches. It allows us to define the pointwise operator M_k such that for all u in $\mathbb{R}^d$,

$$M_k P_k x_{k-1}(u) := \varphi_k(P_k x_{k-1}(u)) \in \mathcal{H}_k.$$

In this paper, we consider homogeneous dot-product kernels K_k operating on $\mathcal{P}_k$, defined in terms of a function $\kappa_k : [-1, 1] \to \mathbb{R}$ that satisfies the following constraints:

$$\kappa_k(u) = \sum_{j=0}^{+\infty} b_j u^j \quad \text{s.t.} \quad \forall j, b_j \geq 0, \quad \kappa_k(1) = 1, \quad \kappa_k'(1) = 1, \tag{A1}$$

assuming convergence of the series $\sum_j b_j$ and $\sum_j j b_j$. Then, we define the kernel K_k by

$$K_k(z, z') = \|z\| \|z'\| \kappa_k\left(\frac{\langle z, z' \rangle}{\|z\| \|z'\|}\right), \tag{2}$$

if $z, z' \in \mathcal{P}_k \setminus \{0\}$, and $K_k(z, z') = 0$ if $z = 0$ or $z' = 0$. The kernel is positive definite since it admits a Maclaurin expansion with only non-negative coefficients (Schoenberg, 1942; Schölkopf and Smola, 2001). The condition $\kappa_k(1) = 1$ ensures that the RKHS mapping preserves the norm—that is, $\|\varphi_k(z)\| = K_k(z, z)^{1/2} = \|z\|$, and thus $\|M_k P_k x_{k-1}(u)\| = \|P_k x_{k-1}(u)\|$ for all u in $\mathbb{R}^d$; as a consequence, $M_k P_k x_{k-1}$ is always in $L^2(\mathbb{R}^d, \mathcal{H}_k)$. The technical condition $\kappa_k'(1) = 1$, where κ_k' is the first derivative of κ_k, ensures that the kernel mapping φ_k is non-expansive, according to Lemma 1 below.

Lemma 1 (Non-expansiveness of the kernel mappings) *Consider a positive-definite kernel of the form* (2) *satisfying* (A1) *with RKHS mapping* $\varphi_k : \mathcal{P}_k \to \mathcal{H}_k$. *Then,* φ_k *is non-expansive—that is, for all* z, z' *in* $\mathcal{P}_k$,

$$\|\varphi_k(z) - \varphi_k(z')\| \leq \|z - z'\|.$$

Moreover, we remark that the kernel K_k *is lower-bounded by the linear one*

$$K_k(z, z') \geq \langle z, z' \rangle. \tag{3}$$

From the proof of the lemma, given in Appendix B, one may notice that the assumption $\kappa_k'(1) = 1$ is not critical and may be safely replaced by $\kappa_k'(1) \leq 1$. Then, the non-expansiveness property would be preserved. Yet, we have chosen a stronger constraint since it yields a few simplifications in the stability analysis, where we use the relation (3) that requires $\kappa_k'(1) = 1$. More generally, the kernel mapping is Lipschitz continuous with constant $\rho_k = \max(1, \sqrt{\kappa_k'(1)})$. Our stability results hold in a setting with $\rho_k > 1$, but with constants $\prod_k \rho_k$ that may grow exponentially with the number of layers.

Examples of functions κ_k that satisfy the properties (A1) are now given below:

exponential	$\kappa_{\exp}(\langle z, z' \rangle) = e^{\langle z, z' \rangle - 1}$
inverse polynomial	$\kappa_{\text{inv-poly}}(\langle z, z' \rangle) = \frac{1}{2 - \langle z, z' \rangle}$
polynomial, degree p	$\kappa_{\text{poly}}(\langle z, z' \rangle) = \frac{1}{(c+1)^p}(c + \langle z, z' \rangle)^p \quad$ with $\quad c = p - 1$
arc-cosine, degree 1	$\kappa_{\text{acos}}(\langle z, z' \rangle) = \frac{1}{\pi}\left(\sin(\theta) + (\pi - \theta)\cos(\theta)\right)$ with $\theta = \arccos(\langle z, z' \rangle)$
Vovk's, degree 3	$\kappa_{\text{vovk}}(\langle z, z' \rangle) = \frac{1}{3}\left(\frac{1 - \langle z, z' \rangle^3}{1 - \langle z, z' \rangle}\right) = \frac{1}{3}\left(1 + \langle z, z' \rangle + \langle z, z' \rangle^2\right)$

We note that the inverse polynomial kernel was used by Zhang et al. (2016, 2017b) to build convex models of fully connected networks and two-layer convolutional neural networks, while the arc-cosine kernel appears in early deep kernel machines (Cho and Saul, 2009). Note that the homogeneous exponential kernel reduces to the Gaussian kernel for unit-norm vectors. Indeed, for all z, z' such that $\|z\| = \|z'\| = 1$, we have

$$\kappa_{\exp}(\langle z, z' \rangle) = e^{\langle z, z' \rangle - 1} = e^{-\frac{1}{2}\|z - z'\|^2},$$

and thus, we may refer to kernel (2) with the function $\kappa_{\exp}$ as the homogeneous Gaussian kernel. The kernel $\kappa(\langle z, z' \rangle) = e^{\alpha(\langle z, z' \rangle - 1)} = e^{-\frac{\alpha}{2}\|z - z'\|^2}$ with $\alpha \neq 1$ may also be used here, but we choose $\alpha = 1$ for simplicity since $\kappa'(1) = \alpha$ (see discussion above).

Pooling operator. The last step to build the layer x_k consists of pooling neighboring values to achieve local shift-invariance. We apply a linear convolution operator A_k with a Gaussian filter of scale σ_k, $h_{\sigma_k}(u) := \sigma_k^{-d} h(u/\sigma_k)$, where $h(u) = (2\pi)^{-d/2}\exp(-|u|^2/2)$. Then, for all u in $\mathbb{R}^d$,

$$x_k(u) = A_k M_k P_k x_{k-1}(u) = \int_{\mathbb{R}^d} h_{\sigma_k}(u - v) M_k P_k x_{k-1}(v) dv \in \mathcal{H}_k, \tag{4}$$

where the integral is a Bochner integral (see, Diestel and Uhl, 1977; Muandet et al., 2017). By applying Schur's test to the integral operator A_k (see Appendix A), we obtain that the operator norm $\|A_k\|$ is less than 1. Thus, x_k is in $L^2(\mathbb{R}^d, \mathcal{H}_k)$, with $\|x_k\| \leq \|M_k P_k x_{k-1}\|$. Note that a similar pooling operator is used in the scattering transform (Mallat, 2012).

Multilayer construction and prediction layer. Finally, we obtain a multilayer representation by composing multiple times the previous operators. In order to increase invariance with each layer and to increase the size of the receptive fields (that is, the neighborhood of the original signal considered in a given patch), the size of the patch S_k and pooling scale σ_k typically grow exponentially with k, with σ_k and the patch size $\sup_{c \in S_k} |c|$ of the same order. With n layers, the maps x_n may then be written

$$x_n := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 x_0 \ \in \ L^2(\mathbb{R}^d, \mathcal{H}_n). \tag{5}$$

It remains to define a kernel from this representation, that will play the same role as the "fully connected" layer of classical convolutional neural networks. For that purpose, we simply consider the following linear kernel defined for all x_0, x_0' in $L^2(\mathbb{R}^d, \mathcal{H}_0)$ by using the corresponding feature maps x_n, x_n' in $L^2(\mathbb{R}^d, \mathcal{H}_n)$ given by our multilayer construction (5):

$$\mathcal{K}_n(x_0, x_0') = \langle x_n, x_n' \rangle = \int_{u \in \mathbb{R}^d} \langle x_n(u), x_n'(u) \rangle du. \tag{6}$$

Then, the RKHS $\mathcal{H}_{\mathcal{K}_n}$ of $\mathcal{K}_n$ contains all functions of the form $f(x_0) = \langle w, x_n \rangle$ with w in $L^2(\mathbb{R}^d, \mathcal{H}_n)$ (see Appendix A).

We note that one may also consider nonlinear kernels, such as a Gaussian kernel:

$$\mathcal{K}_n(x_0, x_0') = e^{-\frac{\alpha}{2}\|x_n - x_n'\|^2}. \tag{7}$$

Such kernels are then associated to a RKHS denoted by $\mathcal{H}_{n+1}$, along with a kernel mapping $\varphi_{n+1} : L^2(\mathbb{R}^d, \mathcal{H}_n) \to \mathcal{H}_{n+1}$ which we call *prediction layer*, so that the final representation is given by $\varphi_{n+1}(x_n)$ in $\mathcal{H}_{n+1}$. We note that φ_{n+1} is non-expansive for the Gaussian kernel when $\alpha \leq 1$ (see Section B.1), and is simply an isometric linear mapping for the linear kernel. Then, we have the relation $\mathcal{K}_n(x_0, x_0') := \langle \varphi_{n+1}(x_n), \varphi_{n+1}(x_n') \rangle$, and in particular, the RKHS $\mathcal{H}_{\mathcal{K}_n}$ of $\mathcal{K}_n$ contains all functions of the form $f(x_0) = \langle w, \varphi_{n+1}(x_n) \rangle$ with w in $\mathcal{H}_{n+1}$, see Appendix A.

2.1. Signal Preservation and Discretization

In this section, we show that the multilayer kernel representation preserves all information about the signal at each layer, and besides, each feature map x_k can be sampled on a discrete set with no loss of information. This suggests a natural approach for discretization which will be discussed after the following lemma, whose proof is given in Appendix C.

Lemma 2 (Signal recovery from sampling) *Assume that* $\mathcal{H}_k$ *contains all linear functions* $z \mapsto \langle g, z \rangle$ *with* g *in* $\mathcal{P}_k$ *(this is true for all kernels* K_k *described in the previous section, according to Corollary 12 in Section 4.1 later); then, the signal* x_{k-1} *can be recovered from a sampling of* x_k *at discrete locations in a set* Ω *as soon as* $\Omega + S_k = \mathbb{R}^d$ (i.e., *the union of patches centered at these points covers* $\mathbb{R}^d$*). It follows that* x_k *can be reconstructed from such a sampling.*

The previous construction defines a kernel representation for general signals in $L^2(\mathbb{R}^d, \mathcal{H}_0)$, which is an abstract object defined for theoretical purposes. In practice, signals are discrete, and it is thus important to discuss the problem of discretization. For clarity, we limit the

presentation to 1-dimensional signals ($d = 1$), but the arguments can easily be extended to higher dimensions d when using box-shaped patches. Notation from the previous section is preserved, but we add a bar on top of all discrete analogues of their continuous counterparts. *e.g.*, $\bar{x}_k$ is a discrete feature map in $\ell^2(\mathbb{Z}, \bar{\mathcal{H}}_k)$ for some RKHS $\bar{\mathcal{H}}_k$.

Input signals x_0 and $\bar{x}_0$. Discrete signals acquired by a physical device may be seen as local integrators of signals defined on a continuous domain (*e.g.*, sensors from digital cameras integrate the pointwise distribution of photons in a spatial and temporal window). Then, consider a signal x_0 in $L^2(\mathbb{R}^d, \mathcal{H}_0)$ and s_0 a sampling interval. By defining $\bar{x}_0$ in $\ell_2(\mathbb{Z}, \mathcal{H}_0)$ such that $\bar{x}_0[n] = x_0(ns_0)$ for all n in $\mathbb{Z}$, it is thus natural to assume that $x_0 = A_0 x$, where A_0 is a pooling operator (local integrator) applied to an original continuous signal x. The role of A_0 is to prevent aliasing and reduce high frequencies; typically, the scale σ_0 of A_0 should be of the same magnitude as s_0, which we choose to be $s_0 = 1$ without loss of generality. This natural assumption is kept later for the stability analysis.

Multilayer construction. We now want to build discrete feature maps $\bar{x}_k$ in $\ell^2(\mathbb{Z}, \bar{\mathcal{H}}_k)$ at each layer k involving subsampling with a factor s_k with respect to $\bar{x}_{k-1}$. We now define the discrete analogues of the operators P_k (patch extraction), M_k (kernel mapping), and A_k (pooling) as follows: for $n \in \mathbb{Z}$,

$$\bar{P}_k \bar{x}_{k-1}[n] := \frac{1}{\sqrt{e_k}}(\bar{x}_{k-1}[n], \bar{x}_{k-1}[n+1], \dots, \bar{x}_{k-1}[n+e_k-1]) \in \bar{\mathcal{P}}_k := \bar{\mathcal{H}}_{k-1}^{e_k}$$
$$\bar{M}_k \bar{P}_k \bar{x}_{k-1}[n] := \bar{\varphi}_k(\bar{P}_k \bar{x}_{k-1}[n]) \in \bar{\mathcal{H}}_k$$
$$\bar{x}_k[n] = \bar{A}_k \bar{M}_k \bar{P}_k \bar{x}_{k-1}[n] := \frac{1}{\sqrt{s_k}} \sum_{m \in \mathbb{Z}} \bar{h}_k[ns_k - m] \bar{M}_k \bar{P}_k \bar{x}_{k-1}[m] = (\bar{h}_k * \bar{M}_k \bar{P}_k \bar{x}_{k-1})[ns_k] \in \bar{\mathcal{H}}_k,$$

where (i) $\bar{P}_k$ extracts a patch of size e_k starting at position n in $\bar{x}_{k-1}[n]$, which lives in the Hilbert space $\bar{\mathcal{P}}_k$ defined as the direct sum of e_k times $\bar{\mathcal{H}}_{k-1}$; (ii) $\bar{M}_k$ is a kernel mapping identical to the continuous case, which preserves the norm, like M_k; (iii) $\bar{A}_k$ performs a convolution with a Gaussian filter and a subsampling operation with factor s_k. The next lemma shows that under mild assumptions, this construction preserves signal information.

Lemma 3 (Signal recovery with subsampling) *Assume that $\bar{\mathcal{H}}_k$ contains the linear functions $z \mapsto \langle w, z \rangle$ for all w in $\bar{\mathcal{P}}_k$ and that $e_k \geq s_k$. Then, $\bar{x}_{k-1}$ can be recovered from $\bar{x}_k$.*

The proof is given in Appendix C. The result relies on recovering patches using linear "measurement" functions and deconvolution of the pooling operation. While such a deconvolution operation can be unstable, it may be possible to obtain more stable recovery mechanisms by also considering non-linear measurements, a question which we leave open.

Links between the parameters of the discrete and continuous models. Due to subsampling, the patch size in the continuous and discrete models are related by a multiplicative factor. Specifically, a patch of size e_k with discretization corresponds to a patch S_k of diameter $e_k s_{k-1} s_{k-2} \dots s_1$ in the continuous case. The same holds true for the scale parameter σ_k of the Gaussian pooling.

2.2. Practical Implementation via Convolutional Kernel Networks

Besides discretization, convolutional kernel networks add two modifications to implement in practice the image representation we have described. First, it uses feature maps with finite spatial support, which introduces border effects that we do not study (like Mallat, 2012), but which are negligible when dealing with large realistic images. Second, CKNs use finite-dimensional approximations of the kernel feature map. Typically, each RKHS's mapping is approximated by performing a projection onto a subspace of finite dimension, which is a classical approach to make kernel methods work at large scale (Fine and Scheinberg, 2001; Smola and Schölkopf, 2000; Williams and Seeger, 2001). If we consider the kernel mapping $\varphi_k : \mathcal{P}_k \to \mathcal{H}_k$ at layer k, the orthogonal projection onto the finite-dimensional subspace $\mathcal{F}_k = \mathrm{span}(\varphi_k(z_1), \dots, \varphi_k(z_{p_k})) \subseteq \mathcal{H}_k$, where the z_i's are p_k anchor points in $\mathcal{P}_k$, is given by the linear operator $\Pi_k : \mathcal{H}_k \to \mathcal{F}_k$ defined for f in $\mathcal{H}_k$ by

$$\Pi_k f := \sum_{1 \leq i,j \leq p_k} (K_{ZZ}^{-1})_{ij} \langle \varphi_k(z_i), f \rangle \varphi_k(z_j), \tag{8}$$

where K_{ZZ}^{-1} is the inverse (or pseudo-inverse) of the $p_k \times p_k$ kernel matrix $[K_k(z_i, z_j)]_{ij}$. As an orthogonal projection operator, Π_k is non-expansive, *i.e.*, $\|\Pi_k\| \leq 1$. We can then define the new approximate version $\tilde{M}_k$ of the kernel mapping operator M_k by

$$\tilde{M}_k P_k x_{k-1}(u) := \Pi_k \varphi_k(P_k x_{k-1}(u)) \in \mathcal{F}_k. \tag{9}$$

Note that all points in the feature map $\tilde{M}_k P_k x_{k-1}$ lie in the p_k-dimensional space $\mathcal{F}_k \subseteq \mathcal{H}_k$, which allows us to represent each point $\tilde{M}_k P_k x_{k-1}(u)$ by the finite dimensional vector

$$\psi_k(P_k x_{k-1}(u)) := K_{ZZ}^{-1/2} K_Z(P_k x_{k-1}(u)) \in \mathbb{R}^{p_k}, \tag{10}$$

with $K_Z(z) := (K_k(z_1, z), \dots, K_k(z_{p_k}, z))^\top$; this finite-dimensional representation preserves the Hilbertian inner product and norm[1] in $\mathcal{F}_k$ so that $\|\psi_k(P_k x_{k-1}(u))\|_2^2 = \|\tilde{M}_k P_k x_{k-1}(u)\|_{\mathcal{H}_k}^2$.

Such a finite-dimensional mapping is compatible with the multilayer construction, which builds $\mathcal{H}_k$ by manipulating points from $\mathcal{H}_{k-1}$. Here, the approximation provides points in $\mathcal{F}_k \subseteq \mathcal{H}_k$, which remain in $\mathcal{F}_k$ after pooling since $\mathcal{F}_k$ is a linear subspace. Eventually, the sequence of RKHSs $\{\mathcal{H}_k\}_{k \geq 0}$ is not affected by the finite-dimensional approximation. Besides, the stability results we will present next are preserved thanks to the non-expansiveness of the projection. In contrast, other kernel approximations such as random Fourier features (Rahimi and Recht, 2007) do not provide points in the RKHS (see Bach, 2017), and their effect on the functional space derived from the multilayer construction is unclear.

It is then possible to derive theoretical results for the CKN model, which appears as a natural implementation of the kernel constructed previously; yet, we will also show in Section 4 that the results apply more broadly to CNNs that are contained in the functional space associated to the kernel. However, the stability of these CNNs depends on their RKHS norm, which is hard to control. In contrast, for CKNs, stability is typically controlled by the norm of the final prediction layer.

1. We have $\langle \psi_k(z), \psi_k(z') \rangle_2 = \langle \Pi_k \varphi_k(z), \Pi_k \varphi_k(z') \rangle_{\mathcal{H}_k}$. See Mairal (2016) for details.

3. Stability to Deformations and Group Invariance

In this section, we study the translation invariance and the stability under the action of diffeomorphisms of the kernel representation described in Section 2 for continuous signals. In addition to translation invariance, it is desirable to have a representation that is stable to small local deformations. We describe such deformations using a C^1-diffeomorphism $\tau : \mathbb{R}^d \to \mathbb{R}^d$, and let L_τ denote the linear operator defined by $L_\tau x(u) = x(u - \tau(u))$. We use a similar characterization of stability to the one introduced by Mallat (2012): the representation $\Phi(\cdot)$ is *stable* under the action of diffeomorphisms if there exist two non-negative constants C_1 and C_2 such that

$$\|\Phi(L_\tau x) - \Phi(x)\| \leq (C_1 \|\nabla\tau\|_\infty + C_2 \|\tau\|_\infty)\|x\|, \tag{11}$$

where $\nabla\tau$ is the Jacobian of τ, $\|\nabla\tau\|_\infty := \sup_{u\in\mathbb{R}^d} \|\nabla\tau(u)\|$, and $\|\tau\|_\infty := \sup_{u\in\mathbb{R}^d} |\tau(u)|$. The quantity $\|\nabla\tau(u)\|$ measures the size of the deformation at a location u, and like Mallat (2012), we assume the regularity condition $\|\nabla\tau\|_\infty \leq 1/2$, which implies that the deformation is invertible (Allassonnière et al., 2007; Trouvé and Younes, 2005) and helps us avoid degenerate situations. In order to have a near-translation-invariant representation, we want C_2 to be small (a translation is a diffeomorphism with $\nabla\tau = 0$), and indeed we will show that C_2 is proportional to $1/\sigma_n$, where σ_n is the scale of the last pooling layer, which typically increases exponentially with the number of layers n. When $\nabla\tau$ is non-zero, the diffeomorphism deviates from a translation, producing local deformations controlled by $\nabla\tau$.

Additional assumptions. In order to study the stability of the representation (5), we assume that the input signal x_0 may be written as $x_0 = A_0 x$, where A_0 is an initial pooling operator at scale σ_0, which allows us to control the high frequencies of the signal in the first layer. As discussed previously in Section 2.1, this assumption is natural and compatible with any physical acquisition device. Note that σ_0 can be taken arbitrarily small, so that this assumption does not limit the generality of our results. Then, we are interested in understanding the stability of the representation

$$\Phi_n(x) := A_n M_n P_n A_{n-1} M_{n-1} P_{n-1} \cdots A_1 M_1 P_1 A_0 x.$$

We do not consider a prediction layer φ_{n+1} here for simplicity, but note that if we add one on top of Φ_n, based on a linear of Gaussian kernel, then the stability of the full representation $\varphi_{n+1} \circ \Phi_n$ immediately follows from that of Φ_n thanks to the non-expansiveness of φ_{n+1} (see Section 2). Then, we make an assumption that relates the scale of the pooling operator at layer $k-1$ with the diameter of the patch S_k: we assume indeed that there exists $\kappa > 0$ such that for all $k \geq 1$,

$$\sup_{c\in S_k} |c| \leq \kappa\sigma_{k-1}. \tag{A2}$$

The scales σ_k are typically exponentially increasing with the layers k, and characterize the "resolution" of each feature map. This assumption corresponds to considering patch sizes that are adapted to these intermediate resolutions. Moreover, the stability bounds we obtain hereafter increase with κ, which leads us to believe that small patch sizes lead to more stable representations, something which matches well the trend of using small, 3x3 convolution filters at each scale in modern deep architectures (*e.g.*, Simonyan and Zisserman, 2014).

Finally, before presenting our stability results, we recall a few properties of the operators involved in the representation Φ_n, which are heavily used in the analysis.

1. **Patch extraction operator**: P_k is linear and preserves the norm;
2. **Kernel mapping operator**: M_k preserves the norm and is non-expansive;
3. **Pooling operator**: A_k is linear and non-expansive $\|A_k\| \leq 1$;

The rest of this section is organized into three parts. We present the main stability results in Section 3.1, explain their compatibility with kernel approximations in Section 3.3, and provide numerical experiment for demonstrating the stability of the kernel representation in Section 3.4. Finally, we introduce mechanisms to achieve invariance to any group of transformations in Section 3.5.

3.1. Stability Results and Translation Invariance

Here, we show that our kernel representation Φ_n satisfies the stability property (11), with a constant C_2 inversely proportional to σ_n, thereby achieving near-invariance to translations. The results are then extended to more general transformation groups in Section 3.5.

General bound for stability. The following result gives an upper bound on the quantity of interest, $\|\Phi_n(L_\tau x) - \Phi_n(x)\|$, in terms of the norm of various linear operators which control how τ affects each layer. An important object of study is the commutator of linear operators A and B, which is denoted by $[A, B] = AB - BA$.

Proposition 4 (Bound with operator norms) *For any x in $L^2(\mathbb{R}^d, \mathcal{H}_0)$, we have*

$$\|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \left(\sum_{k=1}^n \|[P_k A_{k-1}, L_\tau]\| + \|[A_n, L_\tau]\| + \|L_\tau A_n - A_n\|\right)\|x\|. \tag{12}$$

For translations $L_\tau x(u) = L_c x(u) = x(u - c)$, it is easy to see that patch extraction and pooling operators commute with L_c (this is also known as *covariance* or *equivariance* to translations), so that we are left with the term $\|L_c A_n - A_n\|$, which should control translation invariance. For general diffeomorphisms τ, we no longer have exact covariance, but we show below that commutators are stable to τ, in the sense that $\|[P_k A_{k-1}, L_\tau]\|$ is controlled by $\|\nabla\tau\|_\infty$, while $\|L_\tau A_n - A_n\|$ is controlled by $\|\tau\|_\infty$ and decays with the pooling size σ_n.

Bound on $\|[P_k A_{k-1}, L_\tau]\|$. We note that $P_k z$ can be identified with $(L_c z)_{c\in S_k}$ isometrically for all z in $L^2(\mathbb{R}^d, \mathcal{H}_{k-1})$, since $\|P_k z\|^2 = \int_{S_k} \|L_c z\|^2 d\nu_k(c)$ by Fubini's theorem. Then,

$$\begin{aligned}\|P_k A_{k-1} L_\tau z - L_\tau P_k A_{k-1} z\|^2 &= \int_{S_k} \|L_c A_{k-1} L_\tau z - L_\tau L_c A_{k-1} z\|^2 d\nu_k(c) \\ &\leq \sup_{c\in S_k} \|L_c A_{k-1} L_\tau z - L_\tau L_c A_{k-1} z\|^2,\end{aligned}$$

so that $\|[P_k A_{k-1}, L_\tau]\| \leq \sup_{c\in S_k} \|[L_c A_{k-1}, L_\tau]\|$. The following result lets us bound the commutator $\|[L_c A_{k-1}, L_\tau]\|$ when $|c| \leq \kappa\sigma_{k-1}$, which is satisfied under assumption (A2).

Lemma 5 (Stability of shifted pooling) *Consider A_σ the pooling operator with kernel $h_\sigma(u) = \sigma^{-d}h(u/\sigma)$. If $\|\nabla\tau\|_\infty \leq 1/2$, there exists a constant C_1 such that for any σ and $|c| \leq \kappa\sigma$, we have*

$$\|[L_c A_\sigma, L_\tau]\| \leq C_1 \|\nabla\tau\|_\infty,$$

where C_1 depends only on h and κ.

A similar result can be found in Lemma E.1 of Mallat (2012) for commutators of the form $[A_\sigma, L_\tau]$, but we extend it to handle integral operators $L_c A_\sigma$ with a shifted kernel. The proof (given in Appendix C.4) follows closely Mallat (2012) and relies on the fact that $[L_c A_\sigma, L_\tau]$ is an integral operator in order to bound its norm via Schur's test. Note that κ can be made larger, at the cost of an increase of the constant C_1 of the order κ^{d+1}.

Bound on $\|L_\tau A_n - A_n\|$. We bound the operator norm $\|L_\tau A_n - A_n\|$ in terms of $\|\tau\|_\infty$ using the following result due to Mallat (2012, Lemma 2.11), with $\sigma = \sigma_n$:

Lemma 6 (Translation invariance) *If $\|\nabla\tau\|_\infty \leq 1/2$, we have*

$$\|L_\tau A_\sigma - A_\sigma\| \leq \frac{C_2}{\sigma}\|\tau\|_\infty,$$

with $C_2 = 2^d \cdot \|\nabla h\|_1$.

Combining Proposition 4 with Lemmas 5 and 6, we now obtain the following result:

Theorem 7 (Stability bound) *Assume* (A2). *If $\|\nabla\tau\|_\infty \leq 1/2$, we have*

$$\|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \left(C_1\,(1+n)\,\|\nabla\tau\|_\infty + \frac{C_2}{\sigma_n}\|\tau\|_\infty\right)\|x\|. \tag{13}$$

This result matches the desired notion of stability in Eq. (11), with a translation-invariance factor that decays with σ_n. We discuss implications of our bound, and compare it with related work on stability in Section 3.2. We also note that our bound yields a worst-case guarantee on stability, in the sense that it holds for any signal x. In particular, making additional assumptions on the signal (*e.g.*, smoothness) may lead to improved stability. The predictions for a specific model may also be more stable than applying (1) to our stability bound, for instance if the filters are smooth enough.

Remark 8 (Stability for Lipschitz non-linear mappings) *While the previous results require non-expansive non-linear mappings φ_k, it is easy to extend the result to the following more general condition*

$$\|\varphi_k(z) - \varphi_k(z')\| \leq \rho_k\|z - z'\| \quad \text{and} \quad \|\varphi_k(z)\| \leq \rho_k\|z\|.$$

Indeed, the proof of Proposition 4 easily extends to this setting, giving an additional factor $\prod_k \rho_k$ in the bound (11). *The stability bound* (13) *then becomes*

$$\|\Phi_n(L_\tau x) - \Phi_n(x)\| \leq \left(\prod_{k=1}^{n}\rho_k\right)\left(C_1\,(1+n)\,\|\nabla\tau\|_\infty + \frac{C_2}{\sigma_n}\|\tau\|_\infty\right)\|x\|. \tag{14}$$

This will be useful for obtaining stability of CNNs with generic activations such as ReLU (see Section 4.3), and this also captures the case of kernels with $\kappa_k'(1) > 1$ in Lemma 1.

3.2. Discussion of the Stability Bound (Theorem 7)

In this section, we discuss the implications of our stability bound (13), and compare it to related work on the stability of the scattering transform (Mallat, 2012) as well as the work of (Wiatowski and Bölcskei, 2018) on more general convolutional models.

Role of depth. Our bound displays a linear dependence on the number of layers n in the stability constant $C_1(1+n)$. We note that a dependence on a notion of depth (the number of layers n here) also appears in Mallat (2012), with a factor equal to the maximal length of "scattering paths", and with the same condition $\|\nabla\tau\|_\infty \leq 1/2$. Nevertheless, the number of layers is tightly linked to the patch sizes, and we now show how a deeper architecture can be beneficial for stability. Given a desired level of translation-invariance σ_f and a given initial resolution σ_0, the above bound together with the discretization results of Section 2.1 suggest that one can obtain a stable representation that preserves signal information by taking small patches at each layer and subsampling with a factor equal to the patch size (assuming a patch size greater than one) until the desired level of invariance is reached: in this case we have $\sigma_f/\sigma_0 \approx \kappa^n$, where κ is of the order of the patch size, so that $n = O(\log(\sigma_f/\sigma_0)/\log(\kappa))$, and hence the stability constant $C_1(1+n)$ grows with κ as $\kappa^{d+1}/\log(\kappa)$, explaining the benefit of small patches, and thus of deeper models.

Norm preservation. While the scattering representation preserves the norm of the input signals when the length of scattering paths goes to infinity, in our setting the norm may decrease with depth due to pooling layers. However, we show in Appendix C.5 that a part of the signal norm is still preserved, particularly for signals with high energy in the low frequencies, as is the case for natural images (*e.g.*, Torralba and Oliva, 2003). This justifies that the bounded quantity in (13) is relevant and non-trivial. Nevertheless, we recall that despite a possible loss in norm, our (infinite-dimensional) representation $\Phi(x)$ preserves signal information, as discussed in Section 2.1.

Dependence on signal bandwidth. We note that our stability result crucially relies on the assumption $\sigma_0 > 0$, which effectively limits its applicability to signals with frequencies bounded by $\lambda_0 \approx 1/\sigma_0$. While this assumption is realistic in practice for digital signals, our bound degrades as σ_0 approaches 0, since the number of layers n grows as $\log(1/\sigma_0)$, as explained above. This is in contrast to the stability bound of Mallat (2012), which holds uniformly over any such σ_0, thanks to the use of more powerful tools from harmonic analysis such as the Cotlar-Stein lemma, which allows to control stability simultaneously at all frequencies thanks to the structure of the wavelet transform, something which seems more challenging in our case due to the non-linearities separating different scales.

We note that it may be difficult to obtain meaningful stability results for an unbounded frequency support given a fixed architecture, without making assumptions about the filters of a specific model. In particular, if we consider a model with a high frequency Fourier or cosine filter at the first layer, supported on a large enough patch relative to the corresponding wavelength, this will cause instabilities, particularly if the input signal has isolated high frequencies (see, *e.g.*, Bruna and Mallat, 2013). By the arguments of Section 4, such an unstable model g is in the RKHS, and we then have that the final representation $\Phi(\cdot)$ is

also unstable, since

$$\|\Phi(L_\tau x) - \Phi(x)\| = \sup_{f \in \mathcal{H}_{\mathcal{K}_n}, \|f\| \leq 1} \langle f, \Phi(L_\tau x) - \Phi(x) \rangle$$
$$\geq \frac{1}{\|g\|} \langle g, \Phi(L_\tau x) - \Phi(x) \rangle = \frac{1}{\|g\|} (g(L_\tau x) - g(x)).$$

Comparison with Wiatowski and Bölcskei (2018). The work of Wiatowski and Bölcskei (2018) also studies deformation stability for generic convolutional network models, however their "deformation sensitivity" result only shows that the representation is as sensitive to deformations as the original signal, something which is also applicable here thanks to the non-expansiveness of our representation. Moreover, their bound does not show the dependence on deformation size (the Jacobian norm), and displays a translation invariance part that degrades linearly with $1/\sigma_0$. In contrast, the translation invariance part of our bound is independent of σ_0, and the overall bound only depends logarithmically on $1/\sigma_0$, by exploiting architectural choices such as pooling layers and patch sizes.

3.3. Stability with Kernel Approximations

As in the analysis of the scattering transform of Mallat (2012), we have characterized the stability and shift-invariance of the data representation for continuous signals, in order to give some intuition about the properties of the corresponding discrete representation, which we have described in Section 2.1.

Another approximation performed in the CKN model of Mairal (2016) consists of adding projection steps on finite-dimensional subspaces of the RKHS's layers, as discusssed in Section 2.2. Interestingly, the stability properties we have obtained previously are compatible with these steps. We may indeed replace the operator M_k with the operator $\tilde{M}_k z(u) = \Pi_k \varphi_k(z(u))$ for any map z in $L^2(\mathbb{R}^d, \mathcal{P}_k)$, instead of $M_k z(u) = \varphi_k(z(u))$; $\Pi_k : \mathcal{H}_k \to \mathcal{F}_k$ is here an orthogonal projection operator onto a linear subspace, given in (8). Then, $\tilde{M}_k$ does not necessarily preserve the norm anymore, but $\|\tilde{M}_k z\| \leq \|z\|$, with a loss of information equal to $\|M_k z - \tilde{M}_k z\|$ corresponding to the quality of approximation of the kernel K_k on the points $z(u)$. On the other hand, the non-expansiveness of M_k is satisfied thanks to the non-expansiveness of the projection. In summary, it is possible to show that the conclusions of Theorem 7 remain valid when adding the CKN projection steps at each layer, but some signal information is lost in the process.

3.4. Empirical Study of Stability

In this section, we provide numerical experiments to demonstrate the stability properties of the kernel representations defined in Section 2 on discrete images.

We consider images of handwritten digits from the Infinite MNIST dataset of Loosli et al. (2007), which consists of 28x28 grayscale MNIST digits augmented with small translations and deformations. Translations are chosen at random from one of eight possible directions, while deformations are generated by considering small smooth deformations τ, and approximating $L_\tau x$ using a tangent vector field ∇x containing partial derivatives of the signal x along the horizontal and vertical image directions. We introduce a deformation

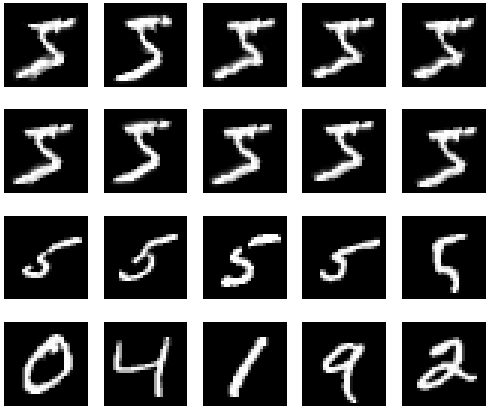


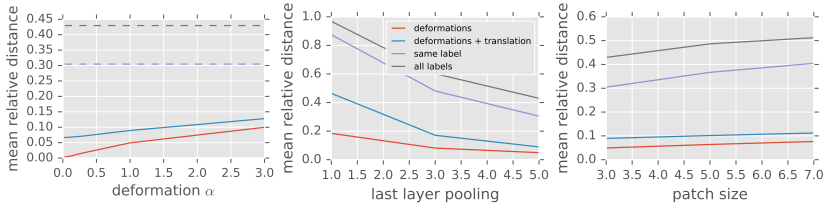
Figure 2: MNIST digits with transformations considered in our numerical study of stability. Each row gives examples of images from a set of digits that are compared to a reference image of a "5". From top to bottom: deformations with $\alpha = 3$; translations and deformations with $\alpha = 1$; digits from the training set with the same label "5" as the reference digit; digits from the training set with any label.

parameter α to control such deformations, which are then given by

$$L_{\alpha\tau} x(u) = x(u - \alpha\tau(u)) \approx x(u) - \alpha\tau(u) \cdot \nabla x(u).$$

Figure 2 shows examples of different deformations, with various values of α, with or without translations, generated from a reference image of the digit "5". In addition, one may consider that a given reference image of a handwritten digit can be deformed into different images of the same digit, and perhaps even into a different digit (*e.g.*, a "1" may be deformed into a "7"). Intuitively, the latter transformation corresponds to a "larger" deformation than the former, so that a prediction function that is stable to deformations should be preferable for a classification task. The aim of our experiments is to quantify this stability, and to study how it is affected by architectural choices such as patch sizes and pooling scales.

We consider a full kernel representation, discretized as described in Section 2.1. We limit ourselves to 2 layers in order to make the computation of the full kernel tractable. Patch extraction is performed with zero padding in order to preserve the size of the previous feature map. We use a homogeneous dot-product kernel as in Eq. (2) with $\kappa(z) = e^{\rho(z-1)}$, $\rho = 1/(0.65)^2$. Note that this choice yields $\kappa'(z) = \rho > 1$, giving an ρ-Lipschitz kernel mapping instead of a non-expansive one as in Lemma 1 which considers $\rho = 1$. However, values of ρ larger than one typically lead to better empirical performance for classification (Mairal, 2016), and the stability results of Section 3 are still valid with an additional factor ρ^n (with $n = 2$ here) in Eq. (13). For a subsampling factor s, we apply a Gaussian filter with scale $\sigma = s/\sqrt{2}$ before downsampling. Our C++ implementation for computing the full kernel given two images is available at `https://github.com/albietz/ckn_kernel`.



(a) (b) (c)

Figure 3: Average relative representation distance for various 2-layer models. Lines in the legend corresponds to rows of images in Figure 2. In (b-c), deformations are obtained with $\alpha = 1$. We show the impact on relative distance of: (a) the value of α in deformations, in $\{0.01, 0.03, 0.1, 0.3, 1, 3\}$; (b) the subsampling factor of the final pooling layer, in $\{1, 3, 5\}$; (c) the patch size, in $\{3, 5, 7\}$.

In Figure 3, we show average relative distance in representation space between a reference image and images from various sets of 20 images (either generated transformations, or images appearing in the training set). For a given architecture A and set S of images, the average relative distance to an image x is given by

$$\frac{1}{|S|}\sum_{x' \in S} \frac{\|\Phi_A(x') - \Phi_A(x)\|}{\|\Phi_A(x)\|} = \frac{1}{|S|}\sum_{x' \in S} \frac{\sqrt{K_A(x,x) + K_A(x',x') - 2K_A(x,x')}}{\sqrt{K_A(x,x)}},$$

where Φ_A denotes the kernel representation for architecture A and $K_A(x, x')$ the corresponding kernel. We normalize by $\|\Phi_A(x)\|$ in order to reduce sensitivity to the choice of architecture. We start with a $(3, 2)$-layer followed by a $(3, 5)$-layer, where (p, s) indicates a layer with patch size p and subsampling s. In Figure 3b, we vary the subsampling factor of the second layer, and in Figure 3c we vary the patch size of both layers.

Each row of Figure 2 shows digits and deformed versions. Intuitively, it should be easier to deform an image of a handwritten 5 into a different image of a 5, than into a different digit. Indeed, Figure 3 shows that the average relative distance for images with different labels is always larger than for images with the same label, which in turn is larger than for small deformations and translations of the reference image.

Adding translations on top of deformations increases distance in all cases, and Figure 3b shows that this gap is smaller when using larger subsampling factors in the last layer. This agrees with the stability bound (13), which shows that a larger pooling scale at the last layer increases translation invariance. Figure 3a highlights the dependence of the distance on the deformation size α, which is near-linear as in Eq. (13) (note that α controls the Jacobian of the deformation). Finally, Figure 3c shows that larger patch sizes can make the representations less stable, as discussed in Section 3.

3.5. Global Invariance to Group Actions

In Section 3.1, we have seen how the kernel representation of Section 2 creates invariance to translations by commuting with the action of translations at intermediate layers, and how

the last pooling layer on the translation group governs the final level of invariance. It is often useful to encode invariances to different groups of transformations, such as rotations or reflections (see, *e.g.*, Cohen and Welling, 2016; Mallat, 2012; Raj et al., 2017; Sifre and Mallat, 2013). Here, we show how this can be achieved by defining adapted patch extraction and pooling operators that commute with the action of a transformation group G (this is known as group covariance or equivariance). We assume that G is locally compact such that we can define a left-invariant Haar measure μ—that is, a measure on G that satisfies $\mu(gS) = \mu(S)$ for any Borel set $S \subseteq G$ and g in G. We assume the initial signal $x(u)$ is defined on G, and we define subsequent feature maps on the same domain. The action of an element g in G is denoted by L_g, where $L_g x(u) = x(g^{-1}u)$. In order to keep the presentation simple, we ignore some issues related to the general construction in $L^2(G)$ of our signals and operators, which can be made more precise using tools from abstract harmonic analysis (*e.g.*, Folland, 2016).

Extending a signal on G. We note that the original signal is defined on a domain $\mathbb{R}^d$ which may be different from the transformation group G that acts on $\mathbb{R}^d$ (*e.g.*, for 2D images the domain is $\mathbb{R}^2$ but G may also include a rotation angle). The action of g in G on the original signal defined on $\mathbb{R}^d$, denoted $\tilde{x}(\omega)$ yields a transformed signal $L_g\tilde{x}(\omega) = \tilde{x}(g^{-1} \cdot \omega)$, where $\cdot$ denotes group action. This requires an appropriate extension of the signal to G that preserves the meaning of signal transformations. We make the following assumption: every element ω in $\mathbb{R}^d$ can be reached with a transformation u_ω in G from a neutral element ϵ in $\mathbb{R}^d$ (*e.g.*, $\epsilon = 0$), as $\omega = u_\omega \cdot \epsilon$. Note that for 2D images ($d = 2$), this typically requires a group G that is "larger" than translations, such as the roto-translation group, while it is not satisfied, for instance, for rotations only. A similar assumption is made by Kondor and Trivedi (2018). Then, one can extend the original signal $\tilde{x}$ by defining $x(u) := \tilde{x}(u \cdot \epsilon)$. Indeed, we then have

$$L_g x(u_\omega) = x(g^{-1}u_\omega) = \tilde{x}((g^{-1}u_\omega) \cdot \epsilon) = \tilde{x}(g^{-1} \cdot \omega),$$

so that the signal $(x(u_\omega))_{\omega \in \mathbb{R}^d}$ preserves the structure of $\tilde{x}$. We detail this below for the example of roto-translations on 2D images. Then, we are interested in defining a layer—that is, a succession of patch extraction, kernel mapping, and pooling operators—that commutes with L_g, in order to achieve equivariance to G.

Patch extraction. We define patch extraction as follows

$$Px(u) = (x(uv))_{v \in S} \quad \text{for all} \quad u \in G,$$

where $S \subset G$ is a patch shape centered at the identity element. P commutes with L_g since

$$PL_g x(u) = (L_g x(uv))_{v \in S} = (x(g^{-1}uv))_{v \in S} = Px(g^{-1}u) = L_g Px(u).$$

Kernel mapping. The pointwise operator M is defined exactly as in Section 2, and thus commutes with L_g.

Pooling. The pooling operator on the group G is defined by

$$Ax(u) = \int_G x(uv)h(v)d\mu(v) = \int_G x(v)h(u^{-1}v)d\mu(v),$$

where h is a pooling filter typically localized around the identity element. The construction is similar to Raj et al. (2017) and it is easy to see from the first expression of $Ax(u)$ that $AL_g x(u) = L_g Ax(u)$, making the pooling operator G-equivariant. One may also pool on a subset of the group by only integrating over the subset in the first expression, an operation which is also G-equivariant.

In our analysis of stability in Section 3.1, we saw that inner pooling layers are useful to guarantee stability to local deformations, while global invariance is achieved mainly through the last pooling layer. In some cases, one only needs stability to a subgroup of G, while achieving invariance to the whole group, *e.g.*, in the roto-translation group (Oyallon and Mallat, 2015; Sifre and Mallat, 2013), one might want invariance to a global rotation but stability to local translations. Then, one can perform patch extraction and pooling just on the subgroup to stabilize (*e.g.*, translations) in intermediate layers, while pooling on the entire group at the last layer to achieve the global group invariance.

Example with the roto-translation group. We consider a simple example on 2D images where one wants global invariance to rotations in addition to near-invariance and stability to translations as in Section 3.1. For this, we consider the roto-translation group (see, *e.g.*, Sifre and Mallat, 2013), defined as the *semi-direct* product of translations $\mathbb{R}^2$ and rotations $SO(2)$, denoted by $G = \mathbb{R}^2 \rtimes SO(2)$, with the following group operation

$$gg' = (v + R_\theta v', \theta + \theta'),$$

for $g = (v, \theta)$, $g' = (v', \theta')$ in G, where R_θ is a rotation matrix in $SO(2)$. The element $g = (v, \theta)$ in G acts on a location $u \in \mathbb{R}^2$ by combining a rotation and a translation:

$$\begin{aligned} g \cdot u &= v + R_\theta u \\ g^{-1} \cdot u &= (-R_{-\theta} v, -\theta) \cdot u = R_{-\theta}(u - v). \end{aligned}$$

For a given image $\tilde{x}$ in $L^2(\mathbb{R}^2)$, our equivariant construction outlined above requires an extension of the signal to the group G. We consider the Haar measure given by $d\mu((v, \theta)) := dv d\mu_c(\theta)$, where dv is the Lebesgue measure on $\mathbb{R}^2$ and $d\mu_c$ the normalized Haar measure on the unit circle. Note that μ is left-invariant, since the determinant of rotation matrices that appears in the change of variables is 1. We can then define x by $x((u, \eta)) := \tilde{x}(u)$ for any angle η, which is in $L^2(G)$ and preserves the definition of group action on the original signal $\tilde{x}$ since

$$L_g x((u, \eta)) = x(g^{-1}(u, \eta)) = x((g^{-1} \cdot u, \eta - \theta)) = \tilde{x}(g^{-1} \cdot u) = L_g \tilde{x}(u).$$

That is, we can study the action of G on 2D images in $L^2(\mathbb{R}^2)$ by studying the action on the extended signals in $L^2(G)$ defined above.

We can now define patch extraction and pooling operators $P, A : L^2(G) \to L^2(G)$ only on the translation subgroup, by considering a patch shape $S = \{(v, 0)\}_{v \in \tilde{S}} \subset G$ with $\tilde{S} \subset \mathbb{R}^2$ for P, and defining pooling by $Ax(g) = \int_{\mathbb{R}^d} x(g(v, 0))h(v)dv$, where h is a Gaussian pooling filter with scale σ defined on $\mathbb{R}^2$.

The following result, proved in Appendix C, shows analogous results to the stability lemmas of Section 3.1 for the operators P and A. For a diffeomorphism τ, we denote by L_τ the action operator given by $L_\tau x((u, \eta)) = x((\tau(u), 0)^{-1}(u, \eta)) = x((u - \tau(u), \eta))$.

Lemma 9 (Stability with roto-translation patches) *If $\|\nabla\tau\|_\infty \leq 1/2$, and the following condition holds $\sup_{c \in \tilde{S}} |c| \leq \kappa\sigma$, we have*

$$\|[PA, L_\tau]\| \leq C_1 \|\nabla\tau\|_\infty,$$

with the same constant C_1 as in Lemma 5, which depends on h and κ. Similarly, we have

$$\|L_\tau A - A\| \leq \frac{C_2}{\sigma} \|\tau\|_\infty,$$

with C_2 as defined in Lemma 6.

By constructing a multi-layer representation $\Phi_n(x)$ in $L^2(G)$ using similar operators at each layer, we can obtain a similar stability result to Theorem 7. By adding a global pooling operator $A_c : L^2(G) \to L^2(\mathbb{R}^2)$ after the last layer, defined, for $x \in L^2(G)$, as

$$A_c x(u) = \int x((u, \eta)) d\mu_c(\eta),$$

we additionally obtain global invariance to rotations, as shown in the following theorem.

Theorem 10 (Stability and global rotation invariance) *Assume* (A2) *for patches $\tilde{S}$ at each layer. Define the operator $L_{(\tau,\theta)} x((u, \eta)) = x((\tau(u), \theta)^{-1}(u, \eta))$, and define the diffeomorphism $\tau_\theta : u \mapsto R_{-\theta}\tau(u)$. If $\|\nabla\tau\|_\infty \leq 1/2$, we have*

$$\begin{aligned} \|A_c \Phi_n(L_{(\tau,\theta)} x) - A_c \Phi_n(x)\| &\leq \|\Phi_n(L_{R_{\tau_\theta}} x) - \Phi_n(x)\| \\ &\leq \left(C_1 (1 + n) \|\nabla\tau\|_\infty + \frac{C_2}{\sigma_n} \|\tau\|_\infty \right) \|x\|. \end{aligned}$$

We note that a similar result may be obtained when $G = \mathbb{R}^d \rtimes H$, where H is any compact group, with a possible additional dependence on how elements of H affect the size of patches.

4. Link with Existing Convolutional Architectures

In this section, we study the functional spaces (RKHS) that arise from our multilayer kernel representation, and examine the connections with more standard convolutional architectures. The motivation of this study is that if a CNN model f is in the RKHS, then it can be written in a "linearized" form $f(x) = \langle f, \Phi(x) \rangle$, so that our study of stability of the kernel representation Φ extends to predictions using $|f(x) - f(x')| \leq \|f\| \|\Phi(x) - \Phi(x')\|$.

We begin by considering in Section 4.1 the intermediate kernels K_k, showing that their RKHSs contain simple neural-network-like functions defined on patches with smooth activations, while in Section 4.2 we show that a certain class of generic CNNs are contained in the RKHS $\mathcal{H}_{\mathcal{K}_n}$ of the full multilayer kernel $\mathcal{K}_n$ and characterize their norm. This is achieved by considering particular functions in each intermediate RKHS defined in terms of the convolutional filters of the CNN. A consequence of these results is that our stability and invariance properties from Section 3 are valid for this broad class of CNNs.

4.1. Activation Functions and Kernels K_k

Before introducing formal links between our kernel representation and classical convolutional architectures, we study in more details the kernels K_k described in Section 2 and their RKHSs $\mathcal{H}_k$. In particular, we are interested in characterizing which types of functions live in $\mathcal{H}_k$. The next lemma extends some results of Zhang et al. (2016, 2017b), originally developed for the inverse polynomial and Gaussian kernels; it shows that the RKHS may contain simple "neural network" functions with activations σ that are smooth enough.

Lemma 11 (Activation functions and RKHSs $\mathcal{H}_k$) *Let $\sigma : [-1,1] \to \mathbb{R}$ be a function that admits a polynomial expansion $\sigma(u) := \sum_{j=0}^{\infty} a_j u^j$. Consider a kernel K_k from Section 2, given in (2), with $\kappa_k(u) = \sum_{j=0}^{\infty} b_j u^j$, and $b_j \geq 0$ for all j. Assume further that $a_j = 0$ whenever $b_j = 0$, and define the function $C_\sigma^2(\lambda^2) := \sum_{j=0}^{\infty} (a_j^2/b_j)\lambda^{2j}$. Let g in $\mathcal{P}_k$ be such that $C_\sigma^2(\|g\|^2) < \infty$. Then, the RKHS $\mathcal{H}_k$ contains the function*

$$f : z \mapsto \|z\| \sigma(\langle g, z\rangle / \|z\|), \tag{15}$$

and its norm satisfies $\|f\| \leq C_\sigma(\|g\|^2)$.

Noting that for all examples of κ_k given in Section 2, we have $b_1 > 0$, this result implies the next corollary, which was also found to be useful in our analysis.

Corollary 12 (Linear functions and RKHSs) *The RKHSs $\mathcal{H}_k$ for the examples of κ_k given in Section 2 contain all linear functions of the form $z \mapsto \langle g, z\rangle$ with g in $\mathcal{P}_k$.*

The previous lemma shows that for many choices of smooth functions σ, the RKHS $\mathcal{H}_k$ contains the functions of the form (15). While the non-homogeneous functions $z \mapsto \sigma(\langle g, z\rangle)$ are standard in neural networks, the homogeneous variant is not. Yet, we note that (i) the most successful activation function, namely rectified linear units, is homogeneous—that is, $\mathrm{relu}(\langle g, z\rangle) = \|z\| \mathrm{relu}(\langle g, z\rangle / \|z\|)$; (ii) while relu is nonsmooth and thus not in our RKHSs, there exists a smoothed variant that satisfies the conditions of Lemma 11 for useful kernels. As noticed by Zhang et al. (2016, 2017b), this is for instance the case for the inverse polynomial kernel. In Figure 4, we plot and compare these different variants of ReLU.

4.2. Convolutional Neural Networks and their Complexity

We now study the connection between the kernel representation defined in Section 2 and CNNs. Specifically, we show that the RKHS of the final kernel $\mathcal{K}_n$ obtained from our kernel construction contains a set of CNNs on continuous domains with certain types of smooth homogeneous activations. An important consequence is that the stability results of previous sections apply to this class of CNNs, although the stability depends on the RKHS norm, as discussed later in Section 5. This norm also serves as a measure of model complexity, thus controlling both generalization and stability.

CNN maps construction. We now define a CNN function f_σ that takes as input an image z_0 in $L^2(\mathbb{R}^d, \mathbb{R}^{p_0})$ with p_0 channels, and build a sequence of feature maps, represented at layer k as a function z_k in $L^2(\mathbb{R}^d, \mathbb{R}^{p_k})$ with p_k channels; the map z_k is obtained from z_{k-1} by performing linear convolutions with a set of filters $(w_k^i)_{i=1,\dots,p_k}$, followed by a pointwise

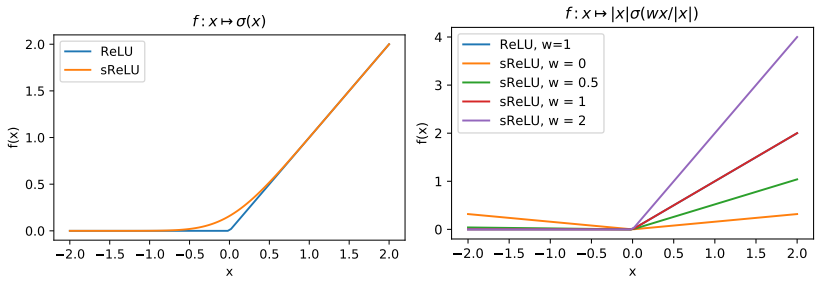


Figure 4: Comparison of one-dimensional functions obtained with relu and smoothed relu (sReLU) activations. (Left) non-homogeneous setting of Zhang et al. (2016, 2017b). (Right) our homogeneous setting, for different values of the parameter w. Note that for $w \geq 0.5$, sReLU and ReLU are indistinguishable.

activation function σ to obtain an intermediate feature map $\tilde{z}_k$, then by applying a linear pooling filter. Note that each w_k^i is in $L^2(S_k, \mathbb{R}^{p_{k-1}})$, with channels denoted by w_k^{ij} in $L^2(S_k, \mathbb{R})$. Formally, the intermediate map $\tilde{z}_k$ in $L^2(\mathbb{R}^d, \mathbb{R}^{p_k})$ is obtained by

$$\tilde{z}_k^i(u) = n_k(u) \sigma\big(\langle w_k^i, P_k z_{k-1}(u)\rangle / n_k(u)\big)\,, \tag{16}$$

where $\tilde{z}_k(u) = (\tilde{z}_k^1(u), \dots, \tilde{z}_k^{p_k}(u))$ is in $\mathbb{R}^{p_k}$, and P_k is a patch extraction operator for finite-dimensional maps. The activation involves a pointwise non-linearity σ along with a quantity $n_k(u) := \|P_k x_{k-1}(u)\|$ in (16), which is due to the homogenization, and which is independent of the filters w_k^i. Finally, the map z_k is obtained by using a pooling operator as in Section 2, with $z_k = A_k \tilde{z}_k$, and $z_0 = x_0$.

Prediction layer. For simplicity, we consider the case of a linear fully connected prediction layer. In this case, the final CNN prediction function f_σ is given by

$$f_\sigma(x_0) = \langle w_{n+1}, z_n\rangle,$$

with parameters w_{n+1} in $L^2(\mathbb{R}^d, \mathbb{R}^{p_n})$. We now show that such a CNN function is contained in the RKHS of the kernel $\mathcal{K}_n$ defined in (6).

Construction in the RKHS. The function f_σ can be constructed recursively from intermediate functions that lie in the RKHSs $\mathcal{H}_k$, of the form (15), for appropriate activations σ. Specifically, we define initial quantities f_1^i in $\mathcal{H}_1$ and g_1^i in $\mathcal{P}_1$ for $i = 1, \dots, p_1$ such that

$$g_1^i = w_1^i \in L^2(S_1, \mathbb{R}^{p_0}) = L^2(S_1, \mathcal{H}_0) = \mathcal{P}_1,$$
$$f_1^i(z) = \|z\| \sigma(\langle g_i^0, z\rangle / \|z\|) \quad \text{for } z \in \mathcal{P}_1,$$

and we define, from layer k–1, the quantities f_k^i in $\mathcal{H}_k$ and g_k^i in $\mathcal{P}_k$ for $i = 1, \dots, p_k$:

$$g_k^i(v) = \sum_{j=1}^{p_{k-1}} w_k^{ij}(v) f_{k-1}^j \quad \text{where} \quad w_k^i(v) = (w_k^{ij}(v))_{j=1,\dots,p_{k-1}},$$
$$f_k^i(z) = \|z\| \sigma(\langle g_k^i, z\rangle / \|z\|) \quad \text{for } z \in \mathcal{P}_k.$$

For the linear prediction layer, we define g_σ in $L^2(\mathbb{R}^d, \mathcal{H}_n)$ by:

$$g_\sigma(u) = \sum_{j=1}^{p_n} w_{n+1}^j(u) f_n^j \quad \text{for all } u \in \mathbb{R}^d,$$

so that the function $f : x_0 \mapsto \langle g_\sigma, x_n \rangle$ is in the RKHS of $\mathcal{K}_n$, where x_n is the final representation given in Eq. (5). In Appendix D.2, we show that $f = f_\sigma$, which implies that the CNN function f_σ is in the RKHS. We note that a similar construction for fully connected multilayer networks with constraints on weights and inputs was given by Zhang et al. (2016).

Norm of the CNN f_σ. We now study the RKHS norm of the CNN constructed above. This quantity is important as it controls the stability and invariance of the predictions of a learned model through (1). Additionally, the RKHS norm provides a way to control model complexity, and can lead to generalization bounds, *e.g.*, through Rademacher complexity and margin bounds (Boucheron et al., 2005; Shalev-Shwartz and Ben-David, 2014). In particular, such results rely on the following upper bound on the empirical Rademacher complexity of a function class with bounded RKHS norm $\mathcal{F}_\lambda = \{f \in \mathcal{H}_{\mathcal{K}_n} : \|f\| \leq \lambda\}$, for a dataset $\{x^{(1)}, \dots, x^{(N)}\}$:

$$R_N(\mathcal{F}_\lambda) \leq \frac{\lambda \sqrt{\frac{1}{N}\sum_{i=1}^N \mathcal{K}_n(x^{(i)}, x^{(i)})}}{\sqrt{N}}. \tag{17}$$

The bound remains valid when only considering CNN functions in $\mathcal{F}_\lambda$ of the form f_σ, since such a function class is contained in $\mathcal{F}_\lambda$. If we consider a binary classification task with training labels $y^{(i)}$ in $\{-1, 1\}$, on can then obtain a margin-based bound for any function f_N in $\mathcal{F}_\lambda$ obtained from the training set and any margin $\gamma > 0$: with probability $1 - \delta$, we have (see, *e.g.*, Boucheron et al., 2005)

$$L(f_N) \leq L_N^\gamma(f_N) + O\left(\frac{\lambda \sqrt{\frac{1}{N}\sum_{i=1}^N \mathcal{K}_n(x^{(i)}, x^{(i)})}}{\gamma \sqrt{N}} + \sqrt{\frac{\log(1/\delta)}{N}} \right), \tag{18}$$

with

$$L(f) = \mathbb{P}_{(x,y)\sim\mathcal{D}}(yf(x) < 0)$$
$$L_N^\gamma(f) = \frac{1}{N}\sum_{i=1}^N \mathbf{1}\{y^{(i)} f(x^{(i)}) < \gamma\},$$

where $\mathcal{D}$ is the distribution of data-label pairs $(x^{(i)}, y^{(i)})$. Intuitively, the margin γ corresponds to a level of confidence, and L_N^γ measures training error when requiring confident predictions. Then, the bound on the gap between this training error and the true expected error $L(f_N)$ becomes larger for small confidence levels, and is controlled by the model complexity λ and the sample size N.

Note that the bound requires a fixed value of λ used during training, but in practice, learning under a constraint $\|f\| \leq \lambda$ can be difficult, especially for CNNs which are typically trained with stochastic gradient descent with little regularization. However, by considering

values of λ on a logarithmic scale and taking a union bound, one can obtain a similar bound with $\|f_N\|$ instead of λ, up to logarithmic factors (see, *e.g.*, Shalev-Shwartz and Ben-David, 2014, Theorem 26.14), where f_N is obtained from the training data. We note that various authors have recently considered other norm-based complexity measures to control the generalization of neural networks with more standard activations (see, *e.g.*, Bartlett et al., 2017; Liang et al., 2018; Neyshabur et al., 2017, 2015). However, their results are typically obtained for fully connected networks on finite-dimensional inputs, while we consider CNNs for input signals defined on continuous domains. The next proposition (proved in Appendix D.2) characterizes the norm of f_σ in terms of the L^2 norms of the filters w_k^{ij}, and follows from the recursive definition of the intermediate RKHS elements f_k^i.

Proposition 13 (RKHS norm of CNNs) *Assume the activation σ satisfies $C_\sigma(a) < \infty$ for all $a \geq 0$, where C_σ is defined for a given kernel in Lemma 11. Then, the CNN function f_σ defined above is in the RKHS $\mathcal{H}_{\mathcal{K}_n}$, with norm*

$$\|f_\sigma\|^2 \leq p_n \sum_{i=1}^{p_n} \|w_{n+1}^i\|_2^2 B_{n,i},$$

where $B_{n,i}$ is defined by $B_{1,i} = C_\sigma^2(\|w_1^i\|_2^2)$ and $B_{k,i} = C_\sigma^2\left(p_{k-1}\sum_{j=1}^{p_{k-1}} \|w_k^{ij}\|_2^2 B_{k-1,j}\right)$.

Note that this upper bound need not grow exponentially with depth when the filters have small norm and C_σ takes small values around zero. However, the dependency of the bound on the number of feature maps p_k of each layer k may not be satisfactory in situations where the number of parameters is very large, which is common in successful deep learning architectures. The following proposition removes this dependence, relying instead on matrix spectral norms. Similar quantities have been used recently to obtain useful generalization bounds for neural networks (Bartlett et al., 2017; Neyshabur et al., 2018).

Proposition 14 (RKHS norm of CNNs using spectral norms) *Assume the activation σ satisfies $C_\sigma(a) < \infty$ for all $a \geq 0$, where C_σ is defined for a given kernel in Lemma 11. Then, the CNN function f_σ defined above is in the RKHS $\mathcal{H}_{\mathcal{K}_n}$, with norm*

$$\|f_\sigma\|^2 \leq \|w_{n+1}\|^2 \; C_\sigma^2(\|W_n\|_2^2 \; C_\sigma^2(\|W_{n-1}\|_2^2 \dots C_\sigma^2(\|W_2\|_2^2 \; C_\sigma^2(\|W_1\|_F^2)) \dots)). \tag{19}$$

The norms are defined as follows:

$$\|W_k\|_2^2 = \int_{S_k} \|W_k(u)\|_2^2 d\nu_k(u), \quad \text{for } k = 2, \dots, n$$
$$\|W_1\|_F^2 = \int_{S_1} \|W_1(u)\|_F^2 d\nu_1(u),$$

where $W_k(u)$ is the matrix $(w_k^{ij}(u))_{ij}$, $\|\cdot\|_2$ the spectral norm, and $\|\cdot\|_F$ the Frobenius norm.

As an example, if we consider $\kappa_1 = \cdots = \kappa_n$ to be one of the kernels introduced in Section 2 and take $\sigma = \kappa_1$ so that $C_\sigma^2(\lambda^2) = \kappa_1(\lambda^2)$, then constraining the norms at each layer to be smaller than 1 ensures $\|f_\sigma\| \leq 1$, since for $\lambda \leq 1$ we have $C_\sigma^2(\lambda^2) \leq C_\sigma^2(1) = \kappa_1(1) = 1$.

If we consider linear kernels and $\sigma(u) = u$, we have $C_\sigma^2(\lambda^2) = \lambda^2$ and the bound becomes $\|f_\sigma\| \leq \|w_{n+1}\|\|W_n\|_2 \cdots \|W_2\|_2\|W_1\|_F$. If we ignore the convolutional structure (*i.e.*, only taking 1x1 patches on a 1x1 image), the norm involves a product of spectral norms at each layer (ignoring the first layer), a quantity which also appears in recent generalization bounds (Bartlett et al., 2017; Neyshabur et al., 2018). While such quantities have proven useful to explain some generalization phenomena, such as the behavior of networks trained on data with random labels (Bartlett et al., 2017; Zhang et al., 2017a), some authors have pointed out that spectral norms may yield overly pessimistic generalization bounds when comparing with simple parameter counting (Arora et al., 2018), and our results may display similar drawbacks. We note, however, that Proposition 14 only gives an upper bound, and the actual RKHS norm may be smaller in practice. It may also be that the norm is not well controlled during training, and that the obtained bounds may not fully explain the generalization behavior observed in practice. Using such quantities to regularize during training may then yield bounds that are less vacuous (Bietti et al., 2019).

Generalization and stability. The results of this section imply that our study of the geometry of the kernel representations, and in particular the stability and invariance properties of Section 3, apply to the generic CNNs defined above, thanks to the Lipschitz smoothness relation (1). The smoothness is then controlled by the RKHS norm of these functions, which sheds light on the links between generalization and stability. In particular, functions with low RKHS norm provide better generalization guarantees on unseen data, as shown by the margin bound in Eq. (18). This implies, for instance, that generalization is harder if the task requires classifying two slightly deformed images with different labels, since separating such predictions by some margin requires a function with large RKHS norm according to our stability analysis. In contrast, if a stable function (*i.e.*, with small RKHS norm) is sufficient to do well on a training set, learning becomes "easier" and few samples may be enough for good generalization.

4.3. Stability and Generalization with Generic Activations

Our study of stability and generalization so far has relied on kernel methods, which allows us to separate learned models from data representations in order to establish tight connections between the stability of representations and statistical properties of learned CNNs through RKHS norms. One important caveat, however, is that our study is limited to CNNs with a class of smooth and homogeneous activations described in Section 4.1, which differ from generic activations used in practice such as ReLU or tanh. Indeed, ReLU is homogeneous but lacks the required smoothness, while tanh is not homogeneous. In this section, we show that our stability results can be extended to the predictions of CNNs with such activations, and that stability is controlled by a quantity based on spectral norms, which plays an important role in recent results on generalization. This confirms a strong connection between stability and generalization in this more general context as well.

Stability bound. We consider an activation function $\sigma : \mathbb{R} \to \mathbb{R}$ that is ρ-Lipschitz and satisfies $\sigma(0) = 0$. Examples include ReLU and tanh activations, for which $\rho = 1$. The CNN construction is similar to Section 4.2 with feature maps z_k in $L^2(\mathbb{R}^d, \mathbb{R}^{p_k})$, and a final prediction function f_σ defined with an inner product $\langle w_{n+1}, z_n \rangle$. The only change is the

non-linear mapping in Eq. (16), which is no longer homogeneized, and can be rewritten as

$$\tilde{z}_k(u) = \varphi_k(P_k z_{k-1}(u)) := \sigma\left(\int_{S_k} W_k(v)(P_k z_{k-1}(u))(v) d\nu_k(v)\right),$$

where σ is applied component-wise. The non-linear mapping φ_k on patches satisfies

$$\|\varphi_k(z) - \varphi_k(z')\| \leq \rho_k \|z - z'\| \quad \text{and} \quad \|\varphi_k(z)\| \leq \rho_k \|z\|,$$

where $\rho_k = \rho\|W_k\|_2$ and $\|W_k\|_2$ is the spectral norm of $W_k : L^2(S_k, \mathbb{R}^{p_k - 1}) \to \mathbb{R}^{p_k}$ defined by

$$\|W_k\|_2 = \sup_{\|z\| \leq 1} \left\| \int_{S_k} W_k(v) z(v) d\nu_k(v) \right\|.$$

We note that this spectral norm is slightly different than the mixed norm used in Proposition 14. By defining an operator M_k that applies φ_k pointwise as in Section 2, the construction of the last feature map takes the same form as that of the multilayer kernel representation, so that the results of Section 3 apply, leading to the following stability bound on the final predictions:

$$|f_\sigma(L_\tau x) - f_\sigma(x)| \leq \rho^n \|w_{n+1}\| \left(\prod_k \|W_k\|_2\right) \left(C_1 (1+n) \|\nabla \tau\|_\infty + \frac{C_2}{\sigma_n} \|\tau\|_\infty\right) \|x\|. \quad (20)$$

Link with generalization. The stability bound (20) takes a similar form to the one obtained for CNNs in the RKHS, with the RKHS norm replaced by the product of spectral norms. In contrast to the RKHS norm, such a quantity does not directly lead to generalization bounds; however, a few recent works have provided meaningful generalization bounds for deep neural networks that involve the product of spectral norms (Bartlett et al., 2017; Neyshabur et al., 2018). Thus, this suggests that stable CNNs have better generalization properties, even when considering generic CNNs with ReLU or tanh activations. Nevertheless, these bounds typically involve an additional factor consisting of other matrix norms summed across layers, which may introduce some dependence on the number of parameters, and do not directly support convolutional structure. In contrast, our RKHS norm bound based on spectral norms given in Proposition 14 directly supports convolutional structure, and has no dependence on the number of parameters.

5. Discussion and Concluding Remarks

In this paper, we introduce a multilayer convolutional kernel representation (Section 2); we show that it is stable to the action of diffeomorphisms, and that it can be made invariant to groups of transformations (Section 3); and finally we explain connections between our representation and generic convolutional networks by showing that certain classes of CNNs with smooth activations are contained in the RKHS of the full multilayer kernel (Section 4). A consequence of this last result is that the stability results of Section 3 apply to any CNN function f from that class, by using the relation

$$|f(L_\tau x) - f(x)| \leq \|f\| \|\Phi_n(L_\tau x) - \Phi_n(x)\|,$$

which follows from (1), assuming a linear prediction layer. In the case of CNNs with generic activations such as ReLU, the kernel point of view is not applicable, and the separation between model and representation is not as clear. However, we show in Section 4.3 that a similar stability bound can be obtained, with the product of spectral norms at each layer playing a similar role to the RKHS norm of the CNN. In both cases, a quantity that characterizes complexity of a model appears in the final bound on predicted values — either the RKHS norm or the product of spectral norms —, and this complexity measure is also closely related to generalization. This implies that learning with stable CNNs is "easier" in terms of sample complexity, and that the inductive bias of CNNs is thus suitable to tasks that present some invariance under translation and small local deformation, as well as more general transformation groups, when the architecture is appropriately constructed.

In order to ensure stability, the previous bounds suggest that one should control the RKHS norm $\|f\|$, or the product of spectral norms when using generic activations; however, these quantities are difficult to control with standard approaches to learning CNNs, such as backpropagation. In contrast, traditional kernel methods typically control this norm by using it as an explicit regularizer in the learning process, making such a stability guarantee more useful. In order to avoid the scalability issues of such approaches, convolutional kernel networks approximate the full kernel map Φ_n by taking appropriate projections as explained in Section 2.2, leading to a representation $\tilde{\Phi}_n$ that can be represented with a practical representation ψ_n that preserves the Hilbert space structure isometrically (using the finite-dimensional descriptions of points in the RKHS given in (10)). Section 3.3 shows that such representations satisfy the same stability and invariance results as the full representation, at the cost of losing information. Then, if we consider a CKN function of the form $f_w(x) = \langle w, \psi_n(x) \rangle$, stability is obtained thanks to the relation

$$|f_w(L_\tau x) - f_w(x)| \leq \|w\| \|\psi_n(L_\tau x) - \psi_n(x)\| = \|w\| \|\tilde{\Phi}_n(L_\tau x) - \tilde{\Phi}_n(x)\|.$$

In particular, learning such a function by controlling the norm of w, *e.g.*, with ℓ_2 regularization, provides a natural way to explicitly control stability. In the context of CNNs with generic activations, it has been suggested (see, *e.g.*, Zhang et al., 2017a) that optimization algorithms may play an important role in controlling their generalization ability, and it may be plausible that these impact the RKHS norm of a learned CNN, or its spectral norms. A better understanding of such implicit regularization behavior would be interesting, but falls beyond the scope of this paper. Nevertheless, modern CNNs trained with SGD have been found to be highly unstable to small, additive perturbations known as "adversarial examples" (Szegedy et al., 2014), which suggests that the RKHS norm of these models may be quite large, and that controlling it explicitly during learning might be important to learn more stable models (Bietti et al., 2019; Cisse et al., 2017).

Acknowledgments

This work was supported by a grant from ANR (MACARON project under grant number ANR-14-CE23-0003-01), by the ERC grant number 714381 (SOLARIS project), and from the MSR-Inria joint centre.

Appendix A. Useful Mathematical Tools

In this section, we present preliminary mathematical tools that are used in our analysis.

Harmonic analysis. We recall a classical result from harmonic analysis (see, *e.g.*, Stein, 1993), which was used many times by Mallat (2012) to prove the stability of the scattering transform to the action of diffeomorphisms.

Lemma A.1 (Schur's test) *Let $\mathcal{H}$ be a Hilbert space and Ω a subset of $\mathbb{R}^d$. Consider T an integral operator with kernel $k : \Omega \times \Omega \to \mathbb{R}$, meaning that for all u in Ω and x in $L^2(\Omega, \mathcal{H})$,*

$$Tx(u) = \int_\Omega k(u, v) x(v) dv, \tag{21}$$

where the integral is a Bochner integral (see, Diestel and Uhl, 1977; Muandet et al., 2017) when $\mathcal{H}$ is infinite-dimensional. If

$$\forall u \in \Omega, \quad \int |k(u, v)| dv \leq C \quad \text{and} \quad \forall v \in \Omega, \quad \int |k(u, v)| du \leq C,$$

for some constant C, then, Tx is always in $L^2(\Omega, \mathcal{H})$ for all x in $L^2(\Omega, \mathcal{H})$ and we have $\|T\| \leq C$.

Note that while the proofs of the lemma above are typically given for real-valued functions in $L^2(\Omega, \mathbb{R})$, the result can easily be extended to Hilbert space-valued functions x in $L^2(\Omega, \mathcal{H})$. In order to prove this, we consider the integral operator $|T|$ with kernel $|k|$ that operates on $L^2(\Omega, \mathbb{R}_+)$, meaning that $|T|$ is defined as in (21) by replacing $k(u, v)$ by the absolute value $|k(u, v)|$. Then, consider x in $L^2(\Omega, \mathcal{H})$ and use the triangle inequality property of Bochner integrals:

$$\|Tx\|^2 = \int_\Omega \|Tx(u)\|^2 du \leq \int_\Omega \left(\int_\Omega |k(u, v)| \|x(v)\| dv \right)^2 du = \||T||x|\|^2,$$

where the function $|x|$ is such that $|x|(u) = \|x(u)\|$ and thus $|x|$ is in $L^2(\Omega, \mathbb{R}_+)$. We may now apply Schur's test to the operator $|T|$ for real-valued functions, which gives $\||T|\| \leq C$. Then, noting that $\||x|\| = \|x\|$, we conclude with the inequality $\|Tx\|^2 \leq \||T||x|\|^2 \leq \||T|\|^2 \|x\|^2 \leq C^2 \|x\|^2$.

The following lemma shows that the pooling operators A_k defined in Section 2 are non-expansive.

Lemma A.2 (Non-expansiveness of pooling operators) *If $h(u) := (2\pi)^{-d/2} \exp(-|u|^2/2)$, then the pooling operator A_σ defined for any $\sigma > 0$ by*

$$A_\sigma x(u) = \int_{\mathbb{R}^d} \sigma^{-d} h\left(\frac{u - v}{\sigma} \right) x(v) dv,$$

has operator norm $\|A_\sigma\| \leq 1$.

Proof With the notations from above, we have $\|A_\sigma x\| \leq \||A_\sigma||x|\| = \|h_\sigma * |x|\|$, where $h_\sigma := \sigma^{-d} h(\cdot/\sigma)$ and $*$ denotes convolution. By Young's inequality, we have $\|h_\sigma * |x|\| \leq \|h_\sigma\|_1 \cdot \||x|\| = 1 \cdot \||x|\| = \|x\|$, which concludes the proof. ∎

Kernel methods. We now recall a classical result that characterizes the reproducing kernel Hilbert space (RKHS) of functions defined from explicit Hilbert space mappings (see, *e.g.*, Saitoh, 1997, §2.1).

Theorem A.1 *Let $\psi : \mathcal{X} \to H$ be a feature map to a Hilbert space H, and let $K(z,z') := \langle \psi(z), \psi(z') \rangle_H$ for $z, z' \in \mathcal{X}$. Let $\mathcal{H}$ be the linear subspace defined by*

$$\mathcal{H} := \{ f_w \ ; \ w \in H \} \quad s.t. \quad f_w : z \mapsto \langle w, \psi(z) \rangle_H,$$

and consider the norm

$$\|f_w\|_{\mathcal{H}}^2 := \inf_{w' \in H} \{ \|w'\|_H^2 \quad s.t. \quad f_w = f_{w'} \}.$$

Then $\mathcal{H}$ is the reproducing kernel Hilbert space associated to kernel K.

A consequence of this result is that the RKHS of the kernel $\mathcal{K}_n(x,x') = \langle \Phi(x), \Phi(x') \rangle$, defined from a given final representation $\Phi(x) \in \mathcal{H}_{n+1}$ such as the one introduced in Section 2, contains functions of the form $f : x \mapsto \langle w, \Phi(x) \rangle$ with $w \in \mathcal{H}_{n+1}$, and the RKHS norm of such a function satisfies $\|f\| \leq \|w\|_{\mathcal{H}_{n+1}}$.

Appendix B. Proofs Related to the Multilayer Kernel Construction

B.1. Proof of Lemma 1 and Non-Expansiveness of the Gaussian Kernel

We begin with the proof of Lemma 1 related to homogeneous dot-product kernels (2).

Proof In this proof, we drop all indices k since there is no ambiguity. We will prove the more general result that φ is ρ_k-Lipschitz with $\rho_k = \max(1, \sqrt{\kappa'(1)})$ for any value of $\kappa'(1)$ (in particular, it is non-expansive when $\kappa'(1) \leq 1$).

Let us consider the Maclaurin expansion $\kappa(u) = \sum_{j=0}^{+\infty} b_j u^j < +\infty$ with $b_j \geq 0$ for all j and all u in $[-1,+1]$. Recall that the condition $b_j \geq 0$ comes from the positive-definiteness of K (Schoenberg, 1942). Then, we have $\kappa'(u) = \sum_{j=1}^{+\infty} j b_j u^{j-1}$. Noting that $j b_j u^{j-1} \leq j b_j$ for $u \in [-1,1]$, we have $\kappa'(u) \leq \kappa'(1)$ on $[-1,1]$. The fundamental theorem of calculus then yields, for $u \in [-1,1]$,

$$\kappa(u) = \kappa(1) - \int_u^1 \kappa'(t) dt \geq \kappa(1) - \kappa'(1)(1-u). \tag{22}$$

Then, if $z, z' \neq 0$,

$$\|\varphi(z) - \varphi(z')\|^2 = K(z,z) + K(z',z') - 2K(z,z') = \|z\|^2 + \|z'\|^2 - 2\|z\|\|z'\|\kappa(u),$$

with $u = \langle z, z' \rangle / (\|z\|\|z'\|)$. Using (22) with $\kappa(1) = 1$, we have

$$\begin{aligned}
\|\varphi(z) - \varphi(z')\|^2 &\leq \|z\|^2 + \|z'\|^2 - 2\|z\|\|z'\| \left(1 - \kappa'(1) + \kappa'(1) u\right) \\
&= (1 - \kappa'(1)) \left(\|z\|^2 + \|z'\|^2 - 2\|z\|\|z'\| \right) + \kappa'(1) \left(\|z\|^2 + \|z'\|^2 - 2\langle z, z' \rangle \right) \\
&= (1 - \kappa'(1)) \left| \|z\| - \|z'\| \right|^2 + \kappa'(1) \|z - z'\|^2 \\
&\leq \begin{cases} \|z - z'\|^2, & \text{if } 0 \leq \kappa'(1) \leq 1 \\ \kappa'(1) \|z - z'\|^2, & \text{if } \kappa'(1) > 1 \end{cases} \\
&= \rho_k^2 \|z - z'\|^2,
\end{aligned}$$

with $\rho_k = \max(1, \sqrt{\kappa'(1)})$, which yields the desired result. Finally, we remark that we have shown the relation $\kappa(u) \geq \kappa(1) - \kappa'(1) + \kappa'(1) u$; when $\kappa'(1) = 1$, this immediately yields (3).

If $z = 0$ or $z' = 0$, the result also holds trivially. For example,

$$\|\varphi(z) - \varphi(0)\|^2 = K(z,z) + K(0,0) - 2K(z,0) = \|z\|^2 = \|z - 0\|^2.$$

■

Non-expansiveness of the Gaussian kernel. We now consider the Gaussian kernel

$$K(z,z') := e^{-\frac{\alpha}{2}\|z - z'\|^2},$$

with feature map φ. We simply use the convexity inequality $e^u \geq 1 + u$ for all u, and

$$\|\varphi(z) - \varphi(z')\|^2 = K(z,z) + K(z',z') - 2K(z,z') = 2 - 2e^{-\frac{\alpha}{2}\|z - z'\|^2} \leq \alpha \|z - z'\|^2.$$

In particular, φ is non-expansive when $\alpha \leq 1$.

Appendix C. Proofs of Recovery and Stability Results

C.1. Proof of Lemma 2

Proof We denote by $\bar{\Omega}$ the discrete set of sampling points considered in this lemma. The assumption on $\bar{\Omega}$ can be written as $\{u + v \ ; \ u \in \bar{\Omega}, v \in S_k\} = \mathbb{R}^d$.

Let B denote an orthonormal basis of the Hilbert space $\mathcal{P}_k = L^2(S_k, \mathcal{H}_{k-1})$, and define the linear function f_w in $\mathcal{H}_k$ such that $f_w : z \mapsto \langle w, z \rangle$ for w in $\mathcal{P}_k$. We thus have

$$\begin{aligned}
P_k x_{k-1}(u) &= \sum_{w \in B} \langle w, P_k x_{k-1}(u) \rangle w \\
&= \sum_{w \in B} f_w(P_k x_{k-1}(u)) w \\
&= \sum_{w \in B} \langle f_w, M_k P_k x_{k-1}(u) \rangle w,
\end{aligned}$$

using the reproducing property in the RKHS $\mathcal{H}_k$. Applying the pooling operator A_k yields

$$\begin{aligned}
A_k P_k x_{k-1}(u) &= \sum_{w \in B} \langle f_w, A_k M_k P_k x_{k-1}(u) \rangle w, \\
&= \sum_{w \in B} \langle f_w, x_k(u) \rangle w.
\end{aligned}$$

Noting that $A_k P_k x_{k-1} = A_k (L_v x_{k-1})_{v \in S_k} = (A_k L_v x_{k-1})_{v \in S_k} = (L_v A_k x_{k-1})_{v \in S_k} = P_k A_k x_{k-1}$, with $L_v x_{k-1}(u) := x_{k-1}(u+v)$, we can choose v in S_k and obtain from the previous relations

$$A_k x_{k-1}(u + v) = \sum_{w \in B} \langle f_w, x_k(u) \rangle w(v).$$

Thus, taking all sampling points $u \in \bar{\Omega}$ and all $v \in S_k$, we have a full view of the signal $A_k x_{k-1}$ on all of $\mathbb{R}^d$ by our assumption on the set $\bar{\Omega}$.

For $f \in \mathcal{H}_{k-1}$, the signal $\langle f, x_{k-1}(u)\rangle$ can then be recovered by deconvolution as follows:

$$\langle f, x_{k-1}(u)\rangle = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(\langle f, A_k x_{k-1}(\cdot)\rangle)}{\mathcal{F}(h_{\sigma_k})}\right)(u),$$

where $\mathcal{F}$ denotes the Fourier transform. Note that the inverse Fourier transform is well-defined here because the signal $\langle f, A_k x_k(\cdot)\rangle$ is itself a convolution with h_{σ_k}, and $\mathcal{F}(h_{\sigma_k})$ is strictly positive as the Fourier transform of a Gaussian is also a Gaussian.

By considering all elements f in an orthonormal basis of $\mathcal{H}_{k-1}$, we can recover x_{k-1}. The map x_k can then be reconstructed trivially by applying operators P_k, M_k and A_k on x_{k-1}. ■

C.2. Proof of Lemma 3

Proof In this proof, we drop the bar notation on all quantities for simplicity; there is indeed no ambiguity since all signals are discrete here. First, we recall that $\mathcal{H}_k$ contains all linear functions on $\mathcal{P}_k = \mathcal{H}_{k-1}^{e_k}$; thus, we may consider in particular functions $f_{j,w}(z) := e_k^{1/2}\langle w, z_j\rangle$ for $j \in \{1, \dots, e_k\}$, $w \in \mathcal{H}_{k-1}$, and $z = (z_1, z_2, \dots, z_{e_k})$ in $\mathcal{P}_k$. Then, we may evaluate

$$\begin{aligned}
\langle f_{j,w}, s_k^{-1/2} x_k[n]\rangle &= \sum_{m\in\mathbb{Z}} h_k[ns_k - m]\langle f_{j,w}, M_k P_k x_{k-1}[m]\rangle \\
&= \sum_{m\in\mathbb{Z}} h_k[ns_k - m]\langle f_{j,w}, \varphi_k(P_k x_{k-1}[m])\rangle \\
&= \sum_{m\in\mathbb{Z}} h_k[ns_k - m] f_{j,w}(P_k x_{k-1}[m]) \\
&= \sum_{m\in\mathbb{Z}} h_k[ns_k - m]\langle w, x_{k-1}[m+j]\rangle \\
&= \sum_{m\in\mathbb{Z}} h_k[ns_k + j - m]\langle w, x_{k-1}[m]\rangle \\
&= (h_k * \langle w, x_{k-1}\rangle)[ns_k + j],
\end{aligned}$$

where, with an abuse of notation, $\langle w, x_{k-1}\rangle$ is the real-valued discrete signal such that $\langle w, x_{k-1}\rangle[n] = \langle w, x_{k-1}[n]\rangle$. Since integers of the form $(ns_k + j)$ cover all of $\mathbb{Z}$ according to the assumption $e_k \geq s_k$, we have a full view of the signal $(h_k * \langle w, x_{k-1}\rangle)$ on $\mathbb{Z}$. We will now follow the same reasoning as in the proof of Lemma 2 to recover $\langle w, x_{k-1}\rangle$:

$$\langle w, x_{k-1}\rangle = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(h_k * \langle w, x_{k-1}\rangle)}{\mathcal{F}(h_k)}\right),$$

where $\mathcal{F}$ is the Fourier transform. Since the signals involved there are discrete, their Fourier transform are periodic with period 2π, and we note that $\mathcal{F}(h_k)$ is strictly positive and bounded away from zero. The signal x_{k-1} is then recovered exactly as in the proof of Lemma 2 by considering for w the elements of an orthonormal basis of $\mathcal{H}_{k-1}$. ■

C.3. Proof of Proposition 4

Proof Define $(MPA)_{k:j} := M_k P_k A_{k-1} M_{k-1} P_{k-1} A_{k-2} \cdots M_j P_j A_{j-1}$. Using the fact that $\|A_k\| \leq 1$, $\|P_k\| = 1$ and M_k is non-expansive, we obtain

$$\begin{aligned}
\|\Phi_n(L_\tau x) - \Phi_n(x)\| &= \|A_n(MPA)_{n:2} M_1 P_1 A_0 L_\tau x - A_n(MPA)_{n:2} M_1 P_1 A_0 x\| \\
&\leq \|A_n(MPA)_{n:2} M_1 P_1 A_0 L_\tau x - A_n(MPA)_{n:2} M_1 L_\tau P_1 A_0 x\| \\
&\quad + \|A_n(MPA)_{n:2} M_1 L_\tau P_1 A_0 x - A_n(MPA)_{n:2} M_1 P_1 A_0 x\| \\
&\leq \|[P_1 A_0, L_\tau]\|\|x\| \\
&\quad + \|A_n(MPA)_{n:2} M_1 L_\tau P_1 A_0 x - A_n(MPA)_{n:2} M_1 P_1 A_0 x\|.
\end{aligned}$$

Note that M_1 is defined point-wise, and thus commutes with L_τ:

$$M_1 L_\tau x(u) = \varphi_1(L_\tau x(u)) = \varphi_1(x(u - \tau(u)) = M_1 x(u - \tau(u)) = L_\tau M_1 x(u).$$

By noticing that $\|M_1 P_1 A_0 x\| \leq \|x\|$, we can expand the second term above in the same way. Repeating this by induction yields

$$\begin{aligned}
\|\Phi_n(L_\tau x) - \Phi_n(x)\| &\leq \sum_{k=1}^{n} \|[P_k A_{k-1}, L_\tau]\|\|x\| + \|A_n L_\tau (MPA)_{n:1} x - A_n (MPA)_{n:1} x\| \\
&\leq \sum_{k=1}^{n} \|[P_k A_{k-1}, L_\tau]\|\|x\| + \|A_n L_\tau - A_n\|\|x\|,
\end{aligned}$$

and the result follows by decomposing $A_n L_\tau = [A_n, L_\tau] + L_\tau A_n$ and applying the triangle inequality. ■

C.4. Proof of Lemma 5

Proof The proof follows in large parts the methodology introduced by Mallat (2012) in the analysis of the stability of the scattering transform. More precisely, we will follow in part the proof of Lemma E.1 of Mallat (2012). The kernel (in the sense of Lemma A.1) of A_σ is $h_\sigma(z - u) = \sigma^{-d} h(\frac{z-u}{\sigma})$. Throughout the proof, we will use the following bounds on the decay of h for simplicity, as in Mallat (2012):[2]

$$\begin{aligned}
|h(u)| &\leq \frac{C_h}{(1 + |u|)^{d+2}} \\
|\nabla h(u)| &\leq \frac{C'_h}{(1 + |u|)^{d+2}},
\end{aligned}$$

which are satisfied for the Gaussian function h thanks to its exponential decay.

We now decompose the commutator

$$[L_c A_\sigma, L_\tau] = L_c A_\sigma L_\tau - L_\tau L_c A_\sigma = L_c(A_\sigma - L_c^{-1} L_\tau L_c A_\sigma L_\tau^{-1}) L_\tau = L_c T L_\tau,$$

2. Note that a more precise analysis may be obtained by using finer decay bounds.

with $T := A_\sigma - L_c^{-1} L_\tau L_c A_\sigma L_\tau^{-1}$. Hence,

$$\|[L_c A_\sigma, L_\tau]\| \leq \|L_c\| \|L_\tau\| \|T\|.$$

We have $\|L_c\| = 1$ since the translation operator L_c preserves the norm. Note that we have

$$2^{-d} \leq (1 - \|\nabla\tau\|_\infty)^d \leq \det(I - \nabla\tau(u)) \leq (1 + \|\nabla\tau\|_\infty)^d \leq 2^d, \tag{23}$$

for all $u \in \mathbb{R}^d$. Thus, for $f \in L^2(\mathbb{R}^d)$,

$$\begin{aligned}\|L_\tau f\|^2 &= \int_{\mathbb{R}^d} |f(z - \tau(z))|^2 dz = \int_{\mathbb{R}^d} |f(u)|^2 \det(I - \nabla\tau(u))^{-1} du \\ &\leq (1 - \|\nabla\tau\|_\infty)^{-d} \|f\|^2,\end{aligned}$$

such that $\|L_\tau\| \leq (1 - \|\nabla\tau\|_\infty)^{-d/2} \leq 2^{d/2}$. This yields

$$\|[L_c A_\sigma, L_\tau]\| \leq 2^{d/2} \|T\|.$$

Kernel of T. We now show that T is an integral operator and describe its kernel. Let $\xi = (I - \tau)^{-1}$, so that $L_\tau^{-1} f(z) = f(\xi(z))$ for any function f in $L^2(\mathbb{R}^d)$. We have

$$\begin{aligned}A_\sigma L_\tau^{-1} f(z) &= \int h_\sigma(z - v) f(\xi(v)) dv \\ &= \int h_\sigma(z - u + \tau(u)) f(u) \det(I - \nabla\tau(u)), du\end{aligned}$$

using the change of variable $v = u - \tau(u)$, giving $\left|\frac{dv}{du}\right| = \det(I - \nabla\tau(u))$. Then note that $L_c^{-1} L_\tau L_c f(z) = L_\tau L_c f(z + c) = L_c f(z + c - \tau(z + c)) = f(z - \tau(z + c))$. This yields the following kernel for the operator T:

$$k(z, u) = h_\sigma(z - u) - h_\sigma(z - \tau(z + c) - u + \tau(u)) \det(I - \nabla\tau(u)). \tag{24}$$

A similar operator appears in Lemma E.1 of Mallat (2012), whose kernel is identical to (24) when $c = 0$.

Like Mallat (2012), we decompose $T = T_1 + T_2$, with kernels

$$\begin{aligned}k_1(z, u) &= h_\sigma(z - u) - h_\sigma((I - \nabla\tau(u))(z - u)) \det(I - \nabla\tau(u)) \\ k_2(z, u) &= \det(I - \nabla\tau(u)) \left(h_\sigma((I - \nabla\tau(u))(z - u)) - h_\sigma(z - \tau(z + c) - u + \tau(u))\right).\end{aligned}$$

The kernel $k_1(z, u)$ appears in (Mallat, 2012), whereas the kernel $k_2(z, u)$ involves a shift c which is not present in (Mallat, 2012). For completeness, we include the proof of the bound for both operators, even though only dealing with k_2 requires slightly new developments.

Bound on $\|T_1\|$. We can write $k_1(z, u) = \sigma^{-d} g(u, (z - u)/\sigma)$ with

$$\begin{aligned}g(u, v) &= h(v) - h((I - \nabla\tau(u))v) \det(I - \nabla\tau(u)) \\ &= (1 - \det(I - \nabla\tau(u)))h((I - \nabla\tau(u))v) + h(v) - h((I - \nabla\tau(u))v).\end{aligned}$$

Using the fundamental theorem of calculus on h, we have

$$h(v) - h((I - \nabla\tau(u))v) = \int_0^1 \langle \nabla h((I + (t - 1)\nabla\tau(u))v), \nabla\tau(u)v \rangle dt.$$

Noticing that

$$|(I + (t - 1)\nabla\tau(u))v| \geq (1 - \|\nabla\tau\|_\infty)|v| \geq (1/2)|v|,$$

and that $\det(I - \nabla\tau(u))) \geq (1 - \|\nabla\tau\|_\infty)^d \geq 1 - d\|\nabla\tau\|_\infty$, we bound each term as follows

$$\begin{aligned}|(1 - \det(I - \nabla\tau(u)))h((I - \nabla\tau(u))v)| &\leq d\|\nabla\tau\|_\infty \frac{C_h}{(1 + \frac{1}{2}|v|)^{d+2}} \\ \left|\int_0^1 \langle \nabla h((I + (t - 1)\nabla\tau(u))v), \nabla\tau(u)v \rangle dt\right| &\leq \|\nabla\tau\|_\infty \frac{C_h' |v|}{(1 + \frac{1}{2}|v|)^{d+2}}.\end{aligned}$$

We thus have

$$|g(u, v)| \leq \|\nabla\tau\|_\infty \frac{C_h d + C_h' |v|}{(1 + \frac{1}{2}|v|)^{d+2}}.$$

Using appropriate changes of variables in order to bound $\int |k_1(z, u)| du$ and $\int |k_1(z, u)| dz$, Schur's test yields

$$\|T_1\| \leq C_1 \|\nabla\tau\|_\infty, \tag{25}$$

with

$$C_1 = \int_{\mathbb{R}^d} \frac{C_h d + C_h' |v|}{(1 + \frac{1}{2}|v|)^{d+2}} dv$$

Bound on $\|T_2\|$. Let $\alpha(z, u) = \tau(z + c) - \tau(u) - \nabla\tau(u)(z - u)$, and note that we have

$$\begin{aligned}|\alpha(z, u)| &\leq |\tau(z + c) - \tau(u)| + |\nabla\tau(u)(z - u)| \\ &\leq \|\nabla\tau\|_\infty |z + c - u| + \|\nabla\tau\|_\infty |z - u| \\ &\leq \|\nabla\tau\|_\infty (|c| + 2|z - u|).\end{aligned} \tag{26}$$

The fundamental theorem of calculus yields

$$k_2(z, u) = -\det(I - \nabla\tau(u)) \int_0^1 \langle \nabla h_\sigma(z - \tau(z + c) - u + \tau(u) - t\alpha(z, u)), \alpha(z, u) \rangle dt.$$

We note that $|\det(I - \nabla\tau(u))| \leq 2^d$, and $\nabla h_\sigma(v) = \sigma^{-d-1} \nabla h(v/\sigma)$. Using the change of variable $z' = (z - u)/\sigma$, we obtain

$$\begin{aligned}&\int |k_2(z, u)| dz \\ &\quad \leq 2^d \int \int_0^1 \left|\nabla h\left(z' + \frac{\tau(u + \sigma z' + c) - \tau(u) - t\alpha(u + \sigma z', u)}{\sigma}\right)\right| \left|\frac{\alpha(u + \sigma z', u)}{\sigma}\right| dt dz'.\end{aligned}$$

We can use the upper bound (26), together with our assumption $|c| \leq \kappa\sigma$:

$$\left|\frac{\alpha(u + \sigma z', u)}{\sigma}\right| \leq \|\nabla\tau\|_\infty (\kappa + 2|z'|). \tag{27}$$

Separately, we have $|\nabla h(v(z'))| \leq C'_h/(1+|v(z')|)^{d+2}$, with

$$v(z') := z' + \frac{\tau(u+\sigma z'+c) - \tau(u) - t\alpha(u+\sigma z', u)}{\sigma}.$$

For $|z'| > 2\kappa$, we have

$$\begin{aligned}\left|\frac{\tau(u+\sigma z'+c) - \tau(u) - t\alpha(u+\sigma z', u)}{\sigma}\right| &= \left|t\nabla\tau(u)z' + (1-t)\frac{\tau(u+\sigma z'+c)-\tau(u)}{\sigma}\right| \\ &\leq t\|\nabla\tau\|_\infty|z'| + (1-t)\|\nabla\tau\|_\infty(|z'|+\kappa) \\ &\leq \frac{3}{2}\|\nabla\tau\|_\infty|z'| \leq \frac{3}{4}|z'|,\end{aligned}$$

and hence, using the reverse triangle inequality, $|v(z')| \geq |z'| - \frac{3}{4}|z'| = \frac{1}{4}|z'|$. This yields the upper bound

$$|\nabla h(v(z'))| \leq \begin{cases} C'_h, & \text{if } |z'| \leq 2\kappa \\ \frac{C'_h}{(1+\frac{1}{4}|z'|)^{d+2}}, & \text{if } |z'| > 2\kappa. \end{cases} \tag{28}$$

Combining these two bounds, we obtain

$$\int |k_2(z,u)|dz \leq C_2\|\nabla\tau\|_\infty,$$

with

$$C_2 := 2^d C'_h\left(\int_{|z'|<2\kappa}(\kappa+2|z'|)dz' + \int_{|z'|>2\kappa}\frac{\kappa+2|z'|}{(1+\frac{1}{4}|z'|)^{d+2}}dz'\right).$$

Note that the dependence of the first integral on κ is of order k^{d+1}. Following the same steps with the change of variable $u' = (z-u)/\sigma$, we obtain the bound $\int|k_2(z,u)|du \leq C_2\|\nabla\tau\|_\infty$. Schur's test then yields

$$\|T_2\| \leq C_2\|\nabla\tau\|_\infty. \tag{29}$$

We have thus proven

$$\|[L_c A_\sigma, L_\tau]\| \leq 2^{d/2}\|T\| \leq 2^{d/2}(C_1+C_2)\|\nabla\tau\|_\infty.$$

■

C.5. Discussion and Proof of Norm Preservation

We now state a result which shows that while the kernel representation may lose some of the energy of the original signal, it preserves a part of it, ensuring that the stability bound in Theorem 7 is non-trivial. We consider in this section the full kernel representation, including a prediction layer, which is given by $\Phi(x) = \varphi_{n+1}(\Phi_n(x))$, where φ_{n+1} is the kernel feature map of either a Gaussian kernel (7) with $\alpha = 1$, or a linear kernel (6). In both cases, φ_{n+1} is non-expansive, which yields

$$\|\Phi(L_\tau x) - \Phi(x)\| \leq \|\Phi_n(L_\tau x) - \Phi_n(x)\|,$$

such that the stability result of Theorem 7 also applies to Φ. For the Gaussian case, we trivially have a representation with norm 1, which trivially shows a preservation of norm, while for the linear case, at least part of the signal energy is preserved, in particular the energy in the low frequencies, which is predominant, for instance, in natural images (Torralba and Oliva, 2003).

Lemma 15 (Norm preservation) *For the two choices of prediction layers, $\Phi(x)$ satisfies*

$$\|\Phi(x)\| = 1 \quad \textit{(Gaussian)}, \qquad \|\Phi(x)\| \geq \|A_n A_{n-1}\dots A_0 x\| \quad \textit{(Linear)}.$$

It follows that the representation Φ is not contractive:

$$\sup_{x,x'\in L^2(\mathbb{R}^d,\mathcal{H}_0)} \frac{\|\Phi(x)-\Phi(x')\|}{\|x-x'\|} = 1. \tag{30}$$

Proof We begin by studying $\|\Phi(x)\|$. The Gaussian case is trivial since the Gaussian kernel mapping φ_{n+1} maps all points to the sphere. In the linear case, we have

$$\begin{aligned}\|\Phi(x)\|^2 = \|\Phi_n(x)\|^2 &= \|A_n M_n P_n x_{n-1}\|^2 \\ &= \int \|A_n M_n P_n x_{n-1}(u)\|^2 du \\ &= \int \langle \int h_{\sigma_n}(u-v) M_n P_n x_{n-1}(v)dv, \int h_{\sigma_n}(u-v') M_n P_n x_{n-1}(v')dv'\rangle du \\ &= \int\int\int h_{\sigma_n}(u-v)h_{\sigma_n}(u-v')\langle\varphi_n(P_n x_{n-1}(v)), \varphi_n(P_n x_{n-1}(v'))\rangle dvdv'du \\ &\geq \int\int\int h_{\sigma_n}(u-v)h_{\sigma_n}(u-v')\langle P_n x_{n-1}(v), P_n x_{n-1}(v')\rangle dvdv'du \\ &= \int \|A_n P_n x_{n-1}(u)\|^2 du = \|A_n P_n x_{n-1}\|^2,\end{aligned}$$

where the inequality follows from $\langle\varphi_n(z), \varphi_n(z')\rangle = K_n(z,z') \geq \langle z, z'\rangle$ (see Lemma 1). Using Fubini's theorem and the fact that A_n commutes with translations, we have

$$\begin{aligned}\|A_n P_n x_{n-1}\|^2 &= \int_{S_n} \|A_n L_v x_{n-1}\|^2 d\nu_n(v) \\ &= \int_{S_n} \|L_v A_n x_{k-1}\|^2 d\nu_n(v) \\ &= \int_{S_n} \|A_n x_{k-1}\|^2 d\nu_n(v) \\ &= \|A_n x_{n-1}\|^2,\end{aligned}$$

where we used the fact that translations L_v preserve the norm. Note that we have

$$A_n x_{n-1} = A_n A_{n-1} M_{n-1} P_{n-1} x_{n-2} = A_{n,n-1} M_{n-1} P_{n-1} x_{n-2},$$

where $A_{n,n-1}$ is an integral operator with positive kernel $h_{\sigma_n} * h_{\sigma_{n-1}}$. Repeating the above relation then yields

$$\|\Phi(x)\|^2 \geq \|A_n x_{n-1}\|^2 \geq \|A_n A_{n-1} x_{n-1}\|^2 \geq \dots \geq \|A_n A_{n-1}\dots A_0 x\|^2,$$

and the result follows.

We now show (30). By our assumptions on φ_{n+1} and on the operators A_k, M_k, P_k, we have that Φ is non-expansive, so that

$$\sup_{x,x' \in L^2(\mathbb{R}^d, \mathcal{H}_0)} \frac{\|\Phi(x) - \Phi(x')\|}{\|x - x'\|} \leq 1.$$

It then suffices to show that one can find x, x' such that the norm ratio $\frac{\|\Phi(x)-\Phi(x')\|}{\|x-x'\|}$ is arbitrarily close to 1. In particular, we begin by showing that for any signal $x \neq 0$ we have

$$\lim_{\lambda \to 1} \frac{\|\Phi(\lambda x) - \Phi(x)\|}{\|\lambda x - x\|} \geq \frac{\|A_\sigma x\|}{\|x\|}, \tag{31}$$

where A_σ is the pooling operator with scale $\sigma = (\sigma_n^2 + \sigma_{n-1}^2 + \ldots + \sigma_1^2)^{1/2}$, and the result will follow by considering appropriate signals x that make this lower bound arbitrarily close to 1.

Note that by homogeneity of the kernels maps φ_k (which follows from the homogeneity of kernels K_k), and by linearity of the operators A_k and P_k, we have $\Phi_n(\lambda x) = \lambda \Phi_n(x)$ for any $\lambda \geq 0$. Taking $\lambda > 0$, we have

$$\|\Phi_n(\lambda x) - \Phi_n(x)\| = (\lambda - 1)\|\Phi_n(x)\| \geq (\lambda - 1)\|A_n A_{n-1} \ldots A_0 x\| = (\lambda - 1)\|A_\sigma x\|,$$

adapting Lemma 15 to the representation Φ_n. Thus,

$$\lim_{\lambda \to 1} \frac{\|\Phi_n(\lambda x) - \Phi_n(x)\|}{\|\lambda x - x\|} \geq \frac{\|A_\sigma x\|}{\|x\|}.$$

When φ_{n+1} is linear, we immediately obtain (31) since $\|\Phi(\lambda x) - \Phi(x)\| = \|\Phi_n(\lambda x) - \Phi_n(x)\|$. For the Gaussian case, we have

$$\begin{aligned}
\|\Phi(\lambda x) - \Phi(x)\|^2 &= 2 - 2e^{-\frac{1}{2}\|\Phi_n(\lambda x) - \Phi_n(x)\|^2} \\
&= 2 - 2e^{-\frac{1}{2}(\lambda-1)^2\|\Phi_n(x)\|^2} \\
&= (\lambda - 1)^2 \|\Phi_n(x)\|^2 + o((\lambda - 1)^2) \\
&= \|\Phi_n(\lambda x) - \Phi_n(x)\|^2 + o((\lambda - 1)^2),
\end{aligned}$$

which yields (31).

By considering a Gaussian signal with scale $\tau \gg \sigma$, we can make $\frac{\|A_\sigma x\|}{\|x\|}$ arbitrarily close to 1 by taking an arbitrarily large τ. It follows that

$$\sup_x \lim_{\lambda \to 1} \frac{\|\Phi(\lambda x) - \Phi(x)\|}{\|\lambda x - x\|} = 1,$$

which yields the result. ■

C.6. Proof of Lemma 9

Proof We have

$$\begin{aligned}
Px((u,\eta)) &= (v \in \tilde{S} \mapsto x((u,\eta)(v,0))) \\
&= (v \in \tilde{S} \mapsto x((u + R_\eta v, \eta))) \\
&= (v \in R_\eta \tilde{S} \mapsto x((u + v, \eta))) \\
Ax((u,\eta)) &= \int_{\mathbb{R}^2} x((u,\eta)(v,0)) h(v) dv \\
&= \int_{\mathbb{R}^2} x((u + R_\eta v, \eta)) h(v) dv \\
&= \int_{\mathbb{R}^2} x((v,\eta)) h(R_{-\eta}(v - u)) dv \\
&= \int_{\mathbb{R}^2} x((v,\eta)) h(u - v) dv,
\end{aligned}$$

where the last equality uses the circular symmetry of a Gaussian around the origin. For a diffeomorphism τ, we denote by L_τ the action operator given by $L_\tau x((u,\eta)) = x((\tau(u), 0)^{-1}(u,\eta)) = x((u - \tau(u), \eta))$. If we denote $x(\cdot, \eta)$ the $L^2(\mathbb{R}^2)$ signal obtained from a signal $x \in L^2(G)$ at a fixed angle, we have shown

$$\begin{aligned}
(Px)(\cdot,\eta) &= \tilde{P}_\eta(x(\cdot,\eta)) \\
(Ax)(\cdot,\eta) &= \tilde{A}(x(\cdot,\eta)) \\
(L_\tau x)(\cdot,\eta) &= \tilde{L}_\tau(x(\cdot,\eta)),
\end{aligned}$$

where $\tilde{P}_\eta, \tilde{A}, \tilde{L}_\tau$ are defined on $L^2(\mathbb{R}^2)$ as in Section 2, with a rotated patch $R_\eta \tilde{S}$ for $\tilde{P}_\eta$. Then, we have, for a signal $x \in L^2(G)$,

$$\begin{aligned}
\|[PA, L_\tau]x\|_{L^2(G)}^2 &= \int \|([PA, L_\tau]x)(\cdot,\eta)\|_{L^2(\mathbb{R}^2)}^2 d\mu_c(\eta) \\
&= \int \|[\tilde{P}_\eta \tilde{A}, \tilde{L}_\tau](x(\cdot,\eta))\|_{L^2(\mathbb{R}^2)}^2 d\mu_c(\eta) \\
&\leq \int \|[\tilde{P}_\eta \tilde{A}, \tilde{L}_\tau]\|^2 \|x(\cdot,\eta)\|_{L^2(\mathbb{R}^2)}^2 d\mu_c(\eta) \\
&\leq \left(\sup_\eta \|[\tilde{P}_\eta \tilde{A}, \tilde{L}_\tau]\|^2 \right) \|x\|_{L^2(G)}^2,
\end{aligned}$$

so that $\|[PA, L_\tau]\|_{L^2(G)} \leq \sup_\eta \|[\tilde{P}_\eta \tilde{A}, \tilde{L}_\tau]\|_{L^2(\mathbb{R}^2)}^2$. Note that we have $\sup_{c \in R_\eta \tilde{S}} |c| = \sup_{c \in \tilde{S}} |c| \leq \kappa\sigma$, since rotations preserve the norm, so that we can bound each $\|[\tilde{P}_\eta \tilde{A}, \tilde{L}_\tau]\|$ as in Section 3.1 to obtain the desired result. Similarly, $\|L_\tau A - A\|$ can be bounded as in Section 3.1. ■

C.7. Proof of Theorem 10

Proof First, note that A_c can be written as an integral operator

$$A_c x(u) = \int x((v,\eta))k(u,(v,\eta))d\mu((v,\eta)),$$

with $k(u,(v,\eta)) = \delta_u(v)$, where δ denotes the Dirac delta function. We have

$$\int |k(u,(v,\eta))|d\mu((v,\eta)) = \int |k(u,(v,\eta))|du = 1.$$

By Schur's test, we thus obtain $\|A_c\| \leq 1$. Then, note that $(\tau(u),\theta) = (0,\theta)(R_{-\theta}\tau(u),0)$, so that $L_{(\tau,\theta)} = L_{(0,\theta)}L_{\tau_\theta}$, where we write $\tau_\theta(u) = R_{-\theta}\tau(u)$. Additionally, it is easy to see that $A_c L_{(0,\theta)} = A_c$. We have

$$\begin{aligned}\|A_c\Phi_n(L_{(\tau,\theta)}x) - A_c\Phi_n(x)\| &= \|A_c\Phi_n(L_{(0,\theta)}L_{\tau_\theta}x) - A_c\Phi_n(x)\| \\ &= \|A_c L_{(0,\theta)}\Phi_n(L_{\tau_\theta}x) - A_c\Phi_n(x)\| \\ &= \|A_c\Phi_n(L_{\tau_\theta}x) - A_c\Phi_n(x)\| \\ &\leq \|\Phi_n(L_{\tau_\theta}x) - \Phi_n(x)\|,\end{aligned}$$

using the fact that the representation Φ_n is equivariant to roto-translations by construction.

We conclude by using Lemma 9 together with an adapted version of Proposition 4, and by noticing that $\|\nabla\tau_\theta\|_\infty = \|\nabla\tau\|_\infty$ and $\|\tau_\theta\|_\infty = \|\tau\|_\infty$. ■

Appendix D. Proofs Related to the Construction of CNNs in the RKHS

D.1. Proof of Lemma 11

Proof Here, we drop all indices k since there is no ambiguity. We will now characterize the functional space $\mathcal{H}$ by following the same strategy as Zhang et al. (2016, 2017b) for the non-homogeneous Gaussian and inverse polynomial kernels on Euclidean spaces. Using the Maclaurin expansion of κ, we can define the following explicit feature map for the dot-product kernel $K_{\text{dp}}(z,z') := \kappa(\langle z,z'\rangle)$, for any z in the unit-ball of $\mathcal{P}$:

$$\begin{aligned}\psi_{\text{dp}}(z) &= \left(\sqrt{b_0}, \sqrt{b_1}z, \sqrt{b_2}z\otimes z, \sqrt{b_3}z\otimes z\otimes z, \ldots\right) \\ &= \left(\sqrt{b_j}z^{\otimes j}\right)_{j\in\mathbb{N}},\end{aligned} \tag{32}$$

where $z^{\otimes j}$ denotes the tensor product of order j of the vector z. Technically, the explicit mapping lives in the Hilbert space $\oplus_{j=0}^n \otimes^j \mathcal{P}$, where $\oplus$ denotes the direct sum of Hilbert spaces, and with the abuse of notation that $\otimes^0\mathcal{P}$ is simply $\mathbb{R}$. Then, we have that $K_{\text{dp}}(z,z') = \langle\psi(z),\psi(z')\rangle$ for all z,z' in the unit ball of $\mathcal{P}$. Similarly, we can construct an explicit feature map for the homogeneous dot-product kernels (2):

$$\begin{aligned}\psi_{\text{hdp}}(z) &= \left(\sqrt{b_0}\|z\|, \sqrt{b_1}z, \sqrt{b_2}\|z\|^{-1}z\otimes z, \sqrt{b_3}\|z\|^{-2}z\otimes z\otimes z, \ldots\right) \\ &= \left(\sqrt{b_j}\|z\|^{1-j}z^{\otimes j}\right)_{j\in\mathbb{N}}.\end{aligned} \tag{33}$$

From these mappings, we may now conclude the proof by following the same strategy as Zhang et al. (2016, 2017b). By first considering the restriction of K to unit-norm vectors z,

$$\sigma(\langle w,z\rangle) = \sum_{j=0}^{+\infty} a_j\langle w,z\rangle^j = \sum_{j=0}^{+\infty} a_j\langle w^{\otimes j}, z^{\otimes j}\rangle = \langle\bar{w},\psi(z)\rangle,$$

where

$$\bar{w} = \left(\frac{a_j}{\sqrt{b_j}}w^{\otimes j}\right)_{j\in\mathbb{N}}.$$

Then, the norm of $\bar{w}$ is

$$\|\bar{w}\|^2 = \sum_{j=0}^{+\infty}\frac{a_j^2}{b_j}\|w^{\otimes j}\|^2 = \sum_{j=0}^{+\infty}\frac{a_j^2}{b_j}\|w\|^{2j} = C_\sigma^2(\|w\|^2) < +\infty.$$

Using Theorem A.1, we conclude that f is in the RKHS of K, with norm $\|f\| \leq C_\sigma(\|w\|^2)$. Finally, we extend the result to non unit-norm vectors z with similar calculations and we obtain the desired result. ■

D.2. CNN construction and RKHS norm

In this section, we describe the space of functions (RKHS) $\mathcal{H}_{\mathcal{K}_n}$ associated to the kernel $\mathcal{K}_n(x_0,x_0') = \langle x_n, x_n'\rangle$ defined in (6), where x_n, x_n' are the final representations given by Eq. (5), in particular showing it contains the set of CNNs with activations described in Section 4.1.

D.2.1. Construction of a CNN in the RKHS.

Let us consider the definition of the CNN presented in Section 4. We will show that it can be seen as a point in the RKHS of $\mathcal{K}_n$. According to Lemma 11, we consider $\mathcal{H}_k$ that contains all functions of the form $z\in\mathcal{P}_k \mapsto \|z\|\sigma(\langle w,z\rangle/\|z\|)$, with $w\in\mathcal{P}_k$.

We recall the intermediate quantities introduced in Section 4. That is, we define the initial quantities $f_1^i \in \mathcal{H}_1, g_1^i \in \mathcal{P}_1$ for $i = 1,\ldots,p_1$ such that

$$\begin{aligned}g_1^i &= w_1^i \in L^2(S_1,\mathbb{R}^{p_0}) = L^2(S_1,\mathcal{H}_0) = \mathcal{P}_1 \\ f_1^i(z) &= \|z\|\sigma(\langle g_i^0, z\rangle/\|z\|) \quad \text{for } z\in\mathcal{P}_1,\end{aligned}$$

and we recursively define, from layer k–1, the quantities $f_k^i \in \mathcal{H}_k, g_k^i \in \mathcal{P}_k$ for $i = 1,\ldots,p_k$:

$$\begin{aligned}g_k^i(v) &= \sum_{j=1}^{p_{k-1}} w_k^{ij}(v)f_{k-1}^j \quad \text{where} \quad w_k^i(v) = (w_k^{ij}(v))_{j=1,\ldots,p_{k-1}} \\ f_k^i(z) &= \|z\|\sigma(\langle g_k^i, z\rangle/\|z\|) \quad \text{for } z\in\mathcal{P}_k.\end{aligned}$$

Then, we will show that $\tilde{z}_k^i(u) = f_k^i(P_k x_{k-1}(u)) = \langle f_k^i, M_k P_k x_{k-1}(u)\rangle$, which correspond to feature maps at layer k and index i in a CNN. Indeed, this is easy to see for $k=1$ by

construction with filters $w_1^i(v)$, and for $k \geq 2$, we have

$$\begin{aligned}
\tilde{z}_k^i(u) &= n_k(u)\sigma\big(\langle w_k^i, P_k z_{k-1}(u)\rangle / n_k(u)\big) \\
&= n_k(u)\sigma\big(\langle w_k^i, P_k A_{k-1}\tilde{z}_{k-1}(u)\rangle / n_k(u)\big) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\sum_{j=1}^{p_{k-1}}\int_{S_k} w_k^{ij}(v) A_{k-1}\tilde{z}_{k-1}^j(u+v)d\nu_k(v)\right) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\sum_{j=1}^{p_{k-1}}\int_{S_k} w_k^{ij}(v)\langle f_{k-1}^j, A_{k-1}M_{k-1}P_{k-1}x_{k-2}(u+v)\rangle d\nu_k(v)\right) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\int_{S_k}\langle g_k^i(v), A_{k-1}M_{k-1}P_{k-1}x_{k-2}(u+v)\rangle d\nu_k(v)\right) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\int_{S_k}\langle g_k^i(v), x_{k-1}(u+v)\rangle d\nu_k(v)\right) \\
&= n_k(u)\sigma\left(\frac{1}{n_k(u)}\langle g_k^i(v), P_k x_{k-1}(u)\rangle\right) \\
&= f_k^i(P_k x_{k-1}(u)),
\end{aligned}$$

where $n_k(u) := \|P_k x_{k-1}(u)\|$. Note that we have used many times the fact that A_k operates on each channel independently when applied to a finite-dimensional map.

The final prediction function is of the form $f_\sigma(x_0) = \langle w_{n+1}, z_n\rangle$ with w_{n+1} in $L^2(\mathbb{R}^d, \mathbb{R}^{p_n})$. Then, we can define the following function g_σ in $L^2(\mathbb{R}^d, \mathcal{H}_n)$ such that

$$g_\sigma(u) = \sum_{j=1}^{p_n} w_{n+1}^j(u) f_n^j,$$

which yields

$$\begin{aligned}
\langle g_\sigma, x_n\rangle &= \sum_{j=1}^{p_n}\int_{\mathbb{R}^d} w_{n+1}^j(u)\langle f_n^j, x_n(u)\rangle du \\
&= \sum_{j=1}^{p_n}\int_{\mathbb{R}^d} w_{n+1}^j(u)\langle f_n^j, A_n M_n P_n x_{n-1}(u)\rangle du \\
&= \sum_{j=1}^{p_n}\int_{\mathbb{R}^d} w_{n+1}^j(u) A_n \tilde{z}_n^j(u) du \\
&= \sum_{j=1}^{p_n}\int_{\mathbb{R}^d} w_{n+1}^j(u) z_n^j(u) du \\
&= \sum_{j=1}^{p_n}\langle w_{n+1}^j, z_n^j\rangle = f_\sigma(x_0),
\end{aligned}$$

which corresponds to a linear layer after pooling. Since the RKHS of $\mathcal{K}_n$ in the linear case (6) contains all functions of the form $f(x_0) = \langle g, x_n\rangle$, for g in $L^2(\mathbb{R}^d, \mathcal{H}_n)$, we have that f_σ is in the RKHS.

D.2.2. Proof of Proposition 13

Proof As shown in Lemma 11, the RKHS norm of a function $f : z \in \mathcal{P}_k \mapsto \|z\|\sigma(\langle w, z\rangle/\|z\|)$ in $\mathcal{H}_k$ is bounded by $C_\sigma(\|w\|^2)$, where C_σ depends on the activation σ. We then have

$$\begin{aligned}
&\|f_1^i\|^2 \leq C_\sigma^2(\|w_1^i\|_2^2) \quad \text{where} \quad \|w_1^i\|_2^2 = \int_{S_1}\|w_1^i(v)\|^2 d\nu_1(v) \\
&\|f_k^i\|^2 \leq C_\sigma^2(\|g_k^i\|^2) \\
&\|g_k^i\|^2 = \int_{S_k}\|\sum_{j=1}^{p_{k-1}} w_k^{ij}(v) f_{k-1}^j\|^2 d\nu_k(v) \\
&\qquad \leq p_{k-1}\sum_{j=1}^{p_{k-1}}\left(\int_{S_k}|w_k^{ij}(v)|^2 d\nu_k(v)\right)\|f_{k-1}^j\|^2 \\
&\qquad = p_{k-1}\sum_{j=1}^{p_{k-1}}\|w_k^{ij}\|_2^2\|f_{k-1}^j\|^2,
\end{aligned}$$

where in the last inequality we use $\|a_1 + \ldots + a_n\|^2 \leq n(\|a_1\|^2 + \ldots + \|a_n\|^2)$. Since C_σ^2 is monotonically increasing (typically exponentially in its argument), we have for $k = 1, \ldots, n-1$ the recursive relation

$$\|f_k^i\|^2 \leq C_\sigma^2\left(p_{k-1}\sum_{j=1}^{p_{k-1}}\|w_k^{ij}\|_2^2\|f_{k-1}^j\|^2\right).$$

The norm of the final prediction function $f \in L^2(\mathbb{R}^d, \mathcal{H}_n)$ is bounded as follows, using similar arguments as well as Theorem A.1:

$$\|f_\sigma\|^2 \leq \|g_\sigma\|^2 \leq p_n\sum_{j=1}^{p_n}\left(\int_{\mathbb{R}^d}|w_{n+1}^j(u)|^2 du\right)\|f_n^j\|^2.$$

This yields the desired result. ∎

D.2.3. Proof of Proposition 14

Proof Define

$$\begin{aligned}
F_k &= (f_k^1, \ldots, f_k^{p_k}) \in \mathcal{H}_k^{p_k} \\
G_k &= (g_k^1, \ldots, g_k^{p_k}) \in \mathcal{P}_k^{p_k} \\
W_k(u) &= (w_k^{ij}(u))_{ij} \in \mathbb{R}^{p_k \times p_{k-1}} \quad \text{for } u \in S_k.
\end{aligned}$$

We will write, by abuse of notation, $G_k(u) = (g_k^1(u), \ldots, g_k^{p_k}(u))$ for $u \in S_k$, so that we can write $G_k(u) = W_k(u)F_{k-1}$. In particular, we have $\|G_k(u)\| \leq \|W_k(u)\|_2\|F_{k-1}\|$. This can be seen by considering an orthonormal basis B of $\mathcal{H}_k$, and defining real-valued vectors $F_k^w = (\langle w, f_k^1\rangle, \ldots, \langle w, f_k^{p_k}\rangle)$, $G_k^w(u) = (\langle w, g_k^1(u)\rangle, \ldots, \langle w, g_k^{p_k}(u)\rangle)$ for $w \in B$. Indeed, we

have $G_k^w(u) = W_k(u)F_{k-1}^w$ and hence $\|G_k^w(u)\| \leq \|W_k(u)\|_2 \|F_{k-1}^w\|$ for all $w \in B$, and we conclude using

$$\|G_k(u)\|^2 = \sum_{w \in B} \|G_k^w(u)\|^2 \leq \|W_k(u)\|_2^2 \sum_{w \in B} \|F_{k-1}^w\|^2 = \|W_k(u)\|_2^2 \, \|F_{k-1}\|^2.$$

Then, we have

$$\begin{aligned}\|G_k\|^2 &= \sum_i \|g_k^i\|^2 = \sum_i \int_{S_k} \|g_k^i(u)\|^2 d\nu_k(u) = \int_{S_k} \|G_k(u)\|^2 d\nu_k(u) \\ &\leq \int_{S_k} \|W_k(u)\|_2^2 \, \|F_{k-1}\|^2 \nu_k(u) = \|W_k\|_2^2 \, \|F_{k-1}\|^2.\end{aligned}$$

Separately, we notice that C_σ^2 is super-additive, *i.e.*,

$$C_\sigma^2(\lambda_1^2 + \ldots + \lambda_n^2) \geq C_\sigma^2(\lambda_1^2) + \ldots + C_\sigma^2(\lambda_n^2).$$

Indeed, this follows from the definition of C_σ^2, noting that polynomials with non-negative coefficients are super-additive on non-negative numbers. Thus, we have

$$\begin{aligned}\|F_1\|^2 &= \sum_{i=1}^{p_1} \|f_1^i\|^2 \leq \sum_{i=1}^{p_1} C_\sigma^2(\|w_1^i\|^2) \leq C_\sigma^2(\|W_1\|_F^2) \\ \|F_k\|^2 &\leq \sum_{i=1}^{p_k} C_\sigma^2(\|g_k^i\|^2) \leq C_\sigma^2(\|G_k\|^2), \qquad \text{for } k = 2, \ldots, n.\end{aligned}$$

Finally, note that

$$\|g_\sigma(u)\|^2 \leq \left(\sum_{j=1}^{p_n} |w_{n+1}^j(u)| \|f_n^j\| \right)^2 \leq \|w_{n+1}(u)\|^2 \|F_n\|^2,$$

by using Cauchy-Schwarz, so that $\|g_\sigma\|^2 \leq \|w_{n+1}\|^2 \|F_n\|^2$. Thus, combining the previous relations yields

$$\|f_\sigma\|^2 \leq \|g_\sigma\|^2 \leq \|w_{n+1}\|^2 \, C_\sigma^2(\|W_n\|_2^2 \, C_\sigma^2(\|W_{n-1}\|_2^2 \ldots C_\sigma^2(\|W_1\|_F^2) \ldots)),$$

which is the desired result. ■

References

S. Allassonnière, Y. Amit, and A. Trouvé. Towards a coherent statistical framework for dense deformable template estimation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(1):3–29, 2007.

F. Anselmi, L. Rosasco, C. Tan, and T. Poggio. Deep convolutional networks are hierarchical kernel machines. *preprint arXiv:1508.01084*, 2015.

F. Anselmi, L. Rosasco, and T. Poggio. On invariance and selectivity in representation learning. *Information and Inference*, 5(2):134–158, 2016.

S. Arora, R. Ge, B. Neyshabur, and Y. Zhang. Stronger generalization bounds for deep nets via a compression approach. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

F. Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research (JMLR)*, 18:1–38, 2017.

P. Bartlett, D. J. Foster, and M. Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

A. Bietti and J. Mairal. Invariance and stability of deep convolutional representations. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

A. Bietti, G. Mialon, D. Chen, and J. Mairal. A kernel perspective for regularizing deep neural networks. *preprint arXiv:1810.00363*, 2019.

L. Bo, K. Lai, X. Ren, and D. Fox. Object recognition with hierarchical kernel descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: A survey of some recent advances. *ESAIM: probability and statistics*, 9:323–375, 2005.

J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on pattern analysis and machine intelligence (PAMI)*, 35(8):1872–1886, 2013.

J. Bruna, A. Szlam, and Y. LeCun. Learning stable group invariant representations with convolutional networks. *preprint arXiv:1301.3537*, 2013.

Y. Cho and L. K. Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2017.

T. Cohen and M. Welling. Group equivariant convolutional networks. In *International Conference on Machine Learning (ICML)*, 2016.

A. Daniely, R. Frostig, and Y. Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

A. Daniely, R. Frostig, V. Gupta, and Y. Singer. Random features for compositional kernels. *preprint arXiv:1703.07872*, 2017.

J. Diestel and J. J. Uhl. *Vector Measures*. American Mathematical Society, 1977.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research (JMLR)*, 2:243–264, 2001.

G. B. Folland. *A course in abstract harmonic analysis*. Chapman and Hall/CRC, 2016.

B. Haasdonk and H. Burkhardt. Invariant kernel functions for pattern analysis and machine learning. *Machine learning*, 68(1):35–61, 2007.

R. Kondor and S. Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

T. Liang, T. Poggio, A. Rakhlin, and J. Stokes. Fisher-Rao metric, geometry, and complexity of neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

G. Loosli, S. Canu, and L. Bottou. Training invariant support vector machines using selective sampling. In *Large Scale Kernel Machines*, pages 301–320. MIT Press, Cambridge, MA., 2007.

J. Mairal. End-to-End Kernel Learning with Supervised Convolutional Kernel Networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid. Convolutional kernel networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

S. Mallat. Group invariant scattering. *Communications on Pure and Applied Mathematics*, 65(10):1331–1398, 2012.

G. Montavon, M. L. Braun, and K.-R. Müller. Kernel analysis of deep networks. *Journal of Machine Learning Research (JMLR)*, 12:2563–2581, 2011.

Y. Mroueh, S. Voinea, and T. A. Poggio. Learning with group invariant features: A kernel perspective. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

K. Muandet, K. Fukumizu, B. Sriperumbudur, B. Schölkopf, et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends in Machine Learning*, 10 (1-2):1–141, 2017.

B. Neyshabur, R. Tomioka, and N. Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory (COLT)*, 2015.

B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. A PAC-Bayesian approach to spectrally-normalized margin bounds for neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

E. Oyallon and S. Mallat. Deep roto-translation scattering for object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

A. Raj, A. Kumar, Y. Mroueh, T. Fletcher, and B. Schoelkopf. Local group invariant representations via orbit embeddings. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

S. Saitoh. *Integral transforms, reproducing kernels and their applications*, volume 369. CRC Press, 1997.

I. J. Schoenberg. Positive definite functions on spheres. *Duke Mathematical Journal*, 9(1): 96–108, 1942.

B. Schölkopf. *Support Vector Learning*. PhD thesis, Technischen Universität Berlin, 1997.

B. Schölkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. 2001.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

L. Sifre and S. Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2013.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2014.

A. J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2000.

E. M. Stein. *Harmonic Analysis: Real-variable Methods, Orthogonality, and Oscillatory Integrals*. Princeton University Press, 1993.

I. Steinwart, P. Thomann, and N. Schmid. Learning with hierarchical gaussian kernels. *preprint arXiv:1612.00824*, 2016.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

A. Torralba and A. Oliva. Statistics of natural image categories. *Network: computation in neural systems*, 14(3):391–412, 2003.

A. Trouvé and L. Younes. Local geometry of deformable templates. *SIAM journal on mathematical analysis*, 37(1):17–59, 2005.

T. Wiatowski and H. Bölcskei. A mathematical theory of deep convolutional neural networks for feature extraction. *IEEE Transactions on Information Theory*, 64(3):1845–1866, 2018.

C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*, 2017a.

Y. Zhang, J. D. Lee, and M. I. Jordan. ℓ_1-regularized neural networks are improperly learnable in polynomial time. In *International Conference on Machine Learning (ICML)*, 2016.

Y. Zhang, P. Liang, and M. J. Wainwright. Convexified convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2017b.

Journal of Machine Learning Research 20 (2019) 1-6 Submitted 5/18; Revised 10/18; Published 2/19

TensorLy: Tensor Learning in Python

Jean Kossaifi[1] jean.kossaifi@imperial.ac.uk
Yannis Panagakis[1,2] i.panagakis@imperial.ac.uk
Anima Anandkumar[3,4] anima@caltech.edu
Maja Pantic[1] m.pantic@imperial.ac.uk
[1]*Imperial College London* [2]*Middlesex University*
[3] *NVIDIA* [4] *California Institute of Technology*

Editor: Alexandre Gramfort

Abstract

Tensors are higher-order extensions of matrices. While matrix methods form the cornerstone of traditional machine learning and data analysis, tensor methods have been gaining increasing traction. However, software support for tensor operations is not on the same footing. In order to bridge this gap, we have developed *TensorLy*, a Python library that provides a high-level API for tensor methods and deep tensorized neural networks. TensorLy aims to follow the same standards adopted by the main projects of the Python scientific community, and to seamlessly integrate with them. Its BSD license makes it suitable for both academic and commercial applications. TensorLy's backend system allows users to perform computations with several libraries such as NumPy or PyTorch to name but a few. They can be scaled on multiple CPU or GPU machines. In addition, using the deep-learning frameworks as backend allows to easily design and train deep tensorized neural networks. TensorLy is available at `https://github.com/tensorly/tensorly`

1. Introduction

Tensors are higher-order extensions of matrices. While matrices are indexed by two indices (and hence, second order tensors), tensors can be indexed by an arbitrary number of indices. Tensors have a rich history, stretching almost a century, and have been used in diverse fields such as psychometrics, quantum systems, signal processing etc. Only recently have tensors been employed in machine learning and data analytics. This can be attributed to the availability of large-scale multi-dimensional and multi-modal datasets. Tensors form the natural framework to encode and operate on such data structures.

For multi-dimensional datasets, there are natural extensions of traditional dimensionality-reduction methods, such as principal component analysis (PCA), to higher dimensions. These involve tensor decompositions and have been applied in a number of areas, including the theoretical analysis of deep neural nets (Cohen et al., 2015). Another line of work aims to operate on higher-order moments of the data distribution, i.e., beyond pairwise correlations. This is shown to be fruitful for learning a wide range of probabilistic latent-variable models such as Gaussian mixtures, Independent Component Analysis, topic models etc (Anandkumar et al., 2014). More recently, deep tensorized neural networks have been explored for a variety of applications. They extend linear-algebraic operations in various layers to tensor algebraic operations, such as tensor contraction and tensor regression, in convolutional architectures (Novikov et al., 2015; Kossaifi et al., 2017), generative adversarial networks (Cao and Zhao, 2017) and sequence models (Yu et al., 2017).

Interested readers are referred to several surveys on this topic. Some works focus on the basics of multi-linear (tensor) algebra and different types of tensor decompositions, (Kolda and Bader, 2009; Sidiropoulos et al., 2016). Others focus on algorithmic advances, (Cichocki et al., 2009; Lu et al., 2011; Grasedyck et al., 2013; Cichocki et al., 2015; Papalexakis et al., 2016). Recent surveys focus on their applications (Acar and Yener, 2009) and uses in learning probabilistic models (Janzamin et al., 2019).

Thus, tensor methods can have a profound impact on data analytics and machine learning with clear theoretical, algorithmic, and practical advantages over their matrix counterparts. However, as opposed to matrix methods, tensor methods have not yet been widely adopted by data scientists. This can be mainly attributed to the lack of available software libraries for tensor operations and decompositions, in programming languages that data scientists and practitioners are familiar with (e.g., Python, Java, Scala, etc).

Even though some tensor libraries exist, they are implemented in non-free platforms (e.g., MATLAB's TensorToolbox, (Bader and Kolda, 2015) and TensorLab, (Vervliet et al., 2016)) or in low-level languages like C++ (e.g., TH++). Python is emerging as a language of choice for machine learning, as witnessed with the success of scikit-learn (Pedregosa et al., 2011), and is increasingly used in both academic and industrial research projects. However, there is not yet a Python library implementing tensor operations, decomposition, and learning. The existing ones (e.g., scikit-tensor) offer only limited functionalities and/or have restrictive licenses. Moreover, widely-adopted deep-learning libraries such as Tensorflow and Torch lack advanced tensor operations. For applications to data analytics, machine learning, and deep learning, there is an urgent need for well-developed and documented open-source libraries that include methods for tensor decompositions.

In this paper, to address the aforementioned need, the *TensorLy* [1] library is introduced, allowing several contributions over the existing libraries for tensor methods. In particular, TensorLy a) provides state-of-the-art tensor learning, including core tensor operations and algebra, tensor decomposition and tensor regression methods; b) has a flexible *backend* system that allows switching between NumPy, MXNet, PyTorch, TensorFlow, and CuPy to perform the computation, and to easily combine tensor methods with deep learning; c) is open source and BSD licensed; d) depends by default exclusively on NumPy and SciPy; and e) is accompanied by extensive tests and documentation.

2. *TensorLy*: functionalities and implementation

TensorLy has been developed with the goal of making tensor learning more accessible and to allow for seamless integration with the Python scientific environment. It depends by default only on Numpy (van der Walt et al., 2011) and Scipy, with a soft dependency on Matplotlib (Hunter, 2007) for plotting.

To enable algorithms to be easily run at scale, on various hardware, and to combine tensor methods with deep learning, TensorLy has a flexible backend system. This allows tensor operations to be ran using NumPy, MXNet (Chen et al., 2015), PyTorch (Paszke

1. *TensorLy*'s source code available at `https://github.com/tensorly/tensorly` and documentation at `http://tensorly.org`.

©2019 Jean Kossaifi, Yannis Panagakis, Anima Anandkumar and Maja Pantic.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v20/18-277.html`.

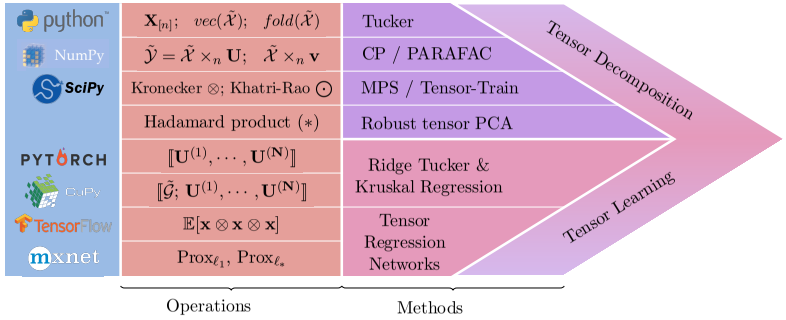


Figure 1: TensorLy builds on top of the Python ecosystem and implements Tensor Algebra *Operations*. These tensor operators are then used for higher level *Methods* such as tensor regression and decomposition, or combined with deep learning.

et al., 2017), Chainer's CuPy (Tokui et al., 2015) or TensorFlow with eager execution (Abadi et al., 2015). **NumPy** is the standard library for numerical computation in Python. It offers high performance structures for manipulating multi-dimensional arrays. **CuPy** works as a drop-in replacement for NumPy with GPU support. **MXNet** is a flexible deep learning library with an NDArray structure that allows to efficiently run code on CPU, GPU and multi-machines. Similarly, **PyTorch** provides a NumPy-like API with GPU acceleration and auto-grad using dynamical graphs. **TensorFlow** is an established machine-learning framework that provides an imperative approach using eager execution.

We aim at making the *API* simple and efficient, following that of scikit-learn (Buitinck et al., 2013), where possible. However, while scikit-learn works with observations (samples) represented as vectors, this library focuses on higher order arrays. *TensorLy*'s main functionalities in term of tensor operations are summarized in Fig. 2, where in the operations column the mathematical notation of Kolda and Bader (2009) is adopted. In the methods column, we summarize the implemented well-known tensor decomposition and regression methods. These include CP and Tucker decompositions, their non-negative versions, matrix-product-state (also known as tensor-train decomposition), Robust tensor PCA, (Goldfarb and Qin, 2014) and low-rank tensor (Kruskal and Tucker) Regression. Additionally, using the deep learning backends, it is easy to combine tensor methods and Deep Learning.

We emphasize code quality and ease of utilization for the end user. To that extent, both testing and documentation are an essential part of the package: all functions come with documentation and unit-tests (at the time of writing, the coverage is of 99%).

3. Performance

TensorLy has been tailored for the Python ecosystem: tensors are multi-dimensional arrays which are effectively manipulated directly by the various methods, decomposition or regression. This allows for competitive performance even though the library is implemented in a high-level, interactive language. The operations are also optimized for speed: tensor operations have been redefined when possible to allow for better performance. In particular,

we propose an efficient unfolding of tensors which differs from the traditional one (Kolda and Bader, 2009) by the ordering of the fibers.

Given a tensor, $\tilde{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the mode-$n$ unfolding of $\tilde{\mathcal{X}}$ is a matrix $\mathbf{X}_{[n]} \in \mathbb{R}^{I_n, I_M}$, and is defined by the mapping from element $(i_1, i_2, \cdots, i_N)$ to (i_n, j),

$$\text{with } j = \sum_{\substack{k=1,\\ k \neq n}}^{N} i_k \times \prod_{\substack{m=k+1,\\ k \neq n}}^{N} I_m, \quad \text{and } M = \prod_{\substack{k=1,\\ k \neq n}}^{N} I_k$$

This formulation both achieves better performance when using C-ordering of the elements (Numpy and most Python libraries' default), and translates into more natural properties.

3.1. Numerical test

We generated random third order tensors of size $500 \times 500 \times 500$ (125 million elements). We then compared the decomposition speed for a rank–50 CANDECOMP-PARAFAC (CP) and rank $(50, 50, 50)$–Tucker decomposition with TensorLy on CPU (NumPy backend) and TensorLy on GPU (MXNet, PyTorch, TensorFlow and CuPy backends), and Scikit-Tensor (*Sktensor*), Fig. 2. In all cases we fixed the number of iterations to 100 to allow for a fair comparison. The experiment was repeated 10 times, with the main bar representing the average CPU time and the tip on the bar the standard deviation of the runs. As can be observed, our library offers competitive speed, thanks to optimized formulation and implementation. Experiments were done on an Amazon Web Services p3 instance, with a NVIDIA VOLTA V100 GPU and 8 Intel Xeon E5 (Broadwell) processors.

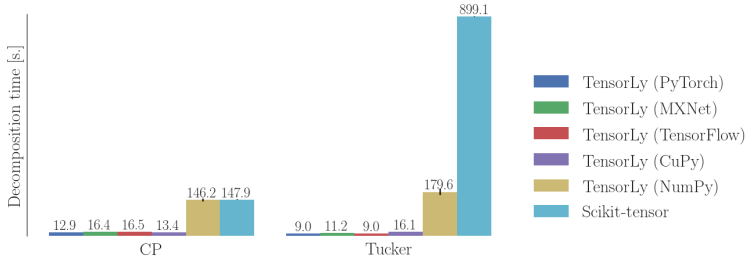


Figure 2: Speed comparison for Tucker and CANDECOMP-PARAFAC decomposition.

4. Conclusion

TensorLy makes tensor learning accessible and straightforward by offering state-of-the-art tensor methods and operations through simple and consistent interfaces, under a permissive license. It is optimized to be fast and robust, with systematic unit-tests and documentation. The library's speed and ease of use allow for an efficient comparison of existing methods and easy implementation of new ones. Its flexible backend system allows to transparently switch between libraries and platforms and help combine tensor methods with deep learning. Going forward, we will further extend the available methods with other state-of-the-art methods and target further improvements in performance, as exemplified by Shi et al. (2016).

References

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

E. Acar and B. Yener. Unsupervised multiway data analysis: A literature survey. *IEEE Transactions on Knowledge and Data Engineering*, 21(1):6–20, Jan 2009.

A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky. Tensor decompositions for learning latent variable models. *JMLR*, 15(1):2773–2832, jan 2014.

B. W. Bader and T. G. Kolda. Matlab tensor toolbox version 2.6. Available online, February 2015.

L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

X. Cao and Q. Zhao. Tensorizing generative adversarial nets. *CoRR*, abs/1710.10772, 2017.

T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *CoRR*, abs/1512.01274, 2015.

A. Cichocki, R. Zdunek, A. H. Phan, and S.-I. Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation*. John Wiley & Sons, Ltd, 2009.

A. Cichocki, D. Mandic, L. D. Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. PHAN. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, 32(2):145–163, March 2015.

N. Cohen, O. Sharir, and A. Shashua. On the expressive power of deep learning: A tensor analysis. *CoRR*, abs/1509.05009, 2015.

D. Goldfarb and Z. T. Qin. Robust low-rank tensor recovery: Models and algorithms. *SIAM Journal on Matrix Analysis and Applications*, 35(1):225–253, 2014.

L. Grasedyck, D. Kressner, and C. Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitteilungen*, 36(1):53–78, 2013.

J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science Engineering*, 9(3): 90–95, May 2007.

M. Janzamin, R. Ge, J. Kossaifi, and A. Anandkumar. Spectral learning on matrices and tensors. *pre-print*, 2019.

T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM REVIEW*, 51(3): 455–500, 2009.

J. Kossaifi, Z. C. Lipton, A. Khanna, T. Furlanello, and A. Anandkumar. Tensor regression networks. *CoRR*, abs/1707.08308, 2017.

H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. A survey of multilinear subspace learning for tensor data. *Pattern Recognition*, 44(7):1540 – 1551, 2011.

A. Novikov, D. Podoprikhin, A. Osokin, and D. Vetrov. Tensorizing neural networks. In *NIPS*, pages 442–450, 2015.

E. E. Papalexakis, C. Faloutsos, and N. D. Sidiropoulos. Tensors for data mining and data fusion: Models, applications, and scalable algorithms. *ACM Trans. Intell. Syst. Technol.*, 8(2):16:1–16:44, Oct. 2016.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Y. Shi, U. N. Niranjan, A. Anandkumar, and C. Cecka. Tensor contractions with extended blas kernels on cpu and gpu. In *IEEE HiPC*, pages 193–202, Dec 2016.

N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos. tensor decomposition for signal processing and machine learning. *arXiv preprint arXiv:1607.01668*, 2016.

S. Tokui, K. Oono, S. Hido, and J. Clayton. Chainer: a next-generation open source framework for deep learning. In *LearningSys Workshop in NIPS*, 2015.

S. van der Walt, S. C. Colbert, and G. Varoquaux. The numpy array: A structure for efficient numerical computation. *Computing in Science Engineering*, 13(2):22–30, March 2011.

N. Vervliet, O. Debals, L. Sorber, M. Van Barel, and L. De Lathauwer. Tensorlab 3.0, Mar. 2016. URL `https://www.tensorlab.net`. Available online.

R. Yu, S. Zheng, A. Anandkumar, and Y. Yue. Long-term forecasting using tensor-train rnns. *CoRR*, abs/1711.00073, 2017.

Journal of Machine Learning Research 20 (2019) 1-42 Submitted 5/18; Revised 11/18; Published 1/19

Monotone Learning with Rectified Wire Networks

Veit Elser VE10@CORNELL.EDU
Department of Physics
Cornell University
Ithaca, NY 14853-2501, USA

Dan Schmidt DANIEL.SCHMIDT@ANALOG.COM
and **Jonathan Yedidia** JONATHAN.YEDIDIA@ANALOG.COM
Analog Devices, Inc.
Boston, MA, USA

Editor: Samy Bengio

Abstract

We introduce a new neural network model, together with a tractable and monotone online learning algorithm. Our model describes feed-forward networks for classification, with one output node for each class. The only nonlinear operation is rectification using a ReLU function with a bias. However, there is a rectifier on every edge rather than at the nodes of the network. There are also weights, but these are positive, static, and associated with the nodes. Our *rectified wire networks* are able to represent arbitrary Boolean functions. Only the bias parameters, on the edges of the network, are learned. Another departure in our approach, from standard neural networks, is that the loss function is replaced by a constraint. This constraint is simply that the value of the output node associated with the correct class should be zero. Our model has the property that the exact norm-minimizing parameter update, required to correctly classify a training item, is the solution to a quadratic program that can be computed with a few passes through the network. We demonstrate a training algorithm using this update, called sequential deactivation (SDA), on MNIST and some synthetic datasets. Upon adopting a natural choice for the nodal weights, SDA has no hyperparameters other than those describing the network structure. Our experiments explore behavior with respect to network size and depth in a family of sparse expander networks.

Keywords: Neural Networks, Online Training Algorithms, Rectified Linear Unit

Some of the simplest modes of machine learning are monotone in character, where the representation of knowledge changes unidirectionally. In the k-nearest-neighbor classification algorithm (Cover and Hart, 1967), for example, exemplars of the classes are amassed, monotonically, over the course of training. It is also possible to have monotone learning where memory does not grow, but decreases over time. Suppose we try to classify Boolean feature vectors using conjunctive normal form formulas, one for each class. Starting with large and randomly initialized formulas, during training we simply discard, from the formula of the feature vector's class, all the inconsistent clauses (Mooney, 1995). Clearly an attractive feature of both of these monotone algorithms is the computational ease of reaching the desired outcome, i.e. additional exemplars for improved discrimination, or discarded clauses to better accommodate a class.

©2019 Veit Elser, Dan Schmidt and Jonathan Yedidia.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-281.html.

We report on a new kind of monotone learning where during training only the values of a fixed number of parameters are changed, and in a monotone fashion. Like the two examples above, there is a simple and computationally tractable objective when learning each new item. But perhaps the most interesting feature of our scheme is that it operates on deep networks of rectifiers, a setting where training is not normally seen as computationally tractable.

Monotone learning with deep rectifier networks is made possible by two changes to the standard paradigm: (i) *conservative* updates that minimize the norm of the parameter changes, and (ii) eliminating all but the bias variables as learned parameters.

Although the feedforward computations in standard neural network models are inspired by biological neuronal computations, the training protocols in widespread use seem far from natural. Humans are able to generalize the shape of the digit 5 without suffering through many training "epochs" through a data set, and are capable of building a rudimentary representation of digits or other classes of objects from relatively few examples. The conservative learning principle, introduced to the best of our knowledge by Widrow et al. (1988) as the "minimal disturbance principle," comes closer to our experience of natural learning. In this mode of learning the parameters of the network are minimally changed to accommodate each example as it is received, with the rationale that the attention to minimality preserves the representation created by earlier examples.

The minimal disturbance principle was reprised by Crammer et al. (2006) as the "passive-aggressive algorithm", in the context of support-vector machines. There are relatively few models where "aggressive" parameter updates — minimal, yet achieving immediate results — are easily computed. To emphasize the norm-minimizing characteristic of this learning mode we use the term "conservative learning." An approximate implementation of conservative learning was successfully used to discover the Strassen rules of matrix multiplication (Elser, 2016).

In the present work we make changes to the "standard neural network model" to enable conservative learning. In broad outline, the changes guarantee monotonicity, making the computation of the conservative updates tractable. A key step was elevating the role of the additive parameter, or bias. The metaphor that replaces Hebbian synapse (weight) learning is that of a silting river delta, with myriad channels whose levels (bias) rise differentially, but monotonically, over time.

Below is an overview of our contributions.

- Inspired by analog implementations of logic, we propose replacing the conventional rectified sum computed by a standard neuron using a ReLU function

$$y \leftarrow \max\left(0,\ \sum_i w_i x_i - b\right),$$

by a sum of rectifications:

$$y \leftarrow w \sum_i \max(0,\ x_i - b_i). \tag{1}$$

Only the bias parameters are learned; the weights are static.

- Even with just positive weights w in (1), we show that networks of *rectified wires* can represent arbitrary Boolean functions. This construction makes use of doubled inputs, where TRUE is encoded as $(1, 0)$, FALSE as $(0, 1)$, and generalizes, for symbolic data, to one-hot encoding.

- The node values of our networks are non-negative. We propose defining class membership of data by the corresponding output node having value zero.

- Because the biases on all the wires are non-negative and can only increase during training, we choose to initialize them at zero.

- We show that the 2-norm minimizing change of the bias parameters, that sets the class output node for a given input to zero, is the solution to a quadratic program.

- An iterative algorithm similar to stochastic gradient descent, including both backward propagation and a new kind of forward propagation, is proposed to approximately find the 2-norm minimizing bias changes. Executing the minimum number of iterations required to make the class output node the smallest in value defines the *sequential deactivation* algorithm (SDA).

- We show that a particular limit of SDA, on networks with a single hidden layer, learns arbitrary Boolean functions.

- Networks with better scaling have multiple layers, and their training is sensitive to the settings of the static weights. We propose *balanced weights* based just on the in-degrees and out-degrees in this general setting.

- A two-parameter family of random sparse expander networks is introduced to explore the size and depth behavior of learning in our model.

- Conservative learning on rectified wire networks with the SDA algorithm is demonstrated for MNIST and synthetic datasets.

- Despite our model's capacity for depth in its representations, we find that the best experimental results with the SDA algorithm are obtained in a limit where deactivation occurs only in the final layer, where the model is equivalent to a network with just a single hidden layer of rectifier gates. The pursuit of conditions that fully utilize activation/decativation in all layers is identified as the goal of future research.

1. Rectified wire networks

One motivation for our network model is the implementation of logic by analog computations without multiplications. The analog counterparts of TRUE and FALSE are, respectively, the numbers 1 and 0. Before presenting our "rectified wire" network model, we consider networks constructed from biased rectifier gates. A biased rectifier gate with K inputs,

$$R[b](x_1, \ldots, x_K) = \max(0, x_1 + \cdots + x_K - b),$$

generalizes AND and OR gates. AND gates are realized with bias $b = K - 1$. We would get the OR gate with $b = 0$ if we could saturate the output at the value 1. We show below how this detail can be fixed with additional rectifiers and a suitable network design.

Negations are completely absent from our rectifier implementations of general logic circuits. This is made possible by the process of *demorganization*, where NOT gates are pushed through the circuit from output to input, exchanging AND and OR gates as dictated by De Morgan's laws. After all the NOT gates have been pushed through, the only remaining NOT gates act directly on the inputs. We accommodate this by allowing each input to the logic circuit to be replaced by an analog pair with values $(1, 0)$ for TRUE and $(0, 1)$ for FALSE. We refer to this encoding scheme as *input doubling*.

It is relatively straightforward to show, as we do in theorem 1.1, that networks comprising only rectifier gates, with bias variables as the only parameters, can mimic any logic circuit. From the machine learning perspective, our network model has at least the capacity to represent the classes defined by the truth value of arbitrary Boolean functions. However, the parameter settings that realize these Boolean classes are very special points in the continuous parameter space of the model, and are not claimed to be directly relevant for the intended use of these networks.

Theorem 1.1 *Any Boolean function on N inputs and computed with M binary* AND/OR *gates and any number of* NOT *gates can be implemented by an analog network comprising at most $5M$ biased rectifier gates taking the corresponding $2N$ doubled analog inputs.*

Proof As a circuit, the function takes N Boolean inputs and produces one Boolean output. We associate negations in the circuit with gate outputs as shown for the case of the AND gate in the left panel of Figure 1. Upon output, both the gate output y_1 and its negation y_2 by a NOT gate (rendered as an open circle), are made available to gates receiving inputs. One of the output forks might not be used, as for the final gate of the circuit whose single output is the value of the Boolean function. The diagram also shows how each of the gate inputs is derived from one fork of the output of another gate, or an input to the circuit.

The body of the proof consists of confirming that the Boolean gate relationship between the inputs (x_1, x_2), and the output y_1 and its negation y_2, is exactly reproduced by a corresponding rectifier gate network. The replacement for the AND gate is shown in the right panel of Figure 1. All bias parameters are either 0 or 1, and the only values that arise on the nodes are also 0 and 1. The replacement rule for the OR gate is shown in Figure 2. For both of the circuit replacement rules just described there are three others, where one or both of the inputs are negated. These are trivially generated from the ones shown by inserting a negation right after the inputs (x_1 or x_2 or both), which is equivalent to swapping the branches of the input-forks.

For completeness we include the cases where a gate receives both inputs from the same output fork. The case of the AND gate is shown in Figure 3. The replacement rule shown on the left applies when both inputs come from the same branch of the input fork, while the rule on the right applies when the inputs come from both branches. As before, the rule for the other choice of input branch is obtained by swapping branches in the replacement. For the OR gate the rule on the left of Figure 3 is unchanged, while the one on the right has the two bias values swapped.

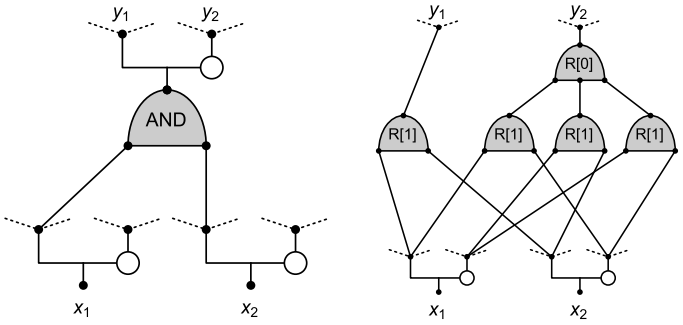


Figure 1: AND gate and its replacement with rectifier gates in the analog network.

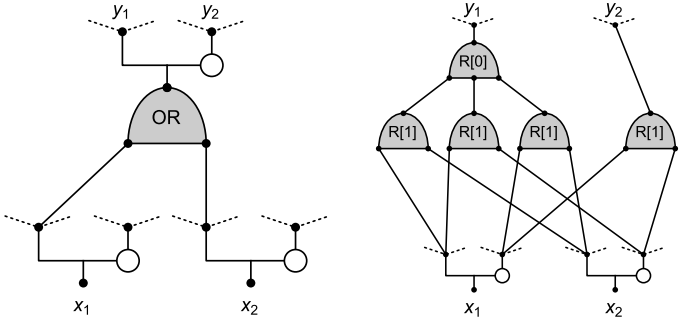


Figure 2: OR gate and its replacement with rectifier gates in the analog network.

The only negations remaining after all AND and OR gates have been replaced are those in the N input forks to the circuit. These are taken care of by defining the input to the analog network as the $2N$ values on the branches of those forks. The resulting network will have only rectifier gates. From the gate replacement rules (Figs. 1-3) we see that the number of rectifier gates is at most $5M$.

■

We now generalize our rectifier gate model in two ways. The first follows from the observation that a rectifier on K inputs with bias b is equivalent to summing those inputs and placing a (single-input) rectifier with bias b on each of the wires leaving the rectifier.

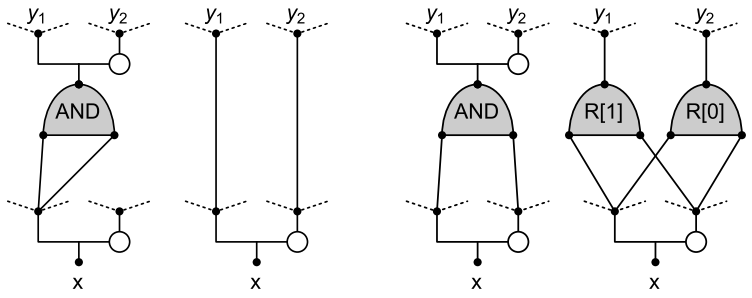


Figure 3: AND gate replacement rules when both inputs come from the same output fork, either the same branch (left) or both branches (right). The rule for the OR gate is the same for the case on the left, and has $R[0]$ and $R[1]$ exchanged for the case on the right.

We give these biases the freedom to take different values, increasing the expressivity of the network and boosting the parameter count commensurate with "standard model" networks, which have a weight on each edge. By setting the biases on selected edges in a completely connected network to large values, thereby disabling them, our model has the capacity to select its architecture.

The second way we generalize the model is to introduce weights at all the summations. These are static — not intended to be learned — and positive. We show in section 6 that in layered networks one can apply a rescaling to the biases that has the effect of restoring all the weights to 1 without changing the classification behavior of the network. The weights therefore only play a role during training, through the gain/decay they impart to the analog signal as it propagates from input to output, and the response this elicits from the bias parameters.

We now formally define our model.

Definition 1.2 *A rectified wire network is a model of computation on a directed acyclic graph. Associated with each wire or edge $i \to j$ joining node i to j is the node value x_i and edge output $y_{i\to j}$ related by*

$$y_{i\to j} = \max(0, x_i - b_{i\to j}),$$

where $b_{i\to j}$ is the bias parameter for the edge. The edge outputs $y_{i\to j}$, from all edges $i \to j$ incident on node j, are summed to give the value x_j of node j:

$$x_j = w_j \sum_{i\to j} y_{i\to j}.$$

The positive constant w_j is the weight of the node. The node values x, edge outputs y, and bias parameters b of a rectified wire network are general real numbers.

The number of edge outputs y summed by a node j is its in-degree $|{\to}j|$, and the number of edges receiving x_j as input is its out-degree $|j{\to}|$. Nodes with in-degree zero are input nodes and have their x values assigned. The outputs of the network are the x values of the nodes with out-degree zero, the output nodes. Nodes which are neither input nor output nodes are the hidden nodes of the network. The connectivity of rectified wire networks is such that there always exists a path between any input node and any output node.

The content of theorem 1.1, re-expressed in terms of rectified wires, takes the following form:

Corollary 1.3 *The truth value of a Boolean function on N variables that can be computed with M binary* AND/OR *gates and any number of* NOT *gates can be represented by a two-output rectified wire network taking $2N$ doubled inputs and having at most $7M$ hidden nodes. Like the doubled inputs, the two output nodes encode the truth of the function with values $(1, 0)$ or $(0, 1)$.*

Proof A logic circuit with M binary AND/OR gates has M nodes, one at each gate output. When re-expressed as a rectified wire network, each gate output is replaced by two nodes and, in the worst case (Figs. 1-2), the gate itself results in five additional nodes. The number of hidden nodes in the resulting rectified wire network is therefore bounded by $7M$. The wiring of the nodes, including the input and output nodes, can be the complete acyclic graph because large bias settings, by deselection, realize any network on the given number of nodes. ■

While possible in principle, we should not expect it to be easy for a rectified wire network to learn bias values on the modest networks promised by corollary 1.3 when presented with data generated by a Boolean function. On the other hand, as we show in section 3, this particular network model has a significant advantage over the "standard model" in having tractable conservative learning. A different, equally valid representation might be learned instead, most likely on a much larger network.

We render rectified wire networks as simple directed graphs, with input nodes at the bottom, output nodes at the top, and all edges directed in the upward direction. The rendering of the bias values — the only learned parameters — is through the edge thickness. Edges with low bias are thick, those with high bias are thin. When the bias on an edge is so great that the rectified value is zero for any input to the network, the edge is effectively absent (zero thickness). Figure 4 shows this style of rendering of a network with four input, four hidden, and two output nodes.

We can generalize both the inputs and the outputs of rectified wire networks to expand their scope beyond Boolean function-value classifiers. In many applications the feature vectors are strings of symbols from a finite alphabet: {TRUE, FALSE}, {A,C,G,T}, etc. As in the Boolean case, for an alphabet of size K we use the one-hot encoding scheme where the k^{th} symbol is expanded into K numerical values all zero except for a 1 in the k^{th} position. Numerical (non-symbolic) data may be encoded in the same style. Suppose the data vectors have length L: $v = (v_1, \ldots, v_L)$. We first obtain the cumulative probability function θ_i for

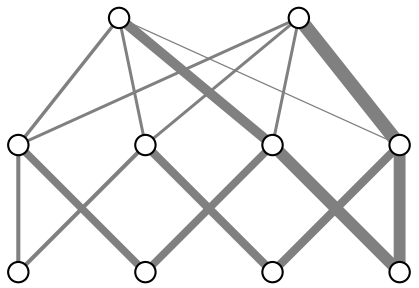


Figure 4: Rectified wire network with four input nodes (bottom row), four hidden nodes, and two output nodes (top row). Thinnest edges have the highest bias.

each component $i = 1, \ldots, L$ over the dataset. The data vector for input to the network, corresponding to v, is then

$$d(v) = [\, \theta_1(v_1)\,, 1 - \theta_1(v_1)\,,\ \ldots\ ,\ \theta_L(v_L)\,, 1 - \theta_L(v_L)\,]\,. \tag{2}$$

With this convention, the input to the network is always a vector of numbers between 0 and 1 whose sum equals the number of components or string length of the data. The mapping by the cumulative probability functions, in the case of analog data, ensures that the distribution in each channel is uniform.

The number of output nodes equals the number of data classes. A Boolean function classifier would have two output nodes to classify the truth value of the input. To classify MNIST data (LeCun et al., 1998), the network would have 10 output nodes. Before we describe how these outputs are interpreted we make a trivial observation:

Lemma 1.4 *In a rectified wire network with input vector $d \geq 0$, there exist settings of the bias parameters where, for any of the output nodes $c \in C$, we have $x_c = 0$ and $x_j > 0$ for $j \in C \setminus \{c\}$.*

Proof All the output nodes will have positive values for zero bias, because the weights are positive and each output is connected by a path to one of the inputs with positive value (see definition 1.2). To exclusively arrange for $x_c = 0$, we set $b_{i\to c} = x_i$ on all the edges $i \to c$ incident on c (keeping all other biases at zero). ■

We will explain in section 3 why zero is the appropriate indicator function value for defining classes on a conservatively trained rectified wire network.

2. Notation

Now that most of the elements of our network model have been introduced, we summarize our notational conventions. Lower case symbols represent variables or parameters on the

network. These bear a latin index when defined on a node of the network. Examples are the node values, x_i, or the weights w_i. The index is written as a directed edge between nodes when defined on an edge of the network. Examples are the bias parameters, $b_{i\to j}$, and edge-outputs $y_{i\to j}$. The same lower case symbols, both for nodes and edges, when written without subscripts denote the vector of all the indexed values. For example, b is the vector of all biases in the network and d the vector of data values on the input nodes. When such symbols appear in an equality/inequality, the relation applies componentwise. Round brackets attached to a variable identify its functional dependence, when relevant. The value of node k, for example, might be written $x_k(b,d)$ to emphasize its dependence on the biases and input data.

Upper case symbols are used for sets. The symbols D, H, C are reserved for the data (input), hidden, and class (output) nodes respectively, while E is the set of directed edges. Upper case symbols are also used for continuous sets with the same convention for functional dependence as variables. For example, $B(d)$ might be the set of bias values b such that for $b \in B(d)$ the node value $x_i(b,d)$ is zero for input d. Cardinality is denoted with vertical bars, e.g. $|E|$ for the number of edges in the network. Node i in a network has in-degree $|{\to}i|$ and out-degree $|i{\to}|$.

3. Counterfactual classification and conservative learning

Monotone learning on rectified wire networks is made possible by the fact that, for a fixed input $d \geq 0$ to the network, the value $x_k(b,d)$ at any node k is a non-negative and non-increasing function of the bias parameters b. This, combined with the fact (lemma 1.4) that zero is always a feasible value on any of the output nodes, motivates the following definition of a classifier.

Definition 3.1 *The output nodes C of a rectified wire network serve as a classifier of $|C|$ classes of inputs when there are biases b such that for inputs d in class $c \in C$, $x_c(b,d) = 0$ and $x_i(b,d) > 0$ for $i \in C \setminus \{c\}$.*

To see why this definition makes sense in the context of conservative learning, suppose that at some stage of training all the output nodes are positive for some input d in class c. By increasing the biases b, the output value $x_c(b,d)$ can always be driven to zero for input d, and by increasing them as little as possible, i.e. conservatively, there is a good chance the other output nodes can be kept positive. If some of the other outputs are also set to zero in the process of increasing biases, the classification is ambiguous. In any case, and after any amount of subsequent training where biases are increased, monotonicity of $x_c(b,d)$ with respect to b ensures $x_c(b,d) = 0$ continues to be a valid indicator for class c.

The outputs of the proposed classifier are counterfactual in the sense that a large positive value x_c on the node for class c represents strong evidence that c is not the correct class. In the case of a class defined by the truth value of a Boolean function, we would use the function that returns FALSE for members of the class, so the value of the corresponding analog circuit output node is zero.

We shall see that we do not need to separately insist that, during training, biases are never decreased, as it will follow automatically from the conservative learning principle which assigns the following cost when biases are changed:

Definition 3.2 *The cost, in a rectified wire network, for changing the biases from b to b', is the p-norm $\|b' - b\|_p$.*

The uniqueness of the cost-minimizing bias change, for fixing a misclassification, depends on the norm exponent p. We mostly use $p = 2$ in this study, where uniqueness can be proved. Our definition of the conservative learning rule, which sidesteps uniqueness, is the following:

Definition 3.3 *Suppose output node $c \in C$ for class c has value $x_c(b,d) > 0$ for an input d, in a network with biases set at b. A conservative bias update is any b' that minimizes $\|b' - b\|_p$ and produces output value $x_c(b',d) = 0$.*

Lemma 3.4 *If b' is any conservative update of b, then $b' \geq b$.*

Proof Biases b' have just learned an input d in some class $c \in C$, so $x_c(b',d) = 0$. Now suppose $b'_{i\to j} < b_{i\to j}$ for some edge $i \to j \in E$. The update b'', where $b''_{i\to j} = b_{i\to j}$ and all others are unchanged, has lower cost than b' and yet, by monotonicity of node values with respect to b, all output nodes with value zero will still be zero, including $x_c(b'',d)$. ■

The outputs of a rectified wire network are compositions of rectifier functions. Composition of convex functions are in general not convex, but for the rectifier function this is true. We give an inductive proof based on the following.

Lemma 3.5 *If $x(b) : \mathbb{R}^K \to \mathbb{R}$ is convex, then $y(a,b) : (\mathbb{R}, \mathbb{R}^K) \to \mathbb{R}$, defined by*

$$y(a,b) = \max(0, x(b) - a),$$

is also convex.

Proof Because x is convex, for arbitrary $b_1, b_2 \in \mathbb{R}^K$ we have

$$x(\lambda b_1 + (1-\lambda)b_2) \leq \lambda\, x(b_1) + (1-\lambda)\, x(b_2) \tag{3}$$

for $0 \leq \lambda \leq 1$. For arbitrary $a_1, a_2 \in \mathbb{R}$,

$$\begin{aligned} y\left(\lambda a_1 + (1-\lambda)a_2,\ \lambda b_1 + (1-\lambda)b_2\right) &= \max(0, x(\lambda b_1 + (1-\lambda)b_2) - \lambda a_1 - (1-\lambda)a_2) \\ &\leq \max(0, \lambda\, x(b_1) + (1-\lambda)\, x(b_2) - \lambda a_1 - (1-\lambda)a_2) \\ &= \max(0, \lambda\, z_1 + (1-\lambda)\, z_2), \end{aligned}$$

where we have used (3) and defined

$$\begin{aligned} z_1 &= x(b_1) - a_1 \\ z_2 &= x(b_2) - a_2. \end{aligned}$$

Since for arbitrary $s, t \in \mathbb{R}$

$$\max(0, s+t) \leq \max(0, s) + \max(0, t),$$

we obtain

$$\begin{aligned}\max(0, \lambda z_1 + (1-\lambda) z_2) &\leq \max(0, \lambda z_1) + \max(0, (1-\lambda) z_2) \\ &= \lambda \max(0, z_1) + (1-\lambda) \max(0, z_2) \\ &= \lambda\, y(a_1, b_1) + (1-\lambda)\, y(a_2, b_2)\end{aligned}$$

as claimed. ■

Theorem 3.6 *The node values $x(b,d)$ of a rectified wire network, given an input d, are convex functions of the bias parameters b.*

Proof The values of the input nodes D, set to the constant values $x_i = d_i, i \in D$, are trivially convex. Since our network is an acyclic graph, the nodes can be indexed by consecutive integers so that all edge outputs $y_{i\to j}$ incident on a node $j \in H \cup C$ are from input nodes $i \in D$ or from hidden nodes $i \in H$ that have $i < j$. For any $j \in H \cup C$, we use induction and suppose all x_i, $i \in H$ with $i < j$ are convex functions of the bias parameters. Consider any of the edge outputs $y_{i\to j}(b)$ incident on j:

$$y_{i\to j}(b) = \max(0, x_i(b) - b_{i\to j}).$$

Seen as a function of bias parameters, where $x_i(b)$ is a convex function of biases not including $b_{i\to j}$, $y_{i\to j}(b)$ fits the hypothesis of lemma 3.5 and is therefore convex. This conclusion also applies to the base case, the hidden node numbered $j = 1$, where all the x_i are inputs $i \in D$ and constant. Since the node value

$$x_j(b) = w_j \sum_{i\to j} y_{i\to j}(b)$$

is a positive multiple of a sum of convex functions, it too is convex and completes the induction. ■

For the case $p = 2$, we can prove uniqueness of the conservative bias update.

Corollary 3.7 *The set $B(d)$, of biases b of a rectified wire network with fixed input d for which $x_c(b,d) = 0$ for a given output node c, is non-empty and convex. Consequently, the conservative bias update $b' \in B(d)$ that minimizes the cost $\|b' - b\|_2$ with respect to the biases b is unique.*

Proof The set $B(d)$ may be defined as the feasible set of $x_c(b,d) \leq 0$ (since this function is non-negative). By lemma 1.4 $B(d)$ is non-empty, and by theorem (3.6) $B(d)$ is closed and convex because $x_c(b,d)$ is a convex function of b. Now suppose $b'_1 \in B(d)$ and $b'_2 \in B(d)$ are distinct minimizers of $\|b' - b\|_2$ for some bias b. Because $B(d)$ is convex we have a contradiction because $b'_3 = (b'_1 + b'_2)/2 \in B(d)$ and $\|b'_3 - b\|_2 < \|b'_1 - b\|_2 = \|b'_2 - b\|_2$ implies b'_1 and b'_2 were not minimizers. ■

To design algorithms for computing the conservative bias update and to establish the complexity of this task we cast the problem as a mathematical program. Consider a rectified wire network with edges E on which we are given bias parameters b, say from previous training. Suppose we now wish to learn the pair (c,d), a vector of input data d in the class associated with output node c. Unless $x_c(b,d) = 0$, the biases must be conservatively updated to b' so this is true. Before we do this we partition E into the set of *active* edges A and its complement, $\bar{A}$.

Definition 3.8 *The set A of active edges of a rectified wire network, given input data d and bias parameters b, are those edges $i \to j$ where $x_i(b,d) > b_{i\to j}$, or equivalently, where $y_{i\to j}(b,d) > 0$.*

The biases $b_{i\to j}$ on the inactive edges $i \to j \in \bar{A}$, when increased, have no effect on any node values, in particular the output nodes, because the corresponding edge-outputs $y_{i\to j}$ are zero by monotonicity of $x(b)$ (non-increasing). To find b' we may therefore work with the network induced by the active edges A, called the "active network."

Our mathematical program makes use of a set of *reduced bias* variables.

Definition 3.9 *Let A be the active edges of a rectified wire network for input data d and bias parameters b. For each edge $i \to j$ of the network, define the reduced bias by*

$$b^-_{i\to j} = \begin{cases} b_{i\to j}, & i \to j \in A \\ x_i(b,d), & i \to j \in \bar{A}. \end{cases}$$

Now consider a conservative update b' of b, and the reduced biases b^- for b'. We always have $b^- \leq b'$, and the reduced values have the property that $x(b^-, d) = x(b', d)$. In particular, for the class output node c we have $x_c(b^-, d) = x_c(b', d) = 0$. The only reason that a conservative update b' of b would not already be reduced ($b' = b^-$) is that the freedom $b^- \leq b'$ may allow a reduction in cost, that is, give a more conservative update.

We now define the mathematical program. For given data d, biases b and weights w on a rectified wire network, we are given the corresponding active edges A and node c of the correct class. The unknowns are the updated biases b', reduced biases b^-, and edge-outputs y on A, and the values of x on c and the input and hidden nodes:

$$\begin{aligned}
&\text{minimize:} && \|b' - b\|_p^p && && \text{(4a)}\\
&\text{such that:} && b^- \leq b', && && \text{(4b)}\\
& && 0 \leq b^-, && && \text{(4c)}\\
& && x_i = d_i, && i \in D && \text{(4d)}\\
& && y_{i\to j} = x_i - b^-_{i\to j}, && i \to j \in A && \text{(4e)}\\
& && 0 \leq y_{i\to j}, && i \to j \in A && \text{(4f)}\\
& && x_j = w_j \sum_{i\to j \in A} y_{i\to j}, && j \in H \cup \{c\} && \text{(4g)}\\
& && x_c = 0. && && \text{(4h)}
\end{aligned}$$

Theorem 3.10 *The mathematical program* (4)*, defined for the network A that is active for given biases b and data d, as part of its solution b' gives a conservative bias update for the class associated with output node c.*

Proof First observe that any valid ($x_c = 0$) assignment of node values x and edge-outputs y, in the corresponding rectified wire network, is realized by a feasible point of the linear system (4c) – (4h) involving just the reduced biases (not b'). For the variables b' to be rectified wire biases consistent with the reduced biases b^-, it is sufficient for them to satisfy (4b), as monotonicity ensures $x_c(b') = 0$. Optimizing (4a) subject to (4b) and the constraints that define the feasible set of reduced biases, gives the most conservative update. ■

For $p = 2$, (4) is a positive semi-definite quadratic program and can be solved in time that grows polynomially (Kozlov et al., 1980) in the size of the active network, $|A|$. The same conclusion applies for $p = 1$, a linear program, because the objective can be replaced by the sum of all the updated biases b' provided we impose the result of lemma (3.4), $b \leq b'$, as a constraint. While these results allow us to add rectified wire networks to the list of models for which there is a tractable conservative learning rule, in most applications it is also important that training scales nearly linearly with the size of the network. The *sequential deactivation algorithm* (SDA) described in the next section is designed to meet that goal.

We close this section by reviewing the features of our model that make training tractable. The insistence on exactly learning individual items should not be counted as a tractability-enabling feature. Indeed, it is easy to see how the mathematical program (4), to compute the conservative update for one data item, would be generalized to find the norm-minimizing update that classifies an entire mini-batch. All the variables and data, with the exception of the biases b and b', now carry a data index $m \in M$, where M is a mini-batch. For example, (4b) would be replaced by

$$b^-{}_{i\to j,m} \leq b'_{i\to j} \qquad i \to j \in A,\ m \in M$$

and imposes consistency of the bias updates for all instantiations of the reduced biases over the mini-batch. While the size of the mathematical program has grown to $O(|A||M|)$ equations and inequalities, it is still tractable.

Convexity of the variables x and y with respect to the bias parameters is clearly important for tractable learning, and the proofs of lemma 3.5 and theorem 3.6 show that this relies on the non-negativity of the static weights and the form of the activation function. Less obvious in facilitating tractability is the proposal to replace the loss function of neural networks by a constraint, and in particular, the simple constraint (4h). For example, one might consider replacing this single constraint with the following set of inequalities,

$$x_i > x_c + \Delta, \quad i \in C \setminus c, \tag{5}$$

where the fixed positive parameter Δ specifies a margin for avoiding ambiguous classification. After all, we still have a tractable mathematical program after this substitution. Unfortunately, this proposal creates a conflict in the relationship between the reduced biases b^- and the actual bias updates b'. While the b^- variables provide an exhaustive parameterization of the node variables x, even under the constraint (5), we cannot count on the

inequality $b^- \leq b'$ to preserve this constraint when b^- biases are replaced by b' biases in the rectified wire model (the monotonic decrease of a non-class output may exceed that of the class output).

4. Sequential deactivation algorithm

This section describes in detail the SDA algorithm for computing bias updates in the rectified wire model. While these updates are not guaranteed to be the most conservative possible, their computation is significantly faster than solving a general quadratic program. The SDA algorithm resembles the gradient methods used for optimizing standard model networks, the most widely used being stochastic gradient descent (SGD) (Rumelhart et al., 1986).

The program (4) would be greatly simplified if inequality (4f) was somehow automatically satisfied. All active edges would remain active in the update and the updated biases b' on them would be equal to the reduced biases b^-. The node variables x, in particular x_c, would be linear functions of the b':

$$x_c(b') = x_c(b) + \nabla x_c \cdot (b' - b).$$

For the $p = 2$ norm on bias updates, the optimal update that achieves $x_c = 0$ is then

$$b' = b - \frac{x_c(b)}{\|\nabla x_c\|^2} \nabla x_c.$$

The conservative update, in this simplification, is the same as gradient descent.

In the general case, where edges may become inactive, from the monotonicity of x_c we know at least that the gradient ∇x_c never has positive components. If we had the stronger property, that ∇x_c is also never zero while $x_c > 0$, then gradient descent will always find an update b^* that satisfies $x_c(b^*) = 0$. But the latter property follows immediately from the fact that $x_c > 0$ is only possible when there are active edges incident to output node c, so that x_c has non-zero derivatives at least with respect to the biases on them.

While gradient descent cannot guarantee the norm minimizing update promised by the solution of program (4), it has some attractive features. First, the computation of the (piecewise constant) gradient is fast, as is the computation of the step size to the next gradient discontinuity (deactivation event). Second, there is an opportunity to make the gradient descent update b^* more conservative on the inactive edges. Using the condition $y_{i\to j}(b^*) = 0$ as the indicator of an inactive edge, the improved update is, for all $i \to j \in A$,

$$b'_{i\to j} = \begin{cases} \max\left(b_{i\to j}, x_i(b^*)\right), & y_{i\to j}(b^*) = 0 \\ b^*_{i\to j}, & y_{i\to j}(b^*) > 0. \end{cases} \tag{6}$$

The SDA algorithm is the efficient implementation of gradient descent on the rectified wire model followed by (6). Its name draws attention to the fact that the steps in the descent are defined by deactivation events.

We now turn to the implementation of the SDA algorithm. The elementary operations are defined in Algorithm 1. Procedure EVAL is simply a forward-pass through the network, from data supplied at the input nodes, to the output nodes whose values directly encode

Algorithm 1 Elementary network procedures

procedure EVAL(E, b, d) $\to (A, x, y)$
$x_i \leftarrow d_i, \quad i \in D$
$x_j \leftarrow w_j \sum_{i\to j\in E}\ y_{i\to j}, \quad j \in H \cup C$
$y_{i\to j} \leftarrow \max(0, x_i - b_{i\to j}), \quad i \to j \in E$

$A \leftarrow \{i \to j \in E : y_{i\to j} = 0\}$
end procedure

procedure GRAD(A, c) $\to \nabla x_c$
$(-\nabla x_c)_j \leftarrow w_c \delta_{jc}, \quad j \in C$
$(-\nabla x_c)_i \leftarrow w_i \sum_{i\to j\in A}(-\nabla x_c)_j, \quad i \in H$
end procedure

procedure VELOCITY(A, ∇x_c) $\to \dot{y}$
$(-\dot{y})_{i\to j} \leftarrow (-\nabla x_c)_j + w_i \sum_{k\to i\in A}(-\dot{y})_{k\to i}, \quad i \to j \in A$
end procedure

the class. In addition to the output node values $\{x_i : i \in C\}$, EVAL also provides the edge outputs y and the set of active edges A (on which the edge outputs are non-zero).

Suppose the current bias parameters are $b(0)$. Denote by $(-\nabla x_c)_{i\to j}$ the negative gradient component for the bias on edge $i \to j \in A$. Because the derivatives with respect to all the $b_{i\to j}$ on active edges into the same node j are equal, the node-indexed variables

$$(-\nabla x_c)_j \coloneqq (-\nabla x_c)_{i\to j}, \quad i \to j \in A$$

are well defined. In particular, if j is an output node, then $(-\nabla x_c)_j = w_c\,\delta_{jc}$. As long as the gradient is constant, the biases evolve as

$$b_{i\to j}(t) = b_{i\to j}(0) + t(-\nabla x_c)_j, \tag{7}$$

where t is a continuous "time".

The gradient $(-\nabla x_c)_i$ is positive only if node i is connected by active edges to node c, with value given by the sum over all paths on active edges, each contributing by the product of weights along the path. The effect on x_c of a bias change on an active edge into i is the same as if the same bias change was instead applied to all the active edges leaving node i, but multiplied by w_i. This implies the recursion,

$$(-\nabla x_c)_i = w_i \sum_{i\to j\in A}(-\nabla x_c)_j, \quad i \in H \tag{8}$$

in the procedure GRAD of Algorithm 1 and corresponds to back propagation in the network.

The third procedure, VELOCITY, is derived from the recursion for the edge-outputs y:

$$y_{i\to j} = w_i \sum_{k\to i\in A} y_{k\to i} - b_{i\to j}.$$

Taking the time derivative and using (7),

$$(-\dot{y})_{i\to j} = w_i \sum_{k\to i\in A}(-\dot{y})_{k\to i} + (-\nabla x_c)_j \tag{9}$$

we obtain the velocities of the edge-outputs by forward propagation. The initialization occurs at edges $i \to j$ from all the input nodes i, for which the sum in (9) is absent and $(-\nabla x_c)_j$ is set by GRAD.

By construction, both $y_{i\to j}(0)$ and the (constant) velocities $(-\dot{y})_{i\to j}$ on the active edges $i \to j \in A$ are positive and the first deactivation event occurs at time

$$t^* = \min_{i\to j\in A} \frac{y_{i\to j}(0)}{(-\dot{y})_{i\to j}}. \tag{10}$$

This is the time step in one iteration of the SDA algorithm. After the biases are incremented by (7) and the newly deactivated edges are removed from A, another round is begun. Iterations are terminated when EVAL returns $x_c = 0$. The final biases are obtained by applying (6) to the gradient descent biases.

We now revisit the classification rule of definition 3.1, where data is declared learned when $x_c = 0$. Recall that this has the desired property of not having the "supremacy" (smallest value among all classes) of node c spoiled by subsequent training. The only way that the latter can have a negative effect, through the general decrease in output values with increasing biases, is when output nodes other than c are in a tie at value zero. To mitigate this effect, and also make the bias changes even more conservative, we introduce the *ultra-conservative learning* rule:

Definition 4.1 *In the ultra-conservative mode of learning with the* SDA *algorithm, iterations are terminated when x_c, the value of the output node c of the correct class, is either zero or smaller than the values of all the other output nodes.*

This termination criterion is conservative from the point of view of testing. Consider the early stages of training, when we might want to test the algorithm on which class is being favored. The natural candidate for the latter is the output node with the smallest value. By terminating SDA as soon as this criterion is met, the test (when performed right after training) will succeed with a smaller change to the biases than is required by the $x_c = 0$ criterion.

Because the property of data d producing the smallest output value on node c can be spoiled by training with data $d' \neq d$, it may be necessary for the network to retrain on d, or data similar to d, in order to properly learn the combination (d, c). The implied heuristic is that the net bias change for this mode of learning class c may be more conservative than always insisting on output $x_c = 0$ for every d in this class.

The SDA algorithm with the ultra-conservative termination criterion is summarized in Algorithm 2. As soon as x_c becomes the smallest output or is zero, the algorithm returns the most conservatively updated biases b' consistent with the activity of the network edges, (6). Estimating the work required to correct a misclassification is complicated by two factors. While the work in one iteration scales linearly with the size of the active network, we do not know how many iterations are needed on average to satisfy the ultra-conservative

Algorithm 2 Sequential Deactivation (SDA)

input $(E, b^0), (d, c)$ (network edges, initial biases), (data vector, class)

$b \leftarrow b^0$ initialize biases
$(A, x, y) \leftarrow \text{EVAL}(E, b, d)$ active edges, node and edge output values
$\mathsf{iter} \leftarrow 0$ zero the iteration counter

while $x_c > 0$ and $\arg\min_{i \in C} x_i \neq \{c\}$ **do** iterate until node c is smallest
  $\nabla x_c \leftarrow \text{GRAD}(A, c)$ gradient with respect to active biases
  $\dot{y} \leftarrow \text{VELOCITY}(A, \nabla x_c)$ edge output velocity
  $t^* \leftarrow \min_{i \to j \in A} \; y_{i \to j} / (-\dot{y}_{i \to j})$ step size

  for $i \to j \in A$ **do**
    $b_{i \to j} \leftarrow b_{i \to j} + t^* \, (-\nabla x_c)_j$ increase biases
  end for

  $(A, x, y) \leftarrow \text{EVAL}(A, b, d)$ new active edges, node and edge output values
  $\mathsf{iter} \leftarrow \mathsf{iter} + 1$ increment iteration counter
end while

for $i \to j \in E$ **do**
  if $i \to j \in A$ **then**
    $b'_{i \to j} = b_{i \to j}$ keep biases on active edges
  else
    $b'_{i \to j} = \max(b^0_{i \to j}, x_i)$ conservative update for inactive edges
  end if
end for

output b', iter updated biases, iteration count

termination rule. Empirically (section 9) we find the number of iterations is often quite small and depends only weakly on network size. A second effect, serving to lessen the work, is the sparsification of the active network with time. In any event, the total computation in learning each data item would be at most $O(|A|^2)$, since in each iteration at least one of the active edges is deactivated (and eliminated from subsequent rounds).

A possible direction for future research is the design of an algorithm that solves the program (4), for the most conservative update, with only a modest amount of extra work than used by SDA. In this connection we present the simplest instance we know of where the update found by SDA is non-optimal. Figure 5 shows a fragment of a network where the bias updates on four edges can be analyzed in isolation. We consider the original optimization problem (4), where the bias updates must give zero on node c. There may be additional edges incident on the nodes shown, but these are inactive for the data item being learned. Input node 1 has been zero for all previous data and explains why biases $b_{1\to2}$, $b_{2\to3}$ and $b_{2\to c}$ are still zero. The inactive edge incident on node 3 was active on previous data and

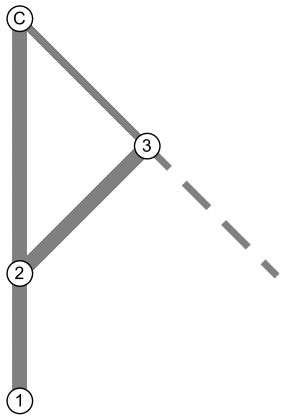


Figure 5: A network fragment on which the SDA update is not optimal. The initial bias on the thick edges is zero, while $b_{3\to c} = b$. Another parameter is the weight w of node 2; all other nodes have unit weight. The dashed edge is inactive initially and irrelevant, but was responsible for $b > 0$ from prior training.

accounts for the positive bias $b_{3\to c} = b$. The nodes all have unit weight except node 2, whose weight is w. Data vector component $d_1 = 1$ (on node 1) is the only one relevant for the fragment shown.

It is easy to see that for the case $w < b$ the SDA solution is optimal, that is, it coincides with the solution of (4). However, with a bit of work one can show that the SDA updates are suboptimal when $w > b$, even when taking advantage of (6). This example only shows there is room for improvement. We do not know how pervasive this type of instance is in practice, let alone how seriously it impacts learning.

We close this section with the observation that SDA is not that different from SGD applied to the loss function x_c. The main differences are that the step size (learning rate) is not a free parameter but set by deactivation events and that multiple steps may be executed when learning each data item. The benefit imparted by the rectified wire model, in this context, is just that the gradient computations and step sizes are very easy to compute. If we set aside monotonicity and solution guarantees, these attributes carry over to loss functions that are much stronger than x_c.

To get a sense of how small a change moves the model into difficult territory, consider the problem of learning the truth value of a Boolean function $f(z)$ on N variables generated by M binary AND/OR gates, as discussed in section 1. The rectified wire network would receive $2N$ doubled inputs and have output nodes f and $\bar{f}$, trained so that $(x_f, x_{\bar{f}}) = (f(z), \bar{f}(z))$. From corollary (1.3) we know such a rectified wire network exists, with unit weights and

having at most $7M$ hidden nodes. The outstanding question, of course, is whether a gradient based algorithm can find the appropriate bias settings.

Instead of the monotone loss function $x_{f(z)}$, fraught with the problem that training cannot prevent $(x_f, x_{\bar{f}}) = (0, 0)$ — an ambiguity — we might try the hinge-loss function

$$\max\left(0,\ x_{f(z)} - x_{\bar{f}(z)} + \Delta\right).$$

By corollary (1.3) we know that it is possible to achieve zero loss over all z with margin $\Delta = 1$, and that the data will be correctly and unambiguously classified. Like $x_{f(z)}$, this loss function is piecewise linear on the rectified wire model and can likewise benefit from fast gradient and step size calculations. Because this loss is no longer monotone decreasing in the bias parameters, there is now no reason to initialize biases at zero. The loss of monotonicity, of course, brings with it the zero gradient problem. We suspect this may be more serious for rectified wires than it is in standard network models.

5. Clause learning

Although the SDA algorithm provides a tractable computation for learning individual data in the rectified wire model, we do not yet have any reason to believe this algorithm can learn any interesting functions. To address that concern, in this section we show that at least in a particular limit, the SDA algorithm is able to learn classes defined by any Boolean function. We already saw in section 1 that rectified wire networks can represent such classes, even with favorable size scaling. To also demonstrate learning we have to rely on an admittedly impractical family of networks, with size growing exponentially with the number of Boolean variables.

Definition 5.1 *The complete Boolean network, for learning a Boolean function $f(z_1, \ldots, z_N)$, has N pairs of input nodes, 2^N hidden nodes, and two output nodes. The input nodes correspond to the literals $z_1, \bar{z}_1, \ldots, z_N, \bar{z}_N$, the hidden nodes to all possible N-clauses $\{z_1, \bar{z}_1\} \times \cdots \times \{z_N, \bar{z}_N\}$ formed from N literals, and the output nodes correspond to the truth value of f. Each hidden node has N edges to each of its constituent literals, and edges to both of the output nodes.*

Figure 6 shows the complete Boolean network for functions of two variables (rendered for zero bias on all edges). In general, the hidden nodes all lie in the same layer. With the weights at the output nodes fixed at 1, the limit of the SDA algorithm we will analyze is where the weight w shared by all the hidden nodes approaches zero. We refer to this limit on complete Boolean networks as *clause learning*

The SDA bias changes in clause learning are very simple. While $(-\nabla x_c)_i \in \{0, 1\}$ when i is an output node, by (8) we have $(-\nabla x_c)_i \in \{0, w\}$ at all the hidden nodes. From (8) it also follows that the bias changes on edges into the hidden nodes are smaller by a factor w relative to edges leaving the hidden nodes. Recall that an SDA iteration is defined by the deactivation of some edge as biases are increased. In clause learning this always occurs in the second layer, in the edges to the outputs. As the hidden node values are $O(w)$, the biases to the output edges will also be $O(w)$ as that suffices for deactivation. While there are corresponding bias changes on edges into the hidden nodes, these are smaller by a factor

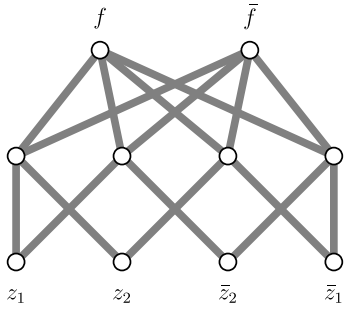


Figure 6: Complete Boolean network for 2-variable Boolean functions $f(z_1, z_2)$. The function value is classified by the smaller of the two output node values, with the convention that node f is smaller when $f(z_1, z_2) = 0$.

of w, or $O(w^2)$, and far from making these edges inactive (since $x_i \in \{0, 1\}$ on the input nodes).

Lemma 5.2 *In clause learning, the hidden nodes and the biases on edges leaving them, always have values of the form $wK + O(w^2)$, $K \in \mathbb{Z}_{\geq 0}$.*

Proof We use induction, with the base case being the start of training where all the biases are zero. For any (doubled) input $d \in \{0, 1\}^{2N}$, the value of a hidden node h is $x_h = wK(d, h)$, where $K(d, h) \in \{0, \ldots, N\}$ counts the number of literals associated with node h that are 1 for input d. Next suppose that a slightly modified statement continues to hold after T iterations of SDA, that is, for any hidden node h and any edge $h \to i$, $x_h, b_{h\to i} \in \{wK + O(w^2) \colon K \in \mathbb{Z}_{\geq 0}\}$. Consider what happens in iteration $T + 1$, invoked when the network makes the wrong prediction and some edge is deactivated by an increase in bias. The deactivated edge will always be an edge leaving a hidden node, and the change of its bias will therefore be the difference of two numbers in the set $\{wK + O(w^2) \colon K \in \mathbb{Z}_{\geq 0}\}$, thus keeping the changed bias in this set. Since the corresponding bias changes on edges into the hidden nodes are $O(w^2)$, the statement about the values of the hidden nodes also continues to hold. ■

Theorem 5.3 *Clause learning succeeds for any Boolean function.*

Proof Consider the values of the two output nodes when clause learning encounters the doubled input $d \in \{0, 1\}^{2N}$ associated with the function argument $z \in \{0, 1\}^N$. By lemma 5.2, $x_f = wK_f + O(w^2)$, $x_{\bar{f}} = wK_{\bar{f}} + O(w^2)$, where $K_f, K_{\bar{f}} \in \mathbb{Z}_{\geq 0}$. As long as the integer for the "wrong" output is greater than zero, SDA will make bias changes, if necessary, to

identify the correct class. For example, suppose $f(z) = 0$ and $K_{\bar{f}} > 0$. Through bias increases SDA can bring about $x_f < x_{\bar{f}}$ because all inactivations will be on edges $h \to f$, that is, from hidden nodes to the "correct" output node. The only effect these changes can have on $x_{\bar{f}}$ is a decrease of order $O(w^2)$, due to $O(w^2)$ bias increases on edges into the hidden nodes. Since each edge $h \to f$ can become inactive just once, the number of SDA iterations required to produce the desired outcome is bounded by the number of hidden nodes.

The above argument does not apply in the case $K_{\bar{f}} = 0$. However, we can argue that this never arises when $f(z) = 0$. If it did, consider the input $d \in \{0,1\}^{2N}$ corresponding to z and the unique hidden node h whose combination of input-literals is such that $x_h = wN + O(w^2)$ (complete Boolean networks sample all N-literal combinations). Since $x_{\bar{f}}$ is a sum of non-negative contributions from all the hidden nodes, $K_{\bar{f}} = 0$ is only possible if $b_{h\to\bar{f}} = wN + O(w^2)$. To finish the proof, we show that this value is inconsistent with conservative learning.

Fixing the hidden node h from above, consider all the arguments z for which $f(z) = 1$ and the bias $b_{h\to\bar{f}}$ required to correctly assign them to the correct class by the smallness of $x_{\bar{f}}$. The corresponding inputs d for such z would produce $x_h = wK + O(w^2)$, with $K < N$, because the clause associated with h is unique in producing $x_h = wN + O(w^2)$ only for an input for which $f(z) = 0$. But to achieve the smallest contribution, $O(w^2)$, to $x_{\bar{f}}$ from edge $h \to \bar{f}$, we only need bias $b_{h\to\bar{f}} = wK + O(w^2)$, $K < N$. This is smaller (more conservative) than the hypothesized value from above.

The same argument, with interchanged output nodes, applies to learning data for which $f(z) = 1$. ■

6. Balanced weights

We say a network is *layered* when the nodes can be partitioned into a sequence of layers $\ell = 0, 1, \ldots, L$, such that all edges in the network are between nodes in adjacent layers. When a rectified wire network has the property that the weights are only layer-dependent, a suitable rescaling applied to the biases will eliminate all the weights — replacing them by 1 — without changing the classification behavior of the network. Denoting the layer of node i as $\ell(i)$, we see from definition 1.2 that the rescalings

$$\begin{aligned} \tilde{x}_i &= x_i / W_{\ell(i)} \\ \tilde{y}_{i\to j} &= y_{i\to j} / W_{\ell(i)} \\ \tilde{b}_{i\to j} &= b_{i\to j} / W_{\ell(i)}, \end{aligned} \tag{11}$$

where

$$W_{\ell(i)} = \prod_{\ell=0}^{\ell(i)} w_\ell,$$

leave the equations unchanged while replacing all the weights by 1. Here $\ell = 0$ and $\ell = L$ correspond to the input nodes and output (class) nodes, respectively. Classification is unchanged because all the output node values are rescaled by the same (positive) factor.

The weights do have an effect on the way the network is trained by the conservative learning rule. In the following we motivate a particular setting of the weights called *balanced*, derived for each node i from its in-degree, $|{\to}i|$, and out-degree, $|i{\to}|$.

From definition 1.2 we see that the choice

$$w_i = \frac{1}{|{\to}i|} \tag{12}$$

will have the effect that there is no net gain or decay in the typical node values x when moving from one layer to the next. The appearance of the in-degree is associated with the forward propagation of x.

A very different choice is suggested by recursion (8), which relates the bias changes $\dot{b} = (-\nabla x_c)$ throughout the network in conservative learning. If we wish the biases to have equal influence on the class output node, so that again there is no net gain/decay moving between layers, then the correct choice is

$$w_i = \frac{1}{|i{\to}|}. \tag{13}$$

Here we have the out-degree because the bias changes are derived from backward propagation.

As edges become inactive over the course of training, the in-degree in (12) and out-degree in (13) should count only the active edges. However, both (12) and (13) are problematic when they are unequal, even at the onset of edges becoming inactive. When (12) and (13) are unequal, the node values x and accumulated bias changes b will grow at different rates from layer to layer. Since edge deactivation is determined by $x - b$, it will not be uniform in the network, but will be concentrated either at the input layer or the output layer.

To promote a more uniform distribution of edge activations in the network we use *balanced weights*:

$$w_i = \frac{1}{\sqrt{|{\to}i|\ |i{\to}|}}, \quad i \in H. \tag{14}$$

With this choice, the propagation of x in the limit of small bias takes the form

$$x_i = \frac{|{\to}i|}{\sqrt{|{\to}i|\ |i{\to}|}}\, x_{{\to}i},$$

where $x_{{\to}i}$ denotes an average over nodes on the in-edges to i. From (8) we have

$$(\nabla x_c)_i = \frac{|i{\to}|}{\sqrt{|{\to}i|\ |i{\to}|}}\, (\nabla x_c)_{i{\to}},$$

where $(\nabla x_c)_{i\to}$ denotes an average over nodes on out-edges of i. The two growth rates, adjusted for the same direction through the network, are now equal:

$$\frac{x_i}{x_{{\to}i}} = \sqrt{\frac{|{\to}i|}{|i{\to}|}} = \frac{(\nabla x_c)_{i\to}}{(\nabla x_c)_i}.$$

Whereas neither x nor $\dot{b} = (-\nabla x_c)$ will have zero layer-to-layer growth when there is a systematic imbalance in the in-degrees and out-degrees, by having equal growth there is a

better chance that deactivations will occur uniformly across layers. Non-zero layer-to-layer growth/decay of variables in a rectified wire network does not present a problem because the equations have no intrinsic scale. Suitable rescalings of the type given at the start of this section, applied after training, can neutralize the layer-to-layer growth without changing the classification behavior of the network.

So far we have only considered the weights of the hidden nodes. The only other nodes with weights are the output nodes. These might be weighted differently based on prior information about the classes, such as their frequency in the data. However, when there is no distinguishing prior information about the classes, the weights on the output nodes should be given equal values. By scale invariance we are free to impose

$$w_i = 1, \quad i \in C.$$

As a tool for the study of rectified wire networks, we introduce a global weight-multiplier hyperparameter q where formula (14) is replaced by

$$w_i = \frac{q}{\sqrt{|{\to}i|\ |i{\to}|}}, \quad i \in H, \tag{15}$$

and $q = 1$ corresponds to balanced weights. The limit $q \to 0$ is interesting because it collapses the rectified wire model to a much simpler one. As explained in section 5, and easily generalized to arbitrary numbers of layers, two things happen in the $q \to 0$ limit: (i) only edges in the final layer ever become inactive, and (ii) all biases except those in the final layer can be neglected. Because of (i), all the layers below the final hidden layer of nodes combine into a single linear layer, resulting in a model with just a single hidden layer. After scaling away the weights, the expression for the value of an output node $k \in C$ takes the form

$$x_k = \sum_{j \in H} \max\left(0, \sum_{i \in D} a_{ji}\, x_i - b_{j \to k}\right), \tag{16}$$

where the integers a_{ji} count the number of paths in the network from an input node i to a hidden layer node j. This model, comprising a single layer of rectifier gates, is of course much easier to analyze than a general, multi-layered rectified wire model. By decreasing q in experiments one can assess the value of depth in the network. In particular, if $q \to 0$ does not compromise performance, then depth is not being utilized in an essential way.

7. Small networks

Because both our network model (rectified wires) and training method (conservative learning) are unconventional, we use this section to demonstrate the model and method on very small networks before turning to more standard demonstrations in section 9. We first describe how the SDA algorithm trains the complete Boolean network on functions of two variables. This is followed by the study of a two-hidden-layer network, for functions on three Boolean variables.

We already proved (section 5) that the 16-edge network shown in Figure 6 learns all 2^{2^2} two-variable Boolean functions with SDA in the limit of vanishing weights in the hidden layer. Here we check that this continues to be true for balanced weights. Thanks to the

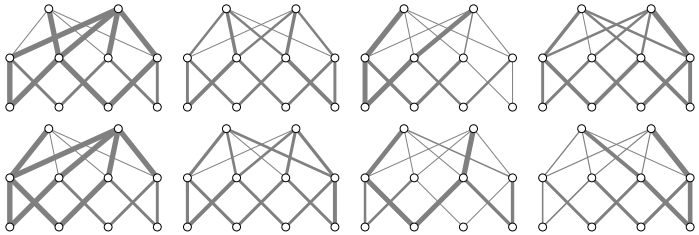


Figure 7: Two examples of final bias settings, in columns, for networks trained on the functions (left to right) f_0, f_1, f_+, $f_\times$ with different data orderings.

symmetry of this network, we only need to check four equivalence classes of functions. Here equivalence is with respect to negating inputs or the output, or swapping the two variables. In all cases, these operations on the function correspond to relabelings of the input and output nodes of the network. Using the algebra of GF(2) for Boolean operations, representatives of the four equivalence classes are

$$\begin{aligned} f_0(z_1, z_2) &= 0 \\ f_1(z_1, z_2) &= z_1 \\ f_+(z_1, z_2) &= z_1 + z_2 \\ f_\times(z_1, z_2) &= z_1 z_2. \end{aligned}$$

We have verified that the four functions above are learned on the 16-edge network with SDA and balanced weights ($q = 1$). The biases were always initialized at zero, as rendered by equal-thickness wires in Figure 6. Recall how the network is trained on a function f with the SDA algorithm. After a data vector $d = (z_1, z_2, \bar{z}_2, \bar{z}_1)$ is forward-propagated, the two output nodes will have equal or unequal values. If unequal, and the correct node — corresponding to class $f(z_1, z_2)$ — is smallest, no biases are changed and the next item is processed. If the smallest output is positive but wrong, or equal to the other output, then the SDA algorithm executes iterations until the correct output is smallest or zero. The network will then have learned d, except for the case where the iterations drive both outputs to zero. This irreversible mode of classification ambiguity does not arise with the 16-edge network and any of the four functions.

In ultra-conservative learning there is no guarantee that successfully learned data is not unlearned in subsequent training. However, by testing on all four data each time biases are changed by SDA, one can establish whether the function is learned. The order in which data are processed matters, as manifested in the final bias values. Figure 7 shows two examples of final bias settings for each of the four functions.

Define a *trial* as a single training experiment where, starting with zero bias, the data are processed in some prescribed order. All trials, for any of the four functions above, are successful on the 16-edge network. That is, there always is a point where the biases stop

changing and the data are correctly and unambiguously classified. Two numbers of interest in a successful trial are (i) $\#_{\text{err}}$, the total number of wrong or ambiguously classified data (invocations of SDA) over the course of training, and (ii) $\#_{\text{iter}}$, the total number of SDA iterations performed. These numbers are equal on the 16-edge network trained on any of the 2-variable functions, as one SDA iteration is sufficient to fix any incorrect classification in this case. Not surprisingly, the constant function f_0 is the easiest to learn, with $\#_{\text{err}} = 1$. All trials with f_1 have $\#_{\text{err}}$ equal to 4 or 5. The XOR function, f_+, always has $\#_{\text{err}} = 3$, while the AND function, $f_\times$, has the greatest variation with $\#_{\text{err}} = 2, 3$ or 4.

The more aggressive variant of SDA, where iterations continue until the class output node is zero, also succeeds for all four functions on the 16-edge network with balanced weights. In this mode of training one pass through the data always suffices. We find, for all four functions, that the network must see all the data ($\#_{\text{err}} = 4$) and $6 \leq \#_{\text{iter}} \leq 9$.

Complete Boolean networks, such as the 16-edge network, quickly become too large for functions of many Boolean variables, and alternative network designs must be considered. For example, the nodes in the single hidden layer could be clauses formed from all small subsets of the variables, or even an incomplete sampling of such clauses. The success of clause learning ($w \to 0$) with such networks would of course depend on the nature of the function.

The opposite limit applied to the hidden-layer weights, $w \to \infty$, suggests a different design for single-hidden-layer networks. Because biases on edges to the outputs never change in this limit, we connect each hidden-layer node to a single output. The hidden-layer nodes are now interpreted as proto-clauses, because their composition in literals is modified during training by changes to the input biases. For example, a proto-clause could sample both a variable and its negation, and rely on training (bias change) to select one or the other. Moving bias changes to the input side of the network alleviates the exponential explosion of static clauses one faces in the $w \to 0$ limit.

The departure from complete Boolean networks we feel is most interesting is increasing the number of hidden node layers. Indeed, keeping training tractable in this setting was the primary motivation for the rectified wire model. To explore this feature we trained a network on three-variable Boolean functions where the hidden layer nodes only have in-degree two. To potentially express relationships among three variables, the hidden nodes are arranged in two layers.

The three-variable network, shown in Figure 8, is complete in the following sense. The nodes in the first hidden layer exhaust the 3×4 ways of collecting inputs from distinct Boolean pairs, with and without negation. One attribute of these nodes is the identity of the Boolean variable — 1, 2 or 3 — that was *not* sampled. The second layer of hidden nodes exhausts all $3 \times 4 \times 4$ combinations of input pairs for which this first hidden layer attribute is different. As in the 16-edge network, the wiring between the final hidden layer and the output nodes is complete in the conventional sense. The resulting network has 216 edges.

Bias renderings, after training, of networks with over 200 edges are not very comprehensible. However, the statistics of the numbers $\#_{\text{err}}$ and $\#_{\text{iter}}$ appear to correlate well with the naive complexity of the Boolean function. We considered three functions that evaluate

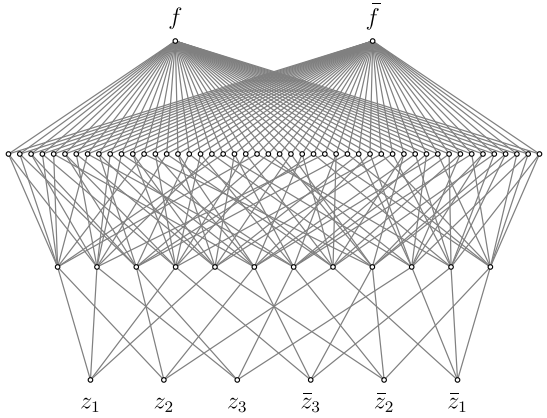


Figure 8: A rectified wire network with two hidden layers that, we conjecture, learns all Boolean functions $f(z_1, z_2, z_3)$ on three variables. All hidden nodes in this network have two inputs.

to 0 and 1 with equal frequency:

$$f_1(z_1, z_2, z_3) = z_1$$
$$f_>(z_1, z_2, z_3) = z_1 z_2 + z_2 z_3 + z_3 z_1$$
$$f_+(z_1, z_2, z_3) = z_1 + z_2 + z_3.$$

Learning f_1, or learning to ignore z_2 and z_3, should be easiest. Function $f_>$, the logical majority function, we expect to be harder because its value, in some but not all cases, is sensitive to all three variables. By the same argument, the parity function f_+ should be the hardest of the three.

All trials we performed on f_1, $f_>$ and f_+, using SDA on the 216-edge network with balanced weights, were successful. Figure 9 shows the distribution of $\#_{\text{err}}$ and $\#_{\text{iter}}$ in 50 trials for each, differing only in the order in which the data are processed. By either metric — the total number of false classifications or the total work in training — the difficulty ranking of the functions is $f_1 < f_> < f_+$. Less extensive trials on the full set of 2^{2^3} Boolean functions all proved successful and leads us to conjecture that the 216-edge network of Figure 8 learns all 3-variable Boolean functions regardless of data order.

We note that the complete Boolean network, with only $2^3(3+2) = 40$ edges, would have been much smaller in this instance and by theorem 5.3 can learn all the Boolean functions in the $w \to 0$ limit. Still, it is reassuring to see that SDA also works in deep networks.

The three-variable Boolean functions also highlight the advantage of the ultra-conservative termination rule for SDA iterations. When we repeat the 50 trials with the stricter zero-

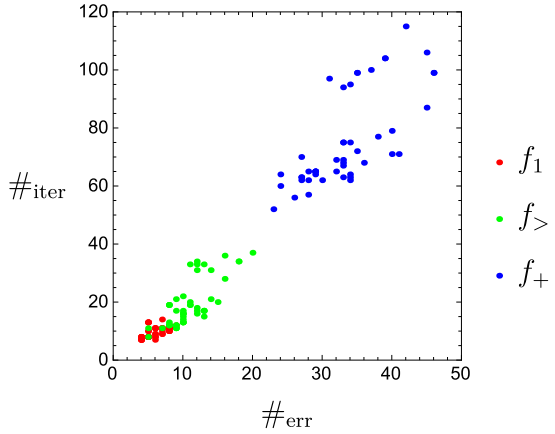


Figure 9: Distribution of $\#_{\text{err}}$ and $\#_{\text{iter}}$ in 50 trials when learning the three-variable Boolean functions f_1, $f_>$ and f_+ on the network shown in Figure 8.

output-value rule ($x_c = 0$), we find that training is not always successful. While training on f_1 was always successful, the success rate on $f_>$ was 76% and on f_+ only 22%. The success rate improves for smaller q and appears to be at 100% for all three functions when $q = 0.5$. Another point in favor of ultra-conservative termination is the greatly reduced number of iterations. For example, by forcing zero outputs, the total number of iterations needed to learn f_1, averaged over data ordering, is about 140, or about 15 times the number needed by the ultra-conservative rule.

8. Sparse expander networks

A nice simplification provided by rectified wire networks is that the training algorithm has very few hyperparameters. Learning succeeds or fails mostly on the basis of the network we choose. However, it is possible that the tractability of rectified wire training comes at the cost of a greater sensitivity to network design. We therefore anticipate that network characteristics — size, depth, sparsity — will take over the role of hyperparameters.

As a general rule, learning improves with larger networks. Since the edge-biases are the only learned parameters of a rectified wire network, the appropriate measure of network size is the number of edges. In this section we introduce a family of networks that adds depth — the number of hidden layers — as a second design parameter.

Layered networks that are completely connected in the conventional sense suffer from a symmetry breaking problem. Since we start with all biases set at zero, a completely connected network has perfect permutation symmetry of its hidden nodes. This symmetry

extends, during training, to the bias updates on all edges between the hidden nodes. To allow for independent settings of the biases, the initial biases must break the symmetry in some data-neutral fashion. Incompletely connected or even sparse networks are an extreme form of symmetry breaking, and correspond to infinite bias settings.

Our network design, called *sparse expander networks*, generalizes the networks of section 7. Sparsity in our case is the property that all the hidden nodes have in-degree of only two. At the same time, the representation of the data is expanded, layer to layer, by a constant growth factor g in the size of layers. The expander structure cannot be applied to the final two layers because of the fixed (and small) number of output nodes. Here the layers are completely connected, thereby making all the information in the last (largest) hidden layer available to each output node. Letting h denote the number of hidden layers, the number of edges in a sparse expander network for $|D|$ inputs and $|C|$ classes, with parameters (g, h), is

$$|E| = |D|(2g + \cdots + 2g^{h-1} + (2 + |C|)g^h). \tag{17}$$

To introduce a degree of uniformity, our networks have a constant out-degree of $2g$ on all input and hidden nodes. This requires that g is an integer. The in-edges to the nodes in a hidden layer are generated by the simplest random algorithm. In a first pass through the hidden layer we create one edge per node to the layer below by drawing uniformly from the nodes that currently have the smaller of two possible out-degrees. A second edge is added in a second pass, at the completion of which all the nodes on the layer below will have out-degree $2g$. Edges from the first hidden layer to the input layer are constructed no differently. This construction is implemented by the publicly available[1] C program `expander` and was used for all the experiments reported in the next section.

We can use the hyperparameter q to assess the role of depth. If performance does not degrade in the limit $q \to 0$, then the network could have been replaced by the much simpler single-hidden-layer model (16). When this limit is applied to a sparse expander network, the number of hidden nodes is $|H| = |D|g^h$. The sampling of the fixed weights a_{ji} in the rectifier gates is also controlled by g and h.

9. Experiments

To describe experiments with conservatively trained rectified wire networks it suffices to specify the network. The SDA algorithm has no other hyperparameters, since there even is a balancing principle that sets the weights. The two-parameter sparse expander networks offer a convenient way to study behavior both with respect to network size and depth. These characteristics were our main focus and guided the choice of experiments going into this study. Follow-up experiments, that featured the weight-multiplier hyperparameter q, later provided critical insights for network design that have yet to be explored.

All our experiments were carried out with a publicly available[1] C implementation of the SDA algorithm called `rainman`. This program requires that the user-supplied network is layered and the number of inputs and outputs are compatible with the data. Our networks were all created by the program `expander`.

1. `github.com/veitelser/rectified-wires`

Although `rainman` reports results at regular intervals, after a specified number of data have been processed, this "batch size" has no bearing on the actual training because bias parameters are only updated in response to wrong classifications. The batch size merely sets the frequency with which the network is tested against a body of test-data. The main result of this test is the classification accuracy: the test-data average of the indicator variable that is 1 for correct, unambiguous classifications, and $1/m$ when the correct class node is in an m-fold tie.

In addition to the classification accuracy, `rainman` also reports a number of other quantities of interest. Two we have already seen in section 7: the running total of misclassifications, $\#_{\text{err}}$, and the work (iterations) performed so far by SDA, $\#_{\text{iter}}$. These should increase while the classification accuracy is below 100%. However, in an "overfitting" situation $\#_{\text{err}}$ and $\#_{\text{iter}}$ stop increasing before the accuracy has reached 100%. Evidence of this phenomenon is more directly discerned through another statistic reported by `rainman`: the average number of SDA iterations required to fix each of the incorrect classifications encountered in the batch, $\langle \#_{\text{iter}} \rangle$. As defined, $\langle \#_{\text{iter}} \rangle$ is never less than 1. However, when there are no misclassifications in the batch at all (keeping $\#_{\text{err}}$ and $\#_{\text{iter}}$ static), the value $\langle \#_{\text{iter}} \rangle = 0$ is reported.

Information more revealing of the internal workings of the network is provided by the layer-averages of the edge activations, $\langle \alpha_1 \rangle, \ldots, \langle \alpha_h \rangle$. Here $\langle \alpha_h \rangle$ is the fraction of edges, in the layer leading to the output nodes, that are active in an average test-data item. These numbers decrease during training, a result of biases being increased.

Lastly, a statistic relevant to the termination of training is the frequency, in the test-data, of irreversible misclassifications, $\langle 0_{\text{err}} \rangle$. The latter arise whenever an output node, not of the correct class, has value zero. The onset of $\langle 0_{\text{err}} \rangle > 0$ signals that training should terminate because activation levels are so low that SDA is forced to make multiple outputs zero even while targeting just the class output. A simple remedy for preventing $\langle 0_{\text{err}} \rangle > 0$ is to increase the size of the network. The case of m-fold classification ambiguities, mentioned above, occurs in practice only when m output nodes are tied at zero. Ambiguous classification is therefore not a problem as long as $\langle 0_{\text{err}} \rangle > 0$.

On a single Intel Xeon 2.00GHz core `rainman` runs at a rate of 50ns per iteration per network edge. Multiplying this number by the product of $\#_{\text{iter}}$ and $|E|$ (17) gives a good estimate of the wall-clock time in all our experiments.

9.1. Nested majority functions

We first describe experiments with a synthetic data set. Synthetic data has some advantages over natural data. Our *nested majority functions* (NMF) data has two nice features. First, through the level of nesting we can systematically control the complexity of the data. Second, because these Boolean functions are defined for all possible Boolean arguments, it is easy to generate large training sets for supervised learning.

NMFs are a parameterized family of Boolean functions that evaluate to 0 and 1 with equal frequency. The first parameter, a prime p, sets the number of arguments at $p-1$. Instances are defined by three integers $a, b, c \in \{1, \ldots, p-1\}$ and a nesting level $n = 0, 1, 2, \ldots$. Starting with

$$f_a^0(z_1, \ldots, z_{p-1}) = z_a,$$

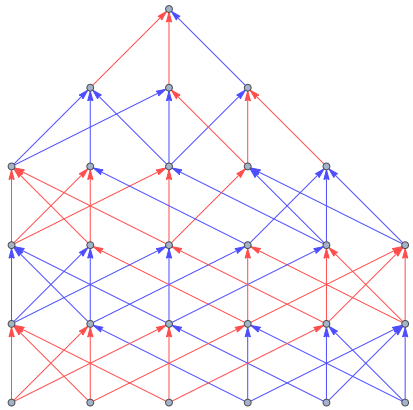


Figure 10: An instance of a depth 5 nested majority function on 6 variables. The nodes are majority gates and the two edge colors indicate the presence/absence of negation. The depth $4, \ldots, 1$ functions correspond to removing $1, \ldots, 4$ layers from the bottom of the circuit.

the other functions (taking values in GF(2)) are defined via the recurrence

$$f_a^n = f_> \left(f^{n-1}_{(ab \bmod p)} + z(ac) \, , \; f^{n-1}_{(2ab \bmod p)} + z(2ac) \, , \; f^{n-1}_{(3ab \bmod p)} + z(3ac) \right) ,$$

where $f_>$ is the 3-argument majority function and

$$z(x) = (x \bmod p) \mod 2$$

should be interpreted as an element of GF(2). The functions f^n at level n correspond to depth-n Boolean circuits taking $p-1$ inputs and constructed from NOT and MAJORITY gates. Figure 10 shows an instance of the function f^5 for $p = 7$.

From the identity

$$f_>(z_1 + 1, z_2 + 1, z_3 + 1) = f_>(z_1, z_2, z_3) + 1$$

it follows that negating all the arguments of a NMF negates its value. The two classes, defined by the NMF value, are therefore equal in size.

Our experiments used data generated by the functions f_1^n for $p = 31$ and $b = 2$, $c = 3$. The easiest of these, with $n = 0$, only depends on a single argument. To learn this function the rectified wire network (taking 2×30 inputs) must identify the correct one and ignore all the others. In the case of f_1^1 it must select three of the arguments and also whether to apply negations. The functions f_1^2, f_1^3, etc. become progressively harder as they depend on

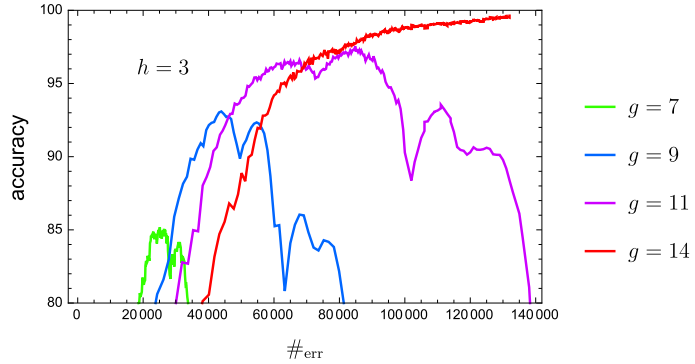


Figure 11: Improvement of the classification accuracy with network size, for the function f_1^3, on rectified wire networks with three hidden layers.

more arguments and with more complicated rules. The possibility of generalizing the value of f^n, from many fewer than 2^{p-1} data, is clearly only possible for small n. Our results go as far as $n = 4$.

We generated training and test data samples (of the 30 Boolean variables) with a pseudo-random number generator. In the longest experiment, with 10^7 training samples, less than 1% of the data was seen more than once by the training algorithm.

We first present results for the function f_1^3. Fixing the number of hidden layers at $h = 3$, Figure 11 shows the effect of increasing the network size via the growth factor g. The accuracy is plotted as a function of $\#_{\text{err}}$, rather than the number of training data, because learning (change of bias) occurs only when there has been a misclassification. In all except the $g = 14$ network (683760 edges), the accuracy reaches a maximum after which there is a sharp decline that coincides with the onset of $\langle 0_{\text{err}} \rangle > 0$. The smaller networks could still serve as imperfect classifiers by terminating their training at the empirical maximum test accuracy, or, in the absence of test data, when $\langle 0_{\text{err}} \rangle$ exceeds a threshold. We have no explanation of the intriguing similarity of the unsuccessful accuracy plots. We also find that many of the seemingly random features in the accuracy plots are nearly reproduced in different random instances of expander networks with the same (h, g).

We emphasize that it is not really surprising that generalization performance can drastically deteriorate with more training examples for small networks. Conservative learning is using a fundamentally different principle than the standard approach that tries to minimize an overall loss function across training examples. Conservative learning is merely trying to myopically account for the most recently seen example, which can have negative consequences for other training examples, to say nothing of unseen test examples.

The effect of changing network depth is shown in Figures 12 and 13. Although the three featured networks have about the same number of edges, only the two deeper networks are

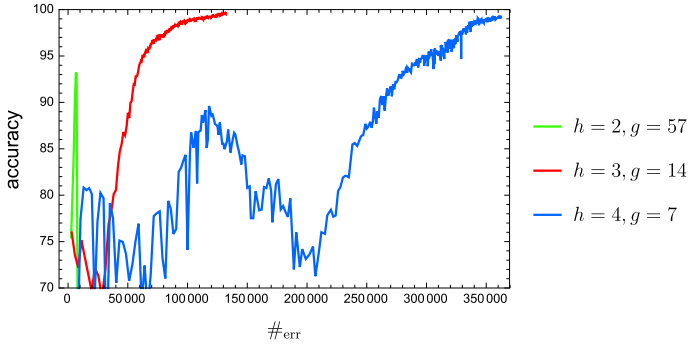


Figure 12: Of three networks with comparable size trained on the function f_1^3, the deeper networks are more successful and require more data.

successful. Figure 12, with accuracy plotted against $\#_{\text{err}}$, shows that learning is faster on shallower networks. On the other hand, when plotted against $\#_{\text{iter}}$ in Figure 13, we see that less work is required to train the deeper networks. Work, as measured in SDA iterations, is not evenly distributed over the course of training. Figure 14 shows $\langle \#_{\text{iter}} \rangle$ plotted against $\#_{\text{err}}$. More iterations are needed early in training, especially so in shallow networks.

The balanced weight heuristic is far from successful in keeping the layer-averaged activation levels uniform. After training the $(h, g) = (3, 14)$ network, we find $\langle \alpha_3 \rangle = 1.1\%$, $\langle \alpha_2 \rangle = 47\%$ with the rest at 100%. For the deeper and also successful $(h, g) = (4, 7)$ network, the diminished activations are $\langle \alpha_4 \rangle = 4.1\%$, $\langle \alpha_3 \rangle = 57\%$, $\langle \alpha_2 \rangle = 76\%$.

The lower complexity of the functions f_1^n for $n < 3$ is reflected in the smaller networks required to learn them. On networks with two hidden layers ($h = 2$), f_1^2 is successfully learned with $g = 21$ (108360 edges) after only 5000 misclassifications. For f_1^1 the corresponding numbers are $g = 6$ (9360 edges) and $\#_{\text{err}} = 900$. On the other hand, if we train f_1^4 on the same network ($h = 4$, $g = 7$) that learned f_1^3 with $\#_{\text{err}} \approx 4 \times 10^5$, the best accuracy, achieved for roughly the same $\#_{\text{err}}$, is only 91.9%.

The prediction accuracy on NMF data, for all of our networks, is improved by decreasing the weight-multiplier hyperparameter q. In fact, it appears that $q \to 0$ is optimal, or that the best performance can be achieved with the model (16) comprising a single layer of rectifier gates with fixed weights a_{ji}. A striking case is the function f_1^3 trained on the sparse expander graph with $h = 2, g = 57$, already encountered in Figure 12. Figure 15 shows the effect of decreasing q from the balanced setting, $q = 1$. We see that the 94% accuracy of the $q = 1$ network reaches 100% already for $q = 0.7$. It seems the only downside in decreasing q is an increase in the number of training data (greater $\#_{\text{err}}$). The $q \to 0$ weight "codewords" a_{ji} associated with the sparse expander network are very simple. Because the network has in-degree two and depth $h = 2$, almost all the rectifiers see four random inputs that are

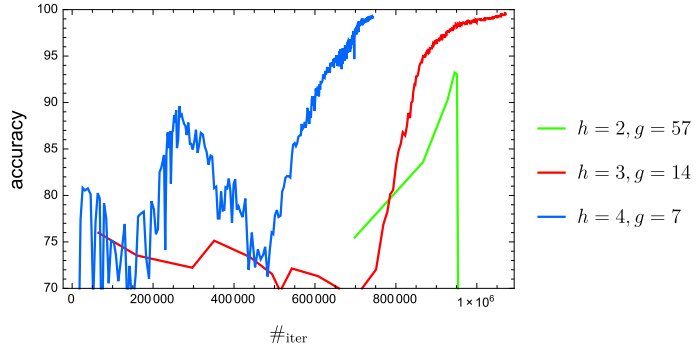


Figure 13: Same as Figure 12, except plotted against the total work (SDA iterations).

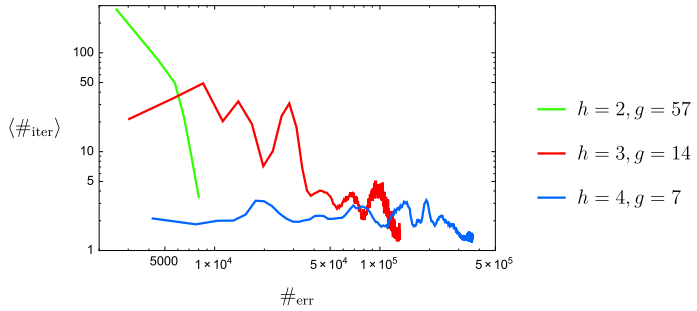


Figure 14: Average number of iterations used by the SDA algorithm to fix a wrong classification over the course of training for the three networks featured in Figs. 12 and 13.

given equal weight. It is likely that the performance of model (16) can be further improved with weight codewords that go beyond those generated by sparse expander networks.

In light of the $q \to 0$ findings, it is not surprising that conventional neural networks, even with just a single hidden layer, easily outperform the rectified wire model on the NMF data. A fully-connected ReLU network with one layer of only 50 hidden nodes was sufficient to learn f_1^3 completely after seeing 5×10^5 samples. This can be compared with the 2×10^4 samples ($\#_{\text{err}}$) that informed bias changes in the rectified wire model when learning the same data (Figure 15). The vast reduction in the number of hidden nodes, from 2×10^5 (penultimate layer of the $h = 2, g = 57$ network) to 50, no doubt is attributable to the

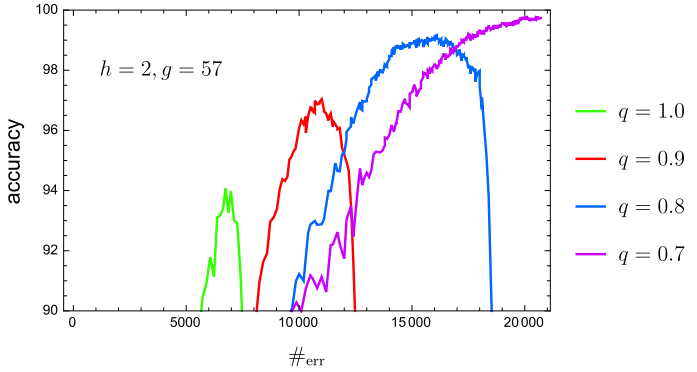


Figure 15: Evolution of the prediction accuracy with the weight-multiplier hyperparameter q on the $h = 2, g = 57$ network for the function f_1^3.

standard model's capacity for "designing" the weight-codewords that in model (16) are static.

9.2. MNIST

Seen as images, the MNIST handwritten digits (LeCun et al., 1998) are analog data. Because the contrast in these images is well approximated as two-valued, it makes sense to map pixels to a two-symbol alphabet and train a rectified wire network on data of that type. We report results with this approach at the end of this section. However, we first try processing MNIST non-symbolically, using the analog data vectors described in equation (2).

One motivation for the analog approach is that MNIST images are highly compressible. Playing to a possible strength of rectified wire networks, it might be advantageous to first map MNIST images, and images in general, to data vectors the components of which are more nearly statistically independent. We use a method based on principal component analysis.

Using singular value decomposition, we approximate the 60000×28^2 matrix X of flattened MNIST training images as the product $X \approx UV$, where the rows of the $r \times 28^2$ matrix V are the orthonormal "eigen-images" associated with the r largest singular values of X. A flattened image x (training or testing) is mapped to r analog channels by $v = Vx$. Applying the component-wise cumulative probability distribution functions to this v then gives us the $2r$-component data vector as in equation (2). We compute the cumulative probability function from the training data and use the same function when processing the test data (with test samples below the minimum or above the maximum training samples mapped to 0 and 1 respectively).

Figure 16 shows the test accuracy when training a sparse expander network on analog MNIST data with $r = 50$ channels. This network has 205400 edges, the same number a conventional network would have with a single fully-connected layer of 259 hidden nodes. After making about $\#_{\text{err}} = 25000$ misclassifications, there are no further misclassifications on the training data and training ceases. The test accuracy attains a maximum of 94.54% at this point. Figure 17 shows the evolution of $\langle \#_{\text{iter}} \rangle$ over the course of training. Generalization (test accuracy) improves when we decrease the q hyperparameter from the value $q = 1$ for balanced weights. However, the improvement, reaching 94.99% at $q = 0.2$, is much more modest than we found for the NMF data. The time used by `rainman` in any of these training runs was about 65 minutes. Although about one million data (16 epochs) are processed, only for about 25000 of these does SDA do anything.

We also used the analog MNIST data to compare SDA training with stochastic gradient descent (SGD) training on the same model and network architecture. Because the sparse networks we use for SDA are a poor fit for GPU-based software, we wrote a C++ SGD optimizer that, like SDA, runs on a single core without calls to a GPU. In the strictest comparison SGD used the same monotone loss function, x_c, used by SDA. For SGD we also tried multi-class hinge loss, with margin parameter 0.1. As discussed in section 3, this loss spoils tractability (monotone bias changes) and is therefore not applicable for SDA. The mini-batch size for SGD was fixed at 100 and we employed standard stochastic gradient descent without momentum.

SGD does poorly when forced to use SDA's monotone loss. With the learning rate set at 0.002, training accuracy reaches a maximum of 91.5% in 6 minutes; this is also the test accuracy for this mode of operation. Since SDA uses adaptive gradient steps (set by deactivations), we also tried SGD with a ten-fold reduction in learning rate. While training now goes through 97 epochs of data and takes about as long as SDA, there is no improvement in either the training or test accuracy. We interpret this as supporting the conservative learning principle: there are gains when parameters are optimally adjusted in response to individual data.

The performance of SGD improves significantly when the rectified wire model is trained with multi-class hinge loss. However, due to the structure of expander networks, this loss is problematic when applied exclusively. When evaluated on the initial expander network (all edges active), the partial derivatives of the loss with respect to all biases, except those on edges to the outputs, are identically zero. This occurs because the effect of all bias changes (except in the final layer) passes through the penultimate layer of nodes that is completely connected to the outputs, and equal changes on these leaves the hinge loss unchanged. To avoid the zero derivative problem we used a hybrid approach, where hinge loss was used only after some number of data had first been processed with the monotone x_c loss to create deactivations that break the symmetry in the final layer. Switching to hinge loss after 5 epochs had no effect on accuracy, but switching after 10 epochs improved training and test accuracies to 94.4% and 94.1%, respectively. This trend in improvement continues and reaches 96.8% (training) and 95.6% (test) when the switch is made after 20 epochs (121 minutes). We attribute the modest improvement in generalization (0.6%), over the SDA trained model, to the margins that hinge loss is able to insert in the class boundaries.

We do not believe the overfitting we saw with SDA was caused by limiting the analog data vectors to only 50 channels, because essentially the same classification accuracy is

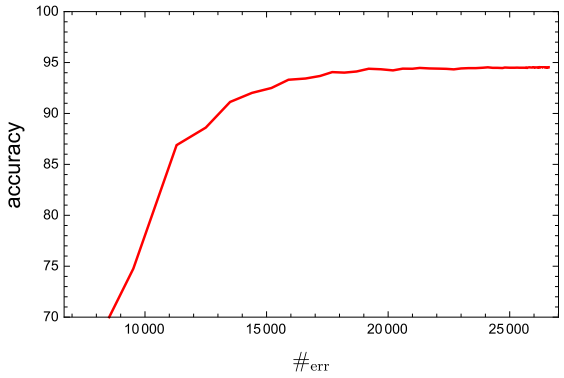


Figure 16: Classification accuracy of MNIST images from 50-principal-component data vectors with the SDA algorithm on a sparse expander network with parameters $h = 2$, $g = 13$.

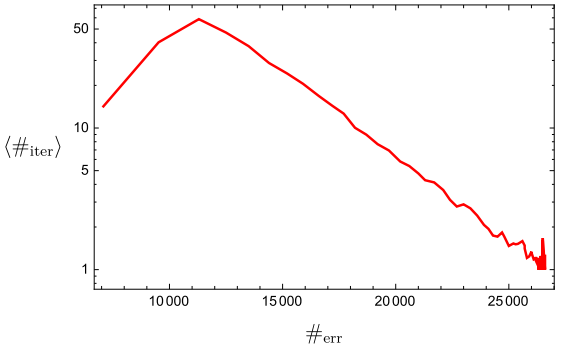


Figure 17: SDA iterations per misclassification for the same data and network as in Figure 16

.

obtained with the uncompressed symbolic data. The latter was generated from the raw MNIST images by thresholding pixel values at 40% of the maximum (the minimum point of the pixel histogram) to define the two symbols. Figure 18 shows the evolution of the accuracy at fixed depth $h = 4$ as the network size increases. In the smallest network ($g = 1$),

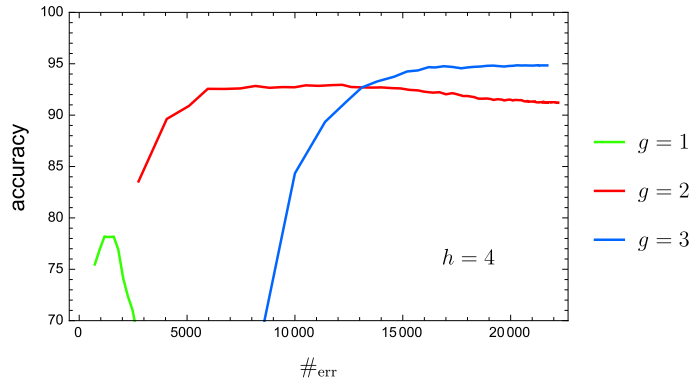


Figure 18: Classification accuracy of binarized MNIST images on rectified wire networks with four hidden layers.

as expected, the accuracy drops precipitously at the onset of $\langle 0_{\rm err}\rangle > 0$. However, already in the next network ($g = 2$) we have the situation where training terminates because most (training) data are correctly classified and the remaining ambiguous cases are ties at zero (in output) and untrainable. At termination we find $\langle 0_{\rm err}\rangle = 2.7\%$, contributing to the reduction in accuracy. Increasing the network size further ($g = 3$) gives accuracy 94.8% at termination, where now $\langle 0_{\rm err}\rangle = 0$. This accuracy is close to that obtained with the analog data.

Although our results for MNIST are an improvement over the simple linear classifier with accuracy 88%, and even a pairwise-boosted linear classifier (92.4%), the appropriate comparison should be with other tractable methods. Of these, we note that support vector machines with a Gaussian kernel already achieve 98.6% accuracy on MNIST. These and other techniques for MNIST are reviewed in LeCun et al. (1998).

9.3. Markov sequences

As our final application we study a synthetic symbolic dataset where, unlike the nested majority function data, samples are generated probabilistically. The data are strings in the alphabet {A,B,C,D} generated by Markov chains. Examples of 12-symbol strings from the two classes are shown in Table 1. Those in class 1 are extracts from the Markov chain defined by the transition matrix

$$T = \frac{1}{10}\begin{bmatrix} 1 & 2 & 1 & 6 \\ 3 & 1 & 4 & 2 \\ 4 & 1 & 4 & 1 \\ 2 & 6 & 1 & 1 \end{bmatrix}. \qquad (18)$$

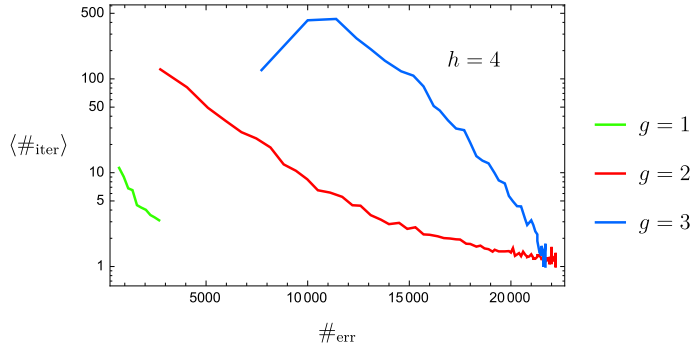


Figure 19: SDA iterations per misclassification for the same data and networks shown in Figure 18.

class 1	class 2
CADCCCACCADA	ACCBDBABDBCC
ADABDCCCDACB	ABDAABACCDDB
CBCCADBBCADB	BDBDADACDACD
DADBDADDBDCA	DBADCCADADDA
BCAADBBABCDB	BDACCCDABBDB
ABDDBADCABBD	CCCBDABCADBD

Table 1: Examples of 12-symbol strings extracted from the Markov chain with transition matrix (18), in class 1, and its transpose, in class 2.

Since T is doubly stochastic, its transpose defines another Markov chain and is the source of the strings in class 2. The doubly stochastic property also ensures that the four symbols occur with equal frequency. Finally, the absence of zeros means that all bigrams have finite probability.

Even a machine that managed to reconstruct T from the data could never have perfect classification accuracy, because every 12-symbol string occurs with finite probability in either chain. The optimal classifier selects the chain (class) that gives each string the highest probability. With this criterion applied to our T, the true-positive rate for 12-symbol strings is 95.1%. We also trained networks on 25-symbol strings, for which the true-positive rate is 99.1%. These numbers set the maximum achievable classification accuracy.

In Figures 20 and 21 we compare classification accuracies for 12- and 25-symbol strings as learned on networks having approximately 2×10^5 edges. We see that here the depth of the network has very little effect on the maximum accuracy. By the probabilistic nature of

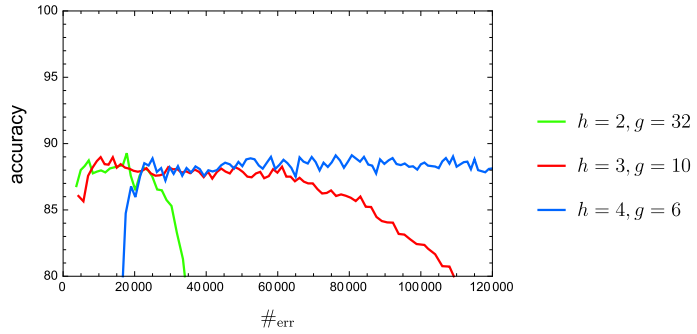


Figure 20: Classification accuracy of 12-symbol strings generated by the Markov chain (18) or its transpose.

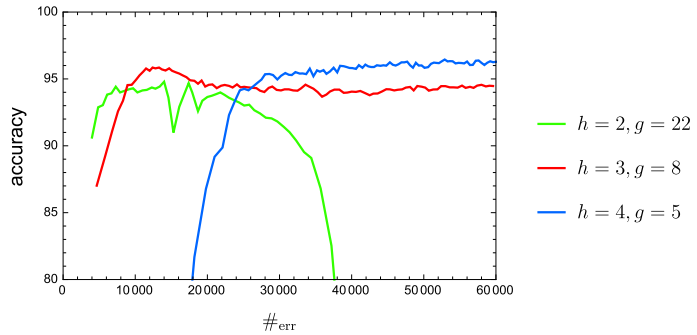


Figure 21: Classification accuracy of 25-symbol strings generated by the Markov chain (18) or its transpose.

the data, these accuracies must eventually decline as strings having higher probability, in the class with the other label, are encountered. Deeper networks are less vulnerable to this phenomenon. The highest accuracy achieved for 12-symbol strings is 6% below that of the optimal classifier; for 25-symbol strings the shortfall is 3%.

We did not explore the effect of decreasing the q hyperparameter for this data set.

10. Discussion

The rectified wire model is a significant departure from the "standard model" of neural networks. Even so, the training algorithm we derived for this model, SDA, deserves comparison with the leading algorithm for the standard model: stochastic gradient descent (SGD) (Rumelhart et al., 1986). In both algorithms the network parameters are updated after only a small amount of data has been seen: a single item in the case of SDA, a mini-batch for SGD. The computation of updates involves propagating data vectors in a forward pass, and the class of the data — or the associated loss — in a backward pass. SDA makes an extra forward pass that determines the size of each update, signaled by a deactivation event. In SGD the size of the update is set by the learning rate, a hyperparameter, and the update itself is an average over the mini-batch, whose size is another hyperparameter.

The characteristic of SDA that most differentiates itself from SGD is the monotone evolution of the network parameters. Ostensibly this is problematic: there is no "going back" on any updates, including those made early when very little of the data had been seen. This tacitly assumes the learned state of the network is unique, or nearly so. But an alternative hypothesis is also worth considering: a high multiplicity of learned states, all equally viable. Our experiments with SDA support this hypothesis. Very different final bias settings, all giving perfect classification, were obtained even for simple classes just by changing the order of the data. Our experiments with Markov-chain generated strings showed that SDA is also not tripped up by outlier data. Although the final network in this probabilistic setting must have 100% classification ambiguity, the accuracy in practice is near that of the optimal classifier long before this asymptotic state is reached.

Not all our experiments with SDA were successful. On the MNIST dataset the algorithm suffered from overfitting: whereas the network had learned to perfectly classify the training data, the classification accuracy on the test data was only 95%. Does this reflect a fundamental deficiency in the rectified wire model, and the conservative learning principle more generally, or is it a result of poor execution?

An even more serious problem with the rectified wire model was exposed, in experiments, upon decreasing the weight-multiplier hyperparameter q. Recall that the primary motivation for the model was tractable training in the presence of network depth. Significant simplifications of the standard neural network model, such as fixing the weights and replacing loss by a constraint, were key in facilitating tractable training. Even so, the expressivity provided by the remaining bias parameters remains high (arbitrary Boolean functions), and the training algorithm (SDA) fully utilizes activations (ReLU nonlinearities) in every layer of the network. The q parameter was introduced to explore behavior with respect to depth: small/large q favors bias changes near the output/input nodes. Though we studied a very limited class of networks — sparse expander nets — the experimental results pointed uniformly in favor of $q \to 0$. In this limit the rectified wire model reduces to a much simpler model (16), comprising a single layer of rectifier gates. While this limiting model is still non-trivial and interesting, it definitely lacks depth.

Our experience with the rectified wire model highlights the difference between the capacity for depth and the learnability of depth. Although there exist bias settings in the model that efficiently represent complex Boolean functions and utilize activations on all levels, experiments show that these are not the kinds of bias settings that are learned by

the tractable, conservative route. At least that is the conclusion we should draw from the experiments performed so far.

There are still some directions to explore before we can conclude that conservatively trained rectified wire networks lack the capacity to learn "in depth". The most obvious is to try architectures very different from the sparse expander networks considered so far. These need not even be layered, although they must break symmetry if we continue to initialize all the bias parameters at zero. Specialized architectures may be inherently better at exploiting depth, as in the case of standard model networks with convolutional layers. Next, we recall that the bias updates given by the SDA algorithm can fall short of the most conservative possible. While we do not know whether the quadratic program (4) can be solved by something as efficient as the SDA algorithm, experiments with general-purpose solvers could study the effect of update quality on training. By the same route one could study the effect of learning, in aggregate, entire mini-batches (section 3, conclusions). Another avenue is to define "conservative" by the 1-norm on the bias updates. Our choice of the 2-norm was motivated by the uniqueness of the optimal updates. However, it seems likely that in the mini-batch setting even the 1-norm will have unique optima. Finally, it might be interesting to allow the weights in the network to vary slowly over the course of training. This could be as simple as favoring different depths over time using the q parameter, or a more sophisticated scheme based on running averages.

Acknowledgments

We thank Alex Alemi, Sergei Burkov, Pat Coady, Cris Moore and Mehul Tikekar for discussions over the course of this project. We also thank the three reviewers whose comments brought significant improvements to the paper.

References

Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7(Mar):551–585, 2006.

Veit Elser. A network that learns strassen multiplication. *Journal of Machine Learning Research*, 17(116):1–13, 2016.

Mikhail K Kozlov, Sergei P Tarasov, and Leonid G Khachiyan. The polynomial solvability of convex quadratic programming. *USSR Computational Mathematics and Mathematical Physics*, 20(5):223–228, 1980.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Raymond J Mooney. Encouraging experimental results on learning cnf. *Machine Learning*, 19(1):79–92, 1995.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533, 1986.

Bernard Widrow, Rodney G Winter, and Robert A Baxter. Layered neural nets for pattern recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(7):1109–1118, 1988.

Journal of Machine Learning Research 20 (2019) 1-6 Submitted 6/18; Revised 1/19; Published 2/19

Pyro: Deep Universal Probabilistic Programming

Eli Bingham ELI.BINGHAM@UBER.COM
Jonathan P. Chen JPCHEN@UBER.COM
Martin Jankowiak JANKOWIAK@UBER.COM
Fritz Obermeyer FRITZO@UBER.COM
Neeraj Pradhan NPRADHAN@UBER.COM
Theofanis Karaletsos THEOFANIS@UBER.COM
Rohit Singh ROHITS@UBER.COM
Paul Szerlip PAS@UBER.COM
Uber AI Labs
Uber Technologies, Inc.
555 Market St., San Francisco, CA, 94103
Paul Horsfall HORSFALLP@GMAIL.COM
Noah D. Goodman NGOODMAN@STANFORD.EDU
Uber AI Labs
555 Market St., San Francisco, CA 94103
Stanford University, Stanford, CA, USA

Editor: Antti Honkela

Abstract

Pyro is a probabilistic programming language built on Python as a platform for developing advanced probabilistic models in AI research. To scale to large data sets and high-dimensional models, Pyro uses stochastic variational inference algorithms and probability distributions built on top of PyTorch, a modern GPU-accelerated deep learning framework. To accommodate complex or model-specific algorithmic behavior, Pyro leverages Poutine, a library of composable building blocks for modifying the behavior of probabilistic programs.

Keywords: Probabilistic programming, graphical models, approximate Bayesian inference, generative models, deep learning

©2019 Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-403.html


```python
def model():
  loc, scale = torch.zeros(20), torch.ones(20)
  z = pyro.sample("z", Normal(loc, scale))
  w, b = pyro.param("weight"), pyro.param("bias")
  ps = torch.sigmoid(torch.mm(z, w) + b)
  return pyro.sample("x", Bernoulli(ps))


def guide(x):
  pyro.module("encoder", nn_encoder)
  loc, scale = nn_encoder(x)
  return pyro.sample("z", Normal(loc, scale))
```

```python
def conditioned_model(x):
  return pyro.condition(model, data={"x": x})()

optimizer = pyro.optim.Adam({"lr": 0.001})
loss = pyro.infer.Trace_ELBO()

svi = pyro.infer.SVI(model=conditioned_model,
                     guide=guide,
                     optim=optimizer,
                     loss=loss)

losses = []
for batch in batches:
    losses.append(svi.step(batch))
```

Figure 1: A complete Pyro example: the generative model (`model`), approximate posterior (`guide`), constraint specification (`conditioned_model`), and stochastic variational inference (`svi`, `loss`) in a variational autoencoder. `encoder` is a `torch.nn.Module` object. `pyro.module` calls `pyro.param` on every parameter of a `torch.nn.Module`.

1. Introduction

In recent years, richly structured probabilistic models have demonstrated promising results on a number of fundamental problems in AI (Ghahramani (2015)). However, most such models are still implemented from scratch as one-off systems, slowing their development and limiting their scope and extensibility. Probabilistic programming languages (PPLs) promise to reduce this burden, but in practice more advanced models often require high-performance inference engines tailored to a specific application. We identify design principles that enable a PPL to scale to advanced research applications while retaining flexibility, and we argue that these principles are fully realized together in the Pyro PPL.

2. Design Principles

First, a PPL suitable for developing state-of-the-art AI research models should be **expressive**: it should be able to concisely describe models with have data-dependent internal control flow or latent variables whose existence depends on the values of other latent variables, or models which only be defined in closed form as unnormalized joint distributions.

For a PPL to be practical, it must be **scalable**: its approximate inference algorithms must be able to seamlessly handle the large data sets and non-conjugate, high-dimensional models common in AI research, and should exploit compiler and hardware acceleration to approach the performance of optimized bespoke implementations of these models.

A PPL targeting research models should be **flexible**: in addition to scalability, many advanced models require inference algorithms with complex, model-specific behavior. A PPL should enable researchers to quickly and easily implement such behavior and should enforce a separation of concerns between between model, inference, and runtime implementations.

Finally, a PPL targeting researchers as users should strive to be **minimal**: in order to minimize cognitive overhead, it should share most of its syntax and semantics with existing languages and and systems and work well with other tools such as libraries for visualization.

As is clear from Table 2, these four principles are often in conflict, with one being achieved at the expense of others. For example, an overly flexible design may be very difficult to implement efficiently and scalably, especially while simultaneously integrating a new language with existing tools. Similarly, enabling development of custom inference algorithms may be difficult without limiting model expressivity. In this section, we describe the design choices we made in Pyro to balance between all four objectives.

Pyro is embedded in Python, and Pyro programs are written as Python functions, or callables, with just two extra language primitives (whose behavior is overridden by inference algorithms): `pyro.sample` for annotating calls to functions with internal randomness, and `pyro.param` for registering learnable parameters with inference algorithms that can change them. Pyro models may contain arbitrary Python code and interact with it in arbitrary ways, including expressing unnormalized models through the `obs` keyword argument to `pyro.sample`. Pyro's language primitives may be used with all of Python's control flow constructs, including recursion, loops, and conditionals. The existence of random variables in a particular execution may thus depend on any Python control flow construct.[1]

Pyro implements several generic probabilistic inference algorithms, including the No U-turn Sampler (Hoffman and Gelman (2014)), a variant of Hamiltonian Monte Carlo. However, the primary inference algorithm is gradient-based stochastic variational inference (SVI) (Kingma and Welling (2014)), which uses stochastic gradient descent to optimize Monte Carlo estimates of a divergence measure between approximate and true posterior distributions. Pyro scales to complex, high-dimensional models thanks to GPU-accelerated tensor math and reverse-mode automatic differentiation via PyTorch, and it scales to large data sets thanks to stochastic gradient estimates computed over mini-batches of data in SVI.

Some inference algorithms in Pyro, such as SVI and importance sampling, can use arbitrary Pyro programs (called *guides*, following webPPL) as approximate posteriors or proposal distributions. A guide for a given model must take the same input arguments as the model and contain a corresponding sample statement for every unconstrained sample statement in the model but is otherwise unrestricted. Users are then free to express complex hypotheses about the posterior distribution, e.g. its conditional independence structure. Unlike webPPL and Anglican, in Pyro guides may not depend on values inside the model.

Finally, to achieve flexibility and separation of concerns, Pyro is built on Poutine, a library of effect handlers (Kammar et al. (2013)) that implement individual control and book-keeping operations used for inspecting and modifying the behavior of Pyro programs, separating inference algorithm implementations from language details.

3. Project Openness and Development

Pyro's source code is freely available under an MIT license and developed by the authors and a community of open-source contributors at `https://github.com/uber/pyro` and documentation, examples, and a discussion forum are hosted online at `https://pyro.ai`. A comprehensive test suite is run automatically by a continuous integration service before code is merged into the main codebase to maintain a high level of project quality and usability.

We also found that while PyTorch was an invaluable substrate for tensor operations and automatic differentiation, it was lacking a high-performance library of probability

1. See the `geometric` example in `http://pyro.ai/examples/intro_part_i.html`

distributions. As a result, several of the authors made substantial open-source contributions upstream to PyTorch Distributions,[2] a new PyTorch core library inspired by TensorFlow Distributions (Dillon et al. (2017)).

4. Existing Systems

System	Expressivity: Dynamic control	Scalability: Subsampling, AD	Flexibility: Flexible inference	Minimality: Host language
Stan	Static control flow	Some, CPU	Automated	None
Church	Yes	No, None	Automated	Scheme
Venture	Yes	No, None	Yes	None
webPPL	Yes	No, CPU	Some	JavaScript
Edward	Static control flow	Yes, CPU/GPU	Yes	TensorFlow
Pyro	Yes	Yes, CPU/GPU	Yes	Python, PyTorch

Figure 2: Simplified summary of design principles of Pyro and some other PPLs.

Probabilistic programming and approximate inference are areas of active research, so there are many existing probabilistic programming languages and systems. We briefly mention several that were especially influential in Pyro's development, regretfully omitting (due to space limitations) many systems for which simple direct comparisons are more difficult. We also emphasize that, as in conventional programming language development, our design decisions are not universally applicable or desirable for probabilistic programming, and that other systems purposefully make different tradeoffs to achieve different goals.

Stan (Carpenter et al. (2017)) is a domain-specific language designed for describing a restricted class of probabilistic programs and performing high-quality automated inference in those models. Church (Goodman et al. (2008)), a probabilistic dialect of Scheme, was an early universal probabilistic programming language, capable of representing any computable probability distribution. Venture (Mansinghka et al. (2018)) is a universal language with a focus on expressiveness and flexibility and a custom syntax and virtual machine. Anglican (Tolpin et al. (2016)) and webPPL (Goodman and Stuhlmüller (2014)) are lightweight successors to Church, embedded as syntactic subsets (Wingate et al. (2011)) in the general-purpose programming languages Clojure and JavaScript. Edward (Tran et al. (2017)) is a PPL built on static TensorFlow graphs that features composable representations for models and inference. ProbTorch (Siddharth et al. (2017)) is a PPL built on PyTorch with a focus on deep generative models and developing new objectives for variational inference. Turing (Ge et al. (2018)) is a PPL embedded in Julia featuring composable MCMC algorithms.

5. Experiments

To demonstrate that Pyro meets our design goals, we implemented several state-of-the-art models.[3] Here we focus on the variational autoencoder (VAE; Kingma and Welling (2014))

2. `http://pytorch.org/docs/stable/distributions.html`
3. `http://pyro.ai/examples/`

# z	# h	PyTorch (ms)	Pyro (ms)
10	400	3.82 ± 0.02	6.79 ± 0.04
30	400	3.73 ± 0.07	6.67 ± 0.03
10	2000	7.65 ± 0.02	10.14 ± 0.06
30	2000	7.66 ± 0.02	10.19 ± 0.03

Figure 3: Times per update of VAE in Pyro versus PyTorch

# IAFs	Test ELBO
0 (theirs)	-6.93
0 (ours)	-6.87
1	-6.82
2	-6.80

Figure 4: Test ELBOs for DMM and extension with IAF guide

and the Deep Markov Model (DMM; Krishnan et al. (2017)), a non-linear state space model that has been used for several applications including audio generation and causal inference. The VAE is a standard example in deep probabilistic modeling, while the DMM has several characteristics that make it ideal as a point of comparison: it is a high-dimensional, non-conjugate model designed to be fit to large data sets; the number of latent variables in a sequence depends on the input data; and it uses a hand-designed approximate posterior.

The models and inference procedures derived by Pyro replicate the original papers almost exactly, except that we use Monte Carlo estimates rather than exact analytic expressions for KL divergence terms. We use the MNIST data set and 2-hidden-layer MLP encoder and decoder networks with varying hidden layer size $\#h$ and latent code size $\#z$ for the VAE and the same data set of digitized music[4] to train the DMM.

To demonstrate that Pyro's abstractions do not reduce its scalability by introducing too much overhead, we compared our VAE implementation with an idiomatic PyTorch implementation.[5] After verifying that they converge to the same test ELBO, we compared the wall-clock time taken to compute one gradient update, averaged over 10 epochs of GPU-accelerated mini-batch stochastic gradient variational inference (batch size 128) on a single NVIDIA GTX 1080Ti. Figure 3 shows that the relative performance gap between the Pyro and PyTorch versions is moderate, and, more importantly, that it shrinks as the total time spent performing tensor operations increases.

We used the DMM to evaluate Pyro's flexibility and expressiveness. As Figure 4 shows, we found that we were able to quickly and concisely replicate the exact DMM model and inference configuration and quantitative results reported in the paper after 5000 training epochs. Furthermore, thanks to Pyro's modular design, we were also able to build DMM variants with more expressive approximate posteriors via autoregressive flows (IAFs) (Kingma et al. (2016)), improving the results with a few lines of code at negligible computational cost.

Acknowledgments

We would like to acknowledge Du Phan, Adam Scibior, Dustin Tran, Adam Paszke, Soumith Chintala, Robert Hawkins, Andreas Stuhlmueller, our colleagues in Uber AI Labs, and the Pyro and PyTorch open-source communities for helpful contributions and feedback.

4. This is the JSB chorales data set from `http://www-etud.iro.umontreal.ca/~Eboulanni/icml2012`
5. We used the PyTorch example at `https://github.com/pytorch/examples/tree/master/vae`

References

Bob Carpenter, Andrew Gelman, Matthew D. Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of Statistical Software*, 76(1), 2017.

Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, and Rif A. Saurous. TensorFlow Distributions. *arXiv:1711.10604*, November 2017.

Hong Ge, Kai Xu, and Zoubin Ghahramani. Turing: a language for flexible probabilistic inference. In *AISTATS*, 2018.

Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521: 452–459, May 2015.

Noah D Goodman and Andreas Stuhlmüller. The Design and Implementation of Probabilistic Programming Languages. `http://dippl.org`, 2014.

Noah D. Goodman, Vikash K. Mansinghka, Daniel Roy, Keith Bonawitz, and Joshua B. Tenenbaum. Church: A language for generative models. In *UAI*, 2008.

Matthew D. Hoffman and Andrew Gelman. The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *J. Mach. Learn. Res.*, 15(1), January 2014.

Ohad Kammar, Sam Lindley, and Nicolas Oury. Handlers in action. In *ICFP*, 2013.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *ICLR*, 2014.

Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *NIPS*. 2016.

Rahul G Krishnan, Uri Shalit, and David Sontag. Structured inference networks for nonlinear state space models. In *AAAI*, 2017.

Vikash K. Mansinghka, Ulrich Schaechtle, Shivam Handa, Alexey Radul, Yutian Chen, and Martin Rinard. Probabilistic programming with programmable inference. In *PLDI*, 2018.

N. Siddharth, Brooks Paige, Jan-Willem van de Meent, Alban Desmaison, Noah D. Goodman, Pushmeet Kohli, Frank Wood, and Philip Torr. Learning disentangled representations with semi-supervised deep generative models. In *NIPS*, 2017.

David Tolpin, Jan-Willem van de Meent, Hongseok Yang, and Frank Wood. Design and implementation of probabilistic programming language anglican. In *IFL*, 2016.

Dustin Tran, Matthew D. Hoffman, Rif A. Saurous, Eugene Brevdo, Kevin Murphy, and David M. Blei. Deep probabilistic programming. In *ICLR*, 2017.

David Wingate, Andreas Stuhlmller, and Noah Goodman. Lightweight implementations of probabilistic programming languages via transformational compilation. In *AISTATS*, 2011.

Journal of Machine Learning Research 20 (2019) 1-28 Submitted 8/18; Revised 12/18; Published 1/19

Iterated Learning in Dynamic Social Networks

Bernard Chazelle CHAZELLE@CS.PRINCETON.EDU
Department of Computer Science, Princeton University,
35 Olden Street, Princeton, NJ 08544

Chu Wang* CHUWANG@AMAZON.COM
Amazon Inc.
500 Boren Avenue, Seattle, WA 98109

Editor: Mehryar Mohri

Abstract

A classic finding by (Kalish et al., 2007) shows that no language can be learned iteratively by rational agents in a self-sustained manner. In other words, if A teaches a foreign language to B, who then teaches what she learned to C, and so on, the language will quickly get lost and agents will wind up teaching their own common native language. If so, how can linguistic novelty ever be sustained? We address this apparent paradox by considering the case of iterated learning in a social network: we show that by varying the lengths of the learning sessions over time or by keeping the networks dynamic, it is possible for iterated learning to endure forever with arbitrarily small loss.

1. Introduction

People typically form opinions by updating their current beliefs and reasons in response to new signals from other sources (friends, colleagues, social media, newspapers, etc.) (Tahbaz-Salehi et al., 2009; Acemoglu and Ozdaglar, 2011; Golub and Jackson, 2010). Suppose there were an information source that made a noisy version of the "truth" available to agents connected through a social network. Under which conditions would the agents reach consensus about their beliefs? What would ensure truthful consensus (meaning that the consensus coincided with the truth)? How long would it take for the process to converge? Addressing these questions requires agreeing on a formal model of distributed learning. Fully rational agents update their beliefs by assuming a prior and using Bayes' rule to integrate all past information available to them (Acemoglu et al., 2011; Mueller-Frank, 2013; Lobel and Sadler, 2015; Mossel et al., 2011; Banerjee, 1992; Bala and Goyal, 1998). Full rationality is intractable in practice (Molavi et al., 2015; Rahimian and Jadbabaie, 2016a), so much effort has been devoted to developing computationally effective mechanisms, including non- (or partially) Bayesian methods (Jadbabaie et al., 2012; Molavi et al., 2015; Golub and Jackson, 2010, 2012; Jadbabaie et al., 2013). Much of this line of work can be traced back to the seminal work of (DeGroot, 1974) on linear opinion pooling.

As the simplest example of iterated learning in a social network, consider a system consisting of one teacher and one learner. The teacher samples data from a distribution and sends it to the learner; the learner updates her belief via Bayes rule repeatedly in order to learn that distribution. This system is equivalent to the usual Bayesian inference scenario. Under mild assumptions, the learner will eventually learn the ground truth asymptotically (Gelman et al., 2013).

∗. Chu Wang did this work prior to joining Amazon

©2019 Bernard Chazelle and Chu Wang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-539.html.

When the network structure comes into play, the dynamics of the learning process becomes more complicated. If the social network forms a chain $X, Y, Z, \ldots$ such that agents teach each other in sequence: X teaches Y, who then teaches Z, and so on, by a classic result of Griffiths and Kalish (Griffiths and Kalish, 2005), the information from the source will vanish after a finite number of iterations. At that point, the agents, assumed to be rational, will be "teaching" each other nothing they don't already know: iterated learning is not self-sustaining (Beppu and Griffiths, 2009; Griffiths and Kalish, 2007, 2005; Rafferty et al., 2009; Kirby et al., 2014; Griffiths et al., 2008; Perfors and Navarro, 2011; Rafferty et al., 2014; Smith, 2009; Kalish et al., 2007). Such findings are hard to validate empirically but variants of it are within the reach of experimental psychology (Kalish et al., 2007). Similar laboratory experiments with human subjects have confirmed the unstainability of iterated learning (Kalish et al., 2007; Beppu and Griffiths, 2009; Tamariz and Kirby, 2015; Bartlett; Griffiths et al., 2008). Similar results of unstainability are found in computational linguistics where, instead of agents sending information, the scenario is about language evolution through generations (Rafferty et al., 2009).

If agents interact in a more complicated and dynamic network, more possibilities of the learning dynamics will emerge. Dynamic networks are common occurrences in opinion dynamics (Hegselmann and Krause, 2002; Mohajer and Touri, 2013; Chazelle and Wang, 2016; Chazelle, 2015), but, to our knowledge, somewhat new in the context of social learning. Following the *Bayesian-Without-Recall* (BWR) model proposed by Rahimian and Jadbabaie in (Rahimian and Jadbabaie, 2016a), we assume the agents to be memoryless and rational: this means that they use Bayesian updates based on current beliefs and signals with no other information from the past, see also (Rahimian et al., 2015a,b; Rahimian and Jadbabaie, 2016b). The BWR model seeks to capture the benefits of rational behavior while keeping both the computation and the information stored to a minimum (Rahimian and Jadbabaie, 2016b).

We focus in this paper on sustainable learning: the conditions to ensure arbitrarily small information loss in truthful consensus (formal definition in the following sections). For chained learning, we show how keeping the length of the training sessions (number of samples transferred) growing slightly allows iterated learning to be sustained in perpetuity. This resolves the paradox raised from language evolution models (Kalish et al., 2007). We further analyze the case when the learner has the ability to reach back to her early ancestors for "fresher" data instead of listening to her direct ancestor, and show how this "hopped" learning mechanism further helps prevent information decay. For the case when the learning network changes over time, we show that under the assumption that each agent hears a noisy signal from the truth at a frequency bounded away from zero, the system reaches truthful consensus almost surely with a convergence rate polynomial in expectation. The relation between the convergence rate and the graph structure is also revealed with a seemingly counter-intuitive finding that agents in a better connected network learn more slowly.

We first introduce the iterated learning model framework with the notation, definitions, and basic properties in Section 2. Then we exam the scenarios of chained learning, hopped learning, and networked learning in Section 3, Section 4, and Section 5, respectively. We show our main results in each section, followed by details of the proofs and discussions.

2. Models, Preliminaries, and Notation

In this section, we formally define the problem mathematically. Assume there are n agents denoted by their indices $1, 2, \ldots, n$. At time $t = 0, 1, \ldots$, the belief of agent i is a probability distribution

over a state space Θ, which is denoted by $\mu_{t,i}$. The interactions between agents are modeled by an infinite sequence $(G_t)_{t\geq 0}$, where each G_t is a directed graph over the node set $\{1,\ldots,n\}$. An edge pointing from i to j in G_t indicates that i receives data from j at time t. Intuitively, the direction of the edge has the same meaning as the "listen-to" activity. Typically, the sequence of graphs is specified ahead of time or is chosen randomly: the only condition that matters is that it should be independent of the randomness used in the data generating and learning process; specifically, taking expectations and variances of the random variables that govern the dynamics will assume a fixed graph sequence (possibly random). The adjacency matrix of G_t is denoted by A_t: it is an $n \times n$ $0/1$ matrix.

2.1. The existence of an information source

When an information source exists whose belief is fixed, we label it agent 0 and refer to it as the *truth*. In such a case, the graph G_t is over the node set $\{0,1,\ldots,n\}$. Because agent 0 (if it exists) holds the truth, no edge out of it points to another node. The adjacency matrix then becomes an $(n+1)\times(n+1)$ matrix whose first row is $(1,0,\ldots,0)$, with a self-loop at agent 0 for simplicity.

2.2. Data generation

At time $t \geq 0$, each agent $i > 0$ samples a state $\theta_{t,i} \in \mathbb{R}$ consistent with her own belief: $\theta_{t,i} \sim \mu_{t,i}$. A noisy measurement $a_{t,i} = \theta_{t,i} + \varepsilon_{t,i}$ is then sent to each agent j such that $(A_t)_{ji} = 1$. All the noise terms $\varepsilon_{t,i}$ are sampled *iid* from $\mathcal{N}(0,\sigma^2)$. An equivalent formulation is to say that the likelihood function $l(a|\theta)$ is drawn from $\mathcal{N}(\theta,\sigma^2)$. In our setting, agent i sends the same data to all of her neighbors; this is done for notational convenience and the same results would still hold if we were to resample independently for each neighbor. Except for the omission of explicit utilities and actions, our setting is easily identified as a variant of the BWR model of (Rahimian and Jadbabaie, 2016a).

2.3. Beliefs update

A single-step update for agent $i > 0$ consists of setting $\mu_{t+1,i}$ as the posterior $\mathbb{P}[\mu_{t+1,i}|d] \propto \mathbb{P}[d|\mu_{t,i}]\mathbb{P}[\mu_{t,i}]$, where d is the data from the neighbors of i received at time t. For the case when the beliefs are Gaussian, we get the classical update rules from Bayesian inference by plugging in the corresponding normal distribution (Box and Tiao, 2011). Updated beliefs remain Gaussian so we can use the notation $\mu_{t,i} \sim \mathcal{N}(x_{t,i}, \tau_{t,i}^{-1})$, where $\tau_{t,i}$ denotes the precision (inverse variance) $\sigma_{t,i}^{-2}$. Writing $\tau = \sigma^{-2}$ and letting $d_{t,i}$ denote the outdegree of i in G_t, for any $i > 0$ and $t \geq 0$, we have

$$\begin{cases} x_{t+1,i} = (\tau_{t,i}x_{t,i} + \tau a_{t,1} + \cdots + \tau a_{t,d_{t,i}})/(\tau_{t,i} + d_{t,i}\tau); \\ \tau_{t+1,i} = \tau_{t,i} + d_{t,i}\tau, \end{cases} \tag{1}$$

where $a_{t,1},\ldots,a_{t,d_{t,i}}$ are the signals received by agent i from its neighbors at time t.

2.4. The influence of the graph sequence

The graph sequence G_t plays a crucial role in the dynamics of the system. A simple starting example is the constant graph of two agents with one directed edge. Such a graph sequence defines the case where one learner repeatedly gets samples from the information source. If the information source is regarded as the ground truth, this system is equivalent to the usual Bayesian inference

scenario. Instead of exhausting all possible graph sequences, in this paper, we focus on three representative types, namely the *chained learning*, *hopped learning*, and *networked learning* models. The fundamental problem we would like to solve is whether the system is able to converge to the *truth*; in other words, whether the truthful information is able to propagate uncorruptedly across the entire system.

3. Chained Learning

Following (Beppu and Griffiths, 2009; Griffiths and Kalish, 2007, 2005; Rafferty et al., 2009; Kirby et al., 2014; Griffiths et al., 2008; Perfors and Navarro, 2011; Rafferty et al., 2014; Smith, 2009; Kalish et al., 2007), we begin with *chained iterated learning*: a learner's state of belief is modeled by a distribution over a hypothesis space $\mathcal{H}$, which is itself equipped with a likelihood function: $\mathbb{P}[d|h]$ indicates the probability of generating data $d \in \mathcal{D}$ given the hypothesis $h \in \mathcal{H}$. A learner's state of belief may change after a learning process, and we naturally call her belief before and after the learning prior and posterior. Notice that the hypothesis space $\mathcal{H}$ is the same as the state space Θ, but we will use $\mathcal{H}$ to emphasize the application background and avoid confusion. The initial hypothesis $\boldsymbol{h}_{\texttt{init}}$ generates m_1 items *iid* for the first learner. These items provide the training data $\mathbf{d}_1 = (d_{1,1},\ldots,d_{1,m_1})$ with which the first learner Bayes-updates its prior. Its posterior is given by setting $t = 1$ in this formula:

$$\mathbb{P}[h|\mathbf{d}_t] = \mathbb{P}[\mathbf{d}_t|h]\,\mathbb{P}[h]/\mathbb{P}[\mathbf{d}_t], \quad \text{with } \mathbb{P}[\mathbf{d}_t] = \sum_{h\in\mathcal{H}} \mathbb{P}[\mathbf{d}_t|h]\,\mathbb{P}[h]. \tag{2}$$

From that point on, each successive learner updates its prior from their predecessor. For any $t > 1$, learner t receives m_t items sampled by the posterior of agent $t-1$ to form the training set $\mathbf{d}_t$. To do that, she picks a random hypothesis h from $\mathcal{H}$ with probability $\mathbb{P}[h|\mathbf{d}_{t-1}]$ (the posterior of learner $t-1$) and then samples m_t items *iid* from h to form $\mathbf{d}_t \in \mathcal{D}^{m_t}$. The posterior $\mathbb{P}[h|\mathbf{d}_t]$ is derived according to (2). Note that learner t has no direct access to the posterior of learner $t-1$ but only to data drawn from a hypothesis sampled from the posterior. Our formulation assumes a discrete space $\mathcal{H}$ but extends to continuous settings, as we show in §3.5.

In the case of linguistic transmission, each hypothesis $h \in \mathcal{H}$ is a "knob" whose setting is given by a number between 0 and 1, specifically the prior probability $\mathbb{P}[h]$. All learners share the same prior. Picking some h from that prior specifies a *language* (also denoted h for convenience). In this case, a language is defined as a probability distribution over $\mathcal{D}$, interpreted here as a set of *sentences*. In this way, the prior can be viewed as a mixture over $\mathcal{H}$: by abuse of terminology, we call it a *mixed* hypothesis, which we distinguish from a *pure* hypothesis of the form $h \in \mathcal{H}$ (corresponding to a single-point distribution). Access to language h is achieved by random sampling: the sentence $d \in \mathcal{D}$ is picked with probability $\mathbb{P}[d|h]$.

Iterated learning proceeds as follows. After selecting language h with probability $\mathbb{P}[h|\mathbf{d}_{t-1}]$, learner t collects m_t independent samples from h. Thus, given a tuple $\mathbf{d}_t = (d_1,\ldots,d_{m_t})$ of sentences from $\mathcal{D}$, the likelihood $\mathbb{P}[\mathbf{d}_t|h]$ is equal to $\prod_{1\leq k\leq m_t} \mathbb{P}[d_k|h]$. The learner is now ready to Bayes-update its prior. Of course, the first one ($t = 1$) samples directly from the language $\boldsymbol{h}_{\texttt{init}}$ chosen for iterated learning. The notation is boldfaced to indicate that $\boldsymbol{h}_{\texttt{init}}$ may be a mixed hypothesis or, in other words, a distribution over hypotheses.

Suppose that $\mathcal{D} = \{d_1, \ldots, d_s\}$ and $\mathcal{H} = \{h_1, \ldots, h_n\}$ are both finite. After observing the data generated by the posterior of learner $t-1$, if learner t winds up choosing h_i then, by Bayesian updating, the probability $P_{ij}^{|t}$ that its posterior picks h_j is given by:

$$P_{ij}^{|t} = \sum_{\mathbf{d} \in \mathcal{D}^{m_t}} \mathbb{P}[h_j|\mathbf{d}]\, \mathbb{P}[\mathbf{d}|h_i] = \sum_{\mathbf{d} \in \mathcal{D}^{m_t}} \frac{\mathbb{P}[\mathbf{d}|h_i]\, \mathbb{P}[\mathbf{d}|h_j]\, \mathbb{P}[h_j]}{\sum_{k=1}^{n} \mathbb{P}[\mathbf{d}|h_k]\, \mathbb{P}[h_k]}\,. \tag{3}$$

To our knowledge, the entire literature on the topic assumes a common, fixed sample size for all the learners: $m_t = m$. Equation (3) can be then interpreted as marginalizing a Gibbs sampler over the data space, which creates a Markov chain over the hypothesis space $\mathcal{H}$: if $\boldsymbol{h}^t$ denotes the row vector formed by the n probabilities $\mathbb{P}[h_k \,|\, \mathbf{d}_t]$, then $\boldsymbol{h}^t = \boldsymbol{h}^{t-1} P^t$, where $\boldsymbol{h}^0 = \boldsymbol{h}_{\mathtt{init}}$. Assuming ergodicity (in this case, a fairly inconsequential technical assumption), the chain can be shown to converge to a unique stationary distribution $\boldsymbol{h}$. It can be easily checked that it coincides with the prior: $\boldsymbol{h} = (\mathbb{P}[h_1], \ldots, \mathbb{P}[h_n])$ (Griffiths and Kalish, 2005; Norris, 1998); see (Rafferty et al., 2009, 2014) for an analysis of the mixing time in specific linguistic scenarios. This convergence reveals the long-term unsustainability of iterated learning. We show how diversifying the sample sizes m_t, hence making the Markov chain time-inhomogeneous, can overcome this weakness. In particular, we prove that it is sufficient for m_t to increase logarithmically with respect to t in order to achieve sustainability.

3.1. Self-Sustainability

We show how to make iterated learning self-sustaining in the presence of a finite hypothesis space $\mathcal{H} = \{h_1, \ldots, h_n\}$. This involves specifying a sequence of training session lengths $m_1, m_2, \ldots$ so that the posterior of any learner ends up differing from $\boldsymbol{h}_{\mathtt{init}}$ by an arbitrarily small amount. Formally, given any $\delta, \varepsilon \geq 0$, we say that iterated learning is (δ, ε)-*self-sustaining* if, with probability at least $1-\varepsilon$, a random $h \in \mathcal{H}$ picked from any learner's posterior distribution differs from $\boldsymbol{h}_{\mathtt{init}}$ in total variation by at most δ. We recall a few facts: the hypothesis h denotes a language modeled as a probability distribution over $\mathcal{D}$; the total variation distance is half the ℓ_1-norm; and the posterior of learner t after the t-th iteration is defined by marginalizing $\mathbb{P}[h|\mathbf{d}_t]$ over all samples $\mathbf{d}_t$ drawn from a random h picked from the posterior of learner $t-1$ (or $\boldsymbol{h}_{\mathtt{init}}$ if $t = 1$). As a shorthand, we speak of ε-self-sustainability to refer to the case $\delta = 0$.

The parameters δ and ε allow us to distinguish between two metrics: the distance between two languages over $\mathcal{D}$ and the distance between two mixtures over $\mathcal{H}$. The two notions could differ widely. For example, if all of $\mathcal{H}$ corresponds to languages very close to $\boldsymbol{h}_{\mathtt{init}}$, to achieve (δ, ε)-self-sustainability might be easy for a tiny $\delta > 0$ but hopelessly difficult for $\delta = 0$. The complexity of iterated learning depends on the geometry of the languages formed by the pure hypotheses. This is best captured by introducing a metric that, though more specialized than the total variation (it works only on the simplex of probability vectors), brings all sorts of technical benefits: the *root-sine distance* between two probability distributions $\boldsymbol{u} = (u_1, \ldots, u_s)$ and $\boldsymbol{v} = (v_1, \ldots, v_s)$ over $\mathcal{D}$ is defined as

$$d_{RS}(\boldsymbol{u}, \boldsymbol{v}) = \sqrt{\frac{1}{2} \sum_{i,j=1}^{s} \left(\sqrt{u_i v_j} - \sqrt{u_j v_i}\right)^2} = \sqrt{1 - \left(\sum_{i=1}^{s} \sqrt{u_i v_i}\right)^2}\,. \tag{4}$$

Note that the *root-sine distance* will be used to measure similarities between two likelihoods, and we will continue the analysis of sustainability defined based on the total variation distance.

It would be surprising if this distance had not been used before, but we could not find a reference. We prove that it is indeed a metric in the Appendix and also explain its name. We show that it is related to the Hellinger, Bhattacharyya and total variation distances, d_H, d_B, d_{TV} by the following relations:

$$\begin{cases} d_H = \sqrt{1 - \sqrt{1 - d_{RS}^2}}\,; \\ d_B = -\frac{1}{2} \ln(1 - d_{RS}^2)\,; \\ d_{TV} \leq \sqrt{2s}\, d_{RS}. \end{cases} \tag{5}$$

3.2. The results

We focus on the "pure" case $\boldsymbol{h}_{\mathtt{init}} \in \mathcal{H}$, and later briefly discuss how to generalize the method to mixed hypotheses. Using the shorthand $\mathtt{d}_{ij}$ for $d_{RS}(\mathbb{P}[\cdot|h_i], \mathbb{P}[\cdot|h_j])$, we define $\mathtt{d}_i := \min_{j:j \neq i} \mathtt{d}_{ij}$. Let $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ be the prior distribution over $\mathcal{H}$, where $p_i := \mathbb{P}[h_i]$. We can obviously assume that each p_i is positive and that all the pure hypotheses are distinct, hence $\mathtt{d}_i > 0$. The two theorems below assume that $\boldsymbol{h}_{\mathtt{init}} = h_1$.

Theorem 1. *For any positive $\varepsilon < 1$, the following sample size sequence makes iterated learning ε-self-sustaining:*

$$m_t = \frac{4}{\mathtt{d}_1^2} \ln \frac{nt}{\varepsilon\, p_1} = \frac{4}{\mathtt{d}_1^2} \Big(\log \frac{t}{\varepsilon} + C\Big),$$

for some $C > 0$ independent of $t, \varepsilon, \mathtt{d}_1$.

The factor 4 can be reduced to $2^{1+o(1)}$ if we adjust the constant C. It is to be expected that the lengths of the training sessions should grow to infinity as p_1 tends to zero, as the vanishing prior makes it increasingly difficult for the posteriors to "attach" to h_1. The session lengths are sensitive to the minimum distance between the languages specified by $\mathcal{H}$ and the target language h_1. Settling for (δ, ε)-self-sustainability allows us to remove this dependency.

Theorem 2. *For any positive $\delta, \varepsilon < 1$, the following sample size sequence makes iterated learning (δ, ε)-self-sustaining:*

$$m_t = \frac{8sn^2}{\delta^2} \Big(\ln \frac{t}{\varepsilon} + C\Big).$$

for some $C > 0$ independent of t, δ, ε.

3.3. The proofs

To establish Theorem 1, we estimate the probability P^* that each leaner ends up picking h_1. Recall that $\boldsymbol{h}^t$ is the posterior distribution of learner t, by the Markovian property of the system,

$$P^* = \mathbb{P}[\boldsymbol{h}^0 = h_1] \prod_{t\geq 0} \mathbb{P}[\boldsymbol{h}^{t+1} = h_1 | \boldsymbol{h}^t = h_1] = \prod_{t\geq 1} P_{11}^{|t}. \tag{6}$$

Since the matrix $P^{|t}$ is the transition matrix of a Markov chain, we proceed by bounding its off-diagonal elements $P_{ij}^{|t}$ for $i \neq j$. We have

$$\begin{aligned} P_{ij}^{|t} &\leq \sum_{\mathbf{d}\in\mathcal{D}^{m_t}} \frac{\mathbb{P}[\mathbf{d}|h_i]\,\mathbb{P}[\mathbf{d}|h_j]p_j}{\mathbb{P}[\mathbf{d}|h_i]p_i + \mathbb{P}[\mathbf{d}|h_j]p_j} = \frac{p_j}{p_i} \sum_{\mathbf{d}\in\mathcal{D}^{m_t}} \frac{\left(\frac{p_i}{p_j}\right)\mathbb{P}[\mathbf{d}|h_i]\,\mathbb{P}[\mathbf{d}|h_j]}{\left(\frac{p_i}{p_j}\right)\mathbb{P}[\mathbf{d}|h_i] + \mathbb{P}[\mathbf{d}|h_j]} \\ &\leq \frac{1}{2}\sqrt{\frac{p_j}{p_i}} \sum_{\mathbf{d}\in\mathcal{D}^{m_t}} \sqrt{\mathbb{P}[\mathbf{d}|h_i]\,\mathbb{P}[\mathbf{d}|h_j]} = \frac{1}{2}\sqrt{\frac{p_j}{p_i}} \Big(\sum_{d\in\mathcal{D}} \sqrt{\mathbb{P}[d|h_i]\,\mathbb{P}[d|h_j]}\Big)^{m_t} \\ &\leq \frac{1}{2}\sqrt{\frac{p_j}{p_i}} \exp\left\{\frac{m_t}{2}\Big(\Big(\sum_{d\in\mathcal{D}} \sqrt{\mathbb{P}[d|h_i]\,\mathbb{P}[d|h_j]}\Big)^2 - 1\Big)\right\}, \end{aligned}$$

where the two equalities are simple deformations, the first inequality is achieved by dropping some non-negative terms from the definition of $P_{ij}^{|t}$ in (3), the second inequality is obtained via Young's inequality, and the last inequality is from Taylor expansion of the natural logarithm function at 1. By definition of the root-sine distance, we have

$$P_{ij}^{|t} \leq \frac{1}{2}\sqrt{\frac{p_j}{p_i}} e^{-\frac{1}{2}\mathtt{d}_{ij}^2 m_t} \quad (i \neq j). \tag{7}$$

Setting $i = 1$ in (7) and summing over $2 \leq j \leq n$, it follows by Cauchy-Schwarz that

$$\sum_{j=2}^{n} P_{1j}^{|t} \leq \frac{1}{2}\sqrt{\frac{n(1-p_1)}{p_1}}\, e^{-\frac{1}{2}\mathtt{d}_1^2 m_t}. \tag{8}$$

Combining (6) and (8) yields

$$P^* \geq \prod_{t\geq 1}\left(1 - \frac{1}{2}\sqrt{\frac{n(1-p_1)}{p_1}}\, e^{-\frac{1}{2}\mathtt{d}_1^2 m_t}\right) \geq 1 - \frac{1}{2}\sqrt{\frac{n(1-p_1)}{p_1}} \sum_{t\geq 1} e^{-\frac{1}{2}\mathtt{d}_1^2 m_t}. \tag{9}$$

Given $0 < \varepsilon < 1$, we constrain the sequence (m_t) to satisfy:

$$\sum_{t\geq 1} e^{-\frac{1}{2}\mathtt{d}_1^2 m_t} < \varepsilon\sqrt{\frac{4p_1}{n(1-p_1)}}. \tag{10}$$

For example, we can pick the sequence

$$m_t = \frac{1}{\mathtt{d}_1^2} \ln \frac{n(1-p_1)t^4}{\varepsilon^2 p_1},$$

which completes the proof. A closer look at the calculation shows that the factor t^4 can be reduced to $C_\alpha t^{2+\alpha}$ for any small $\alpha > 0$ and a suitable constant $C_\alpha > 0$, which makes the dependency on t arbitrarily close to $(2/\mathtt{d}_1^2)\ln t$.

To prove Theorem 2, we set a target distance $\rho := \delta/(n\sqrt{2s})$ and find a subset $A \subseteq \mathcal{H}$ such that (i) $\mathtt{d}_{1j} \leq \rho n$ for $j \in A$ and (ii) $\mathtt{d}_{ij} \geq \rho$ for $i \in A$ and $j \notin A$. To see why such a subset must exist, consider spheres centered at $\boldsymbol{h}_{\mathtt{init}} = h_1$ of radius $k\rho$, for $k = 1, \ldots, n+1$ (with respect to d_{RS}). These define $n+1$ disjoint (open) regions and, by the pigeonhole principle, at least one of them must be empty. We set A to include all the points in the regions preceding the empty one; note that $h_1 \in A$. The claim follows from the triangular inequality. We begin with a straightforward generalization of (8): for any $i \in A$,

$$\sum_{j\notin A} P_{ij}^{|t} \leq \frac{1}{2}\sqrt{\frac{n(1-p_A)}{p_A}}\, e^{-\frac{1}{2}\rho^2 m_t}, \tag{11}$$

where $p_A := \min_{i\in A} p_i$. Now let P^* be the probability that $\boldsymbol{h}^t \in A$ for each t, then (6) and (9) are generalized to

$$P^* \geq \prod_{t\geq 1}\left(1 - \max_{i\in A}\sum_{j\notin A} P_{ij}^{|t}\right) \geq 1 - \frac{1}{2}\sqrt{\frac{n(1-p_A)}{p_A}} \sum_{t\geq 1} e^{-\frac{1}{2}\rho^2 m_t}. \tag{12}$$

Setting

$$m_t = \frac{1}{\rho^2}\ln\frac{n(1-p_A)t^4}{\varepsilon^2 p_A} \tag{13}$$

ensures that $P^* > 1 - \varepsilon$. The root-sine distance between the languages denoted by h_1 and any $h \in A$ is at most ρn, so that, by (5), the total variation distance is bounded by $\sqrt{2s}\rho n = \delta$, which concludes the proof of Theorem 2.

So far, we have analyzed only the "pure" case $\boldsymbol{h}_{\mathtt{init}} \in \mathcal{H}$. The idea of the training is to prevent the prior to "drag" the posterior mixture all across $\mathcal{H}$. It should be clear that a similar result obtains if $\boldsymbol{h}_{\mathtt{init}} \in \Delta\mathcal{H}$ is concentrated on a subset A of $\mathcal{H}$. The proof follows the path charted in Theorem 2 and need not be repeated here. It is crucial to note, however, that this result is to be understood in a coarse-graining sense: iterated learning cannot ensure that the original weights in the mixture $\boldsymbol{h}_{\mathtt{init}}$ are retained but only that A contributes most of the mass in the posteriors. To retain the weights would require changing the stationary distribution to conform with $\boldsymbol{h}_{\mathtt{init}}$, as the process unfolds, something that straightforward Bayesian learning seems unable to do. Learning pure hypotheses bypasses that difficulty.

3.4. Applications

We briefly discuss a direct application of our results to a well-known model of language acquisition via iterated learning and we mention some natural extensions of the techniques.

Language evolution. Rafferty et al. show how iterated learning fails rapidly in a simple model of language evolution (Rafferty et al., 2009). Given n hypotheses, iterated learning with fixed-length training sessions ceases to learn anything new after only $O(\log n \log\log n)$ rounds. Our previous theorems show how to turn this around and achieve self-sustainality. In our notational system, their model is defined on a hypothesis space $\mathcal{H} = \{h_1, \ldots, h_n\}$, where $n = 2^k$ and h_i denotes the

language whose sentences are words in $\{0,1,?\}^k$ with exactly m question marks and $0,1$ matching the binary decomposition of $i-1$ outside the question marks. For example, if $k=4$ and $m=2$, then h_3 denotes the language

$$\{\,00??, 0?1?, ?01?, 0??0, ?0?0, ??10\,\}.$$

We can assume that m is much smaller than k. Each language has the same length $\binom{k}{m}$ and the total number of sentences is $s=\binom{k}{m}2^{k-m}$. The prior is given by $\mathbb{P}[h_i]=p_i=1/n$. Given a hypothesis h_i, $\mathbb{P}[d|h_i]=1/\binom{k}{m}$ if d has m question marks and match the bits of $i-1$ elsewhere; else it is 0 (and d,h are called incompatible). Given $h\in\mathcal{H}$,

$$\begin{cases}\mathbb{P}[d]=\sum_{h\in\mathcal{H}}\mathbb{P}[d|h]\mathbb{P}[h]=2^{m-k}/\binom{k}{m};\\ \mathbb{P}[h|d]=\mathbb{P}[d|h]\mathbb{P}[h]/\mathbb{P}[d]=2^{-m}\quad\text{(or 0 if } d,h \text{ are incompatible).}\end{cases}$$

We easily check that $\mathtt{d}_1^2=1-\left(\sum_{i=1}^s\sqrt{a_ib_i}\right)^2\geq 1-\left(\frac{m}{k}\right)^2>\frac{1}{2}$; hence, by Theorem 1, session lengths m_t no larger than $O(\log\frac{t}{\varepsilon})$ are sufficient to maintain ε-self-sustainability.

Meanings and utterances. In the use of iterated learning for studying language evolution (Griffiths and Kalish, 2005; Perfors and Navarro, 2011), it is common to model the data $\mathbf{d}$ as a joint distribution $(\boldsymbol{x},\boldsymbol{y})$ over a product space $\mathcal{X}^{m_t}\times\mathcal{Y}^{m_t}$. The idea is to distinguish between "meanings" $\boldsymbol{x}$ and "utterances" $\boldsymbol{y}$. In this setting, $\mathbb{P}[\mathbf{d}|h]=\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h]\mu(\boldsymbol{x})$, where $\mu(\boldsymbol{x})$ is the probability of generating $\boldsymbol{x}$. The transition matrix of the Markov chain thus becomes

$$\begin{aligned}P_{ij}^{|t}&=\sum_{\boldsymbol{x}\in\mathcal{X}^{m_t}}\sum_{\boldsymbol{y}\in\mathcal{Y}^{m_t}}\mathbb{P}[h_j|\boldsymbol{x},\boldsymbol{y}]\,\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_i]\mu(\boldsymbol{x})\\&=\sum_{\boldsymbol{x}\in\mathcal{X}^{m_t}}\sum_{\boldsymbol{y}\in\mathcal{Y}^{m_t}}\frac{\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_i]\,\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_j]\,\mathbb{P}[h_j]}{\sum_{k=1}^m\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_k]\,\mathbb{P}[h_k]}\mu(\boldsymbol{x})\,.\end{aligned}\tag{14}$$

Since the output $\boldsymbol{y}$ now depends on both the hypothesis and the input data, we redefine $\mathtt{d}_{ij}$ as the root-sine distance between the two distributions $\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_i]\mu(\boldsymbol{x})$ and $\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_j]\mu(\boldsymbol{x})$:

$$\left(\mathtt{d}'_{ij}\right)^2:=1-\left(\sum_{\boldsymbol{x}\in\mathcal{X}}\sum_{\boldsymbol{y}\in\mathcal{Y}}\sqrt{\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_i]\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_j]}\,\mu(\boldsymbol{x})\right)^2\tag{15}$$

and we define $\mathtt{d}'_i:=\min_{j:j\neq i}\mathtt{d}'_{ij}$. Given any $i\neq j$,

$$\begin{aligned}P_{ij}^{|t}&\leq\sum_{\boldsymbol{x}\in\mathcal{X}^{m_t}}\sum_{\boldsymbol{y}\in\mathcal{Y}^{m_t}}\frac{\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_i]\,\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_j]\,p_j}{\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_i]\,p_i+\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_j]\,p_j}\mu(\boldsymbol{x})\\&\leq\frac{1}{2}\sqrt{\frac{p_j}{p_i}}\sum_{\boldsymbol{x}\in\mathcal{X}^{m_t}}\sum_{\boldsymbol{y}\in\mathcal{Y}^{m_t}}\sqrt{\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_i]\,\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_j]}\,\mu(\boldsymbol{x})\\&\leq\frac{1}{2}\sqrt{\frac{p_j}{p_i}}\Big(\sum_{\boldsymbol{x}\in\mathcal{X}}\sum_{\boldsymbol{y}\in\mathcal{Y}}\sqrt{\mathbb{P}[\boldsymbol{y}|h_i]\,\mathbb{P}[\boldsymbol{y}|h_j]}\,\mu(\boldsymbol{x})\Big)^{m_t}\\&\leq\frac{1}{2}\sqrt{\frac{p_j}{p_i}}\exp\left\{\frac{m_t}{2}\Big(\Big(\sum_{\boldsymbol{x}\in\mathcal{X}}\sum_{\boldsymbol{y}\in\mathcal{Y}}\sqrt{\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_i]\,\mathbb{P}[\boldsymbol{y}|\boldsymbol{x},h_j]}\,\mu(\boldsymbol{x})\Big)^2-1\Big)\right\}.\end{aligned}$$

This gives us this new version of inequality (7), which we can use as the basis for a repeat of the argument of the previous section:

$$P_{ij}^{|t}\leq\frac{1}{2}\sqrt{\frac{p_j}{p_i}}e^{-\frac{1}{2}\mathtt{d}'^2_{ij}m_t}\quad(i\neq j).\tag{16}$$

3.5. Gaussian chained learning

When iterated learning operates over a hypothesis space $\mathcal{H}$ parametrized continuously, say, in $\mathbb{R}$, the minimum root-sine distance usually vanishes and the previous arguments run into singularities and collapse. A new approach is needed. To make our discussion concrete, we assume that the prior distribution of each learner is a Gaussian $\mathbb{P}[h]\sim N(\bar\mu,\bar\sigma^2)$ and that the likelihood of producing data d given hypothesis h is also normal: $\mathbb{P}[d|h]=N(h,\sigma^2)$. The likelihood can also be understood as a noisy measurement of h: $d=h+\phi$, where the noise $\phi\sim N(0,\sigma^2)$. We assume that the data received by the first learner comes from $N(\mu_0,\sigma_0^2)$. This is the simplest instance of a continuous setting in which the root-sine distance argument fails. We discuss it in some detail, considering both chained learning and its generalizations; and then we use the results to treat the case of iterated Bayesian linear regression.

During its training session, the t-th learner receives data $\mathbf{d}_t=(d_{t,1},\ldots,d_{t,m_t})$ from its predecessor: it is obtained by first picking a random hypothesis h from the posterior of learner $t-1$ and then collecting m_t independent random samples from $N(h,\sigma^2)$. For the case $t=1$, we can treat the original teacher as learner 0 with its posterior equal to $N(\mu_0,\sigma_0^2)$. Learner t Bayes-updates its posterior as follows:

$$\mathbb{P}[h|\mathbf{d}_t]\propto\mathbb{P}[\mathbf{d}_t|h]\mathbb{P}[h]\propto\exp\Big(-\frac{1}{2\sigma^2}\sum_{i=1}^{m_t}(d_{t,i}-h)^2\Big)\exp\Big(-\frac{1}{2\bar\sigma^2}(h-\bar\mu)^2\Big),$$

which is still Gaussian, with mean and variance denoted by μ_t and σ_t^2, respectively. Carrying out the usual square completion gives up these update rules: for $t>0$,

$$\begin{cases}\mu_t=\frac{1}{\bar\tau+m_t\tau}\left(\bar\tau\bar\mu+\tau(d_{t,1}+d_{t,2}+\cdots+d_{t,m_t})\right)\\ \tau_t=\bar\tau+m_t\tau,\end{cases}\tag{17}$$

where we define the precisions $\tau=1/\sigma^2$, $\bar\tau=1/\bar\sigma^2$, and $\tau_t=1/\sigma_t^2$. We say that iterated learning is *ε-self-sustaining* if $|\mathbb{E}\,\mu_t-\mu_0|\leq\varepsilon$ and $\sigma_t^2+\operatorname{var}\mu_t$ remains bounded for all t. If $\sigma_t^2+\operatorname{var}\mu_t\to 0$ as $t\to\infty$, we say that iterated learning is *strongly ε-self-sustaining*. We consider successively the case of chained iterated learning and the more challenging "hopping" scenario in which a new learner picks a random teacher from the past (instead of the previous one).

In chained iterated learning, the data $d_{t,i}$ is a noisy message drawn from the posterior of the $(t-1)$-th learner; hence $d_{t,i}\sim N(\mu_{t-1},\sigma_{t-1}^2+\sigma^2)$. In view of (17), μ_t is itself Gaussian. By taking the expectation and variance of equation (17), we find the following recursive relations for $\mathbb{E}\,\mu_t$ and $\operatorname{var}\mu_t$: for $t>0$,

$$\begin{cases}\mathbb{E}\,\mu_t=\frac{1}{\bar\tau+m_t\tau}\left(\bar\tau\bar\mu+m_t\tau\,\mathbb{E}\,\mu_{t-1}\right);\\ \operatorname{var}\mu_t=\frac{m_t\tau^2}{(\bar\tau+m_t\tau)^2}\left(\operatorname{var}\,\mu_{t-1}+\sigma_{t-1}^2+\sigma^2\right).\end{cases}\tag{18}$$

If we define $\beta_t := m_t\tau/(\bar{\tau} + m_t\tau)$, then (18) becomes $\mathbb{E}\,\mu_t = \beta_t\,\mathbb{E}\,\mu_{t-1} + (1-\beta_t)\bar{\mu}$. If $m_t = m$ is a constant, then so is β_t, and the recursive relation (18) becomes

$$\mathbb{E}\,\mu_t - \bar{\mu} = \beta_1^t(\mu_0 - \bar{\mu}),$$

which shows that $\mathbb{E}\,\mu_t$ converges to $\bar{\mu}$ exponentially fast. As in the discrete case, iterated learning is not self-sustainable with constant-length training sessions. By letting m_t increase as $t^{1+o(1)}$ order, however, we can achieve self-sustainability:

Theorem 3 *For any $0 < \varepsilon < 1$, the following sample size sequence makes chained iterated learning strongly ε-self-sustaining:*

$$m_t = \frac{|\mu_0 - \bar{\mu}|}{\varepsilon}\Big(1 + \frac{1}{c}\Big)\Big(\frac{\sigma}{\bar{\sigma}}\Big)^2 t^{1+c},$$

for an arbitrarily small constant $c > 0$.

Proof We observe that $\mathbb{E}\,\mu_t$ is a convex combination of $\bar{\mu}$ and $\mathbb{E}\,\mu_s$ $(s < t)$; specifically,

$$\mathbb{E}\,\mu_t = \prod_{s=1}^{t}\beta_s\mu_0 + \Big(1 - \prod_{s=1}^{t}\beta_s\Big)\bar{\mu}. \tag{19}$$

Because $\sum_{s>0}(1/s)^{1+c} < 1 + \int_1^\infty x^{-1-c}\,dx = 1 + 1/c$, we have

$$\begin{aligned} 1 \;&\geq\; \prod_{s=1}^{t}\beta_s = \prod_{s=1}^{t}\Big(1 - \frac{\bar{\tau}}{m_s\tau + \bar{\tau}}\Big) \geq 1 - \sum_{s=1}^{t}\frac{\bar{\tau}}{m_s\tau + \bar{\tau}} \\ &\geq\; 1 - \frac{\varepsilon}{|\mu_0 - \bar{\mu}|}\Big(\frac{c}{c+1}\Big)\sum_{s=1}^{\infty}\frac{1}{s^{1+c}} > 1 - \frac{\varepsilon}{|\mu_0 - \bar{\mu}|}\,. \end{aligned}$$

This shows that

$$|\mathbb{E}\,\mu_t - \mu_0| = \Big(1 - \prod_{s=1}^{t}\beta_s\Big)|\bar{\mu} - \mu_0| \leq \varepsilon.$$

By (17), $\sigma_t^2 = 1/\tau_t < 1/m_t\tau \to 0$. Since $\sigma_{t-1}^2 \leq \bar{\sigma}^2$ for $t > 1$, it follows from (18) that $\operatorname{var}\mu_t \leq (\operatorname{var}\mu_{t-1} + \sigma^2 + \bar{\sigma}^2)/m_t$ for $t > 1$, and $\operatorname{var}\mu_1 \leq (\sigma_0^2 + \sigma^2)/m_1$. Writing $M_t := m_t m_{t-1}\ldots m_1$, we have

$$\begin{aligned} M_t\operatorname{var}\mu_t &\leq M_{t-1}\operatorname{var}\mu_{t-1} + M_{t-1}(\sigma^2 + \bar{\sigma}^2) \\ &\leq tM_{t-1}(\sigma_0^2 + \sigma^2 + \bar{\sigma}^2), \end{aligned}$$

and thus $\operatorname{var}\mu_t \leq (\sigma_0^2 + \sigma^2 + \bar{\sigma}^2)t/m_t \to 0$ since $m_t = \Omega(t^{1+c})$. ■

3.6. Iterated Bayesian Linear Regression

The iterated version of Bayesian linear regression has been the subject of extensive study in the field of psychology (Kalish et al., 2007; Beppu and Griffiths, 2009; Tamariz and Kirby, 2015; Bartlett; Griffiths et al., 2008). The work has involved experimentation with human subjects but little in

the way of theoretical analysis. This section is a first step toward filling this gap. The task at hand is to estimate a hypothesis $h \in \mathcal{H} := \mathbb{R}^d$ given a noisy measurements on the hyperplane $y = h^T x$, where $x \in \mathbb{R}^d$. In the Bayesian setting, we assume a Gaussian prior on the hypothesis space: $\mathbb{P}[h] \sim N(\bar{\mu}, \bar{\sigma}^2 I_d)$. The data is given by (x, y), where $x \sim N(0, I_d)$ and $y = h^T x + \phi$, for $\phi \sim N(0, \sigma^2)$ (with x, ϕ independent). Since we typically make several measurements, we write this (likelihood) relation in matrix form: $y = Xh + \phi$, where $y \in \mathbb{R}^m$ (with m the number of measurements); $\phi \sim N(0, \sigma^2 I_m)$; and X is an m-by-d matrix each of whose rows denotes a random vector $x \sim N(0, I_d)$. This means that the matrix X is random (a fact of key importance in our discussion below). We have:

$$\begin{cases} \mathbb{P}[\phi] \sim \exp\{-\frac{1}{2\sigma^2}\|\phi\|_2^2\} & \texttt{(noise)} \\ \mathbb{P}[h] \sim \exp\{-\frac{1}{2\bar{\sigma}^2}\|h - \bar{\mu}\|_2^2\} & \texttt{(prior)} \\ \mathbb{P}[y|X,h] \sim \exp\{-\frac{1}{2\sigma^2}\|y - Xh\|_2^2\} & \texttt{(likelihood)} \end{cases}$$

In iterated Bayesian linear regression, the t-th learner receives her data from learner $t-1$. Here, learner 0 is treated just like any other agent, except that his prior $\mathbb{P}[h] \sim N(\mu_0, \bar{\sigma}^2 I_d)$ is the distribution to be learned iteratively. Since sampling from the prior is independent of X, Bayesian updating gives the posterior $N(\mu_t, \Sigma_t)$, where

$$\mathbb{P}[h|X,y] = \mathbb{P}[h]\,\mathbb{P}[y|X,h]/\mathbb{P}[y|X] \sim \exp\Big\{-\frac{1}{2\bar{\sigma}^2}\|h - \bar{\mu}\|_2^2 - \frac{1}{2\sigma^2}\|y - Xh\|_2^2\Big\}.$$

Completing the square in the usual fashion shows that the posterior of learner t is given by:

$$\begin{cases} \Sigma_t = \big(\bar{\sigma}^{-2}I_d + \sigma^{-2}X_t^T X_t\big)^{-1}; \\ \mu_t = \Sigma_t\left(\bar{\sigma}^{-2}\bar{\mu} + \sigma^{-2}X_t^T y_t\right), \end{cases} \tag{20}$$

where (X_t, y_t) is the data gathered by learner t from her predecessor: specifically, $y_t = X_t h + \phi_t$, where h is collected from the $(t-1)$-th learner by sampling his posterior distribution $N(\mu_{t-1}, \Sigma_{t-1})$.

Theorem 4 *Given any small enough $\delta, \varepsilon > 0$, the following sample size sequence for iterated Bayesian linear regression ensures that $\|\mathbb{E}\mu_t - \mu_0\|_2 \leq \delta$ with probability greater than $1 - \varepsilon$:*

$$m_t = D_c\frac{\|\mu_0 - \bar{\mu}\|_2}{\delta}\Big(\frac{\sigma}{\bar{\sigma}}\Big)^2 t^{1+c} + D_c\,d\log\frac{t+1}{\varepsilon},$$

for an arbitrarily small $c > 0$ and a constant D_c that depends only on c.

Proof We proceed in two steps: first, we show that to keep $\mathbb{E}\mu_t$ arbitrarily close to μ_0 for all t hinges on spectral properties of certain random matrices; second, we call on known facts about the singular values of random Gaussian matrices to translate the spectral condition into a high-probability event. The proof unfolds as a series of simple relations, which we state first and then demonstrate. The first one follows directly from (20):

$$\mathbb{E}\mu_t = (I_d + M_t)^{-1}\left(\bar{\mu} + M_t\,\mathbb{E}\mu_{t-1}\right), \quad \text{where} \quad M_t := \Big(\frac{\bar{\sigma}}{\sigma}\Big)^2 X_t^T X_t. \tag{21}$$

Note that (21) is a randomized recursive relation since the data points $X_1, X_2, \ldots$ are themselves random. We note that all the matrices whose inverses are taken are positive definite, hence nonsingular. To move on to our second relation, we define the matrix

$$Q_t := (I_d + M_t)^{-1} M_t (I_d + M_{t-1})^{-1} M_{t-1} \cdots (I_d + M_1)^{-1} M_1,$$

for $t > 0$, with $Q_0 = I_d$, and prove by induction that

$$\mathbb{E}\,\mu_t = Q_t \mu_0 + (I_d - Q_t)\bar{\mu}. \tag{22}$$

The base case is obvious so we assume that $t > 0$: by (21),

$$\begin{aligned}
\mathbb{E}\,\mu_t &= (I_d + M_t)^{-1}(\bar{\mu} + M_t\,\mathbb{E}\,\mu_{t-1}) \\
&= (I_d + M_t)^{-1}(\bar{\mu} + M_t Q_{t-1}\mu_0 + M_t(I_d - Q_{t-1})\bar{\mu}) \\
&= (I_d + M_t)^{-1} M_t Q_{t-1}\mu_0 + (I_d + M_t)^{-1}(I_d + M_t(I_d - Q_{t-1}))\bar{\mu} \\
&= Q_t\,\mu_0 + (I_d - (I_d + M_t)^{-1} M_t Q_{t-1})\bar{\mu},
\end{aligned}$$

which proves (22). Our next goal is to bound the information decay $\|\mathbb{E}\,\mu_t - \mu_0\|_2$. To do that, we investigate the spectral norm of the matrix $I_d - Q_t$, which leads to our third relation. We prove by induction that, for $t > 0$,

$$\|I_d - Q_t\|_2 \le \sum_{s=1}^{t} \|A_s\|_2, \tag{23}$$

where $A_s := (I_d + M_s)^{-1}$. For $t = 1$, $Q_1 = (I_d + M_1)^{-1} M_1 = I_d - (I_d + M_1)^{-1}$ and the claim follows. If $t > 1$, then

$$\begin{aligned}
\|I_d - Q_t\|_2 &= \|(I_d - Q_{t-1}) + (Q_{t-1} - Q_t)\|_2 \\
&\le \|I_d - Q_{t-1}\|_2 + \|Q_t - Q_{t-1}\|_2 \le \sum_{s=1}^{t-1} \|A_s\|_2 + \|\Psi\|_2,
\end{aligned}$$

where $\Psi := (A_t M_t - I_d) Q_{t-1}$. Since $A_t(I_d + M_t) = I_d$, we have $\Psi = -A_t Q_{t-1}$. Each matrix M_s is positive semidefinite, so the eigenvalues of $(I_d + M_s)^{-1} M_s$ are of the form $\lambda/(1+\lambda)$, where $\lambda \ge 0$. This shows that all the eigenvalues of Q_s are between 0 and 1; therefore $\|Q_s\|_2 \le 1$. The eigenvalues of $I_d - A_t M_t$ are the same as those of A_t; hence, by submultiplicativity, $\|\Psi\|_2 \le \|A_t\|_2 \|Q_{t-1}\|_2 \le \|A_t\|_2$, which establishes (23).

We are now ready to express the information decay in spectral terms. Pick an arbitrarily small constant $c > 0$ and assume that

$$\|A_s\|_2 \le \frac{\delta}{\|\bar{\mu} - \mu_0\|_2} \Big(\frac{c}{1+c}\Big)\Big(\frac{1}{s}\Big)^{1+c}. \tag{24}$$

By (22), $\mathbb{E}\,\mu_t - \mu_0 = (I_d - Q_t)(\bar{\mu} - \mu_0)$; therefore, by (23),

$$\begin{aligned}
\|\mathbb{E}\,\mu_t - \mu_0\|_2 &\le \|\bar{\mu} - \mu_0\|_2 \sum_{s=1}^{t} \|A_s\|_2 \le \frac{\delta c}{1+c} \sum_{s=1}^{t} s^{-1-c} \\
&\le \frac{\delta c}{1+c}\Big(1 + \int_1^{\infty} x^{-1-c}\,dx\Big) = \delta,
\end{aligned} \tag{25}$$

The relation says that, on average, the means of any of the agents' posteriors can be brought as close to the original mean to be learned as we want. We can turn this into a high-probability event by using some basic random matrix theory. Recall that $\mathbb{E}\,\mu_t$ is itself a random variable whose stochasticity comes from the matrices X_s, which are all drawn from Gaussians. Because M_s is positive semidefinite,

$$\|A_s\|_2 \le \|M_s^{-1}\|_2 \le \frac{(\sigma/\bar{\sigma})^2}{\lambda_{\min}(X_t^T X_t)} \le \Big(\frac{\sigma/\bar{\sigma}}{\sigma_1(X_t)}\Big)^2, \tag{26}$$

which gives us a relation between the spectral norm of $(I_s + M_s)^{-1}$ and the smallest singular value $\sigma_1(X_t)$ of an m_t-by-d matrix X_t whose elements are drawn *iid* from $N(0,1)$. The asymptotic behavior of $\sigma_1(X_t)$ for large values of m_t has been extensively studied within the field of random matrix theory (Davidson and Szarek, 2001; Edelman, 1988; Rudelson and Vershynin, 2009). Following Theorem II.13 in (Davidson & Szarek (Davidson and Szarek, 2001)), for any $\gamma_t > 0$,

$$\mathbb{P}[\sigma_1(X_t) < \sqrt{m_t} - \sqrt{d} - \gamma_t] \le e^{-\gamma_t^2/2}.$$

We use C below as a generic constant large enough to satisfy the inequalities where it appears. Setting $\gamma_t = C\sqrt{\log((t+1)/\varepsilon)}$ ensures that $\sum_{t>0} e^{-\gamma_t^2/2} < \varepsilon$, hence that $\sigma_1(X_t) < \sqrt{m_t} - \sqrt{d} - \gamma_t$ holds for all t with probability less than ε. With our setting of m_t, this means that, for all $t > 0$,

$$\mathbb{P}\Big[\,\sigma_1(X_t) \ge \frac{\sqrt{m_t}}{2}\,\Big] > 1 - \varepsilon. \tag{27}$$

Assuming the event in (27), it follows from (26) and our setting of m_t that

$$\|A_t\|_2 \le \frac{4}{m_t}\Big(\frac{\sigma}{\bar{\sigma}}\Big)^2 \le \frac{\delta}{\|\bar{\mu} - \mu_0\|_2}\Big(\frac{4}{D_c}\Big)\Big(\frac{1}{t}\Big)^{1+c};$$

hence (24) holds for D_c large enough. By (25, 27), this proves that, with probability greater than $1 - \varepsilon$, $\|\mathbb{E}\,\mu_t - \mu_0\|_2 \le \delta$ for all $t > 0$, which completes the proof. ■

4. Hopped Learning

In this section, we consider the "hopped learning" scenario in which learner t hops back to pick a teacher from $\{0, 1, \ldots, t-1\}$ at random, and then takes m_t samples from her posterior. To model the data generating process, we continue to adopt the Gaussian setting from Sections 3.5 and 3.6. Note that the graph sequence G_t becomes random under the constraint that a learner can only get data from the learners before her. Since multiple samples can be sent from a teacher to a learner, we use $d_{t,s,i}$ to denote the i-th sample generated by agent s at time t. Note that though a learner only receives data from one teacher, without loss of generality, we assume all her predecessors generate samples but she only listens to one of them. The recursive relation for μ_t becomes

$$\mu_t = \frac{\beta_t}{m_t} \sum_{s=0}^{t-1} \chi_{t,s} \sum_{i=1}^{m_t} d_{t,s,i} + (1 - \beta_t)\bar{\mu}, \tag{28}$$

where, given t, the random variable $\chi_{t,s}$ is 1 for a value of s picked at random between 0 and $s-1$, and is zero elsewhere; recall that $\beta_t := m_t\tau/(\bar{\tau} + m_t\tau)$. Hopped iterated learning provides access

to earlier data, so one would expect the lengths of the training sessions to grow more slowly than in chained learning. The change is indeed quite dramatic:

Theorem 5 *For any positive $\varepsilon < |\mu_0 - \bar{\mu}|$, the following sample size sequence makes hopped iterating learning ε-self-sustaining:*

$$m_t = B_c \frac{|\mu_0 - \bar{\mu}|}{\varepsilon} \Big(\frac{\sigma}{\bar{\sigma}}\Big)^2 (1 + \log t)^{1+c},$$

for an arbitrarily small $c > 0$ and a constant B_c that depends only on c.

Proof By taking expectation on both sides of (28), for any $t > 0$,

$$\mathbb{E}\,\mu_t = \frac{\beta_t}{t} \sum_{s=0}^{t-1} \mathbb{E}\,\mu_s + (1 - \beta_t)\bar{\mu},$$

We define $\gamma_1 = \beta_1$ and, for $t > 1$,

$$\gamma_t := (1 + \beta_1)\left(1 + \frac{\beta_2}{2}\right) \cdots \left(1 + \frac{\beta_{t-1}}{t-1}\right) \frac{\beta_t}{t}.$$

We verify easily that $\mathbb{E}\,\mu_t = \gamma_t \mu_0 + (1 - \gamma_t)\bar{\mu}$, for $t > 0$; therefore, the first part in establishing ε-self-sustainability consists of proving that

$$1 \geq \gamma_t \geq 1 - \frac{\varepsilon}{|\mu_0 - \bar{\mu}|}\,, \tag{29}$$

which will show that $|\mathbb{E}\,\mu_t - \mu_0| \leq \varepsilon$. Note that

$$\gamma_t \leq \frac{1}{t} \prod_{s=1}^{t-1} \Big(1 + \frac{1}{s}\Big) = 1.$$

Now define

$$\alpha_s = \frac{\varepsilon}{B_c |\mu_0 - \bar{\mu}| s (1 + \log s)^{1+c}}.$$

for $s > 0$. We pick a constant B_c large enough so that α_s is small enough to carry out first-order Taylor approximations around $1 + \alpha_s$. We find that

$$\begin{aligned} 1 + \frac{\beta_s}{s} &= 1 + \frac{1}{s}\Big(1 - \frac{1}{1 + m_s \tau/\bar{\tau}}\Big) \geq \Big(1 + \frac{1}{s}\Big)\Big(1 - \frac{1}{(s+1) m_s \tau/\bar{\tau}}\Big) \\ &\geq \Big(1 + \frac{1}{s}\Big)\Big(1 - \frac{s\alpha_s}{s+1}\Big) \geq \Big(1 + \frac{1}{s}\Big)(1 - \alpha_s) \geq \Big(1 + \frac{1}{s}\Big) e^{-2\alpha_s}. \end{aligned}$$

Thus,

$$\gamma_t \geq \frac{\beta_t}{t} \prod_{s=1}^{t-1} \Big(1 + \frac{1}{s}\Big) e^{-2\sum_{s=1}^{t-1} \alpha_s} = \beta_t e^{-2\sum_{s=1}^{t-1} \alpha_s} \geq 1 - \frac{\varepsilon}{|\mu_0 - \bar{\mu}|},$$

which establishes (29). Our derivation relies on the fact that

$$\beta_t \geq 1 - \frac{\varepsilon}{B_c |\mu_0 - \bar{\mu}| (1 + \log t)^{1+c}} \geq 1 - \frac{\varepsilon}{2|\mu_0 - \bar{\mu}|}$$

and

$$\sum_{s=1}^{t-1} \frac{1}{s(1 + \log s)^{1+c}} \leq 1 + \frac{1}{(\log e)^{1+c}} \int_2^{t-1} \frac{1}{x(\ln x)^{1+c}}\, dx = O\Big(\frac{1}{c}\Big)\,;$$

hence,

$$e^{-2\sum_{s=1}^{t-1} \alpha_s} \geq e^{-O(\varepsilon/(cB_c|\mu_0 - \bar{\mu}|))} \geq 1 - \frac{\varepsilon}{2|\mu_0 - \bar{\mu}|}.$$

Having shown that $|\mathbb{E}\,\mu_t - \mu_0| \leq \varepsilon$ for all t, it now suffices to prove that $\sigma_t^2 + \operatorname{var} \mu_t$ remains bounded. We note that $\tau_t > m_t \tau \to \infty$, hence $\sigma_t^2 = 1/\tau_t \to 0$, so the remainder of the proof needs to establish that the variance of μ_t stays bounded. Writing $D_{t,s} := d_{t,s,1} + \cdots + d_{t,s,m_t}$, we have $\operatorname{var} D_{t,s} = m_t \operatorname{var} d_{t,s,1} = m_t(\sigma_s^2 + \sigma^2 + \operatorname{var} \mu_s)$; hence

$$\mathbb{E}\, D_{t,s}^2 = \operatorname{var} D_{t,s} + (\mathbb{E}\, D_{t,s})^2 = m_t(\sigma_s^2 + \sigma^2 + \operatorname{var} \mu_s) + m_t^2 (\mathbb{E}\,\mu_s)^2.$$

In (28), the variables $\chi_{t,s}$ and $D_{t,s}$ are independent, for $0 \leq s \leq t-1$; furthermore, $\mathbb{E}\,\chi_{t,s} = \mathbb{E}\,\chi_{t,s}^2 = 1/t$, and $\mathbb{E}\,\chi_{t,s_1}\chi_{t,s_2} = 0$ if $s_1 \neq s_2$; therefore,

$$\begin{aligned} \operatorname{var}[\chi_{t,s} D_{t,s}] &= \mathbb{E}\,\chi_{t,s}^2\, \mathbb{E}\, D_{t,s}^2 - (\mathbb{E}\,\chi_{t,s})^2 (\mathbb{E}\, D_{t,s})^2 = \frac{\mathbb{E}\, D_{t,s}^2}{t} - \frac{(\mathbb{E}\, D_{t,s})^2}{t^2} \\ &= \Big(\frac{m_t}{t}\Big)(\sigma_s^2 + \sigma^2 + \operatorname{var} \mu_s + m_t (\mathbb{E}\,\mu_s)^2) - \Big(\frac{m_t}{t}\Big)^2 (\mathbb{E}\,\mu_s)^2 \end{aligned} \tag{30}$$

and, for $s_1 \neq s_2$,

$$\begin{aligned} \operatorname{cov}[\chi_{t,s_1} D_{t,s_1}, \chi_{t,s_2} D_{t,s_2}] &= \mathbb{E}\,[\chi_{t,s_1}\chi_{t,s_2} D_{t,s_1}, D_{t,s_2}] - \mathbb{E}\,[\chi_{t,s_1} D_{t,s_1}]\mathbb{E}\,[\chi_{t,s_2} D_{t,s_2}] \\ &= \mathbb{E}\,[\chi_{t,s_1}\chi_{t,s_2}]\mathbb{E}\,[D_{t,s_1} D_{t,s_2}] - \mathbb{E}\,\chi_{t,s_1}\mathbb{E}\, D_{t,s_1}\mathbb{E}\,\chi_{t,s_2}\mathbb{E}\, D_{t,s_2} \\ &= -\frac{1}{t^2}\,\mathbb{E}\, D_{t,s_1}\mathbb{E}\, D_{t,s_2} = -\Big(\frac{m_t}{t}\Big)^2 \mathbb{E}\,\mu_{s_1}\mathbb{E}\,\mu_{s_2}. \end{aligned} \tag{31}$$

Then, by taking the variance on both sides of (28), we have

$$\begin{aligned} \operatorname{var} \mu_t &= \Big(\frac{\beta_t}{m_t}\Big)^2 \operatorname{var} \sum_{s=0}^{t-1} \chi_{t,s} D_{t,s} \\ &= \Big(\frac{\beta_t}{m_t}\Big)^2 \Big(\sum_{s=0}^{t-1} \operatorname{var}[\chi_{t,s} D_{t,s}] + \sum_{0 \leq s_1 \neq s_2 \leq t-1} \operatorname{cov}[\chi_{t,s_1} D_{t,s_1}, \chi_{t,s_2} D_{t,s_2}]\Big) \\ &= \Big(\frac{\beta_t}{m_t}\Big)^2 \Big(\sum_{s=0}^{t-1} \Big(\frac{m_t}{t}\Big)(\sigma_s^2 + \sigma^2 + \operatorname{var} \mu_s + m_t (\mathbb{E}\,\mu_s)^2) - \Big(\frac{m_t}{t}\Big)^2 \Big(\sum_{s=0}^{t-1} \mathbb{E}\,\mu_s\Big)^2\Big) \\ &\leq \frac{1}{t m_t} \sum_{s=0}^{t-1} (\sigma_s^2 + \sigma^2 + \operatorname{var} \mu_s + m_t (\mathbb{E}\,\mu_s)^2). \end{aligned}$$

Notice that $\sigma_s^2 \to 0$ and $(\mathbb{E}\,\mu_s)^2$ is bounded since $|\mathbb{E}\,\mu_t - \mu_0| \leq \varepsilon$. We conclude that $\sigma_t^2 + \operatorname{var} \mu_t$ remains bounded for all t. ■

5. Networked Learning

In this section, we study the information transfer and iterated learning with general graph sequence G_t. We assume that the initial belief $\mu_{0,i}$ of agent i is Gaussian: $\mu_{0,i} \sim \mathcal{N}(x_{0,i}, \sigma_{0,i}^2)$. Without loss of generality, the truth is assumed to be a constant (single-point distribution: $\mu_{t,0} = 0$; $\sigma_{t,0} = 0$ for all t) and the standard deviation is the same for all other agents, i.e., $\sigma_{0,i} = \sigma_0 > 0$ for $i > 0$. Because agent 0 holds the truth, no edge points out of it. The adjacency matrix of G_t is denoted by A_t: it is an $(n+1) \times (n+1)$ matrix whose first row is $(1, 0, \ldots, 0)$. Note that n is the number of learners and should not be confused with the number of hypotheses in Section 3.

5.1. The dynamics in matrix form

Let D_t and P_t denote the $(n+1)$-by-$(n+1)$ diagonal matrices $\mathrm{diag}(\eta_{t,i})$ and $(\tau_0/\tau)I + \sum_{k=0}^{t-1} D_k$, respectively, where $\eta_{t,i}$ is the out-degree of agent i at time t, I is the identity matrix and the sum is 0 for $t = 0$. It follows from (1) that $\mu_{t,i} \sim \mathcal{N}(x_{t,i}, (\tau P_t)_{ii}^{-1})$ for $i > 0$. Regrouping the means in vector form, $\boldsymbol{x}_t := (x_{t,0}, \ldots, x_{t,n})^T$, where $x_{t,0} = 0$ and $x_{0,1}, \ldots, x_{0,n}$ are given as inputs, we have

$$\boldsymbol{x}_{t+1} = (P_t + D_t)^{-1} \left(P_t \boldsymbol{x}_t + A_t \left(\boldsymbol{x}_t + \boldsymbol{u}_t + \boldsymbol{\varepsilon}_t\right)\right), \tag{32}$$

where $\boldsymbol{u}_t$ is such that $u_{t,0} \sim \mathcal{N}(0,0)$ and, for $i > 0$, $u_{t,i} \sim \mathcal{N}(\mathbf{0}, (\tau(P_t)_{ii})^{-1})$; and $\boldsymbol{\varepsilon}_t$ is such that $\varepsilon_{t,0} \sim \mathcal{N}(0,0)$ and, for $i > 0$, $\varepsilon_{t,i} \sim \mathcal{N}(\mathbf{0}, 1/\tau)$. We refer to the vectors $\boldsymbol{x}_t$ and $\boldsymbol{y}_t := \mathbb{E}\,\boldsymbol{x}_t$ as the *mean process* and the *expected mean process*, respectively. Taking expectations on both sides of (32) with respect to the random vectors $\boldsymbol{u}_t$ and $\boldsymbol{\varepsilon}_t$ yields the update rule for the expected mean process: $\boldsymbol{y}_0 = \boldsymbol{x}_0$ and, for $t > 0$,

$$\boldsymbol{y}_{t+1} = (P_t + D_t)^{-1} (P_t + A_t)\, \boldsymbol{y}_t. \tag{33}$$

A key observation is that $(P_t + D_t)^{-1}(P_t + A_t)$ is a stochastic matrix, so the expected mean process $\boldsymbol{y}_t$ forms a diffusive influence system (Chazelle, 2015): the vector evolves by taking convex combinations of its own coordinates. What makes the analysis different from standard multiagent agreement systems is that the weights vary over time. In fact, some weights typically tend to 0, which violates one of the cardinal assumptions used in the analysis of averaging systems (Chazelle, 2015; Moreau, 2005). This leads us to the use of arguments, such as fourth-order moment bounds, that are not commonly encountered in this area.

5.2. The results

The belief vector $\boldsymbol{\mu}_t$ is Gaussian with mean $\boldsymbol{x}_t$ and covariance matrix Σ_t formed by zeroing out the top-left element of $(\tau P_t)^{-1}$. We say that the system reaches *truthful consensus* if both the mean process $\boldsymbol{x}_t$ and the covariance matrix tend to zero as t goes to infinity. This indicates that all the agents' beliefs share a common mean equal to the truth and the "error bars" vanish over time. In view of (1), the covariance matrix indeed tends to 0 as long as the degrees are nonzero infinitely often, a trivial condition. To establish truthful consensus, therefore, boils down to studying the mean process $\boldsymbol{x}_t$. We do this in two parts: first, we show that the expected mean process converges to the truth; then we prove that fluctuations around it eventually vanish almost surely.[1]

1. The Kullback-Leibler divergence (Jadbabaie et al., 2012) is not suitable here because the estimator is Gaussian, hence continuous, whereas the truth is a single-point distribution.

Truth-hearing assumption: Given any interval of length $\kappa := \lfloor 1/\gamma \rfloor$, every agent $i > 0$ has an edge $(i, 0)$ in G_t for at least one value of t in that interval.

Theorem 6 *Under the truth-hearing assumption, the system reaches truthful consensus with a convergence rate bounded by $O(t^{-\gamma/2\eta})$, where η is the maximum outdegree over all the networks.*

We prove the theorem in the next two sections. It will follow directly from Lemmas 7 and 8 below. The convergence rate can be improved to the order of $t^{-(1-\varepsilon)\gamma/\eta}$, for arbitrarily small $\varepsilon > 0$. The inverse dependency on γ is not surprising: the more access to the truth the stronger the attraction to it. On the other hand, it might seem counterintuitive that a larger outdegree should slow down convergence. This illustrates the risk of groupthink. It pays to follow the crowds when the crowds are right. When they are not, however, this distracts from the lonely voice that happens to be right.

How essential is the truth-hearing assumption? We show that it is necessary. Simply having access to the truth infinitely often is not enough to achieve truthful consensus.

5.3. The proofs

In this subsection, we demonstrate technical details for the proof of the results. We begin with some repeated used inequalities.

5.3.1. USEFUL MATRIX INEQUALITIES

We highlight certain matrix inequalities to be used throughout. We use the standard element-wise notation $R \le S$ to indicate that $R_{ij} \le S_{ij}$ for all i, j. The infinity norm $\|R\|_\infty = \max_i \sum_j |r_{ij}|$ is submultiplicative: $\|RS\|_\infty \le \|R\|_\infty \|S\|_\infty$, for any matching rectangular matrices. On the other hand, the max-norm $\|R\|_{\max} := \max_{i,j} |r_{ij}|$ is not, but it is transpose-invariant and also satisfies: $\|RS\|_{\max} \le \|R\|_\infty \|S\|_{\max}$. It follows that

$$\begin{aligned} \|RSR^T\|_{\max} &\le \|R\|_\infty \|SR^T\|_{\max} = \|R\|_\infty \|RS^T\|_{\max} \\ &\le \|R\|_\infty^2 \|S^T\|_{\max} = \|R\|_\infty^2 \|S\|_{\max}. \end{aligned} \tag{34}$$

5.3.2. THE EXPECTED MEAN PROCESS DYNAMICS

We analyze the convergence of the mean process in expectation. The expected mean $\boldsymbol{y}_t = \mathbb{E}\,\boldsymbol{x}_t$ evolves through an averaging process entirely determined by the initial value $\boldsymbol{y}_0 = (0, x_{0,1}, \ldots, x_{0,n})^T$ and the graph sequence G_t. Intuitively, if an agent communicates repeatedly with a holder of the truth, the weight of the latter should accumulate and increasingy influence the belief of the agent in question. Our goal in this section is to prove the following result:

Lemma 7 *Under the truth-hearing assumption, the expected mean process $\boldsymbol{y}_t$ converges to the truth asymptotically. If, at each step, no agent receives information from more than η agents, then the convergence rate is bounded by $Ct^{-\gamma/2\eta}$, where C is a constant that depends on $\boldsymbol{x}_0, \gamma, \eta, \sigma_0/\sigma$.*

Proof We define B_t as the matrix formed by removing the first row and the first column from the stochastic $P_{t+1}^{-1}(P_t + A_t)$. If we write $\boldsymbol{y}_t$ as $(0, \boldsymbol{z}_t)$ then, by (33),

$$\begin{pmatrix} 0 \\ \boldsymbol{z}_{t+1} \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{0} \\ \boldsymbol{\alpha}_t & B_t \end{pmatrix} \begin{pmatrix} 0 \\ \boldsymbol{z}_t \end{pmatrix}, \tag{35}$$

where $\boldsymbol{\alpha}_{t,i} = (P_{t+1}^{-1})_{ii}$ if there is an edge $(i, 0)$ at time t and $\boldsymbol{\alpha}_{t,i} = 0$ otherwise. This further simplifies to

$$\boldsymbol{z}_{t+1} = B_t \boldsymbol{z}_t. \tag{36}$$

Let $\mathbf{1}$ be the all-one column vector of length n. Since $P_{t+1}^{-1}(P_t + A_t)$ is stochastic,

$$\boldsymbol{\alpha}_t + B_t \mathbf{1} = \mathbf{1} \tag{37}$$

In matrix terms, the truth-hearing assumption means that, for any $t \geq 0$,

$$\boldsymbol{\alpha}_t + \boldsymbol{\alpha}_{t+1} + \cdots + \boldsymbol{\alpha}_{t+\kappa-1} \geq Q_{t+\kappa}^{-1} \mathbf{1}, \tag{38}$$

where Q_t is the matrix derived from P_t by removing the first row and the last column; the inequality relies on the fact that P_t is monotonically nondecreasing. For any $t > s \geq 0$, we define the product matrix $B_{t:s}$ defined as

$$B_{t:s} := B_{t-1} B_{t-2} \ldots B_s, \tag{39}$$

with $B_{t:t} = I$. By (36), for any $t > s \geq 0$,

$$\boldsymbol{z}_t = B_{t:s}\, \boldsymbol{z}_s. \tag{40}$$

To bound the infinity norm of $B_{t:0}$, we observe that, for any $0 \leq l < \kappa - 1$, the i-th diagonal element of $B_{s+\kappa:s+l+1}$ is lower-bounded by

$$\begin{aligned} \prod_{j=l+1}^{\kappa-1} (B_{s+j})_{ii} &= \prod_{j=l+1}^{\kappa-1} \frac{(P_{s+j} + A_{s+j})_{ii}}{(P_{s+j+1})_{ii}} \\ \geq \prod_{j=l+1}^{\kappa-1} \frac{(P_{s+j})_{ii}}{(P_{s+j+1})_{ii}} &= \frac{(P_{s+l+1})_{ii}}{(P_{s+\kappa})_{ii}} \geq \frac{(P_s)_{ii}}{(P_{s+\kappa})_{ii}}. \end{aligned} \tag{41}$$

The inequalities follow from the nonnegativity of the entries and the monotonicity of $(P_t)_{ii}$. Note that (41) also holds for $l = \kappa - 1$ since $(B_{s+\kappa:s+\kappa})_{ii} = 1$.

Since $P_{t+1}^{-1}(P_t + A_t)$ is stochastic, the row-sum of B_t does not exceed 1; therefore, by pre-multiplying $B_{s+1}, B_{s+2}, \ldots$ on both sides of (37), we obtain:

$$B_{s+\kappa:s} \mathbf{1} \leq \mathbf{1} - \sum_{l=0}^{\kappa-1} B_{s+\kappa:s+l+1} \boldsymbol{\alpha}_{s+l}. \tag{42}$$

Noting that $\|B_t\|_\infty = \|B_t \mathbf{1}\|_\infty$ for any t, as B_t is non-negative, we combine (38), (41), and (42) together to derive:

$$\|B_{s+\kappa:s}\|_\infty \leq 1 - \min_{i>0} \frac{(P_s)_{ii}}{(P_{s+\kappa})_{ii}^2}. \tag{43}$$

Let $\eta := \max_{t \geq 0} \max_{1 \leq i \leq n} \eta_{t,i}$ denote the maximum outdegree in all the networks, and define $\delta = \min\{\tau_0/\tau, 1\}$. For any $i > 0$ and $s \geq \kappa$,

$$\frac{s\delta}{\kappa} \leq (P_s)_{ii} \leq \eta s + \frac{\tau_0}{\tau}; \tag{44}$$

hence,

$$\max_i (P_{s+\kappa})_{ii} \leq \eta(s + \kappa) + \frac{\tau_0}{\tau}. \tag{45}$$

It follows that

$$\frac{(P_{s+\kappa})_{ii} - (P_s)_{ii}}{(P_{s+\kappa})_{ii}} = \frac{\sum_{l=0}^{\kappa-1} \eta_{s+l,i}}{(P_{s+\kappa})_{ii}} \leq \frac{\eta \kappa^2 \delta^{-1}}{s + \kappa}. \tag{46}$$

Thus, we have

$$\begin{aligned} \min_{i>0} \frac{(P_s)_{ii}}{(P_{s+\kappa})_{ii}} &= 1 - \max_{i>0} \frac{(P_{s+\kappa})_{ii} - (P_s)_{ii}}{(P_{s+\kappa})ii} \\ &\geq 1 - \frac{\eta \kappa^2 \delta^{-1}}{s + \kappa}. \end{aligned} \tag{47}$$

We can replace the upper bound of (43) by

$$1 - \frac{1}{\max_{i>0} (P_{s+\kappa})_{ii}} \min_{i>0} \frac{(P_s)_{ii}}{(P_{s+\kappa})_{ii}^2},$$

which, together with (45) and (47) gives us

$$\begin{aligned} \|B_{s+\kappa:s}\|_\infty &\leq 1 - \frac{1}{\eta(s+\kappa) + \tau_0/\tau} \left(1 - \frac{\eta \kappa^2 \delta^{-1}}{s + \kappa}\right) \\ &\leq 1 - \frac{1}{2\eta\kappa(m+2)}. \end{aligned} \tag{48}$$

The latter inequality holds as long as $s = m\kappa > 0$ and

$$m \geq m^* := \frac{2\eta\kappa}{\delta} + \frac{\tau_0}{\eta\kappa\tau}.$$

It follows that, for $m_0 \geq m^*$,

$$\begin{aligned} \|B_{(m_0+m)\kappa:m_0\kappa}\|_\infty &\leq \prod_{j=2}^{m+1} \left(1 - \frac{1}{2\eta\kappa(m_0 + j)}\right) \\ &\leq \exp\left\{-\frac{1}{2\eta\kappa} \sum_{j=2}^{m+1} \frac{1}{m_0 + j}\right\}. \end{aligned} \tag{49}$$

The matrices B_t are sub-stochastic so that

$$\|B_t\, \boldsymbol{z}\|_\infty \leq \|B_t\|_\infty \|\boldsymbol{z}\|_\infty \leq \|\boldsymbol{z}\|_\infty.$$

By (40), for any $t \geq (m_0 + m)\kappa$,

$$\boldsymbol{z}_t = B_{t:(m_0+m)\kappa} B_{(m_0+m)\kappa:m_0\kappa}\, \boldsymbol{z}_{m_0},$$

so that, by using standard bounds for the harmonic series, $\ln(k+1) < 1 + \frac{1}{2} + \cdots + \frac{1}{k} \leq 1 + \ln k$, we find that

$$\begin{aligned}\|\boldsymbol{z}_t\|_\infty &\leq \|B_{(m_0+m)\kappa:m_0\kappa}\, \boldsymbol{z}_{m_0}\|_\infty \\ &\leq \|B_{(m_0+m)\kappa:m_0\kappa}\|_\infty \|\boldsymbol{z}_0\|_\infty \\ &\leq C t^{-1/(2\eta\kappa)},\end{aligned}$$

where $C > 0$ depends on $\boldsymbol{z}_0, \kappa, \eta, \tau_0/\tau$. We note that the convergence rate can be improved to the order of $t^{-(1-\varepsilon)\gamma/\eta}$, for arbitrarily small $\varepsilon > 0$, by working a little harder with (48). ■

5.3.3. THE MEAN PROCESS DYNAMICS

Recall that $\mu_{t,i} \sim \mathcal{N}(x_{t,i}, \tau_{t,i}^{-1})$, where $\tau_{t,i}$ denotes the precision $\sigma_{t,i}^{-2}$. A key observation about the updating rule in (1) is that the precision $\tau_{t,i}$ is entirely determined by the graph sequence G_t and is independent of the actual dynamics. Adding to this the connectivity property implied by the truth-hearing assumption, we find immediately that $\tau_{t,i} \to \infty$ for any agent i. This ensures that the covariance matrix Σ_t tends to 0 as t goes to infinity, which satisfies the second criterion for truthful consensus. The first criterion requires that the mean process $\boldsymbol{x}_t$ should converge to the truth $\mathbf{0}$. Take the vector $\boldsymbol{x}_t - \boldsymbol{y}_t$ and remove the first coordinate $(\boldsymbol{x}_t - \boldsymbol{y}_t)_0$ to form the vector $\boldsymbol{\Delta}_t \in \mathbb{R}^n$. Under the truth-hearing assumption, we have seen that $\boldsymbol{y}_t \to \mathbf{0}$ (Lemma 7), so it suffices to prove the following:

Lemma 8 *Under the truth-hearing assumption, the deviation $\boldsymbol{\Delta}_t$ vanishes almost surely.*

Proof We use a fourth-moment argument. The justification for the high order is technical: it is necessary to make a certain "deviation power" series converge. By (32), $\boldsymbol{x}_t$ is a linear combination of independent Gaussian random vectors $\boldsymbol{u}_s$ and $\boldsymbol{\varepsilon}_s$ for $0 \leq s \leq t-1$, and thus $\boldsymbol{x}_t$ itself is a Gaussian random vector. Therefore $\boldsymbol{\Delta}_t$ is also Gaussian and its mean is zero. From Markov's inequality, for any $c > 0$,

$$\sum_{t\geq 0} \mathbb{P}[|\Delta_{t,i}| \geq c] \leq \sum_{t\geq 0} \frac{\mathbb{E}\,\Delta_{t,i}^4}{c^4}. \tag{50}$$

If we are able to show the right hand side of (50) is finite for any $c > 0$, then, by the Borel-Cantelli lemma, with probability one, the event $|\Delta_{t,i}| \geq c$ occurs only a finite number of times, and so $\Delta_{t,i}$ goes to zero almost surely. Therefore, we only need to analyze the order of the fourth moment $\mathbb{E}\,\Delta_{t,i}^4$. By subtracting (33) from (32), we have:

$$\boldsymbol{\Delta}_{t+1} = B_t \boldsymbol{\Delta}_t + M_t \boldsymbol{v}_t, \tag{51}$$

where $\boldsymbol{v}_t := \boldsymbol{u}_t + \boldsymbol{\varepsilon}_t$ and $M_t := P_{t+1}^{-1} A_t$; actually, for dimensions to match, we remove the top coordinate of $\boldsymbol{v}_t$ and the first row and first column of M_t (see previous section for definition of B_t). Transforming the previous identity into a telescoping sum, it follows from $\boldsymbol{\Delta}_0 = \boldsymbol{x}_0 - \boldsymbol{y}_0 = \mathbf{0}$ and the definition $B_{t:s} = B_{t-1} B_{t-2} \ldots B_s$ that

$$\boldsymbol{\Delta}_t = \sum_{s=0}^{t-1} B_{t:s+1} M_s \boldsymbol{v}_s = \sum_{s=0}^{t-1} R_{t,s} \boldsymbol{v}_s, \tag{52}$$

where $R_{t,s} := B_{t:s+1} M_s$. We denote by $C_1, C_2, \ldots$ suitably large constants (possibly depending on $\kappa, \eta, n, \tau, \tau_0$). By (44), $\|M_s\|_\infty \leq C_1/(s+1)$ and, by (49), for sufficiently large s,

$$\|B_{t:s+1}\|_\infty \leq C_2 (s+1)^\beta (t+1)^{-\beta},$$

where $\beta = 1/2\eta\kappa < 1$. Combining the above inequalities, we obtain the following estimate of $R_{t,s}$ as

$$\|R_{t,s}\|_\infty \leq C_3 (s+1)^{-1+\beta} (t+1)^{-\beta}. \tag{53}$$

In the remainder of the proof, the power of a vector is understood element-wise. We use the fact that $\boldsymbol{v}_s$ and $\boldsymbol{v}_{s'}$ are independent if $s \neq s'$ and that the expectation of an odd power of an unbiased Gaussian is always zero. By Cauchy-Schwarz and Jensen's inequalities,

$$\begin{aligned}\mathbb{E}\,\boldsymbol{\Delta}_t^4 &= \left(\sum_{s=0}^{t-1} R_{t,s} \boldsymbol{v}_s\right)^4 \\ &= \sum_{s=0}^{t-1} \mathbb{E}(R_{t,s}\boldsymbol{v}_s)^4 + \sum_{0\leq s\neq s'<t} 3\,\mathbb{E}(R_{t,s}\boldsymbol{v}_s)^2 \mathbb{E}(R_{t,s'}\boldsymbol{v}_{s'})^2 \\ &\leq \sum_{s=0}^{t-1} \mathbb{E}(R_{t,s}\boldsymbol{v}_s)^4 + 3\left(\sum_{s=0}^{t-1} \mathbb{E}(R_{t,s}\boldsymbol{v}_s)^2\right)^2 \\ &\leq \sum_{s=0}^{t-1} \mathbb{E}(R_{t,s}\boldsymbol{v}_s)^4 + 3t \sum_{s=0}^{t-1} \mathbb{E}^2(R_{t,s}\boldsymbol{v}_s)^2 \\ &\leq (3t+1) \sum_{s=0}^{t-1} \mathbb{E}(R_{t,s}\boldsymbol{v}_s)^4. \end{aligned} \tag{54}$$

Notice that since the variance of $\boldsymbol{v}_t = (v_{t,1}, \ldots, v_{t,n})^T$ is nonincreasing, there exists a constant C_4 such that $\mathbb{E}\, v_{t,i}^4 \leq C_4$. By Jensen's inequality and the fact that the variables $v_{t,i}$ are independent for different values of i, we have, for any i, j, k, l,

$$|\mathbb{E}\, v_{t,i} v_{t,j} v_{t,k} v_{t,l}| \leq \max_k \mathbb{E}\, v_{t,k}^4.$$

By direct calculation, it then follows that

$$\begin{aligned}\max_i \mathbb{E}(R_{t,s}\boldsymbol{v}_s)_i^4 &= \max_i \mathbb{E}\Big(\sum_{j=1}^n (R_{t,s})_{ij} v_{s,j}\Big)^4 \\ &\leq \max_i \Big(\sum_{j=1}^n (R_{t,s})_{i,j}\Big)^4 \max_k \mathbb{E}\, v_{s,k}^4 \\ &= \|R_{t,s}\|_\infty^4 \max_k \mathbb{E}\, v_{s,k}^4 \\ &\leq C_5 (s+1)^{-4+4\beta} (t+1)^{-4\beta}. \end{aligned} \tag{55}$$

Summing (55) over $0 \leq s \leq t-1$, we conclude from (54) that $\mathbb{E}\,\boldsymbol{\Delta}_t^4 \leq C_6 t^{-2}$, and thus

$$\sum_{t\geq 0} \mathbb{E}\,\boldsymbol{\Delta}_t^4 \leq C_6 \sum_{t\geq 1} t^{-2} \leq C_7. \tag{56}$$

By the Borel-Cantelli lemma, it follows that $\boldsymbol{\Delta}_t$ vanishes almost surely. ■

Theorem 6 follows directly from Lemmas 7 and 8.

Why the truth-hearing assumption is necessary. We describe a sequence of graphs G_t that allows every agent infinite access to the truth and yet does not lead to truthful consensus. For this, it suffices to ensure that the expected mean process $\boldsymbol{y}_t$ does not converge. Consider a system with two learning agents with priors $\mu_{0,1}$ and $\mu_{0,2}$ from the same distribution $\mathcal{N}(2,1)$. We have $x_{0,1} = x_{0,2} = y_{0,1} = y_{0,2} = 2$ and, as usual, the truth is assumed to be 0; the noise variance is $\sigma^2 = 1$. The graph sequence is defined as follows: set $t_1 = 0$; for $k = 1, 2, \ldots$, agent 1 links to the truth agent at time t_k and to agent 2 at times $t_k + 1, \ldots, s_k - 1$; then at time s_k, agent 2 links to the truth agent, and then to agent 1 at times $s_k + 1, \ldots, t_{k+1} - 1$. Other than the links mentioned, we assume no additional link exists. The time points s_k and t_k are defined recursively to ensure that

$$y_{s_k,1} \geq 1 + 2^{-2k+1} \quad \text{and} \quad y_{t_k,2} \geq 1 + 2^{-2k}. \tag{57}$$

In this way, the expected mean processes of the two agents alternate while possibly sliding down toward 1 but never lower. The existence of these time points can be proved by induction. Since $y_{0,2} = 2$, the inequality $y_{t_k,2} \geq 1 + 2^{-2k}$ holds for $k = 1$, so let's assume it holds up to $k > 0$. The key to the proof is that, by (33), as agent 1 repeatedly links to agent 2, she is pulled arbitrarily close to it. Indeed, the transition rule gives us

$$y_{t+1,1} = \frac{(P_t)_{11}}{(P_{t+1})_{11}}\, y_{t,1} + \frac{1}{(P_{t+1})_{11}}\, y_{t,2},$$

where $(P_{t+1})_{11} = (P_t)_{11} + 1$, which implies that $y_{t,1}$ can be brought arbitrarily close to $y_{t,2}$ while the latter does not move: this follows from the fact that any product of the form $\prod_{t>t_a}^{t_b} \frac{t}{t+1}$ tends to 0 as t_b grows.[2] Thus a suitably increasing sequence of s_k, t_k ensures the two conditions (57). The beliefs of the two agents do not converge to the truth even though they link to the truth agent infinitely often.

Acknowledgments

We wish to thank the anonymous referees for their useful comments and suggestions. Some results of this work were previously published in the conferences, Innovations in Theoretical Computer Science (ITCS 2017), and American Control Conference (ACC 2017) by the same authors. Chu Wang did this work prior to joining Amazon. The research of Bernard Chazelle was sponsored by the Army Research Office and the Defense Advanced Research Projects Agency and was accomplished under Grant Number W911NF-17-1-0078. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office, the Defense Advanced Research Projects Agency, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

2. We note that the construction shares a family resemblance with one used by Moreau (Moreau, 2005) to show the non-consensual dynamics of certain multiagent averaging systems. The difference here is that the weights of the averaging change at each step by increasing the agent's self-confidence.

References

Daron Acemoglu and Asuman Ozdaglar. Opinion dynamics and learning in social networks. *Dynamic Games and Applications*, 1(1):3–49, 2011.

Daron Acemoglu, Munther A Dahleh, Ilan Lobel, and Asuman Ozdaglar. Bayesian learning in social networks. *The Review of Economic Studies*, 78(4):1201–1236, 2011.

Venkatesh Bala and Sanjeev Goyal. Learning from neighbours. *The review of economic studies*, 65 (3):595–621, 1998.

Abhijit V Banerjee. A simple model of herd behavior. *The quarterly journal of economics*, 107(3): 797–817, 1992.

FC Bartlett. Remembering: a study in experimental and social psychology (1932).

Aaron Beppu and Thomas L Griffiths. Iterated learning and the cultural ratchet. In *Proceedings of the 31st annual conference of the cognitive science society*, pages 2089–2094, 2009.

A Bhattachayya. On a measure of divergence between two statistical population defined by their population distributions. *Bulletin Calcutta Mathematical Society*, 35:99–109, 1943.

George E.P. Box and George C. Tiao. *Bayesian inference in statistical analysis*, volume 40. John Wiley & Sons, 2011.

Bernard Chazelle. Diffusive influence systems. *SIAM Journal on Computing*, 44(5):1403–1442, 2015.

Bernard Chazelle and Chu Wang. Inertial Hegselmann-Krause systems. In *Proceedings of the IEEE American Control Conference (ACC)*, pages 1936–1941, 2016.

Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 142–149. IEEE, 2000.

Kenneth R Davidson and Stanislaw J Szarek. Local operator theory, random matrices and banach spaces. *Handbook of the geometry of Banach spaces*, 1(317-366):131, 2001.

Morris H. DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345): 118–121, 1974.

Alan Edelman. Eigenvalues and condition numbers of random matrices. *SIAM Journal on Matrix Analysis and Applications*, 9(4):543–560, 1988.

Andrew Gelman, Hal S Stern, John B Carlin, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 2013.

Benjamin Golub and Matthew O. Jackson. Naive learning in social networks and the wisdom of crowds. *American Economic Journal: Microeconomics*, 2(1):112–149, 2010.

Benjamin Golub and Matthew O. Jackson. How homophily affects the speed of learning and best response dynamics. 2012.

Thomas L. Griffiths and Michael L. Kalish. A bayesian view of language evolution by iterated learning. In *Proceedings of the 27th annual conference of the cognitive science society*, pages 827–832, 2005.

Thomas L. Griffiths and Michael L. Kalish. Language evolution by iterated learning with bayesian agents. *Cognitive Science*, 31(3):441–480, 2007.

Thomas L Griffiths, Michael L Kalish, and Stephan Lewandowsky. Theoretical and empirical evidence for the impact of inductive biases on cultural evolution. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 363(1509):3503–3514, 2008.

Michiel Hazewinkel. *Encyclopaedia of Mathematics*. Springer Science & Business Media, 2013.

Rainer Hegselmann and Ulrich Krause. Opinion dynamics and bounded confidence models, analysis, and simulation. *Journal of Artificial Societies and Social Simulation*, 5(3), 2002.

Ali Jadbabaie, Pooya Molavi, Alvaro Sandroni, and Alireza Tahbaz-Salehi. Non-bayesian social learning. *Games and Economic Behavior*, 76(1):210–225, 2012.

Ali Jadbabaie, Pooya Molavi, and Alireza Tahbaz-Salehi. Information heterogeneity and the speed of learning in social networks. *Columbia Business School Research Paper*, (13-28), 2013.

Michael L Kalish, Thomas L Griffiths, and Stephan Lewandowsky. Iterated learning: Intergenerational knowledge transmission reveals inductive biases. *Psychonomic Bulletin & Review*, 14(2): 288–294, 2007.

Simon Kirby, Tom Griffiths, and Kenny Smith. Iterated learning and the evolution of language. *Current opinion in neurobiology*, 28:108–114, 2014.

Ilan Lobel and Evan Sadler. Preferences, homophily, and social learning. *Operations Research*, 64 (3):564–584, 2015.

Soheil Mohajer and Behrouz Touri. On convergence rate of scalar hegselmann-krause dynamics. In *American Control Conference (ACC), 2013*, pages 206–210. IEEE, 2013.

Pooya Molavi, Alireza Tahbaz-Salehi, and Ali Jadbabaie. Foundations of non-bayesian social learning. *Columbia Business School Research Paper*, 2015.

Luc Moreau. Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, 50:169–182, 2005.

Elchanan Mossel, Allan Sly, and Omer Tamuz. From agreement to asymptotic learning. *Arxiv preprint arXiv*, 1105, 2011.

Manuel Mueller-Frank. A general framework for rational learning in social networks. *Theoretical Economics*, 8(1):1–40, 2013.

James R Norris. *Markov chains*. Cambridge university press, 1998.

Amy Perfors and Daniel Navarro. Language evolution is shaped by the structure of the world: An iterated learning analysis. Cognitive Science Society, 2011.

Anna N Rafferty, Thomas L Griffiths, and Dan Klein. Convergence bounds for language evolution by iterated learning. In *Proceedings of the Thirty-First Annual Conference of the Cognitive Science Society*, 2009.

Anna N Rafferty, Thomas L Griffiths, and Dan Klein. Analyzing the rate at which languages lose the influence of a common ancestor. *Cognitive science*, 38(7):1406–1431, 2014.

Mohammad Amin Rahimian and Ali Jadbabaie. Learning without recall from actions of neighbors. In *2016 American Control Conference (ACC)*, pages 1060–1065. IEEE, 2016a.

Mohammad Amin Rahimian and Ali Jadbabaie. Naive social learning in ising networks. In *2016 American Control Conference (ACC)*, pages 1088–1093. IEEE, 2016b.

Mohammad Amin Rahimian, Shahin Shahrampour, and Ali Jadbabaie. Learning without recall by random walks on directed graphs. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 5538–5543. IEEE, 2015a.

Mohammad Amin Rahimian et al. Learning without recall: A case for log-linear learning. *IFAC-PapersOnLine*, 48(22):46–51, 2015b.

Mark Rudelson and Roman Vershynin. Smallest singular value of a random rectangular matrix. *Communications on Pure and Applied Mathematics*, 62(12):1707–1739, 2009.

Kenny Smith. Iterated learning in populations of bayesian agents. In *Proceedings of the 31st annual conference of the cognitive science society*, pages 697–702, 2009.

Alireza Tahbaz-Salehi, Alvaro Sandroni, and Ali Jadbabaie. Learning under social influence. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 1513–1519. IEEE, 2009.

Mónica Tamariz and Simon Kirby. Culture: copying, compression, and conventionality. *Cognitive science*, 39(1):171–183, 2015.

Appendix A. Analysis of the Root-Sine-Distance

In this appendix, we provide detailed discussion of the root-sine-distance. We will first show that it is a metric, followed by the proof of its equivalency to Euclidean distance and the discussion of its relation to other similar metrics. The notation in the appendix is local and should not be confused with the one used in the main body of this paper.

The root-sine-distance is a metric. The two forms of the function d_{RS} in (4) make it clear that $0 \le d_{RS}(\boldsymbol{a},\boldsymbol{b}) \le 1$ and $d_{RS}(\boldsymbol{a},\boldsymbol{b}) = 0$ if and only if $\boldsymbol{a}$ and $\boldsymbol{b}$ are identical. We easily check that d_{RS} makes the simplex $\mathcal{S}$ of distributions over $\mathcal{D}$ into a metric space. Indeed, $d_{RS}(\cdot,\cdot)$ is obviously symmetric, and $d_{RS}(\boldsymbol{a},\boldsymbol{b}) = 0$ implies that $\boldsymbol{a} = \boldsymbol{b}$. To check the triangular inequality, notice that

$$d_{RS}(\boldsymbol{a},\boldsymbol{b}) = \sqrt{1 - \Big(\sum_{i=1}^{s} \sqrt{a_i b_i}\Big)^2} = \sin\langle \sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\,\rangle, \tag{58}$$

where $\langle \sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\rangle$ is the angle between the unit vectors $\sqrt{\boldsymbol{a}}$ and $\sqrt{\boldsymbol{b}}$, using the notation $\sqrt{\boldsymbol{v}} = (\sqrt{v_1}, \ldots, \sqrt{v_s})$. To prove that $d_{RS}(\boldsymbol{a},\boldsymbol{b}) + d_{RS}(\boldsymbol{b},\boldsymbol{c}) \ge d_{RS}(\boldsymbol{a},\boldsymbol{c})$ for any $\boldsymbol{a},\boldsymbol{b},\boldsymbol{c} \in \mathcal{S}$, we denote by α,β,γ the corresponding angles in that order, ie, $\alpha = \langle \sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\rangle$, etc. The coordinates in $\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}$ are nonnegative; therefore $0 \le \alpha,\beta,\gamma \le \pi/2$. These form the three angles at the origin of a tetrahedron with a vertex at the origin; therefore, by the triangular inequality in spherical geometry, $\alpha+\beta \ge \gamma$. If $\alpha+\beta \le \frac{\pi}{2}$, then $\sin\alpha + \sin\beta \ge \sin\alpha\cos\beta + \cos\alpha\sin\beta = \sin(\alpha+\beta) \ge \sin\gamma$. On the other hand, if $\alpha+\beta > \pi/2$, then $\sin\alpha+\sin\beta = 2\sin\frac{\alpha+\beta}{2}\cos\frac{\alpha-\beta}{2} \ge 2\sin\frac{\pi}{4}\cos\frac{\pi}{4} = 1 \ge \sin\gamma$, which establishes the triangular inequality.

Relation to the Euclidean distance. Shrinking the simplex $\mathcal{S}$ by a tiny amount, we define $\mathcal{S}_\varepsilon := \{\boldsymbol{a} \in \mathcal{S} : \varepsilon \le a_i \le 1-\varepsilon\}$ and note that

$$d_E(\boldsymbol{a},\boldsymbol{b}) := \|\boldsymbol{a}-\boldsymbol{b}\|_2 = \sqrt{\sum_{i=1}^{s} (\sqrt{a_i} - \sqrt{b_i})^2 (\sqrt{a_i} + \sqrt{b_i})^2}.$$

It follows that, for $\boldsymbol{a},\boldsymbol{b} \in \mathcal{S}_\varepsilon$,

$$\frac{1}{2} d_E(\boldsymbol{a},\boldsymbol{b}) \le d_E(\sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\,) \le \frac{1}{2\sqrt{\varepsilon}}\, d_E(\boldsymbol{a},\boldsymbol{b}). \tag{59}$$

On the other hand, $\|\sqrt{\boldsymbol{a}}\|_2 = \|\sqrt{\boldsymbol{b}}\|_2 = 1$, so the vectors $\sqrt{\boldsymbol{a}}$ and $\sqrt{\boldsymbol{b}}$ form an isosceles triangle; hence

$$d_E(\sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\,) = 2\sin\frac{1}{2}\langle \sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\rangle = \frac{\sin\langle \sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\rangle}{\cos\frac{1}{2}\langle \sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\rangle} = \frac{d_{RS}(\boldsymbol{a},\boldsymbol{b})}{\cos\frac{1}{2}\langle \sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\rangle}\,.$$

Since $0 \le \langle \sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\rangle \le \frac{\pi}{2}$,

$$d_{RS}(\boldsymbol{a},\boldsymbol{b}) \le d_E(\sqrt{\boldsymbol{a}}, \sqrt{\boldsymbol{b}}\,) \le \sqrt{2}\, d_{RS}(\boldsymbol{a},\boldsymbol{b}).$$

Together with (59) this shows that, for any $\boldsymbol{a},\boldsymbol{b} \in \mathcal{S}_\varepsilon$,

$$\frac{1}{2\sqrt{2}}\, d_E(\boldsymbol{a},\boldsymbol{b}) \le d_{RS}(\boldsymbol{a},\boldsymbol{b}) \le \frac{1}{2\sqrt{\varepsilon}}\, d_E(\boldsymbol{a},\boldsymbol{b}), \tag{60}$$

which shows that the Euclidean distance and the metric d_{RS} are equivalent in $\mathcal{S}_\varepsilon$.

Relation to other distances. The metric d_{RS} is related to the Hellinger and Bhattacharyya distances. Writing $C(\boldsymbol{a},\boldsymbol{b}) = \sum_{i=1}^{s} \sqrt{a_i b_i}$ (Comaniciu et al., 2000), then $d_{RS}(\boldsymbol{a},\boldsymbol{b}) = \sqrt{1 - C(\boldsymbol{a},\boldsymbol{b})^2}$. The Hellinger distance is defined as $d_H(\boldsymbol{a},\boldsymbol{b}) = \sqrt{1 - C(\boldsymbol{a},\boldsymbol{b})}$ (Hazewinkel, 2013), while the Bhattacharyya distance is defined as $d_B(\boldsymbol{a},\boldsymbol{b}) = -\ln C(\boldsymbol{a},\boldsymbol{b})$ (Bhattachayya, 1943). The total variation distance d_{TV} is half the ℓ_1-norm; therefore $d_{TV}(\boldsymbol{a},\boldsymbol{b}) \le \frac{1}{2}\sqrt{s}\, d_E(\boldsymbol{a},\boldsymbol{b})$. Combining these observations with (60) establishes (5):

$$\begin{cases} d_H = \sqrt{1 - \sqrt{1 - d_{RS}^2}}\,; \\ d_B = -\frac{1}{2}\ln(1 - d_{RS}^2)\,; \\ d_{TV} \le \sqrt{2s}\, d_{RS}. \end{cases}$$

Journal of Machine Learning Research 20 (2019) 1-34 Submitted 3/16; Revised 1/19; Published 1/19

Exact Clustering of Weighted Graphs via Semidefinite Programming

Aleksis Pirinen ALEKSIS@MATHS.LTH.SE
Centre for Mathematical Sciences
Lund University
Lund, Sweden

Brendan Ames BPAMES@UA.EDU
Department of Mathematics
The University of Alabama
Tuscaloosa, Alabama, AL 35487-0350, USA

Editor: Sujay Sanghavi

Abstract

As a model problem for clustering, we consider the densest k-disjoint-clique problem of partitioning a weighted complete graph into k disjoint subgraphs such that the sum of the densities of these subgraphs is maximized. We establish that such subgraphs can be recovered from the solution of a particular semidefinite relaxation with high probability if the input graph is sampled from a distribution of clusterable graphs. Specifically, the semidefinite relaxation is exact if the graph consists of k large disjoint subgraphs, corresponding to clusters, with weight concentrated within these subgraphs, plus a moderate number of nodes not belonging to any cluster. Further, we establish that if noise is weakly obscuring these clusters, i.e, the between-cluster edges are assigned very small weights, then we can recover significantly smaller clusters. For example, we show that in approximately sparse graphs, where the between-cluster weights tend to zero as the size n of the graph tends to infinity, we can recover clusters of size polylogarithmic in n under certain conditions on the distribution of edge weights. Empirical evidence from numerical simulations is also provided to support these theoretical phase transitions to perfect recovery of the cluster structure.

Keywords: clustering, densest subgraph, stochastic block models, semidefinite programming, sparse graphs

©2019 Aleksis Pirinen and Brendan Ames.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/16-128.html.

1. Introduction

Clustering is a fundamental problem in machine learning and statistics, focusing on the identification and classification of groups, called *clusters*, of similar items in a given data set. Clustering is ubiquitous, playing a prominent role in varied fields such as computational biology, information retrieval, pattern recognition, image processing and computer vision, and network analysis. This problem is inherently ill-posed, as the partition or clustering of any given data set will depend heavily on how we quantify similarity between items in the data set and how we characterize clusters; it is not outside the realm of possibility to have two drastically different clusterings of the same data if two different similarity metrics are used in the clustering process. Regardless of the similarity metric used, clustering is a combinatorial optimization problem at its core: given data, identify a partition or labeling of the data (approximately) maximizing some measure of quality of the clustering. Due to the difficulties inherent with optimization over discrete sets, many popular approaches for clustering involve the approximate solution of an NP-hard combinatorial optimization problem; for example, the spectral clustering heuristic for the normalized cut problem (Dhillon et al., 2004; Ng et al., 2002), the convex relaxation approaches for the correlation clustering problem (Mathieu and Schudy, 2010), robust principal component analysis (Chen et al., 2014a; Oymak and Hassibi, 2011), and the densest k-disjoint-clique problem (Ames and Vavasis, 2014; Ames, 2014), among many others.

In spite of the inherent intractability of clustering, many recent analyses have established that if data is sampled from some distribution of clusterable data, then one can efficiently recover the underlying cluster structure using a variety of clustering algorithms. In particular, the recent results of Abbe et al. (2016); Ailon et al. (2013); Ames and Vavasis (2014); Ames (2014); Amini and Levina (2018); Cai and Li (2015); Chen et al. (2014a,b); Chen and Xu (2014); Guédon and Vershynin (2015); Hajek et al. (2015); Lei and Rinaldo (2015); Mathieu and Schudy (2010); Nellore and Ward (2015); Oymak and Hassibi (2011); Rohe et al. (2011); Qin and Rohe (2013); Vinayak et al. (2014) all establish sufficient conditions under which we can expect to identify the latent cluster structure efficiently. Most of these results assume that the similarity structure of the data can be modeled as a graph sampled from some generalization of the *stochastic block model* proposed by Holland et al. (1983). In this model, the nodes of the graph, called the *similarity graph* of the data, are associated with the items in the data set. An edge is drawn between two items with fixed probability p if the corresponding items belong to the same cluster, and with fixed probability $q < p$ if the corresponding items belong to different clusters. Under this block model, the analyses cited above establish that the block structure of the data can be recovered in polynomial-time with high probability provided that the smallest cluster in the data is sufficiently large,

typically larger than $\tilde{c}\sqrt{n}$, where n denotes the number of items in the data (and nodes in the similarity graph) and $\tilde{c}$ is a polylogarithmic factor in n depending on $p-q$.

Although valuable in establishing sufficient conditions for data to be clusterable, these results are not immediately applicable to data sets seen in many applications, particularly those arising from the analysis of social networks. For example, statistical analysis of social networks suggests that communities, playing the role of clusters, tend to be limited in size to several hundred users, while the networks themselves can contain thousands, if not millions or even billions, of users (Leskovec et al., 2008, 2009). However, the recent analyses of Chen et al. (2014a); Chen and Xu (2014); Guédon and Vershynin (2015); Jalali et al. (2015); Rohe et al. (2014), among others, suggest that these clusterability results are overly conservative with respect to the size of clusters we can expect to recover in polynomial-time. Specifically, these analyses allow the edge probabilities p and q to vary with n, and investigate how the size of the smallest cluster that can be recovered depends on the relative scaling of p, q and n. In this case, the data is often assumed to be sampled from a *sparse* generalized stochastic block model where the parameters p and q governing edge formation are functions depending on the number of items n and one or both tends to 0 as $n \to \infty$. In the case where p tends to 0 much more slowly than q, the noise obscuring the block structure is significantly weaker than in the dense graph case (where p and q are assumed fixed). Here, sparsity refers to the fact that graphs generated according to the block model contain very few edges between clusters with high probability when n is large, and not that the graph itself is sparse in the sense that the nodes have small average degree. In this case, it has been shown that clusters significantly smaller than $\sqrt{n}$ can be recovered efficiently; specifically, several methods have been shown to recover clusters with size polylogarithmic in n under certain assumptions on the probability functions p and q (see Chen et al., 2014a; Chen and Xu, 2014; Guédon and Vershynin, 2015; Rohe et al., 2014). We should note that these results provide evidence of a computational limit for cluster recovery; that is, these results establish that clusters can be recovered in a computationally efficient way if the underlying data satisfies certain sufficient conditions. We should note further that the lower bounds on cluster size given by these sufficient conditions typically do not match information-theoretic limits; it is well-known that it is possible to identify clusters of size on the order of $\log n$ in certain settings, however, no polynomial-time algorithms are known to do so (Chen and Xu 2014; Hajek et al. 2015 provide further details).

The primary contribution of this paper is an analysis establishing similar clusterability results for a particular convex relaxation of the clustering problem. That is, we present an analysis establishing the following theorem, which provides conditions for perfect recovery of the underlying cluster structure from the solution of a particular semidefinite program. As an immediate corollary, the theorem establishes that one may identify clusters as small

as $\Omega(\log n)$, i.e., there exists constant c such that the size of the smallest cluster recoverable cluster is bounded below by $c \log n$ for sufficiently large n, with high probability if the data is sampled from the sparse block model described above for particular choices of p and q. Here, we say that an event occurs *with high probability (w.h.p.)* if the event occurs with probability tending polynomially to 1 as $n \to \infty$.

Theorem 1 *Suppose that the n-node graph $G = (V, E)$ is sampled from the generalized stochastic block model, with k disjoint blocks, in-cluster edge probability p, and between-cluster edge probability q. Let $\boldsymbol{A} \in \mathbf{R}^{n \times n}$ denote the adjacency matrix of G and let $\hat{r}$ and $\tilde{r}$ denote the cardinality of the smallest and largest clusters, respectively, in the block model for G. Then there exists constants $c_1, c_2, c_3 > 0$ such that the columns of the optimal solution $\boldsymbol{X}^*$ of the semidefinite program*

$$\max_{\boldsymbol{X} \in \Sigma_+^n} \{\mathrm{Tr}(\boldsymbol{A}\boldsymbol{X}) : \boldsymbol{X}\boldsymbol{e} \leq \boldsymbol{e}, \mathrm{Tr}(\boldsymbol{X}) = k, \boldsymbol{X} \geq \boldsymbol{0}\}$$

are scalar multiples of the characteristic vectors of the clusters in our underlying block model with high probability if

$$p - q \geq c_3 \max\left\{ \sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}} \right\},$$

where $\tilde{\sigma}^2 = \max\{p(1-p), q(1-q)\}$, and

$$(p-q)\hat{r} \geq c_1 \max\left\{ \sqrt{q(1-q)n}, \sqrt{\log n} \right\} + c_2 \max\left\{ \sqrt{p(1-p)\tilde{r}}, \sqrt{\log n} \right\}.$$

Moreover, in this case, every characteristic vector of a cluster in the block model is a scalar multiple of at least one column of $\boldsymbol{X}^*$.

Here, the characteristic vector of a set $S \subseteq \{1, 2, \ldots, n\}$ is the vector $\boldsymbol{x} \in \{0,1\}^n$ with ith element

$$x_i = \begin{cases} 1, & \text{if } i \in S \\ 0, & \text{otherwise.} \end{cases}$$

In Theorem 1, $\mathrm{Tr}(\boldsymbol{X})$ denotes the trace of the matrix $\boldsymbol{X}$, $\boldsymbol{e}$ denotes the all-ones vector of appropriate dimension, the notation $\boldsymbol{X} \geq \boldsymbol{0}$ indicates that the entries of $\boldsymbol{X}$ are nonnegative, and Σ_+^n denotes the cone of $n \times n$ symmetric positive semidefinite matrices.

Note that if G is sampled from the *dense* block model, i.e., p, q are independent of n, then Theorem 1 suggests that we have exact recovery if $\hat{r} \geq c\sqrt{n}$ with high probability, where c is a constant depending on p, q; this bound matches that established by Ames (2014) (among many others) up to constant terms. On the other hand, when G is sampled from the *sparse* block model, we see that Theorem 1 suggests that we may have perfect

recovery of significantly smaller clusters. For example, suppose that $p = 1$ is fixed and $q = \log n/n$. Then we have exact recovery with high probability if the smallest cluster has size $\hat{r} = \Omega(\log n)$; see the discussion following Theorem 3.

We will show that analogous phenomena occur in what we will call *approximately* sparse graphs. In many practical applications, the expectation that we have a binary labeling indicating whether any pair of items in a given data set are similar or dissimilar is unrealistic. However, it is often possible to describe the *level* of similarity between any two items using some affinity function based on distance between the items in question. For example, we could consider the discrepancy in pixel intensity and geographic location in image segmentation applications or Euclidean distance between two items represented as vectors in a Euclidean space (or some other vector space with corresponding norm). In this case, we can summarize the pairwise similarity relationships within our data using a weighted graph, called a *weighted similarity graph*. Specifically, given a data set with affinity function f, the weighted similarity graph is the weighted complete graph with nodes corresponding to the items in the data set, and edge weight w_{ij} between nodes i and j given by the value of $f(i, j)$. Clearly, this contains the similarity graphs discussed earlier as a special case where $w_{ij} = 1$ if items i and j are known to be similar and $w_{ij} = 0$ otherwise; note that we assume that we have an undirected graph with symmetric adjacency matrix.

We can generalize the stochastic block model in an identical fashion. We assume that items in the same cluster are significantly more similar than pairs of items in different clusters. This corresponds to edge weights within clusters being larger, on average, than edge weights between clusters. This motivates the following random graph model, which we will call the *planted cluster model*. Let $G = (V, \boldsymbol{W})$ be the weighted complete graph whose node set represents the items in some data set containing k clusters and (potentially) some nodes that will not be assigned to a cluster. For each pair of nodes u, v in the same cluster C_ℓ, $\ell \in \{1, 2, \ldots, k\}$, we randomly sample edge weight $w_{uv} \geq 0$, and w_{vu} by symmetry, from some probability distribution Ω_ℓ with mean $\alpha_\ell \geq \alpha > 0$. If $u \in C_i$, $v \in C_j$, where $i \neq j$, i.e., u, v do not belong to the same cluster, we sample $w_{uv} = w_{vu} \geq 0$ from a different probability distribution Ω_{ij} with mean $\beta_{ij} \leq \beta \in [0, \alpha)$. Note that this model contains the generalized stochastic block model discussed earlier as a special case when Ω_ℓ and Ω_{ij} are Bernoulli distributions with probabilities of success $p_\ell = p$ and $q_{ij} = q$, respectively.

It was shown by Ames (2014) that if $G = (V, \boldsymbol{W})$ is sampled from the planted cluster model with minimum cluster size at least $c\sqrt{n}$ in the homogeneous case where all within-cluster edges are i.i.d. with mean α and all between-cluster edges are i.i.d. with mean β, where c is a constant depending on α and β, then we can recover the clusters from the

optimal solution of the semidefinite program

$$\max_{\boldsymbol{X} \in \Sigma_+^n} \left\{ \operatorname{Tr}(\boldsymbol{W}\boldsymbol{X}) : \boldsymbol{X}\boldsymbol{e} \leq \boldsymbol{e}, \operatorname{Tr}(\boldsymbol{X}) = k, \boldsymbol{X} \geq \boldsymbol{0} \right\} \tag{1}$$

with high probability, where k is the number of clusters in the graph. We will show that these results can be strengthened to establish that much smaller clusters can be recovered in the presence of *approximately* sparse noise. That is, we will see that if the between-cluster edge weights have expectation β and variance σ_2^2 approaching zero sufficiently quickly as $n \to \infty$, then we may recover clusters containing as few as $\Omega(\log n)$ nodes with high probability. We will derive the semidefinite program (1) as a relaxation of a particular model problem for clustering in Section 2.1 and formally state our recovery guarantees in Section 2.2; we will see that these results immediately specialize to those stated in Theorem 1 for the semidefinite program (1).

2. Semidefinite Relaxations of the Densest k-Disjoint Clique Problem

In this section, we derive a semidefinite relaxation for the densest k-disjoint clique problem and present an analysis illustrating a sufficient condition ensuring that this relaxation is exact. This problem will act as a model problem for clustering and we will see that we should expect to accurately recover the underlying cluster structure if the given data satisfies this sufficient condition.

2.1. The Densest k-disjoint Clique Problem

We begin by deriving a heuristic for the clustering problem based on semidefinite relaxation of the densest disjoint clique problem. A similar discussion motivating the relaxation was originally presented by Ames (2014); we repeat it here for completeness. Let $K_n = (V, \boldsymbol{W})$ be a weighted complete graph with vertex set $V = \{1, 2, \ldots, n\}$ and nonnegative edge weights $w_{ij} \in [0, 1]$ for all $i, j \in V$. Given a subgraph H of K_n, the *density* d_H of H is the average edge weight incident at a vertex in H:

$$d_H = \sum_{ij \in E(H)} \frac{w_{ij}}{|V(H)|}.$$

If we assume that K_n is the similarity graph of some data set consisting of k disjoint clusters and that weight is concentrated more heavily on within-cluster edges than between-cluster edges, then we may cluster this data set by finding the set of k disjoint subgraphs, corresponding to these clusters, with maximum density; we call this problem the *densest k-partition problem*. Peng and Wei (2007) established that the densest k-partition problem is

NP-hard. Moreover, this partition model excludes the case where some items in the data set do not naturally associate with any of the clusters in the data. To simultaneously motivate a convex relaxation of the densest k-partition problem and address the inclusion of nodes that do not naturally belong to clusters, we consider the *densest k-disjoint clique problem*.

Given a graph $G = (V, E)$, a *clique* of G is a pairwise adjacent subset of V. That is, $C \subseteq V$ is a clique of G if $ij \in E$ for every pair of nodes $i, j \in C$ or, equivalently, the subgraph $G(C)$ induced by C is complete. We say that H is a *k-disjoint-clique* subgraph of K_n if $V(H)$ consists of k disjoint cliques, i.e., H is the union of k disjoint complete subgraphs of K_n. The *densest k-disjoint-clique problem* seeks a k-disjoint-clique subgraph H^* maximizing the sum of the densities of the disjoint complete subgraphs comprising H^*. Note that if we add the additional constraint that each node in K_n belongs to exactly one k-disjoint-clique subgraph in K_n, then the densest k-disjoint-clique problem becomes the densest k-partition problem. However, in general, the densest k-disjoint-clique problem allows an assignment of nodes to clusters, represented by the disjoint cliques, which excludes some nodes. For example, if such nodes are present in the data, they would not be assigned to a cluster by the optimal k-disjoint-clique subgraph.

The complexity of the densest k-disjoint-clique problem is unknown; in particular, no polynomial-time algorithm for its solution is known. To address this potential intractability, we will attempt to approximately solve the k-disjoint-clique problem by convex relaxation. Suppose that $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ are the characteristic vectors of a set of disjoint cliques $C_1, C_2, \ldots, C_k$ forming a k-disjoint-clique subgraph of K_n. Using this notation, the density of the complete subgraph induced by C_i is equal to

$$d_{G(C_i)} = \sum_{u,v \in C_i} \frac{w_{uv}}{|C_i|} = \frac{\boldsymbol{v}_i^T \boldsymbol{W} \boldsymbol{v}_i}{\boldsymbol{v}_i^T \boldsymbol{v}_i}.$$

If we let $\boldsymbol{P}$ be the $n \times k$ matrix with ith column equal to $\boldsymbol{v}_i / \|\boldsymbol{v}_i\|$, where $\|\cdot\| = \|\cdot\|_2$ denotes the standard Euclidean norm, then it is easy to see that

$$\sum_{i=1}^{k} d_{G(C_i)} = \operatorname{Tr}(\boldsymbol{P}^T \boldsymbol{W} \boldsymbol{P}).$$

We call such a matrix $\boldsymbol{P}$ a *normalized k-cluster matrix* and denote the set of normalized k-cluster matrices of the vertex set V by $ncm(V, k)$. It follows that the densest k-disjoint-clique problem may be formulated as

$$\max \left\{ \operatorname{Tr}(\boldsymbol{P}^T \boldsymbol{W} \boldsymbol{P}) : \boldsymbol{P} \in ncm(V, k) \right\}. \tag{2}$$

Again, the complexity of (2) is unknown, however, the maximization of quadratic functions subject to combinatorial constraints is known to be NP-hard.

A process for relaxation of (2) using matrix lifting is described by Ames (2014); a similar relaxation technique was applied by Ames and Vavasis (2011, 2014) and Ames (2015). In particular, each proposed cluster C_i, with characteristic vector $\boldsymbol{v}_i$, corresponds to the rank-one symmetric matrix

$$\boldsymbol{X}^{(i)} = \frac{\boldsymbol{v}_i \boldsymbol{v}_i^T}{\boldsymbol{v}_i^T \boldsymbol{v}_i}.$$

It is easy to see that the density of $G(C_i)$ is equal to

$$d_{G(C_i)} = \frac{\boldsymbol{v}_i^T \boldsymbol{W} \boldsymbol{v}_i}{\boldsymbol{v}_i^T \boldsymbol{v}_i} = \operatorname{Tr}(\boldsymbol{W} \boldsymbol{X}^{(i)}).$$

Moreover, each of the matrices $\boldsymbol{X}^{(i)}$ has row and column sums equal to either 0 or 1, and trace equal to 1. Finally, for each proposed clustering $C_1, \ldots, C_k$, the corresponding rank-one matrices are orthogonal in the trace inner product, due to the orthogonality of the characteristic vectors of the corresponding disjoint clusters. Thus, the matrix

$$\boldsymbol{X} = \sum_{i=1}^{k} \boldsymbol{X}^{(i)} = \sum_{i=1}^{k} \frac{\boldsymbol{v}_i \boldsymbol{v}_i^T}{\boldsymbol{v}_i^T \boldsymbol{v}_i} \tag{3}$$

has rank equal to k. This suggests that we may relax (2) as the rank-constrained semidefinite program

$$\max_{\boldsymbol{X} \in \Sigma_+^n} \left\{ \operatorname{Tr}(\boldsymbol{W} \boldsymbol{X}) : \boldsymbol{X} \boldsymbol{e} \leq \boldsymbol{e}, \operatorname{rank} \boldsymbol{X} = k, \operatorname{Tr} \boldsymbol{X} = k, \boldsymbol{X} \geq \boldsymbol{0} \right\}. \tag{4}$$

The relaxation (4) can be relaxed further to a semidefinite program by omitting the nonconvex rank constraint:

$$\max_{\boldsymbol{X} \in \Sigma_+^n} \left\{ \operatorname{Tr}(\boldsymbol{W} \boldsymbol{X}) : \boldsymbol{X} \boldsymbol{e} \leq \boldsymbol{e}, \operatorname{Tr} \boldsymbol{X} = k, \boldsymbol{X} \geq \boldsymbol{0} \right\}. \tag{5}$$

We should note that the semidefinite program (5) is remarkably similar to the semidefinite relaxation of the minimum sum of squared distance partition of Peng and Wei (2007) and the semidefinite relaxation of the maximum likelihood estimate of the stochastic block model considered by Amini and Levina (2018), among others, although our relaxation approach differs slightly from that used in these two papers.

2.2. Block Models and Recovery Guarantees

Given a set of clusterable data or, more accurately, a clusterable graph representation of data, Ames (2014) established that one can recover the underlying cluster structure from the optimal solution of the semidefinite program (5). Specifically, it is assumed that data with strong cluster structure should correspond to similarity graphs with heavy weight assigned to edges within clusters, relative to that between cluster edges. This corresponds to pairs of

items within clusters being significantly more similar than pairs of items in different clusters. This motivates the following block model.

Let H^* be a k-disjoint-clique subgraph of $K_n = (V, \boldsymbol{W})$ with vertex set composed of the disjoint cliques $C_1, \ldots, C_k$ and let Σ^n denote the set of all $n \times n$ symmetric matrices. We consider weight matrices $\boldsymbol{W} = [w_{ij}] \in \Sigma^n$ with entries sampled independently from one of two probability distributions Ω_1, Ω_2 as follows.

- For each $i = 1, \ldots, k$ and each $u, v \in C_i$, we sample $w_{uv} = w_{vu}$ from a distribution Ω_1 such that
$$\mathbf{E}[w_{uv}] = \mathbf{E}[w_{vu}] = \alpha, \qquad 0 \leq w_{uv} \leq 1,$$
for fixed $\alpha \in (0, 1]$.
- For each remaining edge uv, $u \in C_i, v \in C_j$, we sample the edge weight $w_{uv} = w_{vu}$ from a second distribution Ω_2 such that
$$\mathbf{E}[w_{uv}] = \mathbf{E}[w_{vu}] = \beta, \qquad 0 \leq w_{uv} \leq 1,$$
for fixed $\beta \in [0, \alpha)$ if $1 \leq i, j \leq k$ or $i = j = k + 1$, and $\mathbf{E}[w_{uv}] = \beta/2$ otherwise.

We should note that the assumption that the entries of $\boldsymbol{W}$ are bounded between 0 and 1 is made for simplicity in the statement and proof of our main result; analogous recovery guarantees hold if we assume that random variables sampled according to Ω_1 and Ω_2 are bounded and nonnegative with high probability. We say that such random matrices $\boldsymbol{W}$ are sampled from the *planted cluster model*. Note that if $\boldsymbol{W}$ is sampled from the planted cluster model, then weight is concentrated on within-cluster edges (in expectation). This provides a natural generalization of the stochastic block model. Indeed, the stochastic block model corresponds to the planted cluster model in the special case that Ω_1 and Ω_2 are Bernoulli distributions with probabilities of success p and q, respectively. Ames (2014) established the following theorem, ensuring recovery of the planted cliques $C_1, \ldots, C_k$ from the optimal solution of (5) under the planted cluster model (see Ames, 2014, Theorem 2.1).

Theorem 2 *Suppose that the vertex sets $C_1, \ldots, C_k$ define a k-disjoint-clique subgraph H^* of the n-node weighted complete graph $K_n(V, \boldsymbol{W})$ and let $C_{k+1} := V \setminus \left(\cup_{i=1}^k C_i\right)$. Let $r_i := |C_i|$ for all $i = 1, \ldots, k+1$ and let $\hat{r} = \min_{i=1,\ldots,k} r_i$. Let $\boldsymbol{W} \in \Sigma^n$ be a random symmetric matrix sampled from the planted cluster model according to distributions Ω_1 and Ω_2 with means α and β, respectively, satisfying*

$$\gamma = \gamma(\alpha, \beta, r) := \alpha - \beta > 0.$$

Let $\boldsymbol{X}^*$ be the feasible solution of (5) corresponding to $C_1, \ldots, C_k$ defined by (3). Then there exist scalars $c_1, c_2, c_3 > 0$ such that if

$$c_1\sqrt{n} + c_2\sqrt{kr_{k+1}} + c_3 r_{k+1} \leq \gamma \hat{r},$$

then $\boldsymbol{X}^*$ is the unique optimal solution of (5), and H^* is the unique maximum density k-disjoint-clique subgraph of K_n with probability tending exponentially to 1 as $\hat{r} \to \infty$.

In contrast to Theorem 1, the result of Theorem 2 implies that we can have perfect recovery if the graph contains a small number of nodes that shouldn't be assigned to any of the planted clusters. Each potential edge from each of these nodes to any other node is added independently to the graph with probability q, so that each node in C_{k+1} has roughly the same number of neighbours in each cluster block. This implies that such a node is not assigned to any of the planted clusters because it is weakly associated with all of the planted clusters. It is important to note that this edge assignment is performed randomly and not deterministically by an adversary attempting to obscure the cluster structure present in the graph. We present a new analysis that improves upon the recovery guarantee of Theorem 2 in two ways. First, the hypothesis of Theorem 2 assumes that between-cluster and within-cluster edge weights are i.i.d. We consider the more general heterogeneous case constructed as follows:

- For each $u \in C_i$, $v \in C_j$, we sample the edge weight $w_{uv} = w_{vu}$ from distribution Ω_{ij} with
$$\mathbf{E}[w_{uv}] = \mathbf{E}[w_{vu}] = \mu_{ij}, \qquad \mathrm{Var}[w_{uv}] = \mathrm{Var}[w_{vu}] = \sigma_{ij}^2, \qquad 0 \leq w_{uv} \leq 1.$$

This forces weights within the same block to be i.i.d., but weight may not be identically distributed in different blocks.

Second, the analysis leading to Theorem 2 assumes that the expectations of Ω_1, Ω_2 in the planted cluster model are fixed and that the variances are bounded by 1. We improve upon the recovery guarantee of Theorem 2 by considering the case where the parameters α and β depend on the number of nodes n in the graph. In particular, our recovery guarantees explicitly depend on the variances of the distributions Ω_{ij}, and their scaling with n, which will expand the set of graphs known to be clusterable by (5). We have the following theorem.

Theorem 3 *Suppose that the vertex sets $C_1, \ldots, C_k$ define a k-disjoint-clique subgraph K^* of the weighted complete graph $K_n = (V, \boldsymbol{W})$ on n vertices and let $C_{k+1} = V \setminus \left(\cup_{i=1}^k C_i\right)$. Let $r_i = |C_i|$ for all $i = 1, \ldots, k+1$ and let $\hat{r} = \min_{i=1,\ldots,k} r_i$. Let $\boldsymbol{W} \in \Sigma^n$ be a random symmetric matrix sampled from the heterogeneous planted cluster model according to distributions $\{\Omega_{ij}\}$*

with expected values $\mu_{ij} = \mu_{ij}(n)$ *and variances* $\sigma_{ij}^2 = \sigma_{ij}^2(n)$. *Let* $\tilde{\sigma} := \max_{q,s} \sigma_{qs}$ *and* $\hat{\sigma} := \max_q \sigma_{qq}$. *Let* $\boldsymbol{X}^*$ *be the feasible solution to* (5) *corresponding to* $C_1, \ldots, C_k$ *defined by* (3). *Let*

$$\gamma := \min_{\substack{q,s=1,2,\ldots,k \\ q \neq s}} \{\mu_{qq} - \mu_{qs}\}.$$

Then there exists scalar $c > 0$ *such that if*

$$\gamma \hat{r} \geq c \max\left\{\sqrt{\tilde{\sigma}^2 n},\ \sqrt{\tilde{\sigma}^2 \hat{r} \log n},\ \sqrt{\hat{\sigma}^2 k r_{k+1}},\ \sqrt{k r_{k+1} \log n / \hat{r}},\ \mu_{k+1,k+1} r_{k+1},\ \log n\right\}, \quad (6)$$

then $\boldsymbol{X}^*$ *is the unique optimal solution for* (5), *and* K^* *is the unique maximum density* k*-disjoint-clique subgraph of* K_n *with high probability.*

The *weak assortativity* condition (6) implies that we have perfect recovery provided that the gap between the cluster block expectation μ_{qq} and the largest between-cluster block expectation μ_{qs} is sufficiently large for all clusters C_q, $q = 1, \ldots, k$, relative to the minimum cluster size, number of unassigned nodes r_{k+1}, number of clusters, and edge weight variances. In the Bernoulli case, i.e., within-cluster and between-cluster edges are added independently with probabilities p and q, respectively, Theorem 3 and, in particular, (6) establish that we can recover the planted clusters provided that

$$\frac{(p-q)^2}{\tilde{\sigma}^2} = \frac{(p-q)^2}{\max\{p(1-p), q(1-q)\}} = \Omega\left(\frac{n}{\hat{r}^2}\right).$$

This result agrees with the Easy Regime for cluster recovery proposed by Chen and Xu (2014), where a polynomial-time algorithm exists for exact recovery of the planted clusters, in this case, the solution of the semidefinite relaxation (5). One distinct advantage of this result over similar recovery guarantees is that our model and phase transition are largely parameter free. For example, Amini and Levina (2018) present an analysis of three semidefinite relaxations that obtain nearly identical conditions on $\{\Omega_{ij}\}$ guaranteeing recovery but restrict their analysis to the case where the clusters are identical in size or otherwise known and when $\{\Omega_{ij}\}$ are Bernoulli distributions; we should note that Amini and Levina (2018) consider heterogeneous Bernoulli distributions where the within-cluster and between-cluster probabilities of adding an edge vary across clusters. Similarly, Chen and Xu (2014) and Jalali et al. (2015) give identical conditions for recovery (up to constants and logarithmic terms) in the Bernoulli case to those in Theorem 3 for semidefinite relaxations that require the sizes of the clusters to be used as input parameters (or all clusters to have identical size), neither of which are realistic assumptions in practice. In contrast, our approach achieves this recovery guarantee using *only the desired number of clusters as a parameter*. Further, our guarantee extends to the general weighted case where the vast

majority of existing recovery guarantees for stochastic block models are restricted to the Bernoulli case.

It is important to note that tighter recovery guarantees than those provided by Theorem 3 are known for specific problem settings. This is a natural consequence of the more general framework of our analysis. For example, Yan et al. (2017) studies a convex relaxation for cluster recovery in the Bernoulli (unweighted) case. The main theorem of this article establishes conditions for perfect recovery that allow larger clusters to have higher variance, although specialized for the unweighted case. Moreover, Yan et al. (2017) consider the use of a tuning parameter to allow recovery without knowledge of the number of clusters k. On the other hand, the results of Amini and Levina (2018); Jalali et al. (2015) also provide tighter recovery guarantees but require knowledge of cluster sizes. The key contribution of this work is the presentation of a recovery guarantee that extends to the weighted case without strict assumptions regarding input parameters, as well as the first-order method for solution of (1) discussed in detail in Section 4.

To further illustrate the consequences of Theorem 3, we consider several examples. In each, we assume that the graph is generated in the homogeneous setting where within-cluster weights are i.i.d. according to Ω_1 with mean α and variance σ_1^2, and between-cluster weights are i.i.d. according to Ω_2 with mean $\beta < \alpha$ and variance σ_2^2.

2.2.1. The Dense Case

When α, β are fixed, we obtain the same recovery guarantee as before, up to constants and logarithmic terms: we have exact recovery w.h.p. if $\hat{r} \geq \tilde{c}_1 \sqrt{n}$ and $\hat{r} \geq \tilde{c}_2 r_{k+1}$ for some constants $\tilde{c}_1$ and $\tilde{c}_2$ depending on Ω_1 and Ω_2. Indeed, each of the pointwise maximums in the first three terms of (6) is bounded above by $O(\sqrt{n})$ since $\tilde{r} \leq n$, and $k r_{k+1} = O(n)$ if $r_{k+1} = O(\hat{r})$.

2.2.2. The Sparse Case

On the other hand, if noise in the form of between-cluster edge-weight is small, then we should expect to be able recover much smaller clusters. For example, suppose that Ω_2 is the Bernoulli distribution with probability of adding an edge q and that Ω_1 is the Bernoulli distribution with probability of adding an edge $p = 1$ (the assumption that $p = 1$ is for the sake of simplicity in this example and we can expect analogous recovery guarantees for any p tending slowly enough to 0). Assume further that $q(1-q) \leq \log n / n$. Finally, again for simplicity, assume that we have k equally sized clusters of size $\hat{r} = n/k$ and $(r_{k+1} = 0)$. In

this case, (6) holds if

$$\gamma\hat{r} \geq c\log n = c\max\left\{\sqrt{\log n},\ \sqrt{\frac{\hat{r}\log^2 n}{n}},\ \log n\right\} \geq c\max\left\{\sqrt{\tilde{\sigma}^2 n},\ \sqrt{\tilde{\sigma}^2 r\log n},\ \log n\right\},$$

since $\tilde{\sigma}^2 = \max\{p(1-p), q(1-q)\} = q(1-q) \leq \log n/n$ and the terms involving r_{k+1} and $1-p$ are equal to zero. This implies that we have exact recovery of the planted clusters w.h.p. provided $\hat{r} = \Omega(\log n)$. This exceeds the state of the art recovery bound of $\hat{r} = \Omega(\sqrt{\log n})$ established in Jalali et al. (2015) by a factor of $\sqrt{\log n}$. However, the convex relaxation proposed by Jalali et al. (2015) requires knowledge of $\sum_{i=1}^{k} r_i^2$, which is often an unrealistic expectation in practice; in contrast, our approach only requires knowledge of the number of clusters k present in the data. Further, the requirement $\hat{r} = \Omega(\log n)$ is enforced by the gap inequality (6), which itself is a consequence of the use of the Bernstein inequality to establish certain dual variables are nonnegative in the proof of Theorem 3 (see Section 3.1 for more details). It may be possible to improve this bound to $\hat{r} = \Omega(\sqrt{\log n})$ with improved concentration inequalities but it is unclear what form these improvements may take.

2.2.3. The Planted Clique and Sparsest Subgraph

In the special case when $k = 1$ and Ω_1 and Ω_2 are Bernoulli distributions, the planted cluster model specializes to the planted clique model considered in Ames and Vavasis (2011) and Ames (2015). In this case, (6) suggests that we can recover a planted clique (in the dense case) of size $r_1 = \Omega\left(\max\left\{\sqrt{n}, r_2\right\}\right) = \Omega\left(\max\left\{\sqrt{n}, n - r_1\right\}\right)$. This recovery guarantee is far more conservative than those provided by Ames and Vavasis (2011) and Ames (2015), among others, which establish that a planted clique of size $\Omega(\sqrt{n})$ can be recovered from the optimal solution of a particular nuclear norm relaxation of the maximum clique problem.

Unfortunately, it appears that this lower bound restricting the size of a recoverable planted clique to a constant multiple of the number of nonclique vertices is tight. For example, let p and q be the probabilities of adding an edge given by Ω_1 and Ω_2. Then the expected value of the proposed solution $\boldsymbol{X}^*$ in (5) is equal to

$$\mathbf{E}[\mathrm{Tr}(\boldsymbol{W}\boldsymbol{X}^*)] = \frac{1}{\hat{r}}\sum_{i\in C_1}\sum_{j\in C_1}\mathbf{E}[w_{ij}] = p\hat{r}.$$

On the other hand, the solution $\frac{1}{n}\boldsymbol{e}\boldsymbol{e}^T$ is also feasible for (5) with expected objective value

$$\mathbf{E}\left[\frac{1}{n}\mathrm{Tr}(\boldsymbol{W}\boldsymbol{e}\boldsymbol{e}^T)\right] \geq \hat{c}qn,$$

for some constant $\hat{c}$. This implies that the proposed solution is suboptimal if $p\hat{r} < \hat{c}qn$, which holds unless $\hat{r} \geq \hat{c}(q/p)n$. We will see that the realized values of these sums are concentrated

near their expectations and thus we cannot reasonably expect to recover planted clusters with unassigned nodes significantly outnumbering the smallest cluster. This implies that we cannot recover planted cliques of size $\omega(n)$ by maximizing density of a complete subgraph because the planted clique is not the index set of the densest such graph; in this case, the entire graph is a denser complete subgraph, as measured by average vertex degree, in expectation.

3. Derivation of the Recovery Guarantee

In this section, we show that if the hypothesis of Theorem 3 is satisfied then the solution $\boldsymbol{X}^*$ constructed according to (3) is optimal for (5) and the corresponding k-disjoint-clique subgraph has maximum density. In particular, we will show that $\boldsymbol{X}^*$ satisfies the following sufficient condition for optimality of a feasible solution of (5) (see Ames, 2014, Theorem 4.1).

Theorem 4 *Let $\boldsymbol{X}$ be feasible for (5) and suppose that there exist some $\tau \in \mathbf{R}$, $\boldsymbol{\lambda} \in \mathbf{R}^n_+$, $\boldsymbol{\Xi} \in \mathbf{R}^{n\times n}_+$ and $\boldsymbol{S} \in \Sigma^n_+$ such that*

$$-\boldsymbol{W} + \boldsymbol{\lambda}\boldsymbol{e}^T + \boldsymbol{e}\boldsymbol{\lambda}^T - \boldsymbol{\Xi} + \tau\boldsymbol{I} = \boldsymbol{S} \tag{7}$$

$$\boldsymbol{\lambda}^T(\boldsymbol{X}\boldsymbol{e} - \boldsymbol{e}) = 0 \tag{8}$$

$$\mathrm{Tr}(\boldsymbol{X}\boldsymbol{\Xi}) = 0 \tag{9}$$

$$\mathrm{Tr}(\boldsymbol{X}\boldsymbol{S}) = 0. \tag{10}$$

Then $\boldsymbol{X}$ is optimal for (5).

Theorem 4 is a restriction of the Karush-Kuhn-Tucker optimality conditions to the semidefinite program (5) (see, for example, Boyd and Vandenberghe, 2004, Section 5.5.3). The goal of this section is to establish that we can construct dual variables $\tau \in \mathbf{R}$, $\boldsymbol{\lambda} \in \mathbf{R}^n_+$, $\boldsymbol{\Xi} \in \mathbf{R}^{n\times n}_+$ and $\boldsymbol{S} \in \Sigma^n_+$ which satisfy the hypothesis of Theorem 4 with high probability if the weight matrix $\boldsymbol{W}$ is sampled from a distribution of clusterable block models. To motivate our proposed choice of dual variables, we note that the complementary slackness condition $\mathrm{Tr}(\boldsymbol{X}\boldsymbol{S}) = 0$ holds if and only if $\boldsymbol{X}\boldsymbol{S} = \boldsymbol{0}$ under the assumption that both $\boldsymbol{X}$ and $\boldsymbol{S}$ are positive semidefinite. Therefore, the block structure of $\boldsymbol{X}$ implies that each block of $\boldsymbol{S}$ corresponding to a cluster block in $\boldsymbol{W}$ must sum to zero.

Before we continue with the construction of our dual variables, let us first remind ourselves of the notation of Theorem 3. Let K^* be a k-disjoint-clique subgraph of K_n with vertex set composed of the disjoint cliques $C_1, \ldots, C_k$ of sizes $r_1, \ldots, r_k$ and let $\boldsymbol{X}^*$ be the corresponding

feasible solution of (5) defined by (3). Let $C_{k+1} := V \setminus (\cup_{i=1}^{k} C_i)$ and $r_{k+1} := n - \sum_{i=1}^{k} r_i$ be the size of C_{k+1}. Moreover, let $\hat{r} := \min_{i=1,\ldots,k} r_i$ and $\tilde{r} := \max_{i=1,\ldots,k} r_i$ be the size of the smallest and largest clusters, respectively. Let $\boldsymbol{W} \in \Sigma^n$ be a random symmetric matrix sampled from the planted cluster model with planted clusters $C_1, \ldots, C_k$ and remaining nodes C_{k+1} according to the distributions $\{\Omega_{ij}\}$ with means $\{\mu_{ij}\}$ and variances $\{\sigma_{ij}^2\}$.

We now propose a choice of dual variables satisfying the complementary slackness condition $\boldsymbol{X}\boldsymbol{S} = \boldsymbol{0}$. Restricting this condition to the blocks $\boldsymbol{X}_{C_q,C_q}$ and $\boldsymbol{S}_{C_q,C_q}$ of $\boldsymbol{X}$ and $\boldsymbol{S}$ with rows and columns indexed by C_q, $q \in \{1, 2, \ldots, k\}$, we see that $\boldsymbol{X}^*\boldsymbol{S} = \boldsymbol{0}$ holds if and only if

$$\boldsymbol{0} = \boldsymbol{S}_{C_q,C_q}\boldsymbol{e} = \tau\boldsymbol{e} + r_q\boldsymbol{\lambda}_{C_q} + (\boldsymbol{\lambda}_{C_q}^T\boldsymbol{e})\boldsymbol{e} - \boldsymbol{W}_{C_q,C_q}\boldsymbol{e},$$

by the block structure of $\boldsymbol{X}^*$; note that $\boldsymbol{\Xi}_{C_q,C_q} = \boldsymbol{0}$ is chosen to satisfy the complementary slackness condition (9). Solving this linear system for $\boldsymbol{\lambda}_{C_q}$ using the Sherman-Morrison-Woodbury Formula (Golub and Van Loan, 2013, Equation (2.1.4)) gives

$$\boldsymbol{\lambda}_{C_q} = \frac{1}{r_q}\left(\boldsymbol{W}_{C_q,C_q}\boldsymbol{e} - \frac{1}{2}\left(\tau + \frac{\boldsymbol{e}^T\boldsymbol{W}_{C_q,C_q}\boldsymbol{e}}{r_q}\right)\boldsymbol{e}\right). \tag{11}$$

On the other hand, we choose $\boldsymbol{\lambda}_{C_{k+1}} = \boldsymbol{0}$ to satisfy the complementary slackness condition (8). Next, we use this choice of $\boldsymbol{\lambda}$ to construct the remaining dual variables.

Fix $q, s \in \{1, 2, \ldots, k+1\}$ such that $q \neq s$. We will choose $\boldsymbol{\Xi}_{C_q,C_s}$ so that $\boldsymbol{S}_{C_q,C_s}\boldsymbol{e} = \boldsymbol{0}$ and $\boldsymbol{S}_{C_s,C_q}\boldsymbol{e} = \boldsymbol{0}$. In particular, we choose

$$\boldsymbol{\Xi}_{C_q,C_s} = \left(\tfrac{1-\delta_{q,k+1}}{2}\left(\mu_{qq} - \tfrac{\tau}{r_q}\right) + \tfrac{1-\delta_{s,k+1}}{2}\left(\mu_{ss} - \tfrac{\tau}{r_s}\right) - \mu_{qs}\right)\boldsymbol{e}\boldsymbol{e}^T + \boldsymbol{y}^{q,s}\boldsymbol{e}^T + \boldsymbol{e}(\boldsymbol{z}^{q,s})^T, \tag{12}$$

where the vectors $\boldsymbol{y}^{q,s}$ and $\boldsymbol{z}^{q,s}$ are unknown vectors parametrizing the entries of $\boldsymbol{\Xi}_{C_q,C_s}$; here $\delta_{i,j}$ is the Kronecker delta function defined by $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise. That is, we choose $\boldsymbol{\Xi}_{C_q,C_s}$ to be the expected value of $\boldsymbol{\lambda}_{C_q}\boldsymbol{e}^T + \boldsymbol{e}\boldsymbol{\lambda}_{C_s}^T - \boldsymbol{W}_{C_q,C_s}$ plus the parametrizing term $\boldsymbol{y}^{q,s}\boldsymbol{e}^T + \boldsymbol{e}(\boldsymbol{z}^{q,s})^T$; the vectors $\boldsymbol{y}^{q,s}$ and $\boldsymbol{z}^{q,s}$ are chosen to be solutions of the systems of linear equations given by the complementary slackness conditions $\boldsymbol{S}_{C_q,C_s}\boldsymbol{e} = \boldsymbol{0}$ and $\boldsymbol{S}_{C_s,C_q}\boldsymbol{e} = \boldsymbol{0}$. It is reasonably straight-forward to show that we may choose

$$\boldsymbol{y}^{q,s} = \frac{1}{r_s}\left(\boldsymbol{b}_{q,s} - \frac{\boldsymbol{b}_{q,s}^T\boldsymbol{e}}{r_q + r_s}\boldsymbol{e}\right) \qquad \boldsymbol{z}^{q,s} = \frac{1}{r_q}\left(\boldsymbol{b}_{s,q} - \frac{\boldsymbol{b}_{s,q}^T\boldsymbol{e}}{r_q + r_s}\boldsymbol{e}\right), \tag{13}$$

where

$$\boldsymbol{b}_{q,s} = \left(\boldsymbol{\lambda}_{C_q}\boldsymbol{e}^T + \boldsymbol{e}\boldsymbol{\lambda}_{C_s}^T - \boldsymbol{W}_{C_q,C_s} - \mathbf{E}\left[\boldsymbol{\lambda}_{C_q}\boldsymbol{e}^T + \boldsymbol{e}\boldsymbol{\lambda}_{C_s}^T - \boldsymbol{W}_{C_q,C_s}\right]\right)\boldsymbol{e}. \tag{14}$$

Indeed, we must choose $\boldsymbol{y} = \boldsymbol{y}^{q,s}$ and $\boldsymbol{z} = \boldsymbol{z}^{q,s}$ to be solutions of the system

$$\begin{pmatrix} r_s\boldsymbol{I} + \boldsymbol{e}\boldsymbol{e}^T & \boldsymbol{0} \\ \boldsymbol{0} & r_q\boldsymbol{I} + \boldsymbol{e}\boldsymbol{e}^T \end{pmatrix}\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{z} \end{pmatrix} = \begin{pmatrix} \boldsymbol{b}^{q,s} \\ \boldsymbol{b}_{s,q} \end{pmatrix}, \tag{15}$$

to ensure that the complementary slackness conditions are satisfied. Note that taking the inner product of each side of (15) with the vector $(\boldsymbol{e}; -\boldsymbol{e})$ yields

$$(r_q + r_s)(\boldsymbol{e}^T\boldsymbol{y} - \boldsymbol{e}^T\boldsymbol{z}) = \boldsymbol{e}^T\boldsymbol{b}^{q,s} - \boldsymbol{e}^T\boldsymbol{b}^{s,q} = 0,$$

by the symmetry of $\boldsymbol{W}$. This establishes that the solution $(\boldsymbol{y}; \boldsymbol{z})$ of (15) is also a solution of the (singular) system of equations,

$$\begin{pmatrix} r_s\boldsymbol{I} & \boldsymbol{e}\boldsymbol{e}^T \\ \boldsymbol{e}\boldsymbol{e}^T & r_q\boldsymbol{I} \end{pmatrix}\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{z} \end{pmatrix} = \begin{pmatrix} \boldsymbol{b}^{q,s} \\ \boldsymbol{b}_{s,q} \end{pmatrix},$$

imposed by the complementary slackness conditions $\boldsymbol{S}_{C_q,C_s}\boldsymbol{e} = \boldsymbol{0}$ and $\boldsymbol{S}_{C_s,C_q} = \boldsymbol{0}$. Solving (15) for $\boldsymbol{y}$ and $\boldsymbol{z}$ using the Sherman-Morrison-Woodbury Formula yields the formula for $\boldsymbol{y}$ and $\boldsymbol{z}$ given by (13). We set the remaining block $\boldsymbol{\Xi}_{C_{k+1},C_{k+1}} = \boldsymbol{0}$. Ames (2014, Section 4.2) provides further details.

Finally, we choose

$$\tau = \min_{\substack{q,s=1,\ldots,k \\ q\neq s}} \{\mu_{qq} - \mu_{qs}\}\, \epsilon\hat{r} =: \gamma\epsilon\hat{r}, \tag{16}$$

where $\epsilon > 0$ is a parameter to be chosen later. In particular, the analysis provided in Sections 3.1, 3.2, and 3.3 establishes that a suitable choice of ϵ exists if the hypothesis of Theorem 3 is satisfied.

The entries of $\boldsymbol{S}$ are chosen according to the stationarity condition (7), but we will also define an auxiliary variable $\tilde{\boldsymbol{S}} \in \Sigma^n$ as the following $(k+1) \times (k+1)$ block matrix:

$$\tilde{\boldsymbol{S}}_{C_q,C_s} = \begin{cases} \mu_{q,s}\boldsymbol{e}\boldsymbol{e}^T - \boldsymbol{W}_{C_q,C_s}, & \text{if } q, s \in \{1, \ldots, k\}, \\ \mu_{q,k+1}\boldsymbol{e}\boldsymbol{e}^T - \boldsymbol{W}_{C_q,C_{k+1}} + (\boldsymbol{\lambda}_{C_q} - \mathbf{E}[\boldsymbol{\lambda}_{C_q}])\boldsymbol{e}^T, & \text{if } s = k+1, \\ \mu_{k+1,s}\boldsymbol{e}\boldsymbol{e}^T - \boldsymbol{W}_{C_{k+1},C_s} + \boldsymbol{e}(\boldsymbol{\lambda}_{C_s} - \mathbf{E}[\boldsymbol{\lambda}_{C_s}])^T, & \text{if } q = k+1. \end{cases} \tag{17}$$

We next provide the following theorem, first stated by Ames (2014, Theorem 4.2), which characterizes when the proposed dual variables satisfy the hypothesis of Theorem 4.

Theorem 5 *Suppose that the vertex sets $C_1, \ldots, C_k$ define a k-disjoint-clique subgraph K^* of the weighted complete graph $K_n = (V, \boldsymbol{W})$, where $\boldsymbol{W} \in \Sigma^n$ is a random symmetric matrix sampled from the planted cluster model according to the distributions $\{\Omega_{ij}\}$ with means $\{\mu_{ij}\}$ and variances $\{\sigma_{ij}^2\}$. Let $r_1, \ldots, r_{k+1}$, and $\hat{r}$ be defined as in Theorem 3. Let $\boldsymbol{X}^*$ be the feasible solution for (5) corresponding to $C_1, \ldots, C_k$ defined by (3). Let $\tau \in \mathbf{R}$, $\boldsymbol{\lambda} \in \mathbf{R}^n$, and $\boldsymbol{\Xi} \in \mathbf{R}^{n\times n}$ be chosen according to (16), (11), and (12), respectively, and let $\tilde{\boldsymbol{S}}$ be chosen according to (17). Suppose that the entries of $\boldsymbol{\lambda}$ and $\boldsymbol{\Xi}$ are nonnegative. Then $\boldsymbol{X}^*$ is optimal for (5), and K^* is the maximum density k-disjoint-clique subgraph of K_n corresponding to $\boldsymbol{W}$, if*

$$\|\tilde{\boldsymbol{S}}\| \leq \epsilon\gamma\hat{r}. \tag{18}$$

Moreover, if (18) *is satisfied and*

$$r_s \boldsymbol{e}^T \boldsymbol{W}_{C_q,C_q} \boldsymbol{e} > r_q \boldsymbol{e}^T \boldsymbol{W}_{C_q,C_s} \boldsymbol{e}, \tag{19}$$

for all $q, s \in \{1, \ldots, k\}$ *such that* $q \neq s$*, then* $\boldsymbol{X}^*$ *is the unique optimal solution of* (5) *and* K^* *is the unique maximum density* k*-disjoint-clique subgraph of* K_n*.*

The proof of Theorem 5 is nearly identical to that by Ames (2014, Theorem 4.2), and is omitted. Theorem 5 provides a clear roadmap for the remainder of the proof; if we can show that if $\boldsymbol{W}$ is sampled from the planted cluster model satisfying (6) then $\boldsymbol{\lambda}$ and $\boldsymbol{\Xi}$ are nonnegative and $\|\tilde{\boldsymbol{S}}\| \leq \epsilon\gamma\hat{r}$ with high probability, then we will have established that we can recover the underlying block structure with high probability in this case. We establish the necessary bounds on $\boldsymbol{\lambda}$, $\boldsymbol{\Xi}$, and $\|\tilde{\boldsymbol{S}}\|$ in the following sections.

3.1. Nonnegativity of $\boldsymbol{\lambda}$ and $\boldsymbol{\Xi}$

We first establish that the entries of $\boldsymbol{\Xi}$, as constructed according to (12), are nonnegative with high probability. To do so, we will make repeated use of the following specialization of the Bernstein inequality which provides a bound on the tail of a sum of bounded independent random variables; see Boucheron et al. (2013, Section 2.8), for more details regarding the Bernstein inequality.

Theorem 6 *Let* $x_1, \ldots, x_m$ *be independent identically distributed (i.i.d.) variables with mean* μ *and variance* σ^2*. Let* $S = x_1 + \cdots + x_m$*. Then*

$$Pr\left(|S - \mu m| > 6\max\left\{\sqrt{\sigma^2 m \log T}, \log T\right\}\right) \leq 2T^{-6}, \tag{20}$$

for all $T > 1$*.*

The following bound on the parametrizing vectors $\boldsymbol{y}^{q,s}$ and $\boldsymbol{z}^{q,s}$ in the choice of the (C_q, C_s) block of $\boldsymbol{\Xi}$ defined by (12) is an immediate consequence of Theorem 6.

Lemma 7 *There exists constant* $c > 0$ *such that*

$$\|\boldsymbol{y}^{q,s}\|_\infty + \|\boldsymbol{z}^{q,s}\|_\infty \leq c\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\},$$

w.h.p., where $\tilde{\sigma} := \max\{\sigma_{ij} : i, j = 1, 2, \ldots, k+1\}$*, for all* $q, s \in \{1, \ldots, k+1\}$ *such that* $q \neq s$*.*

For $q, s \in \{1, \ldots, k+1\}$ such that $q \neq s$, we define $\boldsymbol{y}^{q,s}$ and $\boldsymbol{z}^{q,s}$ as in (13). To bound the absolute values of the entries of $\boldsymbol{y}^{q,s}$ and $\boldsymbol{z}^{q,s}$, we must estimate the sums $\boldsymbol{e}^T \boldsymbol{W}_{C_q,C_q} \boldsymbol{e}$, $\boldsymbol{e}^T \boldsymbol{W}_{C_s,C_s} \boldsymbol{e}$ and $\boldsymbol{e}^T \boldsymbol{W}_{C_q,C_s} \boldsymbol{e}$; applying Theorem 6 to bound the tails of these sums yields Lemma 7. See Appendix A for the full argument.

We have the following bound on the entries of $\boldsymbol{\Xi}$ as an immediate consequence of Lemma 7.

Proposition 8 *Suppose that* $\{\mu_{ij}\}$ *satisfy* (6)*. Then there exists constant* $c > 0$ *such that each entry of* $\boldsymbol{\Xi}$ *is nonnegative w.h.p. if* ϵ *satisfies*

$$0 < \epsilon \leq 1 - \frac{c}{\gamma}\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\}. \tag{21}$$

Proof Fix $q, s \in \{1, \ldots, k\}$ such that $q \neq s$. By construction, we have

$$\begin{aligned}\boldsymbol{\Xi}_{C_q,C_s} &= \mathbf{E}\left[\boldsymbol{\lambda}_{C_q}\boldsymbol{e}^T + \boldsymbol{e}\boldsymbol{\lambda}_{C_s}^T - \boldsymbol{W}_{C_q,C_s}\right] + \boldsymbol{y}^{q,s}\boldsymbol{e}^T + \boldsymbol{e}\left(\boldsymbol{z}^{q,s}\right)^T \\ &= \left(\frac{1}{2}\left(\mu_{qq} - \frac{\tau}{r_q}\right) + \frac{1}{2}\left(\mu_{ss} - \frac{\tau}{r_s}\right) - \mu_{qs}\right)\boldsymbol{e}\boldsymbol{e}^T + \boldsymbol{y}^{q,s}\boldsymbol{e}^T + \boldsymbol{e}\left(\boldsymbol{z}^{q,s}\right)^T.\end{aligned}$$

Using (16) and Lemma 7, we see that

$$\begin{aligned}\Xi_{ij} &\geq \frac{1}{2}\left(\mu_{qq} - \gamma\epsilon\right) + \frac{1}{2}\left(\mu_{ss} - \gamma\epsilon\right) - \mu_{sq} - \|\boldsymbol{y}^{q,s}\|_\infty - \|\boldsymbol{z}^{q,s}\|_\infty \\ &\geq (1-\epsilon)\gamma - c\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\},\end{aligned}$$

for all $i \in C_q$, $j \in C_s$ w.h.p., where c is the constant appearing in Lemma 7. Note that the right-hand side of this inequality is nonnegative if and only if

$$\epsilon \leq 1 - \frac{c}{\gamma}\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\}.$$

The argument for the case when one of q or s is equal to $k+1$ follows analogously. Applying the union bound over all blocks of $\boldsymbol{\Xi}$ shows that each entry of $\boldsymbol{\Xi}$ is nonnegative w.h.p. if ϵ satisfies (21). ■

We have an analogous result ensuring that the entries of $\boldsymbol{\lambda}$ are nonnegative with high probability; we present the proof of this result in Appendix B.

Proposition 9 *Suppose $\{\mu_{ij}\}$ satisfy (6). Then there exists constant $c' > 0$ such that each entry of $\boldsymbol{\lambda}$ is nonnegative w.h.p. if ϵ satisfies*

$$0 < \epsilon \leq \frac{1}{2\gamma}\left(\mu_{qq} - c' \max\left\{\sqrt{\frac{\sigma_{qq}^2 \log n}{r_q}}, \frac{\log n}{r_q}\right\}\right), \tag{22}$$

for all $q \in \{1, \ldots, k\}$.

We conclude this section with a result ensuring that the uniqueness condition (19) of Theorem 5 is satisfied for all $q, s \in \{1, \ldots, k\}$ such that $q \neq s$; we provide a proof in Appendix C.

Proposition 10 *Suppose that*

$$\gamma \geq 12 \max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}^2}}, \frac{\log n}{\hat{r}^2}\right\}. \tag{23}$$

Then $r_s \boldsymbol{e}^T \boldsymbol{W}_{C_q, C_q} \boldsymbol{e} > r_q \boldsymbol{e}^T \boldsymbol{W}_{C_q, C_s} \boldsymbol{e}$ for all $q, s \in \{1, \ldots, k\}$ such that $q \neq s$ with high probability.

3.2. A Bound on $\tilde{\boldsymbol{S}}$

It remains to establish the following bound on the spectral norm of the matrix $\tilde{\boldsymbol{S}}$.

Proposition 11 *There exists scalars $C, C' > 0$ such that*

$$\|\tilde{\boldsymbol{S}}\| \leq C \max\left\{\tilde{\sigma}\sqrt{n}, \sqrt{\log n}\right\} + C'\left(\max\left\{\hat{\sigma}^2, \frac{\log n}{\hat{r}}\right\} k r_{k+1}\right)^{1/2} + \mu_{k+1,k+1} r_{k+1}, \tag{24}$$

where $\hat{\sigma}^2 = \max_{q=1,\ldots,k}\left\{\sigma_{qq}^2\right\}$, with high probability.

The proof of Proposition 11 follows the same structure as that of Ames (2014, Lemma 4.5). In particular, we decompose $\tilde{\boldsymbol{S}}$ as $\tilde{\boldsymbol{S}} = \tilde{\boldsymbol{S}}_1 + \tilde{\boldsymbol{S}}_2 + \tilde{\boldsymbol{S}}_3$, where

$$\tilde{\boldsymbol{S}}_1 = \mathbf{E}[\boldsymbol{W}] - \boldsymbol{W}, \tag{25}$$

$$[\tilde{\boldsymbol{S}}_2]_{C_q, C_s} = \begin{cases} \left(\boldsymbol{\lambda}_{C_q} - \mathbf{E}\left[\boldsymbol{\lambda}_{C_q}\right]\right)\boldsymbol{e}^T, & \text{if } s = k+1, \\ \boldsymbol{e}\left(\boldsymbol{\lambda}_{C_s} - \mathbf{E}\left[\boldsymbol{\lambda}_{C_s}\right]\right)^T, & \text{if } q = k+1, \\ \mathbf{0}, & \text{otherwise,} \end{cases} \tag{26}$$

$$[\tilde{\boldsymbol{S}}_3]_{C_q, C_s} = \begin{cases} -\mu_{k+1,k+1}\boldsymbol{e}\boldsymbol{e}^T, & \text{if } q = s = k+1, \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{27}$$

Note that $\|\tilde{\boldsymbol{S}}_3\| = \mu_{k+1,k+1}\|\boldsymbol{e}\boldsymbol{e}^T\| = \mu_{k+1,k+1} r_{k+1}$. The following lemmas provide the necessary bounds on $\|\tilde{\boldsymbol{S}}_1\|$ and $\|\tilde{\boldsymbol{S}}_2\|$.

Lemma 12 *Suppose that $\tilde{\boldsymbol{S}}_1$ is constructed according to (25) for some $\boldsymbol{W} \in \Sigma^n$ sampled from the heterogeneous planted cluster model. Then there exists constant $C > 0$ such that*

$$\|\tilde{\boldsymbol{S}}_1\| \leq C \max\left\{\tilde{\sigma}\sqrt{n}, \sqrt{\log n}\right\}, \tag{28}$$

with high probability.

Lemma 13 *Suppose that $\tilde{\boldsymbol{S}}_2$ is constructed according to (26) for some $\boldsymbol{W} \in \Sigma^n$ sampled from the heterogeneous planted cluster model. Then there exists constant $C' > 0$ such that*

$$\|\tilde{\boldsymbol{S}}_2\| \leq C'\left(\max\left\{\hat{\sigma}^2, \frac{\log n}{\hat{r}}\right\} k r_{k+1}\right)^{1/2},$$

with high probability, where $\hat{\sigma} := \max_{q=1,\ldots,k} \sigma_{qq}$.

We delay the proof of Lemmas 12 and 13 until Appendix D and Appendix E, respectively. Combining the three bounds on $\|\tilde{\boldsymbol{S}}_1\|$, $\|\tilde{\boldsymbol{S}}_2\|$, and $\|\tilde{\boldsymbol{S}}_3\|$ and applying the triangle inequality one last time shows that (24) holds with high probability.

3.3. The Conclusion of the Proof

According to Theorem 5, it suffices to prove that $\|\tilde{\boldsymbol{S}}\| \leq \epsilon\gamma\hat{r}$ is satisfied with high probability in order to prove Theorem 3. According to Proposition 11, if

$$\gamma\epsilon\hat{r} \geq C \max\left\{\tilde{\sigma}\sqrt{n}, \sqrt{\log n}\right\} + C'\left(\max\left\{\hat{\sigma}^2, \tfrac{\log n}{\hat{r}}\right\} k r_{k+1}\right)^{1/2} + \mu_{k+1,k+1} r_{k+1}, \tag{29}$$

then $\|\tilde{\boldsymbol{S}}\| \leq \epsilon\gamma\hat{r}$ holds with high probability. Hence, we have three conditions, (21), (22) and (29), on $\epsilon > 0$ that need to be satisfied simultaneously; choosing any $\epsilon > 0$ satisfying all three establishes the desired recovery guarantee. We see that (21) and (29) can be simultaneously fulfilled if

$$1 - \tfrac{c}{\gamma}\max\left\{\sqrt{\tfrac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \tfrac{\log n}{\hat{r}}\right\} \geq \tfrac{1}{\gamma\hat{r}}\left(C \max\left\{\tilde{\sigma}\sqrt{n}, \sqrt{\log n}\right\} + C'\left(\max\left\{\hat{\sigma}^2, \tfrac{\log n}{\hat{r}}\right\} k r_{k+1}\right)^{1/2} + \mu_{k+1,k+1} r_{k+1}\right),$$

which holds if and only if

$$\hat{r}\left(\gamma - c \max\left\{\sqrt{\tfrac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \tfrac{\log n}{\hat{r}}\right\}\right) \geq C \max\left\{\tilde{\sigma}\sqrt{n}, \sqrt{\log n}\right\} + C'\left(\max\left\{\hat{\sigma}^2, \tfrac{\log n}{\hat{r}}\right\} k r_{k+1}\right)^{1/2} + \mu_{k+1,k+1} r_{k+1}. \tag{30}$$

Next, we see that (29) and (22) are simultaneously fulfilled if

$$\tfrac{1}{2\gamma}\left(\mu_{qq} - c' \max\left\{\sqrt{\tfrac{\sigma_{qq}^2 \log n}{\hat{r}}}, \tfrac{\log n}{\hat{r}}\right\}\right) \geq \tfrac{1}{\gamma\hat{r}}\left(C \max\left\{\tilde{\sigma}\sqrt{n}, \sqrt{\log n}\right\} + C'\left(\max\left\{\hat{\sigma}^2, \tfrac{\log n}{\hat{r}}\right\} k r_{k+1}\right)^{1/2} + \mu_{k+1,k+1} r_{k+1}\right),$$

which holds if and only if

$$\hat{r}\left(\mu_{qq} - c' \max\left\{\sqrt{\tfrac{\sigma_{qq}^2 \log n}{r_{qq}}}, \tfrac{\log n}{r_{qq}}\right\}\right) \geq 2\left(C \max\left\{\tilde{\sigma}\sqrt{n}, \sqrt{\log n}\right\} + C'\left(\max\left\{\hat{\sigma}^2, \tfrac{\log n}{\hat{r}}\right\} k r_{k+1}\right)^{1/2} + \mu_{k+1,k+1} r_{k+1}\right). \tag{31}$$

Finally, suppose that we choose the parameter $c_4 > \max\{c, c', 12\}$ so that gap condition (23) is satisfied and

$$\gamma > \max\{c, c', 12\} \max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\}.$$

Then there exist constants c_1, c_2, c_3, depending on c_4, such that (30) and (31) are satisfied, i.e., there exists ϵ satisfying (21), (22) and (29) simultaneously, if

$$\gamma\hat{r} \geq c_1 \max\left\{\tilde{\sigma}\sqrt{n}, \sqrt{\log n}\right\} + c_2 \left(\max\left\{\sigma_1^2, \frac{\log n}{\hat{r}}\right\} k r_{k+1}\right)^{1/2} + c_3 \mu_{k+1,k+1} r_{k+1}.$$

This concludes the proof of Theorem 3.

4. Numerical Methods and Simulations

We conclude with a discussion of an algorithm for solution of (5) based on the alternating direction method of multipliers (ADMM), and provide the results of a series of experiments that empirically verify the phase transitions predicted in Section 2.2. In particular, we randomly sample graphs $G = (V, \boldsymbol{W})$ from the planted cluster model and compare the optimal solution of (5) with the planted partition.

4.1. Alternating Direction Method of Multipliers for the Densest k-Disjoint Clique Problem

We solve (5) iteratively using the algorithm proposed by Ames (2014). Specifically, we split the decision variable $\boldsymbol{X}$ to obtain the equivalent formulation

$$\max\left\{\operatorname{Tr}(\boldsymbol{W}\boldsymbol{Y}) : \boldsymbol{X} - \boldsymbol{Y} = \boldsymbol{0}, \boldsymbol{X}\boldsymbol{e} \leq \boldsymbol{e}, \boldsymbol{X} \geq \boldsymbol{0},\ \operatorname{Tr}\boldsymbol{Y} = k, \boldsymbol{Y} \in \Sigma_+^V\right\}.$$

We then apply an approximate dual ascent scheme to maximize the augmented Lagrangian

$$L_\rho(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}) = \operatorname{Tr}(\boldsymbol{W}\boldsymbol{Y}) - \operatorname{Tr}(\boldsymbol{Z}(\boldsymbol{X} - \boldsymbol{Y})) + \frac{\rho}{2}\|\boldsymbol{X} - \boldsymbol{Y}\|_F^2,$$

where $\rho > 0$ is a penalty parameter for violation of the linear equality constraint $\boldsymbol{X} - \boldsymbol{Y} = \boldsymbol{0}$. In particular, we minimize L_ρ with respect to $\boldsymbol{Y}$ and $\boldsymbol{X}$ successively, and then update $\boldsymbol{Z} = \boldsymbol{Z} - \rho(\boldsymbol{X} - \boldsymbol{Y})$ using approximate gradient ascent.

We update $\boldsymbol{Y}$ as the minimizer of the subproblem

$$\boldsymbol{Y}^{t+1} = \operatorname*{arg\,min}_{\boldsymbol{Y} \in \Sigma_+^n} \left\{\left\|\boldsymbol{Y} - \left(\boldsymbol{X}^t - \frac{\boldsymbol{W} + \boldsymbol{Z}^t}{\rho}\right)\right\|_F^2 : \operatorname{Tr}\boldsymbol{Y} = k\right\},$$

where $(\boldsymbol{X}^t, \boldsymbol{Y}^t, \boldsymbol{Z}^t)$ is the current iterate after t iterations. That is, $\boldsymbol{Y}^{t+1}$ is the projection of the matrix $\boldsymbol{U}^t := \boldsymbol{X}^t - (\boldsymbol{W} + \boldsymbol{Z}^t)/\rho$ onto the intersection of the positive semidefinite

cone and the set of matrices with trace equal to zero. Such a projection can be computed explicitly by projecting the vector of eigenvalues $\boldsymbol{\lambda}^t$ of $\boldsymbol{U}^t$ onto the nonnegative simplex $\{\boldsymbol{y} \in \mathbf{R}^n : \boldsymbol{e}^T\boldsymbol{y} = k,\ \boldsymbol{y} \geq \boldsymbol{0}\}$. Zhang and Lu (2011, Proposition 2.6) and Van Den Berg and Friedlander (2008) can be consulted for further details.

We update $\boldsymbol{X}^{t+1}$ as the optimal solution of

$$\boldsymbol{X}^{t+1} = \operatorname*{arg\,min}_{\boldsymbol{X} \in \mathbf{R}^{n\times n}} \left\{\left\|\boldsymbol{X} - \left(\boldsymbol{Y}^t + \boldsymbol{Z}^t/\rho\right)\right\|_F^2 : \boldsymbol{X} \geq \boldsymbol{0},\ \boldsymbol{X}\boldsymbol{e} \leq \boldsymbol{e}\right\}. \tag{32}$$

Applying strong duality, we know that the optimal solution of (32) is given by

$$\boldsymbol{X}^{t+1} = \left[\left(\boldsymbol{Y}^{t+1} + \boldsymbol{Z}^t/\rho\right) - \frac{\boldsymbol{z}^*\boldsymbol{e} + \boldsymbol{e}(\boldsymbol{z}^*)^T}{2}\right]_+,$$

where the operator $[\cdot]_+$ is the projection onto the symmetric nonnegative cone $\Sigma^V \cap \mathbf{R}_+^{V\times V}$ given by $[[\boldsymbol{Z}]_+]_{ij} = \max\{0, Z_{ij}\}$ for all $\boldsymbol{Z} \in \Sigma^V$, and $\boldsymbol{z}^*$ is the optimal solution of the dual problem of (32) given by

$$\min_{\boldsymbol{z} \geq \boldsymbol{0}} \frac{1}{2}\left\|\left[\left(\boldsymbol{Y}^{t+1} + \boldsymbol{Z}^t/\rho\right) - \frac{\boldsymbol{z}\boldsymbol{e} + \boldsymbol{e}\boldsymbol{z}^T}{2}\right]_+\right\|_F^2 + \boldsymbol{z}^T\boldsymbol{e} - \frac{1}{2}\|\boldsymbol{Y}^{t+1} + \boldsymbol{Z}^t/\rho\|_F^2. \tag{33}$$

The objective function of the dual problem (33) is differentiable and coercive in $\boldsymbol{z}$, so it can be solved efficiently by applying the spectral projected gradient method of Birgin et al. (2000). We complete each iteration by performing an approximate dual ascent step to update the dual variable $\boldsymbol{Z}^{t+1}$. We stop the projected gradient method when the relative duality gap, given by $|v_p^{(t)} - v_d^{(t)}|/\max\{v_p^{(t)}, 1\}$, and primal constraint violation are both smaller than a desired error tolerance. We summarize the algorithm as Algorithm 1. Please see the work of Ames (2014, Section 6) for further implementation details.

4.2. Empirical Verification of Exact Recovery

We perform two sets of experiments, one to illustrate the recovery guarantee for dense graphs sampled from the heterogeneous planted cluster model and another to illustrate the guarantee when the noise is sparse. For the dense graph experiments, we fix $n = 1000$, and sample 10 graphs from the heterogeneous planted cluster model corresponding to the Bernoulli distributions $\Omega_{ij} = Bern(p_{ij})$ with probabilities of success p_{ij} given by

$$p_{ij} := \begin{cases} \left(1 - \left(\frac{0.35}{k+1}\right) i\right) p, & \text{if } i = j, \\ \left(1 - \left(\frac{0.35}{k+1}\right) \min\{i, j\}\right) q, & \text{if } i \neq j, \end{cases}$$

for $q = 0.25$ and each $p = \{0.25, 0.275, 0.3, \ldots, 0.975, 1\}$ and $\hat{r} \in \{20, 40, \ldots, 500\}$. We choose the number of clusters $k = \lfloor n/\hat{r} \rfloor$ and distribute the remaining $n - k\hat{r}$ nodes evenly

Algorithm 1 ADMM for (1)

Input: Initial iterates $\boldsymbol{X}^0 = \boldsymbol{Y}^0 = \boldsymbol{Z}^0 = \boldsymbol{0}$, augmented Lagrangian parameter $\rho > 0$, and stopping tolerance $\epsilon > 0$.

Output: Approximate solution $(\boldsymbol{X}^*, \boldsymbol{Y}^*, \boldsymbol{Z}^*)$ of (1).

For $t = 0, 1, 2 \ldots$ until converged

Compute spectral decomposition $\boldsymbol{V}^t \text{Diag}\boldsymbol{\lambda}^t (\boldsymbol{V}^t)^T = \boldsymbol{U}^t = \boldsymbol{X}^t - (\boldsymbol{W} + \boldsymbol{Z}^t)/\rho$.

Project $\boldsymbol{\lambda}^t$ onto the nonnegative simplex $\{\boldsymbol{y} \in \mathbf{R}^n : \boldsymbol{e}^T \boldsymbol{y} = k,\ \boldsymbol{y} \geq \boldsymbol{0}\}$ to obtain $\bar{\boldsymbol{\lambda}}^t$.

Update $\boldsymbol{Y}^{t+1} = \boldsymbol{V}^t \text{Diag}\bar{\boldsymbol{\lambda}}^t (\boldsymbol{V}^t)^T$.

Compute approximate optimal solution $\boldsymbol{z}^*$ of the dual subproblem (33) using spectral projected gradient method of Birgin et al. (2000).

Update $\boldsymbol{X}^{t+1} = \left[\left(\boldsymbol{Y}^{ts+1} + \boldsymbol{Z}^t/\rho \right) - \frac{\boldsymbol{z}^* \boldsymbol{e} + \boldsymbol{e}(\boldsymbol{z}^*)^T}{2} \right]_+$.

Update $\boldsymbol{Z}^{t+1}$ using approximate dual ascent

$$\boldsymbol{Z}^{t+1} = \boldsymbol{Z}^t - \rho(\boldsymbol{X}^{t+1} - \boldsymbol{Y}^{t+1}).$$

Compute primal feasibilty gap

$$pfeas = \min \left\{ \min_{ij} Y_{ij}^t, \min \left(\boldsymbol{e} - \boldsymbol{Y}^t \boldsymbol{e} \right) \right\}.$$

Compute estimates of primal and dual objective values (note that $v_d^{(t+1)}$ is not necessarily a lower bound on the optimal dual value, but is asymptotically converging to the optimal dual value):

$$v_p^{(t+1)} = \text{Tr}(\boldsymbol{W}\boldsymbol{Y}^t) \qquad v_d^{(t+1)} = k\lambda_{min}(\boldsymbol{W} + \boldsymbol{Z}^{t+1}) - \text{Tr}(\boldsymbol{X}^{t+1}\boldsymbol{Z}^{t+1}).$$

Calculate relative duality gap

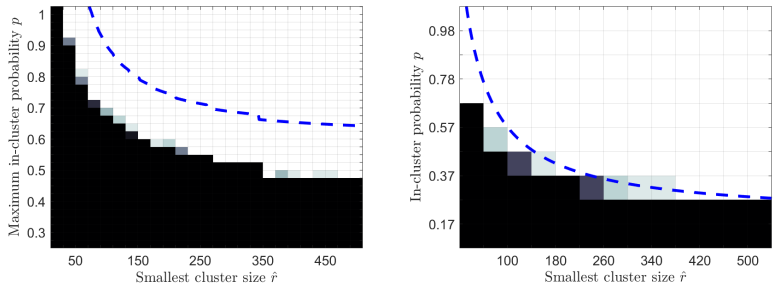
$$relgap = \frac{|v_p^{(t+1)} - v_d^{(t+1)}|}{\max \left\{ |v_p^{(t+1)}|, 1 \right\}}.$$

Declare sequence of iterates to have converged if $relgap < \epsilon$ and $pfeas > -\epsilon$.

End For

among $k-1$ clusters to ensure that at least one cluster has minimum size. Under this choice of p_{ij} the smallest gap between the in-cluster and between-cluster means occurs when $i = 1$ and $j = k$; this implies that

$$\gamma = \left(1 - \frac{0.35k}{k+1} \right) p - q.$$



(a) Dense Noise (b) Sparse Noise

Figure 1: Empirical recovery rate for n-node graph with k planted cliques of size at least $\hat{r}$ and $\boldsymbol{W}$ generated according to the planted heterogeneous cluster model with distributions $\Omega_{ij} = Bern(p_{ij})$. Brighter colors indicate higher rates of recovery, with black corresponding to 0 recoveries and white corresponding to 10 recoveries (out of 10 trials). The dashed curves indicate the phase transition to perfect recovery predicted by Theorem 3.

For each graph G, we call the ADMM algorithm sketched above to solve (5); in the algorithm, we use penalty parameter $\rho = \min\{\max\{5n/k, 80\}, 500\}/2$, stopping tolerance $\epsilon = 10^{-4}$, and maximum number of iterations 100. We declare the block structure of G to be recovered if $\|\boldsymbol{X}^* - \boldsymbol{X}_0\|_F^2 / \|\boldsymbol{X}_0\|_F^2 < 10^{-3}$, where $\boldsymbol{X}^*$ is the solution returned by the ADMM algorithm and $\boldsymbol{X}_0$ is the proposed solution given by (3). Note that Theorem 3 implies that we should expect exact recovery (w.h.p.) provided that $\gamma\hat{r} = \Omega\left(\sqrt{\tilde{\sigma}^2 n}\right)$. Figure 1(a) illustrates the empirical success rate for each choice of $\hat{r}$ and p, as well as the curve $p = \left((k+1)/(0.65k+1)\right)(q + \frac{1}{2}\sqrt{n})$, where we use the upper bound $\tilde{\sigma}^2 \leq 1/4$ to estimate the constant term in (6).

We perform identical experiments for graphs sampled from the homogeneous planted cluster model with sparse noise. In particular, we fix $n = 1000$ and set $q = 1/\sqrt{n}$. We then sample 10 graphs from the planted cluster model corresponding to the Bernoulli distributions $\Omega_{ij} = Bern(p)$ if $i = j$ and $\Omega_{ij} = Bern(q)$ if $i \neq j$ for each $\hat{r} \in \{20, 60, \ldots, 440, 500\}$ and $p = tq$ for 10 equally spaced scaling factors t between 2 and $\lfloor\sqrt{n}\rfloor$. As before, we set $k = \lfloor n/\hat{r} \rfloor$ and distribute the remaining nodes equally amongst the clusters so that the smallest has size $\hat{r}$ and $r_{k+1} = 0$. For each graph G, we call the ADMM algorithm to solve (5) (with the same parameters as before) and declare the block structure of G recovered if $\|\boldsymbol{X}^* - \boldsymbol{X}_0\|_F^2 / \|\boldsymbol{X}_0\|_F^2 < 10^{-3}$. Theorem 3 suggests that we should expect recovery of the

cluster structure in the case that

$$p > \Omega\left(\frac{1}{\sqrt{n}} + \frac{n^{1/4}}{\hat{r}}\right),$$

for this particular choice of p and q. Note that this implies that we have perfect recovery (w.h.p.) for $\hat{r} = \Omega(n^{1/4})$, rather than $\Omega(\sqrt{n})$ (as observed in the dense case). Figure 1(b) provides the empirical success rate for each choice of $\hat{r}$ and p, as well as the curve $p = 1/\sqrt{n} + n^{1/4}/\hat{r}$. It is clear that we are able to recover significantly smaller clusters under sparse noise than under dense noise, in accordance with (6).

5. Conclusions

We have established theoretical guarantees for graph clustering via a semidefinite relaxation of the densest k-disjoint problem. These results add to the growing corpus of evidence that clustering, while intractable in general, is possible if we seek to cluster clusterable data, i.e., data consisting of well-defined and well-separated groups of similar items. Moreover, our results provide further evidence that the $\omega(\sqrt{n})$ barrier can be broken for perfect cluster recovery in approximately sparse graphs and, specifically, that the size of recoverable clusters scales logarithmically with n at worst in the special case that all clusters are roughly the same size. Finally, our semidefinite relaxation requires only an estimate of the number of clusters present in the data as input.

Our results suggest several areas of further research. The numerical simulations suggest that our theoretical guarantees may be overly conservative, especially in the dense noise case; further investigation is needed to determine if tighter estimates on the minimum size of clusters efficiently recoverable exist. Moreover, our model assumes clusters are disjoint. This is clearly not met in many practical applications; for example, returning to the social networking realm, users may belong to several overlapping communities. It would be worthwhile to see how our model and recovery guarantees can be modified to address overlapping clusters. Finally, our algorithm for graph clustering requires the solution of a semidefinite program, which may be impractical for even moderately large graphs. For example, the proposed algorithm, based on the ADMM, has per-iteration cost of $O(n^3)$ flops per iteration, primarily to compute the spectral decomposition needed to update $\boldsymbol{Y}$. Classical methods based on interior-point methods will scale even more poorly. Efficient, scalable methods for solving this semidefinite relaxation, and semidefinite programming in general, are needed.

Acknowledgments

We are grateful to John Bruer and Joel Tropp for their insights and helpful suggestions. Aleksis Pirinen was supported by a California Institute of Technology Summer Undergraduate Research Fellowship (SURF) using funds provided by Office of Naval Research (ONR) award N000014-11-1002. Brendan Ames was supported by University of Alabama Research Grants RG14678 and RG14838.

Appendix A. Proof of Lemma 7

We give the full proof of Lemma 7 in this appendix.

Proof We fix $q, s \in \{1, \ldots, k\}$ such that $q \neq s$ and assume without loss of generality that $r_q \leq r_s$. By the definition (13) of $\boldsymbol{y} := \boldsymbol{y}^{q,s}$ and the triangle inequality, we have

$$\|\boldsymbol{y}\|_\infty \leq \frac{1}{r_s}\left(\|\boldsymbol{b}_{q,s}\|_\infty + \frac{|\boldsymbol{b}_{q,s}^T \boldsymbol{e}|}{r_q + r_s}\right).$$

For simplicity, let $\boldsymbol{b}_1 := \boldsymbol{b}_{q,s}$ and $\boldsymbol{b}_2 := \boldsymbol{b}_{s,q}$. It follows from (14) and our choice of $\boldsymbol{\lambda}$ that the ith element of $\boldsymbol{b}_1$, denoted b_i^1, is given by

$$b_i^1 = r_s\left(\lambda_i - \frac{1}{2r_q}(\mu_{qq} r_q - \tau)\right) + \left(\boldsymbol{\lambda}_{C_s}^T \boldsymbol{e} - \frac{1}{2}(\mu_{ss} r_s - \tau)\right) - \left(\sum_{j \in C_s} w_{ij} - \mu_{qs} r_s\right).$$

It follows from the definition (11) of $\boldsymbol{\lambda}_{C_s}$ that

$$\boldsymbol{\lambda}_{C_s}^T \boldsymbol{e} = \frac{1}{2r_s}\left(\boldsymbol{e}^T \boldsymbol{W}_{C_s, C_s} \boldsymbol{e} - r_s \tau\right),$$

which implies that

$$\left|\boldsymbol{\lambda}_{C_s}^T \boldsymbol{e} - \frac{1}{2}(\mu_{ss} r_s - \tau)\right| = \frac{1}{2r_s}\left|\boldsymbol{e}^T \boldsymbol{W}_{C_s, C_s} \boldsymbol{e} - \mu_{ss} r_s^2\right|.$$

Applying (20) with $T = n$ to the right-hand side in the equation above shows that

$$|\boldsymbol{e}^T \boldsymbol{W}_{C_s, C_s} \boldsymbol{e} - \mu_{ss} r_s^2| \leq 6 \max\{\sqrt{\sigma_{ss}^2 r_s^2 \log n}, \log n\}, \tag{34}$$

with high probability, which in turn implies that

$$\left|\boldsymbol{\lambda}_{C_s}^T \boldsymbol{e} - \frac{1}{2}(\mu_{ss} r_s - \tau)\right| \leq 3 \max\left\{\sqrt{\sigma_{ss}^2 \log n}, \frac{\log n}{r_s}\right\},$$

with high probability. Similarly, applying (20) with $T = n$ to the sum $\sum_{j \in C_s} w_{ij}$ shows that

$$\left|\sum_{j \in C_s} w_{ij} - \mu_{qs} r_s\right| \leq 6 \max\left\{\sqrt{\sigma_{qs}^2 r_s \log n}, \log n\right\},$$

for all $i \in C_q$, with high probability. Finally, we note that

$$\left|\lambda_i - \frac{1}{2r_q}(\mu_{qq}r_q - \tau)\right| \leq \frac{1}{r_q}\left|\sum_{j\in C_q} w_{ij} - \mu_{qq}r_q\right| + \frac{1}{2r_q^2}\left|\boldsymbol{e}^T\boldsymbol{W}_{C_q,C_q}\boldsymbol{e} - \mu_{qq}r_q^2\right|.$$

We bound the first term in the sum using (20) with $T = n$, which establishes that

$$\left|\sum_{j\in C_q} w_{ij} - \mu_{qq}r_q\right| \leq 6\max\left\{\sqrt{\sigma_{qq}^2 r_q \log n}, \log n\right\},$$

w.h.p., and note that the second term has upper bound

$$\left|\boldsymbol{e}^T\boldsymbol{W}_{C_q,C_q}\boldsymbol{e} - \mu_{qq}r_q^2\right| \leq 6\max\left\{\sqrt{\sigma_{qq}^2 r_q^2 \log n}, \log n\right\},$$

w.h.p., by a calculation identical to that used to obtain (34). Applying these bounds using the triangle inequality and the union bound over all $i \in C_q$, we conclude that

$$\begin{aligned}
\|\boldsymbol{b}_1\|_\infty &\leq r_s\left|\lambda_i - \frac{1}{2r_q}(\mu_{qq}r_q - \tau)\right| + \left|\boldsymbol{\lambda}_{C_s}^T\boldsymbol{e} - \frac{1}{2}(\mu_{ss}r_s - \tau)\right| + \left|\sum_{j\in C_s} w_{ij} - \mu_{qs}r_s\right| \\
&\leq r_s\left(\frac{6}{r_q}\max\left\{\sqrt{\sigma_{qq}^2 r_q \log n}, \log n\right\} + 3\max\left\{\sqrt{\sigma_{qq}^2 \log n}, \frac{\log n}{r_q}\right\}\right) \\
&\qquad + 3\max\left\{\sqrt{\sigma_{ss}^2 \log n}, \frac{\log n}{r_s}\right\} + 6\max\left\{\sqrt{\sigma_{qs}^2 r_s \log n}, \log n\right\} \\
&= O\left(r_s\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\}\right), \qquad (35)
\end{aligned}$$

with high probability.

We next bound $\left|\boldsymbol{b}_1^T\boldsymbol{e}\right|$. We have

$$\boldsymbol{b}_1^T\boldsymbol{e} = r_s\left(\boldsymbol{\lambda}_{C_q}^T\boldsymbol{e} - \frac{1}{2}(\mu_{qq}r_q - \tau)\right) + r_q\left(\boldsymbol{\lambda}_{C_s}^T\boldsymbol{e} - \frac{1}{2}(\mu_{ss}r_s - \tau)\right) + \left(\mu_{qs}r_s r_q - \boldsymbol{e}^T\boldsymbol{W}_{C_q,C_s}\boldsymbol{e}\right).$$

Applying (20) to bound the sum of the entries of $\boldsymbol{W}_{C_q,C_s}$ and the above concentration inequalities for $\boldsymbol{\lambda}_{C_q}^T\boldsymbol{e}$ and $\boldsymbol{\lambda}_{C_s}^T\boldsymbol{e}$ we have

$$\begin{aligned}
\left|\boldsymbol{b}_1^T\boldsymbol{e}\right| &\leq r_s\left|\boldsymbol{\lambda}_{C_q}^T\boldsymbol{e} - \frac{1}{2}(\mu_{qq}r_q - \tau)\right| + r_q\left|\boldsymbol{\lambda}_{C_s}^T\boldsymbol{e} - \frac{1}{2}(\mu_{ss}r_s - \tau)\right| + \left|\mu_{qs}r_s r_q - \boldsymbol{e}^T\boldsymbol{W}_{C_q,C_s}\boldsymbol{e}\right| \\
&\leq 3r_s\max\left\{\sqrt{\sigma_{qq}^2 \log n}, \frac{\log n}{r_q}\right\} + 3r_q\max\left\{\sqrt{\sigma_{ss}^2 \log n}, \frac{\log n}{r_s}\right\} \\
&\qquad + 6\max\left\{\sqrt{\sigma_{qs}^2 r_q r_s \log n}, \log n\right\} \\
&= O\left(r_s\max\left\{\sqrt{\tilde{\sigma}^2 \log n}, \frac{\log n}{\hat{r}}\right\}\right), \qquad (36)
\end{aligned}$$

w.h.p. Finally, we bound $\|\boldsymbol{y}\|_\infty$ using (35) and (36):

$$\|\boldsymbol{y}^{q,s}\|_\infty \leq \frac{1}{r_s}\left(\|\boldsymbol{b}_1\|_\infty + \frac{|\boldsymbol{b}_1^T\boldsymbol{e}|}{r_q + r_s}\right) = O\left(\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\}\right),$$

w.h.p. Since this holds for any $q, s \in \{1, \ldots, k\}$ such that $q \neq s$, we conclude that

$$\|\boldsymbol{y}\|_\infty = O\left(\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\}\right),$$

w.h.p. An identical argument shows that

$$\|\boldsymbol{z}^{q,s}\|_\infty = O\left(\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\}\right),$$

w.h.p. We conclude that

$$\|\boldsymbol{y}^{q,s}\|_\infty + \|\boldsymbol{z}^{q,s}\|_\infty = O\left(\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}}}, \frac{\log n}{\hat{r}}\right\}\right),$$

w.h.p. ■

Appendix B. Proof of Proposition 9

We next prove Proposition 9.

Proof We follow the proof of Lemma 4.3 given by Ames (2014). Fix $q \in \{1, \ldots, k\}$ and $i \in C_q$. It follows from (11) that

$$\lambda_i = \sum_{j\in C_q} w_{ij} - \frac{1}{2r_q}\boldsymbol{e}^T\boldsymbol{W}_{C_q,C_q}\boldsymbol{e} - \frac{\tau}{2},$$

for each $i \in C_q$. Applying (20) with $S = \sum_{j\in C_q} w_{ij}$ and $T = n$ yields

$$\sum_{j\in C_q} w_{ij} \geq \mu_{qq}r_q - 6\max\left\{\sqrt{\sigma_{qq}^2 r_q \log n}, \log n\right\},$$

w.h.p. Moreover, by a similar argument, we have

$$\frac{1}{2r_q}\boldsymbol{e}^T\boldsymbol{W}_{C_q,C_q}\boldsymbol{e} \leq \frac{1}{2}\left(\mu_{qq}r_q + 6\max\left\{\sqrt{\sigma_{qq}^2 \log n}, \frac{\log n}{r_q}\right\}\right),$$

w.h.p. Combining the above inequalities shows that

$$\lambda_i \geq r_q\left(\frac{\mu_{qq}}{2} - \epsilon\gamma - O\left(\max\left\{\sqrt{\frac{\sigma_{qq}^2 \log n}{r_q}}, \frac{\log n}{r_q}\right\}\right)\right),$$

w.h.p. Since $\gamma > 0$ by (6), this implies that there exists constant $c > 0$ such that if

$$\epsilon \leq \frac{1}{2\gamma}\left(\mu_{qq} - c\max\left\{\sqrt{\frac{\sigma_{qq}^2 \log n}{r_q}}, \frac{\log n}{r_q}\right\}\right), \tag{37}$$

then $\lambda_i \geq 0$ w.h.p. Applying the union bound over all $q = 1, 2, \ldots, k$ and $i \in C_q$ shows that each entry of $\boldsymbol{\lambda}_{C_q}$ is nonnegative w.h.p. if ϵ is chosen to satisfy (37) for all q. ■

Appendix C. Proof of Proposition 10

Our proof of Proposition 10 follows a similar structure to that of Ames (2014, Lemma 4.4).

Proof Fix $q \neq s$ with $q \in 1, \ldots, k$. Applying (34) and (20) with $S = \boldsymbol{e}^T \boldsymbol{W}_{C_q,C_s}\boldsymbol{e}$ and $T = n$, we have

$$\begin{aligned}
&r_s \boldsymbol{e}^T \boldsymbol{W}_{C_q,C_q}\boldsymbol{e} - r_q \boldsymbol{e}^T \boldsymbol{W}_{C_q,C_s}\boldsymbol{e} \\
&\quad \geq (\mu_{qq} - \mu_{qs}) r_s r_q^2 - 6 r_s \max\left\{\sqrt{\sigma_{qq}^2 r_q^2 \log n}, \log n\right\} - 6 r_q \max\left\{\sqrt{\sigma_{qs}^2 r_s r_q \log n}, \log n\right\} \\
&\quad \geq r_q^2 r_s \left(\gamma - 12\max\left\{\sqrt{\frac{\tilde{\sigma}^2 \log n}{\hat{r}^2}}, \frac{\log n}{\hat{r}^2}\right\}\right),
\end{aligned}$$

w.h.p. This implies that $r_s \boldsymbol{e}^T \boldsymbol{W}_{C_q,C_q}\boldsymbol{e} \geq r_q \boldsymbol{e}^T \boldsymbol{W}_{C_q,C_s}\boldsymbol{e}$ w.h.p. if (23) is satisfied. ■

Appendix D. Proof of Lemma 12

We next prove Lemma 12.

Proof We will make repeated use of the following lemma, which specializes the concentration inequality on the spectral norm of a random symmetric matrix with i.i.d. mean zero entries given by Bandeira and van Handel (2016, Corollary 3.12).

Lemma 14 *Let $\boldsymbol{A} = [a_{ij}] \in \Sigma^n$ be a random symmetric matrix with i.i.d. mean zero entries a_{ij} having variance at most σ^2 and satisfying $|a_{ij}| \leq 1$. Then there exists constant $C > 0$ such that*

$$\Pr\left\{\|\boldsymbol{A}\| > C\max\left\{\sqrt{\sigma^2 n}, \sqrt{T}\right\}\right\} \leq nT^{-7} \tag{38}$$

for all $T > 0$.

Proof (of Lemma 14) Corollary 3.12 of Bandeira and van Handel (2016) establishes that for each $t > 0$ there exists $\tilde{c} = \tilde{c}(t) > 0$ such that

$$\Pr\left\{\|\boldsymbol{A}\| \geq 3\sqrt{\sigma^2 n} + t\right\} \leq ne^{-\tilde{c}t^2}. \tag{39}$$

Here we have substituted the upper bound $\sigma^2 n \geq \tilde{\sigma}^2$, in place of $\tilde{\sigma} := \max_i \sum_j \mathbf{E}[X_{ij}^2]$ in the original statement of Corollary 3.12. Let $t = (C-3)\max\{\sqrt{\sigma^2 n}, \sqrt{\log T}\}$ where C is chosen large enough that $\tilde{c}(C-3)^2 > 7$. In this case, (39) specializes to

$$\Pr\left\{\|\boldsymbol{A}\| \geq C\max\{\sqrt{\sigma^2 n}, \sqrt{\log n}\}\right\} \leq ne^{-7\max\{\sigma^2 n, \log n\}} \leq ne^{-7\log n} = nT^{-7}.$$

This completes the proof. ■

Before we continue with the derivation of the desired bound on $\|\tilde{\boldsymbol{S}}_1\|$, we note that the entries $[\tilde{\boldsymbol{S}}_1]_{ij}$ of $\tilde{\boldsymbol{S}}_1$ all satisfy $|[\tilde{\boldsymbol{S}}_1]_{ij}| \leq 1$ if we assume that $w_{ij} \in [0,1]$ for all i, j; note that an identical argument establishes the result if we make the weaker assumption that the entries of $\boldsymbol{W}$ are bounded with high probability. On the other hand, note that the entries of $\tilde{\boldsymbol{S}}_1$ are not identically distributed (but are independent) since each w_{ij} is sampled according to Ω_{qs}, where $i \in C_q$, $j \in C_s$. However, we know that $\sigma_{qs}^2 \leq \tilde{\sigma}^2$ by our definition of $\tilde{\sigma}^2$. Moreover, $\mathbf{E}[[\tilde{\boldsymbol{S}}_1]_{ij}] = \mathbf{E}[\mu_{qs} - w_{ij}] = 0$. Thus, we can apply Lemma 14 to place a bound on $\|\tilde{\boldsymbol{S}}_1\|$. Doing so establishes that (28) holds w.h.p. ■

Appendix E. Proof of Lemma 13

We conclude with the following proof of Lemma 13.

Proof Note that $\|\tilde{\boldsymbol{S}}_2\| \leq \|\boldsymbol{\lambda} - \mathbf{E}[\boldsymbol{\lambda}]\|\sqrt{r_{k+1}}$. Thus, it remains to bound $\|\boldsymbol{\lambda} - \mathbf{E}[\boldsymbol{\lambda}]\|$. To do so, fix $q \in \{1, 2, \ldots, k\}$. Recall that

$$\boldsymbol{\lambda}_{C_q} - \mathbf{E}[\boldsymbol{\lambda}_{C_q}] = \frac{1}{r_q}(\boldsymbol{W}_{C_q,C_q}\boldsymbol{e} - \mu_{qq} r_q \boldsymbol{e}) - \frac{1}{r_q^2}(\boldsymbol{e}^T \boldsymbol{W}_{C_q,C_q}\boldsymbol{e} - \mu_{qq} r_q^2)\boldsymbol{e}.$$

Applying (38) with $T = n$ establishes that

$$\begin{aligned}
\|\boldsymbol{W}_{C_q,C_q}\boldsymbol{e} - \mu_{qq} r_q \boldsymbol{e}\| &\leq \|\boldsymbol{W}_{C_q,C_q} - \mu_{qq}\boldsymbol{e}\boldsymbol{e}^T\|\|\boldsymbol{e}\| \\
&\leq C\sqrt{r_q}\max\{\sigma_{qq}\sqrt{r_q}, \sqrt{\log n}\},
\end{aligned}$$

w.h.p. On the other hand, Bernstein's inequality establishes that

$$|\boldsymbol{e}^T \boldsymbol{W}_{C_q,C_q}\boldsymbol{e} - \mu_{qq} r_q^2| \leq 6\max\left\{\sqrt{\sigma_{qq}^2 r_q \log n}, \log n\right\},$$

w.h.p. Combining these two inequalities using the triangle inequality establishes that

$$\begin{aligned}\|\boldsymbol{\lambda}_{C_q} - \mathbf{E}[\boldsymbol{\lambda}_{C_q}]\| \leq & C \max\left\{\sigma_{qq}, \sqrt{\frac{\log n}{r_q}}\right\} + 6\max\left\{\sqrt{\frac{\sigma_{qq}^2 \log n}{r_q^2}}, \frac{\log n}{r_q^{3/2}}\right\} \\ = & O\left(\max\left\{\sigma_{qq}, \sqrt{\frac{\log n}{\hat{r}}}\right\}\right),\end{aligned}$$

w.h.p. Finally, applying the union bound over all choices of q shows that

$$\|\boldsymbol{\lambda} - \mathbf{E}[\boldsymbol{\lambda}]\|^2 = \sum_{q=1}^{k} \|\boldsymbol{\lambda}_{C_q} - \mathbf{E}[\boldsymbol{\lambda}_{C_q}]\|^2 = O\left(k \max\left\{\sigma_{qq}^2, \frac{\log n}{\hat{r}}\right\}\right),$$

w.h.p. This establishes that

$$\|\tilde{\boldsymbol{S}}_2\|^2 = O\left(k r_{k+1} \max\left\{\sigma_{qq}^2, \frac{\log n}{\hat{r}}\right\}\right),$$

w.h.p., as required. ■

References

E. Abbe, A. Bandeira, and G. Hall. Exact recovery in the stochastic block model. *IEEE Transactions on Information Theory*, 62(1):471–487, 2016.

N. Ailon, Y. Chen, and H. Xu. Breaking the small cluster barrier of graph clustering. In *International Conference on Machine Learning*, pages 995–1003, 2013.

B. Ames. Guaranteed clustering and biclustering via semidefinite programming. *Mathematical Programming*, 147(1-2):429–465, 2014.

B. Ames. Guaranteed recovery of planted cliques and dense subgraphs by convex relaxation. *Journal of Optimization Theory and Applications*, 167(2):653–675, 2015.

B. Ames and S. Vavasis. Nuclear norm minimization for the planted clique and biclique problems. *Mathematical programming*, 129(1):69–89, 2011.

B. Ames and S. Vavasis. Convex optimization for the planted k-disjoint-clique problem. *Mathematical Programming*, 143(1-2):299–337, 2014.

A. Amini and E. Levina. On semidefinite relaxations for the block model. *The Annals of Statistics*, 46(1):149–179, 2018.

A. Bandeira and R. van Handel. Sharp nonasymptotic bounds on the norm of random matrices with independent entries. *The Annals of Probability*, 44(4):2479–2506, 2016.

E. Birgin, M.J. Martínez, and M. Raydan. Nonmonotone spectral projected gradient methods on convex sets. *SIAM Journal on Optimization*, 10(4):1196–1211, 2000.

S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

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

T. Cai and X. Li. Robust and computationally feasible community detection in the presence of arbitrary outlier nodes. *The Annals of Statistics*, 43(3):1027–1059, 2015.

Y. Chen and J. Xu. Statistical-computational phase transitions in planted models: the high-dimensional setting. In *International Conference on Machine Learning*, pages 244–252, 2014.

Y. Chen, A. Jalali, S. Sanghavi, and H. Xu. Clustering partially observed graphs via convex optimization. *The Journal of Machine Learning Research*, 15(1):2213–2238, 2014a.

Y. Chen, S. Sanghavi, and H. Xu. Improved graph clustering. *IEEE Transactions on Information Theory*, 60(10):6440–6455, 2014b.

I. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: spectral clustering and normalized cuts. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 551–556. ACM, 2004.

G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 4th edition, 2013.

O. Guédon and R. Vershynin. Community detection in sparse networks via Grothendieck's inequality. *Probability Theory and Related Fields*, pages 1–25, 2015.

B. Hajek, Y. Wu, and J. Xu. Achieving exact cluster recovery threshold via semidefinite programming. In *IEEE International Symposium on Information Theory*, pages 1442–1446. IEEE, 2015.

P. Holland, K. Laskey, and S. Leinhardt. Stochastic blockmodels: first steps. *Social Networks*, 5(2):109–137, 1983.

A. Jalali, Q. Han, I. Dumitriu, and M. Fazel. Relative density and exact recovery in heterogeneous stochastic block models. *arXiv preprint arXiv:1512.04937*, 2015.

J. Lei and A. Rinaldo. Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43(1):215–237, 2015.

J. Leskovec, K. Lang, A. Dasgupta, and M. Mahoney. Statistical properties of community structure in large social and information networks. In *International Conference on World Wide Web*, pages 695–704. ACM, 2008.

J. Leskovec, K. Lang, A. Dasgupta, and M. Mahoney. Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009.

C. Mathieu and W. Schudy. Correlation clustering with noisy input. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 712–728. Society for Industrial and Applied Mathematics, 2010.

A. Nellore and R. Ward. Recovery guarantees for exemplar-based clustering. *Information and Computation*, 245:165–180, 2015.

A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. *Advances in Neural Information Processing Systems*, 2:849–856, 2002.

S. Oymak and B. Hassibi. Finding dense clusters via "low rank + sparse" decomposition. *Arxiv preprint arXiv:1104.5186*, 2011.

J. Peng and Y. Wei. Approximating k-means-type clustering via semidefinite programming. *SIAM Journal on Optimization*, 18(1):186–205, 2007.

T. Qin and K. Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. In *Advances in Neural Information Processing Systems*, pages 3120–3128, 2013.

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011.

K. Rohe, T. Qin, and H. Fan. *Statistica Sinica*, pages 1771–1786, 2014.

E. Van Den Berg and M. Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing*, 31(2):890–912, 2008.

R.K. Vinayak, S. Oymak, and B. Hassibi. Sharp performance bounds for graph clustering via convex optimization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8297–8301. IEEE, 2014.

B. Yan, P. Sarkar, and X. Cheng. Provable estimation of the number of blocks in block models. *arXiv preprint arXiv:1705.08580*, 2017.

Y. Zhang and Z. Lu. Penalty decomposition methods for rank minimization. In *Advances in Neural Information Processing Systems*, pages 46–54, 2011.

Journal of Machine Learning Research 20 (2019) 1-45 Submitted 6/16; Revised 2/18; Published 2/19

Kernels for Sequentially Ordered Data

Franz J. Király F.KIRALY@UCL.AC.UK
Department of Statistical Science
University College London
London WC1E 6BT, United Kingdom

Harald Oberhauser OBERHAUSER@MATHS.OX.AC.UK
Mathematical Institute
University of Oxford
Oxford OX2 6GG, United Kingdom

Editor: Le Song

Abstract

We present a novel framework for learning with sequential data of any kind, such as multivariate time series, strings, or sequences of graphs. The main result is a "sequentialization" that transforms any kernel on a given domain into a kernel for sequences in that domain. This procedure preserves properties such as positive definiteness, the associated kernel feature map is an ordered variant of sample (cross-)moments, and this sequentialized kernel is consistent in the sense that it converges to a kernel for paths if sequences converge to paths (by discretization). Further, classical kernels for sequences arise as special cases of this method. We use dynamic programming and low-rank techniques for tensors to provide efficient algorithms to compute this sequentialized kernel.

Keywords: Sequential data, kernels, signature, ordered moments, signature kernels

1. Introduction

Sequentially ordered data are ubiquitous in modern science, occurring as time series, location series, or, more generally, sequentially observed samples of structured objects. Two stylised facts make learning with sequential data an ongoing challenge:

(A) Sequential data is very diverse, e.g. it includes sequences of different length of scalars, letters, images, graphs (in time series, text, video, network evolution) or even heterogeneous combinations of these. This diversity is usually addressed by *ad-hoc approaches, extensive pre-processing and manual extraction of features*, thus specific to the data at hand.

(B) Sequential data is often large, with sequences easily obtaining the length of hundreds, thousands, millions. Hence, the data sets quickly become *very huge*, and with them computational cost.

In this paper, we present a novel approach to learning with sequential data based on joining ideas from *stochastic analysis* and *kernel learning*, addressing the points above:

©2019 Franz J. Király and Harald Oberhauser.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/16-314.html.

(a) The (discretized) *signature* is a *canonical feature map* for sequences. It can intuitively be described as an ordered version of sample moments. It completely describes a sequence and makes sequences of different size and length comparable.

(b) The *kernel trick* applied to signature features avoids the combinatorial explosion of coordinates inherent to the signature feature map (in analogy to the classic polynomial kernel in $\mathbb{R}^d$).

The main results of this article can be described as follows: if $\mathcal{X}$ is a topological space ("structured objects") and we denote with $\mathcal{X}^+ = \bigcup_{\ell \geq 1} \mathcal{X}^\ell$ the set of arbitrary length sequences in $\mathcal{X}$ ("sequences of structured objects") then we construct a map that takes

$$\text{a kernel } \mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \text{ to a kernel } \mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}.$$

We call k^+ the *discretized signature kernel* (or *sequentialization*) over k and provide efficient algorithms to evaluate k^+. This transformation is canonical in the sense that it preserves important properties of k, such as positive definiteness and it is consistent in the sense that $\mathrm{k}(x, y)$ for $x, y \in \mathcal{X}^+$ converges to an inner product of a "feature map" for paths that is classic in stochastic analysis (the path signature) if the sequences x, y converge to paths (from discrete to continuous time).

Classical kernels for sequences such as string, global alignment or relation-convolution kernels can be identified as special cases of the discretized signature kernel k^+; for example string kernels arise by taking $\mathcal{X}$ to be a finite set of "letters". More interestingly, even for $\mathcal{X} = \mathbb{R}$ this yields new kernels for time series by the sequentialization of classic non-linear kernels such as Gaussian, Laplace, etc. From a methodological point of view, this gives a canonical way to transform static features (resp. kernels k) for structured objects into features (resp. kernels k^+) for evolving structured objects.

1.1. Kernels for paths

We first discuss the situation for path-valued observations (continuous time) rather than sequence-valued observations (discrete time) — the latter then follows by discretizing time. That is, if we denote with $\mathcal{P}_\mathcal{X} \subset \mathcal{X}^{[0,1]}$ a sufficiently regular subset of the space of paths that evolve in $\mathcal{X}$ (without loss of generality the paths are parameterized by $[0, 1]$), our aim is to transform

$$\text{a kernel } \mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \text{ into a kernel } \mathrm{k}^{\oplus} : \mathcal{P}_\mathcal{X} \times \mathcal{P}_\mathcal{X} \to \mathbb{R}.$$

Therefore, first recall that $\mathcal{X} \ni u \mapsto \mathrm{k}_u := \mathrm{k}(u, \cdot) \in \mathcal{H} \subset \mathbb{R}^\mathcal{X}$ can be thought of as the "(canonical) feature map" induced by k. It is then natural to proceed as follows:

1. **Lift a path evolving in data space $\mathcal{X}$ to a path evolving in feature space $\mathcal{H}$.** We lift the path x that evolves in data space $\mathcal{X}$ to a path[1] k_x that evolves in the feature space $\mathcal{H}$ via the kernel k. That is, we map

$$x = (x(t))_{t\in[0,1]} \in \mathcal{P}_\mathcal{X} \text{ to } \mathrm{k}_x = \big(\mathrm{k}_{x(t)}\big)_{t\in[0,1]} \in \mathcal{P}_\mathcal{H}$$

where $\mathcal{P}_\mathcal{H}$ denotes a sufficiently regular subset of $\mathcal{H}^{[0,1]}$.

1. Note the slight abuse of notation that $\mathrm{k}_u \in \mathcal{H}$ if $u \in \mathcal{X}$ and $\mathrm{k}_x \in \mathcal{P}_\mathcal{H}$ if $x \in \mathcal{P}_\mathcal{X}$.

2. **Signature features for paths in $\mathcal{H}$.** We use a feature map that is well-known in stochastic analysis for paths that evolve in *linear* spaces such as $\mathcal{H}$. This map is called the signature map S. It maps a path $h \in \mathcal{P}_\mathcal{H}$ into a series of tensors,

$$h \mapsto \mathrm{S}(h) \in \prod_{m\geq 0} \mathcal{H}^{\otimes m}.$$

   By applying this map to the lifted path $\mathrm{k}_x \in \mathcal{P}_\mathcal{H}$, we get the feature map $\mathcal{P}_\mathcal{X} \to \prod_{m\geq 0}\mathcal{H}^{\otimes m}$, defined as $x \mapsto \mathrm{S}(\mathrm{k}_x)$.

3. **Signature kernels**. Taking inner products yields the kernel

$$\mathrm{k}^\oplus : \mathcal{P}_\mathcal{X} \times \mathcal{P}_\mathcal{X} \to \mathbb{R} \text{ defined as inner product } \mathrm{k}^\oplus(x,y) = \langle \mathrm{S}(\mathrm{k}_x), \mathrm{S}(\mathrm{k}_y)\rangle.$$

   It turns out, that $\mathrm{k}^\oplus$ has a recursive structure that allows for an efficient and robust calculation even though $\mathrm{k}_x, \mathrm{k}_y$ are paths evolving in the (in general infinite-dimensional) state space $\mathcal{H}$.

Below we give more details about the last two points of this construction.

1.2. The signature map

The signature S maps a (sufficiently regular) path $h \in \mathcal{P}_\mathcal{H} \subset \mathcal{H}^{[0,1]}$ into a series of tensors,

$$h \mapsto \mathrm{S}(h) = (\mathrm{S}_m(h))_{m\geq 0} \in \prod_{m\geq 0} \mathcal{H}^{\otimes m}$$

where by convention $\mathcal{H}^{\otimes 0} := \mathbb{R}$ and $\mathrm{S}_0(h) := 1$. The remaining terms $\mathrm{S}_m(h)$ are defined as follows: the first degree, $\mathrm{S}_1(h)$, of the signature is simply the average change,

$$\mathrm{S}_1(h) := \mathbb{E}_t[\dot{h}(t)] \in \mathcal{H},$$

where $\dot{h}$ denotes the derivative of h and the expectation is taken over t sampled uniformly from $[0,1]$. The second degree of the signature, $\mathrm{S}_2(h)$, is the (non-centred) covariance of changes at two subsequent time points, that is

$$\mathrm{S}_2(h) := \frac{1}{2}\mathbb{E}_{t_1<t_2}[\dot{h}(t_1)\otimes\dot{h}(t_2)] \in \mathcal{H}^{\otimes 2},$$

where the expectation is over the time points t_1, t_2 sampled from the uniform distribution on $[0,1]$ and put in chronological order. In general, the m-th degree of the signature, $\mathrm{S}_m(h)$, is defined as the m-th moment tensor of the infinitesimal changes, where the expectation is taken over m time points, $t_1, \dots, t_m$, that are sampled from the order statistic[2] of the uniform distribution on $[0,1]$, that is

$$\mathrm{S}_m(h) := \frac{1}{m!}\mathbb{E}_{t_1<\dots<t_m}[\dot{h}(t_1)\otimes\cdots\dot{h}(t_m)] \in \mathcal{H}^{\otimes m}.$$

2. The order statistic for the uniform distribution $[0,1]$ of m points has a density $p(t_1,\dots,t_m) = m!1_{t_1<\cdots t_m<1}$

Hence, the signature $\mathrm{S}(h)$ of a path h characterises the sequential structure in h by quantifying dependencies in their change, similar to classic sample moments in $\mathbb{R}^d$. In fact, signatures are in close mathematical analogy to polynomials: the feature map $x \mapsto \mathrm{S}(h)$ is injective and if $h \mapsto f(h)$ is a *non-linear function on pathspace*, then it can be well approximated as a *linear function* of the signature features $\mathrm{S}(h)$, that is $f(h) \approx \langle \ell, \mathrm{S}(h)\rangle$. However, signature features have the following drawbacks

- Even if $\mathcal{H} = \mathbb{R}^d$ is finite dimensional, we have $\mathrm{S}_m(h) \in \left(\mathbb{R}^d\right)^{\otimes m}$, thus the number of (scalar) coordinates needed to store $\mathrm{S}_m(h)$ grows as $O(d^m)$ as m increases. This combinatorial explosion makes the use of signature features costly for paths that evolve in high-dimensional state spaces $\mathcal{H}$ and infeasible for paths evolving in infinite dimensional state spaces $\mathcal{H}$.
- Signatures are only defined for paths evolving in linear spaces; $\mathrm{S}(h)$ does not make sense when h evolves in a general topological space $\mathcal{X}$ since $\dot{h}(t)dt = dh(t)$ is not well-defined.

However, as we will show, both of these shortcomings can be addressed by kernelization.

1.3. Inner products of signature features

The space $\prod_{m\geq 0}\mathcal{H}^{\otimes m}$ is a linear space and by restricting to a subspace, we can work with an inner product space (in fact, again a Hilbert space after completion); that is we define for $(s_m), (t_m) \in \prod_{m\geq 0}\mathcal{H}^{\otimes m}$

$$\begin{aligned}(s_0, s_1, \dots) + (t_0, t_1, \dots) &:= (s_0 + t_0, s_1 + t_1, \dots),\\ \langle (s_0, s_1, \dots), (t_0, t_1, \dots)\rangle &:= \sum_{m\geq 0}\langle s_m, t_m\rangle_{\mathcal{H}^{\otimes m}}\end{aligned}$$

where $\langle\cdot,\cdot\rangle_{\mathcal{H}^{\otimes m}}$ is given as $\langle u_1\otimes\cdots\otimes u_m, v_1\otimes\cdots\otimes v_m\rangle_{\mathcal{H}^{\otimes m}} := \prod_{i=1}^m\langle u_i, v_i\rangle_\mathcal{H}$. The inner product $\langle \mathrm{S}(g), \mathrm{S}(h)\rangle$ of the signature features of (sufficiently regular) paths $g, h \in \mathcal{P}_\mathcal{H}$ is finite and using conditional expectations we get that $\langle \mathrm{S}(g), \mathrm{S}(h)\rangle$ equals

$$\begin{aligned}&\sum_{m\geq 0}\frac{1}{m!^2}\mathbb{E}_{\substack{s_1<\dots<s_m\\ t_1<\dots<t_m}}[\langle \dot{g}(s_1)\otimes\cdots\otimes\dot{g}(s_m), \dot{h}(t_1)\otimes\cdots\otimes\dot{h}(t_m)\rangle_{\mathcal{H}^{\otimes m}}]\\ =&\sum_{m\geq 0}\frac{1}{m!^2}\mathbb{E}_{\substack{s_1<\dots<s_m\\ t_1<\dots<t_m}}[\langle \dot{g}(s_1), \dot{h}(t_1)\rangle_\mathcal{H}\cdots\langle \dot{g}(s_m), \dot{h}(t_m)\rangle_\mathcal{H}]\\ =&\left(1 + \mathbb{E}_{s_1,t_1}\left[\langle \dot{g}(s_1), \dot{h}(t_1)\rangle\cdot\left(1 + \frac{1}{2^2}\mathbb{E}_{\substack{s_2|s_1\\ t_2|t_1}}\left[\langle \dot{g}(s_2), \dot{h}(t_2)\rangle\cdot\left(1 + \frac{1}{3^2}\mathbb{E}_{\substack{s_3|s_2\\ t_3|t_2}}[\dots]\right)\right]\right)\right]\right).\end{aligned}$$

Now applied to $g = \mathrm{k}_x \in \mathcal{P}_\mathcal{H}$, $h = \mathrm{k}_y \in \mathcal{P}_\mathcal{H}$, using the reproducing property of k we get

$$\langle \dot{\mathrm{k}}_{x(s)}, \dot{\mathrm{k}}_{y(t)}\rangle_\mathcal{H}\,\mathrm{d}s\,\mathrm{d}t = \langle \mathrm{d}g(s), \mathrm{d}h(s)\rangle_\mathcal{H} = \mathrm{d}_s\,\mathrm{d}_t\,\mathrm{k}(x(s), y(t)) = \kappa(\mathrm{d}s, \mathrm{d}t)$$

where κ denotes the measure on $[0,1]^2$ defined as

$$\kappa([s,t]\times[u,v]) = \mathrm{k}(x(t), y(v)) - \mathrm{k}(x(s), y(v)) - \mathrm{k}(x(t), y(u)) + \mathrm{k}(x(s), y(u)).$$

(Note that this measure does not use differentiability of x, y). Replacing above expectations over the order statistic by explicit integration over $[0,1]$, we arrive at

$$\mathrm{k}^{\oplus}(x,y) = 1 + \int_{s_1,t_1} \left[\kappa(\mathrm{d}s_1, \mathrm{d}t_1) \cdot \left(1 + \int_{\substack{s_2>s_1 \\ t_2>t_1}} \left[\kappa(\mathrm{d}s_2, \mathrm{d}t_2) \cdot \left(1 + \int_{\substack{s_3>s_2 \\ t_3>t_2}} [\dots] \right) \right] \right) \right] . \quad (1)$$

Finally, the data we have access is to is only a sequence $(x(t_1), \dots, x(t_n)) \in \mathcal{X}^+$ rather than the whole path $x = (x(t))_{t\in[0,1]} \in \mathcal{P}_\mathcal{X}$, but replacing the integrals by sums immediately gives an explicit recursive formula for a kernel $\mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}$ that is determined by a Horner-type[3] recursive evaluation; especially, one never computes the signature map of a path that evolves in the high/infinite-dimensional Hilber space $\mathcal{H}$ directly. This yields a robust and computationally effective formula for a kernel k^+ for sequences, that has well-defined behaviour in the scaling limit when sequences approximate paths. We make the above informal derivation rigorous in the following sections, study its properties and extended it to non-smooth paths in Appendix B.

1.4. Prior art and literature review

Learning with sequential data is a vast field. Prior art that inspired our present approach can be found in

(i) dynamic programming algorithms for sequence comparison in the engineering community,

(ii) kernel and Gaussian processes for sequences in the machine learning community,

(iii) Rough paths in the stochastic process community.

Beyond the above, we are not aware of literature in statistics that deals with sequence-valued data points in a way other than first identifying one-dimensional sequences with real vectors of same size, or even forgetting the sequence structure entirely and replacing the sequences with (order-agnostic) aggregates such as cumulants, quantiles or principal component scores. Below we discuss above three points in more detail:

(i) Dynamic programming algorithms for sequence comparison.

The earliest use of order information in sequences for learning can probably be found in Sakoe and Chiba (1970) and Sakoe (1979) by using editing or distortion distances. These distances are then employed for classification by maximum similarity/minimum distance principles. Through theoretical appeal and efficient computability, sequence comparison methods, later synonymously called dynamic time warping methods, have become one of the standard methods in comparing sequential data Kruskal (1983); Giorgino (2009). Sequence comparison methods in their original formulation can only be directly applied to relatively simple distance-based learning algorithms. This has been addressed by combining such methods with kernel learning (discussed below).

3. Horner's scheme to evaluate a uni-variate polynomial $p(x) = \sum_{i=0}^m a_i x^i$ is to write $p(x) = a_0 + x\,(a_1 + x\,(a_2 + \cdots + x\,(a_{m-1} + x a_m)))$ and compute the brackets from the inside to the outside. This needs only n additions and n multiplications which is optimal and in contrast to the n additions and $\frac{n^2+n}{2}$ multiplications needed for the naive evaluation of $p(x)$; further, it is numerically more stable since one never adds floats that live on different scales (such as $a_1 x$ and $a_m x^m$). This structural similarity with formula (1) will be the key to our algorithms.

(ii) Kernels and Gaussian processes for sequences.

Kernel learning is one of the most popular approaches to make non-linear data of arbitrary kind amenable to classic and scalable linear algorithms and it provides learning theoretical guarantees, see Schölkopf and Smola (2002); Shawe-Taylor and Cristianini (2004); Cucker and Smale (2002). Kernels for strings, that is, sequences of symbols, were among the first to be considered by Haussler (1999) and fast algorithms were obtained a few years later Lohdi et al. (2002); Leslie and Kuang (2004). Kernels based on the above-mentioned dynamic programming (time warping) approach were developed, for sequences of arbitrary objects leading to so-called alignment kernel, see Bahlmann et al. (2002); Noma (2002); Cuturi et al. (2007); Cuturi (2011). All of the mentioned kernels can be viewed from Haussler's original, visionary relation-convolution kernel framework, Haussler (1999), the existing literature provides no unifying approach to kernels for sequences: for example the relation between string kernels and dynamic time warping/global alignment kernels, or to the classical theory of time series has remained unclear. Further, the only known kernels for sequences of arbitrary objects, the dynamic time warping/global alignment kernels, are in general not positive definite.

(iii) Rough paths and stochastic processes.

Series of iterated integrals appear in diverse areas such as control theory, combinatorics, homotopy theory, physics (Feynman–Dyson–Schwinger theory) and more recently probability theory; see (Lyons, 2004, Section "Historic papers", p97). This series is treated under various names in the literature like "Magnus expansion", "time ordered" or "non-commutative exponential", or the one we follow, namely "the signature of a path". Signatures have found many applications in stochastic analysis (rough path theory, regularity structures), but their use in statistics and machine learning is very recent: Papavasiliou et al. (2011) applies it to SDE parameter estimation; Levin et al. (2013); Lyons et al. (2014) applies it to forecasting and classification; Yang et al. (2015) uses signature features as input to deep neural nets. So far, all applications are restricted to paths evolving in low-dimensional linear spaces due to the computational bottleneck given by the combinatorial explosion of signature coordinates.

1.5. Outline

Section 2 introduces the signature as a canonical feature map for path-valued data. Section 3 shows that this signature can be kernelized to get a kernel $\mathrm{k}^{\oplus}$ for paths and studies its properties. Section 4 studies the discretization to turn $\mathrm{k}^{\oplus}$ into a kernel k^+ for sequences. Section 5 shows that classic kernels for sequences arise as special cases of our construction. Section 6 presents effective algorithms for computing k^+ using above recursion with low-rank techniques. Section 7 presents a Numpy implementation and basic benchmarks.

1.6. Notation

Spaces, sequences and paths	
$\mathcal{X}$	an arbitrary set
$\mathcal{X}^+$	$\mathcal{X}^+ := \bigcup_{\ell \geq 1} \mathcal{X}^\ell$ sequences of arbitrary length in $\mathcal{X}$
$\mathcal{P}_\mathcal{X}$	a set of continuous paths from $[0,1]$ to $\mathcal{X}$
$\mathcal{H}$	the reproducing kernel Hilbert space, $\mathcal{H} \subset \mathbb{R}^\mathcal{X}$, of the kernel k
$\mathcal{H}^+$	$\mathcal{H}^+ := \bigcup_{\ell \geq 1} \mathcal{H}^\ell$ sequences of arbitrary length in $\mathcal{H}^+$
$\mathcal{P}_\mathcal{H}$	a set of continuous bounded variation paths from $[0,1]$ to $\mathcal{H}$
Signature features for paths and sequences in $\mathcal{H}$	
$\prod_{m\geq 0} \mathcal{H}^{\otimes m}$	the linear space of series of tensors in $\mathcal{H}$
S	the signature map $\mathrm{S} : \mathcal{P}_\mathcal{H} \to \prod_{m\geq 0} \mathcal{H}^{\otimes m}$
S^+	the discretized signature map $\mathrm{S}^+ : \mathcal{H}^+ \to \prod_{m\geq 0} \mathcal{H}^{\otimes m}$
$\langle \cdot, \cdot \rangle$	inner product on the subset of square-summable elements of $\prod_{m\geq 0} \mathcal{H}^{\otimes m}$
$\| \cdot \|$	norm given by $\langle \cdot, \cdot \rangle$ for square summable elements of $\prod_{m\geq 0} \mathcal{H}^{\otimes m}$
Kernels	
k	a kernel $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$
k_x with $x \in \mathcal{X}$	the element $\mathrm{k}_x := \mathrm{k}(x, \cdot) \in \mathcal{H} \subset \mathbb{R}^\mathcal{X}$
k_x with $x \in \mathcal{X}^+$	the sequence $\mathrm{k}_x := (\mathrm{k}_{x_i})_{i=1,\dots,\ell} \in \mathcal{H}^+$ with $x = (x_i)_{i=1,\dots,l}$
k_x with $x \in \mathcal{P}_\mathcal{X}$	the path $\mathrm{k}_x := (\mathrm{k}_{x(t)})_{t\in[0,1]} \in \mathcal{P}_\mathcal{H}$ with $x = (x(t))_{t\in[0,1]}$
k^+	a kernel $\mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}$, $\mathrm{k}^+(x,y) := \langle \mathrm{S}^+(\mathrm{k}_x), \mathrm{S}^+(\mathrm{k}_y) \rangle$
$\mathrm{k}^\oplus$	a kernel $\mathrm{k}^\oplus : \mathcal{P}_\mathcal{X} \times \mathcal{P}_\mathcal{X} \to \mathbb{R}$, $\mathrm{k}^\oplus(x,y) := \langle \mathrm{S}(\mathrm{k}_x), \mathrm{S}(\mathrm{k}_y) \rangle$
Partitions of $[0,1]$	
π	a partition of $[0,1]$, i.e. a collection of points $0 = t_1 < \cdots < t_l = 1$
$\mathrm{mesh}(\pi)$	$\mathrm{mesh}(\pi) := \max_{i=1,\dots,l-1} \lvert t_{i+1} - t_i \rvert$
x^π for $x \in \mathcal{P}_\mathcal{X}$	$x^\pi := (x(t_i))_{i=1,\dots,l} \in \mathcal{X}^+$, the path x sampled along π

1.7. Author's contribution

Both authors contributed equally, HO is the corresponding author.

2. The signature feature map

Mapping paths to series of tensors is well-known technique in stochastic analysis. In this section we give a short introduction and highlight aspects of the signature map that make it a good feature map for paths.

2.1. Bounded variation paths

Definition 2.1 *Let $\mathcal{H}$ a Hilbert space. We denote the set of $\mathcal{H}$-valued paths of bounded variation on $[0,1]$ starting at the origin by*

$$C^1([0,1],\mathcal{H}) := \{h \in C([0,1],\mathcal{H}) \,:\, h(0) = 0, \quad \|h\|_1 < \infty\}$$

where $\|h\|_1 = \sup_\pi \sum_{i=1}^{l-1} \|h(t_{i+1}) - h(t_i)\|_\mathcal{H}$ and the supremum is taken over all finite partitions $\pi = \{(t_i)_{i=1,\dots,l} : 0 = t_1 < \cdots < t_l = 1\}$ of $[0,1]$.

As in the finite-dimensional case, the integral $\int_0^1 y(t)\,\mathrm{d}h(t) \in \mathcal{H}$ is defined as the limit of Riemann–Stieltjes sums, see Appendix A for details. The integral itself is in general not a scalar, but an element of the Hilbert space, $\int y\,\mathrm{d}h \in \mathcal{H}$.

Definition 2.2 *We define the sequence $\left(\int \mathrm{d}h^{\otimes m}\right)_{m\geq 0} \in \prod_{m\geq 0} \mathcal{H}^{\otimes m}$ as*[4]

$$\int \mathrm{d}h^{\otimes 0} := 1 \text{ and } \int \mathrm{d}h^{\otimes m} := \int_0^1 \left(\int_0^t \mathrm{d}h^{\otimes (m-1)} \right) \otimes \mathrm{d}h(t).$$

We define

$$\mathrm{S} : C^1([0,1],\mathcal{H}) \to \prod_{m\geq 0} \mathcal{H}^{\otimes m}, \; h \mapsto \left(\int \mathrm{d}h^{\otimes m} \right)_{m\geq 0}$$

and call $\mathrm{S}(h)$ the signature of the path h and denote $\mathrm{S}_m(h) = \int \mathrm{d}h^{\otimes m}$.

Example 2.3 *Let $\mathcal{H} = \mathbb{R}^2$ and $h(t) = (a(t), b(t))^\top$. The first levels of the signature of h are*

$$\mathrm{S}_0(h) = 1, \quad \mathrm{S}_1(h) = \begin{pmatrix} \int_0^1 \mathrm{d}a(t) \\ \int_0^1 \mathrm{d}b(t) \end{pmatrix}, \quad \mathrm{S}_2(h) = \begin{pmatrix} \int_0^1 \int_0^{t_2} \mathrm{d}a(t_1)\,\mathrm{d}a(t_2) & \int_0^1 \int_0^{t_2} \mathrm{d}a(t_1)\,\mathrm{d}b(t_2) \\ \int_0^1 \int_0^{t_2} \mathrm{d}b(t_1)\,\mathrm{d}a(t_2) & \int_0^1 \int_0^{t_2} \mathrm{d}b(t_1)\,\mathrm{d}b(t_2) \end{pmatrix}.$$

The degree $m = 3$, $\mathrm{S}_3(h) \in \left(\mathbb{R}^2\right)^{\otimes 3}$, has 2^3-scalar valued coordinates, etc. In general, for a path in $\mathbb{R}^d$ we need d^m scalars to describe $\int \mathrm{d}h^{\otimes m}$ and it is exactly this combinatorial explosion that makes the use of signatures prohibitively expensive already for moderately high-dimensional $\mathcal{H}$ and impossible to use for infinite-dimensional $\mathcal{H}$ (such as the RKHS associated with most kernels relevant in machine learning).

Example 2.4 *For the special case of a linear path in $\mathcal{H} = \mathbb{R}^d$, i.e. $h(t) = t \cdot v$ for a fixed vector $v \in \mathbb{R}^d$, a direct calculation shows that*

$$\mathrm{S}_m(h) = \frac{v^{\otimes m}}{m!} = \frac{h(1)^{\otimes m}}{m!} \text{ for } m \geq 0$$

or in more concise notation[5]*, $\mathrm{S}(h) = \exp(v) = \exp(h(1) - h(0))$. This explains why the signature $\mathrm{S}(h)$ is also known as time-ordered exponential. This analogy with the exponential function will be key motivation for the estimates in Section 4.*

4. Recall that by convention $\mathcal{H}^{\otimes 0} := \mathbb{R}$.

5. where one denotes $\boldsymbol{t} = (t_m)_m \in \prod_{m\geq 0} \mathcal{H}^{\otimes m}$ as $\boldsymbol{t} = t_0 + t_1 + t_2 + \cdots$, set $\exp(\boldsymbol{t}) = \sum_{m\geq 0} \frac{\boldsymbol{t}^{\otimes m}}{m!}$ and identify $v \in \mathbb{R}^d$ as $\boldsymbol{t} = 0 + v + 0 + 0 + \cdots$.

2.2. The signature as a canonical feature map

The signature $\mathrm{S}(h)$ describes a path h by an element in the linear space $\prod_{m\geq 0}\mathcal{H}^{\otimes m}$ and the following properties make it a good feature map

Theorem 1

1. *The map $h \mapsto \mathrm{S}(h)$ is continuous,*
2. *The map $h \mapsto \mathrm{S}(h)$ is injective up to tree-like equivalence*[6]*,*
3. *For any $f \in C(K,\mathbb{R})$, $K \subset C^1([0,1],\mathcal{H})$ compact, and $\epsilon > 0$ there exists a $\ell \in \bigoplus_{m\geq 0}\mathcal{H}^{\otimes m}$ such that*

$$\sup_{h\in K} |f(h) - \langle \ell, \mathrm{S}(h)\rangle| < \epsilon.$$

The proof is classic in stochastic analysis and given in Appendix A; the key insight is that the space of linear combinations of iterated integrals forms an algebra. To sum up

1. the signature features are a *mathematically faithful representation of the underlying path-valued data h.*
2. *linear combinations* of signature coordinates *approximate continuous functions of paths* arbitrarily well. In the kernel learning context this is known as "universality".
3. the signature features are *sequentially ordered analogues to moments.* Similar to polynomials, the feature space has the natural grading by m designating the "polynomial degree".

3. Signature kernels

We come back to our original motivation: we are given a RKHS $(\mathcal{H},\mathrm{k})$ on $\mathcal{X}$ and we want to transform this into a RKHS for paths in $\mathcal{X}$. Therefore recall that the reproducing property implies that

$$\mathrm{k}:\mathcal{X}\times\mathcal{X}\to\mathbb{R} \text{ equals } \mathrm{k}(u,v) = \langle \mathrm{k}_u, \mathrm{k}_v\rangle_{\mathcal{H}}$$

and that one may think of $\mathcal{X}\ni u \mapsto \mathrm{k}_u(\cdot) := \mathrm{k}(u,\cdot)\in\mathcal{H}$ as the (canonical) feature map induced by the kernel k. Now given a path $x\in\mathcal{P}_{\mathcal{X}}$ we lift it to a path[7] $\mathrm{k}_x\in\mathcal{P}_{\mathcal{H}}$ given as

6. We call h, h' tree-like equivalent if the concatenation $h\sqcup h'^{-1}$ is tree-like. A path $z\in C^1([a,b],\mathcal{H})$ is called tree-like if there exists a continuous map $f:[a,b]\to[0,\infty)$ with $f(0)=f(T)=0$ and $|z(t)-z(s)|\leq f(s)+f(t)-2\inf_{u\in[s,t]} f(u)$ for all $s\leq t\in[a,b]$. The key remark is that being tree-like equivalent is a very strong assumption, typically not seen in real data. Even if presented with a tree-like path, adding time as extra coordinate transforms the path to a non-treelike path (that is, working with $t\mapsto(t,h(t))$ instead of $t\mapsto h(t)$).
7. Recall the slight clash of notation $\mathrm{k}_u\in\mathcal{H}$ for $u\in\mathcal{X}$ and $\mathrm{k}_x\in\mathcal{P}_{\mathcal{H}}$ if $x\in\mathcal{P}_{\mathcal{X}}$, but the meaning will always be clear from the context. In fact, later we are also going to use $\mathrm{k}_x\in\mathcal{H}^+$ if $x\in\mathcal{X}^+$.

$t\mapsto \mathrm{k}_x(t) := \mathrm{k}_{x(t)}(\cdot) := \mathrm{k}(x(t),\cdot)$. The results of Section 2 suggest $\mathrm{S}(\mathrm{k}_x)$ as features for k_x, that is the feature map

$$\mathcal{P}_{\mathcal{X}}\ni x\mapsto \mathrm{S}(\mathrm{k}_x)\in\prod_{m\geq 0}\mathcal{H}^{\otimes m}. \tag{2}$$

Theorem 1 guarantees that $\mathrm{S}(\mathrm{k}_x)$ captures all the information about $x\in\mathcal{P}_{\mathcal{X}}$ that can be obtained by looking at the path x through the kernel k.

Most RKHS $\mathcal{H}$ in machine learning are infinite dimensional which makes this approach infeasible; even for finite dimensional $\mathcal{H}$ we suffer from the combinatorial explosion of the number of signature coordinates, Example 2.3. The main result of this section is that the feature map (2) can be complety kernelized, that is by defining

$$(x,y)\mapsto \mathrm{k}^{\oplus}(x,y) = \langle \mathrm{S}(\mathrm{k}_x), \mathrm{S}(\mathrm{k}_y)\rangle$$

we get a kernel for paths in $\mathcal{X}$ that can be very efficiently evaluated using only $\mathrm{k}(x(s),y(t))$ for $s,t\in[0,1]$. That is, only evaluations of the "static" kernel $\mathrm{k}:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$.

We henceforth restrict attention to positive definite kernels on $\mathcal{X}$ and a set of paths $\mathcal{P}_{\mathcal{X}}$ that lift to bounded variation paths. All our arguments extend to unbounded variation such as semi-martingales, diffusion processes, Markov processes or Gaussian processes and we give the needed modifications in Appendix B.

Assumption 3.1 *Throughout the remainder of this article we assume that $\mathrm{k}:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$ is continuous and positive definite. We denote with $\mathcal{H}$ the associated RKHS. Further, we denote with $\mathcal{P}_{\mathcal{X}}\subset\mathcal{X}^{[0,1]}$ a set of paths in a topological space $\mathcal{X}$ and $(\mathcal{H},\mathrm{k})$ a RKHS on $\mathcal{X}$ such that*

$$\{t\mapsto \mathrm{k}_{x(t)} : x\in\mathcal{P}_{\mathcal{X}}\}\subset C^1([0,1],\mathcal{H}).$$

This yields the following kernel,

Definition 3.2 *We call*

$$\mathrm{k}^{\oplus}:\mathcal{P}_{\mathcal{X}}\times\mathcal{P}_{\mathcal{X}}\to\mathbb{R}\,,\quad (x,y)\mapsto\langle \mathrm{S}(\mathrm{k}_x),\mathrm{S}(\mathrm{k}_y)\rangle,$$

the signature kernel *over* k.

Theorem 2 *The signature kernel over* k

$$\mathrm{k}^{\oplus}:\mathcal{P}_{\mathcal{X}}\times\mathcal{P}_{\mathcal{X}}\to\mathbb{R},\quad \mathrm{k}^{\oplus}(x,y)=\langle \mathrm{S}(\mathrm{k}_x),\mathrm{S}(\mathrm{k}_x)\rangle$$

1. *is a positive definite kernel,*
2. *$\mathrm{k}^{\oplus}(x,y)=\sum_{m\geq 0}\int_{\substack{s_1<\cdots<s_m\\ t_1<\cdots<t_m}}\prod_{i=1}^m \mathrm{d}\kappa(s_i,t_i)$ with κ defined as*

$$\kappa([s,t]\times[u,v]) := \mathrm{k}(x(t),y(v)) - \mathrm{k}(x(s),y(v)) - \mathrm{k}(x(t),y(u)) + \mathrm{k}(x(s),y(u))$$

and κ is a signed Borel measure on $[0,1]^2$ for every $x,y\in\mathcal{P}_{\mathcal{X}}$.

Proof Point (1) follows since $\mathrm{k}^{\oplus}$ is given as an inner product, see Schölkopf and Smola (2002). Using the definition of $\mathrm{k}^{\oplus}$ we calculate

$$\begin{aligned}
\mathrm{k}^{\oplus}(x,y) &= \langle \mathrm{S}(\mathrm{k}_x), \mathrm{S}(\mathrm{k}_y)\rangle = \sum_{m\geq 0} \langle \int \mathrm{dk}_x^{\otimes m}, \int \mathrm{dk}_y^{\otimes m}\rangle_{\mathcal{H}^{\otimes m}} \\
&= \sum_{m\geq 0} \langle \int_0^1 \left(\int_0^{s_{m-1}} \mathrm{dk}_x^{\otimes (m-1)}\right) \otimes \mathrm{dk}_{x(s_m)}, \int_0^1 \left(\int_0^{t_{m-1}} \mathrm{dk}_y^{\otimes (m-1)}\right) \otimes \mathrm{dk}_{y(t_m)}\rangle \\
&= 1 + \sum_{m\geq 1} \int_{\substack{s_m\in[0,1]\\ t_m\in[0,1]}} \langle \int_0^{s_m} \mathrm{dk}_x^{\otimes (m)}, \int_0^{t_{m-1}} \mathrm{dk}_y^{\otimes (m-1)}\rangle \langle \mathrm{dk}_{x(s_m)}, \mathrm{dk}_{y(t_m)}\rangle \\
&= \vdots \\
&= 1 + \sum_{m\geq 1} \int_{\substack{s_m\in[0,1]\\ t_m\in[0,1]}} \int_{\substack{s_{m-1}\in[0,s_m]\\ t_{m-1}\in[0,t_m]}} \cdots \int_{\substack{s_1\in[0,s_2]\\ t_1\in[0,t_2]}} \langle \mathrm{dk}_{x(s_1)}, \mathrm{dk}_{y(t_1)}\rangle \cdots \langle \mathrm{dk}_{x(s_m)}, \mathrm{dk}_{y(t_m)}\rangle.
\end{aligned} \tag{3}$$

Now consider two paths x, y that are piecewise linear, that is there are time points $s_1 \leq \cdots \leq s_k$ such that $\dot{x}$ is constant on intervals (s_i, s_{i+1}) and analogous for y there exists $t_1 \leq \cdots \leq t_l$ such that $\dot{y}$ is constant on all intervals (t_i, t_{i+1}). Then

$$\begin{aligned}
&\int_{(s_1,s_k)\times(t_1,t_l)} \langle \mathrm{dk}_{x(s)}, \mathrm{dk}_{y(t)}\rangle_{\mathcal{H}} \\
&= \sum_{\substack{1\leq i\leq k-1\\ 1\leq j\leq l-1}} \int_{(s_i,s_{i+1})\times(t_j,t_{j+1})} \langle \mathrm{dk}_{x(s)}, \mathrm{dk}_{y(t)}\rangle_{\mathcal{H}} \\
&= \sum_{\substack{1\leq i\leq k-1\\ 1\leq j\leq l-1}} \langle \mathrm{k}_{x(s_{i+1})} - \mathrm{k}_{x(s_i)}, \mathrm{k}_{y(t_{j+1})} - \mathrm{k}_{y(t_j)}\rangle_{\mathcal{H}} \\
&= \sum_{\substack{1\leq i\leq k-1\\ 1\leq j\leq l-1}} \mathrm{k}(x(s_{i+1}), y(t_{j+1})) - \mathrm{k}(x(s_i), y(t_{j+1})) - \mathrm{k}(x(s_{i+1}), y(t_j)) + \mathrm{k}(x(s_i), y(t_j))
\end{aligned}$$

where we used the reproducing property $\langle \mathrm{k}_{x(s)}, \mathrm{k}_{y(t)}\rangle = \mathrm{k}(x(s), y(t))$ for the last equality. Note that

$$\kappa([r,s]\times[u,v]) := \mathrm{k}(x(s_{i+1}), y(t_{j+1})) - \mathrm{k}(x(s_i), y(t_{j+1})) - \mathrm{k}(x(s_{i+1}), y(t_j)) + \mathrm{k}(x(s_i), y(t_j))$$

defines a signed measure on $[0,1]^2$, hence above reads

$$\int_{(s_1,s_k)\times(t_1,t_l)} \langle \mathrm{dk}_{x(s)}, \mathrm{dk}_{y(t)}\rangle_{\mathcal{H}} = \sum_{\substack{1\leq i\leq k\\ 1\leq j\leq l}} \kappa([s_1,s_k]\times[t_1,t_l]) = \int_{(s_1,s_k)\times(t_1,t_l)} \mathrm{d}\kappa_{x,y}(s,t).$$

Combined with the identity (3) this shows (2) for piecewise constant paths. For general x, y fix a sequence $\{(s_i^n) : 0 = s_1 < \cdots < s_{l+1}^n = 1\} \subset [0,1]^n$ such that $\max_i |s_{i+1}^n - s_i^n| \to 0$

as $n \to \infty$ and denote with x^n, resp. y^n, the piecewise linear path given by interpolating between points $(x(s_i^n))_i$, resp. $(y(s_j^n))_j$. Then

$$\kappa_{x^n,y^n}([r,s]\times[u,v]) \to \kappa_{x,y}([r,s]\times[u,v])$$

for all $r < s, u < v$ directly by the definition of κ. Thus κ_{x^n,y^n} converges weakly to $\kappa_{x,y}$ On the other hand, $\langle \mathrm{S}(x^n), \mathrm{S}(y^n)\rangle$ converges to $\langle \mathrm{S}(x), \mathrm{S}(y)\rangle$ which finishes the proof by sending $n \to \infty$ ■

The equality in Point (2) in Theorem 2 suggests the expansion

$$\mathrm{k}^{\oplus}(x,y) = 1 + \int_{(s_1,t_1)\in(0,1)\times(0,1)} \left(1 + \int_{(s_2,t_2)\in(0,s_1)\times(0,t_1)} (1+\cdots)\,\mathrm{d}\kappa_{x,y}(s_2,t_2)\right) \mathrm{d}\kappa_{x,y}(s_1,t_1). \tag{4}$$

This recursive structure will be the key for an efficient computation.

4. Discretization and recursion: from paths to sequences

To make the kernel $\mathrm{k}^{\oplus} : \mathcal{P}_{\mathcal{X}} \times \mathcal{P}_{\mathcal{X}} \to \mathbb{R}$ useful for real-world applications, we need to address the following two points:

1. (Sequences). We do not observe the full path $(x(t))_{t\in[0,1]} \in \mathcal{P}_{\mathcal{X}}$ but only a sequence $x^{\pi} = (x(t_i))_{i=1}^{l} \in \mathcal{X}^+$ for a partition $\pi = \{0 = t_1 < \cdots < t_\ell = 1\}$. The partition π might even vary from path to path.
2. (Computational cost). Computing the kernel $\mathrm{k}^{\oplus} : \mathcal{P}_{\mathcal{X}} \times \mathcal{P}_{\mathcal{X}} \to \mathbb{R}$ by evaluating it as an inner product is computationally prohibitive even with $\mathcal{X} = \mathbb{R}^d = \mathcal{H}$ and the linear kernel $\mathrm{k}(\cdot,\cdot) = \langle \cdot, \cdot\rangle_{\mathbb{R}^d}$ is used if d is large (Example 2.3). It is infeasible for standard kernels such as the Gaussian kernel that have an infinite-dimensional RKHS.

To address point (1), we approximate the signature map

$$x \mapsto \mathrm{S}(\mathrm{k}_x) = \left(\int \mathrm{dk}_x^{\otimes m}\right)_{m\geq 0}$$

by discretization of the integrals. This gives a feature map S^+ for sequences

$$x^{\pi} \mapsto \mathrm{S}^+(\mathrm{k}_{x^{\pi}}) = \left(\sum_{1\leq i_1<\cdots<i_m\leq \ell-1} \nabla_{i_1}\mathrm{k}_x \otimes \cdots \otimes \nabla_{i_m}\mathrm{k}_x\right)_{m\geq 0}$$

where we use the notation $\nabla_i \mathrm{k}_x := \mathrm{k}_{x(t_{i+1})} - \mathrm{k}_{x(t_i)}$ and $\mathrm{k}_{x^{\pi}} = (\mathrm{k}_{x(t_i)})_{i=1}^{l} \in \mathcal{H}^+$. A simple calculation shows that this is equivalent to

$$x^{\pi} \mapsto \mathrm{S}^+(\mathrm{k}_{x^{\pi}}) = \prod_{i=1}^{\ell} (1 + \nabla_i \mathrm{k}_{x_i^{\pi}})$$

if $\prod$ denotes the linear extension[8] of the tensor product to the linear space $\prod_{m\geq 0}\mathcal{H}^{\otimes m}$ and this is the definition we are going to use. The kernel k^+ is defined as the inner product of this feature map

$$\mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}, \quad (x,y) \mapsto \langle \mathrm{S}^+(\mathrm{k}_x), \mathrm{S}^+(\mathrm{k}_y)\rangle$$

To address Point (2), we make heavy use of the recursive expansion (4) for $\mathrm{k}^{\oplus}$ and truncate after a given degree m (that is usually determined by cross-validation to avoid overfitting in analogy to the classic polynomial kernel).

4.1. Discretized signatures

Definition 4.1 *We call the map*

$$\mathrm{S}^+ : \mathcal{H}^+ \to \prod_{m\geq 0}\mathcal{H}^{\otimes m}, \quad (h_i)_{i=1}^l \mapsto \prod_{i=1}^{l}(1+\nabla_i h)$$

the discretized signature map (of order 1*).*

To prove quantitative results about the quality of this discretization, we need to understand how fast $\mathrm{S}^+(\mathrm{k}_{x^\pi})$ approximates $\mathrm{S}(\mathrm{k}_x)$ as $\mathrm{mesh}(\pi) := \max_i |t_{i+1}-t_i| \to 0$. We first study this approximation of paths by sequences for general paths $h \in \mathcal{P}_{\mathcal{H}}$.

Definition 4.2 *Define for* $d \geq 1$

$$\exp_d : \mathbb{R}^d \to \mathbb{R}, \quad v = (v_i)_{i=1}^d \mapsto \prod_{i=1}^{d}(1+v_i)$$

Define for a partition $\pi = (t_i)_{i=1}^{\ell}$ *with* $0 = t_1 < \cdots < t_\ell = 1$

$$\|h\|_{(\pi)} : C^1([0,1],\mathcal{H}) \to \mathbb{R}, \quad h \mapsto \left(\|h_{[t_1,t_2]}\|_1, \ldots, \|h_{[t_{\ell-1},t_\ell]}\|_1\right) \in \mathbb{R}^{\ell-1}$$

where $h_{[a,b]}$ *denotes the restriction of* $h \in C^1([0,1],\mathcal{H})$ *to* $C^1([a,b],\mathcal{H})$*.*

Using this notation, the main result of this section reads

Theorem 3 *For* $h \in C^1([0,1],\mathcal{H})$ *and a partition* $\pi = (t_i)_{i=1}^{\ell}$, $0 = t_1 < \cdots < t_\ell = 1$,

$$\left\|\mathrm{S}^+(h^\pi) - \mathrm{S}(h)\right\| \leq \exp\left(\|h\|_1\right) - \exp_1\left(\|h\|_{(\pi)}\right),$$

where $h^\pi = (h(t_i))_{i=1,\ldots,\ell} \in \mathcal{H}^+$*.*

As a consequence, we get explicit convergence rates.

8. For $\boldsymbol{s},\boldsymbol{t} \in \prod_{m\geq 0}\mathcal{H}^{\otimes m}$ with $s_0 = t_0 = 1$ define $\boldsymbol{s}\cdot\boldsymbol{t} \in \prod_{m\geq 0}\mathcal{H}^{\otimes m}$ as having the m-element equal to $\sum_{i=0}^m s_i t_{m-i}$. If we denote $\boldsymbol{s} = (s_0, s_1, \ldots)$ as $\boldsymbol{s} = s_0 + s_1 + \cdots$, then this reads $\boldsymbol{s}\cdot\boldsymbol{t} = 1 + s_1 + t_1 + s_2 + s_1 t_1 + t_2 + \cdots$ and $\prod(1+\nabla_i h)$ refers to this product. Informally speaking, we multiply series of tensors in the same way we multiply polynomials.

Corollary 4.3 *With the same notation as in Theorem 3 above, it holds that*

$$\left\|\mathrm{S}^+(h^\pi) - \mathrm{S}(h)\right\| \leq \|h\|_1 e^{\|h\|_1} \cdot \max_{i=1,\ldots,\ell-1}\|h_{[t_i,t_{i+1}]}\|_1.$$

Hence, the convergence $\lim_{\|h\|_{(\pi)}\to\|h\|_1}\mathrm{S}^+(h^\pi) = \mathrm{S}(h)$*, is of order* $O\left(\max_i\|h_{[t_i,t_{i+1}]}\|_1\right)$*.*

Corollary 4.4 *With the same notation as in Theorem 3 above, it holds that if* π *is chosen such that* $\|h_{[t_i,t_{i+1}]}\|_1 = \frac{\|h\|_1}{\ell-1}$ *for all* $i = 1,\ldots,\ell-1$*, then one has the asymptotically tight bound*

$$\left\|\mathrm{S}^+(h^\pi) - \mathrm{S}(h)\right\| \leq \frac{e^{\|h\|_1}}{\ell-1}\left(1 + \frac{\|h\|_1^{\ell-1}}{(\ell-3)!}\right).$$

The proofs of Theorem 3 and Corollaries 4.3 and 4.4 are given in Appendix A.

Remark 4.5 (About geometric approximations) *The reader familiar with path signatures will note that the approximations* $\mathrm{S}^+(h^\pi)$ *to* $\mathrm{S}(h)$ *are not group-like elements of the tensor (Hopf) algebra* $\prod_{m\geq 0}\mathcal{H}^{\otimes m}$*, thus "non-geometric" in the sense that there can not exist a path* $\hat{h}$ *such that* $\mathrm{S}^+(h^\pi) = \mathrm{S}(\hat{h})$*. However, as we will see in the following sections, from a computational perspective non-geometric approximations have many benefits. In fact, the "non-geometric" approximations of this section recover classic machine learning constructions, see Section 5. Nevertheless, one can modify the discussion of this section to produce geometric approximations and we carry this out in Appendix B; see also Remark B.6.*

4.2. Discretized signature kernels

Definition 4.6 *Given* $x = (x_i)_{i=1}^l \in \mathcal{X}^+$ *denote* $\mathrm{k}_x = (\mathrm{k}_{x_i})_{i=1}^l \in \mathcal{H}^+$*. We refer to*

$$\mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}, \quad \mathrm{k}^+(x,y) = \langle \mathrm{S}^+(\mathrm{k}_x), \mathrm{S}^+(\mathrm{k}_y)\rangle$$

as the discretized signature kernel (or sequentialization) over k *of order* 1*.*

Theorem 4 *The discretized signature kernel over* k*,*

$$\mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}, \quad \mathrm{k}^+(x,y) = \langle \mathrm{S}^+(\mathrm{k}_x), \mathrm{S}^+(\mathrm{k}_x)\rangle$$

1. *is a positive definite kernel,*
2. $\mathrm{k}^+(x,y) = \sum_{m\geq 0}\sum_{\substack{1\leq i_1<\cdots<i_m<|x|\\ 1\leq j_1<\cdots<j_m<|y|}}\prod_{r=1}^m \nabla_{i_r,j_r}\mathrm{k}(x,y)$*,*
3. $\mathrm{k}^+(x,y) = 1 + \sum_{\substack{i_1\geq 1\\ j_1\geq 1}}\nabla_{i_1,j_1}\mathrm{k}(x,y)\cdot\left(1 + \sum_{\substack{i_2>i_1\\ j_2>j_1}}\nabla_{i_2,j_2}\mathrm{k}(x,y)\cdot\left(1+\sum_{\substack{i_3>j_2\\ j_3>j_2}}\cdots\right)\right)$*,*

where we use the notation $x = (x_i)_{i=1}^{|x|}, y = (y_i)_{i=1}^{|y|} \in \mathcal{X}^+$ *and*

$$\nabla_{i,j}\mathrm{k}(x,y) := \mathrm{k}(x_{i+1},y_{j+1}) + \mathrm{k}(x_i,y_j) - \mathrm{k}(x_i,y_{j+1}) - \mathrm{k}(x_{i+1},y_j).$$

Proof The first point follows, since k^+ is an inner product. For the second point note that $\nabla_i \mathrm{k}_x \equiv \mathrm{k}_{x_{i+1}} - \mathrm{k}_{x_i}$, $\nabla_i \mathrm{k}_y \equiv \mathrm{k}_{y_{j+1}} - \mathrm{k}_{y_j}$ and that by the reproducing property $\langle \nabla_i \mathrm{k}_x, \nabla_j \mathrm{k}_y \rangle = \nabla_{i,j} \mathrm{k}(x,y)$. The identity then follows from multiplying out

$$\langle \mathrm{S}^+(\mathrm{k}_x), \mathrm{S}^+(\mathrm{k}_y) \rangle = \langle \prod_i (1 + \nabla_i \mathrm{k}_x), \prod_j (1 + \nabla_j \mathrm{k}_y) \rangle$$

and using the definition of the inner product on $\mathcal{H}^{\otimes m}$. The last point follows by rearranging the summation. ∎

We can now combine this with Theorem 3 to quantify how fast the kernel k^+ for sequences approximates the kernel $\mathrm{k}^\oplus$ for paths as the mesh of partitions gets finer.

Corollary 4.7 *Let $x, y \in \mathcal{P}_\mathcal{X}$, and $\pi = (s_i)_{i=1}^k$ with $0 = s_1 < \cdots < s_k = 1$ and $\rho = (t_j)_{j=1}^l$ with $0 = t_1 < \cdots < t_l = 1$. Then,*

$$\left| \mathrm{k}^+(x^\pi, y^\rho) - \mathrm{k}^\oplus(x,y) \right| \leq 4e^{\|x\|_1 + \|y\|_1} - 2e^{\|x\|_{(\pi)} + \|y\|_1} - 2e^{\|x\|_1 + \|y\|_{(\rho)}}.$$

In particular,

$$\lim_{\substack{\|x\|_{(\pi)} \to \|x\|_1 \\ \|y\|_{(\rho)} \to \|y\|_1}} \mathrm{k}^+(x^\pi, y^\rho) = \mathrm{k}^\oplus(x,y)$$

where convergence is uniform of order $O\left(\max_i \|x_{[s_i, s_{i+1}]}\|_1 + \max_j \|y_{[t_j, t_{j+1}]}\|_1\right)$.

Proof It holds that

$$\begin{aligned} \mathrm{k}^+(x^\pi, y^\rho) - \mathrm{k}^\oplus(x,y) =& \langle \mathrm{S}^+(\mathrm{k}_x^\pi), \mathrm{S}^+(\mathrm{k}_y^\rho) \rangle - \langle \mathrm{S}(\mathrm{k}_x), \mathrm{S}(\mathrm{k}_y) \rangle \\ =& \langle \mathrm{S}^+(\mathrm{k}_x^\pi), \mathrm{S}^+(\mathrm{k}_y^\rho)) - \mathrm{S}(\mathrm{k}_y) \rangle + \langle \mathrm{S}^+(\mathrm{k}_x^\pi) - \mathrm{S}(\mathrm{k}_x), \mathrm{S}(\mathrm{k}_y) \rangle. \end{aligned}$$

The Cauchy–Schwarz inequality implies that

$$|\langle \mathrm{S}^+(\mathrm{k}_x^\pi), \mathrm{S}^+(\mathrm{k}_y^\rho)) - \mathrm{S}(\mathrm{k}_y) \rangle| \leq \| \mathrm{S}^+(\mathrm{k}_x^\pi) \| \cdot \| \mathrm{S}^+(\mathrm{k}_y^\rho) - \mathrm{S}(\mathrm{k}_y) \|.$$

Theorem 3 implies

$$\| \mathrm{S}^+(\mathrm{k}_x^\pi) \| \cdot \| \mathrm{S}^+(\mathrm{k}_y^\rho)) - \mathrm{S}(\mathrm{k}_y) \| \leq 2 \exp(\|x\|_1) \cdot \left(\exp(\|y\|_1 - \exp(\|y\|_\rho) \right)$$

and similarly, one obtains

$$\| \langle \mathrm{S}^+(\mathrm{k}_y^\rho), \mathrm{S}^+(\mathrm{k}_x^\pi) - \mathrm{S}(\mathrm{k}_x) \rangle \| \leq 2 \exp(\|y\|_1) \cdot (\exp(\|x\|_1) - \exp(\|x\|_\pi).$$

Putting all (in-)equalities together yields the main claim, the convergence statement follows from Theorem 3 (ii). ∎

4.3. Truncation and variations on a theme

The final step to get a practically useful kernel for sequences is to truncate the discrete signature features $\mathrm{S}^+(\mathrm{k}_x) \in \prod_{m \geq 0} \mathcal{H}^{\otimes m}$ at a given degree $m \geq 1$. In analogy with the classic polynomial features (resp. kernel), this not only serves a computational purpose but is also necessary to avoid overfitting; in applications the truncation degree m will be typically determined by cross-validation.

Definition 4.8 *Given an integer $m \geq 1$, we call*

$$\mathrm{k}_m^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}, \quad \mathrm{k}_m^+(x,y) = \langle \mathrm{S}^+(\mathrm{k}_x), \mathrm{S}^+(\mathrm{k}_y) \rangle_m$$

the discretized signature kernel (or sequentialization) over k truncated at degree m. Here we use the notation $\mathrm{k}_x = (\mathrm{k}_{x_i})_i \in \mathcal{H}^+$ for $x = (x_i)_i \in \mathcal{X}^+$ and $\langle \cdot, \cdot \rangle_m$ for the inner product truncated[9] at m.

Corollary 4.9 *The discretized signature kernel k_m^+ over k truncated at degree m,*

$$\mathrm{k}_m^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}, \quad \mathrm{k}_m^+(x,y) = \langle \mathrm{S}^+(\mathrm{k}_x), \mathrm{S}^+(\mathrm{k}_y) \rangle_m,$$

1. *is a positive definite kernel,*
2. $\mathrm{k}_m^+(x,y) = \sum_{d=1}^m \sum_{\substack{1 \leq i_1 < \cdots < i_d < |x| \\ 1 \leq j_1 < \cdots < j_d < |y|}} \prod_{r=1}^d \nabla_{i_r, j_r} \mathrm{k}(x,y)$
3. $\mathrm{k}_m^+(x,y) = 1 + \sum_{\substack{|x| > i_1 \geq 1 \\ |y| > j_1 \geq 1}} \nabla_{i_1, j_1} \mathrm{k}(x,y) \cdot \left(1 + \cdots (1 + \sum_{\substack{|x| > i_m > j_{m-1} \\ |y| > j_m > j_{m-1}}} \nabla_{i_m, j_m} \mathrm{k}(x,y)) \right)$

where we denote $x = (x_i)_{i=1}^{|x|}, y = (y_i)_{i=1}^{|y|} \in \mathcal{X}^+$ and

$$\nabla_{i,j} \mathrm{k}(x,y) := \mathrm{k}(x_{i+1}, y_{j+1}) + \mathrm{k}(x_i, y_j) - \mathrm{k}(x_i, y_{j+1}) - \mathrm{k}(x_{i+1}, y_j).$$

While above follows as simple corollary from the discussion so far, the identity Point (3) will be the key to an efficient algorithm; it is clearly more efficient than computing first $\mathrm{S}^+(x)$ and $\mathrm{S}^+(y)$ and then their inner product; but it is also much more efficient than the identity in Point (2) since it only uses $\nabla_{i,j} \mathrm{k}(x,y)$ for each i, j once.

Remark 4.10 (Variations on a theme) *Modifications of the above are possible. For example,*

(i) One can omit the first differences, that is consider the sequentialization

$$\mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}; \quad (x,y) \mapsto 1 + \sum_{\substack{m \geq 1 \\ 1 \leq i_1 < \cdots < i_m \leq |x| \\ 1 \leq j_1 < \cdots < j_m \leq |y|}} \prod_{k=1}^m \mathrm{k}(x_{i_k}, y_{j_k}).$$

In feature space, this amounts to replacing the path $(\mathrm{k}_t)_{t \in [0,1]}$ by the path $(\int_0^t \mathrm{k}_s \, \mathrm{d}s)_{t \in [0,1]}$ and calculating the signature of the latter.

9. $\langle (s_0, s_1, \ldots), (t_0, t_1, \ldots) \rangle_m := \langle (s_0, s_1, \ldots, s_m, 0, \ldots), (t_0, t_1, \ldots, t_m, 0, \ldots) \rangle$

(ii) More generally, one may replace for each m, the inner product $\prod_{k=1}^{m} \mathrm{k}(x_{i_k}, y_{j_k})$ by a kernel $\mathrm{k}^m : \mathcal{X}^m \times \mathcal{X}^m \to \mathbb{R}$. This leads to a sequentialization of the family of kernels $(\mathrm{k}^m)_m$ given as

$$\mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}; \quad (x, y) \mapsto \sum_{\substack{m \geq 1 \\ 1 \leq i_1 < \cdots < i_m \leq |x| \\ 1 \leq j_1 < \cdots < j_m \leq |y|}} \mathrm{k}^m((x_{i_1}, \ldots, x_{i_m}), (y_{j_1}, \ldots, y_{j_m})).$$

This corresponds to choosing different kernels on different levels of the tensor algebra, e.g., re-normalization.

(iii) One may (additionally) opt to remember the position of elements in the sequence. This can be done by mapping sequences in $\mathcal{X}^+$ to a position-remembering sequence in $(\mathcal{X} \times \mathbb{N})^+$ first, and then carrying out the sequentialization in (ii) for the family of kernels $(\mathrm{k}^m)_m$ where $\mathrm{k}^m : (\mathcal{X} \times \mathbb{N})^+ \times (\mathcal{X} \times \mathbb{N})^+ \to \mathbb{R}$ is defined as

$$\kappa\left((i_1, \ldots, i_m), (j_1, \ldots, j_m)\right) \cdot \mathrm{k}^m\left((x_{i_1}, \ldots, x_{i_m}), (y_{j_1}, \ldots, y_{j_m})\right).$$

and $\kappa : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{R}$ denotes an arbitrary positive definite kernel. Further, one can replace the strict inequalities $<$ in the summation over $1 \leq i_1 < \cdots < i_m \leq |x|$ and $1 \leq j_1 < \cdots < j_m \leq |y|$ by $\leq$.

Further, Assumption 3.1, especially the continuity of k *can be dropped and any of above sequentializations gives a kernel for sequences (but convergence to a kernel on paths is no longer guaranteed). There are many other modifications and above choices, while minor, seem somewhat arbitrary. The above have their justification in explaining prior art, see Section 5 below.*

5. Revisiting classic kernels for sequences: string, alignment and convolution kernels

In this section we show that the discretized signature kernel is closely related to the existing kernels for sequential data, in the sense that classic kernels for sequences arise as special cases of our approach, namely:

(a) String kernels Lohdi et al. (2002); Leslie and Kuang (2004) can be seen as a case of discretized signature kernels.

(b) The global alignment kernel Cuturi et al. (2007); Cuturi (2011) can be obtained from a special case of the discretized signature kernel, by deleting some smaller terms, in the process destroying positive definiteness.

(c) The discretized signature kernel can be related to the framework of the relation-convolution kernel Haussler (1999) with the relation "being a subsequence".

The results of this section are not important for using our method as presented in the previous and following section, but it might be of interest to researchers who are familiar with above kernels.

5.1. The string kernel

The string kernel is a kernel for sequences in a finite set $\mathcal{X}$; in this context, the elements of $\mathcal{X}$ are usually called letters and $\mathcal{X}$ is called the alphabet. Two important examples are the Latin alphabet $\mathcal{X} = \{A, B, \ldots, Z, a, b, \ldots, z\}$ for classic text mining, or $\mathcal{X}$ being DNA nucleotides, proteins, etc., in bioinformatics.

Definition 5.1 (Definition 1, page 423 of Lohdi et al. (2002)) *Fix a finite set $\mathcal{X}$ and a parameter $\lambda \in [0, 1]$. The string kernel on $\mathcal{X}$ with parameter λ is defined as*

$$\mathrm{k}_{\mathrm{string}} : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R} \quad (x, y) \mapsto \sum_{w \in \mathcal{X}^+} \Phi_w(x)\Phi_w(y)$$

where

$$\Phi_w(x) = \sum_{1 \leq i_1 < \cdots < i_m \leq r} \lambda^{|i_m - i_1 + 1|} 1_{(x_{i_1}, \ldots, x_{i_k}) = w} \text{ for } w \in \mathcal{X}^+.$$

Proposition 5.2 *Fix a finite set $\mathcal{X}$ and a parameter $\lambda \in [0, 1]$. Define a kernel*

$$\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, \quad \mathrm{k}(a, b) = 1_{a=b}.$$

Then the sequentialization of k *given in Remark 4.10 (i), equals the string kernel with parameter $\lambda = 1$ up to an additive constant 1, that is $\mathrm{k}^+(x, y) = 1 + \mathrm{k}_{\mathrm{string}}(x, y)$ for all $x, y \in \mathcal{X}^+$.*

Proof We apply Theorem 4 in the variation given in Remark 4.10 (i) to the kernel k to get

$$\begin{aligned}
\mathrm{k}^+(x, y) &= 1 + \sum_{m \geq 1} \sum_{\substack{1 \leq i_1 < \cdots < i_m \leq |x| \\ 1 \leq j_1 < \cdots < j_m \leq |y|}} \prod_{r=1}^{m} \mathrm{k}(x_{i_r}, y_{j_r}) = 1 + \sum_{m \geq 1} \sum_{\substack{1 \leq i_1 < \cdots < i_m \leq |x| \\ 1 \leq j_1 < \cdots < j_m \leq |y|}} 1_{x_{i_1} = y_{j_1}} \cdots 1_{x_{i_m} = y_{j_m}} \\
&= 1 + \sum_{m \geq 1} \sum_{\substack{1 \leq i_1 < \cdots < i_m \leq |x| \\ 1 \leq j_1 < \cdots < j_m \leq |y|}} 1_{(x_{i_1}, \ldots, x_{i_m}) = (y_{j_1}, \ldots, y_{j_m})} \\
&= 1 + \sum_{m \geq 1} \sum_{w \in \mathcal{X}^+, |w| = m} \Phi_w(x)\Phi_w(y) = 1 + \sum_{w \in \mathcal{X}^+} \Phi_w(x)\Phi_w(y) = 1 + \mathrm{k}_{\mathrm{string}}(x, y)
\end{aligned}$$

■

More generally, for any $\lambda \in [0, 1]$ one can apply the κ-sequentialization given in Remark 4.10 (iii), to the kernel $\mathrm{k}(a, b) = 1_{a=b}$ and

$$\kappa((i_1, \ldots, i_m), (j_1, \ldots, j_m)) = \lambda^{i_m - i_1 + 1} \lambda^{j_m - j_1 + 1}$$

to recover the string kernel $\mathrm{k}_{\mathrm{string}}$ for the parameter λ; this follows as above. Similarly, various variants of the vanilla string kernel such as gappy kernels, Leslie and Kuang (2004) can be recovered[10]. Note that the $+1$ in $\mathrm{k}_{\mathrm{string}} + 1 = \mathrm{k}^+$ arises since in string kernels one usually does not count the empty word (if we count the empty word then $\mathrm{k}^+ = \mathrm{k}_{\mathrm{string}}$). However, this is merely convention and of non-importance for the Gram matrix.

10. As one of the reviewers pointed out, a very general framework for sequences that encompasses string kernels are so-called rational kernels.

Remark 5.3 *Implicit in above discussion is the identification of a string formed from an alphabet $\mathcal{X}$ as a lattice path in $\mathcal{H} = \mathbb{R}^{|\mathcal{X}|}$, that is the string $x = (a_{i_1}, \dots, a_{i_l})$ is identified as the lattice path in the vector space $\mathbb{R}^{|\mathcal{X}|}$ for which we identify $\mathcal{X}$ as an orthnomal basis (the free vector space over $\mathcal{X}$),*

$$x \in C^1([0,l], \mathcal{H}) \quad x(t) = \{t\} \cdot a_{i_{\lfloor t \rfloor}} + \sum_{r=1}^{\lfloor t \rfloor} a_{i_r},$$

where $\lfloor t \rfloor$ is the floor function of t, and $\{t\}$ is the fractional part of t. We deal with "non-geometric rough paths'', see Remark 4.5.

5.2. The global alignment kernel

The global alignment kernel is one of the most used kernels for sequences. We give its definition in modern terminology (Section 2.2 of Cuturi (2011)).

Definition 5.4 *Define*

$$\mathcal{A}(m,n) = \Big\{ \pi = (\pi_i)_{i=1}^k \in \left(\mathbb{N}^2\right)^+ : k \le m+n-1, \pi_1 = (1,1), \pi_k = (m,n), \\ (\pi_{i+1} - \pi_i) \in \{(0,1),(1,0),(1,1)\} \Big\}$$

We call an element of $\mathcal{A}(m,n)$ an alignment of (m,n).

Definition 5.5 *For a set $\mathcal{X}$ and a kernel $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, the global alignment kernel is defined as*

$$\mathrm{k}_{\mathrm{GA}} : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R} \quad (x,y) \mapsto \sum_{\pi \in \mathcal{A}(|x|,|y|)} e^{-D_{x,y}(\pi)}$$

where $D_{x,y}(\pi) = \sum_{i=1}^{|\pi|} \mathrm{k}(x_{\pi_i^1}, y_{\pi_i^2})$, and we denote with $|x|, |y|$ the length of the sequences x, y and we denote the entries of an alignment $\pi = (\pi_1, \dots, \pi_k)$ as $\pi_i = (\pi_i^1, \pi_i^2) \in \mathbb{N}^2$.

In its native form, the global alignment kernel cannot be written as a discretized signature kernel since these preserve positive definiteness, while the global alignment kernel does not (see the discussion in Cuturi (2011)). However, a simple modification turns the global alignment kernel into a positive definite kernel that can be obtained by sequentialization.

Definition 5.6 *Define*

$$\mathcal{A}_{\frac{1}{2}}(l) = \{\pi = (\pi_1, \dots, \pi_k) \in \left(\mathbb{N}^1\right)^+ : \pi_k \le l,\ \pi_{i+1} - \pi_i \in \{0,1\}\}$$

and call an element of $\mathcal{A}_{\frac{1}{2}} := \bigcup_{l \ge 1} \mathcal{A}_{\frac{1}{2}}(l)$ a half-alignment.

Definition 5.7 *Fix a set $\mathcal{X}$ and a kernel $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The global half-alignment kernel is defined as*

$$\mathrm{k}_{\mathrm{GHA}} : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R} \quad (x,y) \mapsto \sum_{\substack{\pi \in \mathcal{A}_{\frac{1}{2}}(|x|) \\ \rho \in \mathcal{A}_{\frac{1}{2}}(|y|) \\ |\pi| = |\rho|}} e^{-D_{x,y}(\pi,\rho)}$$

where $D_{x,y}(\pi,\rho) = \sum_{i=1}^{|\pi|} \mathrm{k}(x_{\pi_i}, y_{\rho_i})$.

Proposition 5.8 *Fix a set $\mathcal{X}$ and a positive definite kernel $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Then the global half-alignment kernel $\mathrm{k}_{\mathrm{GHA}}$ equals the κ-sequentialization of the kernel $(x,y) \mapsto e^{-\mathrm{k}(x,y)}$ given in Remark 4.10(iii), that is $\mathrm{k}_{\mathrm{GHA}}(x,y) = \mathrm{k}^+(x,y)$ with $\kappa(\pi,\rho) = 1_{\pi \in \mathcal{A}_{\frac{1}{2}}} 1_{\rho \in \mathcal{A}_{\frac{1}{2}}}$.*

Proof The κ-sequentializtion yields

$$\begin{aligned}
\mathrm{k}^+(x,y) &= \sum_{\substack{m \ge 1 \\ 1 \le \pi_1 \le \dots \le \pi_m \le |x| \\ 1 \le \rho_1 \le \dots \le \rho_m \le |y|}} \kappa((\pi_1,\dots,\pi_m),(\rho_1,\dots,\rho_m)) \prod_{r=1}^m \mathrm{k}(x_{\pi_r}, y_{\rho_r}) \\
&= \sum_{m \ge 1} \sum_{\substack{1 \le \pi_1 \le \dots \le \pi_m \le |x| \\ 1 \le \rho_1 \le \dots \le \rho_m \le |y|}} 1_{(\pi_1,\dots,\pi_m) \in \mathcal{A}_{\frac{1}{2}}} 1_{(\rho_1,\dots,\rho_m) \in \mathcal{A}_{\frac{1}{2}}} \prod_{r=1}^m e^{-\mathrm{k}(x_{\pi_r}, y_{\rho_r})} \\
&= \sum_{m \ge 1} \sum_{\substack{\pi \in \mathcal{A}_{\frac{1}{2}}(|x|) \\ \rho \in \mathcal{A}_{\frac{1}{2}}(|y|) \\ |\pi| = |\rho| = m}} e^{-\sum_{i=1}^m \mathrm{k}(x_{\pi_i}, y_{\rho_i})} = \mathrm{k}_{\mathrm{GHA}}(x,y).
\end{aligned}$$

■

Remark 5.9 *The terms missing in k_{GA}, when compared to $\mathrm{k}_{\mathrm{GHA}}$ are those arising from sequence pairs $\pi \in \left(\mathbb{N}^2\right)^+$ in which there is an increment $(0,0)$. In view of the discussion in Section 3.2 of Cuturi (2011), these missing terms are exactly the locus of non-transitivity in the sense of Shin and Kuboyama (2008). Concerning the possible non positive definiteness of k_{GA}, one can follow Cuturi (2011) and study sufficient conditions under which the original global alignment kernel is positive definite.*

5.3. The relation-convolution kernel

Finally, we would like to point out that the discretized signature kernels are closely related to the relation-convolution kernels, see Haussler (1999), Section 2.2.

Definition 5.10 *Fix kernels $\mathrm{k}_i : \mathcal{X}_i \times \mathcal{X}_i \to \mathbb{R}$ for $i = 1, \dots, m$. Fix a relation R between the sets $\mathcal{X}_1 \times \cdots \times \mathcal{X}_m$ and $\mathcal{X}$, that is a subset $R \subset \mathcal{X}_1 \times \cdots \times \mathcal{X}_m \times \mathcal{X}$ and say that $(x_1, \dots, x_m) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_m$ decomposes $x \in \mathcal{X}$ if $(x_1, \dots, x_m, x) \in R$. The relation-convolution kernel of $(\mathrm{k}_1, \dots, \mathrm{k}_m)$ is defined as*

$$\mathrm{k}_{\mathrm{RC}} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, \quad (x,y) \mapsto \sum_{\substack{(x_1,\dots,x_m):((x_1,\dots,x_m),x) \in R \\ (y_1,\dots,y_m):((y_1,\dots,y_m),y) \in R}} \prod_{i=1}^m \mathrm{k}_i(x_i, y_i).$$

The intuition for above definition is that we can measure similarity of two objects $x, y \in \mathcal{X}$ by decomposing them into substructures and comparing these substructures.

Proposition 5.11 *Let $\mathcal{X}$ be any set and $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Then the sequentialization $\mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}$ given in Remark 4.10 (i) is a relation-convolutional kernel for the relation*

"being an ordered subsequence''. More precisely, define $\mathcal{X}_1 = \mathbb{N}$, $\mathcal{X}_i = \mathcal{X}$ *for* $i = 2, \dots, m+1$ *and*

$$\mathrm{k}_1(i,j) = 1_{i=j} \text{ and } \mathrm{k}_i(x,y) = \mathrm{k}(x,y) \text{ for } i = 2, \dots, m+1.$$

Then for

$$R = \{(l, x_1, \dots, x_{m+1}, x') \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_{m+1} \times \mathcal{X}^+ : l \le m \text{ and } \exists i_1 < \cdots < i_l \text{ such that } (x'_1, \dots, x'_l) = (x_{i_1}, \dots, x_{i_l})\}$$

the relation-convolutional kernel k_{RC} *equals* k^+.

The proof follows by spelling out the definitions of k_{RC} and k^+. It is interesting to note the native discretized signature kernel of Theorem 4 does not, at least naturally, fall into Haussler's convolutional kernel framework since it heavily relies on the additional differences $\nabla \mathrm{k}$. (Above variation is only obtained after the summation step described in Remark 4.10).

6. Efficient algorithms

In this section, we design effective algorithms to evaluate or approximate the $(n \times n)$-Gram matrix $\left(\mathrm{k}^+_m(x_i, x_j)\right)_{i,j\in\{1,\dots,n\}}$ of n sequences $x_1, \dots, x_n \in \mathcal{X}^+$. The results are summarized in Table 2. The key are the recursive Horner-type formula given in Corollary 4.9 together with ideas from dynamic programming and low-rank approximations.

method	algorithm	time	storage
naive evaluation	Definition 4.8	$O(n^2 \cdot \ell \cdot \dim(\mathcal{H})^m)$	$O(n^2 \dim(\mathcal{H})^m)$
dynamic programming	Algorithm 3	$O(n^2 \cdot \ell^2 \cdot m)$	$O(\ell^2)$
DP & LR, per-element	Algorithm 4	$O(n^2 \cdot \ell \cdot \rho \cdot m)$	$O(\ell \cdot \rho)$
DP & LR, per-sequence	Section 6.3.1	$O((n+\rho) \cdot \ell^2 \cdot \rho \cdot m)$	$O(n \cdot \ell^2)$
DP & LR, simultaneous	Algorithm 5	$O(n \cdot \ell \cdot \rho \cdot m)$	$O(n \cdot \ell \cdot \rho)$

Table 2: Computational time (in elementary arithmetic operations) and storage cost for computing (an approximation) to the $(n \times n)$-Gram matrix $\left(\mathrm{k}^+_m(x_i, x_j)\right)_{i,j\in\{1,\dots,n\}}$ between sequences $x_1, \dots, x_n \in \mathcal{X}^+$ of length at most ℓ, that is $|x_i| \le \ell$ for $i = 1, \dots, n$. Methods: DP = dynamic programming, LR = low-rank. In the low-rank methods, ρ is a meta-parameter which controls accuracy of approximation.

Remark 6.1 *Naive evaluation of* $\mathrm{k}^+_m(x,y)$ *by directly calculating it as an inner product of signature features incurs an exponential storage cost* $O(\dim(\mathcal{H})^m)$. *Even for* $\mathcal{H} = \mathbb{R}^d$ *this becomes quickly very expensive and infeasible when* $\dim(\mathcal{H}) = \infty$ *which is the case for most practically relevant kernels.*

Remark 6.2 *Above algorithms are already powerful for* $\mathcal{H} = \mathbb{R}^d$ *with the inner product in* $\mathbb{R}^d$ *as kernel* k. *In this case,* $\mathrm{k}^+_x \approx \mathrm{S}(\mathrm{k}_x) = \mathrm{S}(\langle \mathrm{k}_x, \cdot \rangle) \simeq \mathrm{S}(x) \in \prod_{m\ge 0} \left(\mathbb{R}^d\right)^{\otimes m}$, *that is the feature map for a path* x *is the usual signature* $\mathrm{S}(x) \in \prod_{m\ge 0} \left(\mathbb{R}^d\right)^{\otimes m}$. *Compared to naive computation of* $\mathrm{S}(x)$, *Algorithm 5 gives a massive reduction in complexity (from exponential to linear!).*

Remark 6.3 *For readability we have absorbed in Table 2 the computational time* c *that is needed to evaluate* $\mathrm{k}(x,y)$ *for* $x, y \in \mathcal{X}$ *into the big O bound since this is just a multiplicative factor. E.g. the computational time for Algorithm 3 would read* $O(c \cdot n^2 \cdot \ell^2 \cdot m)$ *if we include* c.

6.1. Notation and subroutines for computations with arrays (tensors)

We introduce notation (inspired by Python/Numpy) and fast algorithms for dynamic programming subroutines. This allows us to express our algorithms as operations on arrays alone and makes it easy to implement them in high-level languages that are optimized for operating on arrays (such as Numpy, etc).

Notation 6.4 (Arrays/tensors) *We will denote the* $(i_1, \dots, i_k)$*-th element of a* k*-fold array (= degree* k *tensor)* A *by* $A[i_1, \dots, i_k]$. *Occasionally, for ease of reading, we will use "|" instead of "," as a separator, for example* $A[i_1, i_2|i_3, \dots, i_k]$ *if the indices on the left side of "|" are semantically distinct from those on the right side. The arrays all contain elements in* $\mathbb{R}$, *and the indices will always be positive integers, excluding zero.*

Notation 6.5 (functions applied elementwise) *For a function* $f : \mathbb{R} \to \mathbb{R}$, *and an array* A, *we will denote by* $f(A)$ *the array where* f *is applied element-wise. I.e.,* $f(A)[i_1, \dots, i_k] = f(A[i_1, \dots, i_k])$. *Similarly, for* $f : \mathbb{R}^m \to \mathbb{R}$ *and arrays* $A_1, \dots, A_m$, *we denote*

$$f(A_1, \dots, A_m)[i_1, \dots, i_k] = f(A_1[i_1, \dots, i_k], \dots, A_m[i_1, \dots, i_k]).$$

For example, $\frac{1}{2} \cdot A^2$ *is the array* A *having all elements squared, then divided by two. The array* $A + B$ *contains, element-wise, sums of elements of* A *and* B.

Notation 6.6 (slice, shift, cumulative sum) *Let* A *be a* k*-fold array of size* $(n_1 \times \cdots \times n_k)$.

(i) For an index i_j *(at* j*-th position), we will write* $A[:, \dots, :, i_j, :, \dots, :]$ *for* $(k-1)$*-fold array of size* $(n_1 \times \cdots \times n_{j-1} \times n_{j+1} \times \dots n_k)$ *such that*

$$A[:, \dots, :, i_j, :, \dots, :][i_1, \dots, i_{j-1}, i_{j+1}, \dots, i_k] = A[i_1, \dots, i_k].$$

We define in analogy, iteratively, $A[:, \dots, :, i_j, :, \dots, :, i_{j'}, :, \dots, :]$, *and so on. Arrays of this type are called slices (of* A*).*

(ii) For an integer m, *we will write* $A[:, \dots, :, +m, :, \dots, :]$ *for the* k*-fold array of size* $(n_1 \times \cdots \times n_{j-1} \times (n_j + m) \times n_{j+1} \times \cdots \times n_k)$ *such that*

$$A[:, \dots, :, i_j, :, \dots, :][i_1, \dots, i_{j-1}, i_j + m, i_{j+1}, \dots, i_k] = A[i_1, \dots, i_k],$$

and where non-existing indices of A *are treated as zero. Arrays of this type are called shifted (versions of* A*). For negative* m, *the shifts will be denoted with a "minus"-sign instead of a "plus"-sign.*

(iii) We will write $A[:,\dots,:,\boxplus,:,\dots,:]$, where $\boxplus$ is at the j-th position, for the k-fold array of size $(n_1 \times \cdots \times n_k)$ such that

$$A[:,\dots,:,\boxplus,:,\dots,:][i_1,\dots,i_k] = \sum_{\kappa=i_j}^{n_j} A[i_1,\dots,i_{j-1},\kappa,i_{j+1},\dots,i_k].$$

Arrays of this type are called slice-wise cumulative sums (of A).

(iv) We will write $A[:,\dots,:,\Sigma,:,\dots,:]$, where Σ is at the j-th position, for the $(k-1)$-fold array of size $(n_1 \times \cdots \times n_{j-1} \times n_{j+1} \times \cdots \times n_k)$ such that $A[:,\dots,:,\Sigma,:,\dots,:][i_1,\dots,i_{k-1}] = \sum_{\kappa=1}^{n_j} A[i_1,\dots,i_{j-1},\kappa,i_j,\dots,i_{k-1}]$. Arrays of this type are called slice-wise sums (of A).

We will further use iterations and mixtures of the above notation, noting that the index-wise sub-setting, shifting, and accumulation commute with each other. Therefore expressions such as $A[+1,:|\Sigma,+2]$ or $A[j|:,+3,\boxplus]$ are well-defined, for example. We will also use the notation $A[\boxplus+m,\dots]$ to indicate the shifted variant of the cumulative sum array.

Cumulative sums can be computed efficiently, in the order of the size of an array, as opposed to squared complexity of more naive approaches. The algorithm is classical, we present it for the convenience of the reader in Algorithms 1 and 2.

Algorithm 1 Computing the cumulative sum of a vector.
Input: A 1-fold array A of size (n)
Output: The cumulative sum array $A[\boxplus]$

1: Let $Q \leftarrow A$.
2: **for** $\kappa = 2$ to n **do**
3: $\quad Q[\kappa] \leftarrow Q[\kappa-1] + A[\kappa]$
4: **end for**
5: Return Q

Algorithm 2 Computing the cumulative sum of an array.
Input: A k-fold array A of size $(n_1 \times \cdots \times n_k)$
Output: The cumulative sum array $A[\boxplus,\dots,\boxplus,:,\dots,:]$ (up to the m-th index)

1: Let $Q \leftarrow A$
2: **for** $\kappa = 2$ to m **do**
3: $\quad$ Let $Q \leftarrow Q[:,\dots,:,\boxplus,:,\dots,:]$ (at the κ-th index), where the right side is computed via applying algorithm 1 slice-wise.
4: **end for**
5: Return Q

6.2. Computing the discretized signature kernel

Algorithm 3 Evaluation of k_m^+.
Input: Sequences $x, y \in \mathcal{X}^+$. A kernel $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A truncation degree m.
Output: $\mathrm{k}_m^+(x,y)$

1: Set $\ell_x = |x| - 1, \ell_y = |y| - 1$
2: Compute the $(\ell_x \times \ell_y)$-array K with entries $K[i,j] = \nabla_{i,j}\,\mathrm{k}(x,y)$
3: Initialize an $(m \times \ell_x \times \ell_y)$-array A
4: Set $A[1|:,:] \leftarrow K$
5: **for** $d = 2$ to m **do**
6: $\quad$ Compute $Q \leftarrow A[d-1|\boxplus,\boxplus]$
7: $\quad$ Set $A[d|:,:] \leftarrow K \cdot (1 + Q[+1,+1])$
8: **end for**
9: Compute $R \leftarrow 1 + A[m|\Sigma,\Sigma]$
10: Return R

The correctness of above Algorithm follows directly from the recursive formula (3) in Corollary 4.9: we start by evaluating the innermost parenthesis and then use the summation to build the content of the next parenthesis. In symbols, define for $d = 1,\dots,m-1$,

$$A^1_{i,j} := \nabla_{i,j}\,\mathrm{k}(x,y) \text{ and } A^{d+1}_{i,j} := \nabla_{i,j}\,\mathrm{k}(x,y)\left(1 + \sum_{\substack{i'>i\\ j'>j}} A^d_{i',j'}\right),$$

then $\mathrm{k}_m^+(x,y) = 1 + \sum_{\substack{i\geq 1\\ j\geq 1}} A^m_{i',j'}$. In above Algorithm 3 we denote $A[d|i,j] = A^d_{i,j}$.

Remark 6.7

- *Disregarding the cost of computing the $(\ell_x \times \ell_y)$-matrix $(\nabla_{i,j}\,\mathrm{k}(x,y))_{i,j}$, the computational cost of Algorithm 3 is $O(m|x||y|)$ elementary arithmetic operations (= the number of loop elements) and $O(m|x||y|)$ units of elementary storage. The storage requirement can be reduced to $O(|x||y|)$ by discarding $A[d-1|:,:]$ from memory after step 7 each time.*

- *In each run of the loop, a matrix Q is pre-computed to avoid a five-fold loop that would be necessary with the more naive version of line 7,*

$$A[m|i,j] \leftarrow A[m|i,j] \cdot \left(1 + \sum_{i'>i}\sum_{j'>j} A[m-1|i',j']\right),$$

that leads to a blown up computational cost of $O(m\ell_x^2\ell_y^2)$ at the asymptotically insignificant gain of storing one $(\ell_x \times \ell_y)$-matrix less (the matrix Q).

6.3. Large scale strategies

By using Algorithm 3 one can compute the Gram matrix $\left(\mathrm{k}^+_m(x_i, x_j)\right)_{i,j=1,\dots,n}$ of n sequences $x_1, \dots, x_n \in \mathcal{X}^+$ in $O(n^2 \cdot \ell^2 \cdot m)$ elementary arithmetic operations. While this is for moderate sizes of n,m and ℓ achievable on contemporary desktop computers, it becomes quickly prohibitive for large n, ℓ due to the quadratic growth — especially when combined with parameter tuning or cross-validation schemes (as later in our experiments). We present two low-rank techniques to address this: the first is classic and reduces the quadratic cost in n to linear, the second reduces the quadratic cost in ℓ to linear. Finally, we combine these two techniques to get $O(n \cdot \ell \cdot m \cdot \rho)$ computation time with ρ denoting an approximation parameter.

6.3.1. Low-rank methods for the sequence-vs-sequence kernel matrix

Since $\left(\mathrm{k}^+_m(x_i, x_j)\right)_{i,j=1,\dots,n}$ is a Gram-matrix, it is directly amenable to large-scale variants of low-rank type. Strategies of this kind include the incomplete Cholesky decomposition, Nyström approximation, or the inducing point formalism in a Gaussian process framework. For n sequences, all strategies mentioned above require evaluation of at most an $(n \times r)$ and an $(r \times r)$ matrix (where r is a meta-parameter), which costs $O((r+n) \cdot r \cdot \ell^2 \cdot m)$ elementary operations and $O((r+n) \cdot r \cdot \ell^2)$ storage[11]. Any of these low-rank strategies reduces the complexity of the Gram matrix to be linear in n, but not in ℓ since the strategy is completely independent of how the discretized signature kernel was evaluated at a given pair $(x_i, x_j) \in \mathcal{X}^+ \times \mathcal{X}^+$. For the same reason, any improvements on the cost of evaluating $\mathrm{k}^+_m(x_i, x_j)$ will combine with this strategy.

6.3.2. Low-rank methods for the element-vs-element kernel matrix

The evaluation $\mathrm{k}^+_m(x_i, x_j)$ for a pair $(x_i, x_j) \in \mathcal{X}^+$ is given by nested partial summations and multiplications applied to the $(\ell_j \times \ell_j)$-matrix $(\mathrm{k}(x_{i,r}, x_{j,s}))_{r,s}$ where $r = 1, \dots, \ell_i$, $s = 1, \dots, \ell_j$ and we denote $x_i = (x_{i,1}, \dots, x_{i,\ell_i}), x_j = (x_{j,1}, \dots, x_{j,\ell_j}) \in \mathcal{X}^+$. The low-rank methods of the previous paragraph cannot be applied naively to this $(\ell_i \times \ell_j)$- matrix: firstly, this matrix is in general non-symmetric; secondly, the summation- and multiplication-type operations require in naive form at each recursion step access to the full $(\ell_i \times \ell_j)$-matrix. The first issue is easily addressed by replacing the respective symmetric decomposition with the analogous non-symmetric one. The second issue is much harder to deal with, but below we show that by working exclusively on low-rank factorizations in each recursion step this issue can also be dealt with.

Definition 6.8 *Let A be an $(a \times b)$-matrix. For U an $(a \times r)$-matrix, and V a $(b \times r)$-matrix, we say that (U, V) is a low-rank presentation of A, of rank r, if*

$$A_{i,j} = \sum_{r'=1}^{r} U_{i,r'} V_{j,r'}$$

With $\bullet$ denoting the usual matrix multiplication this is equivalent to $A = U \bullet V^\top$.

11. followed by a slightly modified variant of the learning algorithm itself which usually costs $O((r+n) \cdot r^2)$ elementary operations and storage (at most) instead of the unmodified variant which usually costs $O(n^3)$ (at most).

The fact that A has a low-rank presentation of rank r does imply that A is of rank r or less (by equivalence of matrix rank with decomposition rank), but it does not imply that A is of rank exactly r.

We state a number of straightforward but computationally useful Lemmas that show how low-rank matrices behave under shifting, summation and cumulative summation, componentwise addition and multiplication. This allows us to replace these operations on matrices in Algorithm 4 by the operations on low-rank matrices.

Lemma 6.9 (Low-rank under cumulative summations and shifts) *Let (U, V) be a low-rank representation of rank r for an $(a \times b)$-matrix A. Then*

(i) Define A' and A'' as $A'_{i,j} = \sum_{i' \geq i} A_{i',j}$ and $A''_{i,j} = \sum_{j' \geq j} A_{i,j'}$. Define U' as $U'_{i,r} = \sum_{i' \geq i} U_{i',r}$ and V' as $V'_{j,r} = \sum_{j' \geq j} V_{j',r}$. Then (U', V) is a low-rank representation or rank r of A' of rank and (U, V') is a low-rank representation of rank r of A'.

(ii) Define A' and A'' as $A'_{i,j} = A_{i+s,j}$ and $A''_{i,j} = A_{i,j+s}$ for $s \geq 1$. Define U' and V' as $U'_{i,r} = U_{i+s,r} 1_{i+s \leq a}$ and $V'_{j,r} = V_{j+s,r} 1_{j+s \leq b}$. Then (U', V) is a low-rank representation of A' of rank r and (U, V') is a low rank representation of rank r of A''.

Proof This follows by spelling out the definition: for (i)

$$A'_{i,j} = \sum_{i' \geq i} A_{i',j} = \sum_{i' \geq i} \sum_{r'=1}^{r} U_{i',r'} V_{j,r'} = \sum_{r'=1}^{r} U'_{i,r'} V_{j,r'};$$

for (ii)

$$A'_{i,j} = A_{i+s,j} = \sum_{r'=1}^{r} U_{i+s,r'} V_{j,r'} = \sum_{r'=1}^{r} U'_{i,r'} V_{j,r'}.$$

The statements for A'' follow similarly. ■

Lemma 6.10 (Low rank under addition and multiplication) *Let A_1, A_2 be $(a \times b)$-matrices such that (U, V) is a low-rank representation of A_1 of rank r_1, and (R, S) is a low-rank representation of A_2 of rank r_2. Then*

(i) (U', V') is a low rank representation of $A_1 + A_2$ of rank $r_1 + r_2$ where

$$U'_{i,r} = \begin{cases} U_{i,r} & \text{, if } 1 \leq r \leq r_1 \\ R_{i,r-r_1} & \text{, if } 1 + r_1 \leq r \leq r_1 + r_2 \end{cases}, \; V'_{j,r} = \begin{cases} V_{j,r} & \text{, if } 1 \leq r \leq r_1 \\ S_{j,r-r_1} & \text{, if } 1 + r_1 \leq r \leq r_1 + r_2 \end{cases}$$

(ii) $(\bar{U}, \bar{V})$ is a low rank representation of[12] $A_1 \circ A_2$ of rank $r_1 r_2$ where

$$\bar{U}_{i,r} = U_{i,a} R_{i,b} \text{ and } \bar{V}_{j,r} = V_{j,a} S_{j,b}$$

and a, b are such that $r = (a-1) \cdot r_2 + b - 1$ with $a \in \{1, \dots, r_1\}$ and $b \in \{1, \dots, r_2\}$

12. We denote the elementwise product of matrices with $A \circ B$, that is $(A \circ B)_{i,j} = A_{i,j} B_{i,j}$.

Proof Since the (i,j)-entry of the matrix A_1+A_2 equals $\sum_{r=1}^{r_1} U_{i,r}V_{j,r}+\sum_{r'=1}^{r_2} R_{i,r'}S_{j,r'}$ and the (i,j)-entry of $A_1 \circ A_2$ equals $\sum_{r=1}^{r_1} U_{i,r}V_{j,r}\sum_{r'=1}^{r_2} R_{i,r'}S_{j,r'}$, the result follows by substituting the two sums over $r \in \{1,\dots,r_1\}$ and $r' \in \{1,\dots,r_2\}$ by one sum over $\{1,\dots,r_1+r_2\}$ resp. over $\{1,\dots,r_1 r_2\}$. ■

In view of Point (ii) of above Lemma we introduce new array notation:

Notation 6.11 *Let A_1 be a k-fold array of size $(n_1 \times \cdots \times n_k)$ and A_2 be a k-fold array of size $(n_1 \times \cdots \times n_{k-1} \times n'_k)$. Write $A_1 \star A_2$ for the $(n_1 \times \cdots \times n_{k-1} \times (n_k \cdot n'_k))$ array with entries*

$$A_1 \star A_2[i_1,\dots,i_k] = A_1[i_1,\dots,i_{k-1},a]A_2[i_1,\dots,i_{k-1},b]$$

where a,b are such that $i_k = (a-1)\cdot n_2 + b - 1$ with $a \in \{1,\dots,n_1\}$ and $b \in \{1,\dots,n_2\}$.

Algorithm 4 Evaluation of k_m^+, with low-rank speed-up.
Input: Sequences $x, y \in \mathcal{X}^+$. A kernel $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A truncation degree $m \geq 1$. An integer $1 \leq r \leq \min(\ell_x, \ell_y)$.
Output: An approximation to $\mathrm{k}_m^+(x,y)$

1: Set $\ell_x \leftarrow |x| - 1$ and $\ell_y \leftarrow |y| - 1$
2: Compute a low-rank representation (U,V) of rank r, approximating the element-vs-element matrix/array with entries $K[i,j] = \nabla_{i,j}\,\mathrm{k}(x,y)$, $i \in \{1,\dots,\ell_x\}$, $j \in \{1,\dots,\ell_y\}$
3: Initialize an $(m \times \ell_x \times *)$-array B and an $(m \times \ell_y \times *)$-array C (* means that the size may change dynamically)
4: Set $B[1|\,:,:] \leftarrow U$ and $C[1|\,:,:] \leftarrow V$
5: **for** $d = 2$ to m **do**
6: $\quad P \leftarrow B[d-1|\boxplus+1,:]$ and $Q \leftarrow C[d-1|\boxplus+1,:]$
7: $\quad$ Append an $(\ell_x \times 1)$-array of 1's to P and append an $(\ell_y \times 1)$-array of 1's to Q
8: $\quad$ Set $B[d|\,:,:] \leftarrow (U \star P)[:,:]$ and $C[d|\,:,:] \leftarrow (V \star Q)[:,:]$
9: $\quad$ optional: "simplify" the low-rank presentation (B,C), reducing its rank
10: **end for**
11: Set $R \leftarrow B[m|\Sigma,:]$ and $S \leftarrow C[m|\Sigma,:]$
12: Return $1 + (R \cdot S)[\Sigma]$

Algorithm 4 first approximates the element-vs-element $(\ell_x \times \ell_y)$-matrix $K = (\nabla_{i,j}\,\mathrm{k}(x,y))_{i,j}$ by a low rank approximation (U,V). It then uses the formula for k_m^+ given in Corollary 4.9 and updates the low-rank representation in each recursion step (starting from the innermost parenthesis):

- The first time line 6 is called it uses Lemma 6.9 to calculate a low-rank representation (P,Q) of the $(\ell_x \times \ell_y)$-matrix $\left(\sum_{i'>i,j'>j} \nabla_{i',j'}\,\mathrm{k}(x,y)\right)_{i,j}$ from the low-rank approximation (U,V) for $K = (\nabla_{i,j}\,\mathrm{k}(x,y))_{i,j}$; later calls of line 6 do the same but with K replaced by another matrix with low-rank factorization stored in $B[d|\,:,:]$ and $C[d|\,:,:]$.

- The first time line 7 is called, it calculates a low-rank representation $(\bar{P},\bar{Q})$ of the $(\ell_x \times \ell_y)$-matrix $(1 + (P \bullet Q^\top)_{i,j})_{i,j}$ by using (P,Q) and the trivial identity

$$1 + \sum_{k=1}^{r} P_{i,k}Q_{j,k} = \sum_{k=1}^{r+1} \bar{P}_{i,k}\bar{Q}_{j,k}$$

  where $\bar{P}$ and $\bar{Q}$ are given by adding a $(r+1)$-th column consisting of 1's. Note that the rank increases by one by going from (P,Q) to $(\bar{P},\bar{Q})$. Later calls do the same but with (P,Q) replaced by another matrix with low-rank factorization stored in $B[d|\,:,:]$ and $C[d|\,:,:]$.

- Line 8 calculates a low-rank representation $(\bar{U},\bar{V})$ for the elementwise product between the $(\ell_x \times \ell_y)$-matrices $K = U \bullet V^\top$ and $\bar{P} \bullet \bar{Q}^\top$. By Lemma 6.10 it is given as $\bar{U}_{i,\bar{r}} = U_{i,a}\bar{P}_{i,b}$ and $\bar{V}_{\bar{r},j} = V_{a,j}\bar{Q}_{b,j}$.

- Line 9 is optional, aiming at keeping the low-rank presentation small[13].

- Line 11 uses that

$$\sum_{i\geq 1, j\geq 1} (R \bullet S^\top)_{i,j} = \sum_{i\geq 1, j\geq 1}\sum_{k} R_{i,k}S_{j,k} = \sum_{k}(\sum_{i\geq 1} R_{i,k})(\sum_{j\geq 1} S_{j,k}).$$

  (Recall that $R \cdot S$ denotes elementwise multiplication of arrays R,S and $R \bullet S$ denotes the usual matrix multiplication if R,S are interpreted as matrices).

- The computational cost of Algorithm 4 is of the same order as the maximum size of B and C. That is, if ρ is the smallest integer such that at any time B requires $\ell_x \cdot m \cdot \rho$ space, and C requires $\ell_y \cdot m \cdot \rho$ space, then the computational complexity of Algorithm 4 is[14] $O((\ell_x + \ell_y)\cdot \rho \cdot m)$.

6.3.3. Simultaneous low-rank methods

Algorithm 4 yields an efficient low-rank speed-up for computing a single entry $\mathrm{k}_m^+(x_i,x_j)$. Thus total computational cost for the Gram matrix $(\mathrm{k}_m^+(x_i,x_j))_{i,j\in\{1,\dots,n\}}$ is $O(n^2 \cdot \ell \cdot \rho \cdot m)$ and cost quadratic in the number of data points n may be prohibitive on large scale data. We address this by combining both low-rank strategies mentioned before to achieve a further reduction of computational cost from $O(n^2 \cdot \ell \cdot \rho \cdot m)$ to $O(n \cdot \ell \cdot \rho \cdot m)$.

13. This can be achieved for example via singular value decomposition, sub-sampling, or random projection type techniques.
14. Since one can always choose a low-rank representation of $B[m|\,:,:]$ and $C[m|\,:,:]$ of rank $\min(\ell_x,\ell_y)$ or less, the computational complexity is bounded by $O(\ell_x \cdot \ell_y \cdot m)$ (which is the complexity of Algorithm 3).

Algorithm 5 Computation of the Gram matrix of k_m^+, with (double) low-rank speed-up.
Input: Sequences $x_1, \dots, x_n \in \mathcal{X}^+$. A kernel $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A truncation degree m. An integer $r \geq 1$.
Output: A low-rank approximation (U, U) of the Gram matrix $K = \left(\mathrm{k}_m^+(x_i, x_j)\right)_{i,j=1,\dots,n}$.

1: Compute arrays $U^{(i)}$ such that each pair $(U^{(i)}, U^{(j)})$ $i, j \in \{1, \dots, n\}$ is a low-rank presentation of rank r for the $((|x_i| - 1) \times (|x_j| - 1))$-matrix $K^{(ij)}$ with entries $K^{(ij)}_{a,b} = \nabla_{a,b} \mathrm{k}(x_i, x_j)$
2: Initialize an $(n \times m \times * \times *)$-array B (where * means that the sizes may change dynamically)
3: $B[i|1| :, :] \leftarrow U^{(i)}$ for all $i \in \{1, \dots, n\}$
4: **for** $d = 2$ to m **do**
5: Compute $P \leftarrow B[: |d-1| \boxplus +1, :]$
6: Set κ, ρ such that $(n \times \kappa \times \rho)$ is the size of P
7: Append an $(n \times \kappa \times 1)$-array of ones to P
8: $B[: |d| :, :] \leftarrow B[: |1| :, :] \star P[:, :, :]$
9: optional: "simplify" the low-rank presentation encoded in B, reducing its rank
10: **end for**
11: Compute $U \leftarrow B[: |m|\Sigma, :]$
12: Return U

Algorithm 5 is obtained as follows:

- The previous Algorithm 4 transforms for any pair $x_i, x_j \in \mathcal{X}^+$ a low rank representation $(U^{(ij)}, V^{(ij)})$ of the $(\ell_i \times \ell_j)$-matrix $K^{(ij)} = (\nabla_{a,b} \mathrm{k}(x_{i,a}, x_{j,b}))_{a,b}$ into a low-rank representation $(R^{(ij)}, S^{(ij)})$ such that

$$\mathrm{k}^+(x_i, x_j) = 1 + \sum_{a \geq 1, b \geq 1} \left(R^{(ij)} \bullet (S^{(ij)})^\top \right)_{a,b}.$$

  But if $U^{(ij)}$ is independent of j and if $V^{(ij)}$ is independent i, then $R^{(ij)}$ will depend by Algorithm 4 only on i and $S^{(ij)}$ will depend only on j; hence $R^{(ij)} = S^{(ji)}$. If we denote $R^{(i)} := R^{(ij)} := S^{(ji)}$, then in above Algorithm 5, $B[i, m, a, r]$ equals $R^{(i)}_{a,r}$ when the end of the for loop is reached, line 10.

- Write

$$\mathrm{k}_m^+(x_i, x_j) = 1 + \sum_{\substack{a \geq 1 \\ b \geq 1}} \left(R^{(i)} \bullet (R^{(j)})^\top \right)_{a,b} = 1 + \sum_{k=1}^{r'} \left(\sum_{a \geq 1} R^{(i)}_{a,k} \sum_{b \geq 1} R^{(j)}_{b,k} \right)$$
$$= \sum_{k=1}^{r'+1} \bar{R}_{i,k} \bar{R}_{j,k}$$

  where we denote with r' the rank of $(R^{(i)}, R^{(j)})$ and set $\bar{R}_{i,k} := \sum_{a \geq 1} R^{(i)}_{a,k}$ and $\bar{R}_{a,k} := 1$ if $k = r' + 1$. Hence, $(\bar{R}, \bar{R})$ is a low-rank representation of the $(n \times n)$-Gram matrix $K = (\mathrm{k}_m^+(x_i, x_j))_{i,j}$. In Algorithm 5, the array entry $U[i, r']$ equals $\bar{R}_{i,r'}$, line 11.

Remark 6.12 *Line 1 requires that for each pair of sequences $x_i = (x_{i,a})_{a=1,\dots,\ell_i}, x_j = (x_{j,b})_{b=1,\dots,\ell_j} \in \mathcal{X}^+$ we compute the $(\ell_i \times \ell_j)$-matrix $K^{(ij)}$ with entries $K^{(ij)}_{a,b} = \mathrm{k}(x_{i,a}, x_{j,b})$. That is, the matrices $U^{(i)}$, when row-concatenated, should have low rank*[15]*. Jointly low-rank $U^{(i)}$ can be obtained by running a suitable joint diagonalization or singular value decomposition scheme on the matrices $K^{(ij)}$.*

Remark 6.13 (Fast sequential kernel methods) *By Section 5, fast string kernel methods such as the gappy, substitution, or mismatch kernels presented in Leslie and Kuang (2004) may be transferred to general sequential kernels. In general, this amounts to small modification of Algorithm 3; for example, to obtain a gappy variant of the sequential kernel, summation in line 6 of Algorithm 3 over the whole matrix, of quadratic size, is replaced by summation over a linear part of it.*

7. Experimental validation

We perform two experiments to validate the practical usefulness of the signature kernels:

(1) On a real world data set of hand movement classification (eponymous UCI data set Sapsanis et al. (2013)), we show the discretized signature kernel outperforms the best previously reported predictive performance Sapsanis et al. (2013), as well as non-sequential kernel and aggregate baselines.

(2) On a real world data set on hand written digit recognition (pendigits), we show that the discretized signature kernel over the Euclidean kernel (= linear use of signature features) achieves only sub-baseline performance. Using the discretized signature kernel over a Gaussian kernel improves prediction accuracy to the baseline region.

We emphasize that our experiments do not constitute a systematic benchmark comparison to prior work, only validation that the signature kernel is a practically meaningful concept: experiment (1), shows that the sequentialization of standard kernels can achieve state-of-the-art performance on time series data; experiment (2) shows that even for paths in low dimensions, $\mathcal{X} = \mathbb{R}^2$, using a non-linear static kernel for the sequentialization outperforms the linear kernel (the latter corresponds to learning with signature features of a 2-dimensional path; the former, corresponds to learning with signature features of the path lifted to the RKHS $\mathcal{H}$ of the non-linear kernel).

A systematic benchmark comparison is likely to require a larger amount of work, since it would have to include a number of previous methods (multiple variants of the string and general alignment kernels, dynamic time warping, naive use of signatures), for most of which there is no freely available code with interface to a machine learning toolbox, and benchmark methods (order-agnostic baselines such as summary aggregation and chunking; distributional regression; naive baselines).

15. For example, if k is the Euclidean scalar product, $U^{(i)}$ can be taken as the raw data matrix, where rows are different time points and columns are features.

7.1. Validation and prediction set-up

7.1.1. Prediction tasks

In all data sets, samples are multi-variate (time) series. All learning tasks are supervised classification tasks of predicting class labels attached to series of equal length.

7.1.2. Prediction methods

For prediction, we use eps-support vector classification (as available in the python/scikit-learn package) on the kernel matrices obtained from the following kernels:

(1.a) the Euclidean kernel $\mathrm{k}(x, y) = \langle x, y \rangle$. This kernel has no parameters.

(1.b) the Gaussian kernel $\mathrm{k}(x, y) = \exp\left(\frac{1}{2}\gamma^2 \|x - y\|^2\right)$. This kernel has one parameter, a scaling constant γ.

(2.a) the (truncated) discretized signature kernel k_m^+ over the linear/Euclidean kernel $\mathrm{k}(x, y) = \gamma\langle x, y \rangle$. This kernel has two parameters, a scaling constant γ, and the truncation degree m.

(2.b) the (truncated) discretized signature k_m^+ over the Gaussian kernel $\mathrm{k}(x, y) = \theta \exp\left(\frac{1}{2}\gamma^2 \|x - y\|^2\right)$. This kernel has three parameters: scaling constants γ and θ, and truncation degree m.

(1.a) and (1.b) are considered standard kernels, (2.a) and (2.b) are discretized signature kernels. Note that the kernels (1.a) and (1.b) can only be applied to sequential data samples of equal length which is the case for the data sets considered. Even though (1.a), (1.b) may be applied to sequences of same length, they do not use any information about their ordering: both the Euclidean and the Gaussian kernel are invariant under (joint) permutation of the order of the indexing in the arguments. Another subtlety is that the discretized signature kernels (2.a), (2.b) do use information about the ordering of the sequences, but only for a truncation $m \geq 2$. For $m = 1$, the kernel corresponds to choosing the increment/mean aggregate feature (Euclidean) or a type of distributional classification (Gaussian). We will therefore explicitly compare truncation degrees 1 versus 2 and higher, to enable us to make a statement about whether using the order information was beneficial (or not).

7.1.3. Tuning and error estimation

In all experiments, we use nested (double) cross-validation for parameter tuning (inner loop) and estimation of error metrics (outer loop). In both instances of cross-validation, we perform uniform 5-fold cross-validation.

Unless stated otherwise, parameters are tuned on the tuning grid given in Table 3 (when applicable). Kernel parameters are the same as in the above section "prediction mehods". The best parameter is selected by 5-fold cross-validation, as the parameter yielding the minimum test-f1-score, averaged over the five folds.

parameter	range
kernel param. γ	0.01, 0.1, 1
kernel param. θ	0.01, 0.1, 1
truncation degree m	1,2,3
SVC regularizer	0.1, 1, 10, 100, 1000

Table 3: Tuning grid

7.1.4. Error metrics

The out-of-sample classification error is reported as precision, recall, and f1-score of out-of-sample prediction on the test fold. Errors measures are aggregated with equal weights on classes and folds. These aggregates are reported in the result tables.

7.2. Experiment: Classifying hand movements

We performed classification with the eps-support vector machine (SVC) on the hand movements data set from UCI Sapsanis et al. (2013). The first database in the data set which we considered for this experiment contains, for each of five subjects (two male, three female) 180 samples of hand movement sEMG recordings. Each sample is a time series in two variables (channels) at 3.000 time points. The time series fall into six classes of distinct types of hand movement (spherical, tip, palmar, lateral, cylindrical, hook). For each subject, 30 samples of each class were recorded. Hence, for each subject, there is a total of 180 sequences in $\mathcal{X}^{3000}$ with $\mathcal{X} = \mathbb{R}^2$.

For each of the five subjects, we conducted the classification experiment as described in Section 7.1, comparing prediction via SVC using one of the following kernels: (1.a) the Euclidean kernel, (1.b) the Gaussian kernel, (2.a) the sequentialized Euclidean kernel. For the non-sequential kernels (1.a), (1.b), prediction was performed with and without prior standardization of the data. For the sequential kernel, the tuning grid was considered in two parts: a truncation degree of $m = 1$, corresponding to mean aggregation, and truncation degrees of $m = 2, 3$, corresponding to the case where genuine sequence information is used. Further, we use the low-rank speed, Algorithm 5, to compute the Gram matrix.

The results are reported in Tables 4 to 8. Jackknife standard errors (pooling the five folds) are all 0.04 or smaller. Baseline performance of an uninformed estimator is $1/6 \approx 0.17$.

One can observe that for all five subjects, SVC with sequential kernel of degree 2 or higher

method	precision	recall	f1-score
(1.a) linear	0.37	0.38	0.36
(1.a) linear, standardized	0.33	0.32	0.29
(1.b) Gaussian	0.57	0.59	0.56
(1.b) Gaussian, standardized	0.54	0.50	0.50
(2.a) mean aggregation	0.19	0.20	0.18
(2.a) sequential, degree ≥ 2	0.87	0.86	0.86

Table 4: female1.mat

outperforms SVC using any of the other kernels not using any sequence information. The sequence kernel outperforms the reported methods from the original paper Sapsanis et al. (2013) as well (Figures 11 and 12).

method	precision	recall	f1-score
(1.a) linear	0.47	0.39	0.37
(1.a) linear, standardized	0.31	0.28	0.27
(1.b) Gaussian	0.71	0.71	0.70
(1.b) Gaussian, standardized	0.59	0.58	0.56
(2.a) mean aggregation	0.18	0.20	0.18
(2.a) sequential, degree ≥ 2	0.94	0.97	0.95

Table 5: female2.mat

method	precision	recall	f1-score
(1.a) linear	0.48	0.46	0.46
(1.a) linear, standardized	0.47	0.42	0.43
(1.b) Gaussian	0.66	0.64	0.63
(1.b) Gaussian, standardized	0.54	0.51	0.50
(2.a) mean aggregation	0.26	0.23	0.20
(2.a) sequential, degree ≥ 2	0.96	0.96	0.96

Table 6: female3.mat

method	precision	recall	f1-score
(1.a) linear	0.37	0.33	0.33
(1.a) linear, standardized	0.38	0.36	0.36
(1.b) Gaussian	0.59	0.57	0.57
(1.b) Gaussian, standardized	0.53	0.54	0.53
(2.a) mean aggregation	0.20	0.18	0.17
(2.a) sequential, degree ≥ 2	0.96	0.96	0.96

Table 7: male1.mat

method	precision	recall	f1-score
(1.a) linear	0.36	0.33	0.32
(1.a) linear, standardized	0.37	0.29	0.27
(1.b) Gaussian	0.72	0.71	0.70
(1.b) Gaussian, standardized	0.34	0.39	0.35
(2.a) mean aggregation	0.22	0.23	0.20
(2.a) sequential, degree ≥ 2	0.93	0.93	0.93

Table 8: male2.mat

7.3. Experiment: Pendigits

We performed classification on the pendgits data set from the UCI repository[16]. It contains 10992 samples of digits between 0 and 9 written by 44 different writers with a digital pen on a tablet. One sample consists of a pair of horizontal and vertical coordinates of sampled at 8 different time points, hence we deal with a sequence in $\mathcal{X}^8$ with $\mathcal{X} = \mathbb{R}^2$. The data set comes with a pre-specified training fold of 7494 samples, and a test fold of 3498 samples. Estimation of the prediction error is performed in this validation split, while tuning is done as described via nested 5-fold cross-validation, inside the pre-specified training fold.

We compared prediction via SVC using one of the following three kernels: (2.a) the sequentialized Euclidean kernel, and (2.b) the sequentialized Gaussian kernel. For both, the truncation degree was set to $m = 4$. The results are reported in Table 9. Jackknife standard errors (pooling the five folds) are all 0.01 or smaller. Baseline performance of an uninformed estimator is $1/10 \approx 0.10$.

16. `https://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits`

method\method	precision	recall	f1-score
sequential, linear	0.91	0.90	0.89
sequential, Gaussian	0.97	0.97	0.97

Table 9: Pendigits

The quality of the SVC prediction with the sequentialzed linear kernel is comparable to results reported in Diehl (2013). It is outperformed by SVC prediction with the sequentialized Gaussian kernel. The latter performance is similar to the baseline performance of k-nearest neighbors reported in the documentation of the pendigits data set.

Acknowledgments

FK acknowledges support by the Alan Turing Institute, under EPSRC grant EP/N510129/1. HO is grateful for support by the Oxford-Man Institute of Quantitative finance and would like to thank Ilya Chevyrev for a careful proofreading and helpful remarks and discussions.

Appendix A. Signatures, and proofs of Theorem 1 and Theorem 3

A.1. Signatures

Definition A.1 *Let E be a normed space and denote with $L(\mathcal{H}, E)$ the set of continuous linear maps from $\mathcal{H}$ to E. Given $x \in C^1([0,1], \mathcal{H})$ and $y \in C^1([0,1], L(\mathcal{H}, E))$, the Riemann–Stieltjes integral of y over $[a,b] \subseteq [0,1]$ is defined as the element in E given as*

$$\int_a^b y \, \mathrm{d}x := \lim_{\mathrm{mesh}(\pi) \to 0} \sum_{i=1}^{l} y(t_i) \left(x(t_{i+1}) - x(t_i) \right)$$

where the limit is taken over all partitions $\pi = \{a \leq t_1 < \cdots < t_l \leq b\}$ and $\mathrm{mesh}(\pi) := \max_{i=1,\ldots,l-1} |t_{i+1} - t_i|$. We also use the shorter notation $\int y \, \mathrm{d}x$ if the integration domain is clear from the context.

See (Lyons, 2004, Theorem 1.16) for a proof of a more general result. Applying the above with E being the subspace of $\prod_{m \geq 0} \mathcal{H}^{\otimes m}$ of square summable tensors and the linear map being tensor multiplication, gives the series

$$\mathrm{S}(x) := (\mathrm{S}_m(x))_{m \geq 0} := \left(\int \mathrm{d}x^{\otimes m} \right)_{m \geq 0} \in \prod_{m \geq 0} \mathcal{H}^{\otimes m}$$

with $\int \mathrm{d}x^{\otimes 0} := 1$ and $\int \mathrm{d}x^{\otimes (m+1)} := \int \mathrm{d}x^{\otimes m}$.

A.2. Proof of Theorem 1

This is more or less a folk theorem in rough path theory, but we recall it for readers not familiar with the subject. Point (1) of Theorem 1 follows since $\mathrm{S}(h)$ is itself a solution of a linear ODE driven by h and this solution map is continuous, see Lyons (2004). Point (2) is

more involved and we refer to Hambly and Lyons (2010). For the last point, Point (3), the key insight is that the series $\mathrm{S}(h)$ behaves like monomials.

Theorem 5 (Shuffle product) *Let $h \in C^1([0,1],\mathcal{H})$, then*

$$\int \mathrm{d}h^{\otimes m} \otimes \int \mathrm{d}h^{\otimes m'} = \sum_{\sigma} \sigma\left(\int \mathrm{d}h^{\otimes (m+m')}\right).$$

Here the sum is taken over all ordered shuffles $\mathrm{OS}_{m,m'}$ which are defined as

$$\left\{\sigma : \sigma \text{ permutation of } \left\{1,\dots,m+m'\right\}, \sigma(1) < \cdots < \sigma(m), \sigma(m+1) < \cdots < \sigma\left(m+m'\right)\right\}.$$

and $\sigma \in \mathrm{OS}_{m,m'}$ acts on $\mathcal{H}^{\otimes(m+m')}$ as $\sigma\left(e_{i_1}\otimes\cdots\otimes e_{i_{m+m'}}\right) = e_{\sigma(i_1)}\otimes\cdots\otimes e_{\sigma(i_{m+m'})}$.

The proof of above is given in (Lyons, 2004, Theorem 2.29 (ii)). A direct consequence by applying Stone–Weierstrass (in analogy to classic monomials) is that *linear* functionals of signatures approximate *nonlinear* functions of paths aribtrary well, thus showing Point (3) of Theorem 1.

A.3. Proof of Theorem 3 and Corollaries 4.3 and 4.4

Definition A.2 *Denote with $\Delta^{\ell}_{[a,b]}$ the n-simplex over the interval $[a,b]$, that is*

$$\Delta^{\ell}_{[a,b]} = \{(t_i)_{i=1}^{\ell} : a = t_1 < \cdots < t_\ell = 1\},$$

and with $\Delta_{[a,b]} = \bigcup_{\ell\geq 2} \Delta^{\ell}_{[a,b]}$. If the interval $[a,b]$ is clear from the context is clear, then we just write Δ^{ℓ} and Δ.

The following notation for sequences becomes useful.

Notation A.3 *For $n \geq 1$ denote*

1. *$[n] := \{1,\dots,n\}$,*
2. *sequences in $[n]$ as $\boldsymbol{i} = (i_1,\dots,i_l) \in [n]^l$ and call $|\boldsymbol{i}| = l$ the length of the sequence $\boldsymbol{i}$,*
3. *$d(\boldsymbol{i}) := \max\{r : i_1 = i_2 = \cdots = i_r\}$ the number of repetitions of the first element in the sequence $\boldsymbol{i}$ where by convention $d(\boldsymbol{i}) = 0$ if $i_1 \neq i_2$,*
4. *$\boldsymbol{i}! := n_1!\cdots n_k!$ if $\boldsymbol{i}$ consists of $k = |\{i_1,\dots,i_l\}|$ different elements in $[n]$ and $n_1,\dots,n_k$ denote the number of times they occur in $\boldsymbol{i}$,*
5. *$\boldsymbol{i} \sqsubset [n]$ if $\boldsymbol{i} = (i_1,\dots,i_l) \in [n]^l$ and $i_1 < \cdots < i_l$,*
6. *$\boldsymbol{i} \sqsubseteq [n]$ if $\boldsymbol{i} = (i_1,\dots,i_l) \in [n]^l$ and $i_1 \leq \cdots \leq i_l$,*
7. *$\boldsymbol{i} \sqsubseteq_d [n]$ if $\boldsymbol{i} \sqsubseteq [n]$ and no element in the sequence $\boldsymbol{i}$ appears more than than d times.*

Lemma A.4 *Let $h \in C^1([a,b],\mathcal{H})$, $\pi \in \Delta^{\ell}_{[a,b]}$ and define $h^{\pi} = (h(t_i))_{i=1,\dots,\ell} \in \mathcal{H}^+$. For any sequence $\boldsymbol{i} = (i_k)_{k=1,\dots,m} \in [\ell]^m$ with $1 \leq i_1 \leq \cdots \leq i_m \leq \ell$ denote $\Delta_{\boldsymbol{i}} := \Delta \cap \bigtimes_{k=1}^{m-1}[t_{i_k}, t_{i_k+1}]$. Then*

$$\mathrm{S}_m(h) - \mathrm{S}^+_m(h^{\pi}) = \sum_{\boldsymbol{i}} \int_{\Delta_{\boldsymbol{i}}} \mathrm{d}h^{\otimes m},$$

where the sum is taken over all $\boldsymbol{i} = (i_k)_{k=1}^m$ with $1 \leq i_1 \leq \cdots \leq i_m \leq \ell$ that have at least one repeating index, that is $i_k = i_{k+1}$ for at least one $1 \leq k \leq m-1$. Moreover, each term in above sum can be bounded as follows

$$\left\|\int_{\Delta_{\boldsymbol{i}}} \mathrm{d}h^{\otimes m}\right\|_{\mathcal{H}} \leq \frac{1}{\boldsymbol{i}!} \prod_{i\in\boldsymbol{i}} \|h|_{[t_i,t_{i+1}]}\|_1.$$

where we use $\boldsymbol{i}!$ as introduced in Notation A.3.

Proof We can decompose Δ into a disjoint union of $\Delta_{\boldsymbol{i}}$ over all sequences $\boldsymbol{i}$, with $1 \leq i_1 \leq \cdots \leq i_m \leq \ell$, hence

$$\mathrm{S}_m(h) = \int \mathrm{d}h^{\otimes m} = \sum_{\boldsymbol{i}} \int_{\Delta_{\boldsymbol{i}}} \mathrm{d}h^{\otimes m}.$$

Split the sum $\sum_{\boldsymbol{i}}$ into a sum over $\boldsymbol{i}$ that have an repeating element, i.e. $i_k = i_{k+1}$ for at least one k, and a sum $\sum_{\boldsymbol{i}}$ over indices that do not have repeating elements. For the latter, the integration can be done explicitly and this sum equals

$$\sum_{\boldsymbol{i}} \nabla_{i_1} h \otimes \cdots \otimes \nabla_{i_m} h$$

with our usual notation $\nabla_i h = h(t_{i+1}) - h(t_i)$. However, this is exactly $\mathrm{S}^+_m(h)$, hence the first statement follows.

For the first sum over sequences $\boldsymbol{i}$ with repeating elements, denote with $\bar{i}_1,\dots,\bar{i}_k$ the distinct indices in $\boldsymbol{i}$, and $n_1,\dots,n_k$ the total counts of their respective occurrences. Then $\Delta_{\boldsymbol{i}} = \bigtimes_{j=1}^k \bar{\Delta}_j$ with $\bar{\Delta}_j := \Delta^{n_j} \cap [t_{\bar{i}_j}, t_{\bar{i}_j+1}]^{n_j}$ and it follows that

$$\int_{\Delta_{\boldsymbol{i}}} \mathrm{d}h^{\otimes m} = \bigotimes_{j=1}^k \int_{\bar{\Delta}_j} \mathrm{d}h^{\otimes n_j}.$$

Therefore, we obtain

$$\left\|\bigotimes_{j=1}^k \int_{\bar{\Delta}_j} \mathrm{d}h^{\otimes n_j}\right\|_{\mathcal{H}} = \prod_{j=1}^k \left\|\int_{\bar{\Delta}_j} \mathrm{d}h^{\otimes n_j}\right\|_{\mathcal{H}} \leq \frac{1}{\boldsymbol{i}!} \prod_{i\in\boldsymbol{i}} \|h|_{[t_i,t_{i+1}]}\|_1.$$

■

This is enough to prove our main approximation result:

Theorem 6 *Let $h \in C^1([0,1],\mathcal{H})$ let $\pi \in \Delta^{\ell}$. Then for every $m \geq 0$*

$$\|\mathrm{S}_m(h) - \mathrm{S}^+_m(h^{\pi})\|_{\mathcal{H}^{\otimes m}} \leq g_m$$

where (g_m) are the coefficients in the Taylor expansion around 0 of the function

$$g(z) := \sum_{m=1}^{\infty} g_m \cdot z^m := \exp(z \cdot \|h\|_1) - \prod_{i=1}^{\ell-1} \left(1 + z \cdot \|h|_{[t_i,t_{i+1}]}\|_1\right).$$

Moreover,

$$\|\operatorname{S}(h) - \operatorname{S}^+(h^\pi)\| \leq \exp(\|h\|_1) - \prod_{i=1}^{\ell-1} \left(1 + \|h|_{[t_i,t_{i+1}]}\|_1\right)$$

If π is chosen such that $\|h|_{[t_i,t_{i+1}]}\|_1 = \frac{1}{\ell-1}\|h\|_1$ for all $i = 1, \dots, \ell-1$, then

$$\|\operatorname{S}(h) - \operatorname{S}^+(h)\| \leq \frac{\exp(\|h\|_1)}{\ell - 1} \left(1 + \frac{\|h\|_1^{\ell-1}}{(\ell-3)!}\right).$$

Proof The bound for $\|\operatorname{S}_m(h) - \operatorname{S}_m^+(h^\pi)\|_{\mathcal{H}^{\otimes m}}$ follows from Lemma A.4 by explicitly writing out the coefficient g_m. Applying the triangle inequality and using this bound and one obtains

$$\|\operatorname{S}(h) - \operatorname{S}^+(h^\pi)\| \leq \sum_{m=1}^{\infty} \|\operatorname{S}_m(h) - \operatorname{S}_m^+(h^\pi)\|_{\mathcal{H}^{\otimes m}} \leq \sum_{m=1}^{\infty} g_m = g(1)$$

Finally, for the special choice of a uniform partition π

$$g(1) = \exp\|h\|_1 - \left(1 + \frac{\|h\|_1}{\ell-1}\right)^{\ell-1},$$

to which we apply Euler's approximation theorem, Theorem 7, below using that $\|h|_{[a,b]}\|_1 + \|h|_{[b,c]}\|_1 = \|h|_{a,c}\|_1$. ■

The first part of Theorem 6 implies Theorem 3, the second part implies Corollary 4.4. For readers' convenience we recall Euler's well-known approximation to the exponential function.

Theorem 7 *Let $x \in \mathbb{R}, n \in \mathbb{N}$. Then,*

$$\left(1 + \frac{x}{n}\right)^n - \exp(x) = g(x,n), \quad \text{with } |g(x,n)| \leq \frac{\exp(x)}{n}\left(1 + \frac{x^n}{(n-2)!}\right).$$

In particular, it holds that

$$\lim_{n\to\infty} \left(1 + \frac{x}{n}\right)^n = \exp(x),$$

where convergence is uniform of order $O(n^{-1})$ on any compact subset of $\mathbb{R}$.

Proof All statements follow from the first, which we proceed to prove. By the binomial theorem, it holds that

$$\left(1 + \frac{x}{n}\right)^n = \sum_{k=0}^{n} \binom{n}{k} \cdot \frac{x^k}{n^k}.$$

From the definition of the binomial coefficient and an elementary computation, one obtains

$$\binom{n}{k} \cdot \frac{x^k}{n^k} = \frac{x^k}{k!} + g(x,n,k), \text{ where } \|g(x,n,k)\| \leq \frac{x^k}{k!n},$$

for $k \leq n$. For $k \geq n$, one has

$$\frac{x^k}{k!} \leq \frac{x^n}{n!} \cdot \frac{x^{k-n}}{(k-n)!}.$$

Putting together all inequalities and using the Taylor expansion of exp yields the claim. ■

To prove Corollary 4.3 we need to generalize Euler's approximation of $\exp x$ to the case when x is not divided into uniform parts, but potentially very unbalanced parts. This is done in the proposition below

Proposition A.5 *Let $x \in \mathbb{R}, n \in \mathbb{N}$, $x \geq 0$. Let $x_1, \dots, x_\ell \in \mathbb{R}$, $x_i \geq 0$ for $i = 1, \dots, \ell$ such that $\sum_{i=1}^{\ell} x_i = x$. Then,*

$$\exp(x) = \prod_{i=1}^{\ell} (1 + x_i) + g(x, x_1, \dots, x_\ell), \quad \text{where } 0 \leq g(x, x_1, \dots, x_\ell) \leq x \exp(x) \cdot \max_{i=1,\dots,\ell} x_i.$$

In particular, it holds that

$$\lim_{\max_i x_i \to 0} \prod_i (1 + x_i) = \exp(x),$$

where convergence is uniform of order $O(\max_i x_i)$ on any compact subset of $[0, \infty)$.

Proof All statements follow from the first, which we proceed to prove. We use Notation A.3 Writing out the product, we obtain

$$\prod_{i=1}^{\ell} (1 + x_i) = \sum_{\boldsymbol{i} \sqsubset [\ell]} x^{\boldsymbol{i}},$$

where abbreviatingly we have written $x^{\boldsymbol{i}} := \prod_{i \in \boldsymbol{i}} x_i$. The Taylor expansion of the exponential on the other hand yields

$$\exp(x) = \exp\left(\sum_{i=1}^{\ell} x_i\right) = \sum_{\boldsymbol{i} \sqsubset [\ell]} \frac{1}{\boldsymbol{i}!} x^{\boldsymbol{i}}.$$

Note the major different between both sums above being the repeating indices which may occur in the expansions of $\exp(x)$. More precisely, we obtain

$$\exp(x) - \prod_{i=1}^{m} (1 + x_i) = \sum_{\substack{\boldsymbol{i} \sqsubset [\ell] \\ \boldsymbol{i}! > 1}} \frac{1}{\boldsymbol{i}!} x^{\boldsymbol{i}}.$$

We further split up the sum by length of $\boldsymbol{i}$:

$$\exp(x) - \prod_{i=1}^{\ell}(1+x_i) = \sum_{m=2}^{\infty} \sum_{\substack{\boldsymbol{i}\sqsubseteq[\ell] \\ |\boldsymbol{i}|=m \\ \boldsymbol{i}!>1}} \frac{1}{\boldsymbol{i}!} x^{\boldsymbol{i}}.$$

Positivity of g follows from this equation and positivity of x. Now consider the map ϕ which removes the first duplicated index in an ordered index sequence $\boldsymbol{i}$ yielding a sequence of length $|\boldsymbol{i}| - 1$. On sequences of length m, the map ϕ is at most m-to-one, and surjective onto sequences of length $m-1$. Therefore,

$$\sum_{\substack{\boldsymbol{i}\sqsubseteq[\ell] \\ |\boldsymbol{i}|=m \\ \boldsymbol{i}!>1}} \frac{1}{\boldsymbol{i}!} x^{\boldsymbol{i}} \leq X \cdot \sum_{m=2}^{\infty} \frac{m}{2} \sum_{\substack{\boldsymbol{i}\sqsubseteq[\ell] \\ |\boldsymbol{i}|=m-1}} \frac{1}{\boldsymbol{i}!} x^{\boldsymbol{i}},$$

where $X = \max_i x_i$. Thus,

$$\sum_{m=2}^{\infty} \sum_{\substack{\boldsymbol{i}\sqsubseteq[\ell] \\ |\boldsymbol{i}|=m \\ \boldsymbol{i}!>1}} \frac{1}{\boldsymbol{i}!} x^{\boldsymbol{i}} \leq X \cdot \sum_{m=1}^{\infty} m \cdot \sum_{\substack{\boldsymbol{i}\sqsubseteq[\ell] \\ |\boldsymbol{i}|=m}} \frac{1}{\boldsymbol{i}!} x^{\boldsymbol{i}}.$$

Comparing to the expansion of $\exp(x)$ above, one observes that the right hand side is equal to

$$X \cdot \sum_{m=1}^{\infty} m \cdot \frac{x^m}{m!} = X \cdot x \sum_{m=0}^{\infty} \frac{x^m}{m!} = X \cdot x \cdot \exp(x).$$

■

The bounds in Proposition A.5 are worse than those from Euler's classic approximation result to the exponential in the case of equal x_i, by a factor of x. This is due to the fact that the bound also needs to be valid for heavily imbalanced partitions of x into x_i.

Appendix B. Higher order signature kernels and noisy observations

The discretized signature kernel was defined as inner product

$$\mathrm{k}^+ : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R} \quad \mathrm{k}^+(x,y) = \langle \mathrm{S}^+(\mathrm{k}_x), \mathrm{S}^+(\mathrm{k}_y) \rangle$$

of features S^+ that approximate the signature S, thus k^+ will approximate the signature kernel

$$\mathrm{k}^{\oplus} : \mathcal{P}_{\mathcal{X}} \times \mathcal{P}_{\mathcal{X}} \to \mathbb{R} \quad \mathrm{k}^{\oplus}(x,y) = \langle \mathrm{S}(\mathrm{k}_x), \mathrm{S}(\mathrm{k}_y) \rangle$$

when the sequences are discretizations of paths: in Section 4 we showed under Assumption 3.1 that

$$\mathrm{k}^+(x^\pi, y^{\pi'}) \to \mathrm{k}^{\oplus}(x,y) \text{ as } \max(\mathrm{mesh}(\pi), \mathrm{mesh}(\pi')) \to 0.$$

For essentially all practically relevant kernels $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ Assumption 3.1 holds whever the underlying paths x, y are of bounded variation. However, a common situation is that observations are perturbed by noise which are generically of unbounded variation.

Example B.1 *Let $\mathcal{X} = \mathbb{R}$ and $\mathrm{k}(x,y) = \langle x, y \rangle_{\mathbb{R}}$ (that is $\mathcal{H} = \mathbb{R}$). If $x \in C^1([0,1], \mathcal{X})$ and $B = (B_t)_{t\in[0,1]}$ is a Brownian motion in $\mathcal{X}$ then $\mathrm{S}(\mathrm{k}_{x+B}) \simeq \mathrm{S}(x+B) = \left(\int \mathrm{d}(x+B)^{\otimes m}\right)$ is not well-defined as a Riemann–Stieltjes integral.*

A general strategy is to replace the signature map $x \mapsto \mathrm{S}(x)$ by a a map from paths to $\prod_{m\geq 0} \mathcal{H}^{\otimes m}$ as follows: construct a sequence of bounded variation paths (x_n) that approximation x and such that $t \mapsto \left(\int_0^t \mathrm{d}x_n^{\otimes m}\right)$ converges to a $\prod_{m\geq 0}\mathcal{H}^{\otimes m}$-valued path which is called the "*(geometric) rough path lift*" of x.

Example B.2 *Constructions for mapping a path to an element of $\prod_{m\geq 0}\mathcal{H}^{\otimes m}$ are known for a wide range of stochastic processes; below we mention a few*

- Brownian motion *(leading to p-rough paths for any $p > 2$)*
- *more generally, continuous* Semimartingale *(leading to p-rough paths for any $p > 2$),*
- fractional Brownian motion *of Hurst parameter $H > \frac{1}{4}$,*
- *more generally,* Gaussian processes *(leading to p-rough paths where p depends on the regulary of the covariation process),*
- Markov processes *in continuous time (leading to p-rough paths with p depending on the generator of the Markov process).*

For details and more examples see Lyons (2004).

The notion of a geometric rough path allows to study classes of much "rougher" paths by the same approach we developed for the bounded variation case: below we provide the needed modifications for obtaining a sequentialized kernel such that $\mathrm{k}^+(x^\pi, y^{\pi'})$ is still well-defined as the mesh vanishes when x, y are paths of unbounded variation but a "rough path lift" exists (e.g. all the stochastic processes in Example B.2)

Remark B.3 *A more radical approach is to assume that the we are not only given paths as data but also the first levels of their "rough path lift", see for example Crisan et al. (2013). The same recursion applies to give a kernel, however, we do not spell out details since all benchmark data sets we are aware of, only provide path increments.*

Definition B.4 *Let $d \in \mathbb{N}$. We call the map*

$$\mathrm{S}^+_{(d)} : \mathcal{H}^+ \to \prod_{m\geq 0} \mathcal{H}^{\otimes m}, \quad (h_i)_{i=1}^{\ell} \mapsto \prod_{i=1}^{\ell} \sum_{j=0}^{d} \frac{(\nabla_i h)^{\otimes j}}{j!}$$

the discretized signature map of degree d. We denote with $\mathrm{S}^+_{(d,m)}$ the projection of $\mathrm{S}^+_{(d)}$ to $\bigoplus_{n=0}^{m} \mathcal{H}^{\otimes n}$.

Definition B.5 *Let* $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *and* $d, m \geq 1$, $d \geq m$. *The discretized signature kernel over* k *of order* d *and at degree* m *is defined as*

$$\mathrm{k}^+_{(d,m)}(x,y) : \mathcal{X}^+ \times \mathcal{X}^+ \to \mathbb{R}, \quad \mathrm{k}^+_{(d,m)}(x,y) = \langle \mathrm{S}^+_{(d,m)}(\mathrm{k}_x), \mathrm{S}^+_{(d,m)}(\mathrm{k}_y) \rangle.$$

Remark B.6

- *The sequentialization* k^+_m *from Section 4 arises as special case of above, more general, definition:* $\mathrm{k}^+_m = \mathrm{k}^+_{(d,m)}$ *for* $d = 1$, $m \in \mathbb{N}$.
- *Readers familiar with rough paths will notice that the choice* $m = d = \lfloor p \rfloor$ *recovers the notion of* $(\mathrm{k}_{x(t)})_{t\in[0,1]}$ *lifted to a geometric* p*-rough path. However, for machine learning applications it is often beneficial to take* $d = \lfloor p \rfloor$ *to ensure convergence but to consider* $m > d$ *and find the optimal degree* m *from the data (e.g. by cross-validation).*

In analogy to the order $d = 1$ approximations discussed in Section 4, the central mathematical identity is now

$$\prod_{i=1}^{\ell-1} \sum_{m=0}^{d} \frac{(x(t_{i+1}) - x(t_i))^{\otimes m}}{m!} = \sum_{\boldsymbol{i} \sqsubseteq_d [\ell-1]} \frac{1}{\boldsymbol{i}!} \prod_{r=1}^{|\boldsymbol{i}|} (x(t_{i_r+1}) - x(t_i)) \approx \left(\int \mathrm{d}x^{\otimes m} \right)_{m \geq 0}$$

where we use Notation A.3.

Proposition B.7 *For* $x, y \in \mathcal{X}^+$, $d, m \geq 1$, $m \geq d$,

(a) $\mathrm{S}^+_{(d,m)}(\mathrm{k}_x) = 1 + \sum_{n=1}^{m} \sum_{\substack{\boldsymbol{i} \sqsubseteq_d [|x|-1] \\ |\boldsymbol{i}|=n}} \frac{1}{\boldsymbol{i}!} \prod_{r=1}^{|\boldsymbol{i}|} \nabla_{i} \mathrm{k}_x$,

(b) $\mathrm{k}^+_{(d,m)}(x,y) = 1 + \sum_{n=1}^{m} \sum_{\substack{\boldsymbol{i} \sqsubseteq_d [|x|-1] \\ \boldsymbol{j} \sqsubseteq_d [|y|-1] \\ |\boldsymbol{i}|=|\boldsymbol{j}|=n}} \frac{1}{\boldsymbol{i}!\boldsymbol{j}!} \prod_{r=1}^{|\boldsymbol{i}|} \nabla_{i_r, j_r} \mathrm{k}(x,y)$.

where $\nabla_i \mathrm{k}_x := \mathrm{k}_{x_{i+1}} - \mathrm{k}_{x_i} \in \mathcal{H}$.

Proof By definition

$$\mathrm{S}^+_{(d)}(\mathrm{k}_x) = \prod_{r=1}^{|x|-1} \sum_{n=0}^{d} \frac{1}{n!} (\nabla_r \mathrm{k}_x)^{\otimes n}.$$

Application of the (non-commutative) associative law yields

$$\prod_{r=1}^{|x|-1} \sum_{n=0}^{d} \frac{1}{n!} (\nabla_r \mathrm{k}_x)^{\otimes n} = \sum_{\boldsymbol{i} \sqsubseteq_d [|x|-1]} \frac{1}{\boldsymbol{i}!} \prod_{r=1}^{|\boldsymbol{i}|} \nabla_{i_r} \mathrm{k}_x \,.$$

Using the analogous expression for $\mathrm{S}^+_{(d)}(y)$, taking the scalar product while noting

$$\langle \prod_{r=1}^{|\boldsymbol{i}|} \nabla_{i_r} \mathrm{k}_x, \prod_{r'=1}^{|\boldsymbol{j}|} \nabla \mathrm{k}_{j_{r'}} \rangle = \delta_{|\boldsymbol{i}|,|\boldsymbol{j}|} \cdot \prod_{r=1}^{|\boldsymbol{i}|} \langle \nabla \mathrm{k}_{i_r}, \nabla_{j_r} \mathrm{k}_y \rangle.$$

and truncating at tensor degree m and restricting to $\boldsymbol{i} \sqsubseteq_d [|x|-1]$ and $\boldsymbol{j} \sqsubseteq_d [|y|-1]$ yields the claim. ∎

Remark B.8 *The appearance of the* $\boldsymbol{i}!$ *term together with* $\boldsymbol{i} \sqsubseteq_d [|x|-1]$ *makes a recursion formula more complex since one needs to keep track how many elements in* $\boldsymbol{i}$ *are equal in every recursion step. We give an effective algorithm in Section B.1 that relies on multi-way recursion.*

B.1. Computing the higher order discretized signature kernel

Algorithm 6 Evaluation of $\mathrm{k}^+_{(d,m)}$.
Input: Sequences $x, y \in \mathcal{X}^+$. A kernel $\mathrm{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A truncation degree m. An approximation order d with $1 \leq d \leq m$.
Output: $\mathrm{k}^+_{(d,m)}(x,y)$

Let $\ell_x \leftarrow |x| - 1$ and $\ell_y \leftarrow |y| - 1$
Compute an $(\ell_x \times \ell_y)$ array K such that $K[i,j] = \nabla_{i,j} \mathrm{k}(x,y)$
Initialize an $(m \times d \times d \times \ell_x \times \ell_y)$-array A, all entries zero
for $n = 2$ to m **do**
  $d' \leftarrow \min(d, n-1)$
  $A[n|1,1|:,:] \leftarrow K \cdot (1 + A[n-1|\Sigma,\Sigma|\boxplus+1,\boxplus+1])$
  for $r = 2$ to d' **do**
    $A[n|r,1|:,:] \leftarrow \frac{1}{r} \cdot K \cdot A[n-1|r-1,\Sigma|:,\boxplus+1]$
    $A[n|1,r|:,:] \leftarrow \frac{1}{r} \cdot K \cdot A[n-1|\Sigma,r-1|\boxplus+1,:]$
    for $s = 2$ to d' **do**
      $A[n|r,s|:,:] \leftarrow \frac{1}{rs} \cdot K \cdot A[n-1|r-1,s-1|:,:]$
    end for
  end for
end for
Compute $R \leftarrow 1 + A[m|\Sigma,\Sigma|\Sigma,\Sigma]$
Return R

Recall that multiplications of arrays in Algorithm 6 are entry-wise (not matrix multiplications). At the end of Algorithm 6, the array A contains as elements $A[m|r,s|i,j]$ the contributions from sub-sequences $\boldsymbol{i} \sqsubseteq [\ell_x], \boldsymbol{j} \sqsubseteq [\ell_y]$, beginning at i and j, with start-sequences $iii\ldots$ of length r and $jjj\ldots$ of length s, and of total length at most m.

Proposition B.9 *Let* $x, y \in \mathcal{X}^+$, $m \geq 1$ *and* d *with* $1 \leq d \leq m$. *Define* $\ell_x = |x| - 1$, $\ell_y = |y| - 1$ *and*

$$A^{n,r,s}_{i,j} := \sum_{\substack{\boldsymbol{i} \sqsubseteq_d [\ell_x] \\ \boldsymbol{j} \sqsubseteq_d [\ell_y] \\ |\boldsymbol{i}|=|\boldsymbol{j}|=n}} \frac{1}{\boldsymbol{i}!\boldsymbol{j}!} \prod_{\kappa=1}^{n} \nabla_{i_\kappa, j_\kappa} \mathrm{k}(x_{i_\kappa}, y_{j_\kappa})$$

for every $r, s \geq 1$ *and* $i \in [\ell_x]$, $j \in [\ell_y]$ *where the sum over* $\boldsymbol{i} = (i_1, \ldots, i_n), \boldsymbol{j} = (j_1, \ldots, j_n)$ *is additionally restricted in the following way:*

$$i = i_1 = i_2 = \cdots = i_r \neq i_{r+1} \quad \textit{and} \quad j = j_1 = j_2 = \cdots = j_s \neq j_{s+1}.$$

By convention, set $A_{i,j}^{n,r,s} = 0$ *if* $r < n$ *or* $s < n$*. Then,*

$$\mathrm{k}^{+}_{(d,m)}(x,y) = 1 + \sum_{n=1}^{m}\sum_{i=1}^{\ell_x}\sum_{j=1}^{\ell_y}\sum_{r=1}^{d}\sum_{s=1}^{d} A_{i,j}^{n,r,s} \tag{5}$$

and the following equalities hold

$$A_{i,j}^{n,r,s} = \frac{1}{rs}\cdot\nabla_{i,j}\,\mathrm{k}(x,y)\cdot A_{i,j}^{n-1,r-1,s-1}, \tag{6}$$

$$A_{i,j}^{n,1,1} = \nabla_{i,j}\,\mathrm{k}(x,y)\cdot\left(1+\sum_{i'>i}\sum_{j'>j}\sum_{r=1}^{d}\sum_{s=1}^{d} A_{i',j'}^{n-1,r,s}\right), \tag{7}$$

$$A_{i,j}^{n,r,1} = \frac{1}{r}\cdot\nabla_{i,j}\,\mathrm{k}(x,y)\cdot\sum_{j'>j}\sum_{s=1}^{d} A_{i,j'}^{n-1,r-1,s}, \tag{8}$$

$$A_{i,j}^{n,1,s} = \frac{1}{s}\cdot\nabla_{i,j}\,\mathrm{k}(x,y)\cdot\sum_{i'>i}\sum_{r=1}^{d} A_{i',j}^{n-1,r,s-1}, \tag{9}$$

for $r,s \geq 2$*,* $i \in [\ell_x]$*,* $j \in [\ell_y]$*.*

Proof By Proposition B.7

$$\begin{aligned}\mathrm{k}^{+}_{(d,m)}(x,y) &= 1 + \sum_{n=1}^{m}\sum_{\substack{\boldsymbol{i}\sqsubseteq_d[\ell_x]\\ \boldsymbol{j}\sqsubseteq_d[\ell_y]\\ |\boldsymbol{i}|=|\boldsymbol{j}|=n}}\frac{1}{\boldsymbol{i}!\boldsymbol{j}!}\prod_{r=1}^{n}\nabla_{i_r,j_r}\,\mathrm{k}(x,y)\\ &= 1 + \sum_{\substack{\boldsymbol{i}\sqsubseteq_d[\ell_x]\\ \boldsymbol{j}\sqsubseteq_d[\ell_y]}}\frac{1}{\boldsymbol{i}!\boldsymbol{j}!}\nabla_{i_1,j_1}\,\mathrm{k}(x,y)\,(1+\nabla_{i_2,j_2}\,\mathrm{k}(x,y)\,(1+\cdots(1+\nabla_{i_m,j_m}\,\mathrm{k}(x,y)))\cdots)\end{aligned}$$

and the first equality shows the identity (5) by explicitly summing over that starting point of sequences and the number of repeated elements. Equality (6) follows directly by definition of $A_{i,j}^{n,r,s}$ and that for $\boldsymbol{i} = (i_1, i_2, \ldots, i_n)$ with $d(\boldsymbol{i}) = r$ and $\boldsymbol{i}' = (i_2, \ldots, i_n)$ we have $\boldsymbol{i}! = \frac{1}{r}(\boldsymbol{i}'!)$. Equality (7) follows since

$$\begin{aligned}\{\boldsymbol{i} = (i_1,\ldots,i_n) : d(\boldsymbol{i}) = 0,\, i_1 = i\} &= \{(i,\boldsymbol{i}') : \boldsymbol{i}' = (i_1',\ldots,i_{n-1}'),\, i_1' > i\}\\ &= \bigcup_{r=0}^{d}\{(i,\boldsymbol{i}') : \boldsymbol{i}' = (i_1',\ldots,i_{n-1}'),\, i_1' > i,\, d(\boldsymbol{i}') = r\}\end{aligned}$$

where in above equalities we implicitly additionally assume that all indices $\boldsymbol{i}$, $\boldsymbol{i}'$ are increasing (that is, $i_1 \leq \cdots \leq i_n$, $i_1' \leq \cdots \leq i_{n-1}'$). Equalities (8),(9) follow similarly. ■

Remark B.10 *The computational cost of Algorithm 6 is* $O(d^2 m|x||y|)$ *elementary arithmetic operations (= the number of loop elements) and* $O(d^2|x||y|)$ *units of elementary storage (when freeing up space for array entries directly after the last time they are read out).*

Remark B.11 *The higher order Algorithm 6 can be combined with the low-rank techniques analogous to the treatment of the order* $d = 1$ *algorithm in Section 6 by applying the low-rank representation to the matrices/2D-arrays* $A[m|i,j|\,:,:]$*. Since all assignments and manipulations can be re-phrased in those matrices, the same strategy applies.*

References

Claus Bahlmann, Bernard Haasdonk, and Hans Burkhardt. Online handwriting recognition with support vector machines-a kernel approach. In *Frontiers in handwriting recognition, 2002. proceedings. eighth international workshop on*, pages 49–54. IEEE, 2002.

Dan Crisan, Joscha Diehl, Peter Friz, and Harald Oberhauser. Robust filtering: multdimensional noise and multidimensional observation. *Annals of Applied Probability*, 23(5): 2139–2160, 2013.

Felipe Cucker and Steve Smale. On the mathematical foundations of learning. *Bulletin of the American mathematical society*, 39(1):1–49, 2002.

Marco Cuturi. Fast global alignment kernels. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 929–936, 2011.

Marco Cuturi, J-P Vert, Oystein Birkenes, and Takashi Matsui. A kernel for time series based on global alignments. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 2, pages II–413. IEEE, 2007.

Joscha Diehl. Rotation invariants of two dimensional curves based on iterated integrals. *CoRR*, abs/1305.6883, 2013. URL `http://arxiv.org/abs/1305.6883`.

Toni Giorgino. Computing and visualizing dynamic time warping alignments in r: the dtw package. *Journal of statistical Software*, 31(7):1–24, 2009.

Ben Hambly and Terry J Lyons. Uniqueness for the signature of a path of bounded variation and the reduced path group. *Annals of Mathematics*, 171(1):109–167, 2010.

David Haussler. Convolution kernels on discrete structures. Technical report, Citeseer, 1999.

Joseph B Kruskal. An overview of sequence comparison: Time warps, string edits, and macromolecules. *SIAM review*, 25(2):201–237, 1983.

Christina Leslie and Rui Kuang. Fast string kernels using inexact matching for protein sequences. *Journal of Machine Learning Research*, 2004.

Daniel Levin, Terry J Lyons, Hao Ni, et al. Learning from the past, predicting the statistics for the future, learning an evolving system. *arXiv preprint arXiv:1309.0260*, 2013.

Huma Lohdi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2002.

JMLR 20(31):1-45, 2019

Terry J Lyons. *Differential equations driven by rough paths*. Springer Berlin Heidelberg New York, 2004.

Terry J Lyons, Hao Ni, and Harald Oberhauser. A feature set for streams and an application to high-frequency financial tick data. In *Proceedings of the 2014 International Conference on Big Data Science and Computing*, page 5. ACM, 2014.

Hiroshi Shimodaira Ken-ichi Noma. Dynamic time-alignment kernel in support vector machine. *Advances in neural information processing systems*, 14:921, 2002.

Anastasia Papavasiliou, Christophe Ladroue, et al. Parameter estimation for rough differential equations. *The Annals of Statistics*, 39(4):2047–2073, 2011.

Hiroaki Sakoe. Two-level dp-matching–a dynamic programming-based pattern matching algorithm for connected word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 27(6):588–595, 1979.

Hiroaki Sakoe and Seibi Chiba. A similarity evaluation of speech patterns by dynamic programming. In *Nat. Meeting of Institute of Electronic Communications Engineers of Japan*, page 136, 1970.

Christos Sapsanis, George Georgoulas, and Anthony Tzes. EMG based classification of basic hand movements based on time-frequency features. In *Control & Automation (MED), 2013 21st Mediterranean Conference on*, pages 716–722. IEEE, 2013.

Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

Kilho Shin and Tetsuji Kuboyama. A generalization of Haussler's convolution kernel: mapping kernel. In *Proceedings of the 25th international conference on Machine learning*, pages 944–951. ACM, 2008.

Weixin Yang, Lianwen Jin, and Manfei Liu. Character-level chinese writer identification using path signature feature, dropstroke and deep CNN. abs/1505.04922, 2015. URL `http://arxiv.org/abs/1505.04922`.

Journal of Machine Learning Research 20 (2019) 1-50 Submitted 2/17; Revised 12/18; Published 2/19

NetSDM: Semantic Data Mining with Network Analysis

Jan Kralj JAN.KRALJ@IJS.SI
Jožef Stefan Institute, Department of Knowledge Technologies,
Jamova 39, 1000 Ljubljana, Slovenia

Marko Robnik-Šikonja MARKO.ROBNIK@FRI.UNI-LJ.SI
University of Ljubljana, Faculty of Computer and Information Science,
Večna pot 113, 1000 Ljubljana, Slovenia

Nada Lavrač NADA.LAVRAC@IJS.SI
Jožef Stefan Institute, Department of Knowledge Technologies,
Jamova 39, 1000 Ljubljana, Slovenia

Editor: Bert Huang

Abstract

Semantic data mining (SDM) is a form of relational data mining that uses annotated data together with complex semantic background knowledge to learn rules that can be easily interpreted. The drawback of SDM is a high computational complexity of existing SDM algorithms, resulting in long run times even when applied to relatively small data sets. This paper proposes an effective SDM approach, named NetSDM, which first transforms the available semantic background knowledge into a network format, followed by network analysis based node ranking and pruning to significantly reduce the size of the original background knowledge. The experimental evaluation of the NetSDM methodology on acute lymphoblastic leukemia and breast cancer data demonstrates that NetSDM achieves radical time efficiency improvements and that learned rules are comparable or better than the rules obtained by the original SDM algorithms.

Keywords: data mining, semantic data mining, ontologies, subgroup discovery, network analysis

1. Introduction

In data mining applications, relevant information can be scattered across disparate resources in heterogeneous data formats. Extraction of knowledge from large heterogeneous data is technically challenging and time consuming, and presents a significant obstacle to wider adoption of data mining in real-life applications. While most of the data mining algorithms (Witten and Frank, 2005) work only with tabular data, *relational data mining* (RDM) (Džeroski and Lavrač, 2001) and *relational learning* (De Raedt, 2008) algorithms can use additional relational information about the analyzed data to build richer and more accurate predictive models. One such form of additional information is domain knowledge (relational background knowledge) about instances of a data set. As shown by Page et al. (2012) and Peissig et al. (2014), relational data mining can significantly outperform classical machine learning approaches.

Previous work on RDM has shown that the use of *background knowledge* is vital for data analysis in many domains. However, the issue with using more complex data (i.e., data along with relations and background knowledge) is that it can no longer be easily represented in a tabular format (also referred to as a design matrix (Rendle, 2013)). Better data representations are needed when instances under examination are interconnected to a various (non-fixed) number of instances. Representing each connection from an instance as a separate column would result in a different number of columns for each row. Alternatively, if we encoded connections of an instance with a single column, columns would have to contain composite data structures such as lists. In RDM, this problem is addressed by representing data sets with *multi-relational databases* and by applying specialized relational data mining algorithms for data analysis (Džeroski and Lavrač, 2001). Examples of such instances are genes that are connected through mutual activation, or research papers connected through citations.

An alternative way to describe a data set containing inter-connected instances is to represent it as a *network* (Burt and Minor, 1983), a data structure containing nodes, information about the nodes, and connections between the nodes. In mathematical terms, such a structure is described as a *graph*, where the nodes are referred to as vertices, and the connections as edges. The field of *network analysis* deals with several types of networks. In social networks, nodes represent people and connections represent their social links. Biological networks include gene regulatory networks, protein interaction networks, drug interaction networks, etc. In information networks, directed connections encode information flow between the network nodes. While most researchers deal with *homogeneous information networks*, where all nodes and edges are of the same node/edge type, Sun and Han (2012) addressed the problem of *heterogeneous information network* analysis, where nodes and edges belong to different node or edge types. For example, we may have a network containing both the genes and the proteins they encode, which necessitates the use of two node types to represent the data. Sun and Han (2012) introduced the concept of *authority ranking* for heterogeneous information networks, where the impact of a node is transferred along the edges to simultaneously rank nodes of different types, while class labels can also be propagated through the network (Vanunu et al., 2010).

Network analysis can be considered as a form of relational learning, where instances are linked by relations and connected in a complex graph. Special form of relational data are *ontologies* (Guarino et al., 2009). The challenge of incorporating domain ontologies in the data mining process has been addressed in the work on *semantic data mining* (SDM) (Lavrač and Vavpetič, 2015). Semantic data mining can discover complex rules describing subgroups of data instances that are connected to terms (annotations) of an ontology, where the ontology is referred to as background knowledge used in the learning process. An example SDM problem is to find subgroups of enriched genes in a biological experiment, where background knowledge is the Gene Ontology (Ashburner et al., 2000).

Given that SDM algorithms are relatively slow, the size of the background knowledge used by SDM approaches is usually several orders of magnitude lower than the problems typically handled by network analysis approaches. Take for example SDM algorithm Hedwig (Vavpetič et al., 2013), which is one of the semantic data mining algorithm used in this work. Hedwig performs beam search to explore the space of all possible explanations to find the best rules explaining the data. The search space it explores is very large and even using an adequate heuristic to guide the search, the algorithm takes a long time to find relevant patterns in real-life data annotated by ontological background knowledge. As illustration,

©2019 Jan Kralj, Marko Robnik-Šikonja, and Nada Lavrač.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-066.html.

take a SDM application where Hedwig was applied to a data set of 337 examples and a background knowledge of 21,062 interconnected ontology terms (Vavpetič et al., 2013), which is small compared to typical network analysis applications that deal with much larger data sets, possibly composed of e.g., hundred millions network nodes.

Despite large differences in the current sizes of data sets analyzed by SDM and network analysis approaches, the two research fields are aligned in terms of the research question of interest, which can be posed as follows: *Which part of the network structure is the most important for the analyst's current query?* The challenge addressed in this work is the reduction of the search space of SDM algorithms, which can be achieved by using network analysis approaches. To this end, we have developed an approach that is capable of utilizing network analysis algorithms Personalized PageRank (Page et al., 1999) and node2vec (Grover and Leskovec, 2016) to improve the efficiency of SDM. We show that the proposed approach, named NetSDM, can efficiently generate high quality rules by investigating only a fraction of the entire search space imposed by the background knowledge considered.

The rest of the paper is structured as follows. Section 2 presents the related work, as well as the technologies used; we first introduce the semantic data mining algorithms used, Hedwig and Aleph (Srinivasan, 1999), and then present the two network analysis algorithms, Personalized PageRank and node2vec. Section 3 presents the proposed NetSDM approach that exploits network analysis to reduce the size of the background knowledge used by SDM algorithms. Section 4 presents the experimental setup, and Section 5 presents the experimental results. Section 6 concludes the paper and presents plans for further work.

2. Related work and background technologies

This section presents the related work in the fields of semantic data mining and network analysis. It starts with the related work in the field of semantic data mining in Section 2.1, which presents also the two algorithms used in our experiments: Hedwig and Aleph. It continues with the related work in the field of network analysis in Section 2.2, and includes the two algorithms used in our experiments: Personalized PageRank and node2vec.

2.1. Semantic data mining

To find patterns in data annotated with ontologies, we rely on semantic data mining (SDM) (Dou et al., 2015; Lavrač and Vavpetič, 2015). In SDM, the input is composed of a set of class labeled instances and the background knowledge encoded in the form of ontologies, and the goal is to find descriptions of target class instances as a set of rules of the form `TargetClass` ← `Explanation`, where the explanation is a logical conjunction of terms from the ontology. Semantic data mining has its roots in symbolic rule learning, subgroup discovery and enrichment analysis research, briefly explained below.

2.1.1. Rule learning and subgroup discovery

One of the established techniques for data mining (Piatetsky-Shapiro, 1991) is symbolic rule learning (Fürnkranz et al., 2012). While rule learning was initially focused on learning predictive models in the form of classification rules, there is also a substantial research in descriptive rule learning, including association rule learning (Agrawal and Srikant, 1994), that aims at finding interesting descriptive patterns in the unsupervised as well as in supervised learning settings (Liu et al., 1998).

Building on classification and association rule learning, subgroup discovery techniques aim at finding interesting patterns as sets of rules that best describe the target class (Klösgen, 1996; Wrobel, 1997). Typical output of subgroup discovery algorithms are rules where the rule condition (`Explanation`) is a conjunction of features (attribute values) that characterize the target class instances covered by the rule, and each rule describes a particular interesting subgroup of target class instances.

2.1.2. Using ontologies in enrichment analysis

Enrichment analysis (EA) techniques are statistical methods used to identify putative explanations for a set of entities based on over- or under-representation of their attribute values, which can be referred to as *differential expression*. In life sciences, EA is widely used with the Gene Ontology (GO) (Ashburner et al., 2000) to profile the biological role of genes, such as differentially expressed cancer genes in microarray experiments (Tipney and Hunter, 2010).

While standard EA provides explanations in terms of concepts from a single ontology, researchers are increasingly combining several ontologies and data sets to uncover novel associations. The ability to detect patterns in data sets that do not use only the GO can yield valuable insights into diseases and their treatment. For instance, it was shown that Werner's syndrome, Cockayne syndrome, Burkitt's lymphoma, and Rothmund-Thomson syndrome are all associated with aging related genes (Puzianowska-Kuznicka and Kuznicki, 2005; Cox and Faragher, 2007). On the clinical side, EA can be used to learn adverse events from Electronic Health Record (EHR) data such as increased comorbidities in rheumatoid arthritis patients (LePendu et al., 2013), or to identify phenotypic signatures of neuropsychiatric disorders (Lyalina et al., 2013). An ontology-based EA approach was used to identify genes linked to aging in worms (Callahan et al., 2015), aberrant pathways—the network drivers in HIV infection identifying HIV inhibitors strongly associated with mood disorders (Hoehndorf et al., 2012), or to learn a combination of molecular functions and chromosome positions from lymphoma gene expression (Jiline et al., 2011).

2.1.3. Using ontologies in rule learning

An abundance of taxonomies and ontologies that are readily available can provide higher-level descriptors and explanations of discovered subgroups. In the domain of systems biology the Gene Ontology (Ashburner et al., 2000), KEGG orthology (Ogata et al., 1999) and Entrez gene–gene interaction data (Maglott et al., 2005) are examples of structured domain knowledge. In rule learning, the terms in the nodes of these ontologies can take the role of additional high-level descriptors (generalizations of data instances) used in the induced rules, while the hierarchical relations among the terms can be used to guide the search for conjuncts of the induced rules.

The SEGS algorithm (Trajkovski et al., 2008a) was the first to combine enrichment analysis and machine learning research in the construction of rules explaining gene expression data. SEGS constructs rules where the explanation is a logical conjunction of terms from several ontologies, explaining sets of differentially expressed (target class) genes as

combinations of Gene Ontology (GO) terms, KEGG orthology terms, and terms describing gene–gene interactions obtained from the Entrez database.

The challenge of incorporating domain ontologies in data mining was addressed in SDM research by several other authors. Žáková et al. (2006) used an engineering ontology of Computer-Aided Design (CAD) elements and structures as a background knowledge to extract frequent product design patterns in CAD repositories and to discover predictive rules from CAD data. Using ontologies, the algorithm Fr–ONT for mining frequent concepts was introduced by Lawrynowicz and Potoniec (2011).

Vavpetič and Lavrač (2013) developed a SDM toolkit that includes two semantic data mining systems: SDM-SEGS and SDM-Aleph. SDM-SEGS is an extension of the earlier domain-specific algorithm SEGS (Trajkovski et al., 2008a), which supports semantic subgroup discovery in gene expression data. SDM-SEGS extends and generalizes this approach by allowing the user to input a set of ontologies in the OWL ontology specification language and an empirical data set annotated with domain ontology terms. SDM-SEGS employs a predefined selection of ontologies to constrain and guide a top-down search of the hierarchically structured hypothesis space. SDM-Aleph is an extension of the Inductive Logic Programming system Aleph (Srinivasan, 1999), which does not have the limitations of SDM-SEGS imposed by domain-specific properties of algorithm SEGS, and can accept any number of ontologies as domain background knowledge, including different relations.

In summary, semantic subgroup discovery algorithms SDM-SEGS and SDM-Aleph are either specialized for a specific domain (Trajkovski et al., 2008b) or adapted from systems that do not take into account the hierarchical structure of background knowledge (Vavpetič and Lavrač, 2013), respectively. Therefore in this work, we rather use the Hedwig algorithm (Vavpetič et al., 2013) and the original Aleph algorithm (Srinivasan, 1999), which can both accept any number of input ontologies, where Aleph can also use different types of relations.

2.1.4. Hedwig

To illustrate the rules induced by the Hedwig subgroup discovery algorithm (Vavpetič et al., 2013), take as an example a task of analyzing differential expression of genes in breast cancer patients, originally addressed by Sotiriou et al. (2006). In this task, the `TargetClass` of the generated rules is the class of genes that are differentially expressed in breast cancer patients as compared to the general population.

Rule rank	Explanation
1	chromosome ∧ cell cycle
2	cellular macromolecule metabolic process ∧ intracellular non-membrane-bounded organelle ∧ cell cycle
3	cell division ∧ nucleus ∧ cell cycle
4	regulation of mitotic cell cycle ∧ cytoskeletal part
5	regulation of mitotic cell cycle ∧ microtubule cytoskeleton
6	regulation of G2/M transition of mitotic cell cycle
7	regulation of cell cycle process ∧ chromosomal part
8	regulation of cell cycle process ∧ spindle
9	enzyme binding ∧ regulation of cell cycle process ∧ intracellular non-membrane-bounded organelle
10	ATP binding ∧ mitotic cell cycle ∧ nucleus

Figure 1: An example output of the Hedwig semantic subgroup discovery algorithm, where `Explanations` represent subgroups of genes, differentially expressed in patients with breast cancer.

The Hedwig algorithm generated ten rules (subgroup descriptions), each describing a subgroup of differentially expressed genes. In the rules, the `TargetClass` is the set of differentially expressed genes and the `Explanation` is a conjunction of Gene Ontology terms and covers the set of genes annotated by the terms in the conjunction. The resulting set of `Explanations` (rule conditions) is shown in Figure 1.

Take the first ranked rule, where the `Explanation` is a conjunction of biological concepts `chromosome` ∧ `cell cycle`, which covers all the genes that are covered by both ontology terms. Take as another example the fourth best ranked rule, which explains that the regulation of the mitotic cell cycle and cytoskeletal formation explain the breast cancer based on gene expression.

The semantic subgroup discovery task addressed by Hedwig takes three types of inputs: the training examples, the domain knowledge, and a mapping between the two.

- Data set, composed of training examples expressed as RDF (Resource Description Framework) triples in the form subject-predicate-object, e.g., geneX–suppresses–geneY. Data set S is split into a set of positive examples S_+, i.e. 'interesting' target class instances (for example, genes enriched in a particular biological experiment), and a set of negative examples S_- of non-target class instances (e.g., non-enriched genes).
- Domain knowledge, composed of domain ontologies in RDF form.
- Object-to-ontology mapping, which associates each RDF triple with an appropriate ontological concept. We refer to these object-to-ontology mappings as *annotations*, meaning that object x is *annotated by* ontology term o if the pair (x, o) appears in the mapping.

For given inputs, the output of Hedwig is a set of descriptive patterns in the form of rules, where rule conditions are conjunctions of domain ontology terms that explain a group of target class instances. Ideally, Hedwig discovers explanations that best describe and cover as many target class instances and as few non-target instances as possible.

The Hedwig algorithm uses beam search, where the beam contains the best N rules. It starts with the default rule that covers all the training examples. In every search iteration, each rule from the beam is specialized via one of the four operations: (i) replace the predicate of a rule with a predicate that is a sub-class of the previous one, (ii) negate a predicate of a rule, (iii) append a new unary predicate to the rule, or (iv) append a new binary predicate, introducing a new existentially quantified variable, where the new variable has to be 'consumed' by a literal that has to be added as a conjunction to this clause in the next step of rule refinement.

Hedwig learns rules through a sequence of specialization steps. Each step either maintains or reduces the current number of covered examples. A rule will not be specialized once its coverage is zero or falls below some predetermined threshold. When adding a new conjunct, the algorithm checks if the specialized rule improves the probability of rule consequent—to this end the redundancy coefficient is used (Hämäläinen, 2010); if not, the specialized rule is not added to the list of specializations. After the specialization step is applied to each rule in the beam, a new set of best scoring N rules is selected. If no improvement is made to the rule collection, the search terminates. In principle, the procedure supports any rule scoring function. Numerous rule scoring functions for discrete targets are available: χ^2, Precision, WRAcc (Lavrač et al., 2004), Leverage, and Lift. Note that Lift is the default measure in Hedwig (see Section 2.1.6) that was used in our experiments.

In addition to an illustrative financial use case (Vavpetič et al., 2013), Hedwig was shown to perform well in breast cancer data analysis (Vavpetič et al., 2014), as well as in a biological setting when analyzing DNA aberration data for various cancer types (Adhikari

et al., 2016), where it was a part of three-step methodology, together with mixture models and banded matrices using several ontologies obtained from various sources: hierarchical structure of multiresolution data, chromosomal location of fragile sites, virus integration sites, cancer genes, and amplification hotspots.

2.1.5. Aleph

Aleph (A Learning Engine for Proposing Hypotheses) (Srinivasan, 1999) is a general purpose system for Inductive Logic Programming (ILP) that was conceived as a workbench for implementing and testing concepts and procedures from a variety of different ILP and relational learning systems and papers. The system can construct rules, trees, constraints and features; invent abnormality predicates; perform classification, regression, clustering, and association rule learning; allows for choosing between different search strategies (general-to-specific or specific-to-general, i.e. top-down or bottom-up, bidirectional, etc.), search algorithms (hill-climbing, exhaustive search, stochastic search, etc.) and evaluation functions. It allows users to specify their own search procedure, proof strategy, and visualization of hypotheses.

Similarly to Hedwig, Aleph accepts as input a set of positively and negatively labeled training examples and a file describing the background knowledge, including a mapping between the examples and the background knowledge. The training examples are expressed as Prolog facts, and the background knowledge is in the form of Prolog facts and clauses. In contrast to Hedwig that was used in our experiments for semantic subgroup discovery, we used Aleph in its default operation mode to learn a classifier (a Theory) composed of as a set of classification rules, which also explain the target class examples.

Aleph constructs rules in several steps. It first selects a positive example and builds the most specific rule explaining the example following the steps described by Muggleton (1995). Next, the algorithm searches for a more general rule, performed with a branch-and-bound search strategy, searching first through generalizations of the rule containing the fewest terms. Finally, the rule with the best score is added to the current theory, and all the examples covered by this rule are removed (this step is sometimes called the "cover removal" step). This procedure is repeated until all the positive examples are covered.

Aleph was incorporated into the SDM-toolkit (Vavpetič and Lavrač, 2013), which allows Aleph (named SDM-Aleph) to use the same input data as Hedwig. However, a drawback of using SDM-Aleph is that it can reason only over one type of background knowledge relations, while the original Aleph algorithm can reason over any type of relations. In the case of Gene Ontology, which was used as the background knowledge in our experiments, this generality allows Aleph to encode both the *is_a* and *is_part_of* relations, which are the most frequent relations in ontologies, as well as the relationships composed of these two relations.

2.1.6. Rule quality measures

We use several measures to evaluate the quality of rules discovered by Hedwig and Aleph. Take for example rule R with a condition that is true for 80 positive instances (*True Positives*, TP) and 20 negative instances (*False Positives*, FP) from the set S of 100 positive instances (*Positives*) and 100 negative instances (*Negatives*). Coverage$(R) = TP + FP =$ 100 is the total number of instances covered by the rule. Support of the rule is calculated as follows: Support$(R) = \frac{TP+FP}{|S|} = \frac{\text{Coverage(R)}}{|S|} = 0.5$. Precision$(R) = 0.8$ is calculated as $\frac{TP}{TP+FP}$, and Accuracy$(R) = 0.8$ is calculated as $\frac{TP+TN}{|S|}$, where TN denotes the number of *True Negatives*.

In our experiments we use the Lift metric to evaluate the performance of SDM algorithms Hedwig and Aleph. Lift(R) is defined as the ratio between the precision of a rule and the proportion of positive examples in the data set (i.e. the precision of the empty default rule that classifies all examples as positive). In our example Lift$(R) = \frac{0.8}{0.5} = 1.6$, which is calculated using the following formula:

$$\text{Lift(R)} = \frac{\text{Precision}(R)}{PR} \tag{1}$$

where $PR = \frac{|Positives|}{|S|}$ is the ratio of positive instances in the data set. The range of values Lift can take is from 0 to ∞ and larger values indicate better rules.

2.2. Network analysis

Network analysis uses concepts from graph theory to investigate characteristics of networked structures in terms of nodes (such as individual actors, people, or any other objects in the network) and edges or links that connect them (relationships, interactions or ties between the objects). This section focuses on the algorithms which we used or adapted to rank nodes in information networks.

2.2.1. Node ranking: Using Personalized PageRank algorithm

The objective of node ranking in information networks is to assess the relevance of a given node either globally (with regard to the whole graph) or locally (relative to some other node or group of nodes in the graph). A network node ranking algorithm assigns a *score* (or a *rank*) to each node in the network, with the goal of ranking the nodes in terms of their relevance.

A well known node ranking method is PageRank (Page et al., 1999), which was used in the Google search engine. Other methods for node ranking include Weighted PageRank method (Xing and Ghorbani, 2004), SimRank (Jeh and Widom, 2002), diffusion kernels (Kondor and Lafferty, 2002), hubs and authorities (Kleinberg, 1999), and spreading activation (Crestani, 1997). More recent network node ranking methods include PL-ranking (Zhang et al., 2016) and NCDawareRank (Nikolakopoulos and Garofalakis, 2013). Another way to rank nodes in a network is to use network centrality measures, such as Freeman's network centrality (Freeman, 1979), betweenness centrality (Freeman, 1977), closeness centrality (Bavelas, 1950), and Katz centrality (Katz, 1953).

In this paper, we are interested in network node ranking methods that can be 'localized', i.e. which do not compute the global importance/score of a node, but rather the score of a node in the context of a given subset of other nodes. An example of this type of network node ranking approaches is the Personalized PageRank (P-PR) algorithm (Page et al., 1999), sometimes referred to as random walk with restart (Tong et al., 2006). P-PR calculates the node score locally to a given network node—it is hence especially interesting for our work because it calculates the importance of network nodes relative to a given set of

starting nodes, in our case, those representing the positive examples. In the context of data set annotation with ontologies, this allows us to estimate the importance of background knowledge terms in relation to the positive examples in the data set.

Personalized PageRank uses a random walk approach to calculate the significance of nodes in an information network. Given a set of 'starting' nodes A, the Personalized PageRank calculated for A (denoted P-PR$_A$) is defined as the stationary distribution of random walker positions, where the walk starts in a randomly chosen member of A and then at each step either selects one of the outgoing connections or teleports back to a randomly selected member of A. Probability p of continuing the walk is a parameter of the Personalized PageRank algorithm and is usually set to 0.85. In our context, the P-PR algorithm is used to calculate the importance of network nodes *with respect to* a given set of starting nodes.

Remark 1 *P-PR$_A$ is a vector, and each value of the vector corresponds to some network node u. If u is the i-th node of the network, notation P-PR$_A(u)$ is used (rather than P-PR$_{A_i}$) to denote the i-th value of P-PR$_A$.*

2.2.2. Network embedding: Using node2vec algorithm

Network embedding is a mechanism for converting networks into a tabular/matrix representation format, where each network node is represented as a vector of some predefined fixed length k, and a set of nodes is represented as a table with k columns. The goal of such a conversion is to preserve structural properties of a network, which is achieved by preserving node similarity in both representations, i.e. node similarity is converted into vector similarity.

Network nodes vectorization can be effectively performed by the node2vec algorithm (Grover and Leskovec, 2016), which uses a random walk approach to calculate features that express the similarities between node pairs. The node2vec algorithm takes as input a network of n nodes, represented as a graph $G = (V, E)$, where V is the set of nodes and E is the set of edges in the network. For a user-defined number of columns k, the algorithm returns a matrix $f^* \in \mathbb{R}^{|V|\times k}$, defined as follows:

$$f^* = \text{node2vec}(G) = \underset{f\in\mathbb{R}^{|V|\times k}}{\text{argmax}} \sum_{u\in V} \left(-\log(Z_u) + \sum_{n\in N(u)} f(n)\cdot f(u) \right) \tag{2}$$

where $N(u)$ denotes the network neighborhood of node u (to be defined below).

Remark 2 *Each matrix f is a collection of k-dimensional feature vectors, with the i-th row of the matrix corresponding to the feature vector of the i-th node in the network. We write $f(u)$ to denote the row of matrix f corresponding to node u.*

The goal of the node2vec algorithm is to construct feature vectors $f(u)$ in such a way that feature vectors of nodes that share a certain neighborhood will be similar. The inner sum of the value, maximized in Equation 2, calculates the similarities between node u and all nodes in its neighborhood: it is large if the feature vectors of nodes in the same neighborhood are collinear, however it also increases if feature vectors of nodes have a large

norm. Next, value Z_u calculates the similarities between node u and all the nodes in the network as follows:

$$Z_u = \sum_{v\in V} e^{f(u)\cdot f(v)}$$

Note that value of $-\log(Z_u)$ decreases when the norms of feature vectors $f(v)$ increase, thereby penalizing collections of feature vectors with large norms.

Equation 2 has a probabilistic interpretation that models a process of randomly selecting nodes from the network (Grover and Leskovec, 2016). In this process, probability $P(n|u)$ of node n following node u in the selection process is proportional to $e^{f(n)\cdot f(u)}$. Assuming that selecting a node is independent from selecting any other node, we can calculate the probability of selecting all nodes from a given set N as $P(N|u) = \prod_{n\in N} P(n|u)$, and Equation 2 can then be rewritten as follows:

$$f^* = \text{node2vec}(G) = \underset{f\in\mathbb{R}^{|V|\times k}}{\text{argmax}} \sum_{u\in V} \log\left(P(N(u)|f(u))\right) \tag{3}$$

Finally, let us explain neighborhood $N(u)$ in Equations 2 and 3, calculated by simulating a random walker traversing the network starting at node u. Unlike the PageRank random walker, the transition probabilities for traversing from node n_1 to node n_2 depend on node n_0 that the walker visited before node n_1, making the process of traversing the network a second order random walk. The non-normalized transition probabilities are set using two parameters, p and q, and are equal to:

$$P(n_2|\text{moved from node } n_0 \text{ to } n_1 \text{ in previous step}) = \begin{cases} \frac{1}{p} & \text{if } n_2 = n_0 \\ 1 & \text{if } n_2 \text{ can be reached from } n_1 \\ \frac{1}{q} & \text{otherwise} \end{cases}$$

The parameters of the expression are referred to as the *return* parameter p and the *in-out* parameter q. A low value of the return parameter p means that the random walker is more likely to backtrack its steps and the random walk will be closer to a breadth first search. On the other hand, a low value of parameter q encourages the walker to move away from the starting node and the random walk resembles a depth first search of the network. To calculate the maximizing matrix f^*, a set of random walks of limited size is simulated starting from each node in the network to generate several samples of sets $N(u)$.

The function maximizing Equation 2 is calculated using the stochastic gradient descent. The matrix of feature vectors node2vec(G) in Equation 2 is estimated at each generated sampling of neighborhoods $N(u)$ for all nodes in the network to discover matrix f^* that maximizes the expression for the simulated neighborhood set.

3. NetSDM methodology: Combining SDM with network analysis

SDM algorithms perform well on relatively small real world data sets, but even for small data sets the algorithms search through a very large space of possible patterns to find the 'best' pattern. The more conjuncts we allow in rule conditions and the larger the background knowledge, the larger the search space. This section presents the proposed

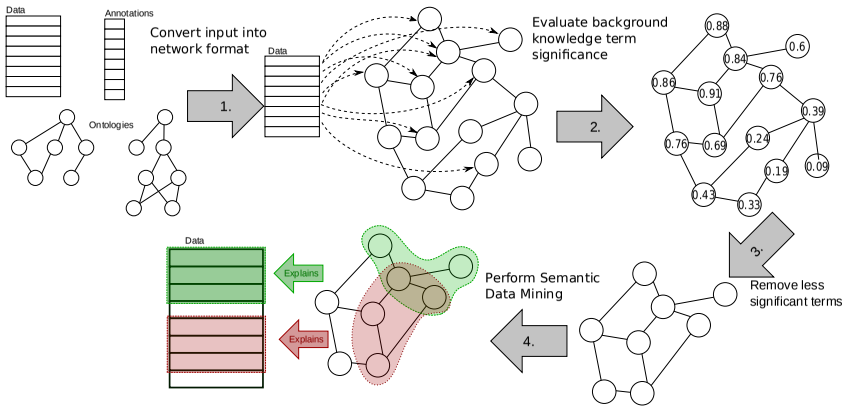


Figure 2: Illustrative outline of the proposed NetSDM methodology.

NetSDM methodology that combines semantic data mining with network analysis to reduce the search space of SDM algorithms. This section starts with an overview of the proposed methodology in Section 3.1, illustrated by an example in Section 3.2. Section 3.3 provides a detailed step-by-step description of the NetSDM methodology.

3.1. Methodology outline

SDM algorithms use heuristic search when mining for patterns in the data to limit the search to only the most promising parts of the search space. Constraining the search is necessary, as traversing the entire search space—consisting of conjuncts of logical expressions containing ontological expressions—is computationally unfeasible. However, heuristic search can produce poor results in some cases. For example, when searching for patterns with Hedwig using beam search, the beam may be too narrow, and consequently a branch that would lead to a high quality solution could be discarded early on; in large search spaces even wide beams can be quickly filled with terms that do not lead to good final solutions. Similarly, the search algorithm in Aleph can miss important patterns if all their sub-patterns exhibit low quality. In this paper we propose a new methodology, which aims to solve this problem. The proposed NetSDM methodology, which is illustrated in Figure 2, consists of the following four steps:

1. Convert a background knowledge ontology into a network format.
2. Estimate the importance of background knowledge terms using a scoring function.
3. Shrink the background knowledge, keeping only a proportion c of the top ranking terms.
4. Apply a semantic data mining algorithm on the original data set and the reduced background knowledge.

Input: SDM algorithm, network conversion method, scoring function, term removal method, background knowledge ontology O, set of examples S annotated with terms from O
Output: set of rules discovered by SDM algorithm
Parameters: shrinkage coefficient $c \in [0, 1]$

```
Convert ontology O into network format G_e (see Section 3.3.1)
Calculate significance scores score(t), for all terms t ∈ T(O) (see Section 3.3.2)
for term t ∈ T(O) do
    if RelativeRank(t) > c then
        mark term t for removal.
    end
end
form G_e' by removing marked terms from G_e (see Section 3.3.3)
convert G_e' to O'
Run SDM algorithm on S using reduced background knowledge O'
return Rules discovered by SDM algorithm
```

Algorithm 1: The NetSDM algorithm, implementing the proposed approach to semantic data mining with network node ranking and ontology shrinking.

The NetSDM methodology improves the efficiency of SDM algorithms by using a shrinking step in which we filter out background knowledge terms that are not likely to appear in significant rules. The methodology is implemented in Algorithm 1. A scoring function used in ontology shrinkage should (i) be able to evaluate the significance of terms based on the data, and (ii) be efficiently computed.
The input for the scoring function is the data used in SDM algorithms, consisting of:

- Set of instances S consisting of target (S_+) and non-target (S_-) instances,
- Ontology O represented as a set of RDF triples (*subject, predicate, object*) (to simplify the notation, for each term t appearing as either a subject or object in a triple, we will write $t \in O$ to denote that t is a term of the ontology O),
- Set of annotations $\mathcal{A}$ connecting instances in $S = S_+ \cup S_-$ with terms in O (annotations are presented as triples $(s, \textit{annotated-with}, t)$, where $s \in S$ and $t \in O$ denoting that data instance s is annotated by an ontology term t.

The output of the scoring function is a vector which, for each term in the background knowledge, contains a score estimating its significance. In other words, using $\mathcal{T}(O)$ to denote the set of all terms appearing (either as subjects or objects) in ontology O, the scoring function is defined as:

$$\text{score}_{S,O,\mathcal{A}} : \mathcal{T}(O) \to [0, \infty)$$

After defining a suitable scoring function *score* (in the subsequent subsections, we examine two possibilities), we use the computed scores to shrink the background knowledge ontology

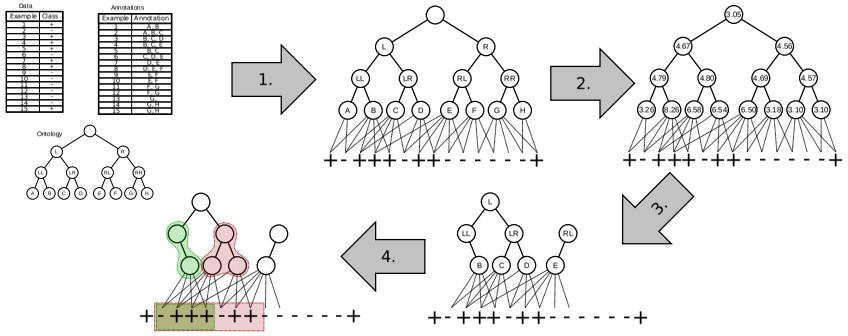


Figure 3: Illustration of using Algorithm 1 on example from Table 1.

O. Using a scoring function, a higher score of term $t \in O$ means that term t is more likely to be used in rules describing the positive examples. For every term $t \in \mathcal{T}(O)$, we use the results of the scoring function to assign *relative ranks* to ontology terms defined as

$$\text{RelativeRank}(t) = \frac{\text{Rank}(t)}{|\mathcal{T}(O)|} = \frac{|\{t' \in \mathcal{T}(O) : \text{score}(t') \geq \text{score}(t)\}|}{|\mathcal{T}(O)|}$$

which calculates the ratio of terms in the ontology that score higher than t, divided by the number of all terms. For example, a value $\text{RelativeRank}(t) = 0.05$ means that only 5% of all terms in the ontology score higher than t using a given scoring function.

3.2. Illustrative example

As an illustrative example consider the data set described in Table 1. The data set consists of 15 instances, 7 of which belong to the target class (positive examples, marked with + in Table 1). The instances are annotated by one or more terms in a hierarchy (i.e., simple ontology) consisting of 8 base nodes and 7 higher-level terms, shown on the left-hand side of Figure 3. Using the Personalized PageRank scoring function (described in Section 3.3.2), high scores are assigned to the background knowledge terms that are strongly related to the positive examples. The resulting scores are shown in the middle of Figure 3. We can see that the left-hand side terms have higher scores compared to the right-hand side terms as they annotate mostly the positive (target) examples.

Table 1: An illustrative data set consisting of 15 examples annotated by 8 possible annotations.

Example	Class	Annotated by
1	+	A,B
2	-	A,B,C
3	+	B,C,D
4	+	B,C,E
5	+	B,C
6	-	C,D,E
7	+	D,E
8	+	D,E,F
9	-	E,F
10	-	E,F
11	-	F,G
12	-	F,G
13	-	G
14	+	G,H
15	+	G,H

In the next step, the algorithm prunes the lower-ranked terms, leaving only the best 8 background terms in the reduced hierarchy of ontology terms. As most of the top scoring terms were in the left part of the hierarchy, the reduced hierarchy mostly contains these terms. In the final step, we use a SDM algorithm—in this case we use Hedwig—to find rules based on the instances from Table 1 and the reduced ontology. The best two rules discovered by Hedwig, which cover most of the positive examples, are:

`Positive` ← `LL` (shown in green in Figure 3), covering 3 positive and 1 negative instance (Coverage = 4, Precision = $\frac{3}{4}$).

`Positive` ← `LR` (shown in red in Figure 3), covering 5 positive and 2 negative instances (Coverage = 7, Precision = $\frac{5}{7}$).

If we run Hedwig using the original ontology for this data set, we get the same rules as on the reduced background knowledge. However, as demonstrated by the experiments on large real-life data sets in Section 5, removing unimportant parts of large ontologies benefits both the execution time and the quality of discovered rules.

3.3. Detailed description of NetSDM

In this section, we describe four steps of the NetSDM methodology outlined in Algorithm 1. We first describe the conversion of a background knowledge ontology into an information network, which we can then analyze using network analysis methods. Next, we describe two methods for term importance estimation in information networks. The section concludes with a discussion on the term deletion step of the algorithm in which low-scoring terms are removed from the network.

3.3.1. Conversion of background knowledge ontology into network format

The first step of the NetSDM methodology is conversion of the background knowledge ontologies into a network. We present two methods, a *direct* method and a *hypergraph* method of conversion.

Direct conversion. Input ontologies for the NetSDM methodology are collections of triples that represent relations between background knowledge terms. As any information network is also composed of a set of nodes and the connections between them, a natural conversion from the background knowledge into an information network is to convert each relation between two ontology terms into an edge between two corresponding nodes in an information network. This gives rise to the first function converting the input ontology into the information network, in which we view the ontology and the data instances as parts of one network. In the conversion, we merge the data set and the background knowledge into one network $G_e = (V_e, E_e)$ where

- $V_e = \mathcal{T}(O) \cup S_+$ are vertices of the new network consisting of all background knowledge terms $\mathcal{T}(O)$ and all target class instances (positive examples) S_+;
- $E_e = \{(t_1, t_2) | \exists r : (t_1, r, t_2) \in O\} \cup \{(s, t) | (s, \textit{annotated-with}, t) \in \mathcal{A}\}$ are edges of the new network consisting of edges that represent background knowledge relations, as well as all edges that represent the *annotated-with* relations between

data instances and background knowledge terms (the meaning of O, S, S_+, S_- and $\mathcal{A}$ is explained in Section 3.1).

Hypergraph conversion. Liu et al. (2013) proposed an alternative approach to converting an ontology into an information network format. They treat every relation in a semantic representation of data as a triple, consisting of a subject, predicate and object. They construct a network (which they call a *hypergraph*) in which every triple of the original background knowledge (along with the background knowledge terms) forms an additional node in the network with three connections: one connection with the subject of the relation, one with the object of the relation, and one with the predicate of the relation. In this conversion, each predicate that appears in the background knowledge is an additional node in the resulting network. This conversion results in a network $G_e = (V_e, E_e)$ where:

- $V_e = \mathcal{T}(O) \cup S_+ \cup \{n_r | r \in O \cup \mathcal{A}\} \cup \{p | \exists s, o : (s, p, o) \in O\} \cup \{\textit{annotated-with}\}$ are vertices of the new network consisting of (i) all background knowledge terms, (ii) all target class instances, (iii) one new node n_r for each relation r either between background knowledge terms or linking background knowledge terms to data instances and (iv) one node for each predicate appearing in the ontology O, as well as (v) one node representing the *annotated-with* predicate;
- $E_e = \bigcup_{r=(s,p,o) \in O \cup \mathcal{A}} \{(s, n_r), (p, n_r), (o, n_r)\}$ are edges of the new network, each relation r inducing three connections to node n_r that it induces (terms V, S, S_+, S_- and $\mathcal{A}$ are defined in Section 3.1).

While hypergraph conversion results in a slightly larger information network, less information is lost in the conversion process than in the direct conversion process presented above.

An important aspect of both conversion methods, presented above, is that they are defined simply on a set of triples that represent relations between background knowledge terms. This means that both methods can be used to transform an arbitrarily complex ontology with any number of predicates.

3.3.2. Term significance score calculation

After converting the background knowledge and the input data into a joined network, we use either Personalized PageRank or node2vec based scoring function to evaluate the significance of background knowledge terms. Term importance scoring function $\text{score}_{S,O,\mathcal{A}}$ is constructed by taking into account the data set S and background knowledge terms $\mathcal{T}(O)$.

P-PR based score computation. The first scoring function is computed using the Personalized PageRank algorithm (Page et al., 1999). While the basic PageRank algorithm evaluates the global importance of each node in a network (i.e. with respect to all other nodes), the Personalized PageRank (described in Section 2.2.1) evaluates the significance of nodes with respect to a given node (or a set of nodes). This fits well with our demand that a scoring function must evaluate the significance of a term

based on the actual data - in our case the importance of the nodes is not global but calculated in the context of the given positive examples. We can therefore reasonably expect that the highest scoring terms will not simply be the most prominent terms of the background knowledge ontology but rather those terms which are the most relevant to the input data set that a researcher is interested in. As each term $t \in \mathcal{T}(O)$ is a vertex in network G_e, we calculate the P-PR score of each term $t \in \mathcal{T}(O)$ as follows:

$$\text{score}_{\text{P-PR}}(t) = \text{P-PR}_{S_+}(t) \tag{4}$$

where the P-PR vector is calculated on the network G_e. A simple algebraic calculation (Grčar et al., 2013) shows that this value is equal to the average of all Personalized PageRank scores of v where the starting set for the random walker is a node from S_+:

$$\text{P-PR}_{S_+}(t) = \frac{1}{|S_+|} \sum_{w \in S_+} \text{P-PR}_{\{w\}}(t) \tag{5}$$

Following the definition of the Personalized PageRank score, the value of $\text{score}_{\text{P-PR}}$ is the stationary distribution of a random walker that starts its walk in one of the target data instances (elements of the set S_+) and follows the connections (either the is-annotated-by relation or a relation in the background knowledge). Another interpretation of $\text{score}_{\text{P-PR}}(t)$ is that it tells us how often we will reach node t in random walks, starting with positive (target class labeled) data instances. Note that the Personalized PageRank algorithm is defined on directed networks, allowing us to calculate the Personalized PageRank vectors of nodes by taking the direction of connections into account. In our experiments, we also tested the performance of the scoring function if network G_e is viewed as an undirected network. To calculate the PageRank vector on an undirected network, each edge between two nodes u and v is interpreted as a pair of directed edges, one going from u to v and another from v to u, allowing the random walker to traverse the original undirected edge in both directions.

node2vec based score computation. The second estimator of background knowledge terms significance is the node2vec algorithm (Grover and Leskovec, 2016). In our experiments, we used the default settings of $p = q = 1$ for the parameters of node2vec, meaning that the generated random walks are balanced between the depth-first and breadth-first search of the network. The maximum length of random walks was set to 15. The function node2vec (described in Section 2.2.2) calculates the feature matrix $f^* = \text{node2vec}(G_e)$, and each row of f^* represents a feature vector $f^*(u)$ for node u in G_e. The resulting feature vectors of nodes can be used to compute the similarity between any two nodes in the network. The approach uses the cosine similarity of the two feature vectors u and v, computed as follows:

$$\text{similarity}_{\text{node2vec}}(u, v) = \frac{f^*(u) \cdot f^*(v)}{|f^*(u)||f^*(v)|}$$

In our approach, we form feature vectors for nodes representing both background knowledge terms and data instances. We use these feature vectors to compute the

similarity between the background knowledge terms and the positive data instances (target class examples) using a formula inspired by Equation 5. With the Personalized PageRank, we evaluate the score of each node as P-PR$_{S_+}(v)$ where S_+ is the set of all target class instances. As the value P-PR$_{\{w\}}(t)$ measures the similarity between w and t, Equation 5 can also be used to construct a node2vec scoring function. We replace the individual P-PR similarities in Equation 5 with the individual node2vec similarities:

$$\text{score}_{\text{node2vec}}(t) = \frac{1}{|S_+|} \sum_{w \in S_+} \frac{f^*(w) \cdot f^*(t)}{|f^*(w)||f^*(t)|} \tag{6}$$

3.3.3. Network node removal

In the third step of Algorithm 1, low-scored nodes are removed from the network. We present and test two options for network node removal.

Naïve node removal. The first (naïve) option is to take every node that is marked for removal and delete both the node and any connection leading to or from it. This method is robust and can be applied to any background knowledge. However, when background knowledge is heavily pruned, this method may cause the resulting network to get decomposed into several disjoint connected components. If we convert the background knowledge into a hypergraph, the hypergraph has to contain relation nodes with *exactly* three neighbors (the subject, predicate and object of the relation) if we are to convert it back into a standard representation of background knowledge. Thus, naïvely removing nodes from the hypergraph may result in a network that we can no longer convert back to the standard background knowledge representation.

Advanced node removal. We tested an alternative approach that takes into account that the relations, encoded by the network edges, are often transitive. For example, in our work we use the *is-a* and *is-part-of* relations that are both transitive. Using transitivity, we design an algorithm for advanced removal of low scoring nodes from information networks, obtained by direct conversion of the background knowledge (Algorithm 2). This advanced node removal step can be used for ontologies with any number of relations. Just like the is-a relation in Algorithm 2, other transitive relation are added back to the network in row three of the algorithm. As the background knowledge is in direct correspondence with the information network obtained from it, removing the node from the information network corresponds to removing a matching term from the background knowledge. In this sense, Algorithm 2 can be used as the term-removal step of Algorithm 1. Algorithm 2 can also be used to remove low scoring nodes from hypergraphs constructed from the background knowledge if we first convert hypergraphs into a simple network form resulting from the direct conversion of background knowledge ontologies.

3.3.4. Applying SDM algorithms

SDM algorithms Hedwig and Aleph (described in Sections 2.1.4 and 2.1.5, respectively) were used in the last step of the NetSDM methodology to discover rules explaining the target class instances.

Input: Information network G_e (obtained by direct conversion of background knowledge into information network format with directed edges) and node $n \in G_e$ (a term of the original background knowledge) we wish to remove
Output: New background knowledge that does not contain node n

```
for b ∈ G_e : (b, n) is an edge in G_e do
    for a ∈ G_e : (n, a) is an edge in G_e do
        Add edge (b, a) into G_e
        Remove edge (n, a) from G_e
    end
    Remove edge (b, n) from G_e
end
Remove node n from G_e
Return G_e
```

Algorithm 2: The algorithm for removing a node from a network, obtained through direct conversion of the background knowledge into the information network format.

Applying Hedwig. Hedwig—which was previously successfully used in a real-life application for explaining biological data (Vavpetič et al., 2014)—can use only one relation type in the background knowledge, i.e. the *is-a* relation in the case of the Gene Ontology. Hedwig uses this relation and its transitivity property (if A *is-a* B and B *is-a* C, then A *is-a* C) to construct rules.

Applying Aleph. In contrast to Hedwig, the input to Aleph can contain several relations as well as interactions between them. Therefore applying Aleph enabled us to use both the *is-a* and *is-part-of* relations in the Gene Ontology. Aleph is capable of using not only these two relations and their transitivity, but also the interaction between the two relations in the form of two facts: (i) if A *is-a* B and B *is-part-of* C, then A *is-part-of* C and (ii) if A *is-part-of* B and B *is-a* C, then A *is-part-of* C.

Since Aleph can use several relations for constructing rules, the background knowledge network used in Aleph experiments contained the edges representing *is-a* relations as well as the edges representing *is-part-of* relations.

4. Experimental settings

This section presents the settings of the experiments with two semantic data mining algorithms (Hedwig and Aleph) and two search space reduction mechanisms (based on Personalized PageRank and node2vec). We first describe the data sets, followed by the outline of the three experimental setups.

4.1. Data sets

In the main experiments, presented in this paper, we used two data sets: the acute lymphoblastic leukemia (ALL) and the breast cancer data set.

ALL (acute lymphoblastic leukemia) data. The ALL data set, introduced by Chiaretti et al. (2004), is a typical dataset for medical research. In data preprocessing we followed the steps used by Podpečan et al. (2011) to obtain a set of 1,000 enriched genes (forming the target set of instances) from a set of 10,000 genes, annotated by concepts from the Gene Ontology (Ashburner et al., 2000), which was used as the background knowledge in our experiments. In total, the data set contained 167,339 annotations (connections between genes and Gene Ontology terms). In previous work on analyzing the ALL data set, the performance of the SegMine methodology (Podpečan et al., 2011) was compared to the performance of the DAVID algorithm (Huang et al., 2008). In this work, we use the same data set to assess if network node ranking and reduction can decrease the run time and improve the performance of the tested SDM algorithms.

Breast cancer data. The breast cancer data set, introduced by Sotiriou et al. (2006), contains gene expression data on patients suffering from breast cancer. In previous work, Vavpetič et al. (2014) first clustered the genes into several subsets and then used a semantic subgroup discovery algorithm to interpret the most important subset (as determined by domain experts). In this work, we constructed rules for the same subset of genes as by Vavpetič et al. (2014) and did not use the input of domain experts. The set contains 990 genes out of a total of 12,019 genes. In our experiments, we used SDM algorithms to describe the 990 interesting genes. The data instances are connected to the Gene Ontology terms by 195,124 annotations.

To test whether the results of NetSDM on the Gene-Ontology-annotated data sets described above can be generalized to other data sets, we ran the NetSDM algorithm also on a set of smaller data sets, annotated by hierarchical background knowledge, which were used in conjunction with the Hedwig algorithm in our previous work (Adhikari et al., 2016). The results of these experiments are shown in Appendix A of this paper.

4.2. Experimental setups

We constructed three experimental setups to test the NetSDM methodology. In all setups, both Hedwig and Aleph are used in their standard modes (subgroup discovery and classification rule learning, respectively) to return the set of rules explaining the target class examples. Hedwig returns a rule set containing all the rules in the search beam at the end of search (meaning that the size of the returned rule set matches the size of the beam). Aleph returns a rule set consisting of rules with a sufficient coverage and low enough noise (i.e. rules covering at least 10 positive examples and at most 100 negative examples). In the first setup, the entire set of rules was analyzed. In the second setup, we used the NetSDM methodology with the direct method of ontology conversion and the naïve version of node removal. Given their different operation modes (Hedwig was used for learning individual patterns and Aleph was used for predictive theory construction), we always analyzed only the top rule discovered by Hedwig, and the top 3 rules of the theory discovered by Aleph. We also list the metrics for the entire theory discovered by Aleph. This allows clearer comparison and is sufficient to demonstrate how shrinking the background knowledge affects the quality of discovered rules. In the third setup, we use the advanced (transitivity based)

node removal approach (described in Section 3.3.3). We compare the quality of the resulting rules by measuring the Lift values of each rule (defined in Equation 1 in Section 2.1.4). Details of each experimental setup are given below.

First experimental setup. We ran the two SDM algorithms on the entire data set to determine the baseline performance and examine the returned rules. Using Hedwig, we ran the algorithm with all combinations of depth (1 or 10), beam width (1 or 10) and support (0.1 or 0.01). For Aleph, we ran the algorithm using the settings recommended by the algorithm author—minimum number of positive examples covered by a rule was set to 10, and maximum number of negative examples covered by a rule was set to 100. These settings resulted in rules comparable to those discovered by Hedwig.

In both cases, our goal was to determine whether the terms used by the SDM algorithms are correlated with the scores of the terms returned by the network analysis algorithms, i.e., calculated by P-PR (Equation 4) or node2vec (Equation 6). We evaluated this relation by observing the ranks of terms used by both algorithms to construct the rules. For each term t appearing in the constructed rules, we computed the relative rank of the term t (as defined in Section 3.1). If the correlation between SDM algorithm's use of terms and returned scores is strong, the relative rank calculated for a term t and used by the algorithm will be low. Low percentages ensure that we can retain only a small percentage of the highest ranked scores.

Second experimental setup. In the second set of experiments, we exploited the correlations discovered in the first experimental setup. We used the scores of background knowledge terms to prune the background knowledge. Specifically, we ran the NetSDM algorithm by setting the shrinkage coefficient values of c to values between 0.005 and 1, thereby running the SDM algorithms on the gene ontology (GO) background knowledge containing only a subset of as little as 0.5% of the terms with the highest scores. We calculated the P-PR values of GO terms in two ways: (i) we viewed *is-a* relations as directed edges pointing from a more specific GO term to a more general term, and (ii) we viewed the relations as undirected edges. When running the experiments with Hedwig, we set the beam size, depth and minimum coverage parameters to the values that returned the best rules on each data set in the first set of experiments. For example, the results of the first experimental setup showed that the rules obtained by setting the beam size or rule depth to one are too general to be of any biological interest; we therefore decided to set both values to 10. In the case of rule depth, this value allows Hedwig to construct arbitrarily long rules (given that in our data sets Hedwig returned no rules of length greater than five). Setting the beam size to 10 allows Hedwig to discover important rules (as shown in the first set of experiments) in a reasonable amount of time; note that the run time of Hedwig increases drastically with increased beam size, and at size 10 the algorithm takes several hours to complete. Running the experiments with the Aleph algorithm, we again used the setting recommended by the algorithm author.

Third experimental setup. In the first two sets of experiments, we used the direct method for converting background knowledge into an information network and the naïve method for deleting low scoring nodes. In the third set of experiments, we

Table 2: The best rules discovered by the Hedwig algorithm on the ALL data set. The conjuncts (Gene Ontology terms) of each top ranking rule are separated by a horizontal line in the table with the 4 right-most columns denoting the settings used to discover the rule, and the Lift value of the entire rule. The second (third) column can be understood as the percentage of GO terms with the Personalized PageRank (node2vec) score higher than the corresponding term.

Term	RelativeRank (P-PR)[%]	RelativeRank (Node2vec)[%]	Beam	Depth	Support	Lift
GO:0003674	0.297	46.383	1	1	0.01	1.000
GO:0003674	0.297	46.383	1	10	0.01	1.000
GO:0003674	0.297	46.383	1	1	0.1	1.000
GO:0003674	0.297	46.383	1	10	0.1	1.000
GO:0050851	4.467	2.066	10	1	0.01	2.687
GO:0002376	0.603	62.137	10	10	0.01	3.420
GO:0002429	2.506	13.985				
GO:0005886	0.076	0.876				
GO:0005488	0.050	16.979				
GO:0002376	0.603	62.137	10	1	0.1	1.292
GO:0002376	0.603	62.137	10	10	0.1	1.414
GO:0005488	0.050	16.979				
GO:0048518	1.056	93.724				
GO:0003674	0.0046	46.38	1	10	0.01	1

tested the advanced versions of both steps: hypergraph conversion and advanced node deletion, respectively. As the node2vec function proved inferior to the Personalized PageRank scoring function in the second set of experiments, we ran the third round of experiments only using the P-PR scoring function. The results of this third set of experiments can then be compared to the results of the second set.

5. Experimental results

We present the results of algorithms Hedwig and Aleph using two scoring functions, two network conversion methods and two node deletion methods on the two data sets. We analyze the results obtained from the three experimental setups.

5.1. First setup: Scores of terms used in SDM algorithms

We begin by presenting the results of the first experimental setup where we analyze the scores and the ranks of terms appearing in rules constructed by Hedwig and Aleph, respectively.

5.1.1. Using Hedwig

Tables 2 and 3 present the best ranked rules discovered by Hedwig using different parameter settings in the first experimental setup using Personalized PageRank and node2vec node relevance scores for the ALL and breast cancer data sets, respectively. The results show for all the terms used by Hedwig that the relative ranks calculated by the P-PR scoring function are remarkably high. In all the cases, the terms used to construct rules are in top

Table 3: The best rules discovered by the Hedwig algorithm on the breast cancer data set. The meaning of columns is the same as explained in the caption of Table 2.

Term	RelativeRank (P-PR)[%]	RelativeRank (Node2vec)[%]	Beam	Depth	Support	Lift
GO:0043230	11.350	31.000	1	1	0.01	1.400
GO:0043230	11.350	31.000	1	10	0.01	1.400
GO:0043230	11.350	31.000	1	1	0.1	1.400
GO:0043230	11.350	31.000	1	10	0.1	1.400
GO:0000785	0.821	24.693	10	1	0.01	1.868
GO:0003674	0.202	36.568	10	10	0.01	3.743
GO:0044427	0.409	20.538				
GO:0000278	0.091	0.665				
GO:0022402	0.312	59.868				
GO:0043228	9.062	25.750	10	1	0.1	1.439
GO:0071840	29.454	58.599	10	10	0.1	1.556
GO:0044428	0.051	13.580				
GO:0003674	0.202	36.568				

percentiles of all term scores, giving credence to our hypothesis that high scoring terms are usually used in rules. On the other hand, use of the node2vec based scoring function was much less successful in this experiment.

This phenomenon is clearly shown in the top two charts of Figure 4 showing all the terms used by Hedwig and their ranks. Even taking into account all (not only the best) rules, we see that predominantly the used terms come from the top 5 percent of all the terms as scored by the Personalized PageRank scoring function. On the other hand, this phenomenon does not occur with the node2vec scoring function, as the results in Table 2 and Table 3 show for the ALL and breast cancer data sets. The rules used by Hedwig score quite low according to the node2vec function, however as seen in the bottom two charts of Figure 4, the node2vec score of the used terms is still above average, and it is possible that the node2vec scores are useful. We consider the node2vec scores to contain sufficient information to warrant testing both scores in the second experimental setup.

The rules discovered in the first experimental setup are biologically relevant, except when the search beam for the Hedwig algorithm was set to 1. In this case the only significant rule discovered contained in its condition a single gene ontology term GO:0003674 that denotes molecular function. This is a very general term that offers little insight and shows that a larger search beam is necessary in order for Hedwig to make significant discoveries. The most interesting results are uncovered when the beam size is set to 10 and the support is set to 0.01. When the depth is set to 1, the most important term GO:0050851 (antigen receptor-mediated signaling pathway) relates to the immune system related cell type. When searching with a depth of 10, we discovered a conjunct of four terms: immune system process (GO:0002376), immune response-activating cell surface receptor signaling pathway, (GO:0002429), plasma membrane (GO:0005886) and binding (GO:0005488). This conjunct provides additional insights about the action (binding), effect (immune response signaling pathway), and location (plasma membrane).

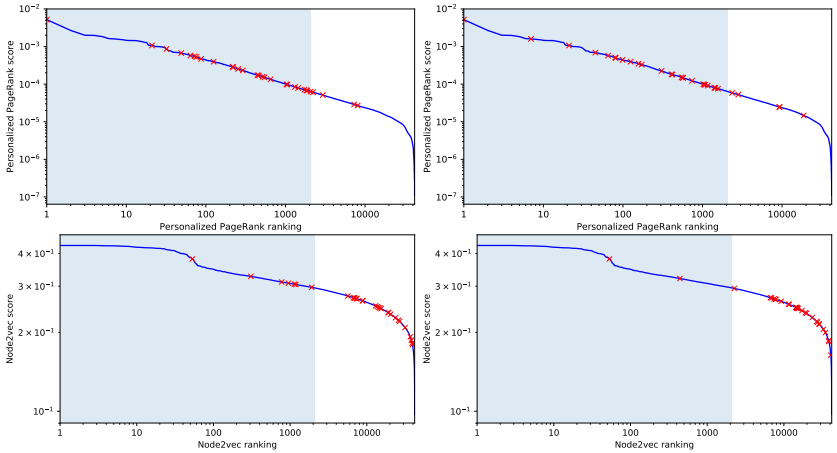


Figure 4: The ranks of the terms used by Hedwig to construct rules on the ALL (left) and breast cancer (right) data sets. The blue line shows the score for each of the terms in the background knowledge—the scores on the top two charts were calculated with Personalized PageRank and the scores on the bottom two charts were produced with node2vec. The terms in each chart are sorted by descending score. Each red cross highlights one of the terms used by Hedwig. The blue area on the left denotes the top 5 percent of all nodes (i.e. nodes with a relative rank of 0.05 or lower). Note the logarithmic scale of both axes.

5.1.2. Using Aleph

Tables 4 and 5 show the terms, used in the 10 best rules discovered by the Aleph algorithm that describe the ALL and breast cancer data sets, respectively, and the P-PR and node2vec scores of the terms in the rules. The scores were calculated taking both *is-a* and *is-part-of* relations into account.

The results again show that the Personalized PageRank scoring function ranks all the terms used by Aleph remarkably high. Moreover, similar to the experiments with Hedwig, the node2vec scoring function is much less successful in this experiment. The high Personalized PageRank ranking of terms appearing in rules discovered by the Aleph algorithm is also seen in Figure 5.

Similar to the results using the Hedwig algorithm, the discovered rules contain terms predominantly from the top 5 percent of all terms as scored by the Personalized PageRank scoring function. A high value of the node2vec scoring function is again a weaker predictor of whether a rule will be useful or not. Nevertheless, as seen in the bottom two charts of Figures 4 and 5, the node2vec score of the used terms is still above average, and it is

possible that the node2vec scores are useful. We consider the node2vec scores to contain sufficient information to warrant testing both scores in the second experimental setup.

5.2. Second setup: Shrinking the background knowledge ontology

This section presents the result of the second set of experiments with the NetSDM methodology. We present the results of shrinking the background knowledge for Hedwig and Aleph.

5.2.1. Ontology shrinking for Hedwig

The results of the first experimental setup show that the scores, calculated using the network analysis techniques, are relevant to determine whether background knowledge nodes will be used by SDM algorithms in the construction of rules describing the target instances. Here we present the results of using the computed scores to shrink the background knowledge used by Hedwig. We first present the results of both scoring functions on the ALL data set and then on the breast cancer data set. We set the beam and depth parameters of Hedwig to 10, and the minimum coverage parameter to 0.01; these are the settings that produced the best results for both data sets in the first experimental setup with Hedwig with no shrinking of the background knowledge (see Tables 2 and 3).

For Personalized PageRank-based network shrinkage we also test if taking into account edge directions makes a difference. Table 6 (undirected edges) and Table 7 (directed edges) show the results of semantic rule discovery using the P-PR function on the ALL data set. Both tables show a similar phenomenon: by decreasing the size of the background knowledge

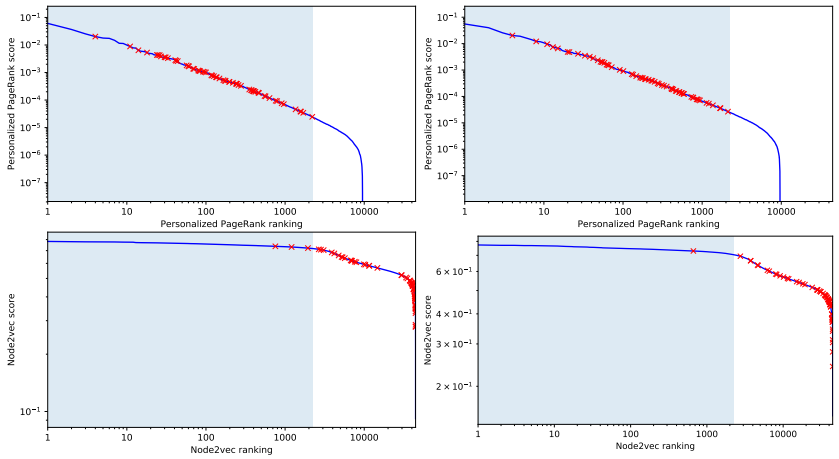


Figure 5: The ranks of the terms used by Aleph to construct rules on the ALL (left) and breast cancer (right) data sets. The chart interpretation is the same as explained in the caption of Figure 4. Note again the logarithmic scale of both axes.

Table 4: The 10 best rules discovered by the Aleph algorithm on the ALL data set. The meaning of the second and third column is the same as in Table 2. The scores were calculated by taking the *is-a* and *is-part-of* relations into account.

Term	RelativeRank (P-PR)[%] %	RelativeRank (Node2vec)[%]	Coverage	TP	Accuracy
GO:0016462	0.444	66.115	65	13	0.884
GO:0031224	0.092	11.634			
GO:0050839	2.161	12.756	69	15	0.884
GO:0031982	0.397	5.980			
GO:0098609	0.855	99.684	76	16	0.883
GO:0016021	0.094	11.982			
GO:0002703	4.948	97.078	86	21	0.883
GO:0030155	0.886	99.031	92	21	0.882
GO:0005634	0.056	6.891			
GO:0031224	0.092	11.634	98	24	0.882
GO:0055114	0.184	99.827			
GO:0050851	0.792	86.346	139	44	0.882
GO:0005488	0.009	94.384			
GO:0044255	0.283	97.785	117	33	0.882
GO:1901615	1.030	99.553			
GO:0006796	0.188	98.970	83	15	0.882
GO:0051129	3.045	97.949			
GO:0007165	0.058	99.764	117	32	0.882
GO:0045321	0.622	92.021			
GO:0044425	0.074	25.869			

Table 5: The 10 best rules discovered by the Aleph algorithm on the breast cancer data set. The meaning of the columns is the same as in Table 4.

Term	RelativeRank (P-PR)[%] %	RelativeRank (Node2vec)[%]	Coverage	TP	Accuracy
GO:0022610	0.399	98.582	39	11	0.914
GO:0005578	1.041	13.932			
GO:0050794	0.045	99.980	57	11	0.912
GO:0042592	0.435	93.017			
GO:0044428	0.085	10.539	77	20	0.912
GO:0000226	1.243	80.543			
GO:0043230	0.512	10.349	69	14	0.911
GO:0043087	0.969	80.674			
GO:0044424	0.018	25.560	93	21	0.911
GO:0044843	2.028	53.978			
GO:0044427	0.330	8.428	103	26	0.911
GO:0050790	0.227	99.147			
GO:0008509	3.826	74.934	76	12	0.910
GO:0044699	0.047	99.996			
GO:0005488	0.009	92.797			
GO:0016491	0.289	99.942	86	14	0.910
GO:0080090	0.426	99.760			
GO:0051186	1.227	97.193	84	13	0.910
GO:0043169	0.108	43.342			
GO:0043226	0.029	17.895	97	19	0.910
GO:0004842	0.615	79.657			

the rules discovered by the Hedwig algorithm either stay the same or change to rules with a higher lift value. When searching for longer rules, reducing the size of the network by

Table 6: Results of NetSDM using direct network conversion with *undirected* edges, Personalized PageRank scoring function, naïve node removal and the Hedwig SDM algorithm on the ALL data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0002376	immune system process	3.235	111	40
	GO:0002694	regulation of leukocyte activation			
	GO:0034110	regulation of homotypic cell-cell adhesion			
10	GO:0002376	immune system process	4.090	90	41
	GO:0002694	regulation of leukocyte activation			
	GO:0044459	plasma membrane part			
20	GO:0003824	catalytic activity	4.257	116	55
	GO:0044283	small molecule biosynthetic process			
	GO:0044444	cytoplasmic part			
50	GO:0002376	immune system process	3.420	105	40
	GO:0002429	immune response-activating cell surface receptor signaling pathway			
	GO:0005886	plasma membrane			
	GO:0005488	binding			
100	GO:0002376	immune system process	3.420	105	40
	GO:0002429	immune response-activating cell surface receptor signaling pathway			
	GO:0005886	plasma membrane			
	GO:0005488	binding			

50% still allows us to discover the same high quality conjunct of the terms GO:0002376, GO:0002429, GO:0005886 and GO:0005488 as before.

Comparing Table 6 and Table 7 we see that there is a slight difference in the resulting rules if we take the direction of the relations in the background knowledge into account. Better Lift values and more consistent results are obtained when we choose to ignore the direction of the relations. Taking edge directions into account, for example, leads to a discovery of a very low quality rule with Lift value of only 2.219 when the background knowledge is at 20% of its original size. When we do not ignore the direction of the relations, the score of each node is only allowed to propagate to the 'parent' nodes in the background knowledge, i.e. high scores will be given to those GO terms whose child terms (specializations) have a high score. On the other hand, when ignoring the direction, we allow nodes to pass their high scores also to sibling nodes, allowing Hedwig in the final step of our NetSDM methodology (outlined in Algorithm 1) to choose the correct specialization of the parent node. More experiments are needed to analyze this phenomenon.

Table 8 and Table 9 show the results of the experiments using the node2vec function on the ALL data set. The results show that node2vec based ontology shrinkage did not achieve the same performance as the shrinkage using the Personalized PageRank function. The results in both tables show that lower shrinkage coefficients consistently decrease the quality of the rules discovered by Hedwig, meaning that shrinking of the background knowledge with the node2vec scoring function removes several informative terms and connections. Without these terms Hedwig could not extract the relevant information from the data, causing decreased performance.

Another interesting phenomenon in the node2vec results is that as the background knowledge decreases in size, the resulting rules become shorter, with the lowest values

Table 7: Results of NetSDM using direct network conversion with *directed* edges, Personalized PageRank scoring function, naïve node removal and the Hedwig SDM algorithm on the ALL data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0003824 GO:0044283 GO:0044444 GO:0044238	catalytic activity small molecule biosynthetic process cytoplasmic part primary metabolic process	3.741	132	55
10	GO:0003824 GO:0044283 GO:0044444 GO:0044238	catalytic activity small molecule biosynthetic process cytoplasmic part primary metabolic process	3.769	131	55
20	GO:0045936 GO:0003824 GO:0009892	negative regulation of phosphate metabolic process catalytic activity negative regulation of metabolic process	2.219	89	22
50	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40
100	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40

Table 8: Results of NetSDM using direct network conversion with *undirected* edges, node2vec scoring function, naïve node removal and the Hedwig SDM algorithm on the ALL data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0050852	T cell receptor signaling pathway	2.586	125	36
10	GO:0050852	T cell receptor signaling pathway	2.586	125	36
20	GO:0050852	T cell receptor signaling pathway	2.586	125	36
50	GO:0050863	regulation of T cell activation	3.076	108	37
100	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40

returning a single GO term as the only important node. This may be a consequence of so many terms pruned that the remaining terms describe non-intersecting sets of enriched genes (or sets with a very small intersection). In this case, term GO:0044281 covers the largest number of instances and is therefore selected as the best term. Upon selecting this term, Hedwig is not able to improve the rule by adding further conjuncts, as adding any other term to the conjunction drastically decreases the coverage of the resulting rule.

Tables 11 and 10 show the results of the Personalized PageRank scoring function used in reducing the background knowledge for the breast cancer data set. As in the ALL data set, we see that decreasing the background knowledge size does not drastically decrease the

Table 9: Results of NetSDM using direct network conversion with *directed* edges, node2vec scoring function, naïve node removal and the Hedwig SDM algorithm on the ALL data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0050852	T cell receptor signaling pathway	2.586	125	36
10	GO:0050852	T cell receptor signaling pathway	2.586	125	36
20	GO:0050852	T cell receptor signaling pathway	2.586	125	36
50	GO:0007507 GO:0032502	heart development developmental process	2.124	93	22
100	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40

Table 10: Results of NetSDM using direct network conversion with *undirected* edges, Personalized PageRank scoring function, naïve node removal and the Hedwig SDM algorithm on the breast cancer data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0071840 GO:0000278 GO:0005515 GO:0044260 GO:0022402	cellular component organization or biogenesis mitotic cell cycle protein binding cellular macromolecule metabolic process cell cycle process	4.114	121	41
10	GO:0043232 GO:0000278 GO:0005515	intracellular non-membrane-bounded organelle mitotic cell cycle protein binding	3.055	151	38
20	GO:0071840 GO:0007049 GO:0003824 GO:0005515	cellular component organization or biogenesis cell cycle catalytic activity protein binding	3.781	122	38
50	GO:0003824 GO:0000278 GO:0022402 GO:0005515	catalytic activity mitotic cell cycle cell cycle process protein binding	3.469	126	36
100	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37

quality of rules discovered by Hedwig. In fact, the rules discovered from only 20% of the original background knowledge achieve almost the same Lift score as the rules produced from the entire data set. Further shrinking the data set allows Hedwig to discover an even better rule. Comparing the two tables we see—even more clearly than in the ALL data set—that ignoring the direction of edges resulted in better overall performance of the algorithm. Especially for small background knowledge sizes, NetSDM finds rules with substantially lower Lift values using scores calculated on directed edges compared to those calculated by ignoring the edge direction.

Table 11: Results of NetSDM using direct network conversion with *directed* edges, Personalized PageRank scoring function, naïve node removal and the Hedwig SDM algorithm on the breast cancer data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0003824 GO:0007049 GO:0044446	catalytic activity cell cycle intracellular organelle part	3.035	136	34
10	GO:0044421 GO:0003824 GO:0070062 GO:0044707	extracellular region part catalytic activity extracellular vesicular exosome single-multicellular organism process	2.587	122	26
20	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
50	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
100	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37

Table 12: Results of NetSDM using direct network conversion with *undirected* edges, node2vec scoring function, naïve node removal and the Hedwig SDM algorithm on the breast cancer data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
10	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
20	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
50	GO:1903047 GO:0005515	mitotic cell cycle process protein binding	2.916	204	49
100	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37

The results of NetSDM using the node2vec scoring funciton on the breast cancer data set, shown in Tables 12 and 13 are comparable to those on the ALL data set. The Hedwig algorithm does not discover important rules to explain the positive examples with the pruned background knowledge. As on the ALL data set, too many important nodes were pruned. Again we observe the phenomenon that as the background knowledge is shrunk, the length of the rules decreases to 1.

While the increased rule quality shows that using the Personalized PageRank as a filter before applying the Hedwig algorithm can improve the performance of the algorithm, the results are also significant considering the fact that with small background knowledge sizes

Table 13: Results of NetSDM using direct network conversion with *directed* edges, node2vec scoring function, naïve node removal and the Hedwig SDM algorithm on the breast cancer data set.

Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
5	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
10	GO:0044699 GO:0000278	single-organism process mitotic cell cycle	3.541	120	35
20	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
50	GO:0000082	G1/S transition of mitotic cell cycle	2.589	136	29
100	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37

the search space of Hedwig (and thus the computational complexity of the algorithm) is much smaller.

Figure 6 shows the relationship between the time taken by the Hedwig semantic rule learning algorithm (together with network preprocessing) to discover the relevant rules and the network shrinkage coefficient c, i.e. the relative size of the background knowledge left for Hedwig to analyze. As the network preprocessing steps were completed in seconds, the times reported in Figure 6 correspond almost entirely to the work of the Hedwig algorithm. The fact that shrinking takes orders of magnitude less time than rule discovery is a prerequisite for the SDM algorithms to be useful in real-life applications. We timed the algorithm on the ALL data set using different settings for the beam, depth and support on 8 core 2.60 GHz Intel Xeon(R)E5-2697 v3 machine with 64GB of RAM. The graph clearly shows that smaller background knowledge ontologies result in much shorter run times (note the logarithmic scale). This confirms our hypothesis that shrinking the background knowledge decreases the time it takes Hedwig to search the space of possible hypotheses. We can conclude that using the NetSDM background knowledge shrinking approach is beneficial both for increasing the quality of the rules and for speeding up the Hedwig algorithm.

5.2.2. Ontology shrinking for Aleph

This section examines the results of the second set of experiments with the Aleph algorithm used in the final step of the NetSDM methodology. Due to low quality of results achieved using node2vec in the first set of experiments with Aleph and Hedwig (presented in Section 5.1) as well as in the second set of experiments with Hedwig, we only tested Personalized PageRank based background knowledge shrinkage with the Aleph algorithm. Furthermore, due to low effect of taking directions of edges into account in the experiments with Hedwig, we only used the undirected network to calculate the scores for the terms in the background knowledge.

Table 14 shows the results of the Aleph algorithm on the ALL data set in the second set of experiments. The results of the algorithm on the breast cancer data set are shown in Table 24 of Appendix B. The results show that unlike with the Hedwig algorithm, using the Aleph algorithm on very heavily pruned data set does not result in the discovery of high quality rules. When the background knowledge is very small, the quality of the discovered rules, as

Table 14: Results of NetSDM on the ALL data set using the P-PR scoring function, direct network conversion and the naïve node removal, and using Aleph to learn a rule set (a theory, where only top 3 best rules are shown).

Shrinkage coefficient [%]	Rule	Coverage	*TP*	Accuracy	Lift
0.5	Theory	6,594	772	0.306	1.024
	GO:0005813 GO:0043167	92	19	0.880	1.807
	GO:0005829 GO:0055114	229	45	0.870	1.719
	GO:0044281 GO:0005829 GO:0044260	216	39	0.870	1.580
1	Theory	4,867	634	0.473	1.140
	GO:0000287 GO:0005575	118	26	0.878	1.928
	GO:0009897 GO:0031224	126	27	0.877	1.875
	GO:0055114 GO:0005829	229	45	0.870	1.719
2	Theory	4,180	601	0.544	1.258
	GO:0002250 GO:0005488	129	33	0.878	2.238
	GO:0005515 GO:0030027	100	24	0.880	2.100
	GO:0005886 GO:0001775 GO:0009987	238	53	0.871	1.949
3	Theory	4,333	599	0.526	1.210
	GO:0005515 GO:0050851	132	39	0.880	2.585
	GO:0030027 GO:0009987	103	25	0.880	2.124
	GO:0046649 GO:0071944	141	34	0.877	2.110
4	Theory	3,977	583	0.563	1.283
	GO:0002253 GO:0071944 GO:0008150	180	52	0.877	2.528
	GO:0030027 GO:0009987	103	25	0.880	2.124
	GO:0005634 GO:0048731 GO:0065007	144	33	0.877	2.005
5	Theory	4,136	589	0.546	1.246
	GO:0002253 GO:0071944	183	54	0.877	2.582
	GO:0030027 GO:0009987	103	25	0.880	2.124
	GO:0071944 GO:0005768	151	32	0.876	1.854
10	Theory	3,789	597	0.588	1.379
	GO:0005515 GO:0050851 GO:0065007	128	37	0.880	2.529
	GO:0009987 GO:0006066	171	45	0.876	2.303
	GO:0044763 GO:0008289 GO:0009653	65	16	0.882	2.154
20	Theory	3,214	548	0.643	1.492
	GO:0050851 GO:0071944	82	36	0.885	3.841
	GO:0080135 GO:0044249	138	35	0.878	2.219
	GO:0030027 GO:0009987	100	24	0.880	2.100
50	Theory	2,797	529	0.686	1.655
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0050851 GO:0005488	139	44	0.880	2.770
	GO:0080135 GO:0044249	138	35	0.878	2.219
100	Theory	2,765	528	0.690	1.671
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0002684 GO:0098552	82	23	0.882	2.454
	GO:0050852 GO:0005515	111	31	0.880	2.444

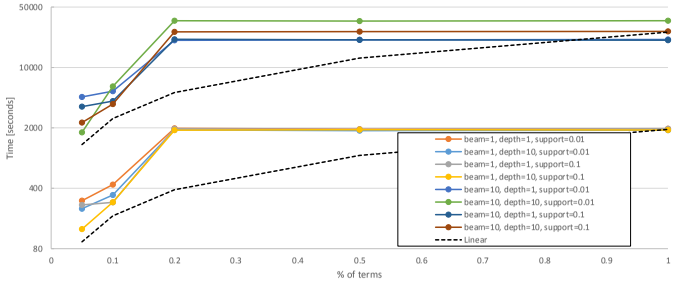


Figure 6: The times (in seconds) on logarithmic scale taken by Hedwig to discover rules on the ALL data set with different settings for beam, depth and support at varying sizes of the background knowledge set (different percentages of terms remaining in the background knowledge) in the second experimental setup. The times are plotted using a logarithmic scale on the y axis, and two linear trendlines (dashed) are drawn for comparison.

well as of the set of all rules, drops significantly. The benefits of network shrinkage for Aleph are visible when the background knowledge is pruned to 20% of its original size. At this background knowledge size, while the time taken to discover the rules was comparable to the time taken on the original background knowledge, the quality of the top two discovered rules is actually higher. Similar results were obtained using the Aleph algorithm on the breast cancer data set, shown in table 24 in Appendix B.

5.3. Third setup: Advanced network conversion and node deletion

This section presents the results of the final set of experiments using the advanced network conversion and node removal on both data sets using both learning algorithms.

5.3.1. Advanced network conversion for Hedwig

The results of the third set of experiments show somewhat different results from those achieved in the second experimental setup. Table 15 shows the results of both network conversion methods on the ALL data set, while Table 16 shows the results of the methods on the breast cancer data set. Unlike in the second set of experiments, the results of this setup show that the quality of rules, and also the discovered rules, remain the same when we reduce the size of background knowledge to one tenth of its original size. When further reducing the network size we see that the direct method of network conversion outperforms the advanced hypergraph based construction; while the hypergraph version of background knowledge conversion returns significantly worse results at 1% of its original size (in the breast cancer data set), the direct version still finds the same rule we find using the original background knowledge. When shrinking the network even further, the quality of the best

Table 15: Results of NetSDM on the ALL data set using the P-PR scoring function with advanced node removal, using Hedwig. We only list the smallest background knowledge size at which a network conversion method (direct or hypergraph) discovers the same rules as with no pruning, and one size below that.

Conversion method	Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
None	1	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40
Direct	0.05	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40
Direct	0.01	GO:0016020 GO:0050851 GO:0005488	membrane antigen receptor-mediated signaling pathway binding	3.413	89	34
Hypergraph	0.2	GO:0002376 GO:0002429 GO:0005886 GO:0005488	immune system process immune response-activating cell surface receptor signaling pathway plasma membrane binding	3.420	105	40
Hypergraph	0.1	GO:0002376 GO:0002694 GO:0005886 GO:0034110	immune system process regulation of leukocyte activation plasma membrane regulation of homotypic cell-cell adhesion	3.395	100	38

discovered rule decreases, but the decrease in quality occurs at a smaller size for the direct network conversion.

Figure 7 shows the run times of Hedwig on the pruned networks in the third set of experiments. The graph shows that unlike with the naïve network reduction methods in the second set of experiments, the time required to discover rules decreases linearly as we lower the shrinkage coefficient. When reducing the background knowledge network to 1% of its original size, the algorithm was able to discover the rules in two minutes, compared to over 11 hours on the unpruned network.

The result of this set of experiments can be compared with the results of the second set of experiments to draw two conclusions. First, the naïve version of network node removal causes Hedwig to discover rules of a slightly higher quality, but the results of shrinking the background knowledge are unpredictable both in terms of the quality of the rules discovered (which for some background knowledge sizes falls below the base value obtained on the unpruned network). Second, the times taken to discover the rules do not decrease significantly until we decrease the size of the background knowledge to 20% of its original size. On the other hand, the advanced version of node removal allows the algorithm to discover the same results as on the unpruned network even for very heavily pruned background knowledge sets. The results remain consistent, and the times taken to discover the rules decrease much more predictably with this node deletion method.

Table 16: Results of NetSDM on the breast cancer data set using the P-PR scoring function with advanced node removal, using Hedwig.

Conversion method	Shrinkage coefficient [%]	GO-term		Lift	Coverage	TP
None	1	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
Direct	0.01	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
Direct	0.005	GO:0022402 GO:0005654	cell cycle process nucleoplasm	2.942	120	30
Hypergraph	0.1	GO:0003674 GO:0044427 GO:0000278 GO:0022402	molecular function chromosomal part mitotic cell cycle cell cycle process	3.743	120	37
Hypergraph	0.05	GO:0022411 GO:0005515 GO:0044422 GO:0032991	cellular component disassembly protein binding mitotic cell cycle macromolecular complex	3.743	120	37

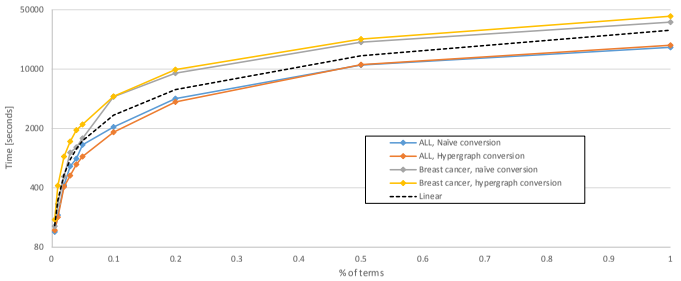


Figure 7: The times (in seconds) on logarithmic scale taken by Hedwig in the third experimental setup to discover rules with different conversion methods at varying sizes of the background knowledge (different percentages of terms remaining in the background knowledge). The times are plotted using a logarithmic scale on the y axis, and a linear trendline (dashed) is drawn for comparison.

5.3.2. Advanced network conversion for Aleph

Table 17 shows the results of the Aleph algorithm on the background knowledge pruned using advanced node reduction and direct network conversion, while Table 18 shows the results of background knowledge reduction using hypergraph network conversion. The results for all possible shrinkage coefficients are shown in Tables 22 and 23 in Appendix B. The results are comparable to the results obtained using the Hedwig algorithm: when the background knowledge is pruned to 10% (for the direct conversion) and 50% (for the hypergraph con-

version) of its original size, Aleph discovers the same best three rules. Furthermore, while shrinking the background knowledge further to only 3% of its original size, the quality of the discovered rules (measured by their Accuracy and Lift value) remains very close to the quality of the rules discovered on the entire background knowledge. The drop in quality is slightly smaller when using the hypergraph network conversion. Interestingly, for both network conversion methods, on a heavily pruned background knowledge (1% and 2% of its original size for the direct network conversion and for the hypergraph network conversion, respectively), the algorithms are able to discover rules of a higher quality than those on the original background knowledge. Similar results were obtained on the breast cancer data set and are shown in Appendix B in Tables 25 and 26.

Table 17: Results of NetSDM on the ALL data set using the P-PR scoring function, direct network conversion and advanced node removal. For each percentage of terms remaining in the reduced ontology, we show only the top 3 rules discovered by Aleph, and the properties of the entire set of rules (Theory).

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2,561	489	0.704	1.671
	GO:0005783, GO:0005789, GO:0009058	63	19	0.883	2.639
	GO:0044459, GO:0002376, GO:0007275	137	38	0.879	2.427
	GO:0006950, GO:0015630	79	19	0.881	2.104
1	Theory	2,673	515	0.697	1.686
	GO:0050851, GO:0071944	82	36	0.885	3.841
	GO:0006066, GO:0044238, GO:0003824	122	40	0.881	2.869
	GO:0001775, GO:0006793	121	35	0.880	2.531
10	Theory	2,760	529	0.690	1.677
	GO:0008610, GO:0044283	127	41	0.881	2.825
	GO:0045321, GO:0005886	117	35	0.880	2.618
	GO:0002250, GO:0005623	119	33	0.880	2.426
100	Theory	2,568	510	0.708	1.738
	GO:0008610, GO:0044283	127	41	0.881	2.825
	GO:0045321, GO:0005886	116	34	0.880	2.565
	GO:0010008, GO:0032991	97	28	0.881	2.526

The run times of Aleph are shown in Figure 8. Compared to Hedwig, shrinking the background knowledge to a relatively small set does not have the same drastic effect on the run time. For example, shrinking the background knowledge to 10% of its original size for the direct network construction, which allows us to discover the same rules as with no ontology shrinkage, only decreases the run time by 14%, compared to the 90% time decrease for the Hedwig algorithm. Using the hypergraph network construction, a slightly larger speed-up was achieved in our experiments, however this is counter-balanced by the fact that for hypergraph network construction, the shrinkage coefficient must be set to a higher value in order to obtain the same rules.

The most promising finding of this series of experiments is that for very heavily pruned background knowledge (1% and 2% of the original size), Aleph (using the hypergraph conversion method) is capable of discovering high quality rules explaining the data. The times taken to discover the rules at this background knowledge size are halved when compared to times on the original background knowledge, and the resulting rules are of a higher quality

Table 18: Results of NetSDM on the ALL data set using the P-PR scoring function, hypergraph network conversion and advanced node removal. For each percentage of terms remaining in the reduced ontology, we show only the top 3 rules discovered by Aleph, and the properties of the entire set of rules (Theory).

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2,577	492	0.703	1.671
	GO:1901362, GO:0044255, GO:0044281	84	26	0.882	2.708
	GO:0005874, GO:0005515, GO:0044707	48	14	0.883	2.552
	GO:0007166, GO:0050776, GO:0016021	138	40	0.879	2.536
2	Theory	2,534	516	0.713	1.782
	GO:0050870, GO:0005886	87	35	0.884	3.520
	GO:0007166, GO:0050778, GO:0044459	141	48	0.881	2.979
	GO:0007017, GO:0006950	48	15	0.884	2.734
50	Theory	2,648	527	0.703	1.741
	GO:0002376, GO:0044459, GO:0050865	140	48	0.881	3.000
	GO:0002429 GO:0071944	138	47	0.881	2.980
	GO:0006066, GO:0003824, GO:0044238	118	38	0.881	2.818
100	Theory	2,616	521	0.705	1.743
	GO:0002376, GO:0044459, GO:0050865	140	48	0.881	3.000
	GO:0002429, GO:0071944	138	47	0.881	2.980
	GO:0006066, GO:0003824, GO:0044238	112	38	0.882	2.969

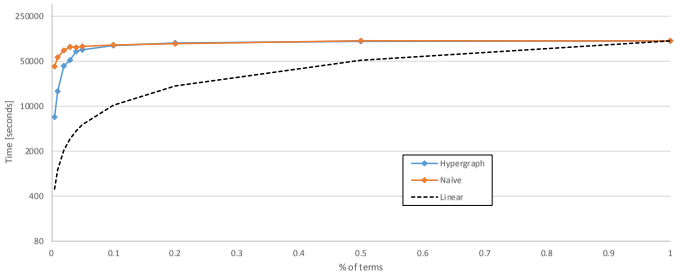


Figure 8: The times (in seconds) taken by the Aleph algorithm for different sizes of the background knowledge to discover rules in the third round of experiments on the ALL data set. Note the logarithmic scale of the y axis of the graph. A linear trendline (dashed) is drawn for comparison.

than even the best rule discovered without ontology shrinking. The main drawback of this result at this point is the fact that the ideal background knowledge size at which this occurs is different for different shrinkage methods, meaning that for a new data set we have to tune the background knowledge size.

6. Conclusions and further work

This paper presents an efficient approach to semantic data mining that uses network analysis for shrinking of background knowledge to significantly reduce the search space and make semantic data mining practically applicable. We tested several variants of combining network analysis with semantic data mining. The result is a four-step NetSDM framework for data analysis: background knowledge conversion, network node importance calculation, network shrinkage, and semantic data mining. We tested NetSDM using two rule learning algorithms: Hedwig and Aleph. In all the setups Hedwig and Aleph were used in their standard modes of operation: Hedwig was used for subgroup discovery and Aleph was used for classification rule learning. Nevertheless, as output, both algorithms return a set of rules explaining the target class examples.

The results of using Hedwig as the semantic data mining algorithm for rule learning in the final step of the NetSDM methodology show that using a direct network conversion, the Personalized PageRank function to estimate term importance, and an advanced version of network shrinkage (using transitivity of underlying relations) provides the most consistent results. Using this setup, the entire algorithm was able to discover the same rules as the original Hedwig algorithm, being faster by a factor of almost 100. If we replace the advanced node removal function with the naïve version, the time savings are less predictable, but the algorithm discovers different rules than if run on the unpruned data set. Consequently, when applying Hedwig the advanced node removal strategy is recommended. The results of comparing NetSDM by taking/not taking network edge directions into account were less conclusive and require further experiments for detailed analyisis.

Using Aleph as the SDM algorithm also yielded interesting results. Using any of the three shrinking methods, our results show that on largely reduced background knowledge, Aleph is capable of discovering high quality rules explaining the data set. Rule discovery in this case is faster by a factor of 2 which is a significant improvement. Using larger background knowledge, NetSDM is still capable of shrinking the background knowledge and (with direct network conversion and the Personalized PageRank scoring function) achieve a noticeable (15%) speed-up in the run time of the Aleph algorithm.

In summary, the purpose of this work is to examine the effect of using network analysis inspired network shrinkage on semantic data mining algorithms in the context of the proposed NetSDM methodology. Due to the different modes of operation of algorithms Hedwig and Aleph (performing subgroup discovery and classification rule learning, respectively), comparing their results does not provide additional insights regarding their operation. Nevertheless, comparing Aleph and Hedwig, our results show that the Hedwig algorithm is very consistent in that Hedwig can discover the same rules, but in a much shorter time, when applying NetSDM to reduce the background knowledge. In contrast, the speed-up—while still noticeable—was much less pronounced when using NetSDM with Aleph. The quality of the discovered rules also decreased more quickly with Aleph used as the final algorithm of NetSDM.

In future work, we plan a more comprehensive examination of how the performance of NetSDM compares to enrichment analysis based methods like SEGS (Trajkovski et al., 2008b) and SegMine (Podpečan et al., 2011). We plan further experiments with different methods for network reduction. For example, other network ranking methods or even other

network analysis methods, such as community detection, could be used to identify the most relevant parts of the background knowledge. The existing two methods, PageRank and node2vec, contain parameters (especially parameters p and q of node2vec) which may affect the resulting scoring function.

Additionally, a non-deterministic approach to background knowledge shrinking could be used instead of the hard shrinkage coefficient used in the current experiments. With a non-deterministic approach, the scores of background knowledge terms may represent (non-scaled) probabilities of sampling a given node. Such an approach has the benefit of not eliminating any background knowledge term outright. However, the probabilistic nature of the sampling would require multiple runs of the entire algorithm, including the SDM rule learner, causing additional computations that would negate the improvements in performance achieved by network shrinking. Alternatively, we could more tightly integrate network analysis scores and the SDM algorithms. Instead of using the relevance scores to prune the ontology and feed the pruned ontology to SDM algorithms, we could use the scores, calculated by the network analysis algorithms, to guide the search in SDM algorithms. The main drawback of this approach would be the lack of modularity: while it is now relatively simple to replace one SDM algorithm in NetSDM with another (for example, replacing Hedwig with Aleph), this would no longer be possible if the scores were an integral part of a SDM algorithm.

We plan applications of the NetSDM methodology to larger data sets. Our experiments so far have been limited to hierarchies (experiments described in Appendix A) and the Gene Ontology, which is a big ontology but relatively simple in structure and logic relationships (in the experiments, we used only two relationships out of five relationships available in GO). Without our shrinking approach, Hedwig required several hours to discover interesting rules using this ontology, and therefore using larger ontologies was not feasible without dramatically reducing the parameters guiding the search. However, using the NetSDM, the algorithm can now be applied to a much wider range of background knowledge sets, including more complex ontologies and large knowledge graphs, such as the BioMine graph (Eronen and Toivonen, 2012) of biological entities and connections between them, as well as the Bio2RDF knowledge graph (Callahan et al., 2013), two knowledge graphs we intend to examine in our further work.

Acknowledgments

We express our thanks to Michel Dumontier for joint work in defining this challenging research problem. We are grateful to Anže Vavpetič for useful discussions and support in using the Hedwig algorithm, to Ashwin Srinivasan for the support and advice in running the experiments with the Aleph algorithm, and to anonymous reviewers whose constructive feedback helped us to improve this manuscript. We acknowledge the financial support from the Slovenian Research Agency through core research programmes P2-0103 and P6-0411 and project *HinLife: Analysis of Heterogeneous Information Networks for Knowledge Discovery in Life Sciences* (J7-7303) and *Semantic Data Mining for Linked Open Data* (financed under the ERC Complementary Scheme, N2-0078).

References

Prem Raj Adhikari, Anže Vavpetič, Jan Kralj, Nada Lavrač, and Jaakko Hollmén. Explaining mixture models through semantic pattern mining and banded matrix visualization. *Machine Learning*, 105(1):3–39, 2016.

Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, San Francisco, California, USA, 1994.

Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, et al. Gene Ontology: Tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.

Alex Bavelas. Communication patterns in task-oriented groups. *Journal of the Acoustical Society of America*, 22:725–730, 1950.

Ronald S. Burt and Michael J. Minor. *Applied Network Analysis: A Methodological Introduction*. Sage Publications, 1983.

Alison Callahan, Jose Cruz-Toledo, Peter Ansell, and Michel Dumontier. Bio2RDF Release 2: Improved coverage, interoperability and provenance of life science linked data. In *ESWC*, volume 7882 of *Lecture Notes in Computer Science*, pages 200–212. Springer, 2013.

Alison Callahan, Juan José Cifuentes, and Michel Dumontier. An evidence-based approach to identify aging-related genes in Caenorhabditis elegans. *BMC Bioinformatics*, 16(1):1, 2015.

Sabina Chiaretti, Xiaochun Li, Robert Gentleman, Antonella Vitale, Marco Vignetti, Franco Mandelli, Jerome Ritz, and Robin Foa. Gene expression profile of adult T-cell acute lymphocytic leukemia identifies distinct subsets of patients with different response to therapy and survival. *Blood*, 103(7):2771–2778, 2004.

Lynne S. Cox and Richard Faragher. From old organisms to new molecules: Integrative biology and therapeutic targets in accelerated human ageing. *Cellular and Molecular Life Sciences*, 64(19-20):2620–2641, 2007.

F. Crestani. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, 11(6):453–482, December 1997.

L. De Raedt. *Logical and Relational Learning*. Springer, 2008.

Dejing Dou, Hao Wang, and Haishan Liu. Semantic data mining: A survey of ontology-based approaches. In *Semantic Computing (ICSC), 2015 IEEE International Conference on*, pages 244–251. IEEE, 2015.

Sašo Džeroski and Nada Lavrač, editors. *Relational Data Mining*. Springer, 2001.

Lauri Eronen and Hannu Toivonen. BioMine: Predicting links between biological entities using network models of heterogeneous databases. *BMC Bioinformatics*, 13:119, 2012.

Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40: 35–41, 1977.

Linton C. Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239, 1979.

Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač. *Foundations of Rule Learning*. Springer, 2012.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, USA, 2016.

Miha Grčar, Nejc Trdin, and Nada Lavrač. A methodology for mining document-enriched heterogeneous information networks. *The Computer Journal*, 56(3):321–335, 2013.

Nicola Guarino, Daniel Oberle, and Steffen Staab. What Is an Ontology? In *Handbook on Ontologies*, pages 1–17. Springer, 2009.

Robert Hoehndorf, Michel Dumontier, and Georgios V. Gkoutos. Identifying aberrant pathways through integrated analysis of knowledge in pharmacogenomics. *Bioinformatics*, 28 (16):2169–2175, 2012.

Da Wei Huang, Brad T. Sherman, and Richard A. Lempicki. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature Protocols*, 4(1): 44–57, 2008.

Wilhelmiina Hämäläinen. *Efficient search for statistically significant dependency rules in binary data*. PhD thesis, Department of Computer Science, University of Helsinki, Finland, 2010.

Glen Jeh and Jennifer Widom. SimRank: A measure of structural-context similarity. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 538–543. ACM, 2002.

Mikhail Jiline, Stan Matwin, and Marcel Turcotte. Annotation concept synthesis and enrichment analysis: A logic-based approach to the interpretation of high-throughput experiments. *Bioinformatics*, 27(17):2391–2398, 2011.

Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1): 39–43, 1953.

Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

Willi Klösgen. Explora: A multipattern and multistrategy discovery assistant. In *Advances in Knowledge Discovery and Data Mining*, pages 249–271. American Association for Artificial Intelligence, 1996.

Risi Imre Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *Proceedings of the 19th International Conference on Machine Learning*, pages 315–322, 2002.

Nada Lavrač and Anže Vavpetič. Relational and semantic data mining. In *Proceedings of the Thirteenth International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 20–31, Lexington, KY, USA, 2015.

Nada Lavrač, Branko Kavšek, Peter Flach, and Ljupčo Todorovski. Subgroup discovery with CN2-SD. *Journal of Machine Learning Research*, 5:153–188, 2004.

Agnieszka Lawrynowicz and Jedrzej Potoniec. Fr-ONT: An algorithm for frequent concept mining with formal ontologies. In *Foundations of Intelligent Systems, Proceedings of 19th International Symposium on Methodologies for Intelligent Systems (2011)*, volume 6804 of *Lecture Notes in Computer Science*, pages 428–437, 2011.

Paea LePendu, Srinivasan V. Iyer, Anna Bauer-Mehren, Rave Harpaz, Jonathan M. Mortensen, Tanya Podchiyska, Todd A. Ferris, and Nigam H Shah. Pharmacovigilance using clinical notes. *Clinical Pharmacology & Therapeutics*, 93(6):547–555, 2013.

Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 80–86, 1998.

Haishan Liu, Dejing Dou, Ruoming Jin, Paea LePendu, and Nigam Shah. Mining biomedical ontologies and data using RDF hypergraphs. In *Proceedings of the 12th International Conference on Machine Learning and Applications (ICMLA), 2013*, volume 1, pages 141–146. IEEE, 2013.

Svetlana Lyalina, Bethany Percha, Paea LePendu, Srinivasan V. Iyer, Russ B. Altman, and Nigam H. Shah. Identifying phenotypic signatures of neuropsychiatric disorders from electronic medical records. *Journal of the American Medical Informatics Association*, 20 (e2):e297–e305, 2013.

Donna Maglott, Jim Ostell, Kim D. Pruitt, and Tatiana Tatusova. Entrez Gene: Gene-centered information at NCBI. *Nucleic Acids Research*, 33(Database issue):D54–D58, 2005.

Stephen Muggleton. Inverse entailment and Progol. *New generation computing*, 13(3-4): 245–286, 1995.

Athanasios N Nikolakopoulos and John D Garofalakis. NCDawareRank: A novel ranking method that exploits the decomposable structure of the web. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, pages 143–152. ACM, 2013.

Hiroyuki Ogata, Susumu Goto, Kazushige Sato, Wataru Fujibuchi, Hidemasa Bono, and Minoru Kanehisa. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, 27(1):29–34, 1999.

David Page, Vítor Santos Costa, Sriraam Natarajan, Aubrey Barnard, Peggy Peissig, and Michael Caldwell. Identifying adverse drug events by relational learning. In *Proceedings of the Twenty-sixth AAAI Conference on Artificial Intelligence*, volume 2012, page 790, Toronto, Canada, 2012.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, November 1999.

Peggy L Peissig, Vitor Santos Costa, Michael D Caldwell, Carla Rottscheit, Richard L Berg, Eneida A Mendonca, and David Page. Relational machine learning for electronic health record-driven phenotyping. *Journal of Biomedical Informatics*, 52:260–270, 2014.

Gregory Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, pages 229–248. Menlo Park, CA: AAI/MIT, 1991.

Vid Podpečan, Nada Lavrač, Igor Mozetič, Petra Kralj Novak, Igor Trajkovski, Laura Langohr, Kimmo Kulovesi, Hannu Toivonen, Marko Petek, Helena Motaln, et al. SegMine workflows for semantic microarray data analysis in Orange4WS. *BMC Bioinformatics*, 12(1):416, 2011.

Monika Puzianowska-Kuznicka and Jacek Kuznicki. Genetic alterations in accelerated ageing syndromes: Do they play a role in natural ageing? *The International Journal of Biochemistry & Cell Biology*, 37(5):947–960, 2005.

Steffen Rendle. Scaling factorization machines to relational data. *Proceedings of the VLDB Endowment*, 6(5):337–348, March 2013. ISSN 2150-8097.

Christos Sotiriou, Pratyaksha Wirapati, Sherene Loi, Adrian Harris, Steve Fox, Johanna Smeds, Hans Nordgren, Pierre Farmer, Viviane Praz, Benjamin Haibe-Kains, et al. Gene expression profiling in breast cancer: Understanding the molecular basis of histologic grade to improve prognosis. *Journal of the National Cancer Institute*, 98(4):262–272, 2006.

Ashwin Srinivasan. The Aleph Manual, 1999. Available at http://www.cs.ox.ac.uk/activities/machinelearning/Aleph/aleph.

Yizhou Sun and Jiawei Han. *Mining Heterogeneous Information Networks: Principles and Methodologies*. Morgan & Claypool Publishers, 2012.

Hannah Tipney and Lawrence Hunter. An introduction to effective use of enrichment analysis software. *Human Genomics*, 4(3):1, 2010.

Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. Fast random walk with restart and its applications. In *Proceedings of the Sixth International Conference on Data Mining*, pages 613–622, Washington, DC, USA, 2006.

Igor Trajkovski, Nada Lavrač, and Jakub Tolar. SEGS: Search for enriched gene sets in microarray data. *Journal of Biomedical Informatics*, 41(4):588–601, 2008a.

Igor Trajkovski, Filip Železný, Nada Lavrač, and Jakub Tolar. Learning relational descriptions of differentially expressed gene groups. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 38(1):16–25, 2008b.

Oron Vanunu, Oded Magger, Eytan Ruppin, Tomer Shlomi, and Roded Sharan. Associating genes and protein complexes with disease via network propagation. *PLoS Computational Biology*, 6(1), 2010.

Anže Vavpetič, Vid Podpečan, and Nada Lavrač. Semantic subgroup explanations. *Journal of Intelligent Information Systems*, 42(2):233–254, 2014.

Anže Vavpetič and Nada Lavrač. Semantic subgroup discovery systems and workflows in the SDM–toolkit. *The Computer Journal*, 56(3):304–320, 2013.

Anže. Vavpetič, Petra Kralj Novak, Miha Grčar, Igor Mozetič, and Nada Lavrač. Semantic data mining of financial news articles. In *Proceedings of Sixteenth International Conference on Discovery Science (DS 2013)*, volume 8140 of *Lecture Notes in Computer Science*, pages 294–307, Singapore, 2013.

Monika Žáková, Filip Železný, Javier A. Sedano, Cyril Masia Tissot, Nada Lavrač, Petr Kremen, and Javier Molina. Relational data mining applied to virtual engineering of product designs. In *Proceedings of the 16th International Conference on Inductive Logic Programming (ILP'06)*, pages 439–453, Santiago de Compostela, Spain, 2006.

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

Stefan Wrobel. An algorithm for multi-relational discovery of subgroups. In *Proceedings of the First European Conference on Principles of Data Mining and Knowledge Discovery (PKDD '97)*, pages 78–87. Springer, 1997.

Wenpu Xing and Ali Ghorbani. Weighted PageRank algorithm. In *Proceedings of the 2nd Annual Conference on Communication Networks and Services Research*, pages 305–314. IEEE, 2004.

Liang Zhang, Bingpeng Ma, Guorong Li, Qingming Huang, and Qi Tian. PL-ranking: A novel ranking method for cross-modal retrieval. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 1355–1364. ACM, 2016.

Appendix A

The Hedwig algorithm was previously used on several publicly available data sets (Adhikari et al., 2016). To show generality of the proposed NetSDM approach, we tested the performance of the NetSDM algorithm on the same data sets. The results, shown in Tables 19, 21, and 20, show similar trends as for the biological data sets which comprise the main part of this paper. The use of NetSDM can drastically decrease the running time of the algorithm with little to no decrease in the quality of the discovered rules. The speed-ups, achieved on the reported data sets, are around the factor of 5, which is less than for the Gene Ontology background knowledge. This lower, though still very promising speed-up can be explained by the fact that (1) the background knowledge in this case is smaller, meaning that less pruning can take place, (2) because of smaller background knowledge, a larger portion of Hedwig's running time consists of pre-processing and post-processing of the data, the tasks that are not affected by background knowledge pruning.

Table 19: Results of NetSDM on the Cities data set using the P-PR scoring function, regular network conversion, and advanced node removal.

Shrinkage coefficient [%]	Time taken [s]	Rule	Coverage	TP	Precision	Lift
3	2.50	City, Municipality, Capital	50	44	0.880	3.054
		City, Municipality, Center	58	51	0.879	3.052
		City, Center	56	49	0.875	3.037
4	3.85	PopulatedPlace, City, Place, Center	49	49	1	3.47
		Underspecified, City, Place, Center	48	48	1	3.47
		PopulatedPlace, City, Place, Capital	42	42	1	3.47
100	17	PopulatedPlace, City, Place, Center	49	49	1	3.47
		Underspecified, City, Place, Center	48	48	1	3.47
		PopulatedPlace, City, Place, Capital	42	42	1	3.47

Table 20: Results of NetSDM on the NY Daily data set using the P-PR scoring function, regular network conversion, and advanced node removal.

Shrinkage coefficient [%]	Time taken [s]	Rule	Coverage	TP	Precision	Lift
5	57.44	Place, LivingPeople,Location	95	95	1	7.196
		Agent, District,CausalAgent,Place,Person	92	92	0.879	7.196
		Place, LivingPeople,Region	92	92	0.879	7.196
10	84.95	Agent, Location,CausalAgent,Place,Person	102	102	1	7.196
		Agebnt Location,CausalAgent,PopulatedPlace	100	100	1	7.196
		Agent, Region, CausalAgent, Place,Person	97	97	1	7.196
100	297.99	Agent, Location,CausalAgent,Place,Person	102	102	1	7.196
		Agebnt Location,CausalAgent,PopulatedPlace	100	100	1	7.196
		Agent, Region, CausalAgent, Place,Person	97	97	1	7.196

Table 21: Results of NetSDM on the Tweets data set using the P-PR scoring function, regular network conversion and advanced node removal.

Shrinkage coefficient [%]	Time taken [s]	Rule	Coverage	TP	Precision	Lift
0.5	13	Athlete, Agent	139	112	0.806	4.268
		Person, Athlete	147	117	0.796	4.215
		CausalAgent, Athlete	149	117	0.785	4.159
10	24	Athlete, Agent	139	112	0.806	4.268
		Athlete, Contestant	141	113	0.801	4.245
		Athlete, Person	147	117	0.796	4.215
100	113	Athlete, Agent	139	112	0.806	4.268
		Athlete, Contestant	141	113	0.801	4.245
		Athlete, Person	147	117	0.796	4.215

Appendix B

Table 22: Results of NetSDM on the ALL data set using the Personalized PageRank scoring function, direct network conversion, and advanced node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2561	489	0.704	1.671
	GO:0005783 GO:0005789 GO:0009058	63	19	0.883	2.639
	GO:0044459 GO:0002376 GO:0007275	137	38	0.879	2.427
	GO:0006950 GO:0015630	79	19	0.881	2.104
1	Theory	2673	515	0.697	1.686
	GO:0050851 GO:0071944	82	36	0.885	3.841
	GO:0006066 GO:0044238 GO:0003824	122	40	0.881	2.869
	GO:0001775 GO:0006793	121	35	0.880	2.531
2	Theory	2814	547	0.688	1.701
	GO:0002250 GO:0044464	119	33	0.880	2.426
	GO:0045321 GO:0005829	105	28	0.880	2.333
	GO:1901360 GO:0008610	125	33	0.879	2.310
3	Theory	2879	549	0.681	1.669
	GO:0016021 GO:0002684 GO:0044459	137	43	0.880	2.746
	GO:0045321 GO:0044459	117	35	0.880	2.618
	GO:0002250 GO:0044464	119	33	0.880	2.426
4	Theory	2816	547	0.688	1.700
	GO:0016021 GO:0002684 GO:0044459	137	43	0.880	2.746
	GO:0045321 GO:0005886	117	35	0.880	2.618
	GO:0002250 GO:0044464	119	33	0.880	2.426
5	Theory	2852	543	0.683	1.666
	GO:0002684 GO:0016021 GO:0044459	137	43	0.880	2.746
	GO:0045321 GO:0005886	117	35	0.880	2.618
	GO:0002250 GO:0005623	119	33	0.880	2.426
10	Theory	2760	529	0.690	1.677
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0045321 GO:0005886	117	35	0.880	2.618
	GO:0002250 GO:0005623	119	33	0.880	2.426
20	Theory	2620	522	0.705	1.743
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0045321 GO:0005886	117	35	0.880	2.618
	GO:0010008 GO:0032991	97	28	0.881	2.526
50	Theory	2576	510	0.707	1.732
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0045321 GO:0005886	116	34	0.880	2.565
	GO:0010008 GO:0032991	97	28	0.881	2.526
100	Theory	2568	510	0.708	1.738
	GO:0008610 GO:0044283	127	41	0.881	2.825
	GO:0045321 GO:0005886	116	34	0.880	2.565
	GO:0010008 GO:0032991	97	28	0.881	2.526

Table 23: Results of NetSDM on the ALL data set using the Personalized PageRank scoring function, hypergraph network conversion, and advanced node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2,577	492	0.703	1.671
	GO:1901362 GO:0044255 GO:0044281	84	26	0.882	2.708
	GO:0005874 GO:0005515 GO:0044707	48	14	0.883	2.552
	GO:0007166 GO:0050776 GO:0016021	138	40	0.879	2.536
1	Theory	2,627	515	0.702	1.715
	GO:1901362 GO:0044255 GO:0044281	84	26	0.882	2.708
	GO:0007166 GO:0050776 GO:0016021	138	40	0.879	2.536
	GO:0032403 GO:0005886 GO:0032991	114	30	0.880	2.303
2	Theory	2,534	516	0.713	1.782
	GO:0050870 GO:0005886	87	35	0.884	3.520
	GO:0007166 GO:0050778 GO:0044459	141	48	0.881	2.979
	GO:0007017 GO:0006950	48	15	0.884	2.734
3	Theory	2,728	539	0.696	1.729
	GO:0007166 GO:0050778 GO:0044459	141	48	0.881	2.979
	GO:0050863 GO:0016020	140	46	0.880	2.875
	GO:0006066 GO:0003824	129	40	0.880	2.713
4	Theory	2,548	513	0.711	1.762
	GO:0016021 GO:0002684 GO:0005886	137	43	0.880	2.746
	GO:0006066 GO:0003824	129	40	0.880	2.713
	GO:0001775 GO:0019899	132	35	0.879	2.320

% of terms	Rule	Cov	Pos	Acc	Lift
5	Theory	2,522	512	0.714	1.776
	GO:0016021 GO:0002684 GO:0005886	137	43	0.880	2.746
	GO:0006066 GO:0003824	129	40	0.880	2.713
	GO:0001775 GO:0019899	132	35	0.879	2.320
10	Theory	2,642	538	0.706	1.782
	GO:0050865 GO:0002376 GO:0005886	144	49	0.880	2.977
	GO:0006066 GO:0003824 GO:0044238	121	40	0.881	2.893
	GO:0051606 GO:0006950	46	14	0.884	2.663
20	Theory	2,675	527	0.700	1.724
	GO:0044459 GO:0002684 GO:0050865	139	47	0.881	2.959
	GO:0006066 GO:0003824 GO:0044238	120	39	0.881	2.844
	GO:0051606 GO:0006950	46	14	0.884	2.663
50	Theory	2,648	527	0.703	1.741
	GO:0002376 GO:0044459 GO:0050865	140	48	0.881	3.000
	GO:0002429 GO:0071944	138	47	0.881	2.980
	GO:0006066 GO:0003824 GO:0044238	118	38	0.881	2.818
100	Theory	2,616	521	0.705	1.743
	GO:0002376 GO:0044459 GO:0050865	140	48	0.881	3.000
	GO:0002429 GO:0071944	138	47	0.881	2.980
	GO:0006066 GO:0003824 GO:0044238	112	38	0.882	2.969

Table 24: Results of NetSDM on the breast cancer data set using the Personalized PageRank scoring function, direct network conversion, and naïve node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	6,439	659	0.463	1.177
	GO:0070062 GO:0005576 GO:0005634	84	19	0.909	2.602
	GO:0005794 GO:0044428	162	29	0.904	2.060
	GO:0005743 GO:1901363	109	18	0.907	1.900
1	Theory	5,196	575	0.558	1.273
	GO:0051301 GO:0050794	217	41	0.901	2.174
	GO:0005525 GO:0044421	110	20	0.907	2.092
	GO:0046983 GO:0035639 GO:0044424	118	21	0.906	2.048
2	Theory	3,902	522	0.662	1.539
	GO:0000082 GO:0005515	89	21	0.909	2.715
	GO:0008283 GO:0006139	185	37	0.903	2.301
	GO:0008201 GO:0044763 GO:0050896	70	14	0.909	2.301
3	Theory	5,078	603	0.573	1.366
	GO:1901363 GO:0032446	142	35	0.907	2.836
	GO:0005634 GO:0005794 GO:0044446	117	25	0.907	2.458
	GO:0003723 GO:0031967	101	21	0.908	2.392
4	Theory	4,605	582	0.611	1.454
	GO:0005634 GO:0005794 GO:0044446	117	25	0.907	2.458
	GO:0012501 GO:0043232	125	26	0.907	2.393
	GO:0044430 GO:0000278	195	40	0.903	2.360

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
5	Theory	5,137	611	0.569	1.368
	GO:1903047 GO:0000166	129	31	0.907	2.765
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:0050896 GO:0000785	169	34	0.904	2.315
10	Theory	4,629	588	0.610	1.461
	GO:0006890 GO:0043229	60	15	0.910	2.876
	GO:1903047 GO:0000166	129	31	0.907	2.765
	GO:0032993	88	21	0.909	2.746
20	Theory	3,475	528	0.701	1.748
	GO:0000910 GO:0043226	50	12	0.911	2.761
	GO:0005654 GO:0008283	158	37	0.906	2.694
	GO:0032984 GO:0043232	87	19	0.909	2.513
50	Theory	3,006	526	0.742	2.013
	GO:0051301 GO:0032268	85	26	0.910	3.519
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:1903561 GO:0048468	95	22	0.909	2.664
100	Theory	2,637	484	0.767	2.112
	GO:0032993 GO:0009987	76	21	0.910	3.179
	GO:0015931 GO:0031967	43	11	0.911	2.943
	GO:1903047 GO:0035639	120	30	0.908	2.876

Table 25: Results of NetSDM on the breast cancer data set using the Personalized PageRank scoring function, direct network conversion, and advanced node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
0.5	Theory	2,455	422	0.772	1.978
	GO:0000166 GO:0000278	129	31	0.907	2.765
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:0051174 GO:0007049	68	15	0.910	2.538
1	Theory	2,724	474	0.757	2.002
	GO:1903047 GO:0035639	120	30	0.908	2.876
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:0070647 GO:0033554	116	25	0.907	2.480
2	Theory	2,761	484	0.756	2.017
	GO:0005829 GO:0044106	31	10	0.912	3.711
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105
3	Theory	2,749	497	0.759	2.080
	GO:0009308 GO:0005829	34	11	0.912	3.722
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105
4	Theory	2,758	498	0.758	2.077
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105
5	Theory	2,765	492	0.757	2.047
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:0015630 GO:0007059	70	21	0.911	3.452
	GO:0044770 GO:0031981	126	34	0.908	3.105
10	Theory	2,838	501	0.752	2.031
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:0043232 GO:1903047 GO:0051276	75	21	0.910	3.221
	GO:0044770 GO:0031981	126	34	0.908	3.105
20	Theory	2,715	482	0.759	2.043
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:0015630 GO:0007059	70	21	0.911	3.452
	GO:0044770 GO:0031981	126	34	0.908	3.105
50	Theory	2,880	510	0.750	2.037
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105
100	Theory	2,755	498	0.759	2.080
	GO:0005829 GO:0009308	34	11	0.912	3.722
	GO:0051276 GO:0000278	136	41	0.908	3.468
	GO:0044770 GO:0031981	126	34	0.908	3.105

Table 26: Results of NesSDM on the breast cancer data set using the Personalized PageRank scoring function, hypergraph network conversion, and advanced node removal. For each percentage of background knowledge terms left in the ontology, we list only the top 3 rules discovered and also report on the properties of the entire set of rules (Theory) discovered by Aleph.

Shrinkage coefficient [%]	Rule	Coverage	TP	Accuracy	Lift
5	Theory	2,392	430	0.779	2.068
	GO:0008283 GO:1901360	32	10	0.912	3.595
	GO:0005829 GO:1903047 GO:0051301	100	28	0.909	3.221
	GO:0031410 GO:0015031 GO:0005789	43	12	0.911	3.211
1	Theory	2,431	439	0.777	2.078
	GO:0005794 GO:0005634 GO:0044422	85	22	0.909	2.978
	GO:0006260 GO:0032991	82	21	0.910	2.946
	GO:0005654 GO:0003674 GO:1902589	96	24	0.909	2.876
2	Theory	2,721	492	0.761	2.080
	GO:0008283 GO:0006725	32	10	0.912	3.595
	GO:0051276 GO:0000278	136	41	0.908	3.468
	GO:0032993 GO:0009987 GO:0005634	73	21	0.910	3.310
3	Theory	2,791	491	0.754	2.024
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0010604 GO:0051726	86	21	0.909	2.809
	GO:0016567 GO:1901363	114	26	0.908	2.624
4	Theory	2,928	505	0.745	1.984
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0031323 GO:0051726	87	21	0.909	2.777
	GO:0016567 GO:1901363	114	26	0.908	2.624
5	Theory	2,920	499	0.744	1.966
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0005730 GO:0032879	81	20	0.909	2.841
	GO:0044237 GO:0051726	86	18	0.909	2.408
10	Theory	2,885	497	0.747	1.982
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0080090 GO:0051726	84	21	0.909	2.876
	GO:0005488 GO:0022411 GO:0043232	115	28	0.908	2.801
20	Theory	2,694	470	0.759	2.007
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0005730 GO:0051179	94	22	0.909	2.693
	GO:0043226 GO:0051726	82	19	0.909	2.666
50	Theory	2,912	503	0.746	1.987
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0000280 GO:0043170	78	22	0.910	3.245
	GO:0043226 GO:0051726	82	19	0.909	2.666
100	Theory	2,927	505	0.745	1.985
	GO:1903047 GO:0051276	136	41	0.908	3.468
	GO:0003677 GO:0070647 GO:0044424	89	21	0.909	2.715
	GO:0043226 GO:0051726	82	19	0.909	2.666

Journal of Machine Learning Research 20 (2019) 1-38 Submitted 3/17; Revised 1/19; Published 1/19

The Relationship Between Agnostic Selective Classification, Active Learning and the Disagreement Coefficient

Roei Gelbhart roeige@cs.technion.ac.il
Department of Computer Science
Technion — Israel Institute of Technology

Ran El-Yaniv rani@cs.technion.ac.il
Department of Computer Science
Technion — Israel Institute of Technology

Editor: Gabor Lugosi

Abstract

A selective classifier (f, g) comprises a classification function f and a binary selection function g, which determines if the classifier abstains from prediction, or uses f to predict. The classifier is called pointwise-competitive if it classifies each point identically to the best classifier in hindsight (from the same class), whenever it does not abstain. The quality of such a classifier is quantified by its rejection mass, defined to be the probability mass of the points it rejects. A "fast" rejection rate is achieved if the rejection mass is bounded from above by $\tilde{O}(1/m)$ where m is the number of labeled examples used to train the classifier (and $\tilde{O}$ hides logarithmic factors). Pointwise-competitive selective (PCS) classifiers are intimately related to disagreement-based active learning and it is known that in the realizable case, a fast rejection rate of a known PCS algorithm (called Consistent Selective Strategy) is equivalent to an exponential speedup of the well-known CAL active algorithm.

We focus on the agnostic setting, for which there is a known algorithm called LESS that learns a PCS classifier and achieves a fast rejection rate (depending on Hanneke's disagreement coefficient) under strong assumptions. We present an improved PCS learning algorithm called ILESS for which we show a fast rate (depending on Hanneke's disagreement coefficient) without any assumptions. Our rejection bound smoothly interpolates the realizable and agnostic settings. The main result of this paper is an equivalence between the following three entities: (i) the existence of a fast rejection rate for any PCS learning algorithm (such as ILESS); (ii) a poly-logarithmic bound for Hanneke's disagreement coefficient; and (iii) an exponential speedup for a new disagreement-based active learner called Active-ILESS.

Keywords: active learning, selective prediction, disagreement coefficient, selective sampling, selective classification, reject option, pointwise-competitive, selective classification, statistical learning theory, PAC learning, sample complexity, agnostic case

1. Introduction

Selective classification is a unique and extreme instance of the broader concept of confidence-rated prediction (Chow, 1970; Vovk et al., 2005; Bartlett and Wegkamp, 2008; Yuan and Wegkamp, 2010; Cortes et al., 2016a; Wiener and El-Yaniv, 2012; Kocak et al., 2016; Zhang and Chaudhuri, 2014). Given a training sample consisting of m labeled instances, the learning algorithm is required to output a *selective classifier* (El-Yaniv and Wiener, 2010), defined to be a pair (f, g), where f is a prediction function, chosen from some hypothesis class $\mathcal{F}$, and $g : \mathcal{X} \to \{0, 1\}$ is a *selection function*, serving as a qualifier for f as follows: for any x, if $g(x) = 1$, the classifier predicts $f(x)$, and otherwise it abstains. The general performance of a selective classifier is quantified in terms of its *coverage* and *risk*, where coverage is the probabilistic mass of non-rejected instances, and risk is the normalized average loss of f restricted to non-rejected instances. Let f^* be any (unknown) true risk minimizer[1] in $\mathcal{F}$ for the given problem. The selective classifier (f, g) is said to be *pointwise-competitive* if, for each x with $g(x) = 1$, it must hold that $f(x) = f^*(x)$ for all $f^* \in \mathcal{F}$ (Wiener and El-Yaniv, 2015). Thus, pointwise-competitiveness w.h.p. over choices of the training sample, is a highly desirable property: it guarantees, for each non-rejected test point, the best possible classification obtainable using the best in-hindsight classifier from $\mathcal{F}$. We do not restrict g to be from any specific hypothesis class, however, because we use disagreement-based selective prediction, the selection of $\mathcal{F}$ will limit the possibilities of g. The scenario of a predefined decision functions hypothesis class is investigated in Cortes et al. (2016b).

Pointwise-competitive selective classification (PCS) was first considered in the realizable case (El-Yaniv and Wiener, 2010), for which a simple consistent selective strategy (CSS) was shown to achieve a bounded and monotonically increasing (with m) coverage in various non-trivial settings. Note that in the realizable case, any PCS strategy attains zero risk (over the sub-domain it covers). These results were recently extended to the agnostic setting (Wiener and El-Yaniv, 2015; El-Yaniv and Wiener, 2011) with a related but different algorithm called *low-error selective strategy (LESS)*, for which a number of coverage bounds were shown. These bounds relied on the fact that the underlying probability distribution and the hypothesis class $\mathcal{F}$ will satisfy the so-called "(β_1, β_2)-Bernstein property" (Bartlett et al., 2004). The coverage bounds used by Wiener and El-Yaniv (2015); El-Yaniv and Wiener (2011) are dependent on the parameters β_1, β_2. This Bernstein property assumption (as presented in Bartlett et al., 2004), which allows for better concentration, nevertheless, can be problematic. First, it is defined with respect to a unique true risk minimizer f^*, a property that is unlikely to hold in noisy agnostic settings. Moreover, for arbitrary $\mathcal{F}$, even for the 0/1 loss function,there is little knowledge about cases for which the property holds with a non-trivial β_2.[2] We removed the Bernstein assumption from our analysis.

Assuming that a selective classifier is w.h.p. pointwise-competitive, our key goal is a small rejection rate. We will say that a learner has a **fast R^* rejection rate**, if w.h.p. the rejection rate is bounded by

$$\text{polylog}\left(\frac{1}{R(f^*) + 1/m}\right) \cdot R(f^*) + \frac{d \cdot \text{polylog}(m, 1/\delta)}{m},$$

1. We assume that there exists an f^* in $\mathcal{F}$. Otherwise, we can artificially define f^* to be any function whose risk is sufficiently close to $\inf_{f \in \mathcal{F}}(R(f))$, for instance, not greater than a small additive factor from this infimum.
2. It was mentioned by Wiener and El-Yaniv (2015) that, under the Tsybakov noise condition (Tsybakov, 2004), the desired property holds, but this is guaranteed only for cases in which the Bayes classifier is within $\mathcal{F}$, which is a fairly strong assumption in itself.

©2019 Roei Gelbhart, Ran El-Yaniv.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-147.html.

where $R(f^*)$, d and δ are defined in Section 2. Selective classification is very closely related to the field of **active learning (AL)**. In active learning, the learner can actively influence the learning process by selecting the points to be labeled. The incentive for introducing this extra flexibility is to reduce labeling efforts. A key question in theoretical studies of AL is how many label requests are sufficient to learn a given (unknown) target concept to a specified accuracy, a quantity called *label complexity*. For an AL algorithm satisfying the "passive example complexity" property (consuming the same number of unlabeled (and labeled) examples, as a passive algorithm consumes labeled examples for achieving the same error; see Definition 6.2), we will say it has R^* **exponential speedup**, if w.h.p. the number of labels it requests is bounded by

$$\text{polylog}\left(\frac{1}{R(f^*)+1/m}\right)\cdot R(f^*)m + d\cdot\text{polylog}(m,1/\delta).$$

The connection between active learning and confidence-rated prediction is quite intuitive. A pointwise-competitive selective classifier P can be straightforwardly used as the querying component of an active learning algorithm. This reduction is most naturally demonstrated in the stream-based AL model: at each iteration, the active algorithm trains a selective classifier on the currently available labeled samples, and then decides to query a newly introduced (unlabeled) point x if P abstains on x.

Hanneke's **disagreement coefficient** (Hanneke, 2007), see Definition 2.1, is a well-known parameter of the hypothesis class and the marginal distribution; it is used in most of the known label complexity bounds (Hsu, 2010; Hanneke, 2007; Ailon et al., 2012). The disagreement coefficient is the supremum of the relation between the disagreement mass of functions that are r-distanced from f^* to r, over r. PCS classification is based on using generalization bounds to estimate the empirical error of f^*, and more specifically, its distance from the empirical error of the ERM. Whenever all the functions that reside within a ball around the ERM unanimously agree, the classifier chooses to classify. Thus, the abstain rate is dependent on the disagreement mass of the functions within the ball. The radius of the ball depends on the generalization bounds. The generalization bounds we use are of the form $\tilde{O}(R(f^*)+d/m)$ for the agnostic case. After observing m examples, we can bound the disagreement mass of a ball around the ERM, by multiplying the radius of the ball, which is $\tilde{O}(R(f^*)+d/m)$, with the disagreement coefficient. Thus, if for example, the disagreement coefficient is bounded by a constant, the abstain rate of some PCS algorithms can be bounded by $\tilde{O}(R(f^*)+d/m)$. This gives a basic idea of the disagreement coefficient, which will be formally presented later on.

Note that, in principle, the disagreement coefficient can be replaced by another important quantity, namely, the **version space compression set size**, recently shown to be equivalent to it (Wiener et al., 2015; El-Yaniv and Wiener, 2015). Specifically, an $O(\text{polylog}(m)log(1/\delta))$ version space compression set size minimal bound was shown by Wiener et al. (2015, Corollary 11), to be equivalent to an $O(\text{polylog}(1/r))$ disagreement coefficient.

Zhang and Chaudhuri (2014) present a new algorithm that uses LP in order to achieve better label complexity analysis than was previously known. This paper proposes a quantity, denoted φ_c, to replace the disagreement coefficient (see a streamlined definition of φ_c at the top of page 24 in Hanneke, 2016), which is smaller than the disagreement coefficient. They

show that for the case of linear classification under log-concave distributions, their analysis can improve (reduce) the argument inside the logarithm of the label complexity bound, and the dependency on the VC-dimension is also reduced by a square root. However, in our paper we focus mainly on the more basic question: *when* can we achieve an exponential speedup in terms of $1/\epsilon$. Thus, due to the simplicity of the disagreement coefficient, and its widespread use in the literature, we chose to focus on it. See a detailed discussion on the φ_c quantity in Section 9.

The first contribution of this paper is a novel selective classifier, called ILESS, which uses a tighter generalization error bound than LESS and depends on $R(f^*)$ (and interpolates the agnostic and realizable cases). Most importantly, the new strategy can be analyzed completely without the Bernstein condition.

We derive an active learning algorithm, called Active-ILESS, corresponding to our selective classifier, ILESS. Active-ILESS is constructed to work in a stream-based AL model and its querying function is extremely conservative: for each unlabeled example, the algorithm requests its label if and only if the labeling of the optimal classifier (from the same class) on this point cannot be inferred from information already acquired. This querying strategy, which is often termed "disagreement-based," has been used in a number of stream-based AL algorithms such as Agnostic CAL and Oracular CAL (Hsu, 2010), A^2 (*Agnostic Active*), developed by Balcan et al. (2006), RobustCAL, studied in Hanneke (2012, 2014b) and Hanneke and Yang (2012), or the general agnostic AL algorithm of Dasgupta et al. (2007). Paper Huang et al. (2015) presented a computationally efficient algorithm for disagreement-based AL.

We prove that Active-ILESS, despite being very similar to Oracular CAL Hsu (2010), exhibits an improved label complexity, in comparison to that proved for Oracular CAL. Specifically, Active-ILESS achieves the same label complexity as Agnostic CAL, while being simpler in the sense that its consumption of ERM computations is smaller.

The first formal relationship between PCS classification and AL was proposed by El-Yaniv and Wiener (2012); Wiener (2013), where the aforementioned CSS algorithm was shown to be equivalent to the well-known CAL AL algorithm of Cohn et al. (1994), in the sense that a fast coverage rate for CSS was proven to be equivalent to an exponential label complexity speedup for CAL. This result applies to the realizable setting only. Our first contribution is a similar equivalence relation between pointwise-competitive selective classification and AL, which applies to the more challenging agnostic case and smoothly interpolates the realizable and agnostic settings.

Our second and main contribution is to show a complete equivalence between (i) selective classification with a fast R^* rejection rate, (ii) an AL algorithm, Active-ILESS, with an R^* exponential speedup, and (iii) the existence of an f^* with a disagreement coefficient bounded by $\text{polylog}(1/r)$. This is illustrated in Figure 1, where the blue errors indicate the equivalence relationships we prove in this paper, and the red arrow indicates a previously known result (Hsu, 2010; Hanneke, 2007), and can also be deduced from the other arrows.

2. Definitions

Consider a domain $\mathcal{X}$, and a binary label set $\mathcal{Y}=\{\pm 1\}$. A learning problem is specified via a hypothesis class $\mathcal{F}$ and an unknown probability distribution $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$. Given a sequence

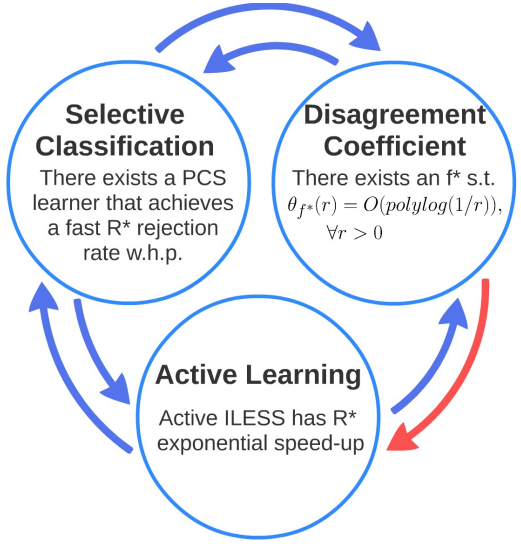


Figure 1: Main results

of labeled training examples $S_m = ((x_1, y_1), (x_2, y_2), ..., (x_m, y_m))$, such that $\forall i, (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, the empirical error of a hypothesis f over S_m is $\hat{R}(f, S_m) \triangleq \frac{1}{m}\sum_{i=1}^{m} \ell(f(x_i), y_i)$, where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ is a loss function. In this paper, we focus mainly on the zero-one loss function, $\ell_{01}(y, y') \triangleq \mathbb{1}\{y \neq y'\}$. The true (zero-one) error of f is $R(f) \triangleq \mathbb{E}_{\mathcal{P}}\left[\ell_{01}(f(x), y)\right]$. An empirical risk minimizer hypothesis (henceforth, an ERM) is

$$\hat{f}(S_m) \triangleq \operatorname*{argmin}_{f \in \mathcal{F}} \hat{R}(f, S_m), \tag{1}$$

and a true risk minimizer is $f^* \triangleq \operatorname{argmin}_{f \in \mathcal{F}} R(f)$.[3]

We acquire the following definitions from Wiener and El-Yaniv (2015). For any hypothesis class $\mathcal{F}$, hypothesis $f \in \mathcal{F}$, distribution $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$, sample S_m, and real number $r > 0$, define the true and empirical *low-error sets*,

$$\mathcal{V}(f, r) \triangleq \left\{f' \in \mathcal{F} : R(f') \leq R(f) + r\right\} \tag{2}$$

and

$$\hat{\mathcal{V}}(f, r) \triangleq \left\{f' \in \mathcal{F} : \hat{R}(f', S_m) \leq \hat{R}(f, S_m) + r\right\}. \tag{3}$$

Let $G \subseteq \mathcal{F}$. The *disagreement set* (Hanneke, 2007) and *agreement set* (El-Yaniv and Wiener, 2010) w.r.t. G are defined, respectively, as

$$DIS(G) \triangleq \{x \in \mathcal{X} : \exists f_1, f_2 \in G \quad , \quad f_1(x) \neq f_2(x)\} \tag{4}$$

3. We assume that f^* exists, and that it need not be unique, in which case f^* refers to any one of the minimizers.

$$\text{and} \quad AGR(G) \triangleq \{x \in \mathcal{X} : \forall f_1, f_2 \in G \quad , \quad f_1(x) = f_2(x)\}. \tag{5}$$

In *selective classification* (El-Yaniv and Wiener, 2010), the learning algorithm receives S_m and is required to output a *selective classifier*, defined to be a pair (f, g), where $f \in \mathcal{F}$ is a classifier, and $g : \mathcal{X} \to \{0, 1\}$ is a *selection function*, serving as a qualifier for f as follows. For any $x \in \mathcal{X}$, $(f, g)(x) = f(x)$ iff $g(x) = 1$. Otherwise, the classifier outputs "I don't know". For any selective classifier (f, g), we define its coverage to be

$$\Phi(f, g) \triangleq \Pr_{X \sim \mathcal{P}_{\mathcal{X}}}(g(X) = 1),$$

and its complement, $1 - \Phi$, is called the **abstain rate**. For any $f \in \mathcal{F}$ and $r > 0$, define the set $B(f, r)$ of all hypotheses that reside within a ball of radius r around f,

$$B(f, r) \triangleq \left\{f' \in \mathcal{F} : \Pr_{X \sim \mathcal{P}_{\mathcal{X}}}\left\{f'(X) \neq f(X)\right\} \leq r\right\}.$$

For any $G \subseteq \mathcal{F}$, and distribution $\mathcal{P}_{\mathcal{X}}$, we denote by ΔG the volume of the disagreement set of G (see (4)), $\Delta G \triangleq \Pr\{DIS(G)\}$.

Definition 2.1 (Disagreement Coefficient) *Let $r_0 \geq 0$. Hanneke's* disagreement coefficient *(Hanneke, 2007) of a classifier $f \in \mathcal{F}$ with respect to the target distribution $\mathcal{P}_{\mathcal{X}}$ is*

$$\theta_f(r_0) \triangleq \sup_{r > r_0} \frac{\Delta B(f, r)}{r}, \tag{6}$$

and the general disagreement coefficient *of the entire hypothesis class $\mathcal{F}$ is*

$$\theta(r_0) \triangleq \sup_{f \in \mathcal{F}} \theta_f(r_0). \tag{7}$$

Notice that this definition of the disagreement coefficient is independent of $\mathcal{P}_{\mathcal{Y}|\mathcal{X}}$. Another commonly used definition of the disagreement coefficient does depend on a true risk minimizer f^*, as follows:

$$\theta'(r_0) = \sup_{r > r_0} \frac{\Delta B(f^*, r)}{r}. \tag{8}$$

Clearly, it always holds that $\theta' \leq \theta$. The independence of θ of unknown quantities such as the underlying distribution (and f^*), however, is a convenient property that sometimes allows for a direct estimation of θ, which only depends on the marginal distribution, $\mathcal{P}_{\mathcal{X}}$. This is, for example, the case in AL, where labels are expensive but information about the marginal distribution (provided by unlabeled examples) is cheap. Note also that the above definition of θ' implicitly assumes a unique f^*. Nevertheless, the definition can be extended to cases where f^* is not unique, in which case the infimum over all f^* can be considered (the analysis can be extended accordingly using limits). For more on the disagreement coefficient, and examples of probabilities distributions and hypothesis classes for which it is bounded, see Hanneke (2014b).

3. Convergence Bounds and LESS

We use a uniform convergence bound from Vapnik and Chervonenkis (1974); Dasgupta et al. (2007); Bousquet et al. (2003). Define convergence slacks $\sigma_{R-\hat{R}}(m,\delta,d,R,\hat{R})$ and $\sigma_{\hat{R}-R}(m,\delta,d,R,\hat{R})$, given in terms of the training sample, S_m, its size, m, the confidence parameter, δ, and the VC-dimension d of the class $\mathcal{F}$. For any $f \in \mathcal{F}$,

$$\sigma_{R-\hat{R}}(m,\delta,d,R,\hat{R}) \triangleq \min\left\{\underbrace{\frac{4d\ln(\frac{16me}{d\delta})}{m}+\sqrt{\frac{4d\ln(\frac{16me}{d\delta})}{m}\cdot\hat{R}}}_{\hat{\sigma}_{R-\hat{R}}(m,\delta,d,\hat{R})},\underbrace{\sqrt{\frac{4d\ln(\frac{16me}{d\delta})}{m}\cdot R}}_{\bar{\sigma}_{R-\hat{R}}(m,\delta,d,R)}\right\} \tag{9}$$

and

$$\sigma_{\hat{R}-R}(m,\delta,d,R,\hat{R}) \triangleq \min\left\{\underbrace{\frac{4d\ln(\frac{16me}{d\delta})}{m}+\sqrt{\frac{4d\ln(\frac{16me}{d\delta})}{m}\cdot R}}_{\bar{\sigma}_{\hat{R}-R}(m,\delta,d,R)},\underbrace{\sqrt{\frac{4d\ln(\frac{16me}{d\delta})}{m}\cdot\hat{R}}}_{\hat{\sigma}_{\hat{R}-R}(m,\delta,d,\hat{R})}\right\}. \tag{10}$$

To simplify the analysis, we further decompose the above slack terms into their empirical and non-empirical components. For (9), we thus have, respectively,

$$\hat{\sigma}_{R-\hat{R}}(m,\delta,d,\hat{R}) \triangleq \frac{4d\ln(\frac{16me}{d\delta})}{m}+\sqrt{\frac{4d\ln(\frac{16me}{d\delta})}{m}\cdot\hat{R}} \tag{11}$$

and

$$\hat{\sigma}_{\hat{R}-R}(m,\delta,d,\hat{R}) \triangleq \sqrt{\frac{4d\ln(\frac{16me}{d\delta})}{m}\cdot\hat{R}}. \tag{12}$$

Similarly, the non-empirical part in these minimums are denoted by $\bar{\sigma}_{R-\hat{R}}$ and $\bar{\sigma}_{\hat{R}-R}$. With this notation, we can write, for example, $\sigma_{R-\hat{R}} = \min\{\hat{\sigma}_{R-\hat{R}}, \bar{\sigma}_{R-\hat{R}}\}$. Our Lemma 1 is taken from the work of Dasgupta et al. (2007, Lemma 1), which is based on Bousquet et al. (2003, Theorem 7) [4].

Lemma 1 *Let $\mathcal{F}$ be a hypothesis class with VC-dimension d. For any $0 < \delta < 1$, with probability of at least $1-\delta$ over the choice of S_m from $\mathcal{P}^m$, any hypothesis $f \in \mathcal{F}$ satisfies*

$$R(f) \le \hat{R}(f) + \sigma_{R-\hat{R}}\left(m,\delta,d,R(f),\hat{R}(f)\right) \tag{13}$$

$$\hat{R}(f) \le R(f) + \sigma_{\hat{R}-R}\left(m,\delta,d,R(f),\hat{R}(f)\right). \tag{14}$$

4. In the original lemma from Dasgupta et al. (2007), $S(\mathcal{H},n)$, the growth function, is given. We insert Sauer's Lemma, $S(\mathcal{H},n) \le (\frac{em}{d})^d$, into Lemma 1 from Dasgupta et al. (2007) to get our lemma.

Strategy 1 is the LESS algorithm of Wiener and El-Yaniv (2015). LESS learns w.h.p. a pointwise-competitive selective classifier, (f,g), where $f \in \mathcal{F}$ and $g : \mathcal{X} \to \{0,1\}$ is its selection function that determines whether to abstain or to classify. A *pointwise-competitive selective classifier* must satisfy the following condition: For each x with $g(x)=1$, it must hold that $f(x) = f^*(x)$ for all $f^* \in \mathcal{F}$. A PCS learning algorithm must output a PCS classifier w.h.p. **for all** $\mathcal{P}_{\mathcal{Y}|\mathcal{X}}$; otherwise, one can consider a tailor-made trivial algorithm for each distribution, which simply returns f^*.

Remark 2 *The original definition of pointwise-competitiveness from Wiener and El-Yaniv (2015) requires a single f^*. We widen the definition to cases for which there are more than one f^*, and require that a pointwise-competitive selective classifier be equal to* all *f^*, wherever $g=1$. This extrapolation seems a bit strict. Nevertheless, even if the requirement would have been relaxed to "any f^*", any pointwise-competitive selective classifier would still have been forced to identify with all f^*, as it is impossible to determine whether a set of functions are all f^*, or one is better than the rest.*

The main idea behind LESS is that, w.h.p., all f^* lie within a ball around an ERM hypothesis with an error radius of $2\sigma(m,\delta/4,d)$, where

$$\sigma(m,\delta,d) \triangleq 2\sqrt{\frac{2d\left(\ln\frac{2me}{d}\right)+\ln\frac{2}{\delta}}{m}} \tag{15}$$

is the slack term of a certain uniform convergence bound. Therefore, if all the functions in that ball agree over the labeling of any instance x, we know with high probability that all f^* label x the same way as the ERM. This property ensures that LESS is pointwise-competitive w.h.p.

Strategy 1 Agnostic Low-Error Selective Strategy (LESS)

Input: Sample set of size m, S_m,
Confidence level δ
Hypothesis class $\mathcal{F}$ with VC dimension d
Output: A selective classifier (h,g)
1: Set $\hat{f} = ERM(\mathcal{F},S_m)$, i.e., $\hat{f}$ is any empirical risk minimizer from $\mathcal{F}$
2: Set $G = \hat{\mathcal{V}}\left(\hat{f}, 2\sigma(m,\delta/4,d)\right)$
3: Construct g such that $g(x)=1 \iff x \in \{\mathcal{X} \setminus DIS(G)\}$
4: $f = \hat{f}$

4. ILESS

We now introduce an improved version of LESS, called ILESS, which uses a radius of the form $d\cdot\text{polylog}(m,1/\delta)\cdot\left(\frac{1}{m}+\sqrt{\frac{R(f^*)}{m}}\right)$. Noting that the radius, $2\sigma(m,\delta/4,d)$, used by LESS to define $G=\hat{\mathcal{V}}$, is of the form $d\cdot\text{polylog}(m,1/\delta)/\sqrt{m}$, we observe that in cases where $R(f^*)\approx\frac{C}{m}$, this new radius behaves as $\frac{d\cdot\text{polylog}(m,1/\delta)}{m}$. We later show that this radius allows ILESS to achieve a faster rejection decay rate than the one achieved by LESS.

Consider the pseudo-code of ILESS given in Strategy 2. We now analyze ILESS, and begin by showing in Lemma 3 that ILESS is pointwise-competitive w.h.p., i.e., for any x

Strategy 2 Improved Low-Error Selective Strategy (ILESS)

Input: Sample set of size m, S_m,
Confidence level δ
Hypothesis class $\mathcal{F}$ with VC dimension d
Output: A selective classifier (h, g)
1: Set $\hat{f} = ERM(\mathcal{F}, S_m)$, i.e., $\hat{f}$ is any empirical risk minimizer from $\mathcal{F}$
2: Set $\sigma_{\text{ILESS}} = \hat{\sigma}_{R-\hat{R}}\left(m, \delta, d, \hat{R}(\hat{f}, S_m)\right) + \bar{\sigma}_{\hat{R}-R}\left(m, \delta, d, \hat{R}(\hat{f}, S_m) + \hat{\sigma}_{R-\hat{R}}(m, \delta, d, \hat{R}(\hat{f}, S_m))\right)$
3: Set $G = \hat{\mathcal{V}}\left(\hat{f}, \sigma_{\text{ILESS}}\right)$
4: Construct g such that $g(x) = 1 \Longleftrightarrow x \in \{\mathcal{X} \setminus DIS(G)\}$
5: $h = \hat{f}$

for which $g(x) = 1$, $f(x) = f^*(x)$ for all f^*. The calculation of g appears to be very problematic, as for a specific x, a unanimous decision over an infinite number of functions must be ensured. This problem was shown to be reducible to finding an ERM under one constraint (Lemma 6.1 in El-Yaniv and Wiener, 2011, a.k.a. the disbelief principle). This is a difficult problem, nonetheless, albeit one that could be estimated with heuristics.

Definition 4.1 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, and $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown probability distribution. Given a sample set S_m, drawn from $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$, we denote by $\mathcal{E}$ the event where both inequalities (13) and (14) of Lemma 1 simultaneously hold. We know from the lemma that $\mathcal{E}$ occurs with probability of at least $1 - \delta$.*

Lemma 3 (ILESS **is pointwise-competitive)** *Given that event $\mathcal{E}$ occurred (see Definition 4.1), for all $f^* \in \mathcal{F}$, f^* resides within G (from Strategy 2), and therefore,* ILESS *is pointwise-competitive w.h.p.*

Proof From (14), it follows that

$$\begin{aligned}\hat{R}(f^*, S_m) &\leq R(f^*) + \sigma_{\hat{R}-R}(m, \delta, d, R(f^*), \hat{R}(f^*, S_m)) \\ &\leq R(f^*) + \bar{\sigma}_{\hat{R}-R}\left(m, \delta, d, R(f^*)\right). \end{aligned} \tag{16}$$

Additionally, by the definition of f^*, we know that it has the lowest true error, and using Inequality (13) from Lemma 1 we obtain,

$$\begin{aligned}R(f^*) &\leq R(\hat{f}) \\ &\leq \hat{R}(\hat{f}, S_m) + \sigma_{R-\hat{R}}(m, \delta, d, R(\hat{f}), \hat{R}(\hat{f}, S_m)) \\ &\leq \hat{R}(\hat{f}, S_m) + \hat{\sigma}_{R-\hat{R}}(m, \delta, d, \hat{R}(\hat{f}, S_m)). \end{aligned} \tag{17}$$

Finally, by applying (17) in (16), we have,

$$\begin{aligned}\hat{R}(f^*, S_m) \leq \hat{R}(\hat{f}, S_m) &+ \hat{\sigma}_{R-\hat{R}}(m, \delta, d, \hat{R}(\hat{f}, S_m)) \\ &+ \bar{\sigma}_{\hat{R}-R}\left(m, \delta, d, \hat{R}(\hat{f}, S_m) + \hat{\sigma}_{R-\hat{R}}(m, \delta, d, \hat{R}(\hat{f}, S_m))\right),\end{aligned}$$

which means that $f^* \in G$. ■

Lemma 4 below bounds the radius σ_{ILESS} of ILESS. The lemma uses the notation

$$A \triangleq 4d \ln(\frac{16me}{d\delta}),$$

with which, by the definition of σ_{ILESS} (see Strategy 2), we have,

$$\begin{aligned}\sigma_{\text{ILESS}} &= \hat{\sigma}_{R-\hat{R}}(m, \delta, d, \hat{R}(\hat{f}, S_m)) + \bar{\sigma}_{\hat{R}-R}\left(m, \delta, d, \hat{R}(\hat{f}, S_m) + \hat{\sigma}_{R-\hat{R}}(m, \delta, d, \hat{R}(\hat{f}, S_m))\right) \\ &= \frac{A}{m} + \sqrt{\frac{A}{m} \cdot \hat{R}(\hat{f}, S_m)} + \frac{A}{m} + \sqrt{\frac{A}{m} \cdot \left[\hat{R}(\hat{f}, S_m) + \frac{A}{m} + \sqrt{\frac{A}{m} \cdot \hat{R}(\hat{f}, S_m)}\right]}. \end{aligned} \tag{18}$$

Lemma 4 *Given that event $\mathcal{E}$ (see Definition 4.1) occurred, the radius of* ILESS *satisfies*

$$\sigma_{\text{ILESS}} \leq 6\frac{A}{m} + 3\sqrt{\frac{A}{m} \cdot R(f^*)} = O\left(\frac{A}{m} + \sqrt{\frac{A}{m} \cdot R(f^*)}\right), \tag{19}$$

where $A \triangleq 4d \ln(\frac{16me}{d\delta})$.

Proof Under our assumption, inequalities (13) and (14) hold for every $f \in \mathcal{F}$. We thus have

$$\hat{R}(\hat{f}, S_m) \leq \hat{R}(f^*, S_m) \leq R(f^*) + \frac{A}{m} + \sqrt{\frac{A}{m} \cdot R(f^*)}. \tag{20}$$

Replacing the three occurrences of $\hat{R}(f^*, S_m)$ in (18) with the R.H.S. of (20), and using the basic inequalities $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ and $\sqrt{ab} \leq a/2 + b/2$, we get,

$$\begin{aligned}
\sigma_{\text{ILESS}} &\leq \frac{A}{m} + \sqrt{\frac{A}{m} \cdot \left(R(f^*) + \frac{A}{m} + \sqrt{\frac{A}{m} \cdot R(f^*)} \right)} + \frac{A}{m} + \\
&\quad + \sqrt{\frac{A}{m} \cdot \left[R(f^*) + \frac{2A}{m} + \sqrt{\frac{A}{m} \cdot R(f^*)} + \sqrt{\frac{A}{m} \cdot \left(R(f^*) + \frac{A}{m} + \sqrt{\frac{A}{m} \cdot R(f^*)} \right)} \right]} \\
&\leq \frac{2A}{m} + \sqrt{\frac{A}{m} \cdot \left(R(f^*) + \frac{A}{m} + \frac{A}{2m} + \frac{1}{2} R(f^*) \right)} + \\
&\quad + \sqrt{\frac{A}{m} \cdot \left[R(f^*) + \frac{3A}{2m} + \frac{1}{2} R(f^*) + \frac{A}{m} + \sqrt{\frac{A}{m} \cdot \left(R(f^*) + \frac{3A}{2m} + \frac{1}{2} R(f^*) \right)} \right]} \\
&\leq \frac{2A}{m} + \frac{3A}{2m} + \frac{3}{2}\sqrt{\frac{A}{m} R(f^*)} + \sqrt{\frac{A}{m} \cdot \left[\frac{5A}{2m} + \frac{3}{2} R(f^*) + \sqrt{\frac{A}{m} \cdot \left(\frac{3A}{2m} + \frac{3}{2} R(f^*) \right)} \right]} \\
&\leq \frac{7A}{2m} + \frac{3}{2}\sqrt{\frac{A}{m} \cdot R(f^*)} + \sqrt{\frac{A}{m} \cdot \left[\frac{5A}{2m} + \frac{3}{2} R(f^*) + \frac{3A}{2m} + \sqrt{\frac{3A}{2m} \cdot R(f^*)} \right]} \\
&\leq \frac{7A}{2m} + \frac{3}{2}\sqrt{\frac{A}{m} \cdot R(f^*)} + \sqrt{\frac{A}{m} \cdot \left[\frac{5A}{2m} + \frac{3}{2} R(f^*) + \frac{3A}{2m} + \frac{3A}{4m} + \frac{3}{4} R(f^*) \right]} \\
&\leq \frac{7A}{2m} + \frac{3}{2}\sqrt{\frac{A}{m} \cdot R(f^*)} + \sqrt{\frac{19}{4}} \frac{A}{m} + \sqrt{\frac{A}{m} \cdot \frac{9}{4} R(f^*)} \\
&\leq 6\frac{A}{m} + 3\sqrt{\frac{A}{m} \cdot R(f^*)}. \qquad (21)
\end{aligned}$$

■

In comparison, the radius of LESS is of order $O(\sqrt{\frac{A}{m}})$, which can be significantly larger when $R(f^*)$ is small. This potential radius advantage translates into a potential coverage advantage of ILESS, as stated in the following theorem.

Theorem 5 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, and let $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown probability distribution. Given that event $\mathcal{E}$ (see Definition 4.1) occurred, G defined in Strategy 2, holds that*

$$G \subseteq B(f^*, R_0),$$

where

$$R_0 \triangleq 2 \cdot R(f^*) + 11 \cdot \frac{A}{m} + 6 \cdot \sqrt{\frac{A}{m} \cdot R(f^*)},$$

and thus, for all f^*, the abstain rate is bounded by

$$1 - \Phi(\text{ILESS}) \leq \theta_{f^*}(R_0) \cdot R_0.$$

This immediately implies (by definition) that

$$1 - \Phi(\text{ILESS}) \leq \theta(R_0) \cdot R_0.$$

Remark 6 *Note that $R_0 = O\left(R(f^*) + \frac{A}{m}\right)$ due to $\sqrt{\frac{A}{m} \cdot R(f^*)} \leq \frac{1}{2}(\frac{A}{m} + R(f^*))$.*

Proof We start by showing that G, defined in Strategy 2, resides within a ball around any specific f^*. To do so, we need to bound the true error of all functions in G.

$$\begin{aligned}
f \in G \quad &\Rightarrow \quad \hat{R}(f, S_m) \leq \hat{R}(\hat{f}, S_m) + \sigma_{\text{ILESS}} \qquad (22) \\
&\Rightarrow \quad \hat{R}(f, S_m) \leq R(f^*) + \frac{A}{m} + \sqrt{\frac{A}{m} \cdot R(f^*)} + 6\frac{A}{m} + 3\sqrt{\frac{A}{m} \cdot R(f^*)} \qquad (23) \\
&\Rightarrow \quad \hat{R}(f, S_m) \leq R(f^*) + 7 \cdot \frac{A}{m} + 4 \cdot \sqrt{\frac{A}{m} \cdot R(f^*)}, \qquad (24)
\end{aligned}$$

where Inequality (22) is explained by the definition of G, and inequality (23) follows from (20) and (21) (under event $\mathcal{E}$). We then have,

$$\begin{aligned}
R(f) &\leq \hat{R}(f, S_m) + \hat{\sigma}_{R-\hat{R}}(m, \delta, d, \hat{R}) \qquad (25) \\
&\leq \hat{R}(f, S_m) + \frac{A}{m} + \sqrt{\frac{A}{m} \cdot \hat{R}(f, S_m)} \qquad (26) \\
&\leq R(f^*) + 8 \cdot \frac{A}{m} + 4 \cdot \sqrt{\frac{A}{m} \cdot R(f^*)} + \\
&\quad + \sqrt{\frac{A}{m} \cdot \left[R(f^*) + 7 \cdot \frac{A}{m} + 4 \cdot \sqrt{\frac{A}{m} \cdot R(f^*)} \right]} \qquad (27) \\
&\leq R(f^*) + 8 \cdot \frac{A}{m} + 4 \cdot \sqrt{\frac{A}{m} \cdot R(f^*)} + \sqrt{\frac{A}{m} \cdot \left[3R(f^*) + 9 \cdot \frac{A}{m} \right]} \qquad (28) \\
&\leq R(f^*) + 11 \cdot \frac{A}{m} + 6 \cdot \sqrt{\frac{A}{m} \cdot R(f^*)}, \qquad (29)
\end{aligned}$$

where inequality (25) is (13) (which holds given $\mathcal{E}$), inequality (26) follows directly from the definition of $\hat{\sigma}_{R-\hat{R}}$, inequality (27) is obtained using (24), Inequality (28) follows from $\sqrt{ab} \leq a/2 + b/2$, and (29) from $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$.

Using (29), for all $f \in G$, and any f^*, we have,

$$\begin{aligned}
\Pr_{X \sim \mathcal{P}_{\mathcal{X}}} \{f(X) \neq f^*(X)\} &= \Pr_{X,Y \sim \mathcal{P}_{\mathcal{X},\mathcal{Y}}} \{f(X) \neq f^*(X) \wedge f^*(X) = Y\} + \\
&\quad \Pr_{X,Y \sim \mathcal{P}_{\mathcal{X},\mathcal{Y}}} \{f(X) \neq f^*(X) \wedge f^*(X) \neq Y\} \\
&\leq \Pr_{X,Y \sim \mathcal{P}_{\mathcal{X},\mathcal{Y}}} \{f(X) \neq f^*(X) \wedge f^*(X) = Y\} + R(f^*) \\
&\leq \Pr_{X,Y \sim \mathcal{P}_{\mathcal{X},\mathcal{Y}}} \{f(X) \neq Y\} + R(f^*) \\
&= R(f) + R(f^*) \\
&\leq 2 \cdot R(f^*) + 11 \cdot \frac{A}{m} + 6 \cdot \sqrt{\frac{A}{m} \cdot R(f^*)}. \qquad (30)
\end{aligned}$$

It follows that

$$f \in B\left(f^*, 2 \cdot R(f^*) + 11 \cdot \frac{A}{m} + 6 \cdot \sqrt{\frac{A}{m} \cdot R(f^*)}\right) = B(f^*, R_0),$$

and, in particular,

$$G \subseteq B(f^*, R_0),$$

so,

$$\Delta G \leq \Delta B(f^*, R_0).$$

The abstain rate of ILESS equals ΔG. We can now use the disagreement coefficient to bound the abstain rate from above,

$$\Delta G \leq \Delta B(f^*, R_0) = \frac{\Delta B(f^*, R_0)}{R_0} \cdot R_0 \leq \theta(R_0) \cdot R_0, \tag{31}$$

which concludes the proof. ■

According to Theorem 5, assuming the disagreement coefficient is $\theta(r) = O(\text{polylog}(1/r))$ for $r \geq R(f^*)$, the rejection mass of ILESS, defined as the probability that the classifier trained by ILESS will output "I don't know" is bounded w.h.p. by

$$\text{polylog}_1\left(\frac{1}{R(f^*) + 1/m}\right) \cdot R(f^*) + \frac{d \cdot \text{polylog}_2(m, 1/\delta)}{m}. \tag{32}$$

In many cases, the disagreement coefficient, $\theta(r)$, is bounded by a constant, or by $O(\text{polylog}(1/r))$ for all $r > 0$ (see Hanneke, 2014b). For example, it was shown in Wiener et al. (2015), that for linear separators under mixture of Gaussians, and for axis-aligned rectangles with probability mass bounded away from zero under product densities over $\mathbb{R}^k$, $\theta(r)$ is bounded by $O(\text{polylog}(1/r))$ for all $r > 0$. For such cases, we know that (32) always holds, regardless of the size of $R(f^*)$. The disagreement coefficient is only dependent on the marginal $\mathcal{P}_\mathcal{X}$, the hypothesis class $\mathcal{F}$, and the identity of the true risk minimizers, f^* (which is not necessarily unique). This fact motivates the following definition of a rejection rate of a selective learning algorithm, which is only dependent on $\mathcal{P}_\mathcal{X}$,$\mathcal{F}$ and f^*.

Definition 4.2 (Fast R^* Rejection Rate) *Given $\mathcal{P}_\mathcal{X}$,$\mathcal{F}$ and f^*, if for all $\delta > 0$, and all $\mathcal{P}_{\mathcal{Y}|\mathcal{X}}$ for which f^* is a true risk minimizer, the rejection mass of a selective classifier learning algorithm is bounded by (32) with probability of at least $1 - \delta$, we say that the algorithm achieves a **fast** R^* **rejection rate**, with* polylog_1 *and* polylog_2 *as its parameters.*

Corollary 7 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d. Given $\mathcal{P}_\mathcal{X}$ and f^*, if $\theta_{f^*}(r)$ is bounded by $O(\text{polylog}(1/r))$ for all $r > 0$, then there exists a* PCS *learning algorithm (*ILESS*) which achieves a fast R^* rejection rate.*

Proof The proof is immediate from Theorem 5. ■

In the next section, we will show the other direction; that is, if there is a PCS learning algorithm that has a fast R^* rejection rate, then $\theta(r) = O(\text{polylog}(1/r))$ for all $r > 0$.

As long as the number of training examples that ILESS receives is not "too large" relative to $1/R(f^*)$, i.e., $m \ll \frac{1}{R(f^*)}$, the rejection mass of ILESS is

$$O\left(\frac{d \cdot \text{polylog}(m, 1/\delta)}{m}\right).$$

When m is large, and $R(f^*)$ becomes more dominant than $\frac{1}{m}$, our coverage bound is dominated by $R(f^*)$. This should not surprise us, as ILESS achieves *pointwise-competitiveness* w.h.p., and any strategy that achieves pointwise-competitiveness cannot ensure a better rejection mass than $R(f^*)$ without making more assumptions about the error or the distribution. This can be seen in the following example, in which $\theta(r) \leq 1$ for all $r > 0$, but the rejection mass of any pointwise-competitive strategy is always at least $R(f^*)$.

Example 1 *Given any $0 < \epsilon < 0.5$, let $\mathcal{X} = [0, 1]$, and $\mathcal{F} = \{f_1, f_2\}$ where*

$$f_1(x) = \begin{cases} 1, & x < \epsilon \\ 0, & \textit{otherwise} \end{cases}, f_2(x) = \begin{cases} 1, & x > 1 - \epsilon \\ 0, & \textit{otherwise.} \end{cases}$$

Let $\mathcal{P}_\mathcal{X}$ be the uniform distribution over $[0, 1]$. Assume that Y will always be zero. f_1 and f_2 are both f^*. Every pointwise-competitive classifier will have to output $g(x) = 0$ for every x in the disagreement set of f_1 and f_2. $R(f^*) = \epsilon$, and the rejection mass is $2\epsilon (= 2R(f^*))$.

5. From Selective Classification to the Disagreement Coefficient

We now turn to show a reduction from selective classification, to the disagreement coefficient.

Theorem 8 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, and $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown distribution. Let* PCS *be an algorithm that returns a pointwise-competitive selective classifier w.h.p. for all distributions whose marginal is $\mathcal{P}_\mathcal{X}$. If for every $m \leq 1/R(f^*)$, $0 < \delta$, with probability of at least $1-\delta$, the abstain rate $1-\Phi$ of* $\text{PCS}(S_m, \delta, \mathcal{F}, d)$ *is bounded by a monotonic*[5] *polylog as follows:*

$$1 - \Phi(\text{PCS}) \leq \frac{\text{polylog}_0(\text{m}, 1/\delta)}{\text{m}}. \tag{33}$$

Then for every f^* (every true risk minimizer), for every $r \geq R(f^*)$,

$$\theta_{f^*}(r) \leq 20\left(\text{polylog}_0(1/r, 1/r) + 3\right).$$

Proof For any $m \in \{4, 5, ..., \lfloor 1/R(f^*) \rfloor\}$, denote by $\mathcal{S}_m$ a random training sample drawn from $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$. Let Z be a random variable representing a single random unlabeled example sampled from $\mathcal{P}_\mathcal{X}$, and let f^* to be a specific true risk minimizer.

For $Z \in DIS\left(B(f^*, \frac{1}{m})\right)$, we use the following argument from Hanneke (2012, Lemma 47). We know that there exists a function $h_Z \in \mathcal{F}$ s.t. $h_Z(Z) \neq f^*(Z)$ and $Pr(h_Z(X) \neq f^*(X)) \leq \frac{1}{m}$. We denote by $\mathcal{P}_{\mathcal{X},\mathcal{Y}_z}$ a new probability distribution, where the marginal, $\mathcal{P}_\mathcal{X}$,

5. If the polylog is not monotonic, the following holds: $\theta_{f^*}(r) \leq 16(\text{polylog}(\lfloor 1/r \rfloor, \lfloor 1/r \rfloor) + 3) \cdot \frac{5}{4}$; see Eq. (44).

remains the same, but Y becomes $Y \triangleq h_Z(x)$. Clearly, h_Z is a true risk minimizer (f^*) for such a distribution.

Denote by e_1 the probability event where (33) holds (for a specific $m \leq 1/R(f^*)$). Denote by e_2 the event where PCS has succeeded in returning a pointwise-competitive selective classifier (f_{s_m}, g_{s_m}) under S_m.

Define S'_m to be a modified S_m where, for every $(x,y) \in S_m$ s.t. $h_Z(x) \neq y$, y is modified to become $y = h_Z(x)$. S'_m is a random training sample drawn from $\mathcal{P}_{\mathcal{X},\mathcal{Y}_z}$. Denote by e_{3z} the event where PCS has succeeded in returning a pointwise-competitive selective classifier $(f_{s'_m}, g_{s'_m})$ under S'_m. h_Z is only defined for cases in which $Z \in DIS\left(B(f^*, \frac{1}{m})\right)$, and thus we define that e_{3z} will vacuously hold when $Z \notin DIS\left(B(f^*, \frac{1}{m})\right)$.

Under our assumptions, $\Pr(e_1), \Pr(e_2) \geq 1-\delta$. For every $Z \in DIS(B(f^*, \frac{1}{m}))$, $\Pr(e_{3z}|Z) \geq 1-\delta$, and for every $Z \notin DIS(B(f^*, \frac{1}{m}))$, $\Pr(e_{3z}|Z) = 1$, which implies that $\Pr(e_{3z}) \geq 1-\delta$. We denote by $h_Z(X_m) = Y_m$ the event where $h_Z(x) = y$ for all $(x_i, y_i) \in S_m$. The explanations for the following (in)equalities follow.

$$
\begin{aligned}
& \Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right) \wedge h_Z(X_m) = Y_m\right\} && (34)\\
= \;& \Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right) \wedge h_Z(X_m) = Y_m \wedge e_1 \wedge e_2 \wedge e_{3z}\right\} && (35)\\
& + \Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right) \wedge h_Z(X_m) = Y_m \mid \neg(e_1 \wedge e_2 \wedge e_{3z})\right\} \cdot \Pr(\neg(e_1 \wedge e_2 \wedge e_{3z}))\\
\leq \;& \Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right) \wedge h_Z(X_m) = Y_m \wedge e_1 \wedge e_2 \wedge e_{3z}\right\} + 3\delta && (36)\\
\leq \;& \Pr\{g_{s_m}(Z) = 0 \wedge e_1 \wedge e_2 \wedge e_{3z}\} + 3\delta && (37)\\
\leq \;& \Pr\{g_{s_m}(Z) = 0 \wedge e_1\} + 3\delta\\
\leq \;& \Pr\{g_{s_m}(Z) = 0 \mid e_1\} + 3\delta\\
\leq \;& \frac{\text{polylog}_0(m, 1/\delta)}{m} + 3\delta. && (38)
\end{aligned}
$$

In (34), it is convenient to view the random experiment as if we draw z first, and then S_m. If $Z \in DIS\left(B(f^*, \frac{1}{m})\right)$, then consider h_Z to be any function that holds $h_Z(Z) \neq f^*(Z)$ and $Pr(h_Z(X) \neq f^*(X) \leq \frac{1}{m})$. If $Z \notin DIS\left(B(f^*, \frac{1}{m})\right)$, then the event described in (34) does not occur, and h_Z is undefined. In (35), we use conditional probability, and in (36) we apply the union bound. Inequality (37) is justified as follows. If $h_Z(X_m) = Y_m$, then the algorithm received the same input under $\mathcal{P}_{\mathcal{X},\mathcal{Y}_z}$ and $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$. Given that e_2 and e_{3z} occurred, we know that the algorithm has successfully output a pointwise-competitive selective classifier for both probabilities, which means that whenever f^* and h_Z disagree, g_{s_m} has to output zero; otherwise, it will not be pointwise-competitive for one of the distributions. By the definition of h_Z, $h_Z(Z) \neq f^*(Z)$, which explains the inequality. Inequality (38) follows from the definition of e_1. Taking $\delta = \frac{1}{m}$, we get,

$$
\Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right) \wedge h_Z(X_m) = Y_m\right\} \;\leq\; \frac{\text{polylog}_0(m,m) + 3}{m}. \qquad (39)
$$

The following inequalities are derived using elementary conditional probability. In Equation (41) we use an argument taken from the proof of Hanneke (2012, Lemma 47). $h_Z \in$

$B(f^*, \frac{1}{m})$, and thus the probability that f^* and h_Z will have a different label for a specific example is bounded by $1/m$. The probability that f^* will mispredict is by definition $R(f^*)$, and thus, using the union bound, we get (41).

$$
\begin{aligned}
& \Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right) \wedge h_Z(X_m) = Y_m\right\}\\
= \;& \Pr\left\{h_Z(X_m) = Y_m \mid Z \in DIS\left(B(f^*, \frac{1}{m})\right)\right\} \cdot \Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right)\right\}\\
\geq \;& \Pr\left\{f^*(X_m) = Y_m \wedge h_Z(X_m) = f^*(X_m) \mid Z \in DIS\left(B(f^*, \frac{1}{m})\right)\right\} \cdot && (40)\\
& \cdot \Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right)\right\}\\
\geq \;& \left(1 - R(f^*) - \frac{1}{m}\right)^m \cdot \Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right)\right\} && (41)\\
\geq \;& \left(1 - \frac{1}{m} - \frac{1}{m}\right)^m \cdot \Pr\left\{Z \in DIS\left(B(f^*, \frac{1}{m})\right)\right\}\\
\geq \;& \frac{1}{16} \cdot \Delta B\left(f^*, \frac{1}{m}\right). && (42)
\end{aligned}
$$

Combining (39) and (42), we get that for every $m \in \{4, 5, ..., \lfloor 1/R(f^*) \rfloor\}$,

$$
\frac{\Delta B(f^*, 1/m)}{1/m} \;\leq\; 16\left(\text{polylog}_0(m,m) + 3\right). \qquad (43)
$$

The following inequalities follow from (43), and from the fact that $\Delta B(f^*, x)$ and $\text{polylog}_0(\cdot)$ are non-decreasing. For any r in $[R(f^*), \frac{1}{5}]$, and noting that $\frac{1}{\lfloor 1/r \rfloor} \geq r$,

$$
\begin{aligned}
\frac{\Delta B(f^*, r)}{r} &\leq \frac{\Delta B\left(f^*, \frac{1}{\lfloor 1/r \rfloor}\right)}{\frac{1}{\lfloor 1/r \rfloor}} \cdot \frac{1}{r \cdot \lfloor 1/r \rfloor}\\
&\leq 16(\text{polylog}_0(\lfloor 1/r \rfloor, \lfloor 1/r \rfloor) + 3) \cdot \frac{1}{r \cdot (1/r - 1)}\\
&\leq 16\left(\text{polylog}_0(\lfloor 1/r \rfloor, \lfloor 1/r \rfloor) + 3\right) \cdot \frac{1}{1-r}\\
&\leq 16(\text{polylog}_0(\lfloor 1/r \rfloor, \lfloor 1/r \rfloor) + 3) \cdot \frac{5}{4} && (44)\\
&\leq 20(\text{polylog}_0(1/r, 1/r) + 3)
\end{aligned}
$$

and for r in $[\frac{1}{5}, 1]$,

$$
\frac{\Delta B(f^*, r)}{r} \;\leq\; \frac{1}{1/5} = 5, \qquad (45)
$$

which concludes the proof. ■

Corollary 9 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, and let $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown distribution. If for every $m \leq 1/R(f^*)$, $0 < \delta$, with probability of at least $1-\delta$, the abstain rate $1-\Phi$ of* ILESS$(S_m, \delta, \mathcal{F})$ *is bounded by a monotonic* polylog *as follows:*

$$1 - \Phi(\text{ILESS}) \leq \frac{\text{polylog}_0(m, 1/\delta)}{m}.$$

Then for every f^* (every true risk minimizer), for every $r \geq R(f^*)$,

$$\theta_{f^*}(r) \leq 20(\text{polylog}_0(1/r, 1/r) + 3).$$

Proof This is a direct result from Theorem 8, and from the fact that ILESS is PCS. ■

Given $\mathcal{P}_{\mathcal{X}}$,$\mathcal{F}$ and f^*, if any PCS learning algorithm has a fast R^* rejection rate, we can apply Theorem 8 with a deterministic $\mathcal{P}_{\mathcal{Y}|\mathcal{X}}$ distribution for which $Y = f^*(X)$, and get that $R(f^*) = 0$. Thus, by definition,

$$1 - \Phi(\text{PCS}) \leq \text{polylog}_1\left(\frac{1}{\text{R(f*)} + 1/\text{m}}\right) \cdot 0 + \frac{\text{d} \cdot \text{polylog}_2(\text{m}, 1/\delta)}{\text{m}}.$$

We can now apply Theorem 8 with $R(f^*) = 0$, and get that the disagreement coefficient is bounded by polylog$(1/r)$ for all $r > 0$. Thus, together with Corollary 7, we complete a two sided equivalence from PCS with a fast R^* rejection rate to a disagreement coefficient bounded by polylog$(1/r)$ for all $r > 0$.

6. Active-ILESS

In this section in Strategy 3 we introduce an agnostic active learning algorithm called Active-ILESS. Active-ILESS is very similar to Agnostic CAL and to Oracular CAL (Dasgupta et al., 2007), which were inspired by A^2 (Balcan et al., 2006). Both Agnostic CAL and Oracular CAL use a $LEARN_{\mathcal{H}}(S,T)$ subroutine, which returns an ERM over a certain set T with the constraint that a zero training error must be achieved over a set S (see details in Hsu, 2010). Oracular CAL has an advantage over Agnostic CAL in that it only uses one such constraint while using the $LEARN_{\mathcal{H}}$ subroutine. Nevertheless, as Hsu mentions, this weakens the label complexity analysis of Oracular CAL, and makes it dependent on the square of the disagreement coefficient. Similar to Oracular CAL, Active-ILESS can be implemented with an ERM computation under only one constraint (as seen from the disbelief principle in El-Yaniv and Wiener, 2011, already discussed in Section 4), but its proven label complexity is only dependent linearly on the disagreement coefficient. In this sense, Active-ILESS enjoys the best of both worlds. The label complexity analysis of Active-ILESS appears in Section 8.

As Agnostic CAL and Oracular CAL, Active-ILESS creates artificial labels (step 1), but unlike these other algorithms, Active-ILESS works in batches (inside each batch, the decision whether to query an example is made instantly and not at the end of the batch). This allows Active-ILESS to be a bit more conservative with its deltas. This also allows us to achieve a tighter label complexity.

In Section 7 we use Active-ILESS to show an equivalence between active learning (represented by Active-ILESS) and selective classification (represented by a variant of ILESS,

Strategy 3 Agnostic Low-Error Active Learning Strategy (Active-ILESS)

Input: ϵ and/or m depending on the desired termination condition (error or labeling budget, respectively)
Confidence level δ
Hypothesis class $\mathcal{F}$ with VC dimension d
An unlabeled input sequence sampled i.i.d from $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$: $x_1, x_2, x_3, \ldots$
Output: A classifier $\hat{f}$
Initialize:
Set $\hat{S} = \emptyset$, $G_0 = \mathcal{F}$, $t = 1$
Perform for each example x_t received:
1: if $x_t \in AGR(G_{t-1})$: don't request label for x_t and set $y_t = f(x_t)$ using any $f \in G_{t-1}$
otherwise: request label y_t
2: Set $\hat{S} = \hat{S} \cup \{(x_t, y_t)\}$
3: Set $\hat{f} = \hat{f}(\hat{S})$
4: if $\log_2(t) \in \mathbb{N}$:

- Set $\sigma_{Active} = \hat{\sigma}_{R-\hat{R}}\left(\frac{t}{2}, \frac{\delta}{2t}, d, \hat{R}(\hat{f}, \hat{S})\right) + \bar{\sigma}_{\hat{R}-R}\left(\frac{t}{2}, \frac{\delta}{2t}, d, \hat{R}(\hat{f}, \hat{S}) + \hat{\sigma}_{R-\hat{R}}(\frac{t}{2}, \frac{\delta}{2t}, d, \hat{R}(\hat{f}, \hat{S}))\right)$
- If ϵ was given as input and $\sigma_{Active} < \epsilon$, terminate and return $\hat{f}$
- If m was given as input and $t > m/2$, terminate and return $\hat{f}$
- Set $G_t = \hat{\mathcal{V}}\left(\hat{f}, \sigma_{Active}\right)$
- Set $\hat{S} = \emptyset$

otherwise:

- $G_t = G_{t-1}$

5: Set t = t +1

"Batch-ILESS"). The introduction of these new variants facilitates a straightforward proof of the equivalence relationship. This equivalence implies a novel relationship between selective and active classification in the agnostic setting.

We begin by analyzing Active-ILESS and showing that much like ILESS, $f^* \in G_t$ in each iteration t. The low-error set G, maintained by ILESS, contains all the hypotheses that have an empirical error smaller than $\hat{R}(\hat{f}) + \sigma_{\text{ILESS}}$. In Lemma 1 we showed that this condition implies that f^* resides within the low-error set G of ILESS. A proof that $f^* \in G_t$, after each iteration of Active-ILESS, cannot follow the same argument due to the fact that Active-ILESS, shown in Strategy 3, labels by itself each example whose label is not requested from the teacher. Further, since we consider an agnostic setting, these self-labels can differ from the true labels.

Active-ILESS, as seen in Strategy 3, receives as a termination condition either $\epsilon > 0$ and/or m, and terminates when the radius of its low-error set, G_t, is smaller than ϵ, or when it has processed m examples.

Active-ILESS changes its low-error set, G_t, only for t that are natural powers of 2. For each change, Active-ILESS begins to create fake labels for $x_t \in AGR(G_{t-1})$ that may or may not be equal to the real label of x_t (under the original distribution). In fact, this G_t defines a new distribution, $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)$, and this distribution changes for every t that is a natural power of 2. With respect to a run of Active-ILESS, and $t = 2^i, i \in \mathbb{N}$, we denote by $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)$, the new probability distribution implied by G_t, and the fake labels created by the algorithm. $R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)}(f)$ will be the true risk under the new distribution, while $R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(f)$ is the true risk of f under the original distribution.

Definition 6.1 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, and let $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown distribution. Given a run of Active-ILESS, we denote by $\mathcal{K}$ the event where both inequalities (46) and (47) hold simultaneously for every $f \in \mathcal{F}$, for all iterations of Active-ILESS where $t = 2^i, i \in \mathbb{N}$. $\hat{R}(f) \triangleq \hat{R}(f, \hat{S})$ for $\hat{S}$ before it was initialized:*

$$R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)}(f) \leq \hat{R}(f) + \sigma_{R-\hat{R}}\left(\frac{t}{2}, \frac{\delta}{2t}, d, R(f), \hat{R}(f)\right) \tag{46}$$

$$\hat{R}(f) \leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)}(f) + \sigma_{\hat{R}-R}\left(\frac{t}{2}, \frac{\delta}{2t}, d, R(f), \hat{R}(f)\right) \tag{47}$$

Lemma 10 *$\mathcal{K}$ occurs with probability of at least $1 - \delta$.*

Proof G_t changes only for iterations of the type $2^i, i \in \mathbb{N}$. We know by Lemma 1 that the probability that inequalities (46) and (47) do not hold is smaller than $\delta/(2t)$. By the union bound, the probability that one of these inequalities does not hold after any iteration is smaller than

$$\sum_{t=2^i, i\in\mathbb{N}} \frac{\delta}{2t} \leq \delta.$$

■

Lemma 11 *If f^*, a true risk minimizer under probability distribution $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$, resides within G_t, then it is also a true risk minimizer under probability distribution $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)$.*

Proof

$$\operatorname*{argmin}_{f\in\mathcal{F}} R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)}(f) = \operatorname*{argmin}_{f\in\mathcal{F}} \left(\underbrace{R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(f)}_{A} + \underbrace{R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)}(f) - R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(f)}_{B} \right).$$

We know that f^* minimizes A, and we note that every function that resides within G_t minimizes B, because every difference in the labeling between $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ and $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)$ was done according to the label given by the unanimous decision of functions in G_t. In other words, whenever $x \in AGR(G_t)$, the labelling of x is done according to any function in G_t, for instance, f^*. Thus,

$$\begin{aligned} R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)}(f) - R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(f) &= \mathbb{E}\left[\mathbb{1}[X \in AGR(G_t)](\mathbb{1}[f(X) \neq f^*(X)] - \mathbb{1}[f(X) \neq Y])\right] \\ &= \mathbb{E}\left[\mathbb{1}[X \in AGR(G_t) \wedge f^*(X) \neq Y] \cdot (-1)^{\mathbb{1}(f(X)=f^*(X))}\right] \end{aligned}$$

and we see that f^* minimizes the last term. Hence, f^* minimizes $A + B$. ■

The proofs of the following four lemmas appear in Appendix A. They all show basic good qualities of Active-ILESS.

Lemma 12 *Given that event $\mathcal{K}$ (see Definition 6.1) occurred, each f^* of the **original** distribution $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ resides within G_t for all t. This implies that $R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_t)}(f^*) \leq R(f^*)$, for all t, as every change in the labeling is done according to f^*.*

Lemma 13 *Given that event $\mathcal{K}$ (see Definition 6.1) occurred, and under the assumption that Active-ILESS terminated with the ϵ condition, the hypothesis returned by Active-ILESS, $\hat{f}$, holds:*

$$R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(\hat{f}) \leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(f^*) + \epsilon.$$

Lemma 14 *Given that event $\mathcal{K}$ (see Definition 6.1) occurred, the final radius of Active-ILESS satisfies*

$$\sigma_{Active} = O\left(\frac{B}{m} + \sqrt{\frac{B}{m} \cdot R(f^*)}\right), \tag{48}$$

where $B \triangleq 16d\ln(\frac{16m^2 e}{d\delta})$.

Lemma 15 *Given that event $\mathcal{K}$ (see Definition 6.1) occurred, the total number of examples that Active-ILESS(ϵ) processed (without necessarily requesting labels) is*

$$O\left(\frac{1}{\epsilon}\ln(\frac{1}{\epsilon}) + \frac{R(f^*)}{\epsilon^2}\ln(\frac{R(f^*)}{\epsilon^2})\right),$$

where we hide factors of $d, \ln(1/\delta)$ under the O.

Definition 6.2 *An active learner that generates a hypothesis whose true error is smaller than ϵ w.h.p., has **passive example complexity**, if it observes up to*

$$O\left(\frac{1}{\epsilon}\ln(\frac{1}{\epsilon}) + \frac{R(f^*)}{\epsilon^2}\ln(\frac{R(f^*)}{\epsilon^2})\right)$$

examples (not necessarily labeled).

By Lemmas 13 and 15 we know that Active-ILESS has passive example complexity.

The definition of a fast R^* rejection rate for selective classification induces the following related definition for the exponential speedup of active learning algorithms.

Definition 6.3 (R^* Exponential Speedup) *Given $\mathcal{P}_{\mathcal{X}}$,$\mathcal{F}$ and f^*, we say that an active learner has an R^* **exponential speedup**, with polylog_1 and polylog_2 as its parameters, if for every $\mathcal{P}_{\mathcal{Y}|\mathcal{X}}$ for which f^* is a true risk minimizer, and for every $m > 0$, with probability of at least $1 - \delta$, the number of labels requested by the active learner after observing m examples is not greater than*

$$\text{polylog}_1\left(\frac{1}{R(f^*) + 1/m}\right) \cdot R(f^*)m + d \cdot \text{polylog}_2(m, 1/\delta).$$

Hsu (2010) introduced the agnostic CAL algorithm and showed (Theorem 4.3, page 41) that if the disagreement coefficient is bounded, then Agnostic CAL has an R^* exponential speedup (under our new definition). Any active algorithm that has passive example complexity and achieves an R^* exponential speedup requires w.h.p. no more than $O\left(\text{polylog}(\frac{R(f^*)}{\epsilon^2})\frac{R(f^*)^2}{\epsilon^2} + \text{polylog}(\frac{1}{\epsilon})\right)$ labels to reach a true error smaller than ϵ. The proof is immediate by considering the cases $\frac{R(f^*)}{\epsilon} \geq 1$ and $\frac{R(f^*)}{\epsilon} < 1$. The leading term of this bound is $\frac{R(f^*)^2}{\epsilon^2}$, which is also the case for A^2 (Balcan et al., 2006).

7. A Reduction from Active-iLess to Batch-ILESS

In Strategy 4 we define a selective classifier, called Batch-ILESS, which uses Active-ILESS as its engine. Given a labeled sample S_m, Batch-ILESS simulates the active algorithm, by applying it over S_m in a straightforward manner (i.e., it sequentially introduces to the active algorithm an unlabeled example and reveals the label only if the active algorithm requests it). Upon termination, after the active algorithm has consumed all examples, our batch algorithm receives $\hat{f}$ from the active algorithm and uses its last low-error set G_t to define its selection function.

Lemma 12 implies that Batch-ILESS is pointwise-competitive. We note that Lemma 4, Theorem 5 and Theorem 9, which were proven for ILESS, can also be proven for Batch-ILESS. We chose to prove it for ILESS, as it is simpler than Batch-ILESS, and does not require an active algorithm as its engine. We state these ideas formally, and give sketches of their proofs, in Lemma 23 and Theorem 24 which appear in Appendix C.

Strategy 4 Batch Improved Low-Error Selective Strategy (Batch-ILESS)

Input: Sample set of size m, S_m,
Confidence level δ
Hypothesis class $\mathcal{F}$ with VC dimension d
Output: A selective classifier (h, g)
1: Simulate Active-ILESS with S_m as its input stream; let G_t be the low-error set obtained by Active-ILESS in its last round, and let $\hat{f}$ be its resulting classifier.
2: Construct g such that $g(x) = 1 \iff x \in \{\mathcal{X} \setminus DIS(G_t)\}$
3: $h = \hat{f}$

Theorem 16 shows a deep connection between the speedup of Active-ILESS and the rejection mass of Batch-ILESS for specific $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$. An immediate corollary of this theorem is that if Active-ILESS has R^* exponential speedup (see Definition an 6.3), then Batch-ILESS has a fast R^* rejection rate (see Definition 4.2).

Theorem 16 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, and let $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown distribution. If for every $m \leq m_{max}$, after observing m examples, with probability of at least $1-\delta$, the number of labels requested by Active-ILESS is not greater than*

$$\text{polylog}_1\left(\frac{1}{R(f^*)+1/m}\right)\cdot R(f^*)m + d\cdot\text{polylog}_2(m,1/\delta),$$

then the rejection mass of Batch-ILESS for every $m \leq \frac{m_{max}}{2}$ is bounded w.h.p. by

$$8\cdot\text{polylog}_1\left(\frac{1}{R(f^*)+1/m}\right)\cdot R(f^*) + \frac{2\left(\sqrt{\ln(2/\delta)}+\sqrt{\ln(2/\delta)+2d\cdot\text{polylog}_2(2m,2/\delta)}\right)^2}{m}.$$

Proof Consider an application of Active-ILESS with $\delta = \delta_0$ over $m_0 \triangleq 2^{\lceil\log(m+1)\rceil}$ examples. Denote by X_i an indicator random variable for the labeling of its ith example, $1 \leq i \leq m_0$. With probability of at least $1-\delta_0$ over the choice of samples from $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$,

$$\sum_{i=1}^{m_0} X_i \leq \text{polylog}_1\left(\frac{1}{R(f^*)+1/m_0}\right)\cdot R(f^*)m_0 + d\cdot\text{polylog}_2\left(m_0,1/\delta_0\right). \tag{49}$$

We know by the definition of Active-ILESS (Strategy 3), that the last $m_0/2$ examples had the exact same probability, $\Delta G_{m_0/2}$, of requiring a label, and that this is exactly the probability that Batch-ILESS will decide to abstain after receiving m examples, according to Strategy 4.

We now estimate $\Delta G_{m_0/2}$ using the version of the Chernoff bound given by Canny (2012). For the sake of self-containment, Canny's statement and proof of the bound are provided in Lemma 21 in Appendix B.

The statement of the lemma is as follows. Let $Z_1, Z_2, \ldots, Z_n$ be independent Bernoulli trials with $Pr[Z_i = 1] = p$, let $Z \triangleq \sum_{i=1}^{n} Z_i$, and $\mu = \mathbb{E}Z$. Then, for every $\alpha > 0$:

$$\Pr\left(Z < (1-\alpha)\mu\right) \leq \exp(-\mu\alpha^2/2).$$

Applying the Chernoff bound with the indicator variables of the last $m_0/2$ examples, we have $X = \sum_{m_0/2}^{m_0} X_i$, $\mu = p\frac{m_0}{2}$, and set $p \triangleq \Delta G_{m_0/2}$. Select α such that

$$\exp\left(-p\frac{m_0}{2}\alpha^2/2\right) = \delta_2.$$

Solving for α,

$$\alpha = \sqrt{\frac{4\ln(1/\delta_1)}{m_0 p}}.$$

We conclude that with probability of at least $1-\delta_1$,

$$\begin{aligned} & X \geq \left(1-\sqrt{\frac{4\ln(1/\delta_1)}{m_0 p}}\right)\cdot p\frac{m_0}{2} \\ \Leftrightarrow\quad & 0 \geq \frac{pm_0}{2} - \sqrt{pm_0\cdot\ln(1/\delta_1)} - X. \end{aligned} \tag{50}$$

Solving the quadratic equation (50) for $\sqrt{pm_0}$, we get that

$$\begin{aligned} & \sqrt{pm_0} \leq \sqrt{\ln(1/\delta_1)} + \sqrt{\ln(1/\delta_1)+2X} \\ \Rightarrow\quad & p \leq \frac{(\sqrt{\ln(1/\delta_1)}+\sqrt{\ln(1/\delta_1)+2X})^2}{m_0}. \end{aligned} \tag{51}$$

Combining (49) and (51), from the union bound we get that with probability of at least $1-\delta_0-\delta_1$,

$$\begin{aligned} &\Delta G_{m_0/2} \leq \\ &\leq \frac{\left(\sqrt{\ln(\frac{1}{\delta_1})}+\sqrt{\ln(\frac{1}{\delta_1})+2\text{polylog}_1\left(\frac{1}{R(f^*)+1/m_0}\right)R(f^*)m_0 + 2d\cdot\text{polylog}_2(m_0,\frac{1}{\delta_0})}\right)^2}{m_0}. \end{aligned}$$

If we take $\delta_0 = \delta_1 = \delta/2$, then, since $m \leq m_0 \leq 2m$, we can use $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ and $(a+b)^2 \leq 2a^2 + 2b^2$, to obtain

$$\begin{aligned}
&\Delta G_{m_0/2} \leq \\
\leq\ & \frac{\left(\sqrt{\ln(\frac{2}{\delta})} + \sqrt{\ln(\frac{2}{\delta}) + 4\text{polylog}_1\left(\frac{1}{R(f^*)+1/m}\right)\cdot R(f^*)m + 2d\cdot\text{polylog}_2(2m,\frac{2}{\delta})}\right)^2}{m} \\
\leq\ & \frac{\left(\sqrt{\ln(\frac{2}{\delta})} + \sqrt{\ln(\frac{2}{\delta}) + 2d\cdot\text{polylog}_2(2m,\frac{2}{\delta})} + \sqrt{4\text{polylog}_1\left(\frac{1}{R(f^*)+1/m}\right)R(f^*)m}\right)^2}{m} \\
\leq\ & \frac{2\left(\sqrt{\ln(\frac{2}{\delta})} + \sqrt{\ln(\frac{2}{\delta}) + 2d\cdot\text{polylog}_2(2m,\frac{2}{\delta})}\right)^2}{m} + \\
& + \frac{2\left(\sqrt{4\text{polylog}_1\left(\frac{1}{R(f^*)+1/m}\right)\cdot R(f^*)m}\right)^2}{m} \\
=\ & \frac{2\left(\sqrt{\ln(\frac{2}{\delta})} + \sqrt{\ln(\frac{2}{\delta}) + 2d\cdot\text{polylog}_2(2m,\frac{2}{\delta})}\right)^2}{m} + \\
& + 8\cdot R(f^*)\cdot\text{polylog}_1\left(\frac{1}{R(f^*)+1/m}\right)
\end{aligned}$$

■

Corollary 17 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, and let $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown distribution. If for every $m \leq 2/R(f^*)$, $0 < \delta$ after observing m examples, with probability of at least $1-\delta$, the number of labels requested by Active-ILESS is not greater than*

$$\text{polylog}_1\left(\frac{1}{R(f^*)+1/m}\right)\cdot R(f^*)m + d\cdot\text{polylog}_2(m, 1/\delta),$$

then for every $r \geq R(f^*)$,

$$\begin{aligned}
\theta_{f^*}(r) &\leq 20\left(2\left(\sqrt{\ln(2/r)} + \sqrt{\ln(2/r) + 2d\cdot\text{polylog}_2(2/r, 2/r)}\right)^2 + 8\cdot\text{polylog}_1(1/r) + 3\right) \\
&= O\left(d\cdot\text{polylog}(1/r)\right).
\end{aligned}$$

Proof The proof follows from Theorems 16 and 8. Applying Theorem 16, we know that for $m \leq 1/R(f^*)$, the rejection mass of Batch-ILESS is bounded w.h.p. by,

$$\frac{2\left(\sqrt{\ln(2/\delta)} + \sqrt{\ln(2/\delta) + 2d\cdot\text{polylog}_2(2m, 2/\delta)}\right)^2 + 8\cdot\text{polylog}_1(\frac{1}{R(f^*)+1/m})}{m}.$$

We know by Lemma 12 that $f^* \in G_t$, and thus Batch-ILESS is PCS. We apply Theorem 8, and get that for every $r \geq R(f^*)$,

$$\begin{aligned}
\theta_{f^*}(r) &\leq 20\left(2\left(\sqrt{\ln(\frac{2}{r})} + \sqrt{\ln(\frac{2}{r}) + 2d\cdot\text{polylog}_2(\frac{2}{r},\frac{2}{r})}\right)^2 + 8\cdot\text{polylog}_1\left(\frac{1}{R(f^*)+r}\right) + 3\right) \\
&\leq 20\left(2\left(\sqrt{\ln(\frac{2}{r})} + \sqrt{\ln(\frac{2}{r}) + 2d\cdot\text{polylog}_2(\frac{2}{r},\frac{2}{r})}\right)^2 + 8\cdot\text{polylog}_1(1/r) + 3\right).
\end{aligned}$$

■

8. From the Disagreement Coefficient to Active Learning

In this section we show that when $\theta'(r)$ is bounded by $\text{polylog}_1(1/r)$ for all $r > R(f^*)$ for some specific $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$, then the label complexity of Active-ILESS under the same $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ is bounded by

$$\text{polylog}_2\left(\frac{1}{R(f^*)+1/m}\right)\cdot R(f^*)m + d\cdot\text{polylog}_3(m, 1/\delta), \tag{52}$$

where the parameters of polylog_2 and polylog_3 are only dependent on $\text{polylog}_1(1/r)$. Thus, if $\theta'(r) \leq \text{polylog}_1(1/r)$ for all $r > 0$, we get that Active-ILESS has an R^* exponential speedup. This direction has been shown before in Hsu (2010); Hanneke (2007) for Oracular CAL, Agnostic CAL and A^2. For the sake of self-containment, we show it here for Active-ILESS. Due to the fact that Active-ILESS relies on ILESS, for which we already have bounds, the proof is straightforward.

As a preparation for Theorem 19, we present Lemma 18 (shown before in Hanneke, 2014b, Theorem 7.1), which introduces a small feature of the disagreement coefficient that will serve us later.

Lemma 18 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, and let $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown distribution. For every $f \in \mathcal{F}$ and $0 < r \leq 1$, $\theta_f(r)\cdot r$ is a non-decreasing function.*

Theorem 19 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown distribution, and f^*, a true risk minimizer of $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$. The label complexity of Active-ILESS$(m, \delta/2)$ is bounded w.h.p. by*

$$\theta'\left(5R(f^*) + 14\frac{A}{m}\right)\cdot 2e\cdot mR(f^*) + \log_2(2/\delta) + 56e\cdot\log_2 m\cdot A\cdot\theta'\left(5R(f^*) + 14\frac{A}{m}\right).$$

Proof Each run of Active-ILESS$(m, \delta/2)$ simulates $\lceil\log_2 m\rceil$ runs of ILESS. Denote by T indices of iterations where $\log_2(t) \in \mathbb{N}$. We know by Lemma 10 that with probability of at least $1 - \delta/2$, inequalities (46) and (47) hold for each run. Recall that we denoted by $\mathcal{K}$ the event where both inequalities hold throughout all runs of ILESS, which is exactly the definition of event $\mathcal{E}$ per run (see Definition 4.1). Under event $\mathcal{K}$, Lemma 12 implies that

all f^* of the original distribution $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ reside within G_T for all T. This also implies that all f^* of the original distribution remain the true risk minimizers under $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_T)$, for all T, as they always benefit from the creation of the artificial labels.

Because the marginal of the distribution does not change during the run of Active-ILESS, and because event $\mathcal{E}$ holds for each iteration of ILESS, we can apply Theorem 5 for all the runs of ILESS. We thus get that for every run of ILESS, which we denote by ILESS_T,

$$G_T \subseteq B(f^*, R_T),$$

and thus, the rejection mass is bounded as follows

$$1 - \Phi(\text{ILESS}_T) \leq \Delta B(f^*, R_T) \leq \theta(R_T) \cdot R_T,$$

where

$$R_T \triangleq 2 \cdot R(f^*) + 11 \cdot \frac{A}{T} + 6 \cdot \sqrt{\frac{A}{T} \cdot R(f^*)}.$$

We denoted by $R(f^*)$ the true error according to the original distribution, which might be larger than the true error implied by the fake label distributions that the algorithm induces. According to Lemma 18, enlarging R_T can only weaken the bound, and thus, there is no problem doing so. We additionally bound R_T using $\sqrt{ab} \leq a/2 + b/2$ to get

$$R_T \leq 5 \cdot R(f^*) + 14 \cdot \frac{A}{T}.$$

and thus,

$$\Delta B(f^*, R_T) \leq \theta' \left(5 \cdot R(f^*) + 14 \cdot \frac{A}{T}\right) \cdot \left(5 \cdot R(f^*) + 14 \cdot \frac{A}{T}\right).$$

In this proof we bound the sum $\sum_{t=1}^{m} X_t$, where $X_t \triangleq \mathbb{1}[x_t \in \Delta G_t]$, as ΔG_t is by definition the probability that Active-ILESS requests a label. To do so, we will bound the sum $\sum_{t=1}^{m} P_t$ where $P_t \triangleq \mathbb{1}[x_t \in \Delta B(f^*, R_t)]$, as we know that when $\mathcal{K}$ holds, $\Delta G_t \subseteq \Delta B(f^*, R_t)$.

According to the definition of G_t in Strategy 3, the probability distribution of the artificial labeling done by Active-ILESS changes only when t is a natural power of 2. Thus, R_t only changes when $\log_2(t) \in \mathbb{N}$, and we get that

$$P_t \leq \theta' \left(5 \cdot R(f^*) + 14 \cdot \frac{A}{T}\right) \cdot 5R(f^*) + \frac{14A \cdot \theta' \left(5 \cdot R(f^*) + 14 \cdot \frac{A}{T}\right)}{T}, \tag{53}$$

where $T = 2^{\lfloor \log_2(t-1) \rfloor - 1}$.

We now have a series of Poisson trials, $P_1, P_2, \ldots, P_m$. We use a version of the Chernoff bound (as shown by Canny, 2012) to bound the label complexity.[6] The statement and a sketch of the proof of this bound are provided in Lemma 22 in Appendix B.

For independent Poisson variables $P_1, P_2, \ldots, P_m$, where $Pr[P_i = 1] = p_i$, $P \triangleq \sum_{i=1}^{n} P_i$, and $\mu = \mathbb{E}P$, for every $\alpha > 2e - 1$:

$$\Pr(P > (1+\alpha)\mu) \leq 2^{-\mu\alpha}.$$

6. This useful bound was used by Hanneke (2014a, Theorem 5.4).

To bound $\mu = \mathbb{E}P$ from above, we use Inequality (53) and plug it into the definition of μ.

$$\begin{aligned}
\mu &= P_1 + P_2 + \sum_{i=3}^{m} P_t \\
&\leq 2 + \sum_{k=1}^{\lceil \log_2 m \rceil - 1} 2^k P_{2^{k+1}} \\
&\leq 2 + m \cdot \theta' \left(5R(f^*) + 14\frac{A}{m}\right) \cdot R(f^*) + \sum_{k=1}^{\lceil \log_2 m \rceil - 1} 2^k \frac{14A \cdot \theta' \left(5R(f^*) + 14\frac{A}{m}\right)}{2^{k-1}} \\
&\leq 2 + m \cdot \theta' \left(5R(f^*) + 14\frac{A}{m}\right) \cdot R(f^*) + 28 \log_2 m \cdot A \cdot \theta' \left(5R(f^*) + 14\frac{A}{m}\right).
\end{aligned} \tag{54}$$

We need to choose an α that satisfies both $2^{-\mu\alpha} \leq \delta/2$, and $\alpha > 2e - 1$. Clearly, $\alpha = \frac{\log_2(2/\delta)}{\mu} + 2e - 1$ suffices. Hence, we get that with probability of at least $1 - \delta/2$,

$$\begin{aligned}
P &\leq \left(1 + \frac{\log_2(\frac{2}{\delta})}{\mu} + 2e - 1\right)\mu \\
&= \log_2(\frac{2}{\delta}) + 2e\mu.
\end{aligned}$$

Inequality (54) holds with probability of at least $1 - \delta/2$, and using the union bound, we get that with probability of at least $1 - \delta$,

$$\begin{aligned}
P &\leq \log_2(\frac{2}{\delta}) + \\
&\quad +2e\left(2 + m \cdot \theta' \left(5R(f^*) + 14\frac{A}{m}\right) \cdot R(f^*) + 28 \log_2 m \cdot A \cdot \theta' \left(5R(f^*) + 14\frac{A}{m}\right)\right) \\
&= \theta' \left(5R(f^*) + 14\frac{A}{m}\right) 2e \cdot mR(f^*) + \log_2(\frac{2}{\delta}) + 56e \cdot \log_2 m \cdot A \cdot \theta' \left(5R(f^*) + 14\frac{A}{m}\right)
\end{aligned} \tag{55}$$

■

Corollary 20 *Let $\mathcal{F}$ be a hypothesis class with a finite VC dimension d, $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown distribution, and f^* a true risk minimizer of $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$. If for all $r > R(f^*)$,*

$$\theta'(r) \leq \text{polylog}_1(1/r),$$

then the label complexity of Active-ILESS$(m, \delta/2)$ is bounded w.h.p. by

$$\text{polylog}_1 \left(\frac{1}{5R(f^*) + 14\frac{A}{m}}\right) 2e \cdot mR(f^*) + \log_2(2/\delta) + 56e \cdot \log_2 m \cdot A \cdot \text{polylog}_1 \left(\frac{1}{5R(f^*) + 14\frac{A}{m}}\right),$$

which has the same form as Equation (52).

Hsu (2010) calculated the label complexity of Agnostic CAL and Oracular CAL under the assumption that the disagreement coefficient is bounded by a constant, θ for all $r > 0$. If we adopt this assumption and use it in our analysis, we get that the label complexity is bounded by

$$O\left(\theta m R(f^*) + \ln(1/\delta) + \theta d \ln m \ln\left(\frac{m}{d\delta}\right)\right).$$

This result is nearly identical (up to some logarithmic factors) to the result shown by Hsu (2010, Theorem 4.3) for Agnostic CAL. Agnostic CAL, however, as mentioned before, requires calculation of ERM under several constraints. Oracular CAL only requires one constrained ERM calculation, and its label complexity depends on $\theta^2 \ln^3 m$ (Hsu, 2010, Theorem 5.2). Our algorithm thus enjoys the good properties of both algorithms.

The dominant factor of Equation (55), if we ignore the logarithmic factors, is $mR(f^*)$. Active-ILESS has passive example complexity (see Definition 6.2), which means that the total sample complexity is bounded by $\tilde{O}(\frac{1}{\epsilon} + \frac{R(f^*)}{\epsilon^2})$, where $\tilde{O}(\cdot)$ hides logarithmic factors. Plugging the sample complexity into m in (55), we get that the total label complexity is bounded by $\tilde{O}(\frac{R(f^*)^2}{\epsilon^2})$, in cases for which ILESS has a fast R^* rejection rate. Kääriäinen (2006, Theorem 3) showed that for every AL algorithm, under a specific (non-trivial) hypothesis class $\mathcal{F}$, there exists a deterministic target function g, and a marginal distribution $\mathcal{P}_\mathcal{X}$, s.t. the label complexity is $\tilde{\Omega}(\frac{R(f^*)^2}{\epsilon^2})$ (where $\tilde{\Omega}(\cdot)$ hides logarithmic factors).

9. Concluding Remarks

In this paper we focused on disagreement-based methods. Namely, we always required that f^* remains inside a low-error subset of hypotheses w.h.p., and made decisions based on disagreement considerations. We have always chosen to abstain (in selective classification) or require a label (in AL) whenever there was no consensus in the low error set for a given example. However, this approach is not necessary when you have an epsilon budget for error. Zhang and Chaudhuri (2014) presented an AL algorithm that uses a confidence-rated predictor to decide which labels to query. This predictor (see top of page 6 in their paper) uses LP to minimize the query probability under a constraint on the total error. They also provided a new complexity measure, which is nicely presented in Hanneke (2016), and termed φ_c. Up to certain constants, φ_c replaces the disagreement coefficient in the error bounds of Equation (16) and Theorem 16 in that paper. φ_c is a potential improvement because it is shown that $\varphi_c(r) \leq \theta_f(r)$. Zhang and Chaudhuri show an example where this improvement can be as large as $\sqrt{d}$ for linear classification under the log-concave distribution.

In our paper we focused only on consensus based decisions. This approach did not affect our results significantly as our main concern was the dependency on $1/\epsilon$ (logarithmic or linear). Consensus was an important element in the proof of Theorem 8. Pointwise-competitive selective classifiers are based on consensus decisions, and thus the LP idea is not an appropriate AL equivalent for PCS. Nevertheless, our reduction from PCS to $\theta_{f^*}(r)$ (Theorem 8), is immediately translated to $\varphi_c(r)$, as $\varphi_c(r) \leq \theta_{f^*}(r)$. It is interesting, however, to define a selective classifier that is not PCS, has an error budget, and apply the LP technique. We leave this for future work.

We introduced a new selective classification algorithm, called ILESS, whose rejection "engine" uses sharp generalization bounds (which depend on $R(f^*)$). Our analysis proves

that ILESS has sometimes significantly better rejection guarantees relative to the best known pointwise-competitive selective strategy of Wiener and El-Yaniv (2015). Moreover, the guarantees we provide for ILESS do not depend at all on the Bernstein assumption. For the general agnostic setting, we showed an equivalence relation between pointwise-competitive selective classification, active learning with Active-ILESS, and the disagreement coefficient (see Figure 1). This equivalence is formulated in terms of a fast R^* rejection rate and an R^* exponential speedup (Definitions 4.2 and 6.3).

Theorems 8 and 5 show that selective classification with a fast R^* rejection rate is completely equivalent to having a disagreement coefficient bounded by polylog$(1/r)$ for $r > 0$. In Section 6, in Strategy 3, we define Active-ILESS using ILESS implicitly as its engine (see Statement 4 in Strategy 3). We can replace ILESS with another pointwise-competitive selective algorithm, and thus construct a new active learner, that queries a label whenever the selective classifier abstains, and create a fake label according to the decision of the classifier whenever it decides to predict. Because the selective predictor is pointwise-competitive, we know that the underlying distribution induced by its fake labels is equivalent to a distribution defined by a deterministic labeling according to f^* and the same $\mathcal{P}_\mathcal{X}$. The algorithm will terminate using the exact same termination condition as Active-ILESS (when $\sigma_{Active} < \epsilon$), and thus the total sample complexity (labeled and unlabeled examples) will remain the same. The change will only be in the labeling criterion. Lemmas 11, 12, 13, 14, and 15 can all be generalized to such an algorithm.

Going in the other direction to create a selective classifier from a general active learner is more challenging. If the active learner, however, follows the Active-ILESS paradigm, and in particular, uses a pointwise-competitive selective classifier to decide on label requests, then a new pointwise-competitive selective classifier can be created in the same way that Batch-ILESS was created. We can then restate Theorem 16, providing a reduction from an R^* exponential speedup of the active algorithm to a fast R^* rejection rate of the selective classifier.

Disagreement-based decision making in active and selective learning leads to "defensive" algorithms. For example, in the active learning case, this means that a defensive algorithm will ask for more labels than a more aggressive algorithm. In selective classification, this defensiveness provides the power to be pointwise-competitive, but at a cost of an increased rejection rate. It would be interesting to consider more aggressive algorithms that could, for example, take into consideration an estimation of $\mathcal{P}_\mathcal{X}$ in order to ignore examples that cause disagreement only between functions that are very similar to each other (in terms of the probability mass of their difference). Such algorithms can be seen in Dasgupta (2005); Freund et al. (1997); Gonen et al. (2013); Zhang and Chaudhuri (2014), for the realizable and the low-error scenarios. We believe that there is still work to be done for the agnostic scenario.

Many aggressive algorithms could be devised under assumptions about knowledge of $\mathcal{P}_\mathcal{X}$ (that could be acquired during the algorithm run, and is given in the transductive case), or in a Bayesian setting where a prior distribution on $\mathcal{F}$ exists. When researching this direction, one might also want to define a cost over unlabeled examples, and discuss the trade-off between labeled and unlabeled examples. The main open question inspired by our results would be to identify similar correspondence between aggressive selective classification algorithms and aggressive active learners.

Another aspect of selective classification and active learning, which was not addressed in this paper, is differentiating between more and less noisy areas of the distribution. If we define the noise to be the error of the Bayes classifier, and we assume that the noise behaves similarly in close areas (for instance, we assume that the derivative of the noise is bounded), we can derive an active learner that will estimate the noise, and take it into consideration. A noisy area could be defined as an area for which even the best classifier in the class could not achieve a low-error. This motivates a new type of labeling for selective prediction, where one can abstain for two reasons: (i) lack of knowledge in a specific region of $\mathcal{X}$, i.e., not enough examples were observed in that region, and the generalization bounds are not sufficiently tight. (ii) The region was well explored, but even the best classifier performs poorly, and thus the answer is unknown (the region is noisy). In our paper, an active learner will query for both scenarios; however, a more clever active learner might only query examples of the first type, as examples of the second type cannot reduce its error.

Consider the following experiment in Active Learning; Suppose we are trying to learn a hyperspace hypothesis over R^2, where we have a Gaussian distribution of positive points located at $(-1,0)$, and another Gaussian distribution of negative points located at $(1,0)$. A simple active learner that is based on distance from the margin, will quickly converge to querying points that have $x \sim 0$, but will have no preferation over the value of y. Now consider another Gaussian distribution of noisy points, located at $(0,1)$, where each point is -1 or 1 with equal probability. Querying in areas where $y \sim 1$ is almost completely unbeneficial, however, this is still very close to the margin. We devised a simple active learner that creates a noise hit map over R^2, and consider the noise as part of its heuristics. Thus, achieving faster convergence over the usual margin based algorithm. Of course, this is only a toy example, and when running this experiment over some few real world data sets, we got no improvement. We believe that some more experimental work in this area, might prove to be beneficial.

There has been a lot of progress on the theoretical level, in the field of active learning with noise estimation (Locatelli et al., 2017; Hanneke, 2017; Minsker, 2012; Hanneke and Yang, 2015) However most of the work requires either the Tsybakov noise condition (Tsybakov, 2004), a finite VC dimension, or some other strong smoothing assumptions. It would be interesting to see some experimental work on real data sets, accompanied with some theoretical work with a simplistic noise locality assumption.

Acknowledgments

We are indebted to an anonymous referee for extremely useful comments and for fixing an error in an earlier version of this manuscript. This research was supported by The Israel Science Foundation (grant No. 1890/14).

Appendix A. Proofs for Lemmas in Section 6

Proof of Lemma 12 We prove the claim by induction over t for which G_t is different from G_{t-1}. The base case of the induction is clear. We now show that functions that are true risk minimizers of $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})$ reside within G_t. According to Lemma 11, f^* is a true risk minimizer under $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})$ (given the induction hypothesis), and hence will also be within

G_t. We refer by f^* to a true risk minimizer according to $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})$. Using Inequality (47) and the definition of $\bar{\sigma}_{\hat{R}-R}$,

$$\begin{aligned}\hat{R}(f^*,\hat{S}) &\leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*) + \sigma_{\hat{R}-R}\left(\frac{t}{2},\frac{\delta}{2t},d,R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*),\hat{R}(f^*,\hat{S})\right)\\ &\leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*) + \bar{\sigma}_{\hat{R}-R}\left(\frac{t}{2},\frac{\delta}{2t},d,R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*)\right),\end{aligned} \tag{56}$$

and by Inequality (46) and the definition of $\hat{f}$ we get,

$$\begin{aligned}R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*) &\leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(\hat{f})\\ &\leq \hat{R}(\hat{f},\hat{S}) + \sigma_{R-\hat{R}}\left(\frac{t}{2},\frac{\delta}{2t},d,R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(\hat{f}),\hat{R}(\hat{f},\hat{S})\right)\\ &\leq \hat{R}(\hat{f},\hat{S}) + \hat{\sigma}_{R-\hat{R}}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S})\right).\end{aligned} \tag{57}$$

Plugging (57) into (56) we get,

$$\begin{aligned}\hat{R}(f^*,\hat{S}) \leq\ & \hat{R}(\hat{f},\hat{S}) + \hat{\sigma}_{R-\hat{R}}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S})\right)\\ &+\bar{\sigma}_{\hat{R}-R}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S}) + \hat{\sigma}_{R-\hat{R}}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S})\right)\right)\\ &\Rightarrow f^* \in G_t.\end{aligned}$$

■

Proof of Lemma 13 Let G_{t-1} be the final low-error set of Active-ILESS, and $\hat{S}$ be the final set of examples. The following inequalities are derived from Lemma 10 and inequalities

(56) and (57).

$$\begin{aligned}
R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(\hat{f}) &\leq \hat{R}(\hat{f},\hat{S}) + \hat{\sigma}_{R-\hat{R}}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S})\right)\\
&\leq \hat{R}(f^*,\hat{S}) + \hat{\sigma}_{R-\hat{R}}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S})\right)\\
&\leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*) + \bar{\sigma}_{\hat{R}-R}\left(\frac{t}{2},\frac{\delta}{2t},d,R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*)\right)\\
&\quad +\hat{\sigma}_{R-\hat{R}}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S})\right)\\
&\leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*) + \bar{\sigma}_{\hat{R}-R}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S}) + \hat{\sigma}_{R-\hat{R}}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S})\right)\right)\\
&\quad +\hat{\sigma}_{R-\hat{R}}\left(\frac{t}{2},\frac{\delta}{2t},d,\hat{R}(\hat{f},\hat{S})\right)\\
&\leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*) + \epsilon.
\end{aligned}$$

By Lemma 12 we know that f^* resides within G_{t-1}, which implies that any change in $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})$ in comparison to $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ reduces the true error of f^*. This also means that for every $f \in \mathcal{F}$,

$$R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(f) - R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f) \leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(f^*) - R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*),$$

which results in

$$R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(\hat{f}) \leq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}}(f^*) + \epsilon.$$

■

Proof of Lemma 14 The proof is similar to the proof of Lemma 4. We consider the last modification of G_t as a run of ILESS, under $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})$, with $m_0 \triangleq 2^{\lfloor log_2 m \rfloor - 1}$ examples and delta equal to $\frac{\delta}{4m_0}$.

Under event $\mathcal{K}$, the conditions of Lemma 4 hold, and by Lemma 12, $R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*) \leq R(f^*)$. We simply apply Lemma 4 with these parameters to get A' (A in Lemma 4).

$$A' = 4d\ln\left(\frac{16m_0 e}{d\delta/4m_0}\right) = 4d\ln\left(\frac{64m_0^2 e}{d\delta}\right).$$

The fact that $m/4 \leq m_0 \leq m/2$ completes the proof. ■

Proof of Lemma 15 We know by Lemma 14 that there exist constants C_1,C_2 that depend only on $\ln(\frac{1}{\delta})$ and d, and are independent of m, s.t.

$$\sigma_{Active} \leq C_1\frac{\ln m}{m} + C_2\sqrt{\frac{\ln m}{m}\cdot R(f^*)}.$$

We also know by the definition of Active-ILESS (Strategy 3), that it terminates when σ_{Active} is smaller than the given ϵ. We will find m large enough s.t.

$$C_1\frac{\ln m}{m} \leq \epsilon/2, \tag{58}$$

$$C_2\sqrt{\frac{\ln m}{m}\cdot R(f^*)} \leq \epsilon/2. \tag{59}$$

We assume that $\epsilon \leq 1/e$, as it is easy to find a proper m for $\epsilon > 1/e$. Starting with Equation (58), we want to show that $m = O(\frac{1}{\epsilon}\ln(\frac{1}{\epsilon}))$ satisfies it. Thus, we find k_1 s.t.

$$\begin{aligned}
& C_1\frac{\ln(k_1\frac{1}{\epsilon}\ln\left(\frac{1}{\epsilon}\right))}{k_1\frac{1}{\epsilon}\ln\left(\frac{1}{\epsilon}\right)} \leq \frac{\epsilon}{2}\\
\Leftrightarrow\quad & \frac{\ln\left(k_1\frac{1}{\epsilon}\cdot\ln(\frac{1}{\epsilon})\right)}{\ln\left(\frac{1}{\epsilon}\right)} \leq \frac{k_1}{2C_1}.
\end{aligned} \tag{60}$$

Bounding the left-hand side of Inequality (60) for $\epsilon \leq 1/e$ gives us,

$$\begin{aligned}
\frac{\ln\left(k_1\frac{1}{\epsilon}\cdot\ln(\frac{1}{\epsilon})\right)}{\ln(\frac{1}{\epsilon})} &\leq \frac{\ln\left(k_1\frac{1}{\epsilon}\cdot\frac{1}{\epsilon}\right)}{\ln\left(\frac{1}{\epsilon}\right)}\\
&\leq 2+\ln k_1.
\end{aligned}$$

We need to find k_1 that will satisfy

$$2+\ln k_1 \leq \frac{k_1}{2C_1}.$$

$k_1 = 16C_1^2$ will work for $C_1 \geq 1$; otherwise, we take $k_1 = 10$.

We use the same procedure to show that $m = O\left(\frac{R(f^*)}{\epsilon^2}\ln\left(\frac{R(f^*)}{\epsilon^2}\right)\right)$ satisfies Equation (59). We rewrite the equation in the following way:

$$\frac{\ln m}{m} \leq \frac{\epsilon^2}{4C_2^2 R(f^*)} \triangleq \epsilon_0.$$

We assume that $\epsilon_0 \leq 1/e$ ($m = 4$ holds otherwise) and find k_2 s.t.

$$\frac{\ln\left(k_2\frac{1}{\epsilon_0}\ln\left(\frac{1}{\epsilon_0}\right)\right)}{k_2\frac{1}{\epsilon_0}\ln\left(\frac{1}{\epsilon_0}\right)} \leq \epsilon_0.$$

As before, we reduce the problem to finding k_2 that satisfies

$$2+\ln(k_2) \leq k_2.$$

$k_2 = 4$ suffices. We thus get that $m = O\left(\frac{1}{\epsilon_0^2}\ln\left(\frac{1}{\epsilon_0^2}\right)\right) = O\left(\frac{R(f^*)}{\epsilon^2}\ln\left(\frac{R(f^*)}{\epsilon^2}\right)\right)$ satisfies Equation (59). This implies that there exists a function

$$m(1/\epsilon, R(f^*)) = O\left(\frac{1}{\epsilon}\ln\left(\frac{1}{\epsilon}\right) + \frac{R(f^*)}{\epsilon^2}\ln\left(\frac{R(f^*)}{\epsilon^2}\right)\right)$$

that bounds the total number of labels processed by Active-ILESS. ■

Appendix B. Proofs of Chernoff Bounds

Lemma 21 *Let $X_1, X_2, ..., X_n$ be independent Bernoulli trials with $Pr[X_i = 1] = p$, $X \triangleq \sum_{i=1}^{n} X_i$, and $\mu = \mathbb{E}X$. Then, for every $\alpha \geq 0$:*

$$\Pr(X < (1-\alpha)\mu) \leq \exp(-\mu\alpha^2/2).$$

Proof This proof is taken from the work of John Canny Canny (2012).

For $t > 0$, we have

$$\Pr\left(X < (1-\alpha)\mu\right) = \Pr\left(\exp(-tX) > \exp(-t(1-\alpha)\mu)\right). \tag{61}$$

We use Markov's inequality. For a nonnegative random variable X, and $a > 0$,

$$\Pr(X \geq a) \leq \frac{\mathbb{E}(X)}{a}.$$

We apply the inequality for the right-hand side of Equation (61), to get

$$\Pr(X < (1-\alpha)\mu) \leq \frac{\mathbb{E}(\exp(-tX))}{\exp(-t(1-\alpha)\mu)}. \tag{62}$$

$X_1, X_2, ..., X_n$ are independent and thus

$$\mathbb{E}(\exp(-tX)) = \prod_{i=1}^{n} \mathbb{E}(\exp(-tX_i)).$$

For each X_i

$$\mathbb{E}(\exp(-tX_i)) = pe^{-t} + (1-p) = 1 - p(1-e^{-t}).$$

We use the fact that $1 - x < \exp(-x)$ for all x, with $x = p(1-e^{-t})$, to get

$$\mathbb{E}(\exp(-tX_i)) \leq \exp(-p(1-e^{-t})),$$

and conclude that

$$\begin{aligned}\mathbb{E}(\exp(-tX)) &= \prod_{i=1}^{n} \mathbb{E}\left(\exp(-tX_i)\right) \leq \prod_{i=1}^{n} \exp\left(-p(1-e^{-t})\right) \\ &= \exp\left(\sum_{i=1}^{n} p(e^{-t}-1)\right) = \exp\left(\mu(e^{-t}-1)\right).\end{aligned} \tag{63}$$

Going back to Equation (62), we have,

$$\Pr(X < (1-\alpha)\mu) \leq \frac{\exp\left(\mu(e^{-t}-1)\right)}{\exp\left(-t(1-\alpha)\mu\right)} = \exp\left(\mu(e^{-t}-1+t-t\alpha)\right). \tag{64}$$

We choose $t > 0$ to make the right-hand side of the equation as small as possible. After derivation, we get that the best t is $t = \ln\left(\frac{1}{1-\alpha}\right)$, and plugging it into Equation (64) gives

us,

$$\begin{aligned}\Pr(X < (1-\alpha)\mu) &\leq \exp\left(\mu(1-\alpha-1+\ln\left(\frac{1}{1-\alpha}\right) - \ln\left(\frac{1}{1-\alpha}\right)\alpha)\right) \\ &= \exp\left(\mu(-\alpha+\ln\left(\frac{1}{1-\alpha}\right)(1-\alpha))\right) \\ &= \left(\frac{e^{-\alpha}}{(1-\alpha)^{1-\alpha}}\right)^{\mu}.\end{aligned} \tag{65}$$

We now simplify this bound to get the desired result. We know that $(1-\alpha)^{1-\alpha} = e^{(1-\alpha)ln(1-\alpha)}$, and by Taylor expansion

$$ln(1-\alpha) = -\alpha - \frac{\alpha^2}{2} - \frac{\alpha^3}{3}...,$$

which multiplied by $(1-\alpha)$, gives us

$$(1-\alpha)ln(1-\alpha) = -\alpha + \frac{\alpha^2}{2} + \text{positive terms} > -\alpha + \frac{\alpha^2}{2}. \tag{66}$$

Plugging (66) into Equation (65), we finally get,

$$\begin{aligned}\Pr(X < (1-\alpha)\mu) &\leq \left(\frac{e^{-\alpha}}{(1-\alpha)^{1-\alpha}}\right)^{\mu} \\ &= \left(\frac{e^{-\alpha}}{e^{(1-\alpha)ln(1-\alpha)}}\right)^{\mu} \\ &\leq \left(\frac{e^{-\alpha}}{e^{-\alpha+\frac{\alpha^2}{2}}}\right)^{\mu} \\ &= e^{-\mu\alpha^2/2}\end{aligned}$$

■

Lemma 22 *Let $X_1, X_2, ..., X_n$ be independent Poisson trials with $Pr[X_i = 1] = p_i$, $X \triangleq \sum_{i=1}^{n} X_i$, and $\mu = \mathbb{E}X$. Then, for every $\alpha \geq 2e - 1$:*

$$\Pr(X > (1+\alpha)\mu) \leq 2^{-\mu\alpha}.$$

Proof Sketch This sketch is taken from the work of John Canny Canny (2012). It is almost identical to the proof of Lemma 21.

We start by showing that

$$\Pr(X > (1+\alpha)\mu) \leq \left(\frac{e^{\alpha}}{(1+\alpha)^{1+\alpha}}\right)^{\mu}.$$

For every $t > 0$,

$$\Pr(X > (1+\alpha)\mu) \quad = \quad \Pr[\exp(tX) > \exp\left(t(1+\alpha)\mu\right)].$$

As we did in Lemma 21, we compute the Markov bound,

$$\Pr(X > (1+\alpha)\mu) \quad \leq \quad \frac{\mathbb{E}(\exp(tX))}{\exp(t(1+\alpha)\mu)},$$

and use the fact that X_i are independent, just like in (63), to get that

$$\mathbb{E}(\exp(tX)) \leq \exp\left(\mu(e^t - 1)\right).$$

Thus we get that

$$\Pr(X > (1+\alpha)\mu) \quad \leq \quad \frac{\exp(\mu(e^t-1))}{\exp(t(1+\alpha)\mu)} = \exp\left(\mu(e^t - 1 - t - \alpha t)\right).$$

From deviation, we choose $t = \ln(1+\alpha)$ to get

$$\Pr(X > (1+\alpha)\mu) \quad \leq \quad \left(\frac{e^\alpha}{(1+\alpha)^{1+\alpha}}\right)^\mu.$$

For $\alpha \geq 2e - 1$:

$$\Pr(X > (1+\alpha)\mu) \quad \leq \quad \left(\frac{e^\alpha}{(1+\alpha)^{1+\alpha}}\right)^\mu \leq \left(\frac{e^\alpha}{(2e)^{1+\alpha}}\right)^\mu \leq \left(\frac{e^\alpha}{(2e)^\alpha}\right)^\mu = 2^{-\mu\alpha}.$$

■

Appendix C. The Rejection Rate of Batch-ILESS

Lemma 23 *Given that event $\mathcal{K}$ (see Definition 6.1) occurred, the radius of Batch-ILESS, as defined in Strategy 3, Stage 4, satisfies*

$$\sigma_{Active} = O\left(\frac{B}{m} + \sqrt{\frac{B}{m} \cdot R(f^*)}\right),$$

where $B \triangleq 4d\ln\left(\frac{8m^2e}{d\delta}\right)$.

Proof The proof is very similar to the proof of Lemma 14. ■

Theorem 24 *Let $\mathcal{F}$ be a hypothesis class with VC-dimension d, and $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ be an unknown probability distribution. Assume that event $\mathcal{K}$ (see Definition 6.1) occurred. Then, for all f^*, the abstain rate is bounded by*

$$1 - \Phi(\textit{Batch-ILESS}) \leq \theta_{f^*}(R_0) \cdot R_0,$$

where

$$R_0 \triangleq 2 \cdot R(f^*) + 44 \cdot \frac{B}{m} + 12 \cdot \sqrt{\frac{B}{m} \cdot R(f^*)}.$$

where $B \triangleq 4d\ln\left(\frac{8m^2e}{d\delta}\right)$. This immediately implies (by definition) that

$$1 - \Phi(\textit{Batch-ILESS}) \leq \theta(R_0) \cdot R_0.$$

Proof Sketch The proof is very similar to the proof of Lemma 23. We observe the last modification of G_T, and notice that the change was made according to a run of ILESS, on the implied probability distribution $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{T-1})$. Then we simply activate Theorem 5 with the relevant parameters plugged into it.

Note that by Lemma 12, all f^* of the original distribution reside within G_t for all t, and thus, by Lemma 11, they are all true risk minimizers of $\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{T-1})$. This also implies that $R(f^*) \geq R_{\mathcal{P}_{\mathcal{X},\mathcal{Y}}(G_{t-1})}(f^*)$ and thus can be used to bound Equation (30) of the original theorem that was proven for ILESS. θ_f is independent of $\mathcal{P}_{\mathcal{Y}|\mathcal{X}}$ for all f, and thus the change of the labels does not affect it.

■

References

N. Ailon, R. Begleiter, and E. Ezra. Active learning using smooth relative regret approximations with applications. In 25th *Annual Conference on Learning Theory (COLT)*, 2012.

M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *Proceedings of the* 23rd *International Conference on Machine Learning*, pages 65–72. ACM, 2006.

P.L. Bartlett and M.H. Wegkamp. Classification with a reject option using a hinge loss. *Journal of Machine Learning Research*, 9:1823–1840, 2008.

P.L. Bartlett, S. Mendelson, and P. Philips. Local complexities for empirical risk minimization. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, 2004.

O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. In *Advanced Lectures on Machine Learning*, volume 3176 of *Lecture Notes in Computer Science*, pages 169–207. Springer, 2003. ISBN 3-540-23122-6.

John Canny. Lecture 10, chernoff bounds. *Notes from CS174 offered at UC Berkeley*, 2012.

C.K. Chow. On optimum recognition error and reject trade-off. *IEEE Trans. on Information Theory*, 16:41–36, 1970.

D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

Corinna Cortes, Giulia DeSalvo, and Mehryar Mohri. Boosting with abstention. In *Advances in Neural Information Processing Systems*, pages 1660–1668, 2016a.

Corinna Cortes, Giulia DeSalvo, and Mehryar Mohri. Learning with rejection. In *International Conference on Algorithmic Learning Theory*, pages 67–82. Springer, 2016b.

S. Dasgupta. Coarse sample complexity bounds for active learning. In *Advances in Neural Information Processing Systems 18*, pages 235–242, 2005.

S. Dasgupta, D. J. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. In *Advances in Neural Information Processing Systems 20*, pages 353–360, 2007.

R. El-Yaniv and Y. Wiener. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11:1605–1641, 2010.

R. El-Yaniv and Y. Wiener. Agnostic selective classification. In *Neural Information Processing Systems (NIPS)*, pages 1665–1673, 2011.

R. El-Yaniv and Y. Wiener. Active learning via perfect selective classification. *Journal of Machine Learning Research*, 13:255–279, 2012.

Ran El-Yaniv and Yair Wiener. On the version space compression set size and its applications. In *Measures of Complexity*, pages 341–357. Springer International Publishing, 2015.

Y. Freund, H.S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28:133–168, 1997.

Alon Gonen, Sivan Sabato, and Shai Shalev-Shwartz. Efficient active learning of halfspaces: an aggressive approach. *Journal of Machine Learning Research*, 14(1):2583–2615, 2013.

S. Hanneke. A bound on the label complexity of agnostic active learning. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 353–360, 2007.

S. Hanneke. Activized learning: Transforming passive to active with improved label complexity. *The Journal of Machine Learning Research*, 13(5):1469–1587, 2012.

S. Hanneke. Theory of Active Learning. `http://www.stevehanneke.com`, 2014a.

S. Hanneke and L. Yang. Surrogate losses in passive and active learning. *arXiv:1207.3772*, 2012.

Steve Hanneke. Theory of disagreement-based active learning. *Foundations and Trends® in Machine Learning*, 7(2-3):131–309, 2014b. ISSN 1935-8237. doi: 10.1561/2200000037. URL `http://dx.doi.org/10.1561/2200000037`.

Steve Hanneke. Refined error bounds for several learning algorithms. *The Journal of Machine Learning Research*, 17(1):4667–4721, 2016.

Steve Hanneke. Nonparametric active learning, part 1: Smooth regression functions, 2017.

Steve Hanneke and Liu Yang. Minimax analysis of active learning. *Journal of Machine Learning Research*, 16(12):3487–3602, 2015.

D. Hsu. *Algorithms for Active Learning*. PhD thesis, Department of Computer Science and Engineering, School of Engineering, University of California, San Diego, 2010.

Tzu-Kuo Huang, Alekh Agarwal, Daniel J Hsu, John Langford, and Robert E Schapire. Efficient and parsimonious agnostic active learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2755–2763. Curran Associates, Inc., 2015. URL `http://papers.nips.cc/paper/5939-efficient-and-parsimonious-agnostic-active-learning.pdf`.

Mustafa A Kocak, Elza Erkip, and Dennis E Shasha. Conjugate conformal prediction for online binary classification. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 347–356. AUAI Press, 2016.

M. Kääriäinen. Active learning in the non-realizable case. *Lecture Notes in Computer Science, vol 4264. Springer, Berlin, Heidelberg*, 2006.

Andrea Locatelli, Alexandra Carpentier, and Samory Kpotufe. An adaptive strategy for active learning with smooth decision boundary. *arXiv preprint arXiv:1711.09294*, 2017.

Stanislav Minsker. Plug-in approach to active learning. *Journal of Machine Learning Research*, 13(Jan):67–90, 2012.

A.B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *The Annals of Mathematical Statistics*, 32:135–166, 2004.

Vladimir N Vapnik and Alexey J Chervonenkis. Theory of pattern recognition, 1974.

V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic Learning in a Random World*. Springer, New York, 2005.

Y. Wiener. *Theoretical Foundations of Selective Prediction*. PhD thesis, the Technion — Israel Institute of Technology, 2013.

Y. Wiener and R. El-Yaniv. Agnostic pointwise-competitive selective classification. Journal of AI Research, 52:171–201, 2015.

Y. Wiener, S. Hanneke, and R. El-Yaniv. A compression technique for analyzing disagreement-based active learning. *Journal of Machine Learning Research*, 16:713–745, 2015. URL `http://jmlr.org/papers/v16/wiener15a.html`.

Yair Wiener and Ran El-Yaniv. Pointwise tracking the optimal regression function. In *Advances in Neural Information Processing Systems*, pages 2042–2050, 2012.

Ming Yuan and Marten Wegkamp. Classification methods with reject option based on convex risk minimization. *Journal of Machine Learning Research*, 11(Jan):111–130, 2010.

Chicheng Zhang and Kamalika Chaudhuri. Beyond disagreement-based agnostic active learning. In *Advances in Neural Information Processing Systems*, pages 442–450, 2014.

Journal of Machine Learning Research 20 (2019) 1-44 Submitted 3/17; Revised 10/18; Published 3/19

Matched Bipartite Block Model with Covariates

Zahra S. Razaee razaee@ucla.edu
Arash A. Amini aaamini@stat.ucla.edu
Jingyi Jessica Li jli@stat.ucla.edu
University of California, Los Angeles
Department of Statistics
8125 Math Sciences Bldg., Box 951554
Los Angeles, CA 90095-1554, USA

Editor: Edo Airoldi

Abstract

Community detection or clustering is a fundamental task in the analysis of network data. Many real networks have a bipartite structure which makes community detection challenging. In this paper, we consider a model which allows for matched communities in the bipartite setting, in addition to node covariates with information about the matching. We derive a simple fast algorithm for fitting the model based on variational inference ideas and show its effectiveness on both simulated and real data. A variation of the model to allow for degree-correction is also considered, in addition to a novel approach to fitting such degree-corrected models.

Keywords: bipartite networks, community detection, stochastic block model, bipartite matching, node attributes

1. Introduction

Network analysis has been a very active area of research with applications to social sciences, biology and marketing, to name a few. A fundamental problem in network data analysis is community detection, or clustering: Given a collection of nodes and a similarity matrix among them, interpreted as the adjacency matrix of a (weighted) network, one wants to partition the nodes into clusters, or communities, of high similarity. For undirected networks, a popular model for community-structured networks is the stochastic block model (SBM) (Holland et al., 1983) and its variants (Karrer and Newman, 2011; Gopalan and Blei, 2013), which have been extensively investigated in recent years both in terms of theoretical properties and efficient fitting algorithms. See for instance Bickel and Chen (2009); Decelle et al. (2011); Rohe et al. (2011); Mossel et al. (2015); Zhao et al. (2012); Amini et al. (2013); Qin and Rohe (2013); Mossel et al. (2013); Massoulié (2014); Amini and Levina (2018); Gao et al. (2017); Abbe (2017); Jing and Rinaldo (2015); Hajek et al. (2016); Abbe et al. (2016); Gao et al. (2016) for a sample of the work. On the other hand, a natural structure is often present in many real networks, that of being bipartite, where nodes are divided into two sets, or *sides*, and only connections between nodes of different sides are allowed. Examples include networks of actors and movies, scientific papers and their authors, shoppers and products, and proteins and the genes they regulate. Block-modeling with the explicit aim of taking into account the bipartite nature of a network has received comparatively less attention. Interesting new modeling possibilities emerge in the bipartite case, chief among them being the issue of matching between the communities of the two sides.

Finding matched node communities in a bipartite network is a necessary task for network analysis in many applications, especially in biomedical sciences. For example, in molecular biology, a key question is to understand how thousands of proteins regulate their downstream genes in a collaborative manner (Davidson and Levin, 2005; Hecker et al., 2009). Provided with a protein-gene interaction network, constructed from high-throughput biological data, a much needed task is to find clusters of proteins that co-regulate a cluster of genes (Barabasi and Oltvai, 2004). Understanding this complicated protein-gene relationship will shed new light on understanding molecular mechanisms underlying diseases, such as cancers (Madhamshettiwar et al., 2012) and neurodegenerative diseases (Parikshak et al., 2015). For another example, in evolutionary biology, certain genes in two different species share common ancestors in the evolutionary history (Harvey et al., 1991). The pairwise gene conservation relationship is described by an ortholog bipartite network, where an edge connects two conserved genes, one from each species, and such two genes are referred to as *orthologs* (Liao and Zhang, 2005). How to identify two clusters of genes, one cluster in each species, that are jointly evolutionarily conserved, is a matched community detection question. For a third example, in cancer biology, researchers are often interested in linking DNA profiles with gene expression (RNA) profiles of cancer patients (Hedenfalk et al., 2001; Iwakawa et al., 2015; Robinson et al., 2015). In other words, researchers would like to learn what DNA mutations would cause what gene expression changes, so that they can design treatment strategies at the RNA or protein levels given a patient's DNA profile. It is well known that multiple mutations often have a joint effect on the expression levels of multiple genes (Vogelstein and Kinzler, 2004). Thus how to detect such a joint effect given a mutation-gene bipartite network is a matched community detection problem. Another key question in cancer biology is to discover new and rare subtypes for a given type of cancer (Sørlie et al., 2001, 2003; Banerji et al., 2012; Wang et al., 2011). To address this question, one can consider the bipartite gene-patient network where the edges are mutation status (or the expression value) of a gene in a patient. This is clearly a matched community detection problem where the co-cluster of the genes and patients determine the cancer subtypes.

In addition to biomedical sciences, finding matched clusters in a bipartite network also has broad applications in social sciences. For example, consider the author–paper bipartite networks where an edge signifies the authorship. There is a matching between the authors and the papers in these networks since most authors follow a theme in their publications. The matched communities in this case correspond to high-level fields of study. As an another example, consider the Wikipedia page–user networks where an edge captures the act of editing a page. In these networks, the users also congregate in groups based on their interests and the subject matter of the pages or, say, their language. The common subject or the language constitutes a matched community pair in this case. We refer to Section 5 for a detailed analysis of some of these networks.

The problem of community detection in bipartite networks is closely related to that of co-clustering, also known as bi-clustering, which goes back at least to (Hartingan, 1972). Co-clustering refers to simultaneous clustering of the rows and columns of a matrix, the *bi-adjacency matrix* of a bipartite graph. It has been extensively used in biological applications (Cheng and Church, 2000; Madeira et al., 2010) and text mining (Dhillon, 2001, 2003; Bisson and Hussain, 2008). Recently, (Choi and Wolfe, 2014; Flynn and Perry, 2012) studied

©2019 Zahra S. Razaee, Arash A. Amini, Jingyi Jessica Li.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-153.html.

likelihood-based co-clustering. Rohe et al. (2016) proposed a spectral co-clustering algorithm for directed networks and discussed how it can be applied to bipartite setting. Often, the co-clustering formulation ignores the issue of matching of the clusters, in the sense that in general any row cluster can be in relation to any column cluster.

Another common approach is to reduce community detection in the bipartite setting to two separate instances of usual clustering of (undirected) unipartite networks, by forming *one-mode projections* of the network onto the two sides (Zhou et al., 2007). Despite a moderate reduction in the dimension (having to deal with two smaller networks), the projection approach suffers from information loss and identifiability issues (Zhou et al., 2007). Projection can also turn a structured bipartite network into unstructured unipartite ones or vice versa (Larremore et al., 2014). Another major difficulty is establishing a link between the communities on the two sides. One can come up with ad-hoc association measures between communities of the two sides, e.g., by counting links between each pair. This, however, leads to another bipartite graph on the communities, leading to the difficulty of interpretation. In effect, the problem transfers from community detection on the individual nodes, to that on the newly-discovered communities, or supernodes.

Block-modeling in the bipartite setting has recently gained more attention. Wyse et al. (2014) proposed a method to infer both community memberships as well as the number of communities in a bipartite network using a block model and an algorithm similar to the iterated conditional modes (Besag, 1986). Larremore et al. (2014) has proposed a bipartite stochastic block model (BiSBM) that built on the work of Karrer and Newman (2011) to infer bipartite community structure in both degree-corrected and uncorrected regimes by maximizing a profile likelihood over all partitions. In both cases, the issue of matching of the communities on the two sides is not the main concern.

Motivated by the matching problem, in this paper, we consider the problem of *matched community detection* in a bipartite network. In many practical examples, one either expects a one-to-one correspondence between the communities of the two sides, or it is reasonable to postulate such structure, due to ease of interpretation (Section 2). The problem of "finding communities in a bipartite network that are in one-to-one correspondence between the two sides" is what we refer to as matched community detection. In its simplest form, the model assumes that nodes belonging to matched communities have a higher probability of connection (cf. Equation (3)) and may also have correlated values for their nodal covariates. In other words, we will propose a generative model for such networks where there is a hidden matched community structure that affects the distribution of the observed network and the nodal covariates. This avoids the need for post-hoc matching of the communities: the matching is built into the model and inferred simultaneously along with the communities in the process of fitting the model. Our model is a natural extension of the well-known stochastic block model (SBM) and is discussed in detail in Section 2. We also discuss an extension of our model to allow for degree-correction (Section A.3), providing a matched version of the degree-corrected block model (DC-SBM).

In another direction, many networks come with metadata, often in the form of node attributes, or covariates. In our four motivating examples from biomedical sciences, node covariates are often available and provide useful information for detecting matched communities. In the protein-gene example, expression levels of the proteins and genes are natural node covariates (Segal et al., 2003; Bansal et al., 2006). Incorporating the covariate information will allow us to more accurately identify a group of proteins that co-regulate a cluster of genes,

because it is expected that the proteins in a group should share similar expression patterns, and likewise for the genes. In the ortholog example, the expression levels of genes in each species are useful node covariates, which will help researchers better identify clusters of genes that are not only conserved between the two species but also share similar functional characteristics within each species (Li et al., 2014; Gerstein et al., 2014). In the mutation-gene network, gene expression levels are the covariates of the gene nodes. Using this covariate information will be useful for identifying the active genes as a result of mutations (Hedenfalk et al., 2001; Iwakawa et al., 2015; Robinson et al., 2015). For cancer sub-type detection example, using the patient's data such as age, gender, race, progression-free and overall survival time, the primary site of the tumor as well as its stage and grade, etc. as node covariates will help in identifying rare and often more aggressive subtypes (Cheang et al., 2009; Arvold et al., 2011; Von Minckwitz et al., 2012). For the author–paper example, the frequencies of the words in each paper can be considered as node covariates for the paper side. In the Wikipedia user–page networks, the location of the users can be informative covariates.

The potential for improving quality of the clusters by incorporating node covariates has been explored in recent work, in the context of unipartite networks (Binkiewicz et al., 2017; Zhang et al., 2016; Yan and Sarkar, 2016; Newman and Clauset, 2016). The bipartite setting adds another challenge to modeling node covariates, in particular, how to jointly model the covariates on the two sides, considering that one often has covariates of different dimensions on each side. (The extreme case is when only one side has node covariates.) We extend our proposed model to allow for the presence of node covariates that are aware of the matching between communities of the two sides. In other words, covariates corresponding to nodes in matched communities are statistically linked. The linkage can be tuned using a general cross-covariance matrix, allowing for varying degrees of covariate influence on the community detection problem (Section 2). It is worth noting that we specifically model the problems where the network and node covariates are driven by a single latent community structure. This is often a plausible assumption in many applications. In these cases, one expects to obtain more accurate community estimates by properly combining the two sources of information; our model allows for a natural incorporation of these two sources by maximizing the joint likelihood. Modeling cases where the node covariates and the network provide conflicting information about a potential clustering remains a challenging task and is outside the scope of the present paper.

To fit our models, we derive an algorithm based on the variational inference, also known as the variational Bayes (Jordan et al., 1999; Blei et al., 2017) ideas (Section 3). We derive both the degree-corrected and uncorrected versions of algorithm within the same unified framework, namely, sequential block-coordinate ascent on the variational likelihood. This in particular leads to a novel approach to fitting degree-corrected likelihoods using methods of continuous optimization (as opposed to profiling out the degree-correction parameters and optimizing over the space of discrete labels.). As part of the initialization of the algorithm, we revisit a bipartite spectral clustering algorithm, `biSC` first proposed in Dhillon (2001), and identify it as an effective algorithm for matched bipartite clustering. We show the effectiveness of our approach on simulated (Section 4) and real data, namely, page-user networks collected from Wikipedia and two sets of author-paper networks, one extracted from Arnetminer collection by Tang et al. (2012) and the other scraped from DBLP(Section 5).

To summarize, our contributions in this paper are the following:

(i) Identify the matching problem in bipartite community detection more clearly and give it the prominent role, by showing that it is possible to consider matched communities from the start in the modeling process. Bringing attention to matched bipartite clustering (or community detection) as a well-defined problem also allows us to identify an earlier spectral algorithm, namely that of Dhillon (2001), originally proposed in the context of topic modeling, as effectively solving the matched version of bipartite clustering. At present, we are unaware of any other algorithm that attempts to solve this problem directly.

(ii) Propose a natural bipartite extension of the SBM and DC-SBM: *matched bipartite stochastic block model* (`mbiSBM`), which has a latent structure of matched communities and allows for node covariates that are potentially informative about the matching (see Section 2). Some of the challenges involved in joint modeling of the node covariates of the two sides are resolved by appealing to hierarchical Bayesian modeling ideas (Gelman et al., 2003).

(iii) Show the effectiveness of the variational Bayes approach in fitting the overall `mbiSBM` model, when combined with good initialization, especially a variant of the `biSC` algorithm of Dhillon (2001). The algorithm is a block-coordinate ascent with a closed-form, fairly cheap iterations, and can be scaled to large networks.

Notation. We write $[K] := \{1, \dots, K\}$ and $\mathcal{P}_K := \{p \in \mathbb{R}_+^K : \mathbf{1}^T p = 1\}$, for the set of probability vectors on $[K]$. Here, $\mathbf{1}$ is the all-ones vector of dimension K. We identify $[K]$ with $\{0,1\}^K \cap \mathcal{P}_K$, the set of binary vectors of length K having exactly a single entry equal to one. The identification is via the so-called *one-hot* encoding: $z = k$ as an element of $[K]$ iff $z_k = 1$, treating z as element of $\{0,1\}^K \cap \mathcal{P}_K$. We will use I_d to denoted the $d \times d$ identity matrix.

The probability density function (PDF) of a multivariate Gaussian distribution with mean μ and covariance Σ is denoted as $x \mapsto N(x; \mu, \Sigma)$. The constant terms in an expression are denoted as "const.". We write $\doteq$ for equality up to additive constants. We use $[Z_1; Z_2]$ to denote the vertical concatenation of two matrices Z_1 and Z_2, having the same number of columns.

2. Matched Bipartite SBM

The stochastic block model (SBM) is a generative model for networks with communities (or blocks). In the most basic SBM, sometimes called the *planted partition model*, each node is assigned to one of the K communities and the edges are placed independently between two nodes i and j, with probability p if i and j belong to the same community, and with probability q otherwise. When $p = q$, one recovers the famous Erdős–Rényi model, where there is no genuine community structure. The interesting cases are the *assortative* model where $p > q$ and the *dissortative* model where $p < q$. Our focus in this paper is mainly on the assortative case, though the results can be easily adapted to the other case.

We start with the ingredients needed to define the matched bipartite SBM (`mbiSBM`). Assume that we have two groups of nodes $[N_1] = \{1, \dots, N_1\}$ and $[N_2] = \{1, \dots, N_2\}$, representing nodes on the two sides of a bipartite network. We assume that there is a partition $\{C_{rk}\}_{k=1}^K$ of $[N_r]$, for each $r = 1, 2$. This is our latent community structure. In referring to C_{rk}, we will use the terms *community* and *cluster* interchangeably. We assume the following implicit (true) *one-to-one matching* between these communities:

$$C_{1k} \leftrightarrow C_{2k}, \quad k = 1, \dots, K. \tag{1}$$

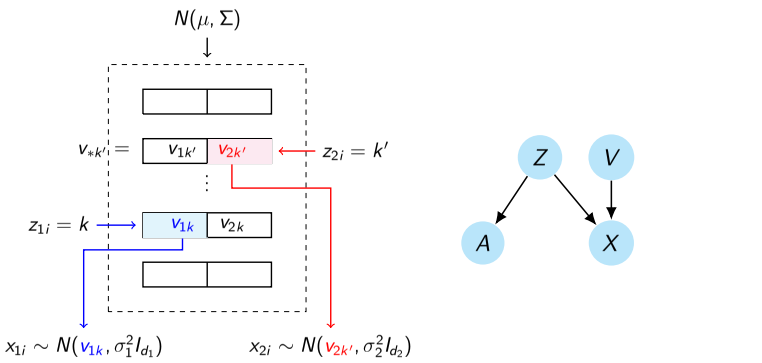


Figure 1: Schematic diagram for the hierarchical generation of node covariates (left) and the overall graphical representation of the model (right). The label z_{ri} selects which of $v_{k*} = (v_{1k}, v_{2k})$ for $k = 1, \dots, K$ we select and the index "$r \in \{1, 2\}$" determines which component of v_{k*} is used to generate x_{ri}. For example, if $z_{1i} = k$ then the v_{1k} component of v_{k*} is used as the mean of x_{1i}, and similarly if you replace 1 with 2. Here, k and k' distinguish two different clusters. For example, if $z_{2i} = k'$ then $v_{2k'}$ will be the mean of x_{2i}, and so on.

To each node i in group r, we assign a community membership variable z_{ri} showing which community it belongs to:

$$z_{ri} = k \iff i \in C_{rk}, \quad \forall i \in [N_r],\ r = 1, 2.$$

Recalling the identification $[K] \cong \{0,1\}^K \cap \mathcal{P}^K$, we treat z_{ri} as both an element of $[K]$ and a binary vector of length K, hence, with some abuse of notation, $z_{ri} = k$ and $z_{rik} = 1$ are equivalent. We collect these labels in *membership matrices* $Z_r := (z_{ri} : i \in [N_r]) \in \{0,1\}^{N_r \times K}$, $r = 1, 2$, where each z_{ri}, treated as a binary vector, appears as a row in Z_r. We also let $Z = [Z_1; Z_2] \in \{0,1\}^{(N_1+N_2)\times K}$ be the *matched membership matrix* obtained by vertical concatenation of Z_1 and Z_2.

For each node i in group r, we observe a covariate vector $x_{ri} \in \mathbb{R}^{d_r}$. If we want to specify the components of this vector we write $x_{rij}, j = 1, \dots, d_r$. Let $X := (x_{ri}, i \in [N_r], r = 1, 2)$. We often think of X as a matrix in $\mathbb{R}^{(N_1+N_2)\times(d_1+d_2)}$, by padding covariate vectors with zeros on the left or right: x_{1i} form rows $(x_{1i}, 0_{d_2})$ for $i \in [N_1]$ and x_{2j} form rows $(0_{d_1}, x_{2j})$ for $j \in [N_2]$.

In addition to the covariate matrix X, we also observe a bipartite network on $[N_1] \times [N_2]$ represented as a *bi-adjacency matrix* $A \in \{0,1\}^{N_1 \times N_2}$. Thus, the observed data is (X, A). We assume that given the latent community labels Z, X is independent of A. Below, we outline how each of these components are generated given Z.

2.1. Generating Covariates

To generate x_{ri}, we use a hierarchical mixture model: First we generate the mean vector $v_{rk} \in \mathbb{R}^{d_r}$ associated with each cluster C_{rk}, and then we draw x_{ri} from a normal distribution with mean v_{rk} when $z_{ri} = k$; see Figure 1. In order to model the correlation (i.e., a statistical link) between covariates of matched clusters, we draw the entire vector $v_{*k} := (v_{rk}, r = 1, 2) =$

(v_{1k}, v_{2k}) from a multivariate normal distribution with possibly nonzero covariance matrix between the two components v_{1k}, v_{2k}. We have the following model:

$$\begin{aligned} z_{ri} &\sim \text{Mult}(1, \pi_r), \\ (v_{rk}, r = 1, 2) &\overset{\text{iid}}{\sim} N(\mu, \Sigma), \quad k = 1, \ldots, K. \\ (x_{ri} \mid z_{ri} = k,\, v_{rk}) &\sim N(v_{rk}, \sigma_r^2 I_{d_r}), \quad i \in [N_r],\ r = 1, 2 \end{aligned} \tag{2}$$

where the draws are independent over r and $i \in [N_r]$, on each line. Here, $\pi_r = (\pi_{r1}, \ldots, \pi_{rK})$ is the prior on cluster proportions for group r ($\pi_r \in [0,1]^K$ with $\sum_{k=1}^K \pi_{rk} = 1$). $(\sigma_r^2, r = 1, 2)$ models the variance of measurement noise in the two groups.

The idea behind the covariate generation is as follows: Each covariate x_{ri} is going to to be determined by v_{rk}, that is x_{1i} will have mean v_{1k} and x_{2j} will have mean v_{2k}, assuming nodes i and j belong to the same cluster k (a single cluster encompasses both sides of the network, based on the matching). That is, we break each $v_{*k} = (v_{1k}, v_{2k})$ into two pieces v_{1k} and v_{2k} and these two pieces determine the covariates of the two sides, x_{1i} and x_{2j}, assuming nodes i and j belong to the same cluster k.

Note that the covariates for two nodes that are in the same cluster on the same side have the same mean: If $z_{1i} = z_{1j} = k$ then both x_{1i} and x_{1j} have mean v_{1k}. On the other hand the covariates for two nodes that are in the same cluster but on different sides will have different means: If $z_{1i} = z_{2j} = k$, then x_{1i} has mean v_{1k} while x_{2j} as mean v_{2k}. These two mean vectors however are still related since they are the components of the single vector $v_{k*} = (v_{1k}, v_{2k})$ which is derived from a multivariate Gaussian distribution. That is, v_{1k} and v_{2k} are jointly Gaussian with a potential nonzero cross-covariance matrix.

To make the correlation structure in $(v_{rk}, r = 1, 2)$ more explicit, we can partition $\mu = (\mu_r, r = 1, 2)$ and Σ, so that

$$\begin{pmatrix} v_{1k} \\ v_{2k} \end{pmatrix} \overset{\text{ind}}{\sim} N\left[\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12}^T & \Sigma_{22} \end{pmatrix} \right], \quad k = 1, \ldots, K.$$

Note that $\mu_r \in \mathbb{R}^{d_r}$. In subsection 2.5, we discuss how this model provides a statistical link between covariates of the two groups. For future reference, $v_{*k} := (v_{rk}, r = 1, 2)$ collects the matched hidden covariate means of the clusters C_{1k} and C_{2k}. On the other hand, we write $v_{r*} = (v_{rk}, k \in [K])$ which collects all the hidden covariate means for side $r = 1, 2$ of the network.

2.2. Generating the Network

Given Z, the bipartite graph is generated as follows: For each node i in $[N_1]$ and each node j in $[N_2]$, we put an edge between them with probability p if they belong to matched clusters, and with probability $q \neq p$ otherwise. With $A = (A_{ij}) \in \{0,1\}^{N_1 \times N_2}$ denoting the resulting bi-adjacency matrix, we have

$$A_{ij} \mid Z \overset{\text{ind}}{\sim} \begin{cases} \text{Ber}(p) & z_{1i} = z_{2j} \\ \text{Ber}(q) & z_{1i} \neq z_{2j} \end{cases}, \quad i \in [N_1],\ j \in [N_2]. \tag{3}$$

Combined, (2) and (3) describe our full matched bipartite SBM model. The objective is to find the posterior probability of Z given A and $X = \{x_{ri} : i \in [N_r], r = 1, 2\}$.

Although we will focus on the simple model (3) in deriving the algorithms, it is possible to allow for a more general edge probability structure as in the usual SBM, by assuming

$$A_{ij} \mid Z \overset{\text{ind}}{\sim} \text{Ber}(\Psi_{z_{1i}, z_{2j}}) \quad i \in [N_1],\ j \in [N_2]. \tag{4}$$

where $\Psi \in [0,1]^{K \times K}$ is a connectivity (or edge probability) matrix. Model (3) corresponds to the case where $\Psi_{kk} = p$ and $\Psi_{k\ell} = q$ for $k \neq \ell$. We will refer to this model as `mbiSBM` for matched bipartite SBM.

Remark 1. Parameter Σ in (2) is key in tuning the effect of the node covariates on community detection. Assume for simplicity that $\sigma_r^2 = 0$, $r = 1, 2$. Then, when $\Sigma = 0$, $v_{*k} = \mu$ a.s. for all k, hence $x_{*i} = \mu$ for all i, and the covariates carry no information about communities. When, $\Sigma \neq 0$ there is variability in v_{*k} across k, hence community detection benefits from the covariate information. On the other hand, it is well-known that the information in the adjacency matrix A about community structure is roughly controlled by the expected degree of the network, i.e., the scaling of $Q = (p, q)$, in addition to the separation of p and q. By scaling of (p, q) we mean the following: One can take $p = a/n$ and $q = b/n$; then, how fast a and b increase as a function of n determines the difficulty of the network community detection problem. For the case $a, b = O(1)$, the so-called sparse regime, only partial recovery of the labels is possible (given only the network information), whereas when $a, b \to \infty$ one can recover with asymptotically vanishing misclassification error; at higher densities, namely $a, b \gtrsim \log n$, it is possible to exactly recover the labels for sufficiently large n. See Abbe (2017) for more details. Thus, by rescaling Σ and $Q = (p, q)$, we can control the balance of the two sources of information (i.e., the network versus node covariate information). This is explored in Section 4 through simulation studies.

2.3. Connection with the Usual SBM

Ignoring the covariate part of the model, one might wonder whether `mbiSBM`, introduced in (4), can be thought of as a sub-model of a usual SBM with perhaps increased number of communities. First, it should be clear that the model is not a usual SBM with K communities. However, it can be thought of as a SBM with $2K$ communities with *restrictions* imposed on both its membership and connectivity matrix. To see this, let us recall the matrix representation of the usual SBM with K blocks, where one has the connectivity matrix $\Psi \in [0,1]^{K \times K}$ and binary membership matrix $Z \in \{0,1\}^{N \times K}$. Such model can be compactly represented as $\mathbb{E}[A|Z] = Z\Psi Z^T$.

Now, consider model (4), and let $Z_r = (z_{ri}) \in \{0,1\}^{N_r \times K}$ for $r = 1, 2$. We express the model compactly as

$$\mathbb{E} \underbrace{\begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}}_{\widetilde{A}} = \underbrace{\begin{pmatrix} Z_1 & 0 \\ 0 & Z_2 \end{pmatrix}}_{\widetilde{Z}} \underbrace{\begin{pmatrix} 0 & \Psi \\ \Psi^T & 0 \end{pmatrix}}_{\widetilde{\Psi}} \begin{pmatrix} Z_1^T & 0 \\ 0 & Z_2^T \end{pmatrix}. \tag{5}$$

given Z_1, Z_2. Letting $N := N_1 + N_2$ and defining the matrices $\widetilde{A} \in \{0,1\}^{N \times N}$, $\widetilde{Z} \in \{0,1\}^{N \times 2K}$ and $\widetilde{\Psi} \in [0,1]^{2K \times 2K}$ as in (5), it is clear that model (4) is equivalent to $\mathbb{E}[\widetilde{A}|\widetilde{Z}] = \widetilde{Z}\widetilde{\Psi}\widetilde{Z}^T$. This is a SBM with restrictions on both $\widetilde{Z}$ and $\widetilde{\Psi}$: Nodes $1, \ldots, N_1$ can only belong to communities $1, \ldots, K$ and nodes $N_1 + 1, \ldots, N_1 + N_2$ can only belong to communities $K + 1, \ldots, 2K$. As for $\widetilde{\Psi}$, the restriction imposes zero connectivity among communities $1, \ldots, K$ and among those of

$K+1, \ldots, 2K$. With these restrictions in place, we have a natural bipartite matching between communities: $\ell \leftrightarrow K+\ell$ for $\ell \in [K]$.

2.4. Degree-corrected Version

A limitation of the SBM is that nodes in the same community have the same expected degree. To allow for degree heterogeneity within communities, bringing the model closer to real networks, a common approach is to use the DC-SBM (Dasgupta et al., 2004; Karrer and Newman, 2011). It is fairly straightforward to introduce degree-correction in our setup. Consider the form of the matched SBM introduced in (4). To each node i in group r, we associate a propensity parameter $\theta_{ri} > 0$. Thus, we have additional parameters $\theta_r := (\theta_{ri},\ i \in [N_r])$ for $r = 1, 2$. The degree-corrected (DC) version of the model replaces (4) with

$$A_{ij} \mid Z \overset{\text{ind}}{\sim} \text{Poi}(\theta_{1i}\theta_{2j}\Psi_{z_{1i},\, z_{2j}}) \quad i \in [N_1],\ j \in [N_2]. \tag{6}$$

Replacing the Bernoulli with Poisson is for convenience in later derivations, and is common in dealing with DC-SBM (Karrer and Newman, 2011). In order for the parameters $(\theta_1, \theta_2, \Psi)$ to be identifiable, we need to agree on a normalization of θ_{ri} per each community. Here, we adopt the following:

$$\frac{1}{|C_{rk}|}\sum_{i \in C_{rk}} \theta_{ri} = 1 \iff \sum_{i=1}^{N_r}(\theta_{ri} - 1)z_{rik} = 0, \quad k \in [K],\ r = 1, 2. \tag{7}$$

With this normalization, we recover the original model when $\theta_{ri} = 1$ for all i and r. Our normalization is similar to the one considered in Gao et al. (2016).

Remark 2. A normalization of the form (7) is often assumed when one considers both $\theta = (\theta_1, \theta_2)$ and Z to be deterministic unknown parameters, or alternatively when working conditioned on θ and Z. Throughout, we assume θ to be an unknown parameter. However, to be pedantic, (7) is inconsistent with i.i.d. random generation of z_{ri} from a $\text{Mult}(1, \pi_r)$ as in (2). One way to get around this is to assume that the labels are generated a priori from the product multinomial distribution described in (2) *conditioned* on the set of labels satisfying (7). We will ignore the change in the label prior this conditioning makes in deriving the algorithms. In the end, we enforce (7) in an "averaged" sense, replacing z_{rik} with the corresponding (approximate) posterior τ_{rik}, as detailed in subsection A.3. Viewed as a set of constraints on the collection of soft-labels (τ_{rik}), (7) is not that restrictive.

2.5. Covariate Correlation on Matched Clusters

One desirable feature in modeling covariates, in the context of a matched bipartite network, is the ability to gain some information about whether a pair $(i, j) \in [N_1] \times [N_2]$ belongs to a matched cluster, by just looking at their respective covariates x_{1i} and x_{2j}. Assume for the moment that there is no measurement noise, i.e., $\sigma_r^2 = 0$, $r = 1, 2$. Then, the question boils down to whether we can tell $(v_{1k}, v_{2k'})$ for $k \neq k'$ apart from (v_{1k}, v_{2k}). According to the model, (v_{1k}, v_{2k}) and $(v_{1k'}, v_{2k'})$ are independent Gaussian vectors, hence $(v_{1k}, v_{2k}, v_{1k'}, v_{2k'})$ is Gaussian with mean $(\mu, \mu) = (\mu_1, \mu_2, \mu_1, \mu_2)$ and covariance $\left(\begin{smallmatrix}\Sigma & 0\\ 0 & \Sigma\end{smallmatrix}\right)$. Recalling the decomposition of Σ, it follows that $(v_{1k}, v_{2k'}) \sim N(\mu, \left(\begin{smallmatrix}\Sigma_{11} & 0\\ 0 & \Sigma_{22}\end{smallmatrix}\right))$ for $k \neq k'$ whereas $(v_{1k}, v_{2k}) \sim N(\mu, \left(\begin{smallmatrix}\Sigma_{11} & \Sigma_{12}\\ \Sigma_{21} & \Sigma_{22}\end{smallmatrix}\right))$. As long as $\Sigma_{12} \neq 0$, these two distributions are different, hence the model is able to distinguish

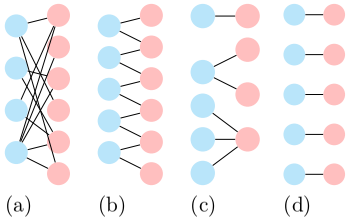


Figure 2: Possible relations between communities of the two sides, in a bipartite network. (a) and (b) are hard to interpret. Structures like (c), i.e., collections of disjoint stars, are interpretable and (d) is the simplest within this class.

between the two cases. In other words, there is information in the covariates about the matching of the clusters in the two groups. However, this information (in itself) is quite weak since it amounts to distinguishing between two multivariate Gaussian distributions, based only on a single draw from each. Fortunately, the model also carries information about the matching in the adjacency matrix A.

2.6. Interpretability and Identifiability

We alluded earlier to the merits of having a 1-1 matching between the communities of the two sides built into the model. Our main argument for the advantage of a 1-1 matching is interpretability. Figure 2 shows some possible relations that could exist between communities of the two side (each circle represents a community). The closer this relation is to a complete bipartite graph, the harder it is to interpret; Figure 2(a) is perhaps the least informative relation among the four. In Figure (b), the relation is much sparser. However, it is still hard to interpret: all communities seem to be related, albeit indirectly. We would like to argue that structures like (c) where the graph is a collection of disjoint *stars* is interpretable. One would like to fit such models, though in full generality, this seems to be a difficult task. One has to somehow control the branching numbers of the stars, which indirectly control the number of communities on either side. Thus, the problem is at least as hard as deciding the number of communities in the usual SBM. Our 1-1 matching relation, Figure 2(d), is the simplest structure of the type depicted in (c). It is in a sense a first-order approximation of the models in this class. It is easiest to fit and is the most interpretable.

3. Model Fitting

In order to fit the model, we derive algorithms based on variational inference ideas. The algorithm starts from some initial guess of the labels and parameters and proceeds to improve the likelihood via simple iterative updates to the parameters and an approximate posterior on the labels. We first discuss the case with no degree correction. The extension to the degree-corrected model is discussed in subsection A.3.

3.1. The Likelihood

Let us introduce some notation. We write $v_{*k} = (v_{rk}, r = 1, 2) \in \mathbb{R}^{d_1+d_2}$ and $v_{r*} = (v_{rk}, k \in [K]) \in \mathbb{R}^{Kd_r}$, and $V = (v_{rk},\, r = 1, 2,\, k \in [K]) \in \mathbb{R}^{K(d_1+d_2)}$. Similarly, $Z = (z_{ri},\, i \in [N_r],\, r = 1, 2)$ and $X = (x_{ri},\, i \in [N_r],\, r = 1, 2)$. (In this section, the particular matrix form of Z and X are not of interest. Z and X are simply placeholders for the collections of labels and covariates.) Let

$$y_{ij} := \mathbb{1}\{z_{1i} = z_{2j}\}, \quad z_{rik} := \mathbb{1}\{z_{ri} = k\}.$$

The joint distribution of all the variables in the model factorizes as follows:

$$\begin{aligned} p(A, X, Z, V) &= p(A|Z)\, p(X|Z, V)\, p(Z)\, p(V) \\ &= \prod_{i=1}^{N_1}\prod_{j=1}^{N_2} p(A_{ij}|z_{1i}, z_{2j}) \prod_{r=1}^{2}\prod_{i=1}^{N_r} \Big\{ p(x_{ri}|z_{ri}, v_{r*}) p(z_{ri}) \Big\} \prod_{k=1}^{K} p(v_{*k}). \end{aligned}$$

We have $p(x_{ri}|z_{ri}, v_{r*}) = \prod_{k=1}^{K} [f_r(x_{ri}; v_{rk})]^{z_{rik}}$ where $f_r(x_{ri}; v_{rk}) := N(x_{ri}; v_{rk}, \sigma_r^2 I_{d_r})$. In addition, $p(z_{ri}) = \prod_{k=1}^{K} \pi_{rk}{}^{z_{rik}}$. For the network part, we in general have

$$\ell_1(\Psi) =: \log p(A|Z) = \sum_{ij}\sum_{k\ell} z_{1ik} z_{2j\ell}\; g(\Psi_{k\ell}, A_{ij}) \tag{8}$$

where g is either the log-likelihood of the Bernoulli, $g_{\text{ber}}(p, \alpha) = \alpha \log \frac{p}{1-p} + \log(1 - p)$, or Poisson, $g_{\text{poi}}(p, \alpha) = \alpha \log p - p$.

In the special planted partition case, $\log p(A|Z)$ greatly simplifies: By breaking up over $k = \ell$ and $k \neq \ell$, we obtain

$$\begin{aligned} \ell_1(\Psi) = \log p(A|Z) &= \sum_{ij} \Big[\sum_{k} z_{1ik} z_{2jk}\; g(p, A_{ij}) + \sum_{k\neq\ell} z_{1ik} z_{2j\ell}\; g(q, A_{ij}) \Big] \\ &= \sum_{ij} y_{ij} g(p, A_{ij}) + (1 - y_{ij}) g(q, A_{ij}). \end{aligned} \tag{9}$$

where we have used $\sum_{k\ell} z_{1ik} z_{2j\ell} = 1$. The complete log-likehood of the model, i.e., assuming we observe the latent variables (Z, V), is then

$$\ell(\mu, \Sigma, \sigma, \pi, \Psi) = \ell_1(\Psi) + \ell_2(\mu, \Sigma, \sigma, \pi)$$

where $\ell_1(\Psi)$ is as defined in (8) and

$$\ell_2(\mu, \Sigma, \sigma, \pi) = \sum_{r=1}^{2}\sum_{i=1}^{N_r}\sum_{k} z_{rik} \log \big[\pi_{rk}\, f_r(x_{ri}; v_{rk})\big] + \sum_{k} \log p(v_{*k}|\mu, \Sigma).$$

3.2. Mean-Field Approximation

Variational inference is often regarded as the approximation of a posterior distribution by solving an optimization problem (Wainwright and Jordan, 2008; Blei et al., 2017). The idea is to pick an approximation q from some tractable family of distributions over the latent variables (Z, V) and try to make this approximation as close as possible in KL divergence to the true

posterior. We prefer to think of the approach as a generalization of the EM algorithm, i.e., a general approach to maximize the incomplete likelihood by maximizing a lower bound on it. This lower bound, which we call variational likelihood, also known as the evidence lower bound (ELBO) (Jordan et al., 1999), involves both the likelihood parameters and a distribution q, namely,

$$J := \mathbb{E}_q[\ell(\mu, \Sigma, \sigma, \pi, \Psi) - \log q(Z, V)]. \tag{10}$$

Here the expectation is taken, assuming $(Z, V) \sim q$. One maximizes J by alternating between maximizing over likelihood parameters $(\mu, \Sigma, \sigma, \pi, \Psi)$ and the variational posterior q. Without additional constraints, the optimization over q leads to the posterior distribution of (Z, V) given (X, A), resulting in the EM algorithm. A genuine variational inference procedure, however, imposes some simplifying constraints on q. In particular, we impose the following factorized form, often referred to as the *mean-field approximation*:

$$q(Z, V) = q_V(V) q_Z(Z), \quad q_Z(Z) = \textstyle\prod_{r,i} q_{ri}(z_{ri}), \quad q_V(V) = \prod_{k=1}^{K} N(v_{*k}; \tilde{\mu}_k, \tilde{\Sigma}_k) \tag{11}$$

where $q_{ri}(z_{ri}) = \prod_{k=1}^{K} \tau_{rik}^{z_{rik}}$ is a multinomial distribution. In keeping up with our notation we write $\tau_{ri} = (\tau_{rik},\ k \in [K])$. Note that $\tau = (\tau_{ri})$ collects the approximate posteriors on node labels. They are the key parameters in our inference.

The particular form assumed for q_V in (11) is motivated by looking at the (true) posterior of V given Z. We could have assumed a factorized form $q(Z, V) = q_Z(Z) p(V|Z)$ where $p(V|Z)$ is the the true posterior of V given Z. However, the parameters of $p(V|Z)$ have a complicated dependence on Z. We have kept the form of $p(V|Z)$ while freeing the parameters, letting them be optimized by the algorithm.

To simplify notation, let us define $\widetilde{\Gamma} := ((\widetilde{\Sigma}_k, \widetilde{\mu}_k), k = 1, \ldots, K)$, collecting the parameters for the variational posterior q_V. Plugging in the variational distribution (11) into the variational likelihood (10) using expression (9) for $\ell_1(\Psi)$, after some algebra detailed in Appendix B.2, we get

$$\begin{aligned} J = &\sum_{i,j} \Big[\gamma_{ij}(\tau) g(p; A_{ij}) + (1 - \gamma_{ij}(\tau)) g(q; A_{ij}) \Big] + \sum_{r,i,k} \tau_{rik} \big[\beta_{rik}(\widetilde{\Gamma}, \sigma^2) + \log \frac{\pi_{rk}}{\tau_{rik}} \big] \\ &- \frac{1}{2} \sum_{r} d_r N_r \log \sigma_r^2 - \frac{K}{2} \big\{ \log |\Sigma| + \operatorname{tr}[\Sigma^{-1} S(\widetilde{\Gamma}, \mu)] \big\} + \frac{1}{2} \sum_{k} \log |\widetilde{\Sigma}_k| + \text{const.} \end{aligned} \tag{12}$$

where

$$\gamma_{ij}(\tau) := \mathbb{E}_{q_Z}(y_{ij}) = \sum_{k=1}^{K} \tau_{1ik} \tau_{2jk}, \quad \beta_{rik}(\widetilde{\Gamma}, \sigma^2) := -\frac{1}{2\sigma_r^2} \Big[\operatorname{tr}\big((\widetilde{\Sigma}_k)_{rr}\big) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2 \Big], \tag{13}$$

$(\widetilde{\Sigma}_k)_{rr}, r = 1, 2$ refers to the two diagonal blocks of $\widetilde{\Sigma}_k$ of sizes $d_r \times d_r$, and

$$S(\widetilde{\Gamma}, \mu) := \frac{1}{K} \sum_{k=1}^{K} \big[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T \big]. \tag{14}$$

3.3. Optimizing the Variational Likelihood

We proceed to maximize J by alternating between the likelihood parameters $(\mu, \Sigma, \sigma, \pi, \Psi)$ and variational parameters $(\tau, \widetilde{\Gamma})$. Each of these two sets of parameters is also optimized by alternating maximization. In other words, the overall optimization algorithm is a block coordinate ascent. The key update is that of label distributions τ, which we describe in details below. The other updates are more or less standard and detailed in Appendix B.4.

Updating node labels (τ). To optimize τ, we use block coordinate ascent, by fixing $\tau_2 := (\tau_{2j}, j \in [N_2])$ and optimizing over $\tau_1 := (\tau_{1j}, j \in [N_1])$ and vice versa. Here we only consider optimization over τ_1 given τ_2. To simplify notation, let $h(p, q; \alpha) := g(p; \alpha) - g(q; \alpha)$. Considering only the terms in J that depend on τ, we have

$$J = \sum_{ij} \gamma_{ij}(\tau)\, h(p, q; A_{ij}) + \sum_{r,i,k} \tau_{rik} \big[\beta_{rik}(\widetilde{\Gamma}, \sigma^2) + \log \frac{\pi_{rk}}{\tau_{rik}}\big] + \text{const.}$$

where const. collects terms that do not depend on τ. Let $\xi_{rik} := \beta_{rik}(\widetilde{\Gamma}, \sigma^2) + \log \pi_{rk}$. Using the definition of $\gamma_{ij}(\tau) = \sum_{k=1}^{K} \tau_{1ik}\tau_{2jk}$,

$$J = \sum_{ij} \Big(\sum_{k} \tau_{1ik}\tau_{2jk} \Big)\, h(p, q; A_{ij}) + \sum_{r,i,k} \tau_{rik} \big[\xi_{rik} - \log \tau_{rik}\big] + \text{const.}$$

Now, assume further that τ_2 is constant. Then,

$$\begin{aligned} J &= \sum_{i} \sum_{k} \tau_{1ik} \Big(\sum_{j} \tau_{2jk}\, h(p, q; A_{ij}) \Big) + \sum_{i,k} \tau_{1ik} \big[\xi_{1ik} - \log \tau_{1ik}\big] + \text{const.} \\ &= \sum_{i} \sum_{k} \tau_{1ik} \Big(\sum_{j} \tau_{2jk}\, h(p, q; A_{ij}) + \xi_{1ik} - \log \tau_{1ik} \Big) + \text{const.} \end{aligned} \tag{15}$$

where const. includes terms also dependent on τ_2, but not on τ_1. The cost function above is separable over i, and for each i we have an instance of the problem given in the following lemma. Recall that $\mathcal{P}_K$ is the set of probability vectors in $\mathbb{R}^K$.

Lemma 1. *For any nonnegative vector $(a_1, \ldots, a_K)$, let $f_a : \mathcal{P}_K \to \mathbb{R}$ be defined by $f_a(p) := \sum_{k=1}^{K} p_k(a_k - \log p_k)$. Then the maximizer of f_a over $\mathcal{P}_K$ is given by the softmax operation:*

$$\operatorname*{argmax}_{p \in \mathcal{P}_k} f_a(p) = \text{softmax}(a) := \frac{e^{a_k}}{\sum_{\ell} e^{a_\ell}}. \tag{16}$$

We write the solution of Lemma 1 simply as $p_k \propto_k e^{a_k}$ where $\propto_k$ means proportional as a function of k. Then, τ_1 update is $\tau_{1ik} \propto_k \exp\big[\sum_j \tau_{2jk}\, h(p, q; A_{ij}) + \xi_{1ik}\big]$, or after unpacking ξ_{1ik},

$$\tau_{1ik} \;\propto_k\; \pi_{1k} \exp\Big[\sum_{j} \tau_{2jk}\, h(p, q; A_{ij}) + \beta_{1ik}(\widetilde{\Gamma}, \sigma^2)\Big] \quad i = 1, \ldots, N_1. \tag{17}$$

The update for τ_2 is similar.

Updating $\widetilde{\Sigma}$ and $\widetilde{\mu}$. Let us define $\bar{\tau}_{rk} := \sum_{i=1}^{N_r} \tau_{rik}$ and $D_k^{-1} := \text{diag}\,(\frac{\bar{\tau}_{1k}}{\sigma_1^2} I_{d_1}, \frac{\bar{\tau}_{2k}}{\sigma_2^2} I_{d_2})$ Then, as a function of $\widetilde{\Sigma}$, J can be written as (see Appendix B.3)

$$J(\widetilde{\Sigma}) \doteq -\frac{1}{2} \sum_{k} \text{tr}\,\big[(D_k^{-1} + \Sigma^{-1})\widetilde{\Sigma}_k\big] - \log|\widetilde{\Sigma}_k|. \tag{18}$$

This is separable over k, with each term being the likelihood of a multivariate Gaussian with covariance parameter. The maximizers are then simply $\widetilde{\Sigma}_k = (D_k^{-1} + \Sigma^{-1})^{-1}$ for $k \in [K]$.

To derive the updates for $\widetilde{\mu}$, let $\bar{x}_{rk} := \sum_{i=1}^{N_r} \tau_{rik} x_{ri}$ and $\bar{\mu}_{rk} := \bar{x}_{rk}/\bar{\tau}_{rk}$. Then, as a function of $\widetilde{\mu}$, J can be written as (see Appendix B.3):

$$J(\widetilde{\mu}) \doteq -\frac{1}{2} \sum_{k} (\widetilde{\mu}_k - m_k)^T (D_k^{-1} + \Sigma^{-1})(\widetilde{\mu}_k - m_k) \tag{19}$$

where $m_k = (D_k^{-1} + \Sigma^{-1})^{-1}(D_k^{-1}\bar{\mu}_k + \Sigma^{-1}\mu)$. It is clear that the optimal value of $\widetilde{\mu}_k$ is equal to m_k, which using the optimal value of $\widetilde{\Sigma}_k$, can be written as $\widetilde{\mu}_k = \widetilde{\Sigma}_k(D_k^{-1}\bar{\mu}_k + \Sigma^{-1}\mu)$.

Updating Σ, μ and σ^2. As a function of Σ, we have $J(\Sigma) \doteq -\frac{K}{2}\big[\log|\Sigma| + \text{tr}(\Sigma^{-1}S(\widetilde{\Gamma}))\big]$ which is the standard Gaussian likelihood, giving the optimal value $\Sigma = S(\widetilde{\Gamma}, \mu)$. Similarly, as a function of μ, $J(\mu) \doteq -\frac{K}{2}\big[\text{tr}(\Sigma^{-1}S(\widetilde{\Gamma}))\big] \doteq -\frac{1}{2}\sum_k \big[(\widetilde{\mu}_k - \mu)^T\Sigma^{-1}(\widetilde{\mu}_k - \mu)\big]$ giving the optimal solution $\mu = \frac{1}{K}\sum_k \widetilde{\mu}_k$. The update for $\sigma^2 = (\sigma_1^2, \sigma_2^2)$ can be easily obtained too (see Appendix B.4)

$$\sigma_r^2 = \frac{1}{N_r d_r}\Big[\sum_k \bar{\tau}_{rk}\,\text{tr}\,\big((\widetilde{\Sigma}_k)_{rr}\big) + \sum_{i,k} \tau_{rik}\|x_{ri} - \widetilde{\mu}_{rk}\|^2\Big], \quad r = 1, 2. \tag{20}$$

Updating π and $\Psi \equiv (p, q)$. Updating these parameters is standard (See Appendix B.4):

$$\pi_r = \frac{(\bar{\tau}_{r1}, \ldots, \bar{\tau}_{rK})}{\sum_k \bar{\tau}_{rk}},\ r = 1, 2, \quad p = \frac{\sum_{ij}\gamma_{ij}(\tau)A_{ij}}{\sum_{ij}\gamma_{ij}(\tau)}, \quad q = \frac{\sum_{ij}\big(1 - \gamma_{ij}(\tau)\big)A_{ij}}{\sum_{ij}\big(1 - \gamma_{ij}(\tau)\big)}. \tag{21}$$

3.4. Extensions

Various extensions and improvements of the basic algorithm discussed in Section 3.3 are presented in Appendix A. A detailed derivation of an algorithm for fitting the degree-corrected model is given in Appendix A.3. The algorithm employs a novel application of the Douglas–Rachford splitting algorithm for optimization over the degree inhomogeneity parameter θ. We also provide in Appendix A.2 an extension of the algorithm to the general matched SBM model (4). Improvement to the speed for the basic algorithm of Section 3.3 is discussed in Appendix A.1.1 as well as the possibility of adding a diagonal restriction on covariate covariance matrices (Appendix A.1.2).

Overall the algorithm has 16 variations based on four options: (1) Poisson versus binomial likelihood, (2) planted partition versus general edge probability matrix, (3) full versus diagonal covariances, (4) with or without degree correction. In the following we will focus on the following defaults for the first three options: A Poisson likelihood with planted partition connectivity and full covariances.

Algorithm 1 Variational block coordinate ascent for fitting `mbiSBM`

1: Initialize τ_r using `biSC`, and $\theta_r = \mathbf{1}_{N_r}$ for $r = 1, 2$. Pick tolerance $\varepsilon \in (0, 1]$.
2: Initialize $\Sigma, \widetilde{\Sigma}_k$ with $I_{d_1+d_2}$ and $\mu, \widetilde{\mu}_k$ with 0, for $k \in [K]$, and $\sigma_r^2 = 1$ for $r = 1, 2$.
3: **while** not `CONVERGED`, nor maximum iterations reached **do**
4: Update (p, q) using (26) and $\pi_r, r = 1, 2$ using (21).
5: Update $(\phi_0, \phi_1) \leftarrow (q - p, \log(p/q))$.
6: **if** DC-version **then**
7: Update θ_r by repeating (36) till convergence.
8: **end if**
9: Update $\beta_r, r = 1, 2$ using (13).
10: $\tau_r^{\text{old}} \leftarrow \tau_r$, $r = 1, 2$.
11: Update τ_1 by repeating (34) till convergence.
12: Update τ_2 by repeating (34), with subscripts 1 and 2 switched and A replaced with A^T, till convergence.
13: Update the following for for $r = 1, 2$ and $k \in [K]$: ▷ Update parameters
14: $\bar{\tau}_{rk} \leftarrow \sum_{i=1}^{N_r} \tau_{rik}$, and $D_k^{-1} \leftarrow \operatorname{diag}\left(\bar{\tau}_{1k} I_{d_1}/\sigma_1^2, \bar{\tau}_{2k} I_{d_2}/\sigma_2^2\right)$,
15: $\bar{x}_{rk} \leftarrow \sum_{i=1}^{N_r} \tau_{rik} x_{ri}$, and $\bar{\mu}_{rk} \leftarrow \bar{x}_{rk}/\bar{\tau}_{rk}$.
16: Update $\widetilde{\Sigma}_k \leftarrow (D_k^{-1} + \Sigma^{-1})^{-1}$ and $\widetilde{\mu}_k \leftarrow \widetilde{\Sigma}_k(D_k^{-1}\bar{\mu}_k + \Sigma^{-1}\mu)$.
17: Update $\mu \leftarrow \frac{1}{K}\sum_k \widetilde{\mu}_k$ and $\Sigma \leftarrow \frac{1}{K}\sum_{k=1}^K \left[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\right]$.
18: Update $\sigma_r^2, r = 1, 2$ using (27).
19: `CONVERGED` $\leftarrow \left[\max\{\delta_1, \delta_2\} < \varepsilon/K\right]$, where $\delta_r := \|\tau_r - \tau_r^{\text{old}}\|_\infty, r = 1, 2$
20: **end while**

3.5. Summary of the Algorithms

Algorithm 1 summarizes the updates for fitting the proposed matched bipartite SBM model, to which we refer as `mbiSBM`. We have stated the general form of the algorithm with degree correction (DC) and covariates. Note for example that if no degree-correction is desired, θ_r remains equal to $\mathbf{1}_{N_r}$ and the iterations (34) in steps 11 and 12, for updating the label distributions (τ_r), automatically reduce to the simple update (25) (that is, the iterations converge in one step.). There are other variations available. For example, if desired, step 5 can be replaced with $(\phi_0, \phi_1) \leftarrow (\log\frac{1-p}{1-q}, \log\frac{p(1-q)}{q(1-p)})$ to use values based on a Bernoulli likelihood instead of a Poisson. Empirically, we have not found much difference between the two. With minor modifications, the algorithm can be used when only one side has covariates or without covariates for either side. The code is available on `Github` (Razaee et al.).

3.5.1. Initialization of the Algorithms

It is known that variational inference is sensitive to initialization (Blei et al., 2017). The main component of the algorithm that needs careful initialization is the matrix of (approximate) posterior node labels $\tau = [\tau_1; \tau_2]$. We propose to initialize τ using a bipartite spectral clustering algorithm, `biSC` for short, which is a variant of the approach of Dhillon (2001). The difference

Algorithm 2 Bipartite Spectral Clustering (`biSC`)

1: Input: bi-adjacency matrix $A \in \{0, 1\}^{N_1 \times N_2}$.
2: Let $D_1 = \operatorname{diag}(\sum_j A_{ij}, i = 1, \ldots, N_1)$ and $D_2 = \operatorname{diag}(\sum_i A_{ij}, j = 1, \ldots, N_2)$.
3: Form $L = D_1^{-1/2} A D_2^{-1/2}$.
4: Let $L = USV^T$ be the SVD of L truncated to K largest singular values ($U \in \mathbb{R}^{N_1 \times K}$ and $V \in \mathbb{R}^{N_2 \times K}$).
5: Normalize each row of U and V to unit ℓ_2 norm to get $\widetilde{U}$ and $\widetilde{V}$, resp., then form

$$Z = \begin{bmatrix} D_1^{-1/2}\widetilde{U} \\ D_2^{-1/2}\widetilde{V} \end{bmatrix}.$$

6: Run k-means with K clusters on the rows of Z.

between our version and that of Dhillon (2001) is that Dhillon (2001) does not normalize the rows of the singular vectors and keeps top $\lceil \log_2 K \rceil$ singular vectors, as opposed to K. We have found that row normalization greatly improves the performance, and it is fairly standard in usual (non-bipartite) Laplacian-based spectral clustering. Algorithm 2 summarizes our version.

In simulation studies, we also consider a couple of competing initializations. One interesting choice is to use the usual Laplacian-based spectral clustering, which is oblivious to the bipartite nature of the problem. For this choice, we use the regularized version described in Amini et al. (2013) as `SCP`. Note that `SCP` will be applied to the (symmetric) adjacency matrix $\widetilde{A}$; see (5). It is also possible to regularize `biSC` using similar ideas, though surprisingly, we found the simple unregularized version of `biSC` is quite robust, and we have used this simple version when reporting results.

When working with simulated data, since we have access to the true labels, we will also consider a perturbed version of truth as an initialization. Specifically, we generate from a mixture of the true label distribution and Dirichlet noise, i.e. $\tau_{ri} = \omega z_{ri} + (1 - \omega)\varepsilon_{ri}$ where $\varepsilon_{ri} \sim \text{Dir}(.5\mathbf{1}_K)$. Here, we treat z_{ri}, the true label of node i in group r, as a distribution on the K labels. Parameter $\omega \in [0, 1]$ measures the degree of initial perturbation towards noise. For example, with $\omega = 0.1$, about 10% of the initial labels are correct. We will refer to this initialization as $\sim$`rnd`, for approximately random. This initialization will act as a proxy for a "good enough" initialization and allows us to study the behavior of our variational inference procedure decoupled from specific initializations produced by spectral clustering (or other methods).

Let us say a few words about the initialization of other parameters. The algorithm is moderately sensitive to the initialization of p, q and $\pi_r, r = 1, 2$. When the quality of the initial labels (τ_1 and τ_2) is good, one can initialize these parameters, based on (τ_r), by running the corresponding updates first, as is done in Algorithm 1, lines 4–5. This is the form we suggest in practice when using the `biSC` initialization, and is used in the real data application (Section 5). However, when the quality of the initial labels is not good, for example, when using `SCP` in the simulations, p and q obtained based on initial (τ_r) can become quite close leading to numerical instability. We have found in those cases that initializing these parameters with fixed values, say $(p, q) = (0.1, 0.01)$ and π_r set to uniform distribution of $[K]$, greatly improves the stability of the algorithm. (This is since even one iteration of the algorithm could significantly

improve upon initial labels.) This fixed initialization of (p, q, π_r), independent of τ_r is what we have used in Monte Carlo simulations on synthetic data, when comparing different label initializations (Section 4).

4. Simulations

In this section, we show that effectiveness of our proposed algorithm in recovering the true labels in synthetic bipartite networks. For the most part, we generate data from our proposed model (2)–(3). In the plots investigating the degree-corrected version of the algorithm, we generate from the degree-corrected version of the network described in subsection 2.4.

4.1. Data generation

Key parameters regarding covariate generation in (2) are (μ, Σ) for generating v_{*k}. We take $\mu = 0$ and $\Sigma = \nu I_{d_1+d_2}$ throughout. Varying ν (or dimensions d_r) changes the information provided by the covariates (Appendix E). Larger ν causes v_{*k} to be further apart, hence covariates are more informative. $\nu = 0$ corresponds to zero covariate information. We also fix covariate noise levels at $\sigma_r = 0.5$ for $r = 1, 2$, and the network size at $N = (N_1, N_2) = (200, 800)$.

Key parameters regarding network generation in (3) are p and q. We reparametrize our planted partition model in terms of expected average degree

$$\lambda = \frac{2N_1N_2}{N_1 + N_2}\big[q + (p - q)\sum_k \pi_{1k}\pi_{2k}\big] \tag{22}$$

(see Appendix D) and the out-in-ratio $\alpha = q/p \in [0, 1)$. Estimation becomes harder when λ decreases (few edges) or when α increases (communities are not well separated). We fix $\alpha = 1/7$ and vary λ in the subsequent simulations.

When generating from the degree-corrected version, we draw $(\theta_i, i \in C_{rk})$ from a Pareto (i.e., power-law) distribution, for each $k \in [K]$ and $r = 1, 2$. Real networks are frequently reported to have power-law degree distributions Barabási and Albert (1999). The Pareto(a, R) in general has density $\theta \mapsto (aR^a)\theta^{-a-1}1\{\theta > a\}$, with mean $aR/(a-1)$ for $a > 1$ and variance $R^2a/[(a-1)^2(a-2)]$ for $a > 2$. Since $|C_{rk}|^{-1}\sum_{i\in C_{rk}}\theta_i$ will be approximately equal to the mean of the Pareto, and we want this average to be 1, we have to choose $R = (a-1)/a$, that is, we generate $\theta_i \overset{\text{iid}}{\sim} \text{Pareto}(a, (a-1)/a)$ for $i \in C_{rk}$. (To comply with our model specification, we further normalize θ_i for their within-community averages to be exactly one; this will have little effect since the average is already close to 1.) The variance in this case is $[a(a-2)]^{-1}$ which is decreasing in a over $(2, \infty)$. In order to get maximum degree heterogeneity (i.e., the worse case in terms of the difficulty of fitting), we take $a = 2$, corresponding to infinite variance. We note that expression (22) remains valid for the degree-corrected case without modification, assuming normalization (7); see Appendix D.

4.2. Matched NMI for evaluation

In general, we measure the accuracy of the algorithms by the normalized mutual information (NMI) between the inferred and correct communities which is defined as the mutual information of the (empirical) joint distribution of the two label assignments divided by the joint entropy (Malvestuto, 1986). NMI has a maximum value of 1 for perfect agreement and a minimum of 0 for no agreement. One could measure NMI individually between $Z_r \in \{0, 1\}^{N_r\times K}$

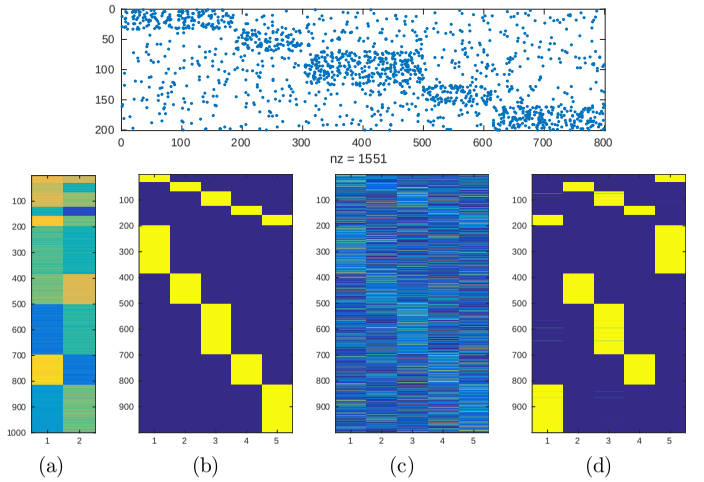


Figure 3: Typical output of the algorithm. Top row: bi-adjacency matrix. Bottom row: (a) Concatenated covariate matrix $[X_1; X_2]$. (b) Concatenated true labels $[Z_1; Z_2]$. (c) Initial labels for the algorithm, Dirichlet-perturbed truth $0.1[Z_1; Z_2] + 0.9\,\text{Dir}(0.5\mathbf{1}_K)$. (d) Concatenated output of the algorithm $[\tau_1; \tau_2]$.

(the true label matrix) and $\tau_r \in [0, 1]^{N_r\times K}$ (the estimated soft-label matrix) for each $r = 1, 2$. However, one can also measure a *matched NMI* by concatenating the labels of two sides vertically, i.e., forming $[Z_1; Z_2]$ and $[\tau_1; \tau_2]$ and measuring a single NMI between the resulting $(N_1 + N_2) \times K$ matrices. Some thought should convince the reader that this the natural way to also measure the effectiveness of the matching between the communities of the two sides: We have a matched NMI of 1, if the true and estimated clusters on each side are in perfect agreement, and the matching between them is perfectly recovered.

4.3. Typical output

Figure 3 shows the typical output of the algorithm on the data generated from the model without degree correction (DC), i.e., $a = \infty$. Here the empirical average degree is $\hat{\lambda} = 3.1$, $K = 5$, $\nu = 10$ and $d = (2, 2)$, the dimensions of the covariates. Concatenated matrices of the true labels and the initial and final labels are shown. Vertical concatenation is used as discussed earlier, giving matrices of dimension $(N_1 + N_2) \times K$. Initial labels are the Dirichlet-perturbed truth with $\rho = 0.1$, i.e. 90% noise, as discussed in Section 3.5.1. It is interesting to note that the output of the algorithm has recovered the communities with a nontrivial permutation of the community labels.

In other words, the perturbation of the initial labels is high enough that the convergence of the algorithm cannot simply be explained by a local perturbation analysis: the algorithm has not converged to the original labels, but to a perfectly valid permuted version of the original labels. That is, $\tau_r \approx Z_rQ_r$ for $r = 1, 2$ where Q_1 and Q_2 are $K \times K$ permutation matrices. The matched NMI and misclassification rate for the algorithm are 0.98 and 0.30% in this case. If

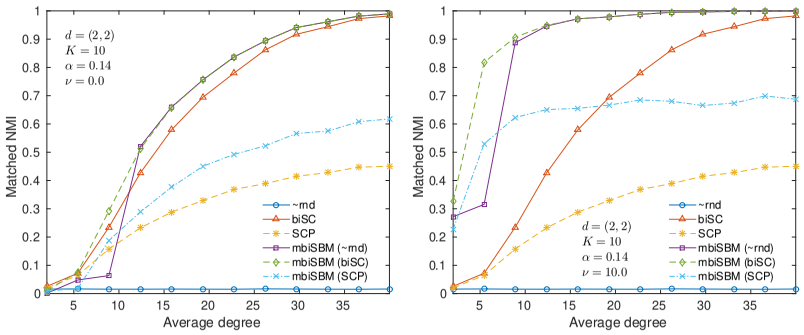


Figure 4: Effect of different initialization methods on mbiSBM with (left) $\nu = 0$ and (right) $\nu = 10$. The case $\nu = 0$ corresponds to no covariate information whereas $\nu > 0$ gives some covariates information.

one runs k-means on the concatenated matrix of covariates $[X_1; X_2]$, disregarding the network information, one gets matched NMI and misclassification rate, 0.44 and 38.50%. That is, the covariates themselves are not as informative alone as in combination with the network.

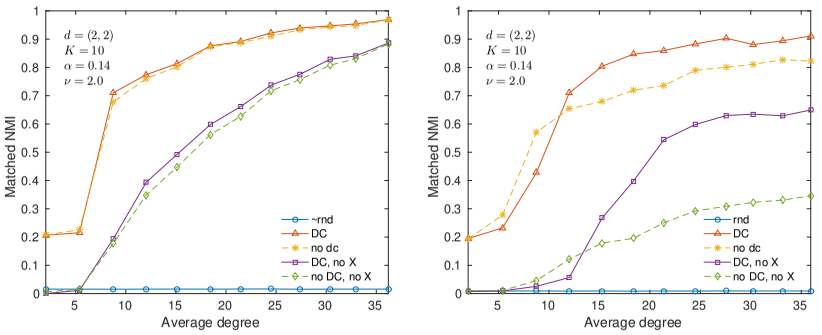
4.4. Average behavior

Figure 4 shows the mathched NMI versus average expected degree λ for various methods. The results are averaged over 50 Monte Carlo replications. Naive (regularized) spectral clustering, denoted as `SCP`, is shown in addition to `biSC` as discussed in Section 3.5.1. Moreover, the plots show our algorithm `mbiSBM`, initialized with both spectral methods and with Dirichlet-perturbed truth ($\rho = 0.1$) denoted as $\sim$`rnd`. The two plots correspond to the case with no covariate information, $\nu = 0$, and the case with covariate information $\nu = 10$. In both cases, covariate dimensions are $d = (2, 2)$, number of communities $K = 10$ and out-in-ratio is $\alpha = 1/7$. There is no degree-correction in the model or `mbiSBM` algorithm.

As can be seen, `biSC` outperforms SCP significantly. Without covariates, `mbiSBM` started with `biSC` slightly improves upon `biSC`; initializing with $\approx 10\%$ truth (`mbiSBM` ($\sim$`rnd`)) has similar performance for sufficiently large λ, showing that `mbiSBM` behaves well with any sufficiently good initialization. Note also that `mbiSBM` initialized with SCP, improves upon SCP for large λ. With covariate information ($\nu = 10$), `mbiSBM` significantly outperforms `biSC` which does not incorporate the covariates.

4.5. Effect of degree correction

Figure 5 investigates the effect of employing degree-correction in the algorithm. In both plots of the figure, we are generating from the same DC-version of the model using within-community Pareto degree distribution with parameter $a = 2$ as described earlier, $d = (2, 2)$, $K = 10$, $\alpha = 1/7$, and $\nu = 2$. The difference between the two plots is how we initialize `mbiSBM` algorithm. The left panel corresponds to "Dirichlet-perturbed truth" initialization ($\rho = 0.1$) denoted as $\sim$`rnd`, whereas the right panel corresponds to completely random initialization, denoted as `rnd`. Four versions of the algorithm are considered, with or without covariate (X) incorporation, and with or without degree-correction (DC).



(a) (b)

Figure 5: Effect of the degree correction steps 6–8 in the algorithm, for a power-law network. Data is generated from DC version of the model (subsection 2.4) with Pareto distribution $p(\theta) \propto \theta^{-3}$, $\theta > 2$ for degree parameters within each community. (Left) shows the results for a good initialization, the Dirichlet perturbed truth, denoted as $\sim$`rnd` (with $\approx 10\%$ true labels) and the (right) shows the results for a completely random initialization, denoted as `rnd`.

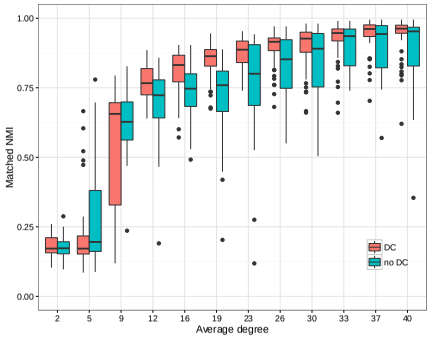


Figure 6: Effect of degree correction steps on variability, for a power-law network.

Surprisingly, as the left panel shows, with sufficiently good initialization ($\sim$`rnd`), degree-correction step of the algorithm provides only a slight improvement. However, the improvement of degree-correction is quite significant when starting from a poor initialization (`rnd`). In general, it is advisable to use the DC version since its solution has less variance. Figure 6, illustrates the algorithm with DC correction and without, in the same setup of the left panel of Figure 5, that is, both cases initialized with $\sim$`rnd` (and both incorporating covariates). Though Figure 5(a) shows that mean behaviors are close, Figure 6 shows that the distributions of the outputs are quite different, with the solution of DC version having less variability. This is expected as the DC version is solving an optimization problem with much restricted feasible region.

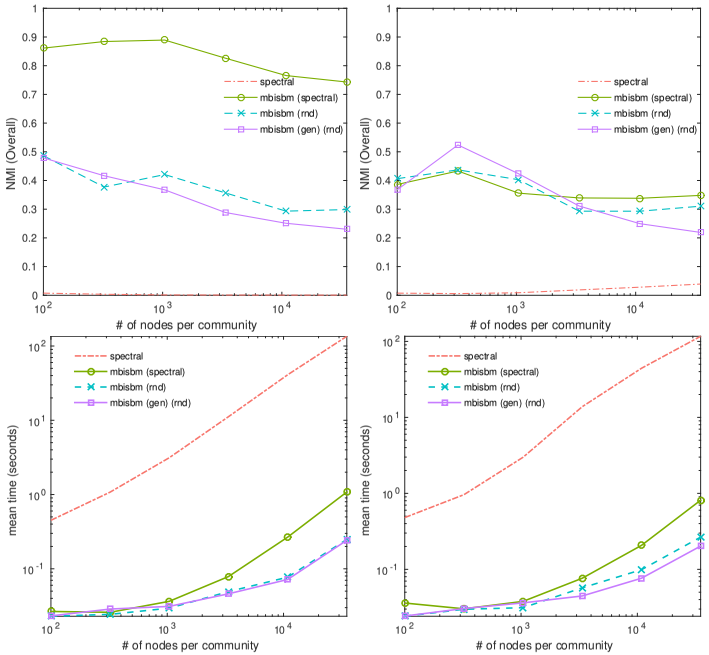


Figure 7: Scaling behavior. Both the average NMI (top) and the mean execution time (time) are shown as a function of the number of clusters per community for three-community ($K = 3$) networks. The two columns show two block models generated according to model (23): (Left) $B_0^{(1)}$ and (Right) $B_0^{(2)}$ as defined in (24).

4.6. Scaling behavior

Figure 7 illustrates how the algorithm scales to large networks compared to the spectral approach. The simulation setup for this figure is as follows: We consider a base connectivity matrix $B_0 \in \mathbb{R}_+^{K\times K}$ and let the edge probability matrix be

$$B = \frac{\log(N_1 N_2)}{\sqrt{N_1 N_2}} B_0, \quad N_1 = n_0 K,\ N_2 = [0.75 n_0 K] \tag{23}$$

where n_0 is the number of nodes per community; that is, we assume that communities of both sides are balanced and each contains n_0 nodes. We then let n_0 vary from 100 to 35000, while $K = 3$ is fixed, to study the large-scale behavior of the network. This simulation setup has recently been considered in Zhou and Amini (2018) and is suitable for studying the semi-sparse asymptotic regime. Note that the average degree of the network will be $\sim \log(n_0)$ as $n_0 \to \infty$ resulting in fairly sparse large networks. We consider two versions of B_0, corresponding to the

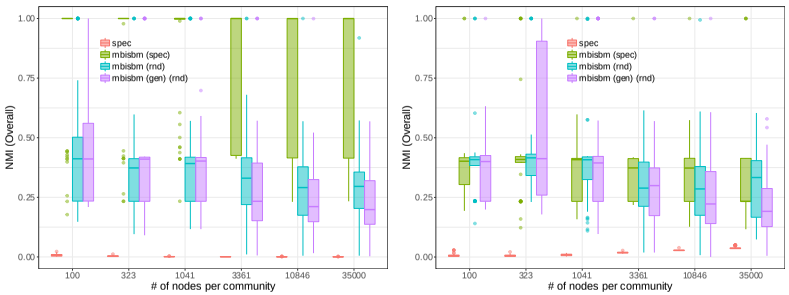


Figure 8: Boxplot of the NMI corresponding to Figure 7.

two columns in Figure 7:

$$B_0^{(1)} = \frac{1}{2}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \quad B_0^{(2)} = \frac{1}{3}\begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 1 & 3 \end{bmatrix}. \tag{24}$$

Note that $B_0^{(1)}$ gives a planted partition model whereas $B_0^{(2)}$ corresponds to the more general class of SBM given by (4), i.e., it contains both assortative and dissortative relations. We also note that for $B_0^{(2)}$, the within-community connectivities are variable (i.e., $B_0^{(2)}$ has unequal diagonal elements). Both models are quite hard as the so-called out-in-ratio is quite high; in the planted partition case $B_0^{(1)}$, for which the ratio directly characterizes hardness, we have $q/p = 0.5$. The covariates are generated similar to Section 4.4 with $\nu = 10$ and $d = (5, 5)$. The results are averaged over 50 replications.

The hardness of the two problems are reflected in the NMI plots of Figure 7 where the spectral method (`spectral`) is barely above the random assignment. In addition to the spectral method, three versions of `mbiSBM` have been shown: (1) Algorithm 1 with spectral initialization, `mbiSBM (spectral)`, (2) Algorithm 1 with random initialization, `mbiSBM (rnd)`, (3) modification of Algorithm 1 to allow for a general connectivity matrix (Section A.2), with random initialization, `mbiSBM (gen)(rnd)`.

All the three versions perform reasonably and comparably except for the `mbiSBM` with `spectral` initialization which significantly outperforms other versions on the planted partition model $B_0^{(1)}$. It is also worth nothing that the performance of `mbiSBM` with general connectivity matrix is not much better than the planted partition version (i.e., the default Algorithm 1), even on $B_0^{(2)}$ which is far from satisfying the simplifying planted partition assumption. We note that the spectral approach alone performs very poorly in both cases. (We also experimented with moderate perturbations for the spectral approach which did not result in noticeable improvement.) Figure 8 shows the box plots corresponding to the NMI plots in Figure 7, illustrating the variability of the results. It is worth noting that for planted partition model, `mbiSBM (spectral)` achieves a NMI of 1 for most replications.

The bottom row in Figure 7 illustrates the average run time of the algorithms. For `mbiSBM (spectral)` only the overhead relative the spectral initialization is considered. It is clear the family of `mbiSBM` algorithms are about an order of magnitude faster than the

`spectral` approach. We conclude that when the signal is high, `mbiSBM` with random initialization is outperforms in both speed and accuracy. When the signal is low, we can initialize with the spectral approach and then boost the performance by computationally cheap `mbiSBM` iterations.

5. Application to Real Data

5.1. Wikipedia networks

We have applied the algorithm to two wikipedia page–user networks, which we will call TopArticles and Cities. Each is a bipartite network between a collection of Wikipedia pages and the users who edited them: An edge is placed between a user and a page if the user has edited that page (at least once). In the TopArticles, the pages are selected from the top articles (based on monthly contributions) from Chinese (CN), Korean (KR) and Japanese (JP) language Wikipedia, corresponding to the period from January to October 2016. In the Cities network, the pages correspond to city names in English language Wikipedia; the cities were chosen from five countries: Unites States (US), United Kingdom (GB), Australia (AU), India (IN), Japan (JP). In both cases, on the user side, only those with IP addresses were retained. Although, not perfect, IP addresses were the only means by which we could obtain additional information about each user, esp. geo-location data. Wikipedia usage statistics were scraped from Wikimedia Statistics using code inspired by Keegan (2014). For geo-data we used both the `ggmap` R package by Kahle and Wickham (2013) and the API provided by ipapi.

In TopArticles, the true labels are the language assigned to each page and each user, that is, matched communities are specified by common language. The user language was assigned based on the dominant language of the country from which the IP address originates. In Cities network, the true labels are the country names assigned to each user based on user's IP address and assigned to each city page based on its geo-location tag. The IPs were also used to obtain latitude and longitude coordinates on each user, providing us with user covariate matrix $X_2 \in \mathbb{R}^{N_2 \times 2}$,

For TopArticles, we do not have any page covariate. For Cities, we use the geo-location data of the city (latitude and longitude) to give us the page covariate matrix $X_1 \in \mathbb{R}^{N_1 \times 2}$.

Figure 9 shows the two networks along with the true communities. Note that Cities is specially hard to cluster based only on network data due to the presence of nodes of different communities among each community (as positioned by the layout algorithm). Tables 1 and 2 show the break-down of pages/users based on community for the two networks. Also shown are the average degrees of each side of the network, as well as the overall average degree. For each of the two networks, we first obtained a 2-core, restricted to the giant component, then removed users from countries not under consideration. (If the last step created disjoint components we restricted again to the giant component. This only happened for Cities and only removed 5 nodes.)

Results on Wikipedia networks. Table 3 illustrates the result of the application of `biSC`, and the `mbiSBM` (`biSC`) algorithm with various combination of covariates. In all cases the degree-corrected (DC) version of `mbiSBM` is used. For TopArticles, without using covariates, there is no improvement on `biSC` while using X_2 gives significant boost to `mbiSBM`. For Cities, `biSC` outperforms `mbiSBM` with no covariates. One the other hand, adding X_2 or both X_1 and X_2 significantly improves the result of `mbiSBM`.

	CN	JP	KR	Total	Avg. deg.	Covariates
Pages	139	143	171	453	14.2	N/A
Users	579	695	828	2102	3.1	X_2 = user (lat.,lon.)
Total	718	838	999	2555	5	

Table 1: TopArticles page–user network

	US	IN	AU	JP	GB	Total	Avg. deg.	Covariates
Pages	267	235	182	113	59	856	10.2	X_1 = city (lat.,lon.)
Users	1054	1029	705	101	201	3090	2.8	X_2 = user (lat.,lon.)
Total	1321	1264	887	214	260	3946	4.4	

Table 2: Cities page–user network

To get a more refined understanding of the relative standing of `biSC` and `mbiSBM` (`biSC`), we have run two Monte Carlo analyses based on these real networks, one using *subsampling* and the other by adding Erdös–Rényi noise. Figure 10 shows the results when we subsample the network to retain a fraction of the nodes on each side (from 95% down to 10%). The x-axis shows the resulting overall average degree of the network at each subsampling level. The results are averaged over 50 replications and the interquartile range (IQR) is also shown as a measure of variability. The figures in Table 3 correspond to the rightmost point of these plots. (Average degrees of Cities vary in these ranges: overall $\in [0.4, 4.2]$, page $\in [1, 9.7]$ and user $\in [0.3, 2.7]$, whereas for TopArticles the ranges are: overall $\in [0.6, 4.9]$, page $\in [1.6, 13.8]$ and user $\in [0.3, 3]$.)

The plots show that `mbiSBM` (`biSC`) with covariates outperforms `biSC`, and the improvement is quite significant the sparser the network becomes. Note for example that in Cities adding X_1 does not have much effect in the original network, however, there is a considerable improvement when average degree starts to drop under subsampling. In the Cities case, the two covariates X_1 and X_2 together are quite strong leading to an NMI ≈ 1 and masking the effect of the network to some extent.

However, by looking at cases where only one of X_1 and X_2 is present, we observe that `mbiSBM` (`biSC`) manges to pass the covariate information via the network to the side without covariates, thus improving matched NMI significantly. To see this, consider for example the TopArticles, where only X_2 is present. In this case, even if a method could cluster X_2 perfectly and, in the absence of network information randomly guessed the labels of the other side, the NMI would be 0.42. That in Figure 10(b), the NMI starts at 0.98 and remains much

Network	`biSC`	`mbiSBM` (`biSC`), DC			
		X_1 & X_2	X_1	X_2	no X
TopArticles	0.86	-	-	0.98	0.86
Cities	0.6	1.0	0.59	0.85	0.47

Table 3: Matched NMI for `biSC` and `mbiSBM` on the two Wikipedia networks.

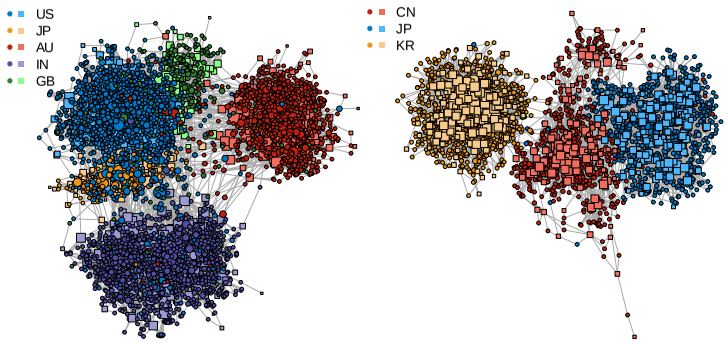


Figure 9: The two Wikipedia networks: (left) Cities (right) TopArticles. Nodes are colored according to true communities. Pages are denoted with squares and users with circles. Node sizes are proportional to log-degrees.

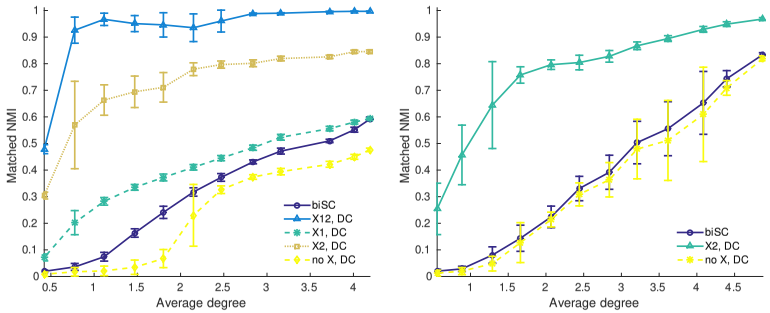


Figure 10: Effect of subsampling on Wikipedia networks: (left) Cities (right) TopArticles.

above 0.42 for most of the range of subsampling illustrates the ability of `mbiSBM` to effectively utilize both covariate and network information to correctly infer the labels of the other side. The same can be observed in the case of Cities. Finally, we note that without covariates, `biSC` usually performs better. We expect this since it is hard for local methods starting from `biSC` to improve upon it. The strength comes when we use the covariate information.

Figure 11 shows another experiment where we added Erdös–Rényi noise of average degree from 0 to 10. Again, the advantage gained by `mbiSBM` from using covariates can be quite clearly observed when one increases the noise. The covariates mitigate the effect of noise and lead to a much graceful degradation of performance for `mbiSBM` relative to `biSC`.

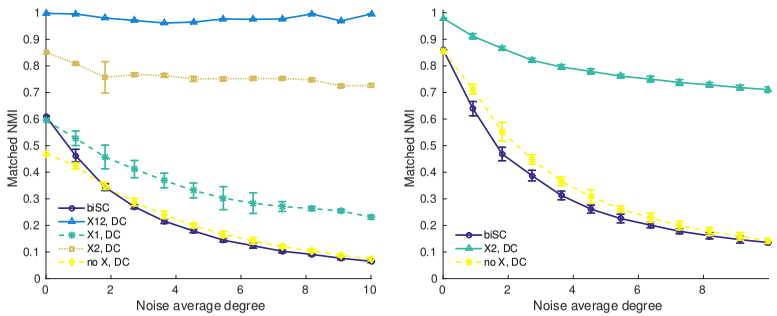


Figure 11: Effect of adding Erdös–Rényi noise on Wikipedia networks: (left) Cities (right) TopArticles.

5.2. Author-Paper networks

In this subsection, we use real data examples to showcase the performance of our algorithm when the the network is large, has high dimensional covariates and under model misspecification, that is, when the covariates are discrete.

We have applied the algorithm to two sets of author-paper networks. The first set represents papers and their authors in different sub-domains of computer science (CS): (1) data mining, (2) medical informatics, (3) visualization , (4) database, and (5) theory. The data for the first set was extracted from the Arnetminer collection, based on papers published from 1990 to 2005 in certain CS venues by Tang et al. (2012). The second set of data was scraped from DBLP, using the papers published in bioinformatics (BI), biomedical/medical informatics (BM) and computational neuroscience (CN) venues from 1995 to 2018. Each set contains various subnetworks of the two basic bipartite network described above.

In both sets, we first consider the paper-author bipartite networks with covariates being high-dimensional vectors of word frequencies appearing in the title of the papers. To deal with the binary nature of the covariates, we take their z-scores. Although the Gaussian assumption about these transformed covariates does not hold, we have found that such transformation allows us to apply our model to discrete covariates with minimal modification and with good empirical performance as discussed below. A more principled approach to dealing with discrete covariates would be to model them using a multinomial likelihood (cf. Section 6). The process of building the network and extracting the covariates is done by restricting to the giant component of the author-paper network such that each paper is associated with at least one word after removing sparse words. For the second set, we also removed some non-informative words to slightly boost the covariate signals.

These datasets have an interesting feature: They can be viewed as tri-partite author-paper-word networks. Thus, although not intuitive, we can also view the paper-word frequency matrix as a bipartite network and use the (binary) author vectors as covariates for the papers. Taking z-scores again bring these binary covariates within our framework.

For these datasets, the ground truth labels are only available for the papers: we treat their venues (i.e., their main topic) as the true community of the paper. As a result, we can only find the NMI for the paper labels. Note that assuming a matching between authors and

	Dimensions			Avg. deg.	
	authors	papers	words	authors	papers
CS12345	5126	8500	1089	4.75	2.87
BM-BI-CN	10561	8482	1649	3.4	4.23

Table 4: Statistics on the author–paper networks with words as covariates

Network	X_2	`biSC`			`mbiSBM (biSC)`			`mbiSBM (rnd)`	
		$\alpha=0$	$\alpha=1$	$\alpha=10$	$\alpha=0$	$\alpha=1$	$\alpha=10$	mean	max
CS12345	0.23	0.29	0.28	0.10	0.23	**0.36**	0.34	0.16	0.23
CS2345	0.30	0.39	0.34	0.15	0.28	**0.45**	0.38	0.13	0.26
CS345	0.38	0.35	0.45	0.09	0.21	**0.62**	0.35	0.14	0.29
CS45	0.35	0.00	**0.76**	0.04	0.00	0.73	0.71	0.02	0.08
BM-BI-CN	0.32	0.00	0.01	0.02	0.00	0.01	0.04	0.23	**0.35**
BI-CN	0.36	0.32	0.00	0.00	**0.55**	0.00	0.00	0.15	0.37
BM-BI	0.00	0.00	0.00	0.04	0.08	0.00	**0.27**	0.17	**0.27**

Table 5: NMI for `biSC` and `mbiSBM` on the Author-Paper networks.

papers is reasonable as most authors have a theme; similarly a matching between words and papers is plausible, as previously has been considered in the text mining applications by Dhillon (2001). Table 4 shows the break-down of authors/papers/words for CS12345 and BM-BI-CN networks. Also shown are the average degrees of each side of the networks.

CS12345 refers to the the dataset including all the five topics. CS2345 refers to the dataset excluding the first topic (data mining), and so on. Note that CS2345, for instance, is not necessarily an induced subnetwork of CS12345 as we take the giant component each time to make the networks for each set of topics.

Results on paper-author-word networks. Tables 5 and 6 illustrate the results for various algorithms. The first column from the left is the NMI obtained by applying k-means on the truncated SVD of the covariates (X_2 column); this approach captures the covariate-only information. The next three columns show the result for the application of `biSC` with various levels of perturbation α. Here, we are adding the constant perturbation $\alpha(N_1N_2)^{-1/2}$ to every entry of the adjacency matrix before forming the Laplacian. This is similar to the approach of Amini et al. (2013) for unipartite setups and is known to have a regularization effect on the spectral clustering, especially for sparse networks. Note that the case $\alpha = 0$ corresponds to the unregularized version as shown in Algorithm 2.

The remaining columns show the performance of `mbiSBM` initialized with each of `biSC` results, collected under `mbiSBM (biSC)` columns, as well as `mbiSBM` initialized at random, where both the mean and the maximum achieved on 25 random initializations is recorded, under `mbiSBM (rnd)` columns.

The results show that `mbiSBM` outperforms covariate-only clustering in all cases and outperforms `biSC` in 12 out of 14 datasets, indicating that utilizing covariates as well as network structure can boost the signals compared to using each source of the information alone. It is

Network	X_2	`biSC`			`mbiSBM (biSC)`			`mbiSBM (rnd)`	
		$\alpha=0$	$\alpha=1$	$\alpha=10$	$\alpha=0$	$\alpha=1$	$\alpha=10$	mean	max
CS12345	0.00	0.26	0.26	0.24	0.43	**0.45**	0.45	0.03	0.11
CS2345	0.00	0.27	0.28	0.28	0.52	0.52	**0.54**	0.02	0.20
CS345	0.00	0.24	0.39	0.37	0.74	0.74	**0.76**	0.02	0.29
CS45	0.00	0.37	0.38	0.36	0.70	0.70	**0.71**	0.01	0.02
BM-BI-CN	0.00	0.00	0.31	0.35	0.00	**0.45**	0.43	0.01	0.03
BI-CN	0.00	0.01	0.02	0.36	0.00	0.00	**0.68**	0.00	0.01
BM-BI	0.00	0.00	0.01	**0.26**	0.00	0.00	0.17	0.00	0.00

Table 6: NMI for `biSC` and `mbiSBM` on the Word-Paper networks.

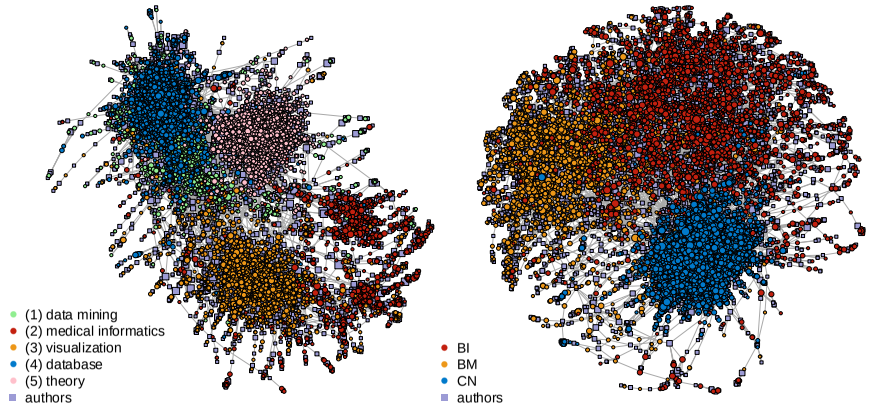


Figure 12: The two paper-author networks: (left) CS12345 (right) BM-BI-CN. Nodes corresponding to the papers are colored according to their true communities. The authors are denoted with squares and the papers with circles. Node sizes are proportional to log-degrees.

also worth noting that perturbation often boosts the performance of `biSC`, but not always. In cases where a version of `biSC` works well, `mbiSBM` often boosts the results; this is especially pronounced for the word-paper networks. Overall random initialization of `mbiSBM` performs respectably and surprisingly in some outperforms `biSC` and `biSC`-initialized `mbiSBM`.

6. Discussion

In this paper, we considered the problem of matched community detection in the bipartite setting, where one assumes a latent one-to-one correspondence between communities of the two sides. This matching is built into the model and inferred simultaneously with the communities in the process of fitting the model. Our model is an extension of the stochastic block model (SBM) and its degree-corrected version (DC-SBM). We extended our proposed model

to allow for the presence of node covariates that are aware of the matching between communities: Covariates corresponding to nodes in matched communities are statistically linked using hierarchical Bayesian modeling ideas.

Although we only considered Gaussian distributions in generating covariates, our hierarchical mixture approach has the potential for extension to more general settings. For example, one can easily model discrete covariates, such as word counts in documents, as mixtures of multinomial distributions. Some care however is needed when deciding the distribution of the top layer if one wants to allow for information sharing among the lower level variables (i.e., v_{rk}) from the two sides. In addition, as mentioned (cf. Section 2.5), our current statistical linkage carries weak information about the matching and it would be interesting to design models in which the degree of covariate information about the matching can be tuned more effectively.

Our model has natural extensions to r-partite ($r > 2$) networks where some of the modes may or may not have node covariates. We note that the general r-partite case is related to the so-called multilayer or multiplex community detection problem (Kivela et al., 2014). Finally, one would like to allow for edge covariates to accommodate many cases in real data, where edges are annotated in some way, say by the ratings as in recommender systems, by time-stamps as in our Wikipedia user-page examples, and so on. Poisson model of edge generation can, to some extent, take simple edge weights into account. Whether one can go beyond that in modeling more complex edge information is an interesting avenue for future work.

Acknowledgment

We would like to thank the referees and Associate Editor for their helpful comments that improved this paper and Nan Deng for helpful discussions. This work used computational and storage services associated with the Hoffman2 Shared Cluster provided by UCLA Institute for Digital Research and Education's Research Technology Group.

Appendix A. Extensions and improvements

A.1. Speed improvement and diagonal restriction

A.1.1. Improving the Speed

For $r = 1, 2$, we treat each $(\tau_{rik})_{ik}$ as a matrix $\tau_r \in [0,1]^{N_r \times K}$. The τ-update in (17) can be simplified to improve computational complexity for sparse networks A. We can write $h(p, q; \alpha) = \alpha\phi_1 + \phi_0$, where for the binary likelihood, $\phi_1 = \log \frac{p(1-q)}{q(1-p)}$ and $\phi_0 = \log \frac{1-p}{1-q}$, and for the Poisson likelihood considered in Section A.3 below, $\phi_0 = q - p$ and $\phi_1 = \log(p/q)$. Then, in matrix notation

$$\tau_{1ik} \ \propto_k \ \pi_{1k} \exp\Big(\phi_1 [A\tau_2]_{ik} + \phi_0 \bar{\tau}_{2k} + \beta_{1ik}(\widetilde{\Gamma}, \sigma^2)\Big) \quad i = 1, \dots, N_1$$

where $[A\tau_2]_{ik} = \sum_j \tau_{2jk} A_{ij}$ and we recall $\bar{\tau}_{2k} := \sum_{i=1}^{N_2} \tau_{2ik}$. When A is sparse, the matrix-vector product $A\tau_2$ can be computed quite fast. Letting $\beta_r = (\beta_{rik})_{ik} \in \mathbb{R}^{N_r \times K}$, we have the τ-update in vector form:

$$\tau_1 = \texttt{row-softmax}\big[\phi_1 A\tau_2 + \phi_0 \mathbf{1}_{N_1} (\bar{\tau}_2 + \log \pi_1)^T + \beta_1\big] \tag{25}$$

and similarly for τ_2. Here, `row-softmax` is the row-wise softmax operator, applying (16) to each row of a matrix.

Further improvements are possible in estimating p and q. Note that we can write $\bar{\tau}_{rk} := [\tau_r^T \mathbf{1}_{N_r}]_k$. Let us treat $\bar{\tau}_r$ as a K-vector, with elements $\bar{\tau}_{rk}$. Then, we have $\sum_{ij} \gamma_{ij}(\tau) A_{ij} = \mathrm{tr}(\tau_1^T A \tau_2)$ and $\sum_{ij} \gamma_{ij}(\tau) = \langle \bar{\tau}_1, \bar{\tau}_2 \rangle$, and

$$p = \frac{\mathrm{tr}(\tau_1^T A \tau_2)}{\langle \bar{\tau}_1, \bar{\tau}_2 \rangle}, \quad q = \frac{\delta\rho - p}{\delta - 1}, \quad \text{where } \frac{1}{\delta} = \langle \frac{\bar{\tau}_1}{N_1}, \frac{\bar{\tau}_2}{N_2} \rangle \text{ and } \rho = \frac{1}{N_1 N_2} \sum_{ij} A_{ij}. \tag{26}$$

Note that ρ is the density of the graph (or A) and that $\langle \bar{\tau}_1, \bar{\tau}_2 \rangle = \mathrm{tr}(\tau_1^T E \tau_2)$ where E is the all-ones matrix of appropriate dimension. Finally, let us define $\beta'_{rik} = \mathrm{tr}\left((\widetilde{\Sigma}_k)_{rr}\right) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2$ noting that $\beta_{rik} = \frac{1}{2\sigma_r^2} \beta'_{rik}$ from definition (13). Letting $\beta'_r = (\beta'_{rik})_{ik}$ be its matrix form, we can write the update for σ_r^2 compactly as

$$\sigma_r^2 = \frac{1}{N_r d_r} \sum_{ik} \tau_{rik} \beta'_{rik} = \frac{1}{N_r d_r} \mathrm{tr}(\tau_r^T \beta'_r). \tag{27}$$

A.1.2. Diagonal covariance restriction

For ultra high-dimensional covariates, one can restrict the covariance matrix Σ to be diagonal, which greatly improves the speed of the algorithm. This choice is also reasonable from a statistical perspective since in high-dimensions, without additional restrictions (such as sparsity), estimates of a full-dimensional covariance matrix are unreliable. Under the diagonal restriction, one only needs to make a minor modification to the algorithm: Recalling the variational likelihood $J(\Sigma) \doteq -\frac{K}{2}\big[\log|\Sigma| + \mathrm{tr}(\Sigma^{-1} S(\widetilde{\Gamma}))\big]$, the maximizer over Σ under the diagonal constraint is $\Sigma_{ii} = [S(\widetilde{\Gamma}))]_{ii}$ and $\Sigma_{ij} = 0$ for $i \neq j$, or compactly

$$\Sigma = \mathrm{ddiag}(S(\widetilde{\Gamma})),$$

where $A \mapsto \mathrm{ddiag}(A)$ is an operator that take a square matrix A and outputs a diagonal matrix with the same diagonal as A. We note that the update for $\widetilde{\Sigma}_k$ goes as before: $\widetilde{\Sigma}_k = (D_k^{-1} + \Sigma^{-1})^{-1}$ for $k \in [K]$. Since both D and Σ are diagonal, $\widetilde{\Sigma}_k$ will be diagonal as well. Thus, all the covariance matrices throughout the algorithm will remain diagonal which improves the speed and scalability.

A.2. Extension to general SBM

Let us now briefly discuss the changes needed for fitting the general SBM model (4). We only consider the case with no degree correction. Let us derive the label updates first. Using the general form of the network likelihood (8), the variational likelihood in (15) is replaced with

$$J = \sum_i \sum_k \tau_{1ik} \Big(\sum_j \sum_\ell \tau_{2j\ell}\, g(\Psi_{k\ell}, A_{ij}) + \xi_{1ik} - \log \tau_{1ik} \Big) + \text{const.} \tag{28}$$

which by the same argument used for (17), gives the following τ_{1ik} update

$$\tau_{1ik} \ \propto_k \ \pi_{1k} \exp\Big[\sum_j \sum_\ell \tau_{2j\ell}\, g(\Psi_{k\ell}, A_{ij}) + \beta_{1ik}(\widetilde{\Gamma}, \sigma^2) \Big] \quad i = 1, \dots, N_1. \tag{29}$$

The update for τ_2 follows similarly. The only other modification to the algorithm is the update for the edge probabilities Ψ, replacing the (p, q) updates. For Ψ updates, we need to maximize

$$\mathbb{E}_q[\ell_1(\Psi)] = \sum_{ij}\sum_{k\ell} \tau_{1ik}\tau_{2j\ell}\; g(\Psi_{k\ell}, A_{ij})$$

over Ψ. Assume that $g(p, \alpha) = \alpha\phi_1(p) + \phi_0(p)$. For any fixed (k, ℓ), we need to maximize

$$\Big(\sum_{ij} \tau_{1ik}\tau_{2j\ell}A_{ij}\Big)\phi_1(\Psi_{k\ell}) + \bar\tau_{1k}\bar\tau_{2\ell}\,\phi_0(\Psi_{k\ell})$$

where we have used $\bar\tau_{rk} := \sum_i \tau_{rik}$. The problem reduces to maximizing $p \mapsto a\phi_1(p) + b\phi_0(p)$ for some positive $a, b \in \mathbb{R}$. It is not hard to see that the maximizer is the same for both the Bernoulli likelihood ($\phi_1(p) = \log[p/(1-p)]$, $\phi_0 p) = \log(1-p)$) and the Poisson ($\phi_1(p) = \log p$, $\phi_0(p) = -p$), and is equal to $p^* = a/b$. This give the following Ψ-update

$$\Psi_{k\ell} = \frac{1}{\bar\tau_{1k}\bar\tau_{2\ell}}\sum_{ij} \tau_{1ik}\tau_{2j\ell}A_{ij}. \tag{30}$$

Speeding up the updates. Using the notation introduced for g, we have

$$\sum_j\sum_\ell \tau_{2j\ell}\, g(\Psi_{k\ell}, A_{ij}) = \sum_\ell \Big(\sum_j \tau_{2j\ell}A_{ij}\Big)\phi_1(\Psi_{k\ell}) + \sum_\ell \bar\tau_{2\ell}\,\phi_0(\Psi_{k\ell}).$$

Let $\Phi_1, \Phi_0 \in \mathbb{R}^{K\times K}$ be matrices with entries $[\Phi_s]_{k\ell} = \phi_s(\Psi_{k\ell})$ for $s = 0, 1$. Then, τ_1 update is

$$\tau_{1ik} \;\propto_k\; \pi_{1k}\exp\Big([A\tau_2\Phi_1^T]_{ik} + [\Phi_0\bar\tau_2]_k + \beta_{1ik}(\widetilde\Gamma, \sigma^2)\Big) \quad i = 1, \dots, N_1$$

or in matrix form (similar to (25))

$$\tau_1 = \texttt{row-softmax}\big(A\tau_2\Phi_1^T + \mathbf{1}_{N_1}[\Phi_0\bar\tau_2 + \log\pi_1]^T + \beta_1\big). \tag{31}$$

Here $\bar\tau_2 = (\bar\tau_{2\ell}) \in \mathbb{R}^K$ is viewed as a column vector.

A.3. Extension to the Degree-Corrected Case

In this case the network-dependent part of the likelihood is replaced with

$$\ell_1(\Psi, \theta) = \sum_{ij}\sum_{k\ell} z_{1ik}z_{2j\ell}\; g(\theta_{1i}\theta_{2j}\Psi_{k\ell}, A_{ij}).$$

Again, we focus on the case where $\Psi_{kk} = p$ and $\Psi_{k\ell} = q$ for $k \neq \ell$. Recalling the notation $h(p, q, \alpha) = g(p, \alpha) - g(q, \alpha)$, we have

$$\ell_1(\Psi, \theta) = \sum_{ij}\big[y_{ij}\, h\big(p\theta_{1i}\theta_{2j},\, q\theta_{1i}\theta_{2j},\, A_{ij}\big) + g\big(\theta_{1i}\theta_{2j}q, A_{ij}\big)\big]. \tag{32}$$

We assume a Poisson log-likelihood with $g(p, \alpha) = \alpha\log p - p$ for which $h(p, q, \alpha) = \alpha\log(p/q) + q - p$. We also recall the normalization assumption (7), $\sum_i \theta_{ri}z_{rik} = \sum_i z_{rik}$, which implies

$\sum_{ij} y_{ij}\theta_{1i}\theta_{2j} = \sum_{ij} y_{ij}$ and $\sum_{ij}\theta_{1i}\theta_{2j} = N_1N_2$. Using these two implications, the first term in (32) simplifies to

$$\sum_{ij} y_{ij}\big[(q-p)\theta_{1i}\theta_{2j} + A_{ij}\log(p/q)\big] + \sum_{ij}\big[-\theta_{1i}\theta_{2j}q + A_{ij}\log(\theta_{1i}\theta_{2j}q)\big]$$
$$= \sum_{ij} y_{ij}\big[(q-p) + A_{ij}\log(p/q)\big] - qN_1N_2 + \sum_{ij} A_{ij}\log(\theta_{1i}\theta_{2j}q)$$

Let $\phi_0 = q - p$ and $\phi_1 = \log(p/q)$.

τ-update. Let us fix θ and obtain updates for the label posteriors τ. Taking expectations of the objective and the constraints, the τ-portion of the update is equivalent to maximizing

$$\sum_{ij}\gamma_{ij}(\tau)\big[\phi_0 + A_{ij}\phi_1\big] + \sum_{rik}\tau_{rik}\big[\beta_{rik}(\widetilde\Gamma, \sigma^2) + \log\frac{\pi_{rk}}{\tau_{rik}}\big]$$

subject to constraints $\sum_i \tau_{rik}(\theta_{ri} - 1) = 0$ for all k. Note that these constraints follow by taking expectations of the normalization constraints (7) under $Z \sim q$. Focusing on updating τ_{1k}, we have the following optimization problem:

$$\begin{array}{ll} \max\limits_{\tau_1} & \sum_{i,k}\tau_{1ik}\big(\sum_j \tau_{2jk}\big[\phi_0 + A_{ij}\phi_1\big] + \xi_{1ik} - \log\tau_{1ik}\big) \\ \text{subject to} & \sum_i \tau_{1ik}(\theta_{1i} - 1) = 0, \quad \sum_k \tau_{1ik} = 1, \quad \tau_{1ik} \geq 0 \end{array} \tag{33}$$

where $\xi_{1ik} = \beta_{1ik} + \log\pi_{1k}$ as before. In Appendix C.1, we derive a dual ascent algorithm for solving this problem with the following updates:

$$\begin{aligned} \tau_1(\lambda) &= \texttt{row-softmax}\big[\phi_1 A\tau_2 + \phi_0\mathbf{1}_{N_1}(\bar\tau_2 + \log\pi_1)^T + \beta_1 + (\theta_1 - \mathbf{1})\lambda^T\big], \\ \lambda^+ &= \lambda - \mu[\tau_1(\lambda)]^T(\theta_1 - \mathbf{1}). \end{aligned} \tag{34}$$

Here, $\lambda \in \mathbb{R}^K$ is the dual variable, λ^+ is its update, $\theta_1 = (\theta_{1i}) \in \mathbb{R}^n$, and μ is a proper step-size. These two iterations are repeated till convergence, before updating other parameters. Note that when $\theta_1 = \mathbf{1}$, the dual ascent algorithm reduces to the single step of (25) obtained for the case without degree correction.

θ-update. Let us now fix τ and the rest of the parameters and optimize over θ. The relevant portion of the objective function is

$$\sum_{ij}\gamma_{ij}(\tau)\big[q - p + A_{ij}\log(p/q)\big] - qN_1N_2 + \sum_{ij} A_{ij}\log(\theta_{1i}\theta_{2j}q).$$

Consider optimizing over (θ_{1i}), which is equivalent to maximizing $\sum_i d_{1i}\log\theta_{1i}$, subject to $\sum_i \tau_{1ik}(\theta_{1i} - 1) = 0$ for all k, and $\theta_{1i} \geq 0$ for all i. This problem is suitable for an application of the Douglas–Rachford (DR) splitting algorithm (Douglas and Rachford, 1956; O'Connor and Vandenberghe, 2014). Let $f_t(\cdot; d) : \mathbb{R}^n_+ \to \mathbb{R}^n_+$ with $d \in \mathbb{R}^n_+$ and $t > 0$, be defined by

$$[f_t(x; d)]_i := \frac{1}{2}\big[x_i + \sqrt{x_i^2 + 4td_i}\big]. \tag{35}$$

Also, let $H_1 := \tau_1(\tau_1^T\tau_1)^{-1}\tau_1^T$ be the projection operator onto the span of $\tau_1 \in \mathbb{R}^{N_1\times K}$. The algorithm performs the following iterations for updating (ξ_1, θ_1) to (ξ_1^+, θ_1^+):

$$\begin{aligned}\theta_1^+ &= f_t(\xi_1; d_1)\\ \xi_1^+ &= \theta_1^+ - H_1(2\theta_1^+ - \xi_1 - \mathbf{1})\end{aligned} \tag{36}$$

where $\xi_1 \in \mathbb{R}^{N_1}$ is an auxiliary variable, $d_1 = (d_{1i}) \in \mathbb{R}^{N_1}$ collects the degrees of side 1, and $t > 0$ is the fixed parameter of DR algorithm (often set to 1). The details for the derivation of this algorithm can be found in Appendix C.2. The same updates apply to θ_2, replacing subscript 1 with 2.

Remark 3. Note that if $\tau_1 = (\tau_{1ik})$ was a hard label assignment, then the optimization for (θ_{1i}) would have a simple solution. To see this, let $C_k(\tau_1)$ be the kth cluster of hard label τ_1. Then, the optimal value of θ_1 is given by

$$\theta_{1i} = \frac{d_{1i}}{\sum_{i'\in C_k(\tau_1)} d_{1i'}}, \quad \text{for } i \in C_{1k}(\tau_1).$$

This is in fact, the choice in profile-likelihood approaches to fitting DC-SBM, where one replaced θ_1 with this optimal value, in addition to optimal values of edge probabilities and class priors, all in terms of $\{C_{1k}(\tau_1)\}$, and then optimize the resulting profile likelihood over $\{C_{1k}(\tau_1)\}$. See for example (Karrer and Newman, 2011). Our approach here, allows us to keep a soft-label assignment τ_1 throughout the algorithm, viewing optimization over θ_1 as another phase of block-coordinate ascent for the overall constrained optimization problem.

(p, q)-update. To optimize over p and q we note that because of the Poisson model, p and q are not tied together and the only constraint we have is $p, q \geq 0$. Optimizing over p is equivalent to maximizing $-p\sum_{ij}\gamma_{ij} + \log p\sum_{ij}\gamma_{ij}A_{ij}$ and optimizing over q, is equivalent to maximizing over $-q\sum_{ij}(1-\gamma_{ij}) + \log q\sum_{ij}(1-\gamma_{ij})A_{ij}$, both giving the same updates as those in (21).

Appendix B. Details of Section 3

B.1. Proof of Lemma 1

We have $f_a(p) = -\sum_k p_k \log(p_k/e^{a_k})$. If $\sum_k e^{a_k} = 1$, then $-f_a(p)$ is the KL-divergence between (p_k) and (e^{a_k}) and the result follows. Otherwise, normalizing only adds a constant to f_a, that is, with $C = 1/\sum_k e^{a_k}$ and $q_k = Ce^{a_k}$, we have $f_a(p) = -\sum_k p_k \log(p_k/q_k) - \log C$ and the result follows.

B.2. Derivation of (12)

We write $\mathbb{E}_q$ for expectation w.r.t. the above joint distribution on (Z, V). Similarly, we write $\mathbb{E}_{q_Z}$ and $\mathbb{E}_{q_V}$ for the expectation (or integration) w.r.t. to each of q_Z and q_V. Note that $\mathbb{E}_q[\cdot] = \mathbb{E}_{q_V}\mathbb{E}_{q_Z}[\cdot]$. Plugging in the variational distribution (11) into the variational likelihood (10). we have

$$J = \mathbb{E}_q\Big[\ell(\mu, \Sigma, \sigma, Q)] - \log q(Z, V)\Big] = T_1 + T_2 + T_3 - T_4 - T_5$$

where

$$T_1 = \mathbb{E}_{q_Z}\Big[\sum_{i,j}\psi(A_{ij}, y_{ij})\Big], \quad T_2 = \mathbb{E}_{q_V}\mathbb{E}_{q_Z}\Big[\sum_{r,i,k} z_{rik}\big[\log f_r(x_{ri}; v_{rk}) + \log\pi_{rk}\big]\Big]$$
$$T_3 = \mathbb{E}_{q_V}\Big[\sum_k \log p(v_{*k}|\mu, \Sigma)\Big], \quad T_4 = \mathbb{E}_{q_Z}\log q(Z), \quad T_5 = \mathbb{E}_{q_V}\log q(V)$$

Let $\gamma_{ij}(\tau) := \mathbb{E}_{q_Z}(y_{ij}) = \sum_{k=1}^K \tau_{1ik}\tau_{2jk}$, so that

$$T_1 = \sum_{i,j}\Big[\gamma_{ij}(\tau)\log(p^{A_{ij}}(1-p)^{1-A_{ij}}) + (1-\gamma_{ij}(\tau))\log(q^{A_{ij}}(1-q)^{1-A_{ij}})\Big] \tag{37}$$

We frequently use the following elementary result in the sequel. Let $x \mapsto N(x; \mu, \Sigma)$ be the PDF of the multivariate normal distribution with mean μ and covariance Σ.

Lemma 2. *Let ε be a random vector with mean $\widetilde{\mu}$ and covariance $\widetilde{\Sigma}$, and x a nonrandom vector. Then,*

$$\mathbb{E}[\varepsilon^T\Lambda\varepsilon] = \mathrm{tr}[\Lambda\widetilde{\Sigma}] + \widetilde{\mu}^T\Lambda\widetilde{\mu}, \tag{38}$$
$$\begin{aligned}\mathbb{E}[\log N(x; \varepsilon, \Sigma)] = \mathbb{E}[\log N(\varepsilon; x, \Sigma)] &= -\frac{1}{2}\Big\{\log|\Sigma| + (x-\widetilde{\mu})^T\Sigma^{-1}(x-\widetilde{\mu}) + \mathrm{tr}(\Sigma^{-1}\widetilde{\Sigma})\Big\}\\ &= -\frac{1}{2}\Big\{\log|\Sigma| + \mathrm{tr}(\Sigma^{-1}\Psi)\Big\},\end{aligned} \tag{39}$$

where $\Psi = \widetilde{\Sigma} + (x-\widetilde{\mu})(x-\widetilde{\mu})^T$.

Proof Let us prove (39). We have $\log N(x; \varepsilon, \Sigma) = -\frac{1}{2}\log|\Sigma| - \frac{1}{2}(x-\varepsilon)^T\Sigma^{-1}(x-\varepsilon)$. Noting that $x - \varepsilon$ has mean $x - \widetilde{\mu}$ and covariance $\widetilde{\Sigma}$ and applying (38) gives the desired result. ■

Recall that $f_r(x_{ri}; v_{rk}) = N(x_{ri}; v_{rk}, \sigma_r^2 I_{d_r})$, and note that under q_V, v_{rk} has mean $\widetilde{\mu}_{rk}$ and covariance $(\widetilde{\Sigma}_k)_{rr}$. Note that we are partitioning $\widetilde{\Sigma}_k$ into four blocks of sizes $\{d_1, d_2\}\times\{d_1, d_2\}$ and $(\widetilde{\Sigma}_k)_{rr}, r = 1, 2$ correspond to the two diagonal blocks in this partition. Using Lemma 2, we have

$$\begin{aligned}T_2 &= \mathbb{E}_{q_V}\Big\{\sum_{r,i,k}\tau_{rik}\big[\log N(x_{ri}; v_{rk}, \sigma_r^2 I_{d_r}) + \log\pi_{rk}\big]\Big\}\\ &= \sum_{r,i,k}\tau_{rik}\big[-\frac{d_r}{2}\log\sigma_r^2 - \frac{\mathrm{tr}\left((\widetilde{\Sigma}_k)_{rr}\right) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2}{2\sigma_r^2} + \log\pi_{rk}\big].\end{aligned}$$

Recall that $\widetilde{\Gamma} := ((\widetilde{\Sigma}_k, \widetilde{\mu}_k), k = 1, \ldots, K)$ and $S(\widetilde{\Gamma}) := \frac{1}{K}\sum_{k=1}^K\big[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\big]$. Another application of Lemma 2 gives

$$\begin{aligned}T_3 = \mathbb{E}_{q_V}\Big[\sum_k \log N(v_{*k}; \mu, \Sigma)\Big] &= -\frac{1}{2}\sum_k\big[\log|\Sigma| + \mathrm{tr}(\Sigma^{-1}\tilde{\Sigma}_k) + (\tilde{\mu}_k - \mu)^T\Sigma^{-1}(\mu_k - \mu)\big]\\ &= -\frac{1}{2}\sum_k\big[\log|\Sigma| + \mathrm{tr}(\Sigma^{-1}\Psi_k(\widetilde{\Gamma}))\big]\\ &= -\frac{K}{2}\big[\log|\Sigma| + \mathrm{tr}(\Sigma^{-1}S(\widetilde{\Gamma}))\big].\end{aligned}$$

Using Lemma 2 once more, we have

$$T_5 = \mathbb{E}_{q_V} \log q(V) = \sum_k \mathbb{E}_{q_V} \log N(v_{*k}; \widetilde{\mu}_k, \widetilde{\Sigma}_k)$$
$$= \sum_k -\frac{1}{2}\big[\log|\widetilde{\Sigma}_k| + \mathrm{tr}(I_{d_1+d_2})\big] = -\frac{1}{2}K(d_1+d_2) - \frac{1}{2}\sum_k \log|\widetilde{\Sigma}_k|$$

Finally, we have

$$T_4 = \mathbb{E}_{q_Z} \log q(Z) = \mathbb{E}_{q_Z} \sum_{r,i,k} z_{rik} \log \tau_{rik} = \sum_{r,i,k} \tau_{rik} \log \tau_{rik}. \tag{40}$$

Putting the pieces together we get expression (12).

B.3. Updates of $\widetilde{\Sigma}$ and $\widetilde{\mu}$

From (12), the relevant portion of J which is a function of $\widetilde{\Gamma} = (\widetilde{\mu}, \widetilde{\Sigma})$ is given by

$$J(\widetilde{\mu}, \widetilde{\Sigma}) \doteq \sum_{r,i,k} \tau_{rik}\beta_{rik}(\widetilde{\Gamma}, \sigma^2) - \frac{K}{2}\,\mathrm{tr}[\Sigma^{-1}S(\widetilde{\Gamma}, \mu)] + \frac{1}{2}\sum_k \log|\widetilde{\Sigma}_k|$$

Substituting $\beta_{rik}(\widetilde{\Gamma}, \sigma^2) := -\frac{1}{2\sigma_r^2}[\mathrm{tr}\left((\widetilde{\Sigma}_k)_{rr}\right) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2]$, and $S(\widetilde{\Gamma}, \mu) := \frac{1}{K}\sum_{k=1}^K \big[\widetilde{\Sigma}_k + (\widetilde{\mu}_k - \mu)(\widetilde{\mu}_k - \mu)^T\big]$ from their definitions, and looking at the result only as a function of $\widetilde{\Sigma}$, we obtain

$$J(\widetilde{\Sigma}) \doteq -\frac{1}{2}\sum_k \Big[\sum_{r,i} \tau_{rik}\frac{\mathrm{tr}\left((\widetilde{\Sigma}_k)_{rr}\right)}{\sigma_r^2} + \mathrm{tr}[\Sigma^{-1}\widetilde{\Sigma}_k] - \log|\widetilde{\Sigma}_k|\Big] \tag{41}$$

Recalling $\bar{\tau}_{rk} := \sum_{i=1}^{N_r} \tau_{rik}$ and $D_k^{-1} := \mathrm{diag}\left(\frac{\bar{\tau}_{1k}}{\sigma_1^2} I_{d_1}, \frac{\bar{\tau}_{2k}}{\sigma_2^2} I_{d_2}\right)$, we have

$$\sum_{r,i} \tau_{rik}\frac{\mathrm{tr}\left((\widetilde{\Sigma}_k)_{rr}\right)}{\sigma_r^2} = \sum_r \bar{\tau}_{rk}\frac{\mathrm{tr}\left((\widetilde{\Sigma}_k)_{rr}\right)}{\sigma_r^2} = \mathrm{tr}\Big[\sum_r \frac{\bar{\tau}_{rk}}{\sigma_r^2}(\widetilde{\Sigma}_k)_{rr}\Big] = \mathrm{tr}(D_k^{-1}\widetilde{\Sigma}_k).$$

Hence, we obtain $J(\widetilde{\Sigma}) \doteq -\frac{1}{2}\sum_k \mathrm{tr}[(\Sigma^{-1} + D_k^{-1})\widetilde{\Sigma}_k] - \log|\widetilde{\Sigma}_k|$ which is the desired result.

Similarly, by substituting $\beta_{rik}(\widetilde{\Gamma}, \sigma^2)$ and $S(\widetilde{\Gamma}, \mu)$ and looking at the result as a function only of $\widetilde{\mu}$, we obtain

$$J(\widetilde{\mu}) = -\frac{1}{2}\sum_k \Big[\sum_{r,i} \tau_{rik}\frac{\|x_{ri} - \widetilde{\mu}_{rk}\|^2}{\sigma_r^2} + (\widetilde{\mu}_k - \mu)^T\Sigma^{-1}(\widetilde{\mu}_k - \mu)\Big].$$

Let us simplify the sum over r and i. Up to constants as function of $\widetilde{\mu}$, we have

$$\sum_i \tau_{rik}\|x_{ri} - \widetilde{\mu}_{rk}\|^2 \doteq \sum_i \tau_{rik}\big(\|\widetilde{\mu}_{rk}\|^2 - 2\langle x_{ri}, \widetilde{\mu}_{rk}\rangle\big)$$
$$= \bar{\tau}_{rk}\|\widetilde{\mu}_{rk}\|^2 - 2\langle \bar{x}_{rk}, \widetilde{\mu}_{rk}\rangle$$
$$= \bar{\tau}_{rk}\big(\|\widetilde{\mu}_{rk}\|^2 - 2\langle \bar{\mu}_{rk}, \widetilde{\mu}_{rk}\rangle\big)$$
$$\doteq \bar{\tau}_{rk}\|\widetilde{\mu}_{rk} - \bar{\mu}_{rk}\|^2$$

where the second to last equality is by definition of $\bar{\mu}_{rk} := \bar{x}_{rk}/\bar{\tau}_{rk}$. Recalling the definitions of $\bar{\tau}_{rk}$ and $\bar{x}_{rk} := \sum_{i=1}^{N_r} \tau_{rik}x_{ri}$. Hence,

$$\sum_{r,i} \tau_{rik}\frac{\|x_{ri} - \widetilde{\mu}_{rk}\|^2}{\sigma_r^2} \doteq \sum_r \frac{\bar{\tau}_{rk}}{\sigma_r^2}\|\widetilde{\mu}_{rk} - \bar{\mu}_{rk}\|^2 = (\widetilde{\mu}_k - \bar{\mu}_k)^T D_k^{-1}(\widetilde{\mu}_k - \bar{\mu}_k)$$

Thus, we obtain

$$J(\widetilde{\mu}) \doteq -\frac{1}{2}\sum_k \big[(\widetilde{\mu}_k - \bar{\mu}_k)^T D_k^{-1}(\widetilde{\mu}_k - \bar{\mu}_k) + (\widetilde{\mu}_k - \mu)^T\Sigma^{-1}(\widetilde{\mu}_k - \mu)\big].$$

Desired expression (19) follows by applying the following lemma.

Lemma 3 (Sum of quadratic forms). *For symmetric matrices $Q_1, Q_2, \ldots,$*

$$\sum_r (x - m_r)^T Q_r^{-1}(x - m_r) = (x - m)^T Q^{-1}(x - m) + const., \quad \forall x$$

where $Q = (\sum_r Q_r^{-1})^{-1}$ and $m = \sum_r QQ_r^{-1}m_r$.

Proof *Since the two sides are quadratic functions, they are equal up to constants if their derivatives up to second-order match. Equating the Hessians gives $\sum_r Q_r^{-1} = Q^{-1}$. Then, equating the gradients gives $\sum_r Q_r^{-1}(x - m_r) = Q^{-1}(x - m)$, which simplifies to $\sum_r Q_r^{-1}m_r = Q^{-1}m$ in light of the Hessian equality.* ■

B.4. Updates of σ^2, π, p and q

The relevant portion of J as a function (σ_r^2) is

$$J((\sigma_r^2)) = -\frac{1}{2}\sum_r \Big[\frac{1}{\sigma_r^2}\sum_{i,k} \tau_{rik}\big[\mathrm{tr}\left((\widetilde{\Sigma}_k)_{rr}\right) + \|x_{ri} - \widetilde{\mu}_{rk}\|^2\big] + d_r N_r \log\sigma_r^2\Big]. \tag{42}$$

The maximizer of the function $x \mapsto Ax^{-1} + B\log x$ is A/B (assuming $A, B > 0$), from which (20) follows.

As a function π, J has the form $J(\pi) \doteq \sum_{r,i,k} \tau_{rik}\log\pi_{rk} = \sum_{r,k} \bar{\tau}_{rk}\log\pi_{rk}$ using the definition of $\bar{\tau}_{rk}$. The following lemma is standard. (Recall that $\mathcal{P}_K$ is the set of probability K-vectors.)

Lemma 4. *For any nonnegative vector $(a_1, \ldots, a_K)$,*

$$\underset{p\,\in\,\mathcal{P}_K}{\mathrm{argmax}} \sum_k a_k \log p_k = \frac{1}{\sum_k a_k}(a_1, \ldots, a_K).$$

Based on the lemma, π_1-update is $\pi_1 = (\bar{\tau}_{11}, \ldots, \bar{\tau}_{1K})/(\sum_k \bar{\tau}_{1k})$. The update for π_2 is similar.

To update p we note that $J(p) \doteq \sum_{i,j} \gamma_{ij}(\tau)(A_{ij}\log p + (1 - A_{ij})\log(1 - p))$. The update is obtained by setting the derivative to zero. The q-update is similar.

Appendix C. Details for degree-corrected algorithm

C.1. τ-update with degree restriction

In this section, we derive a dual ascent algorithm for the optimization problem (33) which has to be solved for updating τ under the degree corrected model. Letting $a_{ik} := \phi_1[A\tau_2]_{ik} + \phi_0\bar{\tau}_{2k} + \xi_{1ik}$, and with some notational simplifications, problem (33) can be stated as

$$\min_{X=(x_{ik})} -\sum_{ik} x_{ik}(a_{ik} - \log x_{ik}), \quad \text{s.t. } X \in \mathcal{P}_{n,K}, \ \sum_i x_{ik}(\theta_i - 1) = 0, \forall k \tag{43}$$

where $\mathcal{P}_{n,K} := \{(x_{ik}) \in \mathbb{R}_+^{n\times K} : \sum_k x_{ik} = 1, \forall i\}$. Let $f(X)$ be the objective function in (43), and let us write the constraint in vector form $\sum_i x_{ik}(\theta_i - 1) = X^T(\theta - \mathbf{1}) = 0$. With the Lagrangian $L(X, \lambda) = f(X) - \lambda^T[X^T(\theta - \mathbf{1})]$, the dual function is

$$\Phi(\lambda) := \min_{X\in\mathcal{P}_{n,K}} L(X, \lambda),$$

and the dual problem is $\max_\lambda \Phi(\lambda)$. A dual-descent algorithm maximizes Φ by performing a gradient ascent on Φ: $\lambda^+ = \lambda + \mu\nabla\Phi(\lambda)$. We know that the gradient of Φ is given by $\partial_\lambda L(X, \lambda)$ evaluated at $X^*(\lambda)$, the optimizer of the Lagrangian. More precisely,

$$\nabla\Phi(\lambda) = [X^*(\lambda)]^T(\theta - \mathbf{1}), \quad \text{where } X^*(\lambda) = \operatorname*{argmax}_{X\in\mathcal{P}_{n,K}} -L(X, \lambda)$$

Solving for $X^*(\lambda)$ is an instance of the problem in Lemma 1. The problem is separable over i and for fixed i, we are maximizing $\sum_k x_{ik}(a_{ik} - \log x_{ik}) + \sum_k \lambda_k x_{ik}(\theta_i - 1) = \sum_k x_{ik}\{[a_{ik} + \lambda_k(\theta_i - 1)] - \log x_{ik}\}$ over $x_{i*} \in \mathcal{P}_{1,K}$, the solution of which is given by the softmax operation

$$x^*_{ik}(\lambda) \propto_k \exp(a_{ik} + \lambda_k(\theta_i - 1)). \tag{44}$$

Thus, the update for the dual descent can be written

$$\lambda_k^+ = \lambda_k - \mu\sum_i x^*_{ik}(\lambda)(\theta_i - 1), \quad \forall k. \tag{45}$$

C.2. θ-update

Simplyfying the notation, let $h(\theta) = -\sum_i d_i \log\theta_i$ and $V := \{\theta : \sum_i \tau_{ik}(\theta_i - 1) = 0\}$. The problem is equivalent to minimizing $h(\theta) + \delta_V(\theta)$ over θ where δ_V is the indicator of V in the sense of convex analysis. Douglas-Rachford algorithm, also known as Spingarn's method of partial inverses in this special case, is given by

$$\begin{aligned}\theta^+ &= \operatorname{prox}_{th}(\xi)\\ \xi^+ &= \xi + P_V(2\theta^+ - \xi) - \theta^+\end{aligned} \tag{46}$$

where prox_{th} is the proximal operator of $th(\cdot)$ and P_V is the projection onto V. Due to separability, it is not hard to see that $[\operatorname{prox}_{th}(\theta)]_i = \operatorname{prox}_{td_i\log(\cdot)}(\theta_i)$. This univariate proximal operator can be easily shown to coincide with $[f_t(\theta, d)]_i$ as given in (35).

As for the projection, in general with $C = \{x : Ax = b\}$, we have $P_C(x) = x + A^T(AA^T)^{-1}(b - Ax)$. Note that $V = \{\theta : \tau^T(\theta - \mathbf{1}) = 0\}$. Applying the general result with $A = \tau^T$ and $b = \tau^T\mathbf{1}$, we get $P_V(\theta) = \theta + \tau(\tau^T\tau)^{-1}\tau^T(\mathbf{1} - \theta) = \theta - H(\theta - \mathbf{1})$, with the obvious choice for H. Thus (46) simplifies to

$$\xi^+ = \xi + (2\theta^+ - \xi) - H(2\theta^+ - \xi - \mathbf{1}) - \theta^+$$

which gives the claimed update.

Appendix D. Expected average degree

Let $\hat{\lambda}_i = \sum_j A_{ij}$ and $\hat{\lambda}_j = \sum_i A_{ij}$ be the degree of node i from group 1, and node j from group 2, respectively. Then, the average degree of the network is

$$\hat{\lambda} = \frac{\sum_i \hat{\lambda}_i + \sum_j \hat{\lambda}_j}{N_1 + N_2} = \frac{2\sum_{i,j} A_{ij}}{N_1 + N_2}.$$

Recall the definition of clusters C_{rk} from (1). The expected average degree can be derived as follows: Assume that $i \in C_{1k}$, then $\mathbb{E}[A_{ij}] = \theta_{1i}\theta_{2j}(p1\{j \in C_{2k}\} + q1\{j \notin C_{2k}\})$. Then

$$\mathbb{E}(\hat{\lambda}_i) = \theta_{1i}\Big(p\sum_{j\in C_{2k}}\theta_{2j} + q\sum_{j\notin C_{2k}}\theta_{2j}\Big) = \theta_i r_k, \quad \text{where} \quad r_k := |C_{2k}|p + (N_2 - |C_{2k}|)q$$

Here, we have used the normalization (7): $\sum_{j\in C_{2\ell}}\theta_{2j} = |C_{2\ell}|$ for all $\ell \in [K]$. Now,

$$\sum_{i=1}^{N_1}\mathbb{E}(\hat{\lambda}_i) = \sum_{i=1}^{N_1}\sum_k 1\{i \in C_{1k}\}\mathbb{E}(\hat{\lambda}_i) = \sum_k r_k\sum_{i=1}^{N_1}\theta_{1i}1\{i \in C_{1k}\} = \sum_k r_k|C_{1k}|$$

using the normalization of θ_{1i}. Dividing by N_1N_2 and using $|C_{rk}|/N_r = \pi_{rk}$ for $r = 1, 2$,

$$\sum_{i=1}^{N_1}\mathbb{E}(\hat{\lambda}_i) = N_1N_2\sum_k\big[\pi_{1k}q + \pi_{1k}\pi_{2k}(p - q)\big] = N_1N_2(q + \Pi(p - q)),$$

where $\Pi := \sum_k \pi_{1k}\pi_{2k}$. By symmetry, $\sum_{j=1}^{N_2}\mathbb{E}(\hat{\lambda}_j) = N_1N_2(q + \Pi(p - q))$. Hence,

$$\lambda = \mathbb{E}[\hat{\lambda}] = \frac{2N_1N_2}{N_1 + N_2}\big(q + \Pi(p - q)\big), \quad \text{where } \Pi := \sum_k \pi_{1k}\pi_{2k}.$$

Note that $\frac{2N_1N_2}{N_1+N_2} = 2/(N_1^{-1} + N_2^{-1})$ is the harmonic mean of N_1 and N_2.

Appendix E. More simulations

Figure 13 shows the effect of varying the dimension of the covariates $d = (d_1, d_2)$ and the scale of the covariance matrix ν. The setup is as in Section 4, and in particular $\Sigma = \nu I$ controls how far apart the centers of the covariate clusters $v_{*k} \in \mathbb{R}^{d_1+d_2}$ are. Only the `mbiSBM` algorithm is shown (with no degree correction and) initialized with Dirichlet-perturbed truth ($\sim$`rnd`). As one expects, increasing the dimensions of the covariates increases the performance (seemingly without bound). Increasing ν improves the performance up to a point, but there is a saturation effect beyond that point, where the performance remains more or less the same.

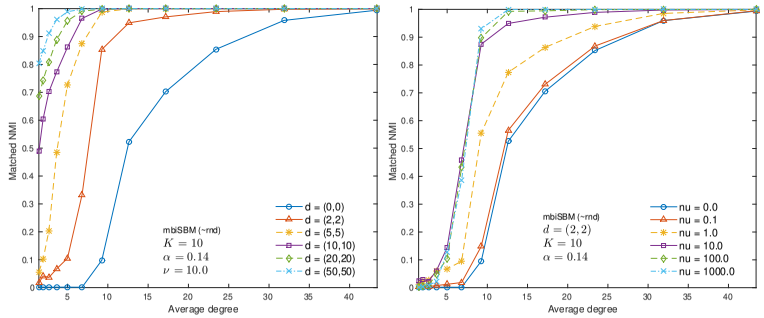


Figure 13: (Left) Effect of changing the dimension of covariates $d = (d_1, d_2)$ and (right) the parameter ν where $\Sigma = \nu I_{d_1+d_2}$.

References

Emmanuel Abbe. Community detection and stochastic block models: recent developments. *arXiv preprint arXiv:1703.10146*, 2017.

Emmanuel Abbe, Afonso S Bandeira, and Georgina Hall. Exact recovery in the stochastic block model. *IEEE Transactions on Information Theory*, 62(1):471–487, 2016.

Arash A. Amini and Elizaveta Levina. On semidefinite relaxations for the block model. *The Annals of Statistics*, 46(1):149–179, 2018.

Arash A. Amini, Aiyou Chen, Peter J. Bickel, and Elizaveta Levina. Pseudo-likelihood methods for community detection in large sparse networks. *Annals of Statistics*, 41(4):2097–2122, 2013.

Nils D Arvold, Alphonse G Taghian, Andrzej Niemierko, Rita F Abi Raad, Meera Sreedhara, Paul L Nguyen, Jennifer R Bellon, Julia S Wong, Barbara L Smith, and Jay R Harris. Age, breast cancer subtype approximation, and local recurrence after breast-conserving therapy. *Journal of Clinical Oncology*, 29(29):3885, 2011.

Shantanu Banerji, Kristian Cibulskis, Claudia Rangel-Escareno, Kristin K Brown, Scott L Carter, Abbie M Frederick, Michael S Lawrence, Andrey Y Sivachenko, Carrie Sougnez, Lihua Zou, et al. Sequence analysis of mutations and translocations across breast cancer subtypes. *Nature*, 486(7403):405, 2012.

Mukesh Bansal, Giusy Della Gatta, and Diego Di Bernardo. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22(7):815–822, 2006.

Albert-László Barabási and Réka Albert. Emergence of Scaling in Random Networks. *Science*, 286(October):509–512, 1999.

Albert-Laszlo Barabasi and Zoltan N Oltvai. Network biology: understanding the cell's functional organization. *Nature reviews genetics*, 5(2):101, 2004.

Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B*, 48(3):259–302, 1986.

Peter J Bickel and Aiyou Chen. A nonparametric view of network models and Newman-Girvan and other modularities. *Proceedings of the National Academy of Sciences of the United States of America*, 106(50):21068–21073, 2009.

Norbert Binkiewicz, Joshua T Vogelstein, and Karl Rohe. Covariate-assisted spectral clustering. *Biometrika*, 104(2):361–377, 2017.

Gilles Bisson and Fawad Hussain. Chi-Sim: A new similarity measure for the co-clustering task. *Proceedings - 7th International Conference on Machine Learning and Applications, ICMLA 2008*, pages 211–217, 2008.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Maggie CU Cheang, Stephen K Chia, David Voduc, Dongxia Gao, Samuel Leung, Jacqueline Snider, Mark Watson, Sherri Davies, Philip S Bernard, Joel S Parker, et al. Ki67 index, her2 status, and prognosis of patients with luminal b breast cancer. *JNCI: Journal of the National Cancer Institute*, 101(10):736–750, 2009.

Yizong Cheng and George M Church. Biclustering of expression data. *Proceedings of the International Conference on Intelligent Systems for Molecular Biology ; ISMB. International Conference on Intelligent Systems for Molecular Biology*, 8:93–103, 2000.

David Choi and Patrick J. Wolfe. Co-clustering separately exchangeable network data. *Annals of Statistics*, 42(1):29–63, 2014.

Anirban Dasgupta, J.E. Hopcroft, and F. McSherry. Spectral Analysis of Random Graphs with Skewed Degree Distributions. *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 602–610, 2004.

Eric Davidson and Michael Levin. Gene regulatory networks. *Proceedings of the National Academy of Sciences*, 102(14):4935–4935, 2005.

DBLP. *Computer Science Bibliography*. URL `https://dblp.uni-trier.de`.

Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Inference and phase transitions in the detection of modules in sparse networks. *Physical Review Letters*, 107(6):1–5, 2011.

Inderjit S Dhillon. Co-clustering documents and words using Bipartite Co-clustering documents and words using Bipartite Spectral Graph Partitioning. *Proc of 7th ACM SIGKDD Conf*, pages 269–274, 2001.

Inderjit S Dhillon. Information-Theoretic Co-clustering. *Proc. of 9th ACM SIGKDD Int'l Conf.*, pages 89–98, 2003.

Jim Douglas and H.H Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Transactions of the American mathematical society*, 82:421–439, 1956.

Cheryl J. Flynn and Patrick O. Perry. Consistent Biclustering. *arXiv preprint arXiv:1206.6927*, 2012.

Chao Gao, Zongming Ma, Anderson Y. Zhang, and Harrison H. Zhou. Community Detection in Degree-Corrected Block Models. *arXiv preprint arXiv:1607.06993*, 2016.

Chao Gao, Zongming Ma, Anderson Y. Zhang, and Harrison H. Zhou. Achieving Optimal Misclassification Proportion in Stochastic Block Model. *The Journal of Machine Learning Research*, 18(1):1980–2024, 2017.

Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian Data Analysis, Second Edition*. Chapman & Hall/CRC Texts in Statistical Science, 2003.

Mark B Gerstein, Joel Rozowsky, Koon-Kiu Yan, Daifeng Wang, Chao Cheng, James B Brown, Carrie A Davis, LaDeana Hillier, Cristina Sisu, Jingyi Jessica Li, et al. Comparative analysis of the transcriptome across distant species. *Nature*, 512(7515):445, 2014.

Prem K Gopalan and David M Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences of the United States of America*, 110(36):14534–9, 2013.

Bruce Hajek, Yihong Wu, and Jiaming Xu. Achieving Exact Cluster Recovery Threshold via Semidefinite Programming: Extensions. *IEEE Transactions on Information Theory*, 62(10): 5918–5937, 2016.

J. A. Hartingan. Direct Clustering of a Data Matrix. *Journal of the American Statistical Society*, 67(337):123–129, 1972.

Paul H Harvey, Mark D Pagel, et al. *The comparative method in evolutionary biology*, volume 239. Oxford university press Oxford, 1991.

Michael Hecker, Sandro Lambeck, Susanne Toepfer, Eugene Van Someren, and Reinhard Guthke. Gene regulatory network inference: data integration in dynamic models-a review. *Biosystems*, 96(1):86–103, 2009.

Ingrid Hedenfalk, David Duggan, Yidong Chen, Michael Radmacher, Michael Bittner, Richard Simon, Paul Meltzer, Barry Gusterson, Manel Esteller, Mark Raffeld, et al. Gene-expression profiles in hereditary breast cancer. *New England Journal of Medicine*, 344(8):539–548, 2001.

Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

ipapi. *IP Lookup API*. URL `http://ipapi.co`.

Reika Iwakawa, Takashi Kohno, Yasushi Totoki, Tatsuhiro Shibata, Katsuya Tsuchihara, Sachiyo Mimaki, Koji Tsuta, Yoshitaka Narita, Ryo Nishikawa, Masayuki Noguchi, et al. Expression and clinical significance of genes frequently mutated in small cell lung cancers defined by whole exome/rna sequencing. *Carcinogenesis*, 36(6):616–621, 2015.

Lei Jing and Alessandro Rinaldo. Consistency of spectral clustering in stochastic block models. *Annals of Statistics*, 43(1):215–237, 2015.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

David Kahle and Hadley Wickham. ggmap : Spatial Visualization with. *The R Journal*, 5(1): 144–161, 2013.

Brian Karrer and M. E J Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1):1–11, 2011.

Brian Keegan. *GitHub repository: 2014-On-Wikipedia*, 2014. URL `https://github.com/brianckeegan/2014-On-Wikipedia`.

Mikko Kivela, Alex Arenas, Marc Barthelemy, James P. Gleeson, Yamir Moreno, and Mason A. Porter. Multilayer networks. *Journal of Complex Networks*, 2(3):203–271, 2014.

Daniel B. Larremore, Aaron Clauset, and Abigail Z. Jacobs. Efficiently inferring community structure in bipartite networks. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 90(1):17–22, 2014.

Jingyi Jessica Li, Haiyan Huang, Peter J Bickel, and Steven E Brenner. Comparison of d. melanogaster and c. elegans developmental stages, tissues, and cells by modencode rna-seq data. *Genome research*, 24(7):1086–1101, 2014.

Ben-Yang Liao and Jianzhi Zhang. Evolutionary conservation of expression profiles between human and mouse orthologous genes. *Molecular biology and evolution*, 23(3):530–540, 2005.

Sara C Madeira, Miguel C Teixeira, Isabel Sa-Correia, and Arlindo L Oliveira. Identification of regulatory modules in time series gene expression data using a linear time biclustering algorithm. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 7(1): 153–165, 2010.

Piyush B Madhamshettiwar, Stefan R Maetschke, Melissa J Davis, Antonio Reverter, and Mark A Ragan. Gene regulatory network inference: evaluation and application to ovarian cancer allows the prioritization of drug targets. *Genome medicine*, 4(5):41, 2012.

Francesco M. Malvestuto. Statistical treatment of the information content of a database. *Information Systems*, 11(3):211–223, 1986.

Laurent Massoulié. Community detection thresholds and the weak Ramanujan property. *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 694–706, 2014.

Elchanan Mossel, Joe Neeman, and Allan Sly. A Proof Of The Block Model Threshold Conjecture. *Combinatorica*, pages 1–44, 2013.

Elchanan Mossel, Joe Neeman, and Allan Sly. Reconstruction and estimation in the planted partition model. *Probab. Theory Relat. Fields*, 162(3):431–461, 2015.

M. E. J. Newman and Aaron Clauset. Structure and inference in annotated networks. *Nature Communications*, 7(May):1–16, 2016.

Daniel O'Connor and Lieven Vandenberghe. Primal-Dual Decomposition by Operator Splitting and Applications to Image Deblurring. *SIAM Journal on Imaging Sciences*, 7(3):1724–1754, 2014.

Neelroop N Parikshak, Michael J Gandal, and Daniel H Geschwind. Systems biology and gene networks in neurodevelopmental and neurodegenerative disorders. *Nature Reviews Genetics*, 16(8):441, 2015.

Tai Qin and Karl Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. *Advances in Neural Information Processing Systems*, pages 1–9, 2013.

Zahra S. Razaee, Arash A. Amini, and Jingyi Li. *GitHub repository: mbiSBM*. URL `https://github.com/aaamini/mbisbm`.

Dan Robinson, Eliezer M Van Allen, Yi-Mi Wu, Nikolaus Schultz, Robert J Lonigro, Juan-Miguel Mosquera, Bruce Montgomery, Mary-Ellen Taplin, Colin C Pritchard, Gerhardt Attard, et al. Integrative clinical genomics of advanced prostate cancer. *Cell*, 161(5):1215–1228, 2015.

Karl Rohe, Sourav Chatterjee, and Bin Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *Annals of Statistics*, 39(4):1878–1915, 2011.

Karl Rohe, Tai Qin, and Bin Yu. Co-clustering directed graphs to discover asymmetries and directional communities. *Proceedings of the National Academy of Sciences of the United States of America*, 113(45):12679–12684, 2016.

Eran Segal, Michael Shapira, Aviv Regev, Dana Pe'er, David Botstein, Daphne Koller, and Nir Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature genetics*, 34(2):166, 2003.

Therese Sørlie, Charles M Perou, Robert Tibshirani, Turid Aas, Stephanie Geisler, Hilde Johnsen, Trevor Hastie, Michael B Eisen, Matt Van De Rijn, Stefanie S Jeffrey, et al. Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. *Proceedings of the National Academy of Sciences*, 98(19):10869–10874, 2001.

Therese Sørlie, Robert Tibshirani, Joel Parker, Trevor Hastie, James Stephen Marron, Andrew Nobel, Shibing Deng, Hilde Johnsen, Robert Pesich, Stephanie Geisler, et al. Repeated observation of breast tumor subtypes in independent gene expression data sets. *Proceedings of the national academy of sciences*, 100(14):8418–8423, 2003.

Jie Tang, Sen Wu, Jimeng Sun, , and Hang. Su. Cross-domain collaboration recommendation. *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1285–1293, 2012.

Bert Vogelstein and Kenneth W Kinzler. Cancer genes and the pathways they control. *Nature medicine*, 10(8):789, 2004.

Gunter Von Minckwitz, Michael Untch, Jens-Uwe Blohmer, Serban D Costa, Holger Eidtmann, Peter A Fasching, Bernd Gerber, Wolfgang Eiermann, Jörn Hilfrich, Jens Huober, et al. Definition and impact of pathologic complete response on prognosis after neoadjuvant chemotherapy in various intrinsic breast cancer subtypes. *J Clin oncol*, 30(15):1796–1804, 2012.

Martin J. Wainwright and Michael I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

Kai Wang, Junsuo Kan, Siu Tsan Yuen, Stephanie T Shi, Kent Man Chu, Simon Law, Tsun Leung Chan, Zhengyan Kan, Annie SY Chan, Wai Yin Tsui, et al. Exome sequencing identifies frequent mutation of arid1a in molecular subtypes of gastric cancer. *Nature genetics*, 43(12): 1219, 2011.

Wikimedia Statistics. URL `https://stats.wikimedia.org`.

Jason Wyse, Nial Friel, and Pierre Latouche. Inferring structure in bipartite networks using the latent block model and exact ICL. (2013):23, 2014.

Bowei Yan and Purnamrita Sarkar. Convex Relaxation for Community Detection with Covariates. *arXiv preprint arXiv:1607.02675*, 2016.

Yuan Zhang, Elizaveta Levina, and Ji Zhu. Community Detection in Networks with Node Features. *Electronic Journal of Statistics*, 10(2):3153–3178, 2016.

Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. Consistency of community detection in networks under degree-corrected stochastic block models. *Annals of Statistics*, 40(4):2266–2292, 2012.

Tao Zhou, Jie Ren, Matúš Medo, and Yi Cheng Zhang. Bipartite network projection and personal recommendation. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 76(4):1–7, 2007.

Zhixin Zhou and Arash A. Amini. Optimal bipartite network clustering. *arXiv preprint arXiv:1803.06031*, 2018.

Journal of Machine Learning Research 20 (2019) 1-93 Submitted 3/17; Revised 2/19; Published 3/19

Optimal Policies for Observing Time Series and Related Restless Bandit Problems

Christopher R. Dance CHRIS.DANCE@NAVERLABS.COM
Tomi Silander TOMI.SILANDER@NAVERLABS.COM
NAVER LABS Europe
6 chemin de Maupertuis
Meylan, 38240, France

Editor: Nikos Vlassis

Abstract

The trade-off between the cost of acquiring and processing data, and uncertainty due to a lack of data is fundamental in machine learning. A basic instance of this trade-off is the problem of deciding when to make noisy and costly observations of a discrete-time Gaussian random walk, so as to minimise the posterior variance plus observation costs. We present the first proof that a simple policy, which observes when the posterior variance exceeds a threshold, is optimal for this problem. The proof generalises to a wide range of cost functions other than the posterior variance. It is based on a new verification theorem by Niño-Mora that guarantees threshold structure for Markov decision processes, and on the relation between binary sequences known as *Christoffel words* and the dynamics of discontinuous nonlinear maps, which frequently arise in physics, control and biology.

This result implies that optimal policies for *linear-quadratic-Gaussian control with costly observations* have a threshold structure. It also implies that the restless bandit problem of observing multiple such time series, has a well-defined *Whittle index policy*. We discuss computation of that index, give closed-form formulae for it, and compare the performance of the associated index policy with heuristic policies.

Keywords: restless bandits, Whittle index, Christoffel words, Sturmian words, Kalman filter, linear-quadratic-Gaussian control

1. Introduction

This paper answers three closely-related questions about discrete-time filtering of scalar time series with costly observations, where the nature of the observations is controlled through a query action. The first two questions concern the structure of optimal policies for observing a single time series so as to minimise either a function of the posterior variance (Theorem 1, Excerpt) or a quadratic function of the system state and control input (Corollary 2). The third question concerns the observation of several such time series with a constraint on the number of time series that can be observed simultaneously. This is an instance of a *restless bandit problem* and it is interesting to know that the problem has a well-defined *Whittle index policy* (Theorem 1).

This introduction begins with the time-series model (Section 1.1) that the three questions have in common. It then motivates, formulates and states the key results for each question in turn (Sections 1.2 to 1.4). It concludes with an intuitive guide to the main concepts involved in the proof (Section 1.5) and a description of the structure of the rest of the paper (Section 1.6).

©2019 Christopher R. Dance and Tomi Silander.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-185.html.

Notation. In this paper, $\mathbb{Z}_+ = \{0, 1, 2, \dots\}$, $\mathbb{Z}_{++} = \{1, 2, 3, \dots\}$, $\mathbb{R}_+ = [0, \infty)$, $\mathbb{R}_{++} = (0, \infty)$ and $\overline{\mathbb{R}} = [-\infty, \infty] = \mathbb{R} \cup \{-\infty, \infty\}$ denotes the extended real numbers. Left limits are denoted by $f(y^-) = \lim_{x \uparrow y} f(x)$ and right limits by $f(y^+) = \lim_{x \downarrow y} f(y)$. The terms *increasing* and *decreasing* are used in the strict sense, with non-decreasing and non-increasing used otherwise.

1.1. Time-Series Model

We consider the classic discrete-time scalar normally-distributed state-space model. In this model, the state is partially observed through measurements as fully described by the conditional dependencies

$$\left.\begin{aligned} Z_0 &\sim \mathcal{N}(z_0, v_0) \\ Z_{t+1}|Z_t, u_t &\sim \mathcal{N}(\bar{A}Z_t + \bar{B}u_t, \Sigma_Z) \\ Y_{t+1}|Z_{t+1}, a_t &\sim \mathcal{N}(Z_{t+1}, \Sigma_Y(a_t)) \end{aligned}\right\} \quad \text{for } t \in \mathbb{Z}_+. \tag{1}$$

The state Z_t is a real-valued random variable with initial mean z_0 and variance v_0. The sequence of states depends on the control or exogenous input $u_t \in \mathbb{R}$. The measurement Y_{t+1} is a real-valued random variable which depends on a query action $a_t \in \{0, 1\}$. The variances $\Sigma_Z, \Sigma_Y(0), \Sigma_Y(1) > 0$ and real-valued parameters $\bar{A}, \bar{B}$ are known. Query action $a_t = 1$ is assumed to correspond to a higher-quality observation than query action $a_t = 0$, so that $\Sigma_Y(1) < \Sigma_Y(0)$ and it is possible that $\Sigma_Y(0) = \infty$ which represents a totally uninformative observation or no observation at all.

The observed history H_t at time t is $z_0, v_0, a_0, a_1, \dots, a_{t-1}, u_0, u_1, \dots, u_{t-1}, Y_1, Y_2, \dots, Y_t$. Under the Bayesian filter, the information state is given by the posterior mean $z_t := \mathbb{E}[Z_t|H_t]$ and variance $v_t := \mathbb{E}[(Z_t - z_t)^2|H_t]$. In this case, the Bayesian filter is the Kalman filter (Thiele, 1880; Kalman, 1960) and it follows that the information state undergoes the following Markovian transitions:

$$\left.\begin{aligned} z_{t+1}|z_t, v_t, a_t, u_t &\sim \mathcal{N}(\bar{A}z_t + \bar{B}u_t, \bar{A}^2 v_t + \Sigma_Z - \Phi_{a_t}(v_t)) \\ v_{t+1}|z_t, v_t, a_t, u_t &= \Phi_{a_t}(v_t) \end{aligned}\right\} \quad \text{for } t \in \mathbb{Z}_+ \tag{2}$$

where $\Phi_a : \mathbb{R}_+ \to \mathbb{R}_+$ for $a \in \{0, 1\}$ is the Möbius transformation

$$\Phi_a(v) := \frac{(\bar{A}^2 v + \Sigma_Z) \times \Sigma_Y(a)}{(\bar{A}^2 v + \Sigma_Z) + \Sigma_Y(a)}. \tag{3}$$

The above model excludes the relatively-simple case of learning a *stationary* parameter Z_t, since we assume that $\Sigma_Z > 0$. Under this assumption, to facilitate analysis, we shall scale the units of variance such that $\Sigma_Z = 1$. Specifically, we consider the (relative) variance state at time t given by $x_t := v_t/\Sigma_Z$ and the (relative) precision of the observations given by $\theta_a := \Sigma_Z/\Sigma_Y(a)$ for $a \in \{0, 1\}$. We also define the parameter $r := |\bar{A}|$, noting that $\bar{A}$ can be negative. After this change of coordinates, the variance transitions are now given by the Möbius transformation

$$\phi_a(x) := \frac{\Phi_a(\Sigma_Z x)}{\Sigma_Z} = \frac{(r^2\Sigma_Z x + \Sigma_Z)\Sigma_Y(a)}{\Sigma_Z(r^2\Sigma_Z x + \Sigma_Z + \Sigma_Y(a))} = \frac{r^2 x + 1}{\theta_a(r^2 x + 1) + 1}. \tag{4}$$

The above model also excludes *totally informative* observations, since we assume that $\Sigma_Y(1) > 0$. Such models are nevertheless partly-addressed in the limit as the precision $\theta_1 \to \infty$ in Proposition 31, where it is possible to find closed-form formulae relating to optimal policies for the three problems discussed in the rest of this introduction. The rationale for excluding such cases from our analysis is that our characterisation of the behaviour of maps-with-gaps (Theorem 12) no longer applies.

1.2. Optimal Policies for Observing a Single Time Series

The simplest problem addressed here involves an uncertainty cost $C(x)$, where x is the (relative variance) state introduced just above, and a price $\lambda \in \mathbb{R}$ to be paid every time query action $a = 1$ is taken. Price λ might reflect costs of energy, labour, communication, computational processing, hardware or risks associated with each measurement. Recall that a policy is *non-anticipative* if it selects actions at time t based only on information available up-to and including time t. The objective is to find a non-anticipative policy π that selects query actions so as to minimise the β-discounted performance functional, for discount factor $\beta \in [0, 1)$, for all initial states x,

$$\mathbb{E}_x^\pi \left[\sum_{t=0}^{\infty} \beta^t (\lambda A_t + C(X_t)) \right] \tag{5}$$

where $\mathbb{E}_x^\pi$ denotes the expectation over sequences $(X_t, A_t)_{t=0}^\infty$ of states X_t and actions A_t with initial state $X_0 = x$, where actions A_t are taken according to policy π and transitions are according to (2). Note that this performance functional depends neither on the control or exogenous input u_t nor on the posterior mean z_0 since these appear neither in the costs nor in the transitions of the posterior variance, which are given by the deterministic mapping (4). So, while in some settings, such as that considered in Section 1.3, it is natural to require policies π to also select the control u_t, in the setting considered here it does not matter which value for u_t is chosen and we only require the policy to select the query action a_t. Thus, this problem corresponds to the following deterministic dynamic program for value function $V : \mathbb{R}_{++} \to \mathbb{R}$,

$$V(x) = \min_{a \in \{0,1\}} \Big\{ \lambda a + C(x) + \beta V(\phi_a(x)) \Big\}. \tag{6}$$

The first question addressed in this paper is: *for what cost functions is a threshold policy optimal for this problem?* For instance, one may intuitively guess that optimal policies for variance minimisation with $C(x_t) = x_t$, for entropy minimisation with $C(x_t) = \log(x_t)$, or for precision maximisation with $C(x_t) = -1/x_t$, might involve making expensive observations at time t when the variance state x_t exceeds a threshold. The following assumption on the model covers these examples.

Assumption A1

(i) The state space $\mathcal{I}$ is either $[0, \infty)$ or $(0, \infty)$.

(ii) The discount factor β is in $[0, 1)$.

(iii) The transition functions $\phi_a : \mathbb{R}_+ \to \mathbb{R}_+$ for $a \in \{0, 1\}$ are of the form

$$\phi_a(x) = \frac{r^2 x + 1}{\theta_a(r^2 x + 1) + 1}$$

for some $0 \le \theta_0 < \theta_1 < \infty$ and some $r \in (0, 1]$.

(iv) The uncertainty cost function $C : \mathcal{I} \to \mathbb{R}$ is of the form $C(x) = \sum_{i=1}^{n_C} C_i(x)$ for some $n_C \in \mathbb{Z}_{++}$, where each of the functions $C_i : \mathcal{I} \to \mathbb{R}$ satisfies one of the following conditions:

C1. For $x \in \mathcal{I}$, the derivatives $C_i'(x) := \frac{d}{dx} C_i(x)$ and $C_i''(x) := \frac{d^2}{dx^2} C_i(x)$ exist and

- *the function $C_i(x)$ is concave,*
- *the function $\frac{1}{x^3} C_i''\left(\frac{1}{x}\right)$ is non-decreasing,*
- *and the function $\frac{1}{x^2} C_i'\left(\frac{1}{x}\right)$ is non-increasing and convex.*

C2. For $x \in \mathcal{I}$, the function $C_i(x)$ is non-decreasing, convex and differentiable.

(v) There exists a measurable function $w : \mathcal{I} \to [1, \infty)$ and constants $M > 0$ and $\gamma \in [\beta, 1)$ such that for every state $x \in \mathcal{I}$:

(a) $|C(x)| \le M w(x)$; and

(b) $\beta \max_{a \in \{0,1\}} w(\phi_a(x)) \le \gamma w(x)$.

Regarding part (i) of the above assumption, note that we may work with the interval $\mathcal{I} = (0, \infty)$ in cases where the cost function $C(x)$ is not a real number for $x = 0$, in order to include cases like $\log(x)$ and $-1/x$, but the results of the paper continue to hold in cases where $C(0)$ is defined. In fact, the main results of the paper also apply to smaller intervals $\mathcal{I}$, as long as the initial state is contained in the interval $\mathcal{I}$ and the open interval (y_1, y_0) is in $\mathcal{I}$, where y_a is the unique solution to $\phi_a(y_a) = y_a$ (see Section 2 for further discussion of such fixed points.)

In part (iii) of the above assumption, the inequality $\theta_0 < \theta_1$ is taken to be strict as otherwise the problem is trivial. Specifically, if $\theta_0 = \theta_1$ then the state sequence does not depend on the policy, so the policy that always takes action 0 is optimal if $\lambda > 0$, whereas the policy that always takes action 1 is optimal if $\lambda < 0$.

Part (iv) of the above assumption is at the heart of our answer to our first question: it represents the most general class of cost functions $C(\cdot)$ that our approach to proving optimality of threshold policies naturally allows for. Delving into the proof, the specific form of part (iv) results from a requirement that certain *majorisation* inequalities (Marshall et al., 2010) hold in Lemma 25.

Uncertainty cost functions $C(\cdot)$ satisfying part (iv) may be neither convex nor concave. For instance in the case $C(x) = (x^3 - 1)/x$, we can write $C(x) = \sum_{i=1}^{n_C} C_i(x)$ for $n_C = 2$ with $C_1(x) = x^2$, a convex function satisfying part C1, and with $C_2(x) = -1/x$, a concave function satisfying part C2. In this instance, the cost $C(x) = (x^3 - 1)/x$ is *unbounded* from both below and above (that is, $\lim_{x \downarrow 0} C(x) = -\infty$ and $\lim_{x \uparrow \infty} C(x) = \infty$). However,

it is possible that the cost is *bounded* both below and above, for instance in the case $C(x) = x/(x+1)$.

In general part (iv) of the above assumption can always be satisfied with $n_C = 2$ terms. This is because finite sums of functions satisfying part C1 also satisfy part C1, since finite sums of functions satisfying part C2 also satisfy part C2, and since the zero-function ($C(x) = 0$ for all x) satisfies both cases.

Also part (iv) of the above assumption requires that functions C_i satisfying C2 have a derivative C_i'. This is simply for convenience in the proofs of Section 3.3. As such functions are real-valued convex functions, one can instead set C_i' equal to any subderivative at points where the derivative is not defined.

Part (v) is key to the *weighted supremum norm* approach to infinite-horizon discounted Markov decision problems, as is explained in detail by Wessels (1977) and Hernández-Lerma and Lasserre (1999, Chapter 8). It ensures that the infinite series in the objective function is well-defined for all policies π and initial states $x \in \mathcal{I}$, even though the cost function $C(\cdot)$ may be unbounded. Furthermore, although Bellman's equation may have multiple unbounded solutions $V(\cdot)$, part (v) guarantees that the optimal value function is the unique solution with $\sup_{x\in\mathcal{I}} |V(x)/w(x)| < \infty$. For $|C(x)| = x^q$ with $q \geq -1$, it is not hard to see that part (v) is satisfied for $w(x) := \max\{y_1/x, \alpha^{x-y_1}\}$ where y_1 is the unique positive solution of $\phi_1(x) = x$ and α is any number in the interval $(1, 1/\beta)$. Further, for this choice of $w(x)$, if $M(q)$ satisfies $x^q \leq M(q)w(x)$ and $C(x) = \sum_{k=1}^m a_k x^{q_k}$ for some $a_k \in \mathbb{R}$ and $q_k \geq -1$, then the triangle inequality gives $|\sum_{k=1}^m a_k x^{q_k}| \leq (\sum_{k=1}^m |a_k| M(q_k))\, w(x)$. Thus part (v) is satisfied for all the examples of the function $C(\cdot)$ considered above. Given the above assumption, we may now answer our first question. The full theorem (see page 14), of which the following is only an excerpt, also characterises the optimal choice of threshold and the associated Whittle index.

Theorem 1 (Excerpt) *Suppose A1 holds. Then for some threshold $s \in \bar{\mathbb{R}}$, the threshold policies $A_t = \mathbf{1}_{X_t > s}$ and $A_t = \mathbf{1}_{X_t \geq s}$ both minimise performance functional (5) for every initial state $x \in \mathcal{I}$.*

Theorem 1 only holds for cost functions that are functions of the posterior variance $v_t = \mathbb{E}[(Z_t - z_t)^2 | H_t]$ alone, although this includes non-quadratic expressions like $\mathbb{E}[|Z_t - z_t|^p | H_t]$ for $p > -1$, which is of the form $k(p)v_t^p$ for an appropriate function $k(\cdot)$. For cost functions that explicitly depend on the posterior mean, we have a Markov decision problem in a 2-dimensional real-valued state space and there are several possible extensions of the notion of a "threshold policy" that might be considered. Also, as transitions of the posterior mean depend on the values of the observations, the problem would no-longer be deterministic if dependence on the posterior mean were introduced.

From one perspective, this answer is a rare example of an explicit solution to a real-state partially-observed Markov decision process (POMDP). From another perspective, this answer is a rare example of an explicit solution to the problem of observation selection in sensor management (Hero and Cochran, 2011). Indeed, given a collection of variables which can (in principle) be observed and a single variable to predict, which are jointly Gaussian with known covariance, even the problem of deciding whether there exists a subset of k observations that reduces the prediction variance below a given threshold is NP-hard (Davis et al., 1997). Work has therefore focused on finding covariance structures for which the

problem is tractable, for instance Das and Kempe (2008) show that selection of Gaussian observations with an exponential covariance can be solved by a simple discrete dynamic program, and on finding appropriate choices of cost functions for which there are guaranteed approximation algorithms (Krause et al., 2008, 2011; Badanidiyuru et al., 2014; Chen et al., 2014).

1.3. The Linear Quadratic Gaussian Problem with Costly Observations

The second question addressed by this paper is: *when are threshold policies optimal for making observations in a generalisation of the linear-quadratic-Gaussian control problem in which observations are costly but controlled through a query action?* Specifically, suppose the states and observations are as in (1) but the objective is to find a non-anticipative policy π that selects a feedback-control action $u_t \in \mathbb{R}$ and a sensor-query action $a_t \in \{0, 1\}$ so as to minimise the β-discounted performance functional

$$\mathbb{E}^{\pi}_{z_0,v_0}\left[\sum_{t=0}^{\infty} \beta^t (DZ_t^2 + Fu_t^2 + \lambda A_t)\right],$$

where $D, F \in \mathbb{R}_+$ and the expectation $\mathbb{E}^{\pi}_{z_0,v_0}$ is over the Markovian transitions (2) under policy π when the initial state Z_0 is normally distributed with mean z_0 and variance v_0. In this expression $\lambda \in \mathbb{R}$ denotes a cost to be paid each time action $A_t = 1$ is taken, just as in (5), and if $\lambda = 0$ then the problem reduces to the classical linear-quadratic-Gaussian control problem.

The problem with price $\lambda \neq 0$ is an old but unsolved problem, although it has been known for a long time that in finite-horizon versions of the problem the optimal timing of observations can in principle be determined *a priori*, and does not depend on the values of the observations (Kushner, 1964; Meier et al., 1967). Thus, the subproblems of state-estimation, observation scheduling and control can be decoupled, and this is often known as the *separation principle*. Meier et al. (1967) addressed the observation scheduling sub-problem using a tree search for forward dynamic programming. Other numerical attacks on the problem have included formulating it as a two-point boundary value problem, whether in continuous time (Athans, 1972) or discrete time (Kerr, 1981). Wu and Arapostathis (2008) study the long-term average and infinite-horizon discounted version of the problem and prove that a separation principle still holds over the infinite horizon, while Molin and Hirche (2009) extended these results to a setting where measurements are subject to random packet losses. Zhao et al. (2014) studied the infinite-horizon average-cost version of the problem, showing that optimal policies can be approximated arbitrarily closely by periodic schedules, but without giving any insight into the structure of such periodic schedules, or results about the discounted case.

Studies of linear quadratic control with costly observations were initially motivated by aerospace applications in which telemetry data from space vehicles was transmitted over band-limited links to ground stations (Kushner, 1964; Meier et al., 1967). More recently, continued research has been motivated by applications to networked control systems for unmanned aerial vehicles (Seiler, 2001), vehicle control (Daoud et al., 2006) and teleoperation (Hirche et al., 2007).

An immediate corollary of Theorem 1 is the following answer to the above question.

Corollary 2 *Suppose $\bar{A} \in [-1, 1]$, $\bar{B} \in \mathbb{R}\backslash\{0\}$, $D \in \mathbb{R}_{++}$, $F \in \mathbb{R}_{+}$, $\beta \in (0, 1)$, $\Sigma_Y(a) \in [0, \infty]$ for $a \in \{0, 1\}$ with $\Sigma_Y(0) \geq \Sigma_Y(1)$, and that $\lambda \in \mathbb{R}$. Then, there exists a threshold $s \in \overline{\mathbb{R}}$ such that an optimal policy for the problem of linear-quadratic-Gaussian control with costly observations is to set*

$$a_t = \begin{cases} 1 & \text{if } v_t \geq s \\ 0 & \text{if } v_t < s \end{cases} \qquad \text{and} \qquad u_t = -\left(\frac{\bar{A}}{\bar{B} + \frac{F}{\beta \bar{B} R}}\right) z_t$$

whatever the initial state, where R is the unique positive root of the quadratic equation

$$-\beta \bar{B}^2 R^2 + (\beta \bar{B}^2 D + \beta \bar{A}^2 F - F)R + DF = 0.$$

A proof of Corollary 2 is presented in Section 3.6.

1.4. Multi-Target Tracking and Restless Bandits

This paper also addresses the problem of monitoring *multiple* time series so as to maintain a precise belief while imposing a *constraint* on the number of time series that may be sensed at each time. This problem is often called the *multi-target tracking* problem. Multiple heuristics have been proposed for this problem. The simplest heuristics include round-robin schedules, and greedy or myopic schedules (Oshman, 1994). More sophisticated heuristics exploit probabilistic sensor allocations based on "steady-state" covariance matrices in continuous time (Mourikis and Roumeliotis, 2006; Le Ny et al., 2011) or in discrete time (Gupta et al., 2006). Also, Whittle (1988) proposed a *restless bandit* heuristic, and one of the applications of that heuristic is to the multi-target tracking problem. As that heuristic is a major focus of the current paper, we review the literature on the restless bandit approaches to multi-target tracking straight after explaining what restless bandits are below.

One example of a real-world application of the discrete-time problem, which was our original motivation for studying the problems in this paper, is the measurement of on-street parking occupancy (Dey, 2014), in a setting where cheap-but-low-quality observations are available through payment data (at parking meters or through mobile phones), expensive-but-high-quality observations are available through portable cameras, which are moved daily or weekly (and thus in discrete time), and there are a limited number of portable cameras with which to observe many streets.

To formulate the problem, suppose there are $n \in \mathbb{Z}_{++}$ independent time series of the form (1), indexed by $i \in \{1, 2, \ldots, n\}$, and time series i has state $Z_{t,i}$ at time $t \in \mathbb{Z}_{+}$. Each time series may have its own parameters $z_{i,0}, v_{i,0}, \bar{A}_i, \bar{B}_i, \Sigma_{Z_i}$, and its own input $u_{i,t}$. As in Section 1.1, we scale the posterior variance of each time series to get a variance state $x_{i,t}$ on a state space $\mathcal{I}_i$. As in Section 1.2, each time series has its own uncertainty cost $C_i : \mathcal{I}_i \to \mathbb{R}$. Corresponding to these time series there are n query actions $a_{i,t} \in \{0, 1\}$ at each time t which specify the nature of the observation $Y_{i,t}$ of time series i. These observations have their own parameters $\Sigma_{Y_i} : \{0, 1\} \to (0, \infty]$. However, these actions are subject to the constraint that only $m \in \mathbb{Z}_{++}$ with $m < n$ expensive observations can be made at each time. As in Section 1.1, the transitions of the posterior variance are given by the Möbius transformation (4), which does not involve the exogenous inputs $u_{i,t}$, and which is deterministic.

The problem is then to find a history-dependent randomised policy π that minimises the total β-discounted uncertainty cost

$$\mathbb{E}_x^{\pi}\left[\sum_{i=1}^{n}\sum_{t=0}^{\infty}\beta^t C_i(X_{t,i})\right]$$

for any initial state $x \in \mathcal{I}_1 \times \cdots \times \mathcal{I}_n$, subject to the constraint that policy π makes m observations at each time, so that

$$\sum_{i=1}^{n} A_{t,i} = m \qquad \text{for } t \in \mathbb{Z}_{+},$$

where $\mathbb{E}_x^{\pi}$ denotes the expectation over sequences $(X_t, A_t)_{t=0}^{\infty}$ with initial state $X_{0,1} = x_1, \ldots, X_{0,n} = x_n$, where actions are taken according to the potentially non-deterministic policy π and transitions are according to (4). It is equally possible to work with the constraint $\sum_{i=1}^{n} A_{t,i} \leq m$ as discussed below.

Restless Bandits. The multi-target tracking problem is an instance of a *restless bandit problem* (Whittle, 1988). Typically, such problems are defined in terms of a set of $n \in \mathbb{Z}_{++}$ two-action Markov decision processes (MDPs), although generalisations to a time-varying number of MDPs (Verloop, 2016) and to more than two actions per MDP (Glazebrook et al., 2011) have been explored. The two actions are usually referred to as *active* or *play* versus *inactive* or *passive* and each of the MDPs is referred to as an *project* or *arm*.

In a restless bandit problem, these n MDPs are coupled into a single MDP as follows. The state space is the Cartesian product of the state spaces of the projects, and the state of each project transitions independently of the other projects given the actions taken on that project. Thus the transitions of a project depend only on the actions taken on that project and on that project's current state. The objective is to find a non-anticipative policy that minimises the sum of the projects' individual performance functionals if those functionals all represent costs (or that maximises the sum of the projects' individual performance functionals if those functionals all represent rewards), for all initial states. The precise notion of the performance functional for an individual project depends on the setting: infinite horizon average cost and infinite horizon discounted cost settings are both commonly considered.

However, the action space is only a subset of the Cartesian product of the action spaces of the projects, as there is a constraint on the number m of projects that are simultaneously active at each time, where $m \in \mathbb{Z}_{++}$ with $m < n$. Typically, the constraint is that *exactly* m projects are active at each time, but this is readily relaxed to a constraint that *at most* m projects are active by including "dummy projects", whose cost is always zero, in the population of n projects. More general constraints have been explored (Niño-Mora, 2015), in which each project consumes resources as a function of both its state and the action taken, and the total cost of the resources consumed at each time is constrained. In the absence of any such action constraint, the problem would be solved by applying an optimal policy for each project independently. Moreover, it turns out that if the constraint were only on the (discounted) time-average number of projects that are simultaneously active, rather than a constraint at each time, the problem could again be separated into n smaller problems

after introducing a Lagrange multiplier. Indeed, this observation was one motivation for the Whittle-index approach first proposed in Whittle (1988), as discussed below.

Let us relate the above definition to the typical usage of the term *bandit* in the machine-learning literature. In that context, multi-armed bandits are reinforcement-learning problems involving a set of projects whose reward distributions are unknown. At each time, the learner must select which project to play. Such bandits involve a trade-off between exploring projects to acquire information about their expected payoffs and exploiting projects with the highest expected payoffs. In the simplest versions of such problems, where the prior on the reward distributions is independent over projects, each project can be viewed as an MDP whose state corresponds to the belief about that project's payoff distribution. Each time the project is played, its reward is observed and this belief is updated. Such updates correspond to state transitions. Each time the project is inactive, its state does not change.

If we allow projects to make general Markovian state transitions, not just transitions corresponding to belief updates, while preserving the requirement that a project only changes state when it is played, then we arrive at a more general class of problems known as *ordinary* or *classical bandits* (Gittins et al., 2011). In turn, restless bandits generalise ordinary bandits in two ways. Firstly, restless bandits allow more than one project to be simultaneously active (if $m > 1$). Secondly, restless bandits allow the state of a project to change even when the project is not active, which is why they are called *restless*.

While this additional generality is important in modelling real-world problems, it comes at a price. On the one hand, the *Gittins index policy* is optimal for ordinary bandit problems and can be computed in polynomial time for problems with finite state spaces (Niño-Mora, 2007). On the other hand, it is in general PSPACE-hard (Papadimitriou and Tsitsiklis, 1999; Guha et al., 2010) to find policies that approximate optimal policies for restless bandit problems with finite state spaces to any non-trivial factor. At first glance, this might suggest that the multi-target tracking problem addressed here, with uncountable state-space $\mathbb{R}_+$ or $\mathbb{R}_{++}$, is impossibly difficult. At second glance, this poses an interesting question: for which restless bandit problems can we find approximately-optimal policies efficiently?

Whittle Index Policy. Whittle (1988) proposed a policy which generalises the Gittins index policy to restless bandit problems. This policy associates a real (or in some definitions an extended-real) number $\lambda_i^*(x_i)$ called the *Whittle index* with the state x_i of each project i. The policy then plays the m projects with the largest Whittle indices at each time, or for restless bandits with more general constraints, it selects a subset of projects with the largest Whittle indices such that the constraint is met. Ties are usually broken uniformly at random or according to a predefined priority ordering.

Whittle's index policy has been the subject of great interest for computational, empirical and theoretical reasons. The policy is potentially attractive in terms of computational cost as it reduces the original restless bandit problem, whose state space is the Cartesian product of the state spaces of the projects, to the computation of n Whittle indexes for individual projects. The policy is also attractive from a systems-architecture point-of-view, as it allows one to mix-and-match different types of projects, and it naturally accommodates the arrival or departure of projects in the sense that the Whittle index does not depend on the number of projects n. Additionally, extensive numerical tests of Whittle's policy in different applications repeatedly demonstrate that it performs remarkably well when the projects are

all indexable. Indeed, 12 references to such empirical work are cited in Section 8 of Verloop (2016).

The literature contains several definitions of the Whittle index $\lambda_i^*(x_i)$ of project i, which are not equivalent in general, although they turn out to be equivalent for the problem addressed in this paper. The definition in Whittle (1988) is slightly informal and does not clearly distinguish between these definitions. All the definitions involve a modified version of project i's MDP, called the *λ-price problem*. For restless bandits with a constraint on the number m of projects that can be simultaneously active, the λ-price problem involves replacing the reward $r_i(x_i, a_i)$ for taking action a_i in state x_i by $r_i(x_i, a_i) - \lambda a_i$ where $\lambda \in \mathbb{R}$ represents a price for taking the active action $a_i = 1$. Verloop (2016) then defines $\lambda_i^*(x_i)$ as the least price λ for which action $a_i = 0$ is optimal for the λ-price problem in state x_i. Meanwhile, Guha et al. (2010) define $\lambda_i^*(x_i)$ as the largest price λ for which the actions $a_i = 0$ and $a_i = 1$ are both optimal for the λ-price problem in state x_i.

In this paper, we use the following definition (Niño-Mora, 2014, 2015) which applies to restless bandits with general resource-consumption constraints. Let us drop the subscript i and consider a single project $\mathcal{P} = \langle \mathcal{X}, c, r, \mathscr{Q}, \beta \rangle$ with state space $\mathcal{X}$, resource function $c : \mathcal{X} \times \{0, 1\} \to \mathbb{R}$, reward function $r : \mathcal{X} \times \{0, 1\} \to \mathbb{R}$, transition law $\mathscr{Q}$ and discount factor $\beta \in [0, 1)$, with the following interpretation. At the start of period $t \in \mathbb{Z}_+$ the state $X_t \in \mathcal{X}$ is observed and an action $A_t \in \{0, 1\}$ is chosen. If $X_t = x$ and $A_t = a$ then $c(x, a)$ units of resource are consumed (the projects of the previous paragraph would have $c(x, a) = a$), the project yields a reward $r(x, a)$ and the state transitions to X_{t+1} which has distribution $\mathscr{Q}(\cdot|x, a)$. Let Π denote the class of all history-dependent randomised policies for project $\mathcal{P}$ and let $\mathbb{E}_x^\pi$ denote the expectation over sequences $(X_t, A_t)_{t=0}^\infty$ with initial state $X_0 = x$, where actions are taken according to policy π and transitions are according to $\mathscr{Q}$. Then, the *λ-price problem* for project $\mathcal{P}$ and price $\lambda \in \mathbb{R}$ is to find a policy $\pi_\lambda^* \in \Pi$ that maximises the performance functional

$$\mathbb{E}_x^\pi \left[\sum_{t=0}^{\infty} \beta^t \left(r(X_t, A_t) - \lambda c(X_t, A_t) \right) \right]$$

for all initial states $x \in \mathcal{X}$.

Definition 3 ***(Niño-Mora, 2014, 2015)***

The ***Whittle index*** *of project $\mathcal{P}$ in state x is a price $\lambda^*(x)$ for which*

1. *Action $a = 1$ is optimal in state x of the λ-price problem if and only if $\lambda \leq \lambda^*(x)$,*
2. *Action $a = 0$ is optimal in state x of the λ-price problem if and only if $\lambda \geq \lambda^*(x)$.*

Project $\mathcal{P}$ is ***indexable*** *if it has a Whittle index $\lambda^*(x)$ for all states x in its state space.*

For all of the above definitions, it is immediate that the Whittle index is unique if it exists. Verloop's definition has the advantage that the Whittle index, and hence the Whittle index policy, exist for a wider range of projects. On the other hand, if we know project i is indexable, the definition used in this paper has the advantage that we know we have found the Whittle index when we find a price $\lambda \in \mathbb{R}$ for which actions $a_i = 0$ and $a_i = 1$ are both optimal in state x_i of the λ-price problem.

Although Whittle's policy is not an optimal policy for general restless bandits, under certain sufficient conditions and for a certain limit, it is an *asymptotically* optimal policy. Specifically, in the limit as the number of projects n tends to infinity, while the number of projects that can be simultaneously active m varies in such a way that m/n is as constant as possible, the ratio of the cost-rate of Whittle's policy to the cost-rate of an optimal policy for the given n, m tends to one. Assuming an average-cost setting, for collections of identical projects whose size n does not vary with time, where each project has a finite state space, Whittle (1988) originally conjectured that it was sufficient that the identical project was indexable for such an asymptotic optimality result to hold. However, Weber and Weiss (1990) found counterexamples to this conjecture. Nevertheless, Weber and Weiss also found sufficient conditions for asymptotic optimality to hold, and those sufficient conditions imply that the projects are indexable (Lemma 2 of that paper). Under similar sufficient conditions, this asymptotic optimality result has recently been generalised by Verloop (2016) to restless bandits with dynamic populations of non-identical projects. Both the results of Weber and Weiss and the results of Verloop assume an average-cost setting and projects with finite state spaces. So new theoretical work may be required to understand asymptotic optimality for projects with uncountable state spaces, as studied here.

The Whittle index for a project is often written as the ratio of a *marginal reward* metric to a *marginal resource* metric, where:

- The *marginal reward* is the (discounted) reward-to-go by taking the active action ($a = 1$) then following an optimal policy minus the (discounted) reward-to-go by taking the passive action ($a = 0$) then following an optimal policy;
- and the *marginal resource* is the (discounted) resource-to-go by taking the active action ($a = 1$) then following an optimal policy minus the (discounted) resource-to-go by taking the passive action ($a = 0$) then following an optimal policy;
- with the additional complexity that *optimal policy* here means optimal for the λ-price problem where λ equals the Whittle index in state x.

For examples of such expressions, see for instance: Niño-Mora (2002, equation (4.14), Theorem 4.7 and Section 6), Niño-Mora (2006, equation 19), Niño-Mora (2007, equation 6), Gittins et al. (2011, Theorem 6.4) and Larrañaga et al. (2016, Proposition 2). In particular, Niño-Mora (2002) appears to have been the first ever paper to use such marginal metrics to study questions of indexability, it also introduced resource metrics that are more general than those considered by Whittle (1988) and it established indexability of a general birth-death model. The intuition behind this expression is as follows. By definition, the Whittle index for state x corresponds to a price λ that makes both action 0 and action 1 optimal when in state x in the λ-price problem. Now, if both of these actions are optimal, then they are *equally good*. Let the reward-to-go by following policy π for a project $\mathcal{P} = \langle \mathcal{X}, c, r, \mathscr{Q}, \beta \rangle$ be denoted by

$$F(x, \pi) := \mathbb{E}_x^\pi \left[\sum_{t=0}^{\infty} \beta^t r(X_t, A_t) \right], \tag{7}$$

let the resource-to-go be denoted by

$$G(x, \pi) := \mathbb{E}_x^\pi \left[\sum_{t=0}^{\infty} \beta^t c(X_t, A_t) \right],$$

let $\langle a, \pi \rangle$ be the policy that first takes action a then follows policy π at subsequent times and let π_λ^* denote an optimal policy for the λ-price problem for project $\mathcal{P}$. Then the condition that the price λ makes actions 0 and 1 *equally good* in state x reads

$$F(x, \langle 0, \pi_\lambda^* \rangle) - \lambda G(x, \langle 0, \pi_\lambda^* \rangle) = F(x, \langle 1, \pi_\lambda^* \rangle) - \lambda G(x, \langle 1, \pi_\lambda^* \rangle).$$

This rearranges to give

$$\lambda = \frac{F(x, \langle 1, \pi_\lambda^* \rangle) - F(x, \langle 0, \pi_\lambda^* \rangle)}{G(x, \langle 1, \pi_\lambda^* \rangle) - G(x, \langle 0, \pi_\lambda^* \rangle)} \tag{8}$$

which is the ratio of a marginal reward to a marginal resource, as claimed. Unfortunately, this expression is usually only an *implicit* expression for λ, since the right-hand side involves π_λ^*. However, for projects of the multi-target tracking problem, this turns out to be an explicit expression, as we shall see below.

Literature on Restless Bandit Approach to Multi-Target Tracking. Whittle (1988) mentioned the problem of m aircraft tracking $n > m$ submarines as an example of a restless bandit problem. That problem seems rather an interesting challenge as it seems to couple multi-target tracking with a pursuit-evasion game. Nevertheless, the idea of taking a restless-bandit approach to multi-target tracking was discussed many times before anyone even experimented with such an approach, at least in the public literature. For instance, La Scala and Moran (2006) pointed out the potential interest of a restless multi-armed bandit approach to the multi-target tracking problem, but they did not pursue the Whittle index approach, rather focussing on trying to find conditions under which a one-step greedy policy is optimal when tracking a pair of targets with a single sensor. Also, Washburn (2008) reviewed applications of multi-armed bandit approaches to partially-observed sensor-management problems, expressing doubts as to whether Whittle's indexability conditions typically hold for such problems.

In contrast to such doubts, Le Ny et al. (2011) claimed to have found conditions under which the continuous-time version of the problem is indexable, at least for a scalar state, in the average-cost case, and with posterior variance as the cost function. A closer reading of that paper reveals that those authors present an incomplete argument leading to hypothetical values for optimal thresholds under the assumption that threshold policies are optimal. In particular, a key step of that argument is those authors' Theorem 1, which is Proposition 8 of Whittle (1988). This result gives an expression for the Whittle index, which the authors then invert to find a cubic equation for an optimal threshold. Now, as remarked by Whittle (1988, p. 295), *"that argument is formal, and conditions are certainly required for the calculations to make sense"*. However, Le Ny et al. (2011) do not try to find a suitable set of such conditions. Further, those authors *assume* the form of the optimal policy hoping that it can be verified a posteriori by substituting their expressions for an optimal threshold into the dynamic programming equation. However, they make no attempt to

perform such a substitution and in fact such a verification may prove challenging since the thresholds are given by a cubic equation, and because the dynamic programming equation involves the trajectory of the posterior variance which is the solution of a Ricatti differential equation. In summary, the arguments presented in Le Ny et al. (2011) are incomplete and do not convincingly demonstrate the optimality of threshold policies and the indexability for continuous-time systems.

Meanwhile, discrete-time versions of the problem have proved to be far more challenging. Niño-Mora and Villar (2009) made the first empirical evaluation of Whittle's index policy applied to the multi-target tracking problem with a scalar state in discrete time. That paper claims that the problem is indexable and suggests an approach to studying its indexability based on partial conservation laws, which is the approach that we take in Section 3 of this paper. However, it provides no proof or argument to substantiate this claim. The dissertation of Villar (2012) began a theoretical investigation of the problem, without establishing indexability of the model of concern. More recently, Dance and Silander (2015) proved that the index function is a monotone function of the variance state, but they did so under the assumption that threshold policies are optimal.

A number of closely-related problems have also been explored. For instance, Le Ny et al. (2008) and Liu and Zhao (2010) both establish indexability results about the tracking of n targets with *binary* state-spaces given $m < n$ sensors. Meanwhile, Niño-Mora (2016) empirically explored a Whittle's index approach to a generalisation of the multi-target tracking problem in which measurements are randomly jammed.

Whittle Index for the Multi-Target Tracking Problem. The above discussion prompts the third question addressed in this paper: *is each project of the multi-target tracking problem indexable, and if so, what is a computationally-convenient expression for the Whittle index of a project?* To state the first part of this question explicitly, consider a single project of the multi-target tracking problem, corresponding to one of n time series to be tracked with $m < n$ sensors. For each $\lambda \in \mathbb{R}$ and initial state $x \in \mathcal{I}$, the corresponding λ-price problem then to minimise the performance functional

$$\mathbb{E}_x^\pi \left[\sum_{t=0}^{\infty} \beta^t \left(\lambda A_t + C(X_t) \right) \right] \tag{9}$$

with respect to the policy π for taking actions A_t where the state-sequence is given in terms of the variance updates of (4) as

$$X_0 = x, \qquad\qquad X_{t+1} = \phi_{A_t}(X_t),$$

for $t = 0, 1, \ldots$. The question is then whether there exists a simple-to-compute function $\lambda^* : \mathcal{I} \to \mathbb{R}$, for which action $A_t = 1$ is optimal if and only if $\lambda \leq \lambda^*(X_t)$, while action $A_t = 0$ is optimal if and only if $\lambda \geq \lambda^*(X_t)$, for all $t \in \mathbb{Z}_+$.

As suggested by Theorem 1 (Excerpt), to state a convenient expression for the Whittle index, some definitions concerning threshold policies should be useful. So, for any threshold $s \in \overline{\mathbb{R}}$, let the *s-threshold policy* be the policy that takes the action 1 if the state exceeds the threshold s and takes action 0 otherwise. Also, let $X_t(x, a; s)$ denote the state at time $t = 0, 1, \ldots$ if the system starts in state $x \in \mathcal{I}$ at time $t = 0$, then action $a \in \{0, 1\}$ is taken

and the s-threshold policy is followed thereafter, so that $A_t(x, a; s) := \mathbf{1}_{X_t(x,a;s)>s}$ for $t > 0$. The answer to our third question is then given by the following theorem, whose proof is given in Section 3.

Theorem 1 *Suppose A1 holds. Then the family of λ-price problems, given by equation (9), is indexable and for each $x \in \mathcal{I}$ the Whittle index is*

$$\lambda^*(x) := \frac{\sum_{t=0}^{\infty} \beta^t (C(X_t(x, 0; x)) - C(X_t(x, 1; x)))}{\sum_{t=0}^{\infty} \beta^t (A_t(x, 1; x) - A_t(x, 0; x))}.$$

Furthermore,

1. *If $\lambda^*(s) = \lambda$ for some $s \in \mathcal{I}$ then both of the following threshold policies are optimal:*

$$A_t = \mathbf{1}_{X_t > s}, \qquad\qquad A_t = \mathbf{1}_{X_t \geq s};$$

2. *If $\lambda^*(s) > \lambda$ for all $s \in \mathcal{I}$ then the always-active policy is the unique optimal policy;*
3. *If $\lambda^*(s) < \lambda$ for all $s \in \mathcal{I}$ then the always-passive policy is the unique optimal policy.*

This paper thus generalises the work of Dance and Silander (2015) by demonstrating that threshold policies are in fact optimal for the single project problem, which was Assumption A1 of (Dance and Silander, 2015). It also generalises by considering the case of multipliers $\bar{A} < 1$ rather than only considering $\bar{A} = 1$, where $\bar{A}$ is as in equation (1), and by considering cost functions $C(x) \neq x$ other than the (scaled) posterior variance.

1.5. Intuitive Guide to the Paper

As with other work on Markov decision processes, we work with the cost-to-go $Q(x, a)$ when starting in initial state x and taking initial action a, but then following an optimal policy. A common way to prove that threshold policies are optimal when the state x is real-valued, is to show that the difference $Q(x, 1) - Q(x, 0)$ is a non-increasing function of x. Such approaches have been studied by Serfozo (1976), Altman and Stidham Jr. (1995) and Altman et al. (2000). Unfortunately, as shown in Figure 1, such an approach fails for the process considered in this paper, even when the cost equals the variance.

Instead, this paper proves the optimality of threshold policies using a new verification theorem by Niño-Mora (Niño-Mora, 2015, 2019). This theorem applies to Markov decision processes that satisfy the so-called *partial conservation law indexability* (PCLI) conditions (Section 3). The central concept underlying the verification theorem is the *marginal productivity index* which turns out to be equal to the ratio $\lambda^*(\cdot)$ given in Theorem 1.

One of the PCLI conditions requires that the marginal productivity index is a non-decreasing and continuous function of the state x and that it is bounded from below. This is the most challenging of the conditions to verify. As a quick check, we plot $\lambda^*(x)$ in Figure 2. Although $\lambda^*(x)$ is increasing, the numerator and denominator have a fractal structure, so it is surprising that the index is continuous. Furthermore, if we subtract a cubic fit to $\lambda^*(x)$, the residual has a complicated sequence of cusps. Therefore the paper then focusses on characterising the sequence of actions $A_t(x, a; x)$ resulting from applying

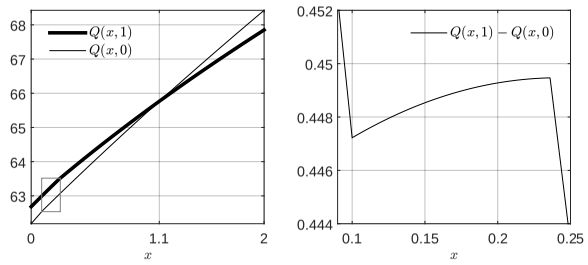


Figure 1: Counterexample to monotonicity of the difference in Q-functions. The functions $Q(x,0)$ and $Q(x,1)$ cross only a single time at $x = 1.1$ (*left plot*). However, the difference $Q(x,1) - Q(x,0)$ is increasing for some x (*right plot*, for x in the left plot's grey box). The model has $\beta = 0.95$, $C(x) = x$, $\phi_0(x) = x + 1$, $\phi_1(x) = 1/(\theta_1 + 1/(x+1))$ with $\theta_1 = 0.1$ and $\nu = 0.7647$.

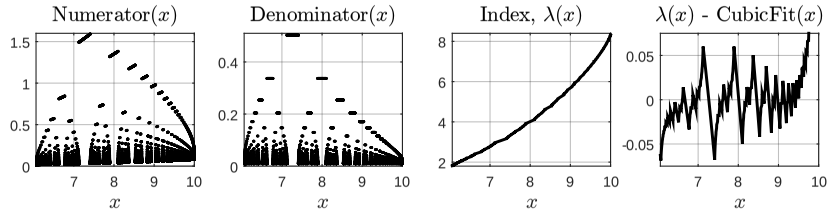


Figure 2: The numerator (*left*) and denominator (*mid-left*) of the index (*mid-right*), and the error in a cubic fit to the index (*right*). The model has cost $C(x) = x$, discount factor $\beta = 0.99$, map-with-a-gap $\phi_0(x) = r^2 x + 1$ and $\phi_1(x) = 1/(\theta_1 + 1/(r^2 x + 1))$ with $r^2 = 0.9$ and $\theta_1 = 0.01$.

an x-threshold policy, that give rise to this fractal pattern. We describe these sequences in terms of special binary strings that we call $\mathcal{M}$-*words* (Corollary 13 of Section 2). Such words include the set of all *Christoffel words* (Berstel et al., 2008) as well as a subset of the family of *Sturmian words* (Lothaire, 2002).

To understand what an $\mathcal{M}$-word is, consider the Cartesian coordinates $(x, y(x))$ of the points on a straight line $y(x) = \alpha x$ through the origin with slope $\alpha \in [0,1]$. To draw this line on a screen with pixel coordinates that are integer pairs, for each integer $\tilde{x}$ one might fill the pixel $(\tilde{x}, \tilde{y}(\tilde{x}))$ where $\tilde{y}(\tilde{x}) = \lfloor \alpha \tilde{x} \rfloor$. Now consider the differences $\Delta_{\tilde{x}} := \tilde{y}(\tilde{x}) - \tilde{y}(\tilde{x} - 1)$ for positive integers $\tilde{x}$. As α is in the range $[0,1]$ it follows that each such difference $\Delta_{\tilde{x}}$ is in the set $\{0,1\}$. Also, the sequence $\Delta := (\Delta_1, \Delta_2, \Delta_3, \dots)$ is periodic when α is rational and aperiodic when α is irrational. Correspondingly, the $\mathcal{M}$-*word of rate* α is the finite sequence $(\Delta_1, \Delta_2, \dots, \Delta_n)$ if Δ is periodic with (shortest) period n, and otherwise it is the infinite sequence Δ itself.

The characterisation of the action sequences $(A_t(x, a; x))_{t=0}^{\infty}$ viewed as a function of $x \in \mathcal{I}$ (noting that x represents both the initial state and the threshold) shows that the interval $\mathcal{I}$ can be partitioned into intervals corresponding to finite $\mathcal{M}$-words and points corresponding to infinite aperiodic $\mathcal{M}$-words. In the most important case for our analysis, x lies in such an interval and the denominator of the index involves periodic sequences of the form

$$\left.\begin{aligned}(A_t(x,1;x))_{t=0}^{\infty} &= (1, 0, p_1, \dots, p_n, 1, 0, p_1, \dots, p_n, \dots)\\ (A_t(x,0;x))_{t=0}^{\infty} &= (0, 1, p_1, \dots, p_n, 0, 1, p_1, \dots, p_n, \dots)\end{aligned}\right\} \tag{10}$$

for some binary sequence $(p_1, \dots, p_n)$. Thus the denominator equals

$$\sum_{t=0}^{\infty} \beta^t (A_t(x,1;x) - A_t(x,0;x)) = 1 - \beta + \beta^{n+2} - \beta^{n+3} + \dots = \frac{1-\beta}{1-\beta^{n+2}}.$$

Hence, the behaviour of the denominator in Figure 2 is given entirely by changes in the period $(n+2)$ of this binary sequence and does not depend on the values of $p_1, \dots, p_n$.

As this denominator equals a positive constant on such intervals, to show that the index is non-decreasing on such an interval, we show that the derivative of the numerator is non-negative. One key to proving this is the fact that the mappings $\phi_0(x)$ and $\phi_1(x)$ are Möbius transformations of the form

$$\mu_B(x) := \frac{B_{11}x + B_{12}}{B_{21}x + B_{22}}$$

for some $B \in \mathbb{R}^{2\times 2}$. Now the composition of Möbius transformations is homomorphic to matrix multiplication, so that

$$\mu_B(\mu_D(x)) = \mu_{BD}(x)$$

for any $B, D \in \mathbb{R}^{2\times 2}$. Further, if $\det(B) = 1$ then the derivative of the corresponding Möbius transformation is

$$\frac{d}{dx}\mu_B(x) = \frac{1}{(B_{21}x + B_{22})^2}$$

which is a convex function for $x \in \mathbb{R}_+$ and $B_{21}, B_{22} \in \mathbb{R}_{++}$. So the derivative of the numerator of the index, in the case $C(x) = x$, is of the form

$$\sum_{t=0}^{\infty} \frac{\beta^t}{(B_{21}^{(0,t)} x + B_{22}^{(0,t)})^2} - \sum_{t=0}^{\infty} \frac{\beta^t}{(B_{21}^{(1,t)} x + B_{22}^{(1,t)})^2}$$

where $(B^{(a,t)})_{t=0}^{\infty}$, for $a \in \{0, 1\}$, are sequences of 2×2 matrices that are determined by the action sequences $(A_t(x, a; x))_{t=0}^{\infty}$, but which otherwise do not depend on the value of x. This derivative is the difference of two sums, and each of these sums involves a sequence of convex functions $(z \mapsto \beta^t / z^2)_{t=0}^{\infty}$ applied to a sequence of linear functions of x. Inequalities involving such sums can be addressed by the theory of *majorisation* (Marshall et al., 2010), provided the sequences $(B_{21}^{(a,t)} x + B_{22}^{(a,t)})_{t=0}^{\infty}$ of linear functions of x satisfy certain *majorisation conditions* involving partial sums of those sequences. In the case where x lies in an interval such that the action sequences are given by (10), it turns out that those majorisation conditions are satisfied because the sequence $p_1, \ldots, p_n$ is a *palindrome*. That is, the sequence reads the same forwards as backwards, so that $p_k = p_{n-k}$ for $k = 1, \ldots, n-1$: see (11) for a proof of this well-known palindromic property of Christoffel words.

1.6. Structure of the Paper

First we relate the sequence of actions under threshold policies to $\mathcal{M}$*-words* (Section 2) before presenting proofs of our main results, Theorem 1 and Corollary 2 (Section 3). The proofs are based on Niño-Mora's theorem about the optimality of threshold policies (Section 3.1), which uses three *partial conservation law indexability (PCLI)* conditions. We use the properties of $\mathcal{M}$-words to demonstrate that each PCLI condition holds. These conditions concern the positivity of a *marginal resource* metric (Section 3.2), the continuity and non-decreasing nature of a *marginal productivity index* (Section 3.3), and a condition that characterises that index as a Radon-Nikodym derivative (Section 3.4). These results are then coupled into a proof of Theorem 1 (Section 3.5) and a proof of Corollary 2 is presented immediately thereafter (Section 3.6).

Having completed the proofs, we then turn to closed-form expressions for the index and numerical methods for evaluating it when such closed forms are not available (Section 4). We demonstrate the accuracy of such numerical methods and show how the index varies as its parameters change. Also, we compare the performance of Whittle's index policy with other well-known heuristics. Finally, we discuss interesting avenues for further work (Section 5). The appendices contain detailed proofs about the relation of itineraries to $\mathcal{M}$-words (Appendix A), of a key majorisation inequality (Appendix B) and about the linear systems orbits to which this majorisation result is applied (Appendix C).

2. Itineraries and Words

The transitions from state-to-state under threshold policies are given by a discontinuous mapping known as a *map-with-a-gap*, and the corresponding action sequence is known as the *itinerary* of that map. A detailed understanding of the properties of such itineraries is central to our proof of the optimality of threshold policies for the problems posed in the Introduction. So the main purpose of this section is to characterise these itineraries in terms

of special binary strings that we call $\mathcal{M}$*-words* and to discuss relevant properties of such words.

This section is structured as follows. We begin by introducing definitions and notations for itineraries (Section 2.1). As itineraries can be viewed as infinite strings, we then remind the reader of notations commonly used in the study of combinatorics on strings (Section 2.2). This notation enables us to define $\mathcal{M}$-words and an important subset of such words called the *Christoffel words* (Section 2.3). It also enables us to discuss properties of such words that are important for our proof of the optimality of threshold policies, notably a palindromic property (11), a description of the lexicographic ordering of the cyclic rotations of such words which are called *conjugates* (Lemma 8), and a description of the prefixes of $\mathcal{M}$-words (Lemma 10). Finally, we introduce three results (Theorem 12, Corollary 13 and Theorem 14) which describe itineraries in terms of $\mathcal{M}$-words, for a variety of relations between the threshold and initial state, and for versions of the threshold policy that are either active or passive at the threshold (Section 2.4).

2.1. Maps-with-Gaps

Many phenomena involve the iterated application of discontinuous maps or *maps-with-gaps* in the terminology of Hogan et al. (2007). Such phenomena are important in control problems (Haddad et al., 2014), in physics, electronics and mechanics (Bernardo et al., 2008; Makarenkov and Lamb, 2012), economics (Tramontana et al., 2010), biology and medicine (Aihara and Suzuki, 2010). Such maps either arise directly from a discrete-time model or they may arise as the Poincaré maps of continuous-time systems. The *orbit* and *itinerary* of the following definition, which is illustrated in Figure 3, are the standard way to describe the behaviour of such maps in the study of dynamical systems (Devaney, 2008).

Definition 5 *Consider an interval $\mathcal{I}$ of $\mathbb{R}$, functions $\phi_0 : \mathcal{I} \to \mathcal{I}$ and $\phi_1 : \mathcal{I} \to \mathcal{I}$, an initial state $x \in \mathcal{I}$ and a threshold $s \in \overline{\mathbb{R}}$. The* ***s*-threshold orbit** *from x for ϕ_0 and ϕ_1, denoted by* $\mathrm{orbit}(x|s, \phi_0, \phi_1)$*, is the sequence $(x_k)_{k=1}^{\infty}$ with*

$$x_1 = x \quad \textit{and} \quad x_{k+1} = \begin{cases} \phi_0(x_k) & \textit{if } x_k \leq s \\ \phi_1(x_k) & \textit{if } x_k > s. \end{cases}$$

In terms of that sequence, the ***s*-threshold itinerary** *from x for ϕ_0 and ϕ_1 is the infinite binary string $\sigma(x|s, \phi_0, \phi_1)$ with letters $\sigma(x|s, \phi_0, \phi_1)_k := \mathbf{1}_{x_k > s}$ for $k \in \mathbb{Z}_{++}$.*

Looking ahead, when we use the results of this section to prove Theorem 1, the mappings in the above definition will represent the transitions of a (relative) variance state. Thus ϕ_0, ϕ_1 and the interval $\mathcal{I}$ will satisfy Assumption A1 after a change of coordinates (as discussed straight after Theorem 12). However, the analysis of this section and Appendix A applies to *any* real interval $\mathcal{I}$ if not otherwise stated, and in most cases the analysis will make Assumption A2.

In most cases, the maps ϕ_0 and ϕ_1 are clear from the context, so we talk of the s-threshold orbit from x, denoted by $\mathrm{orbit}(x|s)$, and the s-threshold itinerary from x, denoted by $\sigma(x|s)$. Also, we use of the s^-*-threshold orbit* from x, which is the left limit $\mathrm{orbit}(x|s^-) =$

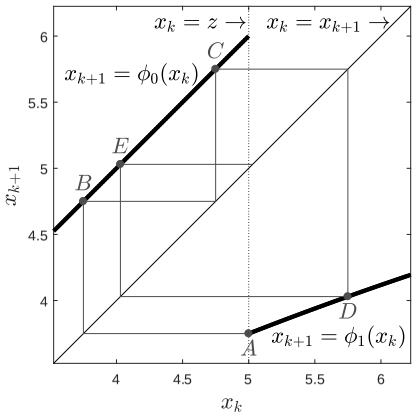


Figure 3: Map-with-a-gap. The s^--threshold orbit from $x = 5$ traces the path $ABCDE\ldots$ corresponding to the itinerary $10010\ldots$. The map has $\phi_0(x) = x + 1$, $\phi_1(x) = 1/(\theta_1 + 1/(x + 1))$ with $\theta_1 = 0.1$ and threshold $s = 5$.

$\lim_{s' \uparrow s} \text{orbit}(x|s') = (x_k)_{k=1}^{\infty}$ with

$$x_1 = x \quad \text{and} \qquad x_{k+1} = \begin{cases} \phi_0(x_k) & \text{if } x_k < s \\ \phi_1(x_k) & \text{if } x_k \geq s \end{cases}$$

and the corresponding *s^--threshold itinerary* from x, with letters $\sigma(x|s^-)_k := \mathbf{1}_{x_k \geq s}$ for $k \in \mathbb{Z}_{++}$. These left limits exist provided the maps ϕ_0, ϕ_1 are continuous, which is true when Assumption A2 holds (Lemma 32).

It is helpful to view such itineraries as *words* as we now explain.

2.2. Standard Definitions about Words

We remind the reader of standard definitions used in the study of the combinatorics of strings: see for instance Lothaire (2002) and Berstel et al. (2008).

In this paper, a *word* w is a string on the alphabet $\{0, 1\}$ and the empty word is denoted by ϵ. The *length* of a word w is the number of letters in the string, which is finite or countably infinite, and is denoted by $|w|$. The k^{th} letter of word w is w_k for $k \in \mathbb{Z}_{++}$ with $k \leq |w|$. Letters i-through-j of word w are denoted by $w_{i:j} := w_i w_{i+1} \ldots w_j$ for $i, j \in \mathbb{Z}_{++}$ with $i \leq j \leq |w|$. For $j < i$, we treat $w_{i:j}$ as the empty word. The *reverse* word of a finite word w is denoted by $w^R := w_{|w|} \ldots w_2 w_1$. A finite word satisfying $w^R = w$ is called a *palindrome*.

The *concatenation* of a finite word u and a word v is denoted by uv. For $n \in \mathbb{Z}_+$, the n-fold concatenation of a finite word w is denoted by w^n, with the convention that $w^0 = \epsilon$, and the word resulting from infinitely concatenating the word w is denoted by w^∞. For an infinite word w and $n \in \mathbb{Z}_{++}$ we define $w^n = w^\infty = w$.

A finite word f is a *factor* of a word w if $w = ufv$ for some finite word u and some word v. The number of times that word f appears in w, overlapping appearances included, is denoted by $|w|_f$. A finite word p is a *prefix* of word w if $w = ps$ for some word s and a word s is a *suffix* of word w if $w = ps$ for some finite word p.

We say that a word u is *lexicographically less than* a word v, written $u \prec v$, if either u is a finite word and $v = ua$ for some non-empty word a, or if $u = a0b$ and $v = a1c$ for some finite word a and some words b and c. We use $\succ$, $\preceq$ and $\succeq$ for the other lexicographic ordering relations.

We say an infinite word w is the *limit* of a sequence of words $(w^{(n)})_{n=1}^{\infty}$ and write $w = \lim_{n\to\infty} w^{(n)}$ if for each $i \in \mathbb{Z}_{++}$ there is an $n \in \mathbb{Z}_{++}$ such that $w_i = w_i^{(m)}$ for all $m \in \mathbb{Z}_{++}$ with $m \geq n$.

The *rate* of any finite non-empty word w is the ratio $\text{rate}(w) := |w|_1/|w|$ whereas for an infinite word w, when the limit exists, we define $\text{rate}(w) := \lim_{n\to\infty} |w_{1:n}|_1/n$. While some authors refer to such ratios as the "slope" of a word, we use the term "rate" as the "slope" of a word w is sometimes defined as the ratio $|w|_1/|w|_0$ and this seems justified from a geometrical point of view in terms of *digital straight lines* (Berstel et al., 2008).

Examples. For $w = 010111$ we have $|w| = 6$, $w_3 = 0$, $w_{2:4} = 101$, $|w|_{01} = |w|_{11} = 2$ and $w^2 = ww = 010111010111$. Also for $a = 01, b = 11$ we have $w = aab$ and $a \prec w \prec b$.

2.3. $\mathcal{M}$-Words

We characterise itineraries of maps-with-gaps in terms of the following type of words.

Definition 6 *The **$\mathcal{M}$-word** of rate $\alpha \in [0, 1]$ is the shortest word w such that*

$$(w^\infty)_n = \lfloor \alpha n \rfloor - \lfloor \alpha(n-1) \rfloor \quad \textit{for } n \in \mathbb{Z}_{++}.$$

*If α is rational then w is called a **Christoffel word**. If α is irrational then w is called a **Sturmian $\mathcal{M}$-word.***

Example. The shortest $\mathcal{M}$-words are the words $0, 1$ and 01 with rates $0, 1$ and $\frac{1}{2}$.

The notion of a Christoffel word is a standard one (Berstel et al., 2008). Also, *Sturmian words* have been studied intensively, for instance Lothaire (2002, Chapter 2) says an infinite word w is Sturmian if it has $n+1$ distinct factors of length n for any integer $n \geq 0$. Lothaire (2002) then goes on to explore many other definitions of Sturmian words that turn out to be equivalent. Our intention in defining $\mathcal{M}$-words is to group together the Christoffel words with a specific subset of Sturmian words, so that together they characterise the possible itineraries of a large class of maps-with-gaps, as described in Theorem 16.

We call such words $\mathcal{M}$-words as our definition is closely related to the set of *mechanical words*. For a given *slope* $\alpha \in [0, 1]$ and *intercept* $\rho \in \mathbb{R}$, Morse and Hedlund (1940) defined the *upper* and *lower mechanical words* to be the infinite sequences, for $n \in \mathbb{Z}_+$,

$$u_n = \lceil \alpha(n+1) + \rho \rceil - \lceil \alpha n + \rho \rceil$$

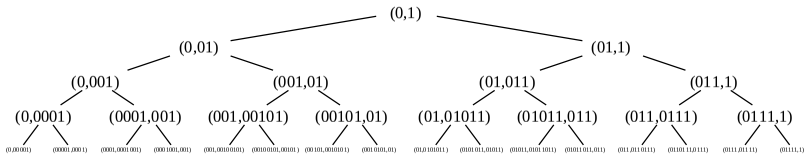


Figure 4: Part of the Christoffel tree.

$$l_n = \lfloor \alpha(n+1) + \rho \rfloor - \lfloor \alpha n + \rho \rfloor$$

Lothaire (2002) and Berstel et al. (2008) offer rich introductions to the mathematics of mechanical words, while Bousch and Mairesse (2002) and Altman et al. (2000) explore other optimisation problems that give rise to such words. Our $\mathcal{M}$-words are prefixes of lower-mechanical-words-of-zero-intercept up to a change of indexing from $n \in \mathbb{Z}_+$ to $n \in \mathbb{Z}_{++}$.

It is not hard to see that the Christoffel word of rate a/b, where a, b are relatively-prime integers, has length b. In contrast, Sturmian $\mathcal{M}$-words are infinite and aperiodic.

In general the $\mathcal{M}$-word w of rate α does have $\text{rate}(w) = \alpha$. Indeed if w is the $\mathcal{M}$-word of rate a/b for some $a, b \in \mathbb{Z}_{++}$, then

$$\text{rate}(w) = |w|_1/|w| = \lfloor (a/b)|w| \rfloor / |w| = a/b,$$

whereas, if w is an $\mathcal{M}$-word of irrational rate α, then

$$\text{rate}(w) = \lim_{n\to\infty} |w_{1:n}|_1/n = \lim_{n\to\infty} \lfloor \alpha n \rfloor / n = \alpha.$$

Furthermore, as remarked by Christoffel (1875), all Christoffel words other than the words 0 and 1 are of the form $0p1$ where the word p is a *palindrome*. Indeed for relatively-prime positive integers $m < n$, the letters of the Christoffel word w of rate m/n satisfy

$$w_{n-k} = \left\lfloor \frac{m}{n}(n-k) \right\rfloor - \left\lfloor \frac{m}{n}(n-k-1) \right\rfloor = \left\lfloor -\frac{m}{n}k \right\rfloor - \left\lfloor -\frac{m}{n}(k+1) \right\rfloor = w_{k+1} \tag{11}$$

for $k = 1, 2, \ldots, n-2$.

The Christoffel words can be defined in other ways. In this paper the most important alternative-but-equivalent definition is in terms of the *Christoffel tree* (Figure 4), which is an infinite complete binary tree (Berstel et al., 2008) in which each node is labelled with a pair (u, v) of words, called a *Christoffel pair*. The root of the tree is labelled with the pair $(0, 1)$ and the left and right children of node (u, v) are the nodes (u, uv) and (uv, v) respectively. In fact the Christoffel words are the words $0, 1$ and the set of concatenations uv for all (u, v) in the Christoffel tree.

Another definition of Christoffel words is in terms of modular arithmetic, as in the following lemma, where we use a bar to denote the remainder modulo the length $n = |w|$ of a Christoffel word w, so that $\overline{x} := x \bmod n$ for $x \in \mathbb{Z}$, and for any positive integer n, the set of integers modulo n is denoted by $\mathbb{Z}_n := \{0, 1, \ldots, n-1\}$. The following lemma does not explicitly appear in Berstel et al. (2008) but is easily related to multiple discussions of modular arithmetic in that book.

Lemma 7 *Suppose w is a Christoffel word of length n. Let $m := |w|_1$ and $p := |w|_0$. Then*

$$w_{i+1} = \mathbf{1}_{\overline{mi} \geq p} \qquad (i \in \mathbb{Z}_n).$$

Proof As $n\lfloor mi/n \rfloor = mi - \overline{mi}$, the definition of Christoffel words gives

$$\begin{aligned} w_{i+1} &= -\lfloor mi/n \rfloor + \lfloor m(i+1)/n \rfloor \\ &= (-mi + \overline{mi} + m(i+1) - \overline{m(i+1)})/n \\ &= (-mi + \overline{mi} + m(i+1) - (\overline{mi} + m - n\mathbf{1}_{\overline{mi} \geq n-m}))/n, \end{aligned}$$

which simplifies to $\mathbf{1}_{\overline{mi} \geq p}$, as claimed. ■

Finally, we give two results about $\mathcal{M}$-words that play a key role elsewhere in the paper. The first result is about *conjugacy* and lexicographic order. In particular, we say two finite words a and b are conjugate if $a = uv$ and $b = vu$ for some words u and v. For instance, the words $a = 00011$ and $b = 01100$ are conjugate. Words that are conjugate can be viewed as cyclic shifts of each other, like the X86 assembler instruction ROL. The notion of conjugate words is also standard in the study of the combinatorics of strings (Lothaire, 2002; Berstel et al., 2008). The following lemma is rather similar to Berstel et al. (2008, Exercise 6.3, p. 49), where an outline of its proof is suggested, although the result that we state is more specific about the lexicographic ordering of the conjugates.

Lemma 8 *Suppose w is a Christoffel word of length n and that l satisfies $\overline{lm} = 1$ where $m = |w|_1$. Then the conjugates $u(i) := w_{\overline{(i+1)}:n} w_{1:\overline{i}}$ satisfy*

$$w = u(0) \prec u(l) \prec u(2l) \prec \cdots \prec u((n-1)l) = w^R.$$

Furthermore, if the words c and d satisfy $w = 0dc1$, then $c01d$ and $c10d$ are lexicographically-consecutive conjugates of w.

Proof Let $x_i := \overline{mi - 1}$, $y_i := \overline{mi}$ and $p := n - m$. Then $x_0 = n - 1$ and $x_{n-1} = p - 1$. As $\gcd(m, n) = 1$, the sequence $x_0, \ldots, x_{n-1}$ is a permutation of $\mathbb{Z}_n$. So, $x_i \notin \{p-1, n-1\}$ for $i \in \{1, \ldots, n-2\}$. As $y_i = \overline{x_i + 1}$ these results give

$$\begin{aligned} \mathbf{1}_{x_i \geq p} &> \mathbf{1}_{y_i \geq p} && \text{for } i = 0 \\ \mathbf{1}_{x_i \geq p} &= \mathbf{1}_{y_i \geq p} && \text{for } i = 1, \ldots, n-2 \\ \mathbf{1}_{x_i \geq p} &< \mathbf{1}_{y_i \geq p} && \text{for } i = n-1. \end{aligned}$$

But Lemma 7 gives $u(0)_{j+1} = \mathbf{1}_{y_j \geq p}$ and $u((n-1)l)_{j+1} = \mathbf{1}_{x_j \geq p}$ for $j \in \mathbb{Z}_n$. Thus $u(0) = 0a1$ and $u((n-1)l) = 1a0$ for some word a. But $u(0) = w$ and w is a Christoffel word, so a is a palindrome. Therefore $u((n-1)l) = w^R$.

Now for $i = 0, \ldots, n-2$, the conjugates $u(il)$ and $u((i+1)l)$ are related to $u((n-1)l)$ and $u(0)$ respectively by the same non-zero cyclic rotation. Thus $u(il) = c01d$ and $u((i+1)l) = c10d$ for some words c and d with $dc = a$. Therefore $u(il) \prec u((i+1)l)$. ■

To illustrate this lemma, consider the Christoffel word $w = 00101$. As $n = |w| = 5$ and $m = |w|_1 = 2$, we set $l = 3$ so that $\overline{lm} = 1$. The sequence $(u(kl))_{k=0}^{n-1}$ of conjugates of w as defined in the lemma is then

$$\begin{aligned} u(0) &= 00101 \\ u(3) &= 01001 \\ u(6) &= 01010 \\ u(9) &= 10010 \\ u(12) &= 10100 \end{aligned}$$

which is indeed the same as the conjugates arranged in increasing lexicographic order. This lemma plays a key role in showing that the index function in Theorem 1 is non-decreasing in the case that the cost function $C(\cdot)$ is convex, by enabling the application of a rearrangement inequality at Steps (29) and (30) of the corresponding proof.

The second result shows how the prefixes of $\mathcal{M}$-words vary as a function of their rates. It requires one more definition, which is well known: see for instance Graham et al. (1994).

Definition 9 *For each positive integer n, the **Farey sequence** F_n is the sequence of rational numbers on $[0, 1]$ whose denominator is at most n, sorted in increasing order.*

Clearly, it is implicit in this definition that the numerator and denominator of each such rational number are relatively prime. For example, the Farey sequence F_5 is

$$0, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, 1.$$

Lemma 10 *Suppose $n \in \mathbb{Z}_{++}$ and $q \in [0, 1]$. Let $q_1 < q_2 < \cdots < q_m$ be the Farey sequence F_n. Let $p(s)$ be the first n letters of the word w^∞ where w is the $\mathcal{M}$-word of rate $s \in [0, 1]$. Then $p(q) = p(q_i)$ if and only if either $q = q_i = 1$ or $q \in [q_i, q_{i+1})$ for some $1 \le i < m$.*

Proof Let $b(q) := (\lfloor q \rfloor, \lfloor 2q \rfloor, \ldots, \lfloor nq \rfloor)$ and consider the intervals $\mathcal{Q}_i := [q_i, q_{i+1})$ for $i < m$ and $\mathcal{Q}_m := \{1\}$. As the line $y = qx$ hits an integer point $(x, y) \in \mathbb{Z}^2$ with $1 \le x \le n$ and $0 \le y \le x$ if and only if q is an element of F_n, it follows that $b(q) = b(q_i)$ if and only if $q \in \mathcal{Q}_i$. Let $g(x_1, x_2, \ldots, x_n) := (x_1, x_2 - x_1, \ldots, x_n - x_{n-1})$. By definition of $\mathcal{M}$-words, $p(q) = g(b(q))$. As g is invertible it follows that $p(q) = p(q_i)$ if and only if $q \in \mathcal{Q}_i$. ■

2.4. Characterising Itineraries with $\mathcal{M}$-Words

Our aim here is to characterise the itineraries of maps-with-gaps. We first set up some notation and then demonstrate a simple result about the lexicographical ordering of itineraries. Then we state an assumption under which Theorem 12 guarantees that itineraries correspond to specific $\mathcal{M}$-words.

Let $\mathcal{I}$ be an interval of $\mathbb{R}$ and consider two mappings $\phi_0 : \mathcal{I} \to \mathcal{I}$ and $\phi_1 : \mathcal{I} \to \mathcal{I}$. For any finite word w, the *composition* $\phi_w : \mathcal{I} \to \mathcal{I}$ is the mapping

$$\phi_w(x) := \phi_{w_{|w|}} \circ \cdots \circ \phi_{w_2} \circ \phi_{w_1}(x) \quad \text{and} \quad \phi_\epsilon(x) := x.$$

A simple application of compositions gives the following result about lexicographic ordering of the itineraries $\sigma(\cdot|s)$ of a map-with-a-gap given by mappings $\phi_0 : \mathcal{I} \to \mathcal{I}$ and $\phi_1 : \mathcal{I} \to \mathcal{I}$ and threshold s. We remind the reader that this paper uses *increasing* and *decreasing* in the strict sense.

Lemma 11 *Suppose ϕ_0, ϕ_1 are increasing mappings and that $x, y \in \mathcal{I}$ with either $\sigma(x|s^-) \prec \sigma(y|s^-)$ or $\sigma(x|s) \prec \sigma(y|s)$. Then $x < y$.*

Proof If $\sigma(x|s^-) \prec \sigma(y|s^-)$ then $\sigma(x|s^-) = a0b$ and $\sigma(y|s^-) = a1c$ for some finite word a and some infinite words b, c, by the definition of lexicographic order. So, the definition of $\sigma(\cdot|s^-)$ gives $\phi_a(x) < s \le \phi_a(y)$. But $\phi_a(\cdot)$ increasing as it is a finite composition of increasing functions. It follows that $x < y$. The proof for $\sigma(\cdot|s)$ is similar. ■

However, without additional assumptions about the mappings ϕ_0, ϕ_1, it is not possible to precisely characterise the itineraries of the associated maps-with-gaps. In order to state such an assumption, we shall say that a map $f : \mathcal{I} \to \mathcal{I}$, where $\mathcal{I}$ is an interval of $\mathbb{R}$ is *contractive* if for all $x, y \in \mathcal{I}$ with $x \ne y$, we have

$$|f(y) - f(x)| < |y - x|.$$

Also, a *fixed point* of such a map f is any $x \in \mathcal{I}$ for which the equation $x = f(x)$ is satisfied.

Assumption A2
Functions $\phi_0 : \mathcal{I} \to \mathcal{I}$ and $\phi_1 : \mathcal{I} \to \mathcal{I}$, where $\mathcal{I}$ is an interval of $\mathbb{R}$, are increasing, contractive and have unique fixed points y_0 and y_1 on $\mathcal{I}$ which satisfy $y_1 < y_0$.

Not all the maps-with-gaps considered in this paper satisfy Assumption A2 directly. For instance, the function $\phi_0(x) = x + 1$, which corresponds to Kalman-filter update of equation (3) in the case of an uninformative observation (for $A = 1, \Sigma_X = 1$ and $\Sigma_Y(0) \to \infty$), is not contractive. This can be addressed by a change of coordinates, and is discussed in the remark following Theorem 12.

Before using Assumption A2 to characterise the itineraries of maps-with-gaps, we must first clarify the notion of fixed points. A *fixed point* for a finite word w, is a solution to the equation $x = \phi_w(x)$. If Assumption A2 holds, then Lemma 32 in Appendix A.1 shows that there is a unique such fixed point on $\mathcal{I}$ for any finite non-empty word w and we shall denote it by y_w.

In general, it is not clear what a "fixed point" corresponding to an *infinite* word w might mean. One approach might be to consider a sequence $(w^{(n)})_{n=1}^\infty$ of words with $w = \lim_{n\to\infty} w^{(n)}$ and to define "y_w" as $\lim_{n\to\infty} y_{w^{(n)}}$ if that limit exists. However, for any word b, the sequence with elements $w^{(n)}b$ also converges to w and it is not hard to find examples where

$$\lim_{n\to\infty} y_{w^{(n)}} \ne \lim_{n\to\infty} y_{w^{(n)}b}.$$

Therefore we shall only define fixed points for a particular class of infinite words, as follows. Let $0s$ be the Sturmian $\mathcal{M}$-word of rate α. Consider the sequence of Christoffel words $0w^{(n)}1$ that lie on the following path through the Christoffel tree. We start from the root,

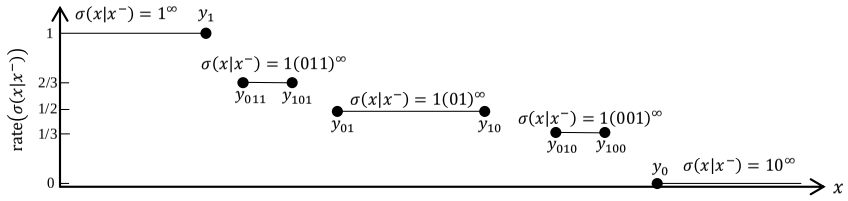


Figure 5: The itinerary $\sigma(x|x^-)$ of Theorem 12. The filled circles in this plot emphasise that the corresponding intervals are closed.

so that $w^{(1)} = \epsilon$. Then for $n \in \mathbb{Z}_{++}$, we set $0w^{(n+1)}1$ equal to the left child of $0w^{(n)}1$ if the rate of $0w^{(n)}1$ exceeds α and equal to the right child otherwise. We call

$$y_s := \lim_{n\to\infty} y_{01w^{(n)}} = \lim_{n\to\infty} y_{10w^{(n)}}$$

the *fixed point* of the Sturmian $\mathcal{M}$-word $0s$. The fact that these limits exist and are equal is proved as Lemma 55 in Appendix A.3.

We are now ready to fully characterise the itineraries of maps-with-gaps, or equivalently to describe the action sequences resulting from applying threshold policies, under Assumption A2. We do so in three cases. Theorem 12 considers the itinerary $\sigma(x|x^-)$ of an active-at-threshold map with initial state equal to the threshold. Corollary 13 considers the pair of itineraries for a passive-at-threshold map with initial states $\phi_0(x)$ or $\phi_1(x)$ for threshold x. Finally, Theorem 14 characterises the itinerary $\sigma(x|s)$ in the general case where the initial state x is unrelated to the threshold s.

Let us begin with an intuitive description of our characterisation of the itineraries $\sigma(x|x^-)$ of active-at-threshold maps with initial state x equal to the threshold. This description is illustrated by Figure 5. Firstly, all such itineraries begin with the letter or action 1, since the initial state equals the threshold. For low values of x, at or below the fixed point y_1 of the map $\phi_1(\cdot)$, taking action 1 leaves the state at or above the threshold, so the itinerary repeats 1 for ever. Similarly, for large values of x, at or above the fixed point y_0 of the map $\phi_0(\cdot)$, after taking action 1, the orbit never goes as high as its initial state again, so the itinerary repeats 0 for ever. Meanwhile, at intermediate values of x, there is an interval $[y_{01}, y_{10}]$ for which the itinerary repeats the Christoffel word 01 for ever. Given that $y_{10} > y_{01}$, such a ping-pong behaviour is very intuitive for x in the set $\{y_{01}, y_{10}\}$, since $\phi_0(y_{01}) = y_{10}$ and $\phi_1(y_{10}) = y_{01}$ by definition of the fixed points.

For other values of x, the itinerary is more complex. In fact, for any Christoffel word, there is an interval of positive length for which the itinerary simply repeats that word over and over again. See for instance the interval $[y_{011}, y_{101}]$ in Figure 5 on which the itinerary repeats the Christoffel word 011, or the interval $[y_{010}, y_{100}]$ on which it repeats the word 001. Now, between these intervals of positive length, there are individual *points* at which the itinerary equals any given Sturmian $\mathcal{M}$-word. Such points are not shown in Figure 5 but lie somewhere in the gaps between the intervals that are shown. Furthermore, as x

increases through the interval $\mathcal{I}$, the itinerary goes through all the $\mathcal{M}$-words in order of decreasing rate, and it never takes a value that is not given by an $\mathcal{M}$-word.

In the following theorem, let $\{0,1\}^\infty$ denote the set of all infinite binary strings and let $\{0,1\}^+$ denote the set of all finite non-empty binary strings. Also, the *image* of a function is the subset of the function's range consisting of those values that the function takes for some point in the function's domain. Although in general an itinerary $\sigma(x|x^-)$ is in $\{0,1\}^\infty$, the following theorem shows that, under Assumption A2, the itinerary comes from a very specific subset of such binary strings that is generated by the set of $\mathcal{M}$-words.

Theorem 12 *Suppose A2 holds, $0p1$ is a Christoffel word and $0s$ is a Sturmian $\mathcal{M}$-word. Then the fixed points y_{01p}, y_{10p}, y_s exist in $\mathcal{I}$. Also, the itinerary $\sigma(x|x^-)$ is a lexicographically non-increasing function of $x \in \mathcal{I}$ and is of the form $\sigma(x|x^-) = 1\ell(x)^\infty$ for some mapping $\ell : \mathcal{I} \to \{0,1\}^\infty \cup \{0,1\}^+$ whose image is the set of $\mathcal{M}$-words. Specifically,*

$$\sigma(x|x^-) = \begin{cases} 1^\infty & \text{if and only if } x \le y_1 \\ (10p)^\infty & \text{if and only if } x \in [y_{01p}, y_{10p}] \\ 10s & \text{if and only if } x = y_s \\ 10^\infty & \text{if and only if } x \ge y_0. \end{cases}$$

See Appendix A.4 for a proof. This result is previously known for *linear* maps-with-gaps (Rajpathak et al., 2012), although those authors do not draw any relation to mechanical words. Dance and Silander (2015) previously extended those authors' proof to the nonlinear case under Assumption A2. The proof presented in Appendix A of this paper can be seen as a simplification of that extension. On the other hand, it is known that itineraries of a broader class of nonlinear maps-with-gaps that do not necessarily satisfy Assumption A2 also correspond to mechanical words (Kozyakin, 2003). However such generality comes at a cost, as it is not clear in that work which range of thresholds gives rise to which words.

Remark. Not all the maps-with-gaps considered in this paper satisfy Assumption A2. However, this does not always prevent the application of Theorem 12. Notably for $\mathcal{I} := [0, \infty)$ and $\theta \in (0, \infty)$, the pair

$$\phi_0(x) = x + 1, \qquad \phi_1(x) = 1/(\theta + 1/(x + 1))$$

involves the non-contractive map ϕ_0. Nevertheless, after the change of coordinates

$$g : x \mapsto x/(x + 1),$$

the transformed functions

$$\tilde{\phi}_0(x) := g(\phi_0(g^{(-1)}(x))) = 1/(2 - x), \qquad \tilde{\phi}_1(x) := g(\phi_1(g^{(-1)}(x))) = 1/(2 + \theta - x)$$

and the interval $\tilde{\mathcal{I}} := [0, 1]$ do satisfy Assumption A2. Indeed

$$\frac{d\tilde{\phi}_1(x)}{dx} = 1/(2 + \theta - x)^2 \in (0, 1]$$

for $x \in \tilde{\mathcal{I}}$ and $\theta \in [0, \infty)$, and this derivative only equals 1 for the endpoint $x = 1$. Thus $\tilde{\phi}_1$ is increasing and contractive on $\tilde{\mathcal{I}}$. Noting that $\tilde{\phi}_0(x) = \lim_{\theta \to 0} \tilde{\phi}_1(x)$, the same holds for $\tilde{\phi}_0$. Also $\tilde{\phi}_1$ has a fixed point at $y(\theta) = (2 + \theta - \sqrt{\theta^2 + 4\theta})/2$ which lies in $\tilde{\mathcal{I}}$ for $\theta \in [0, \infty)$, and $\tilde{\phi}_0$ has a fixed point at $y(0) = 1 > y(\theta)$. As g is an increasing function, all conclusions of Theorem 12 still hold for the original functions ϕ_0, ϕ_1.

Pairs of Itineraries. As we are particularly interested in expressions like the Whittle index in Theorem 1, it is important to have a result characterising itineraries starting from $\phi_0(x)$ and $\phi_1(x)$, as in the following Corollary of Theorem 12, whose proof is given as Appendix A.5.

Corollary 13 *Suppose A2 holds,* $0p1$ *is a Christoffel word and* $0s$ *is a Sturmian* $\mathcal{M}$*-word. Then the pair of itineraries* $(\sigma(\phi_0(x)|x), \sigma(\phi_1(x)|x))$ *is given by*

$$\begin{cases} (1^\infty, 1^\infty) & \text{if } x < y_1 \\ (1^\infty, 01^\infty) & \text{if } x = y_1 \\ ((1p0)^\infty, (0p1)^\infty) & \text{if } x \in [y_{01p}, y_{10p}) \\ ((1p0)^\infty, 0p(01p)^\infty) & \text{if } x = y_{10p} \\ (1s, 0s) & \text{if } x = y_s \\ (0^\infty, 0^\infty) & \text{if } x \geq y_0. \end{cases}$$

Looking ahead, applying this corollary to the problem addressed in Theorem 1 shows that the interval $\mathcal{I}$ can be partitioned into intervals corresponding to Christoffel words on which the marginal resource $g(x, x)$ of Section 3 is a constant function of x and points corresponding to Sturmian $\mathcal{M}$-words. In particular, this partition of the interval $\mathcal{I}$ is central to our analysis of the the marginal productivity index $m^*(x)$ in Section 3.3.

Discontinuities of the Itinerary. We conclude this section by considering the number of discontinuities of the mapping from the threshold to the length-n prefix of the itinerary, for any given positive integer n. We show that this number of discontinuities can be upper-bounded by a polynomial function of n. Now for any finite word w and state $x \in \mathcal{I}$, the map $s \mapsto \mathbf{1}_{s < \phi_w(x)}$ is right continuous. Also, if Assumption A2 holds, then the map ϕ_w is continuous (Lemma 32), and it then follows from Definition 5 that the itinerary satisfies $\sigma(x|s)_{1:n} = \sigma(x|s^+)_{1:n}$ for any $n \in \mathbb{Z}_+$, and that the limit $\sigma(x|s^-)_{1:n}$ exists. Thus, in the following theorem, a *discontinuity* of the mapping $s \mapsto \sigma(x|s)_{1:n}$ is a point $d \in \mathcal{I}$ at which its left and right limits disagree, so that $\sigma(x|d^-)_{1:n} \neq \sigma(x|d^+)_{1:n}$. Also, in the following theorem, we say that a finite word w is a *factor of a lower mechanical word* if there exists a slope $\alpha \in [0, 1]$ and an intercept $\rho \in \mathbb{R}$ such that $w_n = \lfloor \alpha(n+1) + \rho \rfloor - \lfloor \alpha n + \rho \rfloor$ for $n = 1, 2, \ldots, |w|$.

Theorem 14 *Suppose* ϕ_0, ϕ_1 *satisfy A2, that* $n \in \mathbb{Z}_{++}$ *and* $x \in \mathcal{I}$*. Then* $\sigma(x|s)$ *is a lexicographically non-increasing function of* $s \in \mathcal{I}$*. Also, for any fixed* $x, s \in \mathcal{I}$*, we have*

$$\sigma(x|s)_{1:n} = l^m w$$

for some $l \in \{0, 1\}$*, some* $m \in \{0, 1, \ldots, n\}$*, and some factor* w *of a lower mechanical word. Furthermore, for any* $x \in \mathcal{I}$*, the mapping* $s \mapsto \sigma(x|s)_{1:n}$ *for* $s \in \mathcal{I}$ *has at most a polynomial number* $p(n)$ *of discontinuities.*

This result is an immediate consequence of work by Kozyakin (2003) and Mignosi (1991). Its proof is given as Appendix A.6. Looking ahead, this result is important for changing the order of certain summations in the proof of Proposition 30, which shows that the Radon-Nikodym condition (PCLI3) of Section 3 holds.

3. Proof of Main Result

First we present a theorem due to Niño-Mora (Niño-Mora, 2015, 2019) which guarantees the indexability and optimality of threshold policies for projects satisfying three so-called partial conservation law indexability (PCLI) conditions (Section 3.1). The following three subsections show that these three conditions in turn are satisfied for projects satisfying Assumption A1, by making intensive use of our characterisation of the itineraries of maps-with-a-gap from Section 2. Then we couple the results together to prove our main result, Theorem 1, concerning threshold policies for observing time-series and the associated Whittle indices (Section 3.5). Finally, we prove Corollary 2 concerning the structure of optimal policies for the linear-quadratic Gaussian problem with costly observations (Section 3.6).

3.1. Verification Theorem

We present a theorem which guarantees the optimality of threshold policies for two-action Markov decision problems called *projects* under certain hypotheses. Under the same hypotheses, the theorem also guarantees the indexability of such projects. This theorem is due to Niño-Mora (2015) and extends previous work on countable state spaces (Niño-Mora, 2001, 2002, 2006) to problems where the state space is an interval of the real line. Niño-Mora calls the theorem's hypotheses the *partial conservation law indexability (PCLI)* conditions. This terminology was chosen to contrast with the *strong* conservation law conditions of Shanthikumar and Yao (1992) and the *generalised* conservation law conditions of Bertsimas and Niño-Mora (1996), which have their roots in *conservation laws* for queueing systems under which waiting times are invariant to the queueing discipline (Kleinrock, 1965). Although, at the time of writing, Niño-Mora (2015) is not published in a peer-reviewed publication, we have developed our own proofs of a version of the verification theorem and associated results for the special case addressed in this paper in Dance and Silander (2017).

Projects and λ-Price Problems. A *project* $\mathcal{P} = \langle \mathcal{X}, c, r, \mathscr{Q}, \beta \rangle$ describes a stochastic process with state space $\mathcal{X}$, resource function $c : \mathcal{X} \times \{0, 1\} \to \mathbb{R}$, reward function $r : \mathcal{X} \times \{0, 1\} \to \mathbb{R}$, transition law $\mathscr{Q}$ and discount factor $\beta \in [0, 1)$, with the same interpretation as on page 10. Let Π denote the class of all history-dependent randomized policies. Let $\mathbb{E}_x^\pi$ denote the expectation over sequences $(X_t, A_t)_{t=0}^\infty$ with initial state $X_0 = x$, where actions are taken according to policy π and transitions are according to transition law $\mathscr{Q}$. Then the *λ-price problem* for project $\mathcal{P}$ and price $\lambda \in \mathbb{R}$ is to is to find a policy $\pi \in \Pi$ that minimises

$$V_\lambda(x, \pi) := \mathbb{E}_x^\pi \left[\sum_{t=0}^{\infty} \beta^t \left(r(X_t, A_t) - \lambda c(X_t, A_t) \right) \right]$$

for all initial states $x \in \mathcal{X}$. Such a policy is said to be *optimal* for the λ-price problem. We say action $a \in \{0,1\}$ is *optimal in state* x if there is a policy $\pi^* \in \Pi$ such that $V(x, \langle a, \pi^* \rangle) = \sup_{\pi \in \Pi} V(x, \pi)$, where $\langle a, \pi \rangle$ denotes the policy which first takes action a then follows policy π. Niño-Mora (2015) makes the following assumption about projects.

Assumption A3

(i) The state space is $\mathcal{X} = \{x \in \mathbb{R} : l \leq x \leq u\}$ *for some endpoints* $-\infty \leq l < u \leq \infty$.

(ii) For each action $a \in \{0,1\}$, *the reward function* $r(\cdot, a)$ *and resource function* $c(\cdot, a)$ *are continuous on* $\mathcal{X}$, *and* $c(\cdot, a)$ *satisfies* $0 \leq c(x,0) < c(x,1)$ *for all* $x \in \mathcal{X}$.

(iii) There exists a measurable weight function $w : \mathcal{X} \to [1, \infty)$ *and constants* $M > 0$ *and* $\gamma \in [\beta, 1)$, *such that for every state* $x \in \mathcal{X}$ *and action* $a \in \{0,1\}$,

(a) $\max\{|C(x,a)|, |c(x,a)|\} \leq M w(x)$; *and*

(b) $\beta \int_{\mathcal{X}} w(y) \mathcal{Q}(dy|x,a) \leq \gamma w(x)$.

In part (i) of this assumption, the state space $\mathcal{X}$ is of one of the forms $(-\infty, \infty)$, $(-\infty, u']$, $[l', \infty)$ or $[l', u']$ for some $l', u' \in \mathbb{R}$. Although this does not include the state space $\mathcal{I} = (0, \infty)$ in Assumption A1, we show that the result still holds for $\mathcal{X} = (0, \infty)$ in the proof of Theorem 1. The assumption allows c and r to be unbounded functions of the state, however part (iii) ensures that the weighted supremum approach to discounted problems (Wessels, 1977; Hernández-Lerma and Lasserre, 1999) can be applied. This assumption guarantees that discounted sums involving c or r are well-defined for all policies. Furthermore, although Bellman's equations may have multiple unbounded solutions, this assumption identifies the solution corresponding to the optimal value function $V^*(\cdot)$ as the unique solution for which the ratio $|V^*(x)/w(x)|$ is bounded on $\mathcal{X}$.

Optimality of Threshold Policies. For a threshold $s \in \bar{\mathbb{R}}$ and assuming a state-space consisting of real numbers, let π_s denote the *s-threshold policy* that takes actions $A_t = \mathbf{1}_{X_t > s}$ and let π_{s^-} denote the *s^--threshold policy* that takes actions $A_t = \mathbf{1}_{X_t \geq s}$. As on page 11, given a policy π for a project $\mathcal{P}$ in initial state x, we define the reward-to-go or *reward metric* $F(\cdot,\cdot)$ and the resource-to-go or *resource metric* $G(\cdot,\cdot)$ by

$$F(x,\pi) := \mathbb{E}_x^\pi \left[\sum_{t=0}^{\infty} \beta^t r(X_t, A_t) \right], \qquad G(x,\pi) := \mathbb{E}_x^\pi \left[\sum_{t=0}^{\infty} \beta^t c(X_t, A_t) \right].$$

Motivated by expression (8) for the Whittle index, we define the *marginal reward* $f : \mathcal{X} \times \bar{\mathbb{R}} \to \mathbb{R}$ and the *marginal resource* $g : \mathcal{X} \times \bar{\mathbb{R}} \to \mathbb{R}$ by

$$f(x,s) := F(x, \langle 1, \pi_s \rangle) - F(x, \langle 0, \pi_s \rangle), \qquad g(x,s) := G(x, \langle 1, \pi_s \rangle) - G(x, \langle 0, \pi_s \rangle).$$

Finally, we define the *marginal productivity index* $m^*(\cdot)$ to be the ratio of marginal reward to marginal resource when the threshold equals the state, so that

$$m^*(x) := \frac{f(x,x)}{g(x,x)}.$$

Note that the marginal productivity index has the same form as the right-hand side of (8) under the assumption that the x-threshold policy is optimal in the λ-price problem. Niño-Mora (2015) makes the following assumption about the behaviour of the above metrics.

Assumption A4 (Partial Conservation Law Indexability Conditions)

PCLI1. *The marginal resource* $g(x,s)$ *is positive for every state* $x \in \mathcal{X}$ *and threshold* $s \in \bar{\mathbb{R}}$.

PCLI2. *The marginal productivity index* $m^*(\cdot)$ *is non-decreasing, continuous and bounded from below on* $\mathcal{X}$.

PCLI3. *The marginal productivity index* $m^*(\cdot)$ *and metrics* $F(x,\cdot)$ *and* $G(x,\cdot)$ *satisfy*

$$F(x, \pi_{s_2}) - F(x, \pi_{s_1}) = \int_{(s_1, s_2]} m^*(s) G(x, \pi_{ds})$$

for each pair of thresholds $-\infty < s_1 < s_2 < \infty$ *and for each state* $x \in \mathcal{X}$.

In this assumption, PCLI1 ensures that the marginal productivity index is well defined in the sense that its denominator does not vanish. To clarify the notation in PCLI3, for fixed $x \in \mathcal{X}$, let $\tilde{F}(s) := F(x, \pi_s)$ and $\tilde{G}(s) := G(x, \pi_s)$ for $s \in \bar{\mathbb{R}}$. Then PCLI3 requires that $\tilde{F}(s_2) - \tilde{F}(s_1) = \int_{(s_1, s_2]} m^*(s) \tilde{G}(ds)$ where the right-hand side is a Lebesgue-Stieltjes integral (Carter and van Brunt, 2000). Thus, PCLI3 requires that marginal productivity index is a Radon-Nikodym derivative of the reward metric for an s-threshold policy with respect to the resource metric for an s-threshold policy, so that $m^* = \frac{d\tilde{F}}{d\tilde{G}}$. In analyses of discrete-state projects (Niño-Mora, 2001, 2002, 2006), PCLI3 is not required as it is implied by PCLI1 and PCLI2. Niño-Mora (2015, Appendix B) gives sufficient conditions for PCLI3 to hold, but it is not clear that those are satisfied for the problem addressed in this paper.

We now state the verification theorem, which is central to our proof of Theorem 1.

Theorem 15 (Niño-Mora, 2015, Theorem 2.1, p. 13) *Suppose project* $\mathcal{P}$ *satisfies A3 and A4. Then project* $\mathcal{P}$ *is indexable with Whittle index equal to its marginal productivity index* $m^*(\cdot)$. *Furthermore, for every* $\lambda \in \mathbb{R}$, *there exists an* $s \in \bar{\mathbb{R}}$ *such that both the s-threshold and* s^-*-threshold policies are optimal for the* λ*-price problem for project* $\mathcal{P}$.

Remark. For the project in Theorem 1 we have $c(x,a) = a$ and $r(x,a) = -C(x)$. Let us give some expressions for the metrics F, G, f and g for that project that we shall use in the rest of this section. Firstly, let $(X_t(x,a;s))_{t=0}^{\infty}$ be the sequence of states starting from $X_0(x,a;s) = x$ then taking action $a \in \{0,1\}$ to give $X_1(x,a;s) = \phi_a(x)$ then acting according to the s-threshold policy, so that the action sequence is $A_0(x,a;s) = a$ and $A_t(x,a;s) = \mathbf{1}_{X_t(x,a;s) > s}$ for $t > 0$. Let $X_t(x,a;s^-)$ and $A_t(x,a;s^-)$ be the state and actions from initial state x and initial action a when the s^--threshold policy is followed thereafter, so that $A_t(x,a;s) = \mathbf{1}_{X_t(x,a;s) \geq s}$ for $t > 0$. Also, let $X_t(x;s) := X_t(x, \mathbf{1}_{x>s}; s)$, $A_t(x;s) = A_t(x, \mathbf{1}_{x>s}; s)$, $A_t(x,a;s^-) = A_t(x, \mathbf{1}_{x \geq s}; s^-)$ for $t \in \mathbb{Z}_+$. Then

$$f(x,s) = \sum_{t=0}^{\infty} \beta^t \left(C(X_t(x,0;s)) - C(X_t(x,1;s)) \right), \; g(x,s) = \sum_{t=0}^{\infty} \beta^t \left(A_t(x,1;s) - A_t(x,0;s) \right).$$

Given the notation for itineraries (Definition 5) and for compositions of maps ϕ_0, ϕ_1 we can write more-explicit expressions for the reward and cost metric for that project under a threshold policy, for instance

$$F(x, \langle a, \pi_s \rangle) = \sum_{n=1}^{\infty} \beta^{n-1} C(\phi_{(a\sigma(\phi_0(x)|s))_{1:n}}(x)).$$

Furthermore, in the light of Corollary 13, we can find expressions for the marginal reward and marginal resource in terms of $\mathcal{M}$-words. For instance if $0p1$ is a Christoffel word of length N and $x \in [y_{01p}, y_{10p})$ then

$$g(x,x) = \sum_{n=1}^{\infty} \beta^{n-1} \left[((10p)^\infty)_n - ((01p)^\infty)_n \right] = \sum_{k=0}^{\infty} \beta^{kN}(1-\beta) = \frac{1-\beta}{1-\beta^N} \tag{12}$$

since $((10p)^\infty)_n$ and $((01p)^\infty)_n$ only differ when n is 1 or 2 modulo N, whereas for $x = y_{10p}$ we have

$$g(x,x) = \sum_{n=1}^{\infty} \beta^{n-1} \left[(10p(01p)^\infty)_n - ((01p)^\infty)_n \right] = 1 - \beta.$$

In spite of this discontinuity in $g(x,x)$, Lemma 19 shows that the marginal productivity index does not have a discontinuity at points such as y_{10p}. Similarly, if $0s$ is a Sturmian $\mathcal{M}$-word and $x = y_s$, then the same argument gives $g(x,x) = 1 - \beta$.

3.2. Positivity of Marginal Resource Metric (PCLI1)

We prove that condition PCLI1 holds. The argument is based on the following notion of *swapping*, which is partly inspired by results about the Burrows-Wheeler transform of Christoffel words (Berstel et al., 2008, Chapter 6).

Definition 16 *A finite word a **swaps to** a finite word b if either $a = b$ or there exist words $p_1, q_1, p_2, q_2, \ldots, p_n, q_n$ for some $n \in \mathbb{Z}_{++}$ with*

$$a = p_1 10 q_1, \qquad p_1 01 q_1 = p_2 10 q_2, \qquad \ldots, \qquad p_n 01 q_n = b.$$

*We call a transformation $p_k 10 q_k \to p_k 01 q_k$ an **exchange**.*

Example. The word 1100 swaps to the word 0101 via the exchanges

$$1100 \to 1010 \to 0110 \to 0101$$

for which $p_1 = 1$, $q_1 = 0$, $p_2 = \epsilon$, $q_2 = 10$, $p_3 = 01$ and $q_3 = \epsilon$.

The idea of our proof is as follows. First we find conditions on the number of 1's in prefixes of two words that make it possible to swap one word for another (Lemma 17). Proposition 18 shows that those conditions are satisfied by the dynamical system (statement 1), and they imply positivity of the marginal resource metric (statement 2).

Lemma 17 *Suppose a, b are words of common length $|a| = |b| = n \in \mathbb{Z}_+$ with*

$$|a|_1 = |b|_1 \quad \textit{and} \quad |a_{1:k}|_1 \geq |b_{1:k}|_1 \quad \textit{for } k < n.$$

Then a swaps to b.

Proof Given any words u, v of length n, consider the distance

$$d(u,v) := \sum_{i=1}^{n} \left| |u_{1:i}|_1 - |v_{1:i}|_1 \right|.$$

If $a = b$ then a swaps to b after $d(a,b) = 0$ exchanges. Otherwise $a \neq b$ and we shall show that there exists a word a' such that

$$a \text{ and } a' \text{ differ by a single exchange,} \tag{13}$$
$$d(a', b) = d(a, b) - 1, \tag{14}$$
$$\text{and } a' \text{ and } b \text{ satisfy the hypotheses of this Lemma.} \tag{15}$$

Repeating this argument shows that a swaps to b after $d(a,b)$ exchanges.

We now define an appropriate word a'. As $a \neq b$ and $|a_{1:i}|_1 \geq |b_{1:i}|_1$ for $i = 1, 2, \ldots, n-1$, there must exist a first index i such that $|a_{1:i}|_1 > |b_{1:i}|_1$. Also, as $|a|_1 = |b|_1$, there must exist a first index $j > i$ such that $a_j = 0$. As i, j are the first such indices, it follows that $a_k = 1$ for $i \leq k < j$. Thus

$$a = a_{1:(j-2)} 10 a_{(j+1):n}$$

with the convention that $a_{1:0} = a_{(n+1):n} = \epsilon$. Now consider the word

$$a' := a_{1:(j-2)} 01 a_{(j+1):n}.$$

It is immediate that (13) holds. Furthermore, as $a_k = 1$ for $i \leq k < j$, we have

$$\left|a_{1:(j-1)}\right|_1 - \left|b_{1:(j-1)}\right|_1 \geq |a_{1:i}|_1 - |b_{1:i}|_1 > 0$$

where the second inequality follows from the definition of i. Thus the definition of a' gives

$$\left|a'_{1:l}\right|_1 = |a_{1:l}|_1 - \mathbf{1}_{l=j-1} \geq |b_{1:l}|_1 \quad \text{for } l = 1, 2, \ldots, n, \tag{16}$$

so that

$$d(a', b) = \sum_{i=1}^{n} \left(\left|a'_{1:i}\right|_1 - |b_{1:i}|_1 \right) = d(a,b) - 1.$$

Therefore (14) holds.

Finally, combining (16) with the fact that $|a'|_1 = |a|_1 = |b|_1$, we conclude that (15) holds. This completes the proof. ■

The next proposition shows that PCLI1 holds if $\langle \mathcal{I}, C, \phi_0, \phi_1, \beta \rangle$ satisfy Assumption A1. To see that the hypotheses of this proposition follow from A1, note that A1 gives

$$\frac{d\phi_a(x)}{dx} = \frac{r^2}{(\theta_a(r^2x+1)+1)^2} > 0$$

for $a \in \{0,1\}$. Also, putting $u \coloneqq r^2x+1$, the inequality

$$\phi_{10}(x) = \frac{u\theta_1 \quad + \quad (r^2u+1)}{(r^2u+1)\theta_0 + u\theta_1 + (u\theta_0\theta_1+1)} > \frac{u\theta_0 \quad + \quad (r^2u+1)}{(r^2u+1)\theta_1 + u\theta_0 + (u\theta_0\theta_1+1)} = \phi_{01}(x)$$

follows directly from the mediant inequality, as $\theta_1 > \theta_0$.

Proposition 18 *Suppose $\mathcal{I}$ is an interval of $\mathbb{R}$ and that $\phi_0 : \mathcal{I} \to \mathcal{I}, \phi_1 : \mathcal{I} \to \mathcal{I}$ satisfy*

(i) $\phi_0(\cdot), \phi_1(\cdot)$ are increasing functions

(ii) $\phi_{01}(z) < \phi_{10}(z)$ for all $z \in \mathcal{I}$.

Also suppose that $x \in \mathcal{I}$, $s \in \overline{\mathbb{R}}$ and consider the itineraries

$$a := 1\sigma(\phi_1(x)|s) \quad \textit{and} \quad b := 0\sigma(\phi_0(x)|s).$$

Then the following statements hold:

1. *For any $n \in \mathbb{Z}_{++}$, we have $|a_{1:n}|_1 \geq |b_{1:n}|_1$.*
2. *For any $\beta \in (0,1)$, the marginal resource metric $g(x,s)$ is positive.*

Proof We prove statement 1 by induction. In the base case $|a_1|_1 = 1 \geq 0 = |b_1|_1$. For the inductive step, suppose $|a_{1:k}|_1 \geq |b_{1:k}|_1$ for all $k \leq m$ for some $m \in \mathbb{Z}_{++}$. This induction hypothesis shows that either $|a_{1:m}|_1 > |b_{1:m}|_1$ or $|a_{1:m}|_1 = |b_{1:m}|_1$. In the first case, $\left|a_{1:(m+1)}\right|_1 \geq \left|b_{1:(m+1)}\right|_1$ as we are only adding one letter to $a_{1:m}$ and $b_{1:m}$. In the second case, the induction hypothesis shows that the words $a_{1:m}$ and $b_{1:m}$ satisfy the assumptions of Lemma 17, so there is a sequence of swaps that transforms $a_{1:m}$ into $b_{1:m}$. Consider any swap $p10q$ to $p01q$ in this sequence. Then hypothesis (ii) gives $\phi_{10}(\phi_p(x)) > \phi_{01}(\phi_p(x))$ and hypothesis (i) implies that $\phi_q(\cdot)$ is increasing, so

$$\phi_{p10q}(x) > \phi_{p01q}(x).$$

Repeating this argument over the sequence of swaps gives

$$\phi_{a_{1:m}}(x) > \phi_{b_{1:m}}(x).$$

Thus, it follows from the definition of itineraries that the last letters of $a_{1:(m+1)}, b_{1:(m+1)}$ have $a_{m+1}b_{m+1} \in \{00, 10, 11\}$. Hence $\left|a_{1:(m+1)}\right|_1 \geq \left|b_{1:(m+1)}\right|_1$. This proves statement 1.

To prove statement 2, note that the definition of $g(x,s)$ gives

$$g(x,s) = \sum_{k=1}^{\infty} \beta^{k-1}(a_k - b_k)$$

$$\begin{aligned}
&= \sum_{k=1}^{\infty} \beta^{k-1}(|a_{1:k}|_1 - \left|a_{1:(k-1)}\right|_1 - |b_{1:k}|_1 + \left|b_{1:(k-1)}\right|_1) \\
&= (1-\beta)\sum_{k=1}^{\infty} \beta^{k-1}(|a_{1:k}|_1 - |b_{1:k}|_1) \\
&\geq 1-\beta
\end{aligned}$$

where the second line follows as $a_k = |a_{1:k}|_1 - \left|a_{1:(k-1)}\right|_1$ and similarly for b_k, the third line holds because $|a_{1:0}|_1 = |b_{1:0}|_1 = 0$, and the last line follows from statement 1 and the fact that $a_1 = 1, b_1 = 0$. As $\beta < 1$, this completes the proof. ■

3.3. Behaviour of the Marginal Productivity Index (PCLI2)

Condition PCLI2 requires that the marginal productivity index $m^*(x) = f(x,x)/g(x,x)$ is continuous, non-decreasing and bounded from below for $x \in \mathcal{I}$. We begin by showing that this index is *Lipschitz* continuous on any finite interval of $\mathcal{I}$ in Proposition 21. To show the index is non-decreasing, we couple Lipschitz continuity with a piecewise analysis of the index, driven by Theorem 12, which shows that the interval $\mathcal{I}$ can be divided up into intervals corresponding to Christoffel words and points corresponding to Sturmian $\mathcal{M}$-words. The central result of this analysis is Lemma 25, which shows that the marginal reward with threshold equal to initial state $x \mapsto f(x,x)$ is non-decreasing for x in the interval corresponding to any given Christoffel word of the form $0p1$. Since the marginal resource metric $g(x,x)$ equals a positive constant on such intervals, given by (12), this implies that $m^*(x)$ is non-decreasing on such intervals. Since $m^*(\cdot)$ is also Lipschitz continuous on any finite interval, Proposition 27 concludes that $m^*(\cdot)$ is non-decreasing on $\mathcal{I}$. We note that Proposition 27 would not follow if we were only to have shown that $m^*(\cdot)$ is continuous (rather than Lipschitz continuous). Indeed, there are functions that are continuous and have a positive derivative almost everywhere, but are not non-decreasing. One example is the function $x \mapsto x - \text{Cantor}(x)$ where $\text{Cantor}(\cdot)$ is the Cantor-Lebesgue function (Royden and Fitzpatrick, 2010, page 50). Finally, Proposition 28 shows that the index is bounded below, so PCLI2 is satisfied.

A related proof was given by Dance and Silander (2015). However, that proof only covers systems for which the multiplier r in Assumption A1 is $r = 1$, rather than multipliers $r \in (0,1]$ as addressed here. For $r = 0$, we have $\phi_0(x) = \phi_1(x) = 1$ for all $x \in \mathcal{I}$, so the sum in Lemma 25 vanishes, and the analysis presented here is unnecessary. Also, the proof of Dance and Silander (2015) only addressed the cost function $C(x) = x$, whereas here we generalise to any cost function satisfying Assumption A1, which includes any cost function of the form x^q/q for $q \in [-1, \infty)$. A counterexample in Figure 7, shows that the marginal reward is not necessarily increasing on intervals corresponding to Christoffel words for $C(x) = x^q/q$ with $q < -1$.

3.3.1. The Marginal Productivity Index is Continuous

We demonstrate Proposition 21 which shows that the marginal productivity index m^* for systems satisfying Assumption A1 is Lipschitz continuous on any finite interval. We begin by showing that m^* is continuous.

Lemma 19 *If A1 holds then the marginal productivity index $m^*(\cdot)$ is continuous on $\mathcal{I}$.*

Proof Consider the itineraries $h_a(x) := a\sigma(\phi_a(x)|x)$ for $a \in \{0,1\}$ and $x \in \mathcal{I}$. In terms of these itineraries the marginal productivity index is

$$m^*(x) = \frac{\sum_{n=1}^{\infty} \beta^{n-1} \left(C(\phi_{[h_0(x)]_{1:n}}(x)) - C(\phi_{[h_1(x)]_{1:n}}(x))\right)}{\sum_{n=1}^{\infty} \beta^{n-1} \left([h_1(x)]_n - [h_0(x)]_n\right)}.$$

Since A1 holds, this is a continuous function of x at any point where $h_a(x^-) = h_a(x) = h_a(x^+)$. By following paths in the Christoffel tree to find the limiting word just lexicographically higher or lower than a given Christoffel word, and using Corollary 13, we obtain the following limits:

- $h_0(x^-) = 01p(10p)^\infty$ if $0p1$ is a Christoffel word and $x = y_{01p}$; $h_0(x^-) = 010^\infty$ if $x = y_0$; and $h_0(x^-) = h_0(x)$ otherwise.
- $h_1(x^-) = 1^\infty$ if $x = y_1$; $h_1(x^-) = (10p)^\infty$ if $0p1$ is a Christoffel word and $x = y_{10p}$; and $h_1(x^-) = h_1(x)$ otherwise.
- $h_a(x^+) = h_a(x)$ for all $x \in \mathcal{I}$ and all $a \in \{0,1\}$.

In view of these limits, it remains to show that $m^*(x^-) = m^*(x)$ when: (i) $x = y_{01p}$ for some Christoffel word $0p1$; (ii) $x = y_{10p}$ for some Christoffel word $0p1$; (iii) $x = y_0$; and (iv) $x = y_1$. We only consider (i) as the proofs for (ii)-(iv) are similar.

Say $x = y_{01p}$ for a Christoffel word $0p1$. The itineraries above and Corollary 13 give

$$h_0(y_{01p}^-) = 01p(10p)^\infty, \quad h_0(y_{01p}) = (01p)^\infty, \quad h_1(y_{01p}^-) = (10p)^\infty, \quad h_1(y_{01p}) = (10p)^\infty.$$

So the marginal resource satisfies

$$\lim_{x \uparrow y_{01p}} g(x,x) = \sum_{n=1}^{\infty} \beta^{n-1} \left([(10p)^\infty]_n - [01p(10p)^\infty]_n\right) = 1-\beta, \tag{17}$$

$$g(y_{01p}, y_{01p}) = \sum_{n=1}^{\infty} \beta^{n-1} \left([(10p)^\infty]_n - [(01p)^\infty]_n\right) = \frac{1-\beta}{1-\beta^{|0p1|}}. \tag{18}$$

Consider the sum $S(w,x) := \sum_{n=1}^{|w|} \beta^{n-1} C(\phi_{w_{1:n}}(x))$ for any (possibly infinite) word w. Then the marginal reward $f(\cdot,\cdot)$ satisfies

$$\begin{aligned}
\lim_{x \uparrow y_{01p}} f(x,x) &= S(01p(10p)^\infty, y_{01p}) - S((10p)^\infty, y_{01p}) && (19)\\
&= S(01p, y_{01p}) + \beta^{|01p|} S((10p)^\infty, \phi_{01p}(y_{01p})) - S((10p)^\infty, y_{01p})\\
&= S(01p, y_{01p}) - (1-\beta^{|0p1|}) S((10p)^\infty, y_{01p}) && (20)
\end{aligned}$$

$$\begin{aligned}
&= (1-\beta^{|0p1|}) S((01p)^\infty, y_{01p}) - (1-\beta^{|0p1|}) S((10p)^\infty, y_{01p}) && (21)\\
&= (1-\beta^{|0p1|}) f(y_{01p}, y_{01p}). && (22)
\end{aligned}$$

In more detail, (19) follows from the fact that for any word w, we have $S(w,x^-) = S(w,x)$ for any $x \in \mathcal{I}$ as A1 holds; (20) holds as $\phi_{01p}(y_{01p}) = y_{01p}$; (21) follows from the expression for the sum of a geometric progression.

Combining (17), (18) and (22), shows the marginal productivity index $m^*(\cdot)$ satisfies

$$m^*(y_{01p}^-) = \lim_{x \uparrow y_{01p}} \frac{f(x,x)}{g(x,x)} = \frac{(1-\beta^{|0p1|}) f(y_{01p}, y_{01p})}{1-\beta} = \frac{f(y_{01p}, y_{01p})}{g(y_{01p}, y_{01p})} = m^*(y_{01p}).$$

This completes the proof. ■

In order to show that the marginal productivity index $m(\cdot)$ is in fact Lipschitz continuous, Proposition 21 must consider how much $m(\cdot)$ changes over any finite interval $[a,b]$. The analysis proceeds by cutting up any such interval into pieces on each of which the length-k prefixes of the itineraries $[\sigma(\phi_a(u)|u)]_{1:k}$ for $a \in \{0,1\}$ are constant. The following lemma bounds how much $m(\cdot)$ can change over any one such interval.

Lemma 20 *Suppose A1 holds, that $k \in \mathbb{Z}_{++}$ and $0 \le x < y$ with $\sigma(\phi_a(x)|x)_{1:k} = \sigma(\phi_a(y)|y)_{1:k}$ for $a \in \{0,1\}$. Let $K := \sup_{z \in [\phi_1(x), \phi_0(y)]} C'(z)$ where $C'(\cdot)$ is the derivative of $C(\cdot)$. Then*

$$|m^*(x) - m^*(y)| \le \frac{K(3\beta^k(y+1) + 2(y-x))}{(1-\beta)^2}.$$

Proof Let $1p := [\sigma(\phi_0(x)|x)]_{1:k}$ and let s, s' be the shortest words such that $0\sigma_0(\phi_0(x)|x) = (01ps)^\infty$ and $0\sigma_0(\phi_0(y)|y) = (10ps')^\infty$. Also let

$$\begin{aligned}
a_n &:= C(\phi_{((01ps)^\infty)_{1:n}}(x)) - C(\phi_{((10ps)^\infty)_{1:k}}(x)), && e := (1-\beta^{|01ps|})/(1-\beta),\\
b_n &:= C(\phi_{((01ps')^\infty)_{1:n}}(y)) - C(\phi_{((10ps')^\infty)_{1:k}}(y)), && f := (1-\beta^{|01ps'|})/(1-\beta).
\end{aligned}$$

Then the simple bounds

$$\sup_{m \ge 1} |a_m| \le K(y+1), \qquad \sup_{m \ge 1} |a_m - b_m| \le 2K(y+1), \qquad \sup_{m \le k} |a_m - b_m| \le 2K(y-x)$$

follow from the facts that: (i) the lowest and highest points on the z-threshold orbit are $\phi_1(z)$ and $\phi_0(z)$ (by Lemma 39); (ii) $y \ge x$; (iii) function $C(\cdot)$ is non-decreasing and differentiable; (iv) function $\phi_w(\cdot)$ is non-expansive for any word w; (v) for any $z \in \mathbb{R}_+$, $\phi_0(z) \le z+1$ and $\phi_1(z) > 0$.

Also, as $|0p| = k$ it follows that $|e-f| \le \beta^k/(1-\beta)$. Therefore

$$\begin{aligned}
&\beta|m^*(x) - m^*(y)|\\
&\quad = \left| (e-f) \sum_{n=1}^{\infty} \beta^n a_n + f \sum_{n=1}^{k} \beta^n (a_n - b_n) + f \sum_{n=k+1}^{\infty} \beta^n (a_n - b_n) \right|
\end{aligned}$$

$$\leq |e-f| \sum_{n=1}^{\infty} \beta^n \sup_{m \geq 1} |a_m| + f \sum_{n=1}^{k} \beta^n \sup_{m \leq k} |a_m - b_m| + f \sum_{n=k+1}^{\infty} \beta^n \sup_{m > k} |a_m - b_m|$$
$$\leq \frac{\beta^k}{1-\beta} \frac{\beta}{1-\beta} K(y+1) + \frac{1}{1-\beta} \frac{\beta}{1-\beta} 2K(y-x) + \frac{1}{1-\beta} \frac{\beta^{k+1}}{1-\beta} 2K(y+1)$$

which rearranges to the inequality claimed. ■

Proposition 21 *Suppose A1 holds. Then the marginal productivity index m^* is Lipschitz continuous on any finite interval $[x, y]$ of $\mathcal{I}$.*

Proof Let $k \in \mathbb{Z}_{++}$. Now, the length-k prefixes of the itineraries $[\sigma(\phi_a(u)|u)]_{1:k}$ for $a \in \{0, 1\}$ are piecewise constant functions of $u \in \mathcal{I}$. Say there are n_k pieces as u increases from x to y and let $[a, b] \subseteq [x, y]$. Then Lemmas 19 and 20 together show that

$$|m^*(b) - m^*(a)| \leq \frac{3K(y+1)}{(1-\beta)^2} n_k \beta^k + \frac{2K}{(1-\beta)^2}(b-a)$$

where

$$K := \sup_{z \in [\phi_1(x), \phi_0(y)]} C'(z) < \infty.$$

However, the total number of pieces of f_k is at most a polynomial function $p(k)$ of k, by Theorem 14. As k was an arbitrary positive integer, it follows that

$$|m^*(b) - m^*(a)| \leq \frac{3K(y+1)}{(1-\beta)^2} \underbrace{\lim_{k \to \infty} \left(p(k) \beta^k \right)}_{=0} + \frac{2K}{(1-\beta)^2}(y-x).$$

As $[a, b] \subseteq [x, y]$ was arbitrary, we conclude that m^* is Lipschitz continuous on $[x, y]$. ■

3.3.2. The Marginal Productivity Index is Non-Decreasing

We use the following well-known result about *majorisation* (Marshall et al., 2010). A proof is given in Appendix B.

Lemma 22 *Suppose that:*

1. *The sequences $a_{1:n}$ and $b_{1:n}$ are non-decreasing sequences on $\mathbb{R}_{++}$*
2. *The inequality $\sum_{i=1}^{k} a_i \leq \sum_{i=1}^{k} b_i$ holds for $k = 1, 2, \ldots, n$*
3. *For $i = 1, 2, \ldots, n$, the function $f_i : \mathbb{R}_{++} \to \mathbb{R}$ is non-increasing and convex*
4. *For $i = 2, 3, \ldots, n$, the difference $f_{i-1}(x) - f_i(x)$ is non-increasing for $x \in \mathbb{R}_{++}$.*

Then

$$\sum_{i=1}^{n} f_i(a_i) \geq \sum_{i=1}^{n} f_i(b_i).$$

We apply this majorisation result to the sequences appearing in Lemma 24 below. To state that Lemma, we first define some matrices that are motivated by the form of the Kalman Filter variance updates.

Definition 23 *Let I be the 2-by-2 identity matrix. For $r \in (0, 1]$ and $0 \leq a \leq b$, let*

$$F := \begin{pmatrix} r & 1/r \\ ar & (a+1)/r \end{pmatrix}, \qquad G := \begin{pmatrix} r & 1/r \\ br & (b+1)/r \end{pmatrix}.$$

Let $M(\epsilon) = I, M(0) = F, M(1) = G$ and for any finite non-empty word w let

$$M(w) = M(w_{|w|}) \cdots M(w_2) M(w_1).$$

Thus, if $a = \theta_0, b = \theta_1$ and ϕ_0, ϕ_1 are as in Assumption A1, we have

$$\phi_0(x) = \frac{F_{11}x + F_{12}}{F_{21}x + F_{22}}, \qquad \phi_1(x) = \frac{G_{11}x + G_{12}}{G_{21}x + G_{22}}.$$

As remarked in Section 3, the central portion of Christoffel words are palindromes. The following result holds for any palindromes, not just palindromes generated by Christoffel words.

Lemma 24 *Suppose p is a palindrome, $r \in (0, 1]$, $n \in \mathbb{Z}_+$ and x satisfies*

$$\phi_p(0) \leq x \leq \phi_p\left(\frac{1}{1-r^2}\right).$$

Let $m := |01p|$ and for $k = 1, 2, \ldots, m$ let

$$\begin{pmatrix} a_k(x) \\ c_k(x) \end{pmatrix} := M\Big((01p)^n (01p)_{1:k}\Big) \begin{pmatrix} x \\ 1 \end{pmatrix}, \qquad \begin{pmatrix} b_k(x) \\ d_k(x) \end{pmatrix} := M\Big((10p)^n (10p)_{1:k}\Big) \begin{pmatrix} x \\ 1 \end{pmatrix}.$$

Then

1. *The sequences $a_{1:m}(x), b_{1:m}(x), c_{1:m}(x)$ and $d_{1:m}(x)$ are non-decreasing and positive*
2. *The inequality $a_k(x) \leq b_k(x)$ holds for $k = 1, 2, \ldots, m$*
3. *The inequality $\sum_{i=1}^{k} c_i(x) \leq \sum_{i=1}^{k} d_i(x)$ holds for $k = 1, 2, \ldots, m$*
4. *The inequalities $c_1(x) \leq d_1(x)$ and $c_k(x) \geq d_k(x)$ hold for $k = 2, 3, \ldots, m$.*
5. *The fixed points y_{01p} and y_{10p} satisfy*

$$\phi_p(0) \leq y_{01p} < y_{10p} \leq \phi_p\left(\frac{1}{1-r^2}\right)$$

so Claims 1-4 hold if $x \in [y_{01p}, y_{10p}]$.

The proof of the above lemma is given in Appendix C.

Now we are ready to demonstrate that marginal reward is non-decreasing for a wide range of cost functions on the intervals $[y_{01p}, y_{10p}]$ corresponding to a Christoffel word $0p1$ appearing in Theorem 12.

Lemma 25 *Suppose that A1 holds, that* $0p1$ *is a Christoffel word and that* $N \in \mathbb{Z}_+$. *Then*

$$\sum_{k=1}^{n} \beta^k (C(\phi_{(01p)^N(01p)_{1:k}}(x)) - C(\phi_{(10p)^N(10p)_{1:k}}(x)))$$

is a non-decreasing function of x *for* $y_{01p} \leq x \leq y_{10p}$, *where* $n = |0p1|$.

We prove the result for functions satisfying parts C1 and C2 of Assumption A1 separately, as sums of non-decreasing functions are non-decreasing, the result holds for any function satisfying Assumption A1.

Proof [Proof when $C(\cdot)$ satisfies part C1 of Assumption A1 (iv).] Let $a_k(x), b_k(x), c_k(x), d_k(x)$ be as defined in Lemma 24. Then the proposition is proved by the following inequalities (as justified immediately below):

$$\begin{aligned}
&\frac{d}{dx}\sum_{k=1}^{n} \beta^k (C(\phi_{(01p)^N(01p)_{1:k}}(x)) - C(\phi_{(10p)^N(10p)_{1:k}}(x))) \\
&\quad = \sum_{k=1}^{n} \left(\frac{\beta^k}{c_k(x)^2} C'\left(\frac{a_k(x)}{c_k(x)}\right) - \frac{\beta^k}{d_k(x)^2} C'\left(\frac{b_k(x)}{d_k(x)}\right)\right) && (23)\\
&\quad \geq \sum_{k=1}^{n} \left(\frac{\beta^k}{c_k(x)^2} C'\left(\frac{b_k(x)}{c_k(x)}\right) - \frac{\beta^k}{d_k(x)^2} C'\left(\frac{b_k(x)}{d_k(x)}\right)\right) && (24)\\
&\quad = \sum_{k=1}^{n} (f_k(c_k(x)) - f_k(d_k(x))) \quad \text{where } f_k(u) := \beta^k C'(b_k(x)/u)/u^2 \\
&\quad \geq 0. && (25)
\end{aligned}$$

Step (23) follows from the chain rule as the homomorphism of matrix multiplication and composition of Möbius transformations gives

$$\phi_{(01p)^n(01p)_{1:k}}(x) = \frac{\left[M((01p)^N(01p)_{1:k})\begin{pmatrix} x \\ 1 \end{pmatrix}\right]_1}{\left[M((01p)^N(01p)_{1:k})\begin{pmatrix} x \\ 1 \end{pmatrix}\right]_2} = \frac{a_k(x)}{c_k(x)}$$

while the fact that matrices F and G have unit determinant implies that the matrix $M((01p)^N(01p)_{1:k})$ also has unit determinant so that

$$\frac{d}{dx}\phi_{(01p)^N(01p)_{1:k}}(x) = \frac{1}{c_k(x)^2}.$$

Step (24) follows as $\beta^k, c_k(x) > 0$, as $C(\cdot)$ is concave and as $a_k(x) \leq b_k(x)$ by Lemma 24.

Step (25) follows from Lemma 22. In particular, Lemma 24 shows that the sequences $c_{1:n}(x)$ and $d_{1:n}(x)$ satisfy the hypotheses of Lemma 22. Also, the fact that $C(\cdot)$ satisfies part C1 shows that the functions $f_i(\cdot)$ for $i = 1, \ldots, n$ satisfy hypotheses 1 and 2 of Lemma 22. Indeed, $f_i(\cdot)$ is non-increasing and convex as $\frac{1}{u^2}C'\left(\frac{1}{u}\right)$ is non-increasing and convex and $\beta^i, b_i(x) > 0$. Also, as $\frac{1}{u^3}C''\left(\frac{b}{u}\right)$ is non-decreasing in u for $b > 0$ and $0 < b_{i-1}(x) \leq b_i(x)$ for $i = 2, \ldots, n$, by Claim 1 of Lemma 24, the following integral is also non-decreasing in u:

$$\int_{b_{i-1}(x)}^{b_i(x)} \frac{1}{u^3} C''\left(\frac{b}{u}\right) db = \frac{1}{u^2} C'\left(\frac{b_i(x)}{u}\right) - \frac{1}{u^2} C'\left(\frac{b_{i-1}(x)}{u}\right) = \frac{1}{\beta^i}\left(f_i(u) - \beta f_{i-1}(u)\right)$$

So $f_{i-1}(u) - f_i(u)$ is the sum of the non-increasing functions $\beta f_{i-1}(u) - f_i(u)$ and $(1-\beta)f_{i-1}(u)$.

This completes the proof. ■

Proof [Proof when $C(\cdot)$ satisfies part C2 of Assumption A1 (iv).] For $k \in \mathbb{Z}_n$ let

$$\begin{aligned}
a'_k &:= \beta^{k+1}\frac{d}{dx}\phi_{(01p)^N(01p)_{1:(k+1)}}(x), & b'_k &:= \beta^{k+1}\frac{d}{dx}\phi_{(10p)^N(10p)_{1:(k+1)}}(x),\\
a_k &:= C'(y_{(1p0)_{(k+1):n}(1p0)_{1:k}}), & b_k &:= C'(y_{(0p1)_{(k+1):n}(0p1)_{1:k}}).
\end{aligned}$$

Let $r_{[0]} \geq \cdots \geq r_{[n-1]}$ denote real numbers $r_0, \ldots, r_{n-1}$ in non-increasing numerical order. Let $\overline{x}$ denote x modulo n and let l satisfy $\overline{l|0p1|}_1 = 1$. Then the proposition is proved by the following inequalities (as justified immediately below):

$$\begin{aligned}
&\frac{d}{dx}\sum_{k=1}^{n} \beta^k \left(C\left(\phi_{(01p)^N(01p)_{1:k}}(x)\right) - C\left(\phi_{(10p)^N(10p)_{1:k}}(x)\right)\right)\\
&\quad = \sum_{k=1}^{n} \left(C'\left(\phi_{(01p)^N(01p)_{1:k}}(x)\right) a'_{k-1} - C'\left(\phi_{(10p)^N(10p)_{1:k}}(x)\right) b'_{k-1}\right) && (26)\\
&\quad \geq \sum_{k=1}^{n} \left(C'\left(y_{(1p0)_{k:n}(1p0)_{1:(k-1)}}\right) a'_{k-1} - C'\left(y_{(0p1)_{k:n}(0p1)_{1:(k-1)}}\right) b'_{k-1}\right) && (27)\\
&\quad = \sum_{k=0}^{n-1} \left(a_k a'_k - b_k b'_k\right) && (28)\\
&\quad = \sum_{k=0}^{n-1} a_{[k]}\left(a'_{\overline{l(n-k)}} - b'_{\overline{l(n-k-1)}}\right) && (29)\\
&\quad \geq \sum_{k=0}^{n-1} a_{[k]}\left(a'_{\overline{l(n-k)}} - b'_{\overline{l(n-k)}}\right) && (30)\\
&\quad \geq \sum_{k=0}^{n-1} a_{[0]}\left(a'_{\overline{l(n-k)}} - b'_{\overline{l(n-k)}}\right) && (31)\\
&\quad \geq 0 && (32)
\end{aligned}$$

Step (26) follows from the chain rule and definition of a'_k, b'_k.

Step (27) follows as $C(\cdot)$ is convex, as $a'_k, b'_k \geq 0$, and as, for $k = 1, 2, \ldots, n$, we have

$$\begin{aligned}\phi_{(01p)^N(01p)_{1:k}}(x) &= \phi_{(01p)_{1:k}}(\phi_{(01p)^N}(x)) \\ &\geq \phi_{(01p)_{1:k}}(y_{01p}) \\ &= y_{(01p)_{(k+1):n}(01p)_{1:k}} \\ &= y_{(1p0)_{k:n}(1p0)_{1:(k-1)}}\end{aligned}$$

where the inequality holds as $x \geq y_{01p}$ so that $\phi_{(01p)^N}(x) \geq y_{01p}$, and as $\phi_{(01p)_{1:k}}(\cdot)$ is increasing. The same argument using $x \leq y_{10p}$ gives an upper bound on $C'(\phi_{(10p)^N(10p)_{1:k}}(x))$.

Step (28) follows by shifting the summation indices and from the definition of a_k, b_k.

Step (29) follows from Lemmas 8 and 11 and the convexity of $C(\cdot)$. Let $w_{[0]} \succeq \cdots \succeq w_{[n-1]}$ denote words $w(0), \ldots, w(n-1)$ in non-increasing lexicographic order and let $c_k := (1p0)_{(k+1):n}(1p0)_{1:k}, d_k := (0p1)_{(k+1):n}(0p1)_{1:k}$ for $k \in \mathbb{Z}_n$. Then Lemma 8 shows that $c_{[i]} = d_{[i]} = c_{\overline{l(n-i)}} = d_{\overline{l(n-i-1)}}$. Thus Lemma 11 gives $y_{c_{[i]}} = y_{d_{[i]}} = y_{c_{\overline{l(n-i)}}} = y_{d_{\overline{l(n-i-1)}}}$. Therefore the convexity of $C(\cdot)$ gives

$$a_{[i]} = b_{[i]} = a_{\overline{l(n-i)}} = b_{\overline{l(n-i-1)}}.$$

Step (30) follows as $C(\cdot)$ is non-decreasing, so that $a_{[i]} \geq 0$, and as $\phi_0(\cdot), \phi_1(\cdot)$ are non-decreasing and non-expansive, so that b'_i is a product of derivatives where each derivative is in $[0, 1]$. Thus $a_{[0]}, \ldots, a_{[n-1]}$ and $b'_0, \ldots, b'_{n-1}$ are non-negative non-increasing sequences. Therefore the rearrangement inequality $a_{[i]}b'_j + a_{[i+1]}b'_0 \leq a_{[i]}b'_0 + a_{[i+1]}b'_j$ holds for all $i \in \mathbb{Z}_{n-1}$ and all $j \in \mathbb{Z}_n$. But $b'_{\overline{l(n-(n-1)-1)}} = b'_0$ so repeated application of the rearrangement inequality gives

$$\sum_{k=0}^{n-1} a_{[k]} b'_{\overline{l(n-k-1)}} \leq \sum_{k=0}^{n-1} a_{[k]} b'_{\overline{l(n-k)}}.$$

Step (31) follows as $a_{[0]} = a_{\overline{ln}} = a_0$, as a_0 is non-negative, and from Claim 4 of Lemma 24, as for the $c_i(x), d_i(x)$ defined in that lemma, we have

$$a'_{i-1} - b'_{i-1} = \frac{\beta^i}{c_i(x)^2} - \frac{\beta^i}{d_i(x)^2} \begin{cases} \geq 0 & \text{for } i = 1 \\ \leq 0 & \text{for } i = 2, 3, \ldots, n. \end{cases}$$

Step (32) follows from this lemma using the function $\tilde{C}(x) = x$ which satisfies part C1 and which was already proved above.

This completes the proof. ■

It is much simpler to show that the marginal reward is non-decreasing when the itinerary is 0^∞ (that is, $x \geq y_0$) or when the itinerary is 01^∞ (that is, $x \leq y_1$). Now the sums in the following lemma correspond to the numerator of the marginal productivity index m^* in these cases, up to a multiple of β. Also, the denominator of m^* in these cases is

$$\sum_{k=1}^{\infty} \beta^{k-1} \left((10^\infty)_k - (0^\infty)_k\right) = 1 - \beta = \sum_{k=1}^{\infty} \beta^{k-1} \left((1^\infty)_k - (01^\infty)_k\right)$$

which is positive. So, it follows from this lemma that m^* is non-decreasing on such intervals.

Lemma 26 *Suppose A1 holds. Then*

$$\sum_{k=1}^{\infty} \beta^k \left(C(\phi_{0^k}(x)) - C(\phi_{10^{k-1}}(x))\right) \qquad \text{and} \qquad \sum_{k=1}^{\infty} \beta^k \left(C(\phi_{01^{k-1}}(x)) - C(\phi_{1^k}(x))\right)$$

are non-decreasing functions of $x \in \mathcal{I}$.

Proof Consider the first sum. As in Lemma 24, for $k \in \mathbb{Z}_{++}$, we define

$$\begin{pmatrix} a_k(x) & b_k(x) \\ c_k(x) & d_k(x) \end{pmatrix} := \left(M(0^k) \begin{pmatrix} x \\ 1 \end{pmatrix} \quad M(10^{k-1}) \begin{pmatrix} x \\ 1 \end{pmatrix} \right).$$

As $G \geq F$, the entries of F are non-negative and $x \geq 0$, it follows that

$$\begin{pmatrix} b_k(x) - a_k(x) \\ d_k(x) - c_k(x) \end{pmatrix} = F^{k-1}(G - F) \begin{pmatrix} x \\ 1 \end{pmatrix} \geq 0. \tag{33}$$

If C satisfies part C1 of Assumption A1, then for any $k \in \mathbb{Z}_{++}$,

$$\begin{aligned}\frac{d}{dx}\left(C(\phi_{0^k}(x)) - C(\phi_{10^{k-1}}(x))\right) &= \frac{1}{c_k(x)^2} C'\left(\frac{a_k(x)}{c_k(x)}\right) - \frac{1}{d_k(x)^2} C'\left(\frac{b_k(x)}{d_k(x)}\right) \\ &\geq \frac{1}{c_k(x)^2} C'\left(\frac{b_k(x)}{c_k(x)}\right) - \frac{1}{d_k(x)^2} C'\left(\frac{b_k(x)}{d_k(x)}\right) \geq 0.\end{aligned}$$

The first inequality holds as C' is concave and $a_k(x) \leq b_k(x)$ by (33). The second inequality holds as $C'(1/x)/x^2$ is non-increasing and $c_k(x) \leq d_k(x)$ by (33).

If C satisfies part C2 of Assumption A1, then for any $k \in \mathbb{Z}_{++}$,

$$\frac{d}{dx}\left(C(\phi_{0^k}(x)) - C(\phi_{10^{k-1}}(x))\right) = \frac{1}{c_k(x)^2} C'(\phi_{0^k}(x)) - \frac{1}{d_k(x)^2} C'(\phi_{10^{k-1}}(x)) \geq 0.$$

The inequality is justified as follows. As $a \leq b$ in the definition of F, G, we have $\phi_0(x) \geq \phi_1(x)$. As $\phi_{0^{k-1}}(\cdot)$ is an increasing function, it follows that $\phi_{0^k}(x) \geq \phi_{10^{k-1}}(x)$. As C is convex, it follows that $C'(\phi_{0^k}(x)) \geq C'(\phi_{10^{k-1}}(x))$. Furthermore, $c_k(x) \leq d_k(x)$, by (33).

Thus, if C satisfies Assumption A1, the sum

$$\sum_{k=1}^{\infty} \beta^k \left(C(\phi_{0^k}(x)) - C(\phi_{10^{k-1}}(x))\right)$$

is the sum of non-decreasing functions. Therefore this sum is non-decreasing.

The proof for the second sum is similar. This completes the proof. ■

Proposition 27 *Suppose Assumption A1 holds. Then the marginal productivity index m^* is non-decreasing.*

Proof Consider any $a, b \in \mathcal{I}$ with $a < b$. By Proposition 21, the index m^* is Lipschitz continuous on any finite interval of $\mathcal{I}$, such as the interval $[a, b]$. Now, if a function is Lipschitz continuous on a closed interval then it is absolutely continuous on that interval (Royden and Fitzpatrick, 2010, Proposition 7, p. 120). Also, an absolutely continuous function m^* has a derivative $m^{*\prime}$ almost everywhere, which is Lebesgue integrable, and which satisfies (Pugh, 2002, Theorem 38, p. 399)

$$m^*(b) = m^*(a) + \int_a^b m^{*\prime}(t)\, dt. \tag{34}$$

Now, for any Christoffel word $0p1$ and for t in the interval $[y_{01p}, y_{10p}]$, the derivative $m^{*\prime}(t)$ is non-negative, since $m^*(t) = f(t,t)/g(t,t)$, in which $f'(t) \geq 0$ by Lemma 25, and in which $g(t,t)$ is a positive constant depending on $0p1$. Similarly, Lemma 26 shows that the marginal productivity index is non-decreasing on $\mathcal{I}\backslash(y_1, y_0)$. Therefore, $m^*(a) \leq m^*(b)$. This completes the proof. ■

Remark. Having proved that the marginal productivity index is non-decreasing, one might wonder if it is actually always strictly increasing. It is not. In particular, Assumption A1 allows the cost function $C(\cdot)$ to be a constant, in which case the marginal productivity index is a constant. In other cases, the marginal productivity index is not a constant, yet it is constant over a finite interval of the state space $\mathcal{I}$. For instance, this is the case for sufficiently large precision parameter θ_1 that $y_1 < 1$, and for the cost function

$$C(x) = \begin{cases} 0 & x < 1 \\ (x-1)^2 & x \geq 1, \end{cases}$$

which satisfies A1.

3.3.3. The Marginal Productivity Index is Bounded Below

Proposition 28 *Suppose Assumption A1 holds. Then the marginal productivity index m^* is bounded from below.*

Proof Let $[l, u] := \mathcal{I}$. By Proposition 27, m^* is non-decreasing, so

$$\inf_{x \in [l,u]} m^*(x) = f(l,l)/g(l,l).$$

Now $l \leq y_1$ so $\sigma(l|l) = 01^\infty$ and $\sigma(l|l^-) = 1^\infty$. Thus $g(l,l) = 1 - \beta$ and

$$f(l,l) = \sum_{t=0}^{\infty} \beta^t \left(C(\phi_{(01^\infty)_{1:t}}(l)) - C(\phi_{1^\infty_{1:t}}(l)) \right) \geq \frac{C(l) - C(y_1)}{1-\beta}$$

where the inequality follows as $\phi_{(01^\infty)_{1:t}}(l) \geq l$ and $\phi_{(1^\infty)_{1:t}}(l) \leq y_1$ for all t, and as C is increasing. As $C(l) > -\infty$, $C(y_1) < \infty$ and $\beta < 1$, it follows that $m^*(x) > -\infty$ for $x \in \mathcal{I}$. ■

3.4. Radon-Nikodym Condition (PCLI3)

We demonstrate that PCLI3 holds. The argument is based on the fact that the reward metric $F(x, \pi_s)$ and resource metric $G(x, \pi_s)$ for threshold policies are *jump functions* when viewed as a function of the threshold s, because the number of discontinuities of the itinerary map $s \mapsto \sigma(x|s)_{1:n}$ is bounded by a polynomial function of n by Theorem 14. First we recall what a jump function is.

Definition 29 (Kolmogorov and Fomin, 1975, p. 341) *A function $h : \mathbb{R} \to \mathbb{R}$ is a **jump function** if $h(x) = h_0 + \sum_{n=1}^{\infty} h_n \mathbf{1}_{x_n < x} + \sum_{n=1}^{\infty} h'_n \mathbf{1}_{x'_n \leq x}$ for some real-valued sequences $(h_i)_{i=0}^{\infty}$, $(h'_i)_{i=1}^{\infty}$, $(x_i)_{i=1}^{\infty}$ and $(x'_i)_{i=1}^{\infty}$ satisfying $\sum_{i=1}^{\infty} |h_i| < \infty$ and $\sum_{i=1}^{\infty} |h'_i| < \infty$.*

Proposition 30 *Suppose $\langle \mathcal{I}, C, \phi_0, \phi_1, \beta \rangle$ satisfy Assumption A1. Then PCLI3 holds.*

Proof Throughout the proof, let us consider a fixed initial state $x \in \mathcal{I}$. Recall the notation for the state $X_t(x; s)$ and action $A_t(x; s)$ from initial state x under the s-threshold policy discussed in the remark on page 30. Also recall also that a function $h : \mathbb{R} \to \mathbb{R}$ is *càdlàg* if both limits $h(s^-)$ and $h(s^+)$ exist and $h(s) = h(s^+)$ for all $s \in \mathbb{R}$.

First we use induction to show that $A_t(x; s)$ is a càdlàg function of s for any $t \in \mathbb{Z}_+$. In the base case, we have $X_0(x; s) = x$ and $A_0(x; s) = \mathbf{1}_{x>s}$, which are càdlàg in s. Now say $t \geq 0$ and $X_t(x; s)$ and $A_t(x; s)$ are càdlàg in s. Then, $\phi_a(X_t(x; s))$ is càdlàg in s for $a \in \{0, 1\}$ as it is the composition of a continuous function and a càdlàg function. Thus

$$X_{t+1}(x; s) = A_t(x; s)\phi_1(X_t(x; s)) + (1 - A_t(x; s))\phi_0(X_t(x; s))$$

is càdlàg in s, since sums and products of càdlàg functions are càdlàg. Hence $s - X_{t+1}(x; s)$ is càdlàg in s. Noting that $z \mapsto \mathbf{1}_{z<0}$ is càdlàg, it follows that the composition

$$A_{t+1}(x; s) = \mathbf{1}_{s - X_{t+1}(x;s) < 0}$$

is càdlàg in s, completing the induction.

Also, note that $\lim_{s \to -\infty} A_t(x; s) = 1$, since under Assumption A1, the state is never less than zero, so for any $s < 0$, the s-threshold policy always takes action 1.

For $t \in \mathbb{Z}_+$, let $\mathcal{D}_t(x)$ be a sequence consisting of all the values of the threshold s at which $A_t(x; s)$ is discontinuous. As $A_t(x; s)$ is a càdlàg function of s, taking values in $\{0, 1\}$, with $\lim_{s \to -\infty} A_t(x; s) = 1$, it follows from the definition of $\mathcal{D}_t(x)$ that

$$A_t(x; s) = 1 + \sum_{d \in \mathcal{D}_t(x)} (A_t(x; d) - A_t(x; d^-)) \mathbf{1}_{d \leq s},$$

noting that this sum is finite as $\mathrm{card}(\mathcal{D}_t(x)) \leq p(t)$ for some polynomial $p(t)$ by Theorem 14. Now let $d_1, d_2, \ldots$ be the sequence generated by first concatenating $\mathcal{D}_0(x)$ then $\mathcal{D}_1(x)$ then $\mathcal{D}_2(x)$ and so forth, then deduplicating that sequence by including only the first occurrence of each distinct element. Since the sequence $d_1, d_2, \ldots, d_k$ contains all of $\mathcal{D}_t(x)$ for some finite k, and because $A_t(x; d_i) - A_t(x; d_i^-) = 0$ for any d_i in the sequence which is not an element of $\mathcal{D}_t(x)$, it follows that

$$A_t(x; s) = 1 + \sum_{i=1}^{\infty} (A_t(x; d_i) - A_t(x; d_i^-)) \mathbf{1}_{d_i \leq s}.$$

Thus the resource metric is

$$G(x,\pi_s) = (1-\beta)^{-1} + \sum_{t=0}^{\infty}\sum_{i=1}^{\infty} \beta^t (A_t(x;d_i) - A_t(x;d_i^-))\mathbf{1}_{d_i \le s}.$$

As $\text{card}(\mathcal{D}_t(x)) \le p(t)$ for some polynomial $p(t)$ by Theorem 14 and $A_t(x;s) \in \{0,1\}$, we have

$$\sum_{t=0}^{\infty}\sum_{i=1}^{\infty} \left|\beta^t (A_t(x;d_i) - A_t(x;d_i^-))\mathbf{1}_{d_i \le s}\right| \le \sum_{t=0}^{\infty} \beta^t p(t) < \infty.$$

So Fubini's theorem gives

$$G(x,\pi_s) = (1-\beta)^{-1} + \sum_{i=1}^{\infty}\left(\sum_{t=0}^{\infty} \beta^t (A_t(x;d_i) - A_t(x;d_i^-))\right)\mathbf{1}_{d_i \le s}.$$

Thus the mapping $s \mapsto G(x,\pi_s)$ is a càdlàg jump function.

Let $\tau := \inf\{t > 0 : X_t(s;s^-) = s\}$ with $\tau = \infty$ if $X_t(s;s^-) \neq s$ for all $t > 0$. Then

$$\begin{aligned}
G(s,\pi_{s^-}) &= \sum_{t=0}^{\infty} \beta^t A_t(s;s^-) \\
&= \sum_{t=0}^{\infty} \beta^t A_t(s,1;s) + \sum_{t=\tau}^{\infty} \beta^t (A_t(s;s^-) - A_t(s,0;s)) \\
&= G(s,\langle 1,\pi_s\rangle) + \beta^\tau (G(s,\pi_{s^-}) - G(s,\langle 0,\pi_s\rangle)) \\
&= \frac{G(s,\langle 1,\pi_s\rangle) - \beta^\tau G(s,\langle 0,\pi_s\rangle)}{1-\beta^\tau}.
\end{aligned}$$

Let $\tau_1 := \inf\{t \ge 0 : X_t(x;s) = s\}$ with $\tau_1 = \infty$ if $X_t(x;s) \neq s$ for all $t \ge 0$. Then the expression for $G(x,\pi_{s^-})$ just derived gives

$$\begin{aligned}
G(x,\pi_s) - G(x,\pi_{s^-}) &= \beta^{\tau_1}(G(s,\pi_s) - G(s,\pi_{s^-})) \\
&= \beta^{\tau_1}\left(G(s,\langle 0,\pi_s\rangle) - \frac{G(s,\langle 1,\pi_s\rangle) - \beta^\tau G(s,\langle 0,\pi_s\rangle)}{1-\beta^\tau}\right) \\
&= -\frac{\beta^{\tau_1}}{1-\beta^\tau} g(s,s)
\end{aligned}$$

using definition of the marginal resource metric $g(\cdot,\cdot)$.

A similar argument shows that the mapping $s \mapsto F(x,\pi_s)$ for the reward metric is also a càdlàg jump function which can be described in terms of the same sequence of discontinuities $d_1, d_2, \ldots$ as $G(x,\pi_s)$. Furthermore, it is related to the marginal reward $f(s,s)$ through

$$F(x,\pi_s) - F(x,\pi_{s^-}) = -\frac{\beta^{\tau_1}}{1-\beta^\tau} f(s,s).$$

Recalling that $m^*(s) = f(s,s)/g(s,s)$ gives

$$F(x,\pi_s) - F(x,\pi_{s^-}) = m^*(s)(G(x,\pi_s) - G(x,\pi_{s^-})).$$

As $F(x,\pi_s), G(x,\pi_s)$ are càdlàg jump functions of s, the Lebesgue-Stieltjes integral of this expression over any interval $(s_1,s_2]$ of $\mathbb{R}$ is

$$F(x,\pi_{s_2}) - F(x,\pi_{s_1}) = \int_{(s_1,s_2]} m^*(s)\; dG(x,\pi_{ds}).$$

Therefore PCLI3 holds. ■

In the above proof, the expressions for the jumps $G(x,\pi_s) - G(x,\pi_{s^-})$ and $F(x,\pi_s) - F(x,\pi_{s^-})$ fit with a general pattern for such jumps given in Lemma 7.5 of Niño-Mora (2015). Also, Section 7.1 of that paper shows that the metrics $F(x,\pi_s)$ and $G(x,\pi_s)$ are càdlàg functions of the threshold s. However, while Appendix B of that paper also presents sufficient conditions for PCLI3 to hold, we were unable to apply those conditions to the model at hand.

3.5. Proof of Theorem 1

Proof First we show that the system considered in Theorem 1 can be seen as a project in the sense of Theorem 15, except that the former system may have a state space of the form $(0,\infty)$ which is not explicitly addressed by the verification theorem (Step 1). So, we then demonstrate that this choice of interval is in fact implicitly addressed by the verification theorem (Step 2). Finally, we recall that the PCLI conditions and A3 hold (Step 3) and explain the characterisation of the optimal thresholds (Step 4).

Step 1. The system given by $\mathcal{I}, C(\cdot), \phi_0(\cdot), \phi_1(\cdot)$ and factor β that is addressed in Theorem 1 is a project $\mathcal{P} = \langle \mathcal{X}, r, c, \mathscr{Q}, \beta\rangle$ with the same discount factor β, with $\mathcal{X} = \mathcal{I}$, $r(x,a) = -C(x)$, $c(x,a) = a$ and a transition law corresponding to deterministic transitions $\mathscr{Q}(\mathcal{B}|x,a) = \mathbf{1}_{\phi_a(x)\in\mathcal{B}}$ for each Borel subset $\mathcal{B}$ of $\mathcal{X}$, for all $x \in \mathcal{X}$ and $a \in \{0,1\}$.

Step 2. Assumption A1 allows the interval $\mathcal{I} = [0,\infty)$, which explicitly matches the assumptions of Theorem 15. The interval $\mathcal{I} = (0,\infty)$ is also allowed by A1. If Assumption A3 and the PCLI conditions are satisfied for a project $\mathcal{P} = \langle \mathcal{X}, r, c, \mathscr{Q}, \beta\rangle$ with state space $\mathcal{X} = (0,\infty)$ then they are also satisfied by the equivalent project $\tilde{\mathcal{P}} = \langle \tilde{\mathcal{X}}, \tilde{r}, \tilde{c}, \tilde{\mathscr{Q}}, \beta\rangle$ after the change of coordinates $\tilde{x} := \log(x)$. Now project $\tilde{\mathcal{P}}$ has state space $\tilde{\mathcal{X}} := (-\infty,\infty)$, so the conclusions of Theorem 15 apply to project $\tilde{\mathcal{P}}$ and therefore to project $\mathcal{P}$.

(To see this, note that continuity of $\tilde{r}(\tilde{x},a) = r(\exp(\tilde{x}),a)$, $\tilde{c}(\tilde{x},a) = c(\exp(\tilde{x}),a)$ follows from continuity of $\exp(\cdot), r(\cdot,a)$ and $c(\cdot,a)$. Also, for any Borel set $\tilde{\mathcal{B}}$ we have $\tilde{\mathscr{Q}}(\tilde{\mathcal{B}}|\tilde{x},a) = \mathscr{Q}(\{\exp(\tilde{y}) : \tilde{y} \in \tilde{\mathcal{B}}\}|\exp(\tilde{x}),a)$ and this is a valid transition law since the mapping $x \mapsto \log(x)$ is one-to-one. Further, if the weight function $w(\cdot)$ satisfies A3 for $\mathcal{P}$ then $\tilde{w}(\tilde{x}) := w(\exp(\tilde{x}))$ satisfies A3 for $\tilde{\mathcal{P}}$. Finally, for a threshold policy $\pi_s(x) = 1_{x>s}$ consider the threshold policy $\tilde{\pi}_{\tilde{s}}(\tilde{x}) := 1_{\tilde{x}>\tilde{s}}$ where $\tilde{s} := \log(s)$. Then the metrics $\tilde{F}(\cdot,\cdot), \tilde{G}(\cdot,\cdot), \tilde{f}(\cdot,\cdot), \tilde{g}(\cdot,\cdot)$ for $\tilde{\mathcal{P}}$ and $F(\cdot,\cdot), G(\cdot,\cdot), f(\cdot,\cdot), g(\cdot,\cdot)$ for $\mathcal{P}$ satisfy

$$\begin{aligned}
\tilde{F}(\tilde{x},\langle a,\tilde{\pi}_{\tilde{s}}\rangle) &= F(\exp(\tilde{x}),\langle a,\pi_s\rangle), & \tilde{G}(\tilde{x},\langle a,\tilde{\pi}_{\tilde{s}}\rangle) &= G(\exp(\tilde{x}),\langle a,\pi_s\rangle), \\
\tilde{f}(\tilde{x},\tilde{s}) &= f(\exp(\tilde{x}),\exp(\tilde{s})), & \tilde{g}(\tilde{x},\tilde{s}) &= g(\exp(\tilde{x}),\exp(\tilde{s})).
\end{aligned}$$

As $\exp(\cdot)$ is increasing and continuous, and as the PCLI conditions hold for $\mathcal{P}$, it follows that the PCLI conditions hold for $\tilde{\mathcal{P}}$.)

Step 3. Clearly the assumptions about C and w in A1 imply that A3 holds. Furthermore, the PCLI conditions hold by Propositions 18, 21, 27, 28 and 30. Thus it follows from Theorem 15 that the family of λ-price problems in equation (9) is indexable, that the Whittle index has the form claimed and that a threshold policy is optimal.

Step 4. Say $\lambda^*(s) = \lambda$. If $x < s$ then $\lambda^*(x) \leq \lambda$, as $\lambda^*(\cdot)$ is non-decreasing by PCLI2. Yet if $\lambda^*(x) \leq \lambda$ then action 0 is optimal, by the definition of indexability (Definition 3). A similar argument shows that action 1 is optimal for $x > s$. Also, indexability shows that both actions 0 and 1 are optimal at s. Therefore policies $\mathbf{1}_{X_t>s}$ and $\mathbf{1}_{X_t\geq s}$ are both optimal.

If $\lambda^*(s) > \lambda$ for all $s \in \mathcal{I}$ then action 0 is never optimal, by definition of indexability. Therefore the always-active policy is the unique optimal policy. The argument when $\lambda^*(s) < \lambda$ for all $s \in \mathcal{I}$ is similar.

This completes the proof. ■

3.6. Proof of Corollary 2

Proof As $\mathbb{E}[Z_t^2|H_t] = z_t^2 + v_t$, it is not hard to see that the problem corresponds to the dynamic program

$$V(z_t, v_t) = \min_{u_t\in\mathbb{R}, a_t\in\{0,1\}} \left\{ Dz_t^2 + Dv_t + Fu_t^2 + \lambda a_t + \beta\mathbb{E}[V(z_{t+1}, v_{t+1})|z_t, v_t, a_t, u_t] \right\} \quad (35)$$

where the expectation is over the following Markovian transitions of the information state:

$$\begin{aligned} z_{t+1}|z_t, v_t, a_t, u_t &\sim \mathcal{N}(\bar{A}z_t + \bar{B}u_t, \bar{A}^2 v_t + \Sigma_Z - \Phi_{a_t}(v_t)) \\ v_{t+1}|z_t, v_t, a_t, u_t &= \Phi_{a_t}(v_t). \end{aligned}$$

For trial solutions of the form $V(z, v) = Rz^2 + Rv + g(v)$, where $R \in \mathbb{R}$ and $g : \mathbb{R}_+ \to \mathbb{R}$ are to be determined, the expectation in (35) is

$$\begin{aligned} &\mathbb{E}[V(z_{t+1}, v_{t+1})|z_t, v_t, a_t, u_t] \\ &\qquad = R((\bar{A}z_{t+1} + \bar{B}u_t)^2 + \bar{A}^2 v_t + \Sigma_Z - \Phi_{a_t}(v_t) + \Phi_{a_t}(v_t)) + g(\Phi_{a_t}(v_t)). \end{aligned}$$

Thus (35) is solved if

$$\begin{aligned} Rz_t^2 + Rv_t + g(v_t) = \min_{u_t\in\mathbb{R}} &\left\{ Dz_t^2 + Fu_t^2 + \beta R(\bar{A}z_{t+1} + \bar{B}u_t)^2 \right\} \\ &+ \min_{a_t\in\{0,1\}} \left\{ \lambda a_t + \beta R\Sigma_Z + (D + \beta R\bar{A}^2)v_t + \beta g(\Phi_{a_t}(v_t)) \right\}. \end{aligned} \quad (36)$$

Now the minimum with respect to u_t is achieved if the coefficient $(F + \beta\bar{B}^2 R)$ of u_t^2 is positive, in which case the minimiser is

$$u_t = -\frac{\beta\bar{A}\bar{B}R}{F + \beta\bar{B}^2 R} z_t.$$

So (36) is solved if R satisfies

$$R = D + F\left(\frac{\beta\bar{A}\bar{B}R}{F + \beta\bar{B}^2 R}\right)^2 + \beta R\left(\bar{A} - \bar{B}\frac{\beta\bar{A}\bar{B}R}{F + \beta\bar{B}^2 R}\right)^2$$

and if $g(\cdot)$ satisfies the dynamic program

$$g(v) = \min_{a\in\{0,1\}} \left\{ \lambda a + \beta R\Sigma_Z + \alpha v + \beta g(\Phi_a(v)) \right\} \quad (37)$$

where $\alpha := D - (1 - \beta\bar{A}^2)R$.

After simple algebra, the condition on R is equivalent to the quadratic equation

$$-\beta\bar{B}^2 R^2 + (\beta\bar{B}^2 D + \beta\bar{A}^2 F - F)R + DF = 0.$$

Using Descartes' rule of signs and considering the cases $F = 0$ and $F > 0$ separately, we see that this equation has a unique positive root for $\beta\bar{B}^2 > 0$ and $D > 0$.

To apply Theorem 1 to the dynamic program for $g(\cdot)$ we must ensure that Assumption A1 holds, which is not hard to see provided that $\alpha \geq 0$. Noting that $m := 1-\beta\bar{A}^2 > 0$ by the hypotheses about $\bar{A}, \beta$, we see that $\alpha \geq 0$ if $R \leq D/m$. But, substituting $R = y+(D/m)$ in the equation for R gives

$$\begin{aligned} 0 &= [-\beta\bar{B}^2 R^2 + (\beta\bar{B}^2 D + \beta\bar{A}^2 F - F)R + DF]_{R=y+(D/m)} \\ &= -\beta\bar{B}^2 y^2 - ((\beta^2\bar{A}^2 + \beta)\bar{B}^2(D/m) + mF)y - \beta^2\bar{A}^2\bar{B}^2(D/m)^2 \end{aligned}$$

in which the coefficients of y^0, y^1, y^2 are all negative by the hypotheses about $\bar{A}, \bar{B}, D, F$ and β. So this quadratic equation for y has no positive roots and it follows that $\alpha \geq 0$. Therefore Theorem 1 shows that there is an optimal policy for (37) that sets $a_t = 1$ if and only if $v_t \geq s$ for some $s \in \overline{\mathbb{R}}$. This completes the proof. ■

4. Numerical Experiments

We discuss algorithms for computing the Whittle index given in Theorem 1, we present closed-form expressions for that index and compare the performance of the Whittle index policy with two previously-proposed heuristics.

4.1. Approximating the Index for Discount Factor $\beta \leq 0.999$

Truncating the sums in the Whittle index $\lambda^*(x)$ after a suitably large number of terms T suggests the approximation

$$\hat{\lambda}(x) = \frac{\sum_{t=0}^{T} \beta^t \big(C(X_t(x, 0; x)) - C(X_t(x, 1; x))\big)}{\sum_{t=0}^{T} \beta^t \big(A_t(x, 1; x) - A_t(x, 0; x)\big)}. \quad (38)$$

Assuming accurate calculation of the terms in the numerator and denominator, as well as continuity of the cost function C, this approximation requires $O(T)$ basic arithmetical and

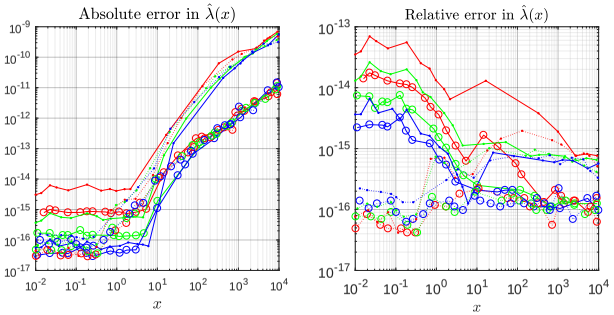


Figure 6: Errors in approximating the index. The cost $C(x)$ corresponds to variance ($C(x) = x$, red), to entropy ($C(x) = \log(x)$, green) or to negative precision ($C(x) = -1/x$, blue). The discount factor β is either 0.9 (open circles) or 0.999 (filled circles). The map-with-a-gap has $\phi_0(x) = x + 1$ and $\phi_1(x) = 1/(\theta_1 + 1/(x + 1))$ with θ_1 equal to 0.001 (solid lines) or 1 (dotted lines), where $\phi_a(\cdot)$ and θ_a for $a \in \{0, 1\}$ are as in Assumption A1.

comparison operations, and setting T to $\Omega(\log(\epsilon)/\log(\beta))$ guarantees absolute errors in the numerator and denominator of $O(\epsilon)$. Of course, the constants hidden by the $O(\cdot)$ or $\Omega(\cdot)$ depend on the detailed properties of C and choice of x.

However, this approximation faces a potential complication. Indeed, some of the iterates $X_t(x, a; x)$ may be so close to the threshold x that an arbitrarily small tolerance is required to correctly decide whether $X_t(x, a; x) \geq x$. This might be problematic as errors in such decisions can result in large changes to the numerator and denominator of this approximation.

Dance and Silander (2015) overcame this problem by constraining the sequence of decisions to correspond to $\mathcal{M}$-words. This resulted in a polynomial-time algorithm for approximating the index $\lambda^*(x)$ using variable-precision arithmetic. Specifically, suppose that basic arithmetic operations to tolerance 2^{-m} on positive numbers less than 2^n require at most $M(n + m)$ operations, and that $C(x) = x$. Then, the approximate index $\hat{\lambda}(x)$ output by their algorithm satisfies $\left|\hat{\lambda}(x) - \lambda^*(x)\right| < \epsilon$ when $x < 2^n$ and $\epsilon > 2^{-m}$ and is computed in $O((n + m)^3 M(n + m))$ operations. Nevertheless, that algorithm requires the tabulation of the fixed points of all Christoffel words of at most a given length.

Here we suggest that such tabulation is an unnecessary expense, and conjecture that standard floating-point approximations of the decision sequences $A_t(x, a; x)$ correspond to $\mathcal{M}$-words (at least for the vast majority of floating point values of x). Perhaps such a conjecture might be proven by extending results by Kozyakin (2003). His results concern mappings ϕ_0, ϕ_1 which are strictly increasing but potentially discontinuous. In the floating-point case

one would only require those mappings to be non-decreasing and piecewise constant, but might perhaps impose additional conditions.

Rather than attempting to prove such a conjecture here, we simply evaluate the accuracy of approximation (38). Figure 6 shows the accuracy based on comparing double-precision and quadruple-precision implementations, with $T = \lceil \log 10^{-17}/\log \beta \rceil$ and $T = \lceil \log 10^{-34}/\log \beta \rceil$ respectively. As the difference between these approximations is a highly variable function of x, we only show points that are local maxima of the error. Specifically, we used a logarithmically-spaced grid with $10^{-2} = x_1, x_2, \ldots, x_{1000} = 10^4$ and plot the error $e(x_i)$ only for points with $e(x_i) = \max_{i-20 \leq j \leq i+20} e(x_j)$. The plot shows no line for x less than the first such point or greater than the last such point.

The worst absolute and relative errors are below 10^{-6} and 10^{-11} respectively. In any practical application, such errors would be swamped by imprecision in the time-series models. The absolute error remains small as x increases to the fixed point y_1 of the mapping ϕ_1, and then it increases due to round-off in computing iterates of the map-with-a-gap for large x. Overall, the worst results are for large discount factors and for variance as the cost function.

Finally, it is possible to substantially accelerate the convergence of the numerator, for instance with Aitken acceleration (Brezinski, 2000), particularly if one has high accuracy requirements. For instance, if the itinerary $\sigma(x|x)$ is periodic with small enough period n, one may accumulate n terms of the sum at a time and apply acceleration methods to such partial sums. Having experimented with such approaches, we find that further work is required in selecting appropriate termination conditions if one is interested in accuracy guarantees for a wide range of problem instances. The difficulty we encountered is that *two* types of linear convergence are going on simultaneously, namely the convergence due to ϕ_0 (when $\theta_0 > 0$ or $r < 1$) and ϕ_1, and the convergence due to β. In such situations, what looks like a healthy stopping time to existing termination criteria can actually be a misleading and unhealthy prematurity.

4.2. Approximating the Index for Discount Factor $\beta \to 1$

For discount factors $\beta > 0.999$ the number of terms T required for accuracy in approximation (38) becomes prohibitively large. In such cases, it makes sense to Taylor expand the numerator of λ as a function of β. For brevity, we only do so here for the case $\beta \to 1$.

Suppose the actions $A_t(x, a; x)$ in the expression in Theorem 1 for the index $\lambda^*(x)$ correspond to a Christoffel word of length n. By (12) the denominator of the index is

$$\lim_{\beta \to 1} \frac{1 - \beta}{1 - \beta^n} = \frac{1}{n}.$$

Let $X_t^{a,\infty} := \lim_{k \to \infty} X_{kn+t}(x, a; x)$. Now, as $C(\cdot)$ is continuous by Assumption A1, we have

$$\lim_{k \to \infty} C(X_{kn+t}(x, a; x)) = C(\lim_{k \to \infty} C(X_{kn+t}(x, a; x)) = C(X_t^{a,\infty}). \tag{39}$$

Thus, putting $\Delta_t := C(X_t^{1,\infty}) - C(X_t^{0,\infty})$, the numerator of index has the limit

$$\lim_{\beta \to 1} \sum_{t=0}^{\infty} \beta^t \left(C(X_t(x, 0; x)) - C(X_t(x, 1; x)) \right)$$

$$= \underbrace{\sum_{t=0}^{\infty} \left(C(X_t(x,0;x)) - C(X_t(x,1;x)) + \Delta_t\right)}_{=:T_1} - \lim_{\beta\to 1} \sum_{k=0}^{\infty} \beta^{kn} \sum_{t=0}^{n-1} \beta^t \Delta_t$$
$$= T_1 - \lim_{\beta\to 1} \frac{\sum_{t=0}^{n-1} \beta^t \Delta_t}{1-\beta^n}$$
$$= T_1 + \frac{1}{n} \sum_{t=0}^{n-1} t\Delta_t$$

having applied L'Hôpital's rule. Now (39) shows that the sequences $C(X_t(x,a;x)) - C(X_t^{a,\infty})$ appearing in sum T_1 converge to zero. This suggests the approximation

$$X_{kn+t}^{a,\infty} \approx X_{Tn-n+t}(x,a;x) \qquad (a \in \{0,1\}, k \in \mathbb{Z}_+, t = 0,1,\ldots,n-1)$$

for a suitably large positive integer T. This also suggests approximating the first sum by truncating it after Tn terms. While the itinerary for x might have a very large and possibly infinite period n, we did not encounter such situations when tabulating the index. If this were an issue, it is possible to find a good rational approximation to the rate of the x-threshold itinerary from initial state x, for instance as in Dance and Silander (2015). To summarise, given an appropriate choice of n and T, the discussion above suggests approximating the index by

$$\hat{\lambda}(x) := n \sum_{k=0}^{T-1} \sum_{t=0}^{n-1} \left(\tilde{C}_{kn+t}^0 - \tilde{C}_{Tn-n+t}^0 - \tilde{C}_{kn+t}^1 + \tilde{C}_{Tn-n+t}^1\right) + \sum_{t=0}^{n-1} t \left(\tilde{C}_{Tn-n+t}^1 - \tilde{C}_{Tn-n+t}^0\right)$$

where $\tilde{C}_k^a := C(X_t(x,a;x))$ for $k \in \mathbb{Z}_+$ and $a \in \{0,1\}$.

4.3. Closed Form Expressions and Graphs

We analyse the behaviour of the index as the cost function $C(\cdot)$ and parameters $\beta, r, \theta_0, \theta_1$ vary.

Given noise free observations for action $a = 1$ and totally uninformative observations for action $a = 0$, it is easy to find a closed form for the index.

Proposition 31 *Suppose the cost function is $C(x) = x$ and the precision $\theta_0 = 0$. Then*

$$\lim_{\theta_1\to\infty} \lambda^*(x) = \frac{1-\beta^{n+1}}{1-\beta}\left(rx + 1 - \frac{\beta}{1-\beta^{n+1}}\left(\frac{1-(\beta r)^n}{1-\beta r} - \beta^n \frac{1-r^n}{1-r}\right)\right),$$

for all $\beta \in [0,1)$, $r \in [0,1)$ and $x \in [0, 1/(1-r))$, where $n := \left\lceil \frac{\log(1-(1-r)x)}{\log r} \right\rceil$. Thus

$$\lim_{\beta\to 1} \lim_{r\to 1} \lim_{\theta_1\to\infty} \lambda^*(x) = \lceil x+1 \rceil \left(x + 1 - \frac{\lceil x \rceil}{2}\right) = \int_0^{x+1} \lceil u \rceil \, du, \qquad \text{for all } x \in \mathbb{R}_+.$$

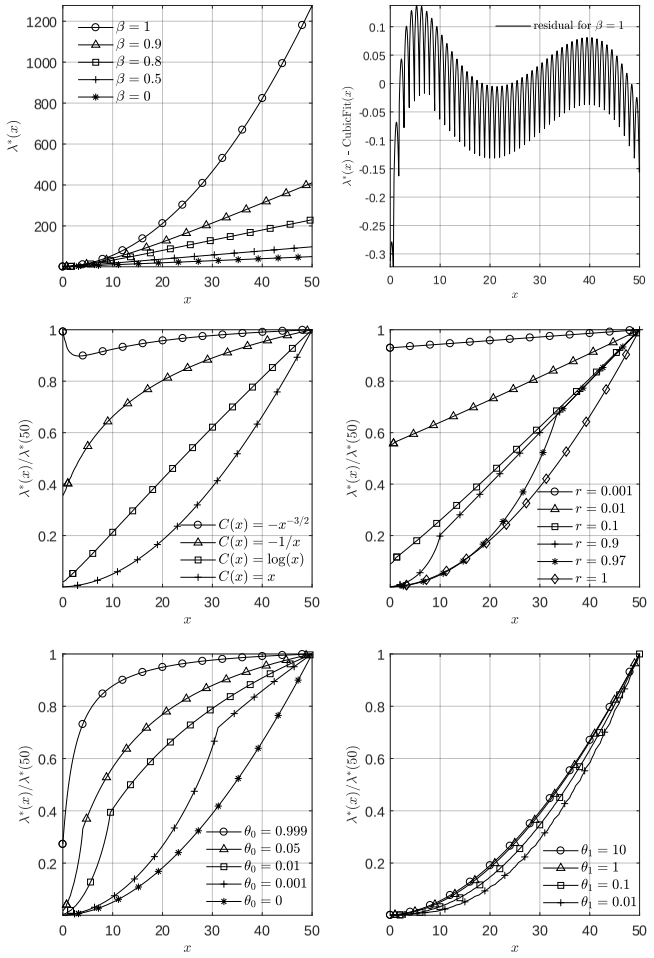


Figure 7: Behaviour of the index. The index as a function of the discount factor β (*top-left*) and the residual after fitting a cubic to this curve in the case $\beta = 1$ (*top-right*). The index, normalised by $\lambda^*(50)$, as the cost function $C(x)$, the multiplier r and the observation precisions θ_0 and θ_1 are varied (*other plots*). In all plots, all parameters (or function) other than that varied are set to $\beta = 0.99$, $C(x) = x$, $\phi_0(x) = 1/(\theta_0 + 1/(rx+1))$ and $\phi_1(x) = 1/(\theta_1 + 1/(rx+1))$, with $\theta_0 = 0$, $\theta_1 = 1$ and $r = 1$.

Proof As $\lim_{\theta_1 \to \infty} \phi_1(x) = 0$, the orbits involve the sequence $0, 1, r+1, r^2+r+1, \ldots, r^{n-1}+\cdots+r+1$, where n is the least integer for which $r^{n-1}+\cdots+r+1 \geq x$. The result then follows from the definition of the index, using well-known summation formulae. ■

Other closed forms exist, for instance in the limit $\beta \to 0$, or whenever the cost is a polynomial function of x or whenever the process tends to the continuous-time process analysed in Le Ny et al. (2011). However, the usefulness of the above proposition to provide practical approximations in circumstances where observations are not perfectly precise, so that $\theta_1 < \infty$, is limited. For instance, if $C(x) = x, \beta \to 1, r = 1, \theta_0 = 0$ and $\theta_1 = 4$, then we have

$$\lambda^*(0.2) = 1.01\cdots, \quad \int_0^{1.2} \lceil u \rceil \, du = 1.4$$

which corresponds to a relative error of over 38%.

Therefore, Figure 7 graphs the index using the algorithms of the previous subsection. Looking at these graphs, one notices that the index is increasing in all-but-one of the cases shown: indeed $C(x) = -x^{-3/2}$ is *not* covered by Assumption A1. Also, the index has cusps at the fixed point $x = \phi_0(x)$ which are clearly visible as θ_0 and r vary. Finally, the index becomes increasingly serrated as $\beta \to 1$ and $\theta_1 \to 0$. One would anticipate such serrations on the basis of the above proposition as

$$\int_0^{x+1} \lceil u \rceil \, du \; - \; \frac{1}{2}(x+1)(x+2) = \frac{1}{2}(\lceil x \rceil - x)(x - \lfloor x \rfloor)$$

and they are visible in the plot of the residual after subtracting a cubic fit, for $\theta_1 = 1$ and $\beta \to 1$. However, in general, the serrations have a complex non-periodic pattern and give a slightly ragged appearance to the plot with $\theta_1 = 0.01$.

4.4. Performance Relative to Heuristic Policies

Two heuristic policies have been commonly used for the problem of multi-sensor scheduling. The *myopic* policy observes the m time series with the highest current cost $C_i(x_{i,t})$ and has been used in radar systems (Moran et al., 2008). Meanwhile, the *round robin* policy chooses an ordering of the n projects and observes the next m projects, according to this ordering, at each time.

Figure 8 compares the costs incurred by these heuristics in a simple scenario in which uncertainty about one of the projects is more expensive than for the other projects. In detail, there are $n = 10$ projects, the number of observations per round is $m = 1$, the cost at time t is $40x_{1,t} + \sum_{i=2}^{n} x_{i,t}$, and the initial variance is $x_{i,0} = 4$ for all projects. The discount factor is $\beta = 0.99$ and the "total cost" for a given method is computed as $\sum_{t=0}^{199} \beta^t (40x_{1,t} + \sum_{i=2}^{n} x_{i,t})$. The projects have the same map-with-a-gap given by $\phi_0(x) = x+1$ and $\phi_1(x) = 1/(0.1 + 1/(x+1))$. The colours in this plot represent the variance state $x_{i,t}$.

Clearly, the myopic policy is overeager to observe project $i = 1$ and does so at the expense of projects $i = 2, \ldots, 10$. In contrast, the round robin policy makes no special effort to observe project $i = 1$ and incurs substantial expense for that project. Meanwhile,

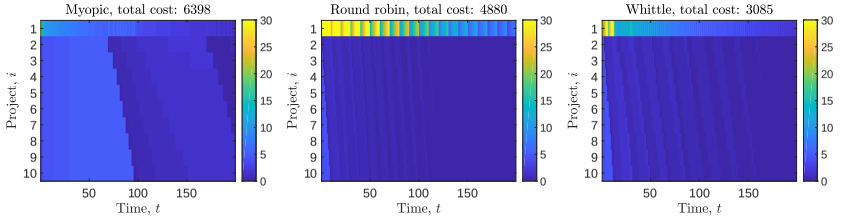


Figure 8: Comparison of heuristic policies. Colour represents the variance state $x_{i,t}$.

the Whittle index policy's behaviour lies between these extremes and it is by far the least costly policy. Indeed the ratio of the total cost of the round robin policy to the total cost of the Whittle index policy is just over 1.5, whereas the ratio of the total cost for the myopic policy to the total cost for the Whittle index policy is just over 2.

5. Further Work

This paper presented conditions under which threshold policies are optimal for observing a single time series with costly observations. It also explored the implications of this result by showing that it leads to optimal policies for the linear-quadratic Gaussian problem with costly observations and that it demonstrates the indexability of related restless bandit problems, which were both long-standing open questions.

It would be natural to extend this work to situations where more than two observation actions are available, perhaps using known generalisations of Sturmian words (Glen and Justin, 2009). Additionally, it would be interesting to complete the work by Le Ny et al. (2011) analysing optimal policies and indexability for scalar-valued time series in continuous time. In particular, there are a number of gaps in their argument to close, as discussed on page 12, and there is as yet no analysis of the discounted case nor of the class of cost function for which threshold policies are optimal. There are also truly-stochastic versions of the one-dimensional problem considered here. These include situations where the costs depend on the posterior mean rather than just the posterior variance (Kuhn et al., 2014), situations where the quality of the observation is a random variable, as well as situations involving non-Gaussian time series. It is also important to understand the structure of optimal policies for making costly observations with discrete-time Kalman filters in *multiple* dimensions. One attack on this problem would begin by extending the verification theorem of Niño-Mora (2015) to multi-dimensional state spaces for a variety of extensions of the notion of a "threshold policy". Finally, we cannot claim the asymptotic optimality of Whittle's index policy for the problem studied here as the results of Verloop (2016) only apply to *countable* state spaces. Furthermore, little is known about the performance of policies for restless bandits in *non-asymptotic* situations involving finite numbers of projects and finite time horizons.

Acknowledgments

We thank José Niño-Mora for pointing out his work on the verification theorem, Sofia Michel for her careful reading of an initial draft of the manuscript, and the reviewers for their tenacity and numerous helpful suggestions. Much of the work described in this paper was done while the authors were with Xerox Research Centre Europe.

Appendix A. Itineraries as Words

We begin with some elementary properties of compositions of functions satisfying Assumption A2 and their fixed points (Section A.1). This enables us to present a proof of Theorem 12 which is based on the Christoffel tree. First we consider the case where itineraries correspond to Christoffel words (Section A.2), then the case where itineraries correspond to Sturmian $\mathcal{M}$-words (Section A.3) and finally we couple these results together to prove the theorem (Subsection A.4). We also present proofs of Corollary 13 concerning x-threshold itineraries from initial state $\phi_0(x)$ and $\phi_1(x)$ (Section A.5) and of Theorem 14 about itineraries from general initial points not equal to the threshold (Section A.6) .

A.1. Compositions and Fixed Points

The following fundamental result about compositions is well known.

Lemma 32 *Suppose A2 holds and that w is a finite non-empty word. Then ϕ_w is increasing, contractive, continuous and has a unique fixed point y_w on $\mathcal{I}$.*

Proof First we show that $\phi_w(x)$ is increasing, by induction on the length of word w. In the base case, as w is non-empty, we suppose that $|w| = 1$. The claim then follows immediately from A2. For the inductive step, assume $\phi_u(x)$ is increasing, where $w = au$ for some letter $a \in \{0, 1\}$ and some finite non-empty word u. Then for any $x, y \in \mathcal{I}$ such that $x < y$,

$$\begin{aligned} \phi_w(y) &= \phi_u(\phi_a(y)) \\ &> \phi_u(\phi_a(x)) && \text{as } \phi_a(y) > \phi_a(x) \text{ and } \phi_u \text{ is increasing} \\ &= \phi_w(x). \end{aligned}$$

Therefore ϕ_w is increasing.

Now we show that $\phi_w(x)$ is contractive, again by induction on $|w|$. If $|w| = 1$ then this follows immediately from A2. Else, say $\phi_u(x)$ is contractive where $w = ua$ and $a \in \{0, 1\}$. Then for any $x, y \in \mathcal{I}$ such that $x < y$,

$$\begin{aligned} \phi_w(y) - \phi_w(x) &= \phi_a(\phi_u(y)) - \phi_a(\phi_u(x)) \\ &< \phi_u(y) - \phi_u(x) && \text{as } \phi_u(y) > \phi_u(x) \text{ and } \phi_a \text{ is contractive} \\ &< y - x && \text{as } \phi_u \text{ is contractive.} \end{aligned}$$

Therefore ϕ_w is contractive.

As ϕ_w is contractive, for any $\epsilon > 0$ and $c \in \mathcal{I}$, it follows that

$$|\phi_w(x) - \phi_w(c)| < |x - c| < \epsilon \quad \text{for any } x \in \mathcal{I} \text{ with } |x - c| < \epsilon.$$

Therefore ϕ_w is continuous.

Now we show that the fixed point y_w exists, using the intermediate value theorem applied to the function $g(x) := x - \phi_w(x)$. First we show that $g(y_0) \geq 0$. Indeed, as ϕ_1 is contractive, the definition of y_1 gives

$$y_0 - y_1 > \phi_1(y_0) - \phi_1(y_1) = \phi_1(y_0) - y_1, \quad \text{so that} \quad \phi_1(y_0) < y_0.$$

So, it follows from the definition of y_0 that the upper bound $\psi(x) := \max\{\phi_0(x), \phi_1(x)\}$ satisfies $\psi(y_0) = \phi_0(y_0) = y_0$. As $\phi_u(x)$ is increasing for any finite word u, it follows that

$$\phi_w(y_0) = \phi_{w_{2:|w|}} \circ \phi_{w_1}(y_0) \leq \phi_{w_{2:|w|}} \circ \psi(y_0) = \phi_{w_{2:|w|}}(y_0) \leq \cdots \leq y_0,$$

so that

$$g(y_0) = y_0 - \phi_w(y_0) \geq y_0 - y_0 = 0.$$

A similar argument, using the lower bound $\min\{\phi_0(x), \phi_1(x)\}$, gives $g(y_1) \leq 0$. In summary, $g(y_1) \leq 0 \leq g(y_0)$, where $y_1 < y_0$ by A2, and $g(x)$ is continuous as $\phi_w(x)$ is continuous. So the intermediate value theorem shows that $g(y) = 0$ for some $y \in [y_1, y_0]$. Therefore a fixed point y_w exists on $\mathcal{I}$.

Now we show that the fixed point y_w is unique. Suppose both y and z are fixed points of ϕ_w with $y > z$. This leads to the following contradiction: as ϕ_w is contractive we have

$$\frac{\phi_w(y) - \phi_w(z)}{y - z} < 1,$$

yet as $\phi_w(y) = y$ and $\phi_w(z) = z$ we have

$$\frac{\phi_w(y) - \phi_w(z)}{y - z} = 1.$$

Therefore the fixed point is unique. This completes the proof. ■

We make widespread use of the following simple lemma. Given a word w, this lemma gives necessary and sufficient conditions for $\phi_w(x)$ to be greater than or less than x.

Lemma 33 *Suppose A2 holds, that $x \in \mathcal{I}$ and w is a finite non-empty word. Then*

$$x < \phi_w(x) \;\Leftrightarrow\; \phi_w(x) < y_w \;\Leftrightarrow\; x < y_w \quad \textit{and} \quad x > \phi_w(x) \;\Leftrightarrow\; \phi_w(x) > y_w \;\Leftrightarrow\; x > y_w.$$

Proof We use Lemma 32 and the definition of y_w throughout without further mention.

As ϕ_w is increasing, if $x < y_w$ then $\phi_w(x) < \phi_w(y_w) = y_w$. Similarly, if $x > y_w$ then $\phi_w(x) > y_w$. Thus if $\phi_w(x) \leq y_w$ then $x \leq y_w$, by the contrapositive. But if $\phi_w(x) \neq y_w$ then $x \neq y_w$, as ϕ_w is increasing and therefore injective. So if $\phi_w(x) < y_w$ then $x < y_w$. Therefore

$$x < y_w \quad \Leftrightarrow \quad \phi_w(x) < y_w.$$

As ϕ_w is contractive, if $x < y_w$ then $\phi_w(y_w) - \phi_w(x) < y_w - x$. As $\phi_w(y_w) = y_w$, this rearranges to give $x < \phi_w(x)$. Similarly, if $x > y_w$ then $x > \phi_w(x)$. Thus if $x \leq \phi_w(x)$ then $x \leq y_w$, by the contrapositive. But if $x \neq \phi_w(x)$ then x is not a fixed point, so $x \neq y_w$. So if $x < \phi_w(x)$ then $x < y_w$. Therefore

$$x < y_w \quad \Leftrightarrow \quad x < \phi_w(x).$$

A similar argument shows that $x > y_w \Leftrightarrow \phi_w(x) > y_w$ and $x > y_w \Leftrightarrow x > \phi_w(x)$. ■

Lemma 34 *Suppose A2 holds and w is a finite word with $|w|_0|w|_1 > 0$. Then*

$$y_1 < y_w < y_0.$$

Proof As $|w|_0 > 0$ we have $w =: s01^q$ for some finite word s and some $q \in \mathbb{Z}_+$. As $y_0 > y_1$ by A2, Lemma 33 gives $\phi_0(y_1) > y_1$. Thus the definition of y_1 and the fact that ϕ_s is increasing give

$$\phi_w(y_1) = \phi_{s01^q}(y_1) = \phi_{s0}(y_1) = \phi_s(\phi_0(y_1)) > \phi_s(y_1) \geq y_1$$

where the last step follows by repeating the same argument. But if $\phi_w(y_1) > y_1$ then Lemma 33 shows that $y_w > y_1$.

A similar argument leads to the conclusion that $y_w < y_0$. ■

Lemma 35 *Suppose A2 holds, $x \in \mathcal{I}$ and w is a finite non-empty word. Then*

$$\lim_{n\to\infty} \phi_{w^n}(x) = y_w.$$

Proof The sequence with elements $x_n := \phi_{w^n}(x)$ for $n \in \mathbb{Z}_{++}$ is monotone and bounded, by Lemma 33. So the monotone convergence theorem for sequences of real numbers shows that $l := \lim_{n\to\infty} x_n$ exists. But as ϕ_w is continuous, by Lemma 32, the limit l satisfies

$$\phi_w\big(l\big) = \phi_w\big(\lim_{n\to\infty} x_n\big) = \lim_{n\to\infty} \phi_w\big(x_n\big) = \lim_{n\to\infty} x_{n+1} = l.$$

So l is a fixed point of ϕ_w. By Lemma 32, y_w is the unique such fixed point. ■

As a contractive function is not necessarily a contraction mapping, some additional work is required to prove the following result which is essential in the Sturmian case addressed in Subsection A.3.

Lemma 36 *Suppose A2 holds, w is an infinite word and $y_0 \geq a > b \geq y_1$. Then*

$$\lim_{n\to\infty} (\phi_{w_{1:n}}(a) - \phi_{w_{1:n}}(b)) = 0.$$

Proof Let $a_n := \phi_{w_{1:n}}(a)$ and $b_n := \phi_{w_{1:n}}(b)$ for $n \in \mathbb{Z}_{++}$. By Assumption A2, ϕ_{w_n} is a contractive function, so $(a_n - b_n)_{n=1}^{\infty}$ is a decreasing sequence and as $a, b \in [y_1, y_0]$ Lemma 33 shows that this is also a bounded sequence. Therefore the monotone convergence theorem for real-valued sequences shows that

$$\lim_{n\to\infty} (a_n - b_n) \text{ exists.}$$

Now we argue that the limit is zero, by contradiction. Assume that $a_n - b_n \geq \epsilon$ for all $n \in \mathbb{Z}_{++}$, where ϵ is a positive real number. Let the domain $\mathcal{D} \subset \mathbb{R}^2$ be

$$\mathcal{D} := \{(h, l) \ : \ h, l \in [y_1, y_0], \ h \geq l + \epsilon\},$$

let the functions $f_c : \mathcal{D} \to \mathbb{R}$ for letters $c \in \{0, 1\}$ be

$$f_c(h, l) := \frac{\phi_c(h) - \phi_c(l)}{h - l}$$

where $(h, l) \in \mathcal{D}$ and define the number $q \in \mathbb{R}$ by

$$q := \sup_{(h,l)\in\mathcal{D}} \left\{ \max_{c\in\{0,1\}} f_c(h, l) \right\}.$$

Now the functions f_0, f_1 are continuous on their domain $\mathcal{D}$ by Lemma 32. Also, the domain $\mathcal{D}$ is a non-empty, bounded and closed set. So the extreme value theorem for functions of several variables shows that the maximum equals the supremum. Thus

$$q = \max_{(h,l)\in\mathcal{D}} \left\{ \max_{a\in\{0,1\}} f_a(h, l) \right\}.$$

As ϕ_c is contractive for $c \in \{0, 1\}$ it follows that

$$q < 1$$

and as ϕ_c is increasing we have $q > 0$. So the definition of q and hypothesis that $a > b$ give

$$a_n - b_n \leq q^n(a - b) < \epsilon \quad \text{for } n > \log((a - b)/\epsilon)/\log(1/q).$$

Thus there is an $n \in \mathbb{Z}_{++}$ with $a_n - b_n < \epsilon$. This contradicts the assumption that $a_n - b_n \geq \epsilon$ for all $n \in \mathbb{Z}_{++}$. Since $\epsilon > 0$ was arbitrary, we conclude that

$$\lim_{n\to\infty} (a_n - b_n) = 0.$$

This completes the proof. ■

A.2. x-Threshold Words as Christoffel Words

We begin with some definitions that go beyond the main text.

In Definition 5 and the text immediately following it, we introduced the s^--threshold itinerary $\sigma(x|s^-, \phi_0, \phi_1)$ from initial state x for a pair of maps ϕ_0 and ϕ_1. Let us focus on the case where the threshold equals the initial state as in Theorem 12. Now, the first letter of this itinerary is always 1 and as that theorem suggests, the remainder of the itinerary is frequently periodic, motivating the following definition.

Definition 37 *Consider an interval $\mathcal{I} \subseteq \mathbb{R}$, maps $\phi_0 : \mathcal{I} \to \mathcal{I}$ and $\phi_1 : \mathcal{I} \to \mathcal{I}$, and a state $x \in \mathcal{I}$. The **x-threshold word** for ϕ_0 and ϕ_1 is the shortest word $\ell(x, \phi_0, \phi_1)$ such that $\sigma(x|x^-, \phi_0, \phi_1) = 1\ell(x, \phi_0, \phi_1)^\infty$.*

If the maps are evident from the context, we talk of the x-threshold word $\ell(x)$ and if we are also focusing on a particular x, we talk of the x-threshold word ℓ.

The set $\{0,1\}^*$ consists of all finite words on the alphabet $\{0,1\}$, including the empty string ϵ. The *morphism* $\mathscr{M} : \{0,1\}^* \to \{0,1\}^*$ generated by a mapping $s : \{0,1\} \to \{0,1\}^*$, substitutes $s(w_k)$ for each letter w_k of a word w, so that

$$\mathscr{M}(\epsilon) = \epsilon \text{ and } \mathscr{M}(w) = s(w_1)s(w_2)\cdots s(w_{|w|}).$$

We work with the morphisms $\mathscr{L} : \{0,1\}^* \to \{0,1\}^*$ and $\mathscr{R} : \{0,1\}^* \to \{0,1\}^*$ given by

$$\mathscr{L} : \begin{cases} 0 \mapsto 0 \\ 1 \mapsto 01 \end{cases} \qquad \mathscr{R} : \begin{cases} 0 \mapsto 01 \\ 1 \mapsto 1 \end{cases}.$$

Let $\circ$ denote composition of morphisms, for example $\mathscr{R} \circ \mathscr{L}(1) = \mathscr{R}(01) = 011$.

Remark 38 *These morphisms generate the Christoffel tree through pre-composition. Say (u, v) is a Christoffel pair and consider the morphism*

$$\mathscr{M} : \begin{cases} 0 \mapsto u \\ 1 \mapsto v \end{cases}.$$

Then pre-composition maps the node (u, v) of the Christoffel tree to its children:

$$\begin{aligned} (\mathscr{M} \circ \mathscr{L}(0),\ \mathscr{M} \circ \mathscr{L}(1)) &= (\mathscr{M}(0),\ \mathscr{M}(01)) = (u, uv) \\ (\mathscr{M} \circ \mathscr{R}(0),\ \mathscr{M} \circ \mathscr{R}(1)) &= (\mathscr{M}(01),\ \mathscr{M}(1)) = (uv, v). \end{aligned} \tag{40}$$

Now let us give a simple upper and lower bound on the elements of x^--threshold orbits.

Lemma 39 *Suppose A2 holds and $(x_k)_{k=1}^\infty = \text{orbit}(\phi_1(x)|x^-)$. Then*

$$x \in [y_1, y_0] \ \Rightarrow\ \phi_1(x) \le x_k < \phi_0(x) \quad \text{for } k \in \mathbb{Z}_{++}.$$

Proof Say $z \in [y_1, y_0)$. Then Lemma 33 gives

$$y_1 \le \phi_1(z) \le z < \phi_0(z) < y_0.$$

An induction using this fact, immediately shows that for $k \in \mathbb{Z}_{++}$,

$$y_1 \le x_k < y_0.$$

Now we prove the claim by induction with hypothesis

$$H_k : \quad \phi_1(x) \le x_k < \phi_0(x)$$

for $k \in \mathbb{Z}_{++}$. The base case H_1 is true as $x_1 = \phi_1(x)$ by definition of orbit$(\phi_1(x)|x^-)$. For the inductive step, say H_k is true for some $k \in \mathbb{Z}_{++}$. Then there are two cases to consider: $x_k \in [\phi_1(x), x)$ and $x_k \in [x, \phi_0(x))$. If $x_k \in [\phi_1(x), x)$ then $x_{k+1} = \phi_0(x_k)$ and

$$\begin{aligned} \phi_1(x) &\le x_k \\ &< \phi_0(x_k) && \text{by Lemma 33 as } x_k < y_0 \\ &< \phi_0(x) && \text{as } x_k < x \text{ and } \phi_0(\cdot) \text{ is increasing,} \end{aligned}$$

so H_{k+1} is true. If $x_k \in [x, \phi_0(x))$ then $x_{k+1} = \phi_1(x_k)$ and

$$\begin{aligned} \phi_1(x) &\le \phi_1(x_k) && \text{as } x \le x_k \text{ and } \phi_1(\cdot) \text{ is increasing} \\ &\le x_k && \text{by Lemma 33 as } y_1 \le x_k \\ &< \phi_0(x), \end{aligned}$$

so H_{k+1} is true. This completes the proof. ■

Now we show that x-threshold words are Christoffel words in three important special cases and then we find the general conditions on x for which x-threshold words are Christoffel words (Lemma 44).

Lemma 40 *Suppose ℓ is the x-threshold word for ϕ_0, ϕ_1 satisfying A2. Then*

$$\begin{aligned} \ell = 1 &\ \Leftrightarrow\ x \le y_1 \\ \ell = 01 &\ \Leftrightarrow\ x \in [y_{01}, y_{10}] \\ \ell = 0 &\ \Leftrightarrow\ x \ge y_0. \end{aligned}$$

Proof If $\ell = 1$ then the definition of the x-threshold word shows that $\phi_1(x) \ge x$. Therefore Lemma 33 shows that $x \le y_1$.

If $x \le y_1$ then Lemma 33 shows that $x \le \phi_{1^n}(x) \le y_1$ for all $n \in \mathbb{Z}_{++}$. Therefore $\ell = 1$.

If $\ell = 01$ then the next letter of ℓ^∞ after $(01)^n$ is 0 for any $n \in \mathbb{Z}_+$. So $x > \phi_{(01)^n}(\phi_1(x))$. Therefore Lemma 35 gives

$$x \ge \lim_{n\to\infty} \phi_{(01)^n}(\phi_1(x)) = y_{01}.$$

If $\ell = 01$ then $\ell_2 = 1$. So $x \le \phi_{10}(x)$. Therefore Lemma 33 gives

$$x \le y_{10}.$$

If $x \ge y_{01}$ then $x > y_1$ by Lemma 34. So $\phi_1(x) < x$ by Lemma 33. As ϕ_{01} is increasing by Lemma 32, it follows that for any $n \in \mathbb{Z}_+$,

$$\phi_{(01)^n}(\phi_1(x)) < \phi_{(01)^n}(x) \le x$$

where the second inequality follows from Lemma 33 as $x \ge y_{01}$. Therefore, if ℓ^∞ begins with $(01)^n$ then the next letter is 0.

If $x \le y_{10}$ then Lemma 33 shows that $\phi_{(10)^n}(x) \ge x$ for all $n \in \mathbb{Z}_+$. Therefore, if ℓ^∞ begins with $(01)^n 0$ then the next letter is 1.

The proof for the case $\ell = 0$ is symmetric to that for $\ell = 1$. This completes the proof. ■

Lemma 41 *Suppose ϕ_0, ϕ_1 satisfy A2, $x \in \mathcal{I}$ and ℓ is the x-threshold word. Then*

$$\begin{cases} |\ell|_{11} > 0 & \Rightarrow \quad x < y_{01} \\ |\ell|_{00} > 0 & \Rightarrow \quad x > y_{10}. \end{cases}$$

Proof If $|\ell|_{11} > 0$ then $x < y_0$ by Lemma 40. So, either $x \le y_1$, in which case Lemma 34 shows that $x < y_{01}$, or $x \in (y_1, y_0)$. In the latter case, let $(x_k)_{k=1}^\infty = \text{orbit}(\phi_1(x)|x^-)$. As $|\ell|_{11} > 0$ there exists a $k \in \mathbb{Z}_{++}$ with $\phi_1(x_k) \ge x$ by definition of the x-threshold word. Now $x_k < \phi_0(x)$ by Lemma 39, so that $\phi_1(\phi_0(x)) > \phi_1(x_k) \ge x$ as ϕ_1 is increasing. But $\phi_{01}(x) > x$ implies that $x < y_{01}$ by Lemma 33.

The proof for $|\ell|_{00} > 0$ is symmetric. This completes the proof. ■

Lemma 42 *Suppose ϕ_0, ϕ_1 satisfy A2. Then for $x \in [y_{10}, y_0]$, there is a unique $z \in \mathcal{I}$ with*

$$\phi_0(z) = x.$$

Furthermore

$$\ell(x, \phi_0, \phi_1) = \begin{cases} \mathscr{L}(\ell(\phi_0^{(-1)}(x), \phi_0, \phi_{01})) & \text{if } x \in [y_{10}, y_0] \\ \mathscr{R}(\ell(x, \phi_{01}, \phi_1)) & \text{if } x \in [y_1, y_{01}]. \end{cases}$$

Proof In the first claim, existence of z follows from the intermediate value theorem, as ϕ_0 is continuous by Lemma 32, as $y_{01} \in [y_0, y_1] \subseteq \mathcal{I}$ by Lemma 34, as $\phi_0(y_{01}) = y_{10} \le x$ by definition of y_{01}, and as $\phi_0(y_0) = y_0 \ge x$ by definition of y_0. Uniqueness follows as ϕ_0 is increasing.

Now say $x \in [y_{10}, y_0]$ consider the claim involving the morphism $\mathscr{L}$. Let

$$V := \ell(\phi_0^{(-1)}(x), \phi_0, \phi_{01})^\infty \quad \text{and} \quad W := \ell(x, \phi_0, \phi_1)^\infty.$$

We show by induction that the hypothesis

$$H_i \ : \ \mathscr{L}(V_{1:i}) \text{ is a prefix of } W$$

is true for all $i \in \mathbb{Z}_+$, noting that this proves the claim.

In the base case, $\mathscr{L}(\epsilon) = \epsilon$ is a prefix of W, so H_0 is true. For the inductive step, say H_{i-1} is true for some $i \in \mathbb{Z}_{++}$. Let

$$(x_n)_{n=1}^\infty := \text{orbit}(\phi_1(x)|x^-, \phi_0, \phi_1),$$
$$(\tilde{x}_n)_{n=1}^\infty := \text{orbit}(\psi_1(\phi_0^{(-1)}(x))|\phi_0^{(-1)}(x)^-, \psi_0, \psi_1)$$

where $\psi_0 := \phi_0, \psi_1 := \phi_{01}$ and let $k := \left|\mathscr{L}(V_{1:(i-1)})\right| + 1$. Then

$$\tilde{x}_1 = \psi_1(\phi_0^{(-1)}(x)) = \phi_1(x) = x_1$$

and, letting ψ_w denote the composition of ψ_0, ψ_1 corresponding to a word w,

$$\begin{aligned} \tilde{x}_i &= \psi_{V_{1:(i-1)}}(\tilde{x}_1) && \text{by definition of } V \\ &= \phi_{\mathscr{L}(V_{1:(i-1)})}(\tilde{x}_1) && \text{by definition of } \psi_0, \psi_1 \\ &= \phi_{W_{1:(k-1)}}(\tilde{x}_1) && \text{as } H_{i-1} \text{ is true} \\ &= \phi_{W_{1:(k-1)}}(x_1) && \text{as } \tilde{x}_1 = x_1 \\ &= x_k && \text{by definition of } W. \end{aligned}$$

As $x \in [y_{10}, y_0]$, we have $\tilde{x} := \phi_0^{(-1)}(x) \in [y_{01}, y_0]$. Letting $\tilde{y}_0, \tilde{y}_1$ be the fixed points of ψ_0, ψ_1, this reads $\tilde{x} \in [\tilde{y}_1, \tilde{y}_0]$. But ψ_0, ψ_1 satisfy A2, as Lemma 32 shows that these functions are increasing and contractive, and Lemma 34 shows that $\tilde{y}_1 < \tilde{y}_0$. Thus Lemma 39 shows that

$$\tilde{x}_i < \psi_0(\tilde{x}) = \phi_0(\phi_0^{(-1)}(x)) = x.$$

But we already showed that $x_k = \tilde{x}_i$ so this gives $x_k < x$. Therefore $W_k = 0$, by definition of the x-threshold word. If $V_i = 0$ then we can conclude that H_i is true. Otherwise $V_i = 1$ so that $\tilde{x}_i \ge \phi_0^{(-1)}(x)$. But we already showed that $W_k = 0$ and $x_k = \tilde{x}_i$, so $x_{k+1} = \phi_0(x_k) = \phi_0(\tilde{x}_i) \ge x$. Therefore $W_{k+1} = 1$ and we conclude that H_i is true.

The proof for the claim involving $\mathscr{R}$ is similar. This completes the proof. ■

Lemma 43 *Suppose ϕ_0, ϕ_1 satisfy A2, $x \in \mathcal{I}$ and $0v1$ is a finite word. Then*

$$\begin{cases} \ell(\phi_0(x), \phi_0, \phi_1) = \mathscr{L}(0v1) & \Leftrightarrow \quad \ell(x, \phi_0, \phi_{01}) = 0v1 \\ \ell(x, \phi_0, \phi_1) = \mathscr{R}(0v1) & \Leftrightarrow \quad \ell(x, \phi_{01}, \phi_1) = 0v1. \end{cases}$$

Proof Consider the claim involving the morphism $\mathscr{L}$.

If $\ell(\phi_0(x), \phi_0, \phi_1) = \mathscr{L}(0v1)$ then $|\ell(\phi_0(x), \phi_0, \phi_1)|_{00} > 0$, as $|0v1|_{01} > 0$ for any finite word v and $\mathscr{L}(01) = 001$. So Lemma 41 shows that $\phi_0(x) > y_{10}$. Thus $x > y_{01}$. Also,

$|\ell(\phi_0(x),\phi_0,\phi_1)|_1 = |\mathscr{L}(0v1)|_1 > 0$, so Lemma 40 shows that $\phi_0(x) < y_0$. Thus $x < y_0$. Hence Lemma 42 gives

$$\mathscr{L}(0v1) = \mathscr{L}(\ell(x,\phi_0,\phi_{01})).$$

As $\mathscr{L}$ is injective, it follows that

$$0v1 = \ell(x,\phi_0,\phi_{01}).$$

If $\ell(x,\phi_0,\phi_{01}) = 0v1$ then $|\ell(x,\phi_0,\phi_{01})|_0 > 0$ and $|\ell(x,\phi_0,\phi_{01})|_1 > 0$. So Lemma 40 shows that $x \in (y_{01}, y_0)$. Hence Lemma 42 gives

$$\ell(\phi_0(x),\phi_0,\phi_1) = \mathscr{L}(0v1).$$

The argument for the claim involving $\mathscr{R}$ is symmetric. This completes the proof. ■

Lemma 44 *Suppose ϕ_0, ϕ_1 satisfy A2 and $0w1$ is a Christoffel word. Then*

$$x \in [y_{01w}, y_{10w}] \;\Leftrightarrow\; \ell(x,\phi_0,\phi_1) = 0w1.$$

Proof We use induction on the depth of $0w1$ in the Christoffel tree, with hypothesis

$$H_n : \begin{cases} \text{If } 0w1 \text{ is at depth } n \text{ of the tree and } \phi_0,\phi_1 \text{ satisfy A2, then} \\ x \in [y_{01w}, y_{10w}] \;\Leftrightarrow\; \ell(x,\phi_0,\phi_1) = 0w1. \end{cases}$$

Lemma 40 shows that the base case (H_1) with $0w1 = 01$ is true. For the inductive step, let $0w1$ be a word at depth $n+1$ of the tree and assume H_n is true. Then either $0w1 = \mathscr{L}(0v1)$ or $0w1 = \mathscr{R}(0v1)$ for some word $0v1$ which is at depth n of the tree. If $0w1 = \mathscr{L}(0v1)$ then Lemma 43 gives

$$\ell(x,\phi_0,\phi_1) = 0w1 \quad\Leftrightarrow\quad \ell(\phi_0^{(-1)}(x),\phi_0,\phi_{01}) = 0v1.$$

Now ϕ_{01} is increasing and contractive by Lemma 32 and $y_{01} < y_0$ by Lemma 34. Thus ϕ_0, ϕ_{01} satisfy A2. So the assumption that H_n is true shows that

$$\ell(\phi_0^{(-1)}(x),\phi_0,\phi_{01}) = 0v1 \quad\Leftrightarrow\quad \phi_0^{(-1)}(x) \in [y_{\mathscr{L}(01v)}, y_{\mathscr{L}(10v)}].$$

But as $0w1 = \mathscr{L}(0v1) = 0\mathscr{L}(v)01$, we have

$$\begin{aligned} \phi_0(y_{\mathscr{L}(01v)}) &= \phi_0(y_{001\mathscr{L}(v)}) = y_{01\mathscr{L}(v)0} = y_{01w} \\ \phi_0(y_{\mathscr{L}(10v)}) &= \phi_0(y_{010\mathscr{L}(v)}) = y_{10\mathscr{L}(v)0} = y_{10w}. \end{aligned}$$

As ϕ_0 is increasing and continuous, it follows that

$$\phi_0^{(-1)}(x) \in [y_{\mathscr{L}(01v)}, y_{\mathscr{L}(10v)}] \quad\Leftrightarrow\quad x \in [y_{01w}, y_{10w}].$$

Therefore H_{n+1} is true.

If $0w1 = \mathscr{R}(0v1)$ then the proof is similar. This completes the proof. ■

A.3. x-Threshold Words as Sturmian $\mathcal{M}$-Words

It turns out that Lemma 44 characterises x-threshold words for nearly all $x \in (y_1, y_0)$. However, its proof cannot be extended to all values of x as it is based on induction on the depth $n \in \mathbb{Z}_{++}$ in the Christoffel tree, and we need to take the limit as $n \to \infty$ to address the remaining values of x. Those remaining values correspond to Sturmian $\mathcal{M}$-words, as we show in this subsection.

First we show how Sturmian $\mathcal{M}$-words can be written as the limit of a sequence of words.

Lemma 45 *A word w is a Sturmian $\mathcal{M}$-word if and only if it is of the form*

$$w = \lim_{n\to\infty} \mathscr{M}_1 \circ \mathscr{M}_2 \circ \cdots \circ \mathscr{M}_n(01)$$

for some sequence of morphisms $(\mathscr{M}_n)_{n=1}^{\infty}$ with $\mathscr{M}_n \in \{\mathscr{L},\mathscr{R}\}$ which is not eventually constant (i.e. there is no $m \in \mathbb{Z}_{++}$ such that $\mathscr{M}_n = \mathscr{M}_m$ for all $n \in \mathbb{Z}_{++}$ with $n > m$).

The above result is well known, for instance, noting the isomorphism of the Christoffel tree and the Stern-Brocot tree, see Chapter 4 of Graham et al. (1994).

Remark 46 *The requirement that the sequence of morphisms generating Sturmian $\mathcal{M}$-words is not eventually constant rules out words such as*

$$\lim_{n\to\infty} \mathscr{L}^n(01) = 0^{\infty}.$$

Indeed, this corresponds to the Christoffel word 0. It also rules out words such as

$$\lim_{n\to\infty} \mathscr{R}^n(01) = 01^{\infty}.$$

Indeed, this is not an $\mathcal{M}$-word because there is no $\alpha \in [0,1]$ with

$$0 = \lfloor \alpha n \rfloor - \lfloor \alpha(n-1) \rfloor \text{ for } n = 1 \text{ and } 1 = \lfloor \alpha n \rfloor - \lfloor \alpha(n-1) \rfloor \text{ for } n = 2, 3, \ldots.$$

Now we show that every x-threshold word corresponds to an $\mathcal{M}$-word.

Lemma 47 *Suppose ϕ_0, ϕ_1 satisfy A2 and $x \in \mathcal{I}$. Then $\ell(x,\phi_0,\phi_1)$ is an $\mathcal{M}$-word.*

Proof Lemma 40 shows that $\ell(x,\phi_0,\phi_1)$ is an $\mathcal{M}$-word for $x \in \mathcal{I}\backslash(y_1, y_0)$. So, for the rest of this proof we assume that $x \in (y_1, y_0)$. We shall now define procedure which generates a sequence of morphisms $\mathscr{M}_k$, thresholds $x_k \in \mathcal{I}$, mappings $\phi_{k,a} : \mathcal{I} \to \mathcal{I}$ for $a \in \{0,1\}$ and we denote the compositions of those mappings by

$$\phi_{k,w} = \phi_{k,w_{|w|}} \circ \cdots \phi_{k,w_2} \circ \phi_{k,w_1}$$

for any finite word w. The procedure is as follows:

1. **let** $k \leftarrow 1$
2. **let** $(x_k, \phi_{k,0}, \phi_{k,1}) \leftarrow (x,\phi_0,\phi_1)$
3. **let** $y_{k,01}, y_{k,10}$ be the fixed points of $\phi_{k,01}, \phi_{k,10}$

4. **while** $x_k \notin [y_{k,01}, y_{k,10}]$
5. **if** $x_k < y_{k,01}$
6. **set** $(x_{k+1}, \phi_{k+1,0}, \phi_{k+1,1}, \mathscr{M}_k) \leftarrow (x_k, \phi_{k,01}, \phi_{k,1}, \mathscr{R})$
7. **else**
8. **set** $(x_{k+1}, \phi_{k+1,0}, \phi_{k+1,1}, \mathscr{M}_k) \leftarrow (\phi_{k,0}^{(-1)}(x), \phi_{k,0}, \phi_{k,01}, \mathscr{L})$
9. **end**
10. **let** $k \leftarrow k+1$
11. **let** $y_{k,01}, y_{k,10}$ be the fixed points of $\phi_{k,01}, \phi_{k,10}$
12. **end**

The sequence of morphisms $\mathscr{M}_1, \mathscr{M}_2, \ldots$ generated by this procedure is either empty, of finite length $n \in \mathbb{Z}_{++}$ or of infinite length. We consider each of these cases in turn.

If the sequence is empty, then $x \in [y_{1,01}, y_{1,10}] = [y_{01}, y_{10}]$. So Lemma 40 shows that $\ell(x, \phi_0, \phi_1) = 01$, which is an $\mathcal{M}$-word.

If the sequence has finite length, let n be that length. As the morphisms $\mathscr{L}$ and $\mathscr{R}$ generate the Christoffel tree by pre-composition, as remarked at (40), the word

$$w := \mathscr{M}_1 \circ \cdots \circ \mathscr{M}_n(01)$$

is a Christoffel word. Also, when the procedure sets $\mathscr{M}_k = \mathscr{R}$ for some $k \in \mathbb{Z}_{++}$, we have $x_k < y_{k,01} < y_{k,10}$ so Lemma 42 shows that

$$\ell(x_k, \phi_{k,0}, \phi_{k,1}) = \mathscr{R}(\ell(x_{k+1}, \phi_{k+1,0}, \phi_{k+1,1})).$$

Similarly, when the procedure sets $\mathscr{M}_k = \mathscr{L}$ we have

$$\ell(x_k, \phi_{k,0}, \phi_{k,1}) = \mathscr{L}(\ell(x_{k+1}, \phi_{k+1,0}, \phi_{k+1,1})).$$

Therefore the x-threshold word is $\ell(x, \phi_0, \phi_1) = w$ which is an $\mathcal{M}$-word.

Finally, if the sequence does not terminate, then Lemma 45 shows that the word

$$w := \lim_{n\to\infty} \mathscr{M}_1 \circ \cdots \circ \mathscr{M}_n(01)$$

is a Sturmian $\mathcal{M}$-word *provided* there is no $m \in \mathbb{Z}_{++}$ such that $\mathscr{M}_n = \mathscr{M}_m$ for all $n \in \mathbb{Z}_{++}$ with $n \geq m$. Also, the argument given for finite sequences above shows that $w = \ell(x, \phi_0, \phi_1)$. If there were such an m and $\mathscr{M}_m = \mathscr{R}$, then

$$\begin{aligned} w &= \mathscr{M}_1 \circ \cdots \circ \mathscr{M}_{m-1} \circ \lim_{k\to\infty} \mathscr{R}^k(01) \\ &= \mathscr{M}_1 \circ \cdots \circ \mathscr{M}_{m-1}(01^\infty). \end{aligned}$$

Thus the word at stage m is $\ell(x_m, \phi_{m,0}, \phi_{m,1}) = 01^\infty$. But this is impossible, as the fact that the first letter is 0 requires $\phi_{m,1}(x_m) < x_m$, so that $x_m > y_{m,1}$, whereas the fact that remaining letters are 1 requires $\phi_{m,101^n}(x_m) \geq x_m$ for all $n \in \mathbb{Z}_+$, so that $y_{m,1} \geq x_m$, which is a contradiction. If there were such an m and $\mathscr{M}_m = \mathscr{L}$, then

$$w = \mathscr{M}_1 \circ \cdots \circ \mathscr{M}_{m-1} \circ \lim_{k\to\infty} \mathscr{L}^k(01)$$

$$= \mathscr{M}_1 \circ \cdots \circ \mathscr{M}_{m-1}(0^\infty)$$

Therefore at stage m the word $\ell(x_m, \phi_{m,0}, \phi_{m,1}) = 0$, but this is impossible as $x_m < y_{m,0}$. In conclusion, $\ell(x, \phi_0, \phi_1)$ is an $\mathcal{M}$-word.

This completes the proof. ■

Lemma 48 *Suppose $0 \leq \alpha < \beta \leq 1$ and let a, b be the $\mathcal{M}$-words of rate α, β respectively. Then $a^\infty \prec b^\infty$.*

Proof Consider the first $n \in \mathbb{Z}_{++}$ with $(a^\infty)_n \neq (b^\infty)_n$. Then

$$\lfloor \alpha(n-1) \rfloor = \left|(a^\infty)_{1:(n-1)}\right|_1 = \left|(b^\infty)_{1:(n-1)}\right|_1 = \lfloor \beta(n-1) \rfloor,$$

by definition of $\mathcal{M}$-words, so that

$$(a^\infty)_n = \lfloor \alpha n \rfloor - \lfloor \alpha(n-1) \rfloor = \lfloor \alpha n \rfloor - \lfloor \beta(n-1) \rfloor < \lfloor \beta n \rfloor - \lfloor \beta(n-1) \rfloor = (b^\infty)_n$$

where the inequality holds as $(a^\infty)_n \neq (b^\infty)_n$, as $\alpha < \beta$ and as the floor function is non-decreasing. Therefore $a^\infty \prec b^\infty$. ■

Consider a tree and a node x of the tree. Recall that the *subtree rooted at* x is the tree of all descendants of x that has node x as a root. If the tree is a binary tree, the *left subtree of* x is the subtree rooted at the left child of x, and the *right subtree of* x is the subtree rooted at the right child of x. Thus the subtree rooted at x contains node x, but the left and right subtrees of x do not contain node x.

Lemma 49 *Suppose (u, v) is a Christoffel pair. Considering uv as a node of the Christoffel tree, let l and r be Christoffel words in the left and right subtrees of uv. Then*

$$\text{rate}(u) < \text{rate}(l) < \text{rate}(uv) < \text{rate}(r) < \text{rate}(v).$$

Proof First we use induction to show that for any Christoffel pair (u, v), we have

$$\text{rate}(u) < \text{rate}(v). \tag{41}$$

In the base case $(u, v) = (0, 1)$ and (41) is true. For the inductive step, say (u, v) is the left child of a Christoffel pair (a, b) with $\text{rate}(a) < \text{rate}(b)$. Then $(u, v) = (a, ab)$ so that

$$\text{rate}(u) = \frac{|a|_1}{|a|} < \frac{|a|_1 + |b|_1}{|a| + |b|} = \text{rate}(v)$$

by the mediant inequality. The proof for right children is similar. Therefore (41) is true.

As l is in the left subtree of uv, it follows from the construction of the Christoffel tree that l consists of m copies of u and n copies of uv concatenated in some order, for some $m, n \in \mathbb{Z}_{++}$. Thus the mediant inequality and (41) give

$$\text{rate}(l) = \text{rate}(u^m (uv)^n) = \frac{m|u|_1 + n|uv|_1}{m|u| + n|uv|} \in \left(\frac{|u|_1}{|u|}, \frac{|uv|_1}{|uv|} \right)$$

Therefore $\text{rate}(u) < \text{rate}(l) < \text{rate}(uv)$, as claimed.

The proof for a node r in the right subtree of uv is similar. This completes the proof. ■

Lemma 50 *Suppose* $(0a1, 0b1)$ *is a Christoffel pair. Then* $a10b = b01a$.

Proof As $a, b, a10b$ are palindromes, we have $a10b = (a10b)^R = b^R01a^R = b01a$. ■

Lemma 51 *Suppose the word* $0c1$ *is in the subtree of the Christoffel tree rooted at* $0p1$. *Then* p *is both a prefix and suffix of* c.

Proof There are four cases to consider:

1. The word $0c1$ has no parent, in which case $c = p = \epsilon$ and the claim holds.
2. The word c is of one of the forms 0^m or 0^m10^m for some $m \in \mathbb{Z}_{++}$. In that case either $p = c$ or $p = 0^l$ for some $l \in \mathbb{Z}_+$ with $l < m$ and the claim holds.
3. The word c is of one of the forms 1^m or 1^m01^m for some $m \in \mathbb{Z}_{++}$. This is similar to the previous case.
4. The Christoffel pair of the parent of $0c1$ is of the form $(0a1, 0b1)$.

In the last case, we use induction on the length of the path $n \in \mathbb{Z}_+$ through the Christoffel tree from $0p1$ to $0c1$. In the base case, $n = 0$, we have $c = p$ and the claim is true. For the inductive step, say $0c1$ is a child of the node with Christoffel pair $(0a1, 0b1)$, that $0a10b1$ is n steps along the path from $0p1$, and that p is both a prefix and suffix of $a10b$, so that $a10b = pq = rp$ for some words q, r. If $0c1$ is a left-child, then $c = a10a10b = a10rp$ and Lemma 50 gives

$$c = a10a10b = a10b01a = pq01a$$

so p is a prefix and suffix of c. Similarly, if $0c1$ is a right child, then $c = a10b10b$ and

$$pq10b = a10b10b = b01a10b = b01rp.$$

This completes the proof. ■

Lemma 52 *Suppose* ϕ_0, ϕ_1 *satisfy A2. Then*

$$\ell(x, \phi_0, \phi_1)^\infty \text{ is a lexicographically non-increasing function of } x \in \mathcal{I}.$$

Proof If $a, b \in \mathcal{I}$ and $\ell(a, \phi_0, \phi_1)^\infty \succ \ell(b, \phi_0, \phi_1)^\infty$, then there is a finite word u such that

$$\ell(a, \phi_0, \phi_1)^\infty = u1v, \qquad \ell(b, \phi_0, \phi_1)^\infty = u0w,$$

for some words v, w. So the definition of threshold orbits and Lemma 33 give

$$y_{1u} \geq \phi_{1u}(a) \geq a, \qquad y_{1u} < \phi_{1u}(b) < b.$$

Therefore $a < b$. This completes the proof. ■

In the main text, we defined the fixed point y_s of a Sturmian $\mathcal{M}$-word as the limit of a sequence of fixed points $y_{01w^{(n)}}$ or $y_{10w^{(n)}}$ where the words $(0w^{(n)}1)_{n=1}^\infty$ correspond to a particular path in the Christoffel tree. We now define a subsequence associated with each of these sequences of words and fixed points. Consider a Sturmian $\mathcal{M}$-word

$$0s = \lim_{n\to\infty} \mathscr{L}^{a_1} \circ \mathscr{R}^{b_1} \circ \cdots \circ \mathscr{L}^{a_n} \circ \mathscr{R}^{b_n}(01)$$

where $a_1 \in \mathbb{Z}_+$ and $a_{n+1}, b_n \in \mathbb{Z}_{++}$ for $n \in \mathbb{Z}_{++}$. We define the sequences $(u^{(n)})_{n=1}^\infty$ and $(l^{(n)})_{n=1}^\infty$ of the central portions of the Christoffel words as

$$0u^{(n)}1 := \mathscr{L}^{a_1} \circ \mathscr{R}^{b_1} \circ \cdots \circ \mathscr{L}^{a_n}(01), \quad 0l^{(n)}1 := \mathscr{L}^{a_1} \circ \mathscr{R}^{b_1} \circ \cdots \circ \mathscr{L}^{a_n} \circ \mathscr{R}^{b_n}(01). \quad (42)$$

These words form a subsequence of $(0w^{(n)}1)_{n=1}^\infty$, so if $\lim_{n\to\infty} y_{01w^{(n)}}$ exists, then so does $\lim_{n\to\infty} y_{01l^{(n)}}$ and these limits are equal.

Lemma 53 *Suppose A2 holds,* $n \in \mathbb{Z}_{++}$ *and that* $0u^{(n)}1$ *and* $0l^{(n)}1$ *are as in (42). Then*

$$y_{01l^{(n)}} < y_{01l^{(n+1)}} < y_{10u^{(n+1)}} < y_{10u^{(n)}}.$$

Proof By definition, $0l^{(n+1)}1$ is in the left subtree of $0l^{(n)}1$, so Lemma 49 gives

$$\text{rate}(0l^{(n+1)}1) < \text{rate}(0l^{(n)}1).$$

Hence Lemma 44 and Lemma 48 give

$$\ell(y_{01l^{(n+1)}})^\infty = (0l^{(n+1)}1)^\infty \prec (0l^{(n)}1)^\infty = \ell(y_{10l^{(n)}})^\infty.$$

Therefore Lemma 52 shows that

$$y_{01l^{(n)}} < y_{01l^{(n+1)}}.$$

But $0l^{(n+1)}1$ is in the right subtree of $0u^{(n+1)}1$. So the same argument gives

$$y_{01l^{(n+1)}} < y_{10u^{(n+1)}}.$$

Similarly, $0u^{(n+1)}1$ is in the right subtree of $0u^{(n)}1$, so

$$y_{10u^{(n+1)}} < y_{10u^{(n)}}.$$

This completes the proof. ■

Note that the argument in the above proof also shows that any word $0w^{(m)}1$ lying strictly between $0u^{(n)}1$ and $0l^{(n)}1$ on the path through the Christoffel tree from $0u^{(n)}1$ to $0l^{(n)}1$ has

$$y_{10l^{(n)}} < y_{01w^{(m)}} < y_{10w^{(m)}} < y_{01u^{(n)}}.$$

Similarly, any word $0w^{(m)}1$ lying strictly between $0l^{(n)}1$ and $0u^{(n+1)}1$ on the path from $0l^{(n)}1$ to $0u^{(n+1)}1$ has

$$y_{10l^{(n)}} < y_{01w^{(m)}} < y_{10w^{(m)}} < y_{01u^{(n+1)}}.$$

Thus if the subsequences $(y_{01l^{(n)}})_{n=1}^{\infty}$ and $(y_{10u^{(n)}})_{n=1}^{\infty}$ have limits, then so do the full sequences $(y_{01w^{(m)}})_{n=1}^{\infty}$ and $(y_{10w^{(m)}})_{n=1}^{\infty}$.

Lemma 54 *Suppose $(0w^{(n)}1)_{n=1}^{\infty}$ is the sequence of words traversed along an infinite path down the Christoffel tree. Let $(n_i)_{i=1}^{\infty}$ be an increasing sequence on $\mathbb{Z}_{++}$. Then there exists an infinite word s and an increasing sequence $(k_i)_{i=1}^{\infty}$ on $\mathbb{Z}_{++}$ such that $s_{1:k_i}$ is a suffix of both $w^{(n_i)}$ and $w^{(n_{i+1})}$, for all $i \in \mathbb{Z}_{++}$.*

Proof We show that $s_{1:k_i} := w^{(n_i)}$ for $i \in \mathbb{Z}_{++}$, is well-defined and satisfies this claim. As $w^{(n_i)}$ is a prefix of all of its descendants by Lemma 51, it follows that $s_{1:k_i}$ is a prefix of $s_{1:k_{i+1}}$. Also, as $w^{(n_i)}$ is a suffix of all of its descendants, it follows that $s_{1:k_i}$ is a suffix of both $w^{(n_i)}$ and $w^{(n_{i+1})}$. ■

Lemma 55 *Suppose ϕ_0, ϕ_1 satisfy A2 and $0s$ is a Sturmian $\mathcal{M}$-word. Consider the sequence of Christoffel words $(0w^{(n)}1 : n \in \mathbb{Z}_{++})$ traversed on the infinite path through the Christoffel tree towards $0s$ (as defined in the main text just before Theorem 12). Then the fixed points*

$$y_{01s} := \lim_{n\to\infty} y_{01w^{(n)}} \qquad \text{and} \qquad y_{10s} := \lim_{n\to\infty} y_{10w^{(n)}}$$

exist and are equal.

Proof Let $a_n, b_n, 0u^{(n)}1, 0l^{(n)}1$ for $n \in \mathbb{Z}_{++}$ be as in (42).

Existence of the fixed points follows from the monotone convergence theorem for real-valued sequences. Indeed, Lemma 53 shows that $(y_{01l^{(n)}})_{n=1}^{\infty}$ is an increasing sequence, that $(y_{10u^{(n)}})_{n=1}^{\infty}$ is a decreasing sequence, and that these sequences are bounded.

By Lemma 54 we have $u^{(n)} = c^{(n)}w_{1:k_n}$ and $l^{(n)} = d^{(n)}w_{1:k_n}$ for all $n \in \mathbb{Z}_{++}$ for some sequences of finite words $(c^{(n)})_{n=1}^{\infty}$ and $(d^{(n)})_{n=1}^{\infty}$, for some infinite word w and for some increasing sequence $(k_n)_{n=1}^{\infty}$ on $\mathbb{Z}_{++}$.

As $\phi_a(x)$ is an increasing function of $x \in \mathcal{I}$ for any finite word a, by Lemma 32,

$$\begin{aligned} y_{10u^{(n)}} &= \phi_{10u^{(n)}}(y_{10u^{(n)}}) \\ &< \phi_{10u^{(n)}}(y_0) \\ &= \phi_{10c^{(n)}w_{1:k_n}}(y_0) \\ &= \phi_{w_{1:k_n}}(\phi_{10c^{(n)}}(y_0)) \end{aligned}$$

$$< \phi_{w_{1:k_n}}(y_0).$$

Similarly, we have

$$y_{01l^{(n)}} > \phi_{w_{1:k_n}}(y_1).$$

Thus

$$\lim_{n\to\infty}(y_{10u^{(n)}} - y_{01l^{(n)}}) \le \lim_{n\to\infty}(\phi_{w_{1:k_n}}(y_0) - \phi_{w_{1:k_n}}(y_1)) = 0$$

where the last step is Lemma 36. But $y_{10u^{(n)}} > y_{01l^{(n)}}$ for $n \in \mathbb{Z}_{++}$ by Lemma 53. Therefore

$$y_{10s} = \lim_{n\to\infty} y_{10u^{(n)}} = \lim_{n\to\infty} y_{01l^{(n)}} = y_{01s}.$$

This completes the proof. ■

In view of the above Lemma, from now on we shall write $y_s = y_{10s} = y_{01s}$.

Lemma 56 *Suppose ϕ_0, ϕ_1 satisfy A2 and $0s$ is a Sturmian $\mathcal{M}$-word. Then*

$$\ell(x, \phi_0, \phi_1) = 0s \;\Leftrightarrow\; x = y_s.$$

Proof Let $(u^{(n)})_{n=1}^{\infty}$ and $(l^{(n)})_{n=1}^{\infty}$ be the sequences (42) appearing in the definition of the fixed point of $0s$. For fixed ϕ_0, ϕ_1 let us write $\ell(x)$ in place of $\ell(x, \phi_0, \phi_1)$.

Say $x = y_s$. Recall that $x = y_{10s} = y_{01s}$ by Lemma 55. As $y_{10u^{(n)}} > y_{10s}$ for all $n \in \mathbb{Z}_{++}$ by Lemma 53, and $\ell(z)^{\infty}$ is a lexicographically non-increasing function of z, by Lemma 52, it follows that

$$\ell(x)^{\infty} \succ \ell(y_{10u^{(n)}})^{\infty} = (0u^{(n)}1)^{\infty}$$

where the equality follows from Lemma 44. A similar argument gives $\ell(x)^{\infty} \prec (0l^{(n)}1)^{\infty}$ for $n \in \mathbb{Z}_{++}$. Therefore

$$0s = \lim_{n\to\infty}(0u^{(n)}1)^{\infty} \preceq \ell(x)^{\infty} \preceq \lim_{n\to\infty}(0l^{(n)}1)^{\infty} = 0s.$$

Now say $\ell(x) = 0s$ for some $x \in \mathcal{I}$. Then $y_{01l^{(n)}} < x < y_{10u^{(n)}}$ as $\ell(z)^{\infty}$ is a lexicographically non-increasing function of z and $(0u^{(n)}1)^{\infty} \prec 0s \prec (0l^{(n)}1)^{\infty}$ for all $n \in \mathbb{Z}_{++}$. Therefore

$$y_s = \lim_{n\to\infty} y_{01l^{(n)}} \le x \le \lim_{n\to\infty} y_{10u^{(n)}} = y_s.$$

This completes the proof. ■

A.4. Proof of Theorem 12

Proof The existence of fixed points y_{01p}, y_{10p} follows from Lemma 32 and the existence of y_s follows from Lemma 55.

The fact that $\sigma(z|z^-) = 1\ell(z, \phi_0, \phi_1)^\infty$ is a lexicographically non-decreasing function of $z \in \mathcal{I}$ follows from Lemma 52.

Lemma 40 addresses the value of $\sigma(z|z^-)$ for $z \le y_1$ and $z \ge y_0$, Lemma 44 addresses the case $z \in [y_{01p}, y_{10p}]$ and Lemma 56 address the case $z = y_s$. Thus the image of $\ell(z, \phi_0, \phi_1)$ as z ranges over $\mathcal{I}$ contains all $\mathcal{M}$-words. Using Lemma 47, it follows that this image is exactly the set of $\mathcal{M}$-words.

This completes the proof. ■

A.5. Proof of Corollary 13

Proof First we give a simple result about the relation between $\mathcal{M}$-words of rates r and $1-r$. Then we demonstrate the result for itineraries of the form $\sigma(\phi_0(x)|x)$ and finally the result for itineraries of the form $\sigma(\phi_1(x)|x)$.

Relation between $\mathcal{M}$-words. Let bitflip$(\cdot)$ denote the bitwise NOT operation, that maps any finite or infinite word w to the word of the same length whose n^{th} letter is $1 - w_n$. We now show that

$$\text{if } 0s \text{ is a Sturmian } \mathcal{M}\text{-word then } 0\,\text{bitflip}(s) \text{ is also a Sturmian } \mathcal{M}\text{-word.}$$

Let $r := \text{rate}(0s)$. By definition, the word w of rate $1-r$ has

$$w_n = \lfloor (1-r)n \rfloor - \lfloor (1-r)(n-1) \rfloor = 1 - \lceil rn \rceil + \lceil r(n-1) \rceil$$

for any positive integer n. As $0s$ is Sturmian, its rate r is irrational, so rm is not an integer for any positive integer m and it follows that $\lceil rm \rceil = \lfloor rm \rfloor$. Thus

$$w_n = 1 - (\lfloor rn \rfloor - \lfloor r(n-1) \rfloor) = \text{bitflip}(0s)_n$$

as claimed. A similar argument shows that

$$\text{if } 0p1 \text{ is a Christoffel word then } 0\,\text{bitflip}(p)1 \text{ is also a Christoffel word.}$$

Result for $\sigma(\phi_0(x)|x)$. Let $\tilde{\phi}_0(x) := -\phi_1(-x)$, $\tilde{\phi}_1(x) := -\phi_0(-x)$ and $\tilde{\mathcal{I}} := \{-x : x \in \mathcal{I}\}$. For a word w, let $\tilde{y}_w$ be the fixed point $y_w = \tilde{\phi}_w(y_w)$ of the composition $\tilde{\phi}_w$ of $\tilde{\phi}_0, \tilde{\phi}_1$ corresponding to w. Since A2 holds for $\mathcal{I}, \phi_0$ and ϕ_1, it is easy to see that it also holds for $\tilde{\mathcal{I}}, \tilde{\phi}_0$ and $\tilde{\phi}_1$. Let $\tilde{\sigma}(x|s^-)$ denote the s^--threshold itinerary from $x \in \tilde{\mathcal{I}}$ for the maps $\tilde{\phi}_0, \tilde{\phi}_1$. From these definitions, for any $x \in \mathcal{I}$ we have

$$\sigma(\phi_0(x)|x) = \text{bitflip}(\tilde{\sigma}(\tilde{\phi}_1(-x)|(-x)^-)).$$

But if $x \le y_1$ then $-x \ge \tilde{y}_0$, so Theorem 12 gives $\tilde{\sigma}(-x|(-x)^-) = 10^\infty$ and thus $\sigma(\phi_0(x)|x) = 1^\infty$. The argument that $\sigma(\phi_0(x)|x) = 0^\infty$ for $x \ge y_0$ is similar. Also if $x \in [y_{01p}, y_{10p}]$ then $-x \in [\tilde{y}_{01\,\text{bitflip}(p)}, \tilde{y}_{10\,\text{bitflip}(p)}]$, and the relation between $\mathcal{M}$-words above shows that

$0\,\text{bitflip}(p)1$ is a Christoffel word, so Theorem 12 gives $\tilde{\sigma}(-x|(-x)^-) = (10\,\text{bitflip}(p))^\infty$ and thus $\sigma(\phi_0(x)|x) = (1p0)^\infty$. Finally if $x = y_s$ then $-x = \tilde{y}_{\text{bitflip}(s)}$, and the relation between $\mathcal{M}$-words above shows that $0\,\text{bitflip}(s)$ is a Sturmian $\mathcal{M}$-word, so Theorem 12 gives $\tilde{\sigma}(-x|(-x)^-) = 10\,\text{bitflip}(s)$ and thus $\sigma(\phi_0(x)|x) = 1s$.

Result for $\sigma(\phi_1(x)|x)$. Let $\text{orbit}(x|s^-)$ denote the sequence of states in the s^--threshold orbit from initial state x under maps ϕ_0 and ϕ_1. The core of our argument is that if $\text{orbit}(x|s^-)$ does not contain s then $\sigma(x|s) = \sigma(x|s^-)$. Indeed if s is not in $\text{orbit}(x|s^-) =: (x_1, x_2, \dots)$, the decisions $\mathbf{1}_{x_k > s}$ and $\mathbf{1}_{x_k \ge s}$ are the same for each $k = 1, 2, \dots$.

Say $x \in [y_{01p}, y_{10p})$. First we show that assuming that x is is in $\text{orbit}(\phi_1(x)|x^-)$ leads to a contradiction. If indeed x were in that orbit, then either $x = \phi_{(10p)^N}(x)$ for some $N \in \mathbb{Z}_{++}$ or $x = \phi_{(10p)^N(10p)_{1:m}}(x)$ for some $N \in \mathbb{Z}_+$ and $1 \le m < |10p|$. The first case gives $x = y_{10p}$ which contradicts $x < y_{10p}$. The second case gives $\sigma(x|x^-) = (10p)^N(10p)_{1:m}\sigma(x|x^-) = ((10p)^N(10p)_{1:m})^\infty$ and as m is not a multiple of $|10p|$, this contradicts Theorem 12 which shows that $\sigma(x|x^-) = (10p)^\infty$. It follows that

$$\sigma(\phi_1(x)|x) = \sigma(\phi_1(x)|x^-).$$

Since $\sigma(x|x^-) = (10p)^\infty$ by Theorem 12, we conclude that $\sigma(\phi_1(x)|x) = (0p1)^\infty$.

If $x = y_{10p}$ then $\sigma(x|x^-) = (10p)^\infty$ by Theorem 12 and the first time $\text{orbit}(x|x^-)$ returns to x is after the letters $10p$. Thus $\sigma(\phi_1(x)|x)$ and $\sigma(\phi_1(x)|x^-)$ agree up to that point, so $\sigma(\phi_1(x)|x) = 0p0\sigma(\phi_0(x)|x) = 0p(0p1)^\infty$ using the result about $\sigma(\phi_0(x)|x)$ proved earlier.

Say $x = y_s$. Theorem 12 gives $\sigma(x|x^-) = 10s$. Say $\phi_{(10s)_{1:k}}(x) = x$ for some $k \in \mathbb{Z}_{++}$. Then $\sigma(x|x^-) = (10s)_{1:k}\sigma(x|x^-) = ((10s)_{1:k})^\infty$ which contradicts the fact that Sturmian $\mathcal{M}$-words are not periodic. Therefore $\text{orbit}(x|x^-)$ does not return to x and it follows that $\sigma(\phi_1(x)|x) = \sigma(\phi_1(x)|x^-) = 0s$.

This completes the proof. ■

A.6. Proof of Theorem 14

Based on the work of Kozyakin (2003), we begin by showing that maps-with-gaps formed from functions satisfying Assumption A2 are *locally-growing relaxation functions* (Lemmas 58 and 59) and that the itineraries of such functions are 1-*balanced words* (Lemmas 61 and 62). As 1-balanced words correspond to factors of lower mechanical words (Lemma 63), this enables us to describe the itineraries of maps-with-gaps, for arbitrary initial states and thresholds (Lemma 64). We couple this description with an easy result about lexicographic ordering (Lemma 65) and with a result about the number of factors of mechanical words (Lemma 66) to bound the number of discontinuities of the itinerary of a map-with-a-gap as a function of its threshold (Lemma 67). Finally, we prove Theorem 14.

Definition 57 *Let $\mathcal{I}$ be an interval of $\mathbb{R}$. A function $f : \mathcal{I} \to \mathcal{I}$ is a* ***locally-growing relaxation function*** *with threshold $z \in \mathcal{I}$ if*

1. $f(z) < z < f(z^-) < \infty$
2. $f(f(z^-)) \le f(f(z))$

3. $f(x)$ is increasing for $x \in [f(z), z)$

4. $f(x)$ is increasing for $x \in [z, f(z^-))$.

In Kozyakin (2003), this terminology was used for a smaller class of functions f. In particular, the domain and range were restricted to the interval $[0, 1)$ and there was a requirement that f is continuous on each of the intervals $[f(z), z)$ and $[f(z^-), f(z))$.

The following two Lemmas show that maps-with-gaps whose parts satisfy Assumption A2 lead to locally-growing relaxation functions.

Lemma 58 *Suppose ϕ_0, ϕ_1 satisfy A2 and $x \in [y_1, y_0]$. Then $\phi_{01}(x) < \phi_{10}(x)$.*

Proof Suppose that $x \in [y_1, y_0)$. Then using Lemma 33 gives

$$\begin{aligned} \phi_1(\phi_0(x)) - \phi_1(x) &< \phi_0(x) - x && \text{as } \phi_1 \text{ is contractive and } x < y_0 \\ \phi_0(x) - \phi_0(\phi_1(x)) &\le x - \phi_1(x) && \text{as } \phi_0 \text{ is contractive and } x \ge y_1 \\ \phi_{01}(x) &< \phi_{10}(x) && \text{by adding these inequalities.} \end{aligned}$$

A symmetric argument holds if $x \in (y_1, y_0]$. This completes the proof. ■

Lemma 59 *Suppose ϕ_0, ϕ_1 satisfy A2, that $z \in (y_1, y_0)$ and let $f(x) := \phi_{\mathbf{1}_{x \ge z}}(x)$ for $x \in \mathcal{I}$. Then f is a locally-growing relaxation function with threshold z.*

Proof By definition $f(z^-) = \phi_0(z^-) = \phi_0(z)$, as ϕ_0 is continuous by Lemma 32. Also $f(z) = \phi_1(z)$, $f(f(z^-)) = \phi_{01}(z)$ and $f(f(z)) = \phi_{10}(z)$. So the conditions defining a locally-growing relaxation function read as follows:

1. $\phi_1(z) < z < \phi_0(z) < \infty$, which is true for $z \in (y_1, y_0)$ by Lemma 33
2. $\phi_{01}(z) < \phi_{10}(z)$, which is the result of Lemma 58
3. $\phi_0(x)$ is increasing for $x \in [\phi_1(z), z)$, which holds by A2
4. $\phi_1(x)$ is increasing for $x \in [z, \phi_0(z))$, which holds by A2.

This completes the proof. ■

The following definition is due to Morse and Hedlund (1940).

Definition 60 *An word w is 1-**balanced** if*

$$\big|\, |u|_1 - |v|_1 \big| \le 1$$

for all factors u, v of w with $|u| = |v|$.

The next two Lemmas use an argument from Kozyakin (2003) to show that the itineraries of a locally-growing relaxation function are 1-balanced. We denote the fractional part of a real number x by $\{x\} := x - \lfloor x \rfloor$.

Lemma 61 *Suppose f is a locally-growing relaxation function with threshold z. Let*

$$F(x) := \frac{f(t\{x\} - t\mathbf{1}_{\{x\} \in [\alpha, 1)} + z) - z}{t} + \lfloor x \rfloor + 1 \qquad \text{for } x \in \mathbb{R}$$

where $t := f(z^-) - f(z)$ and $\alpha := (f(z^-) - z)/t$. Then

F1. $F(0) \in [0, 1)$

F2. $F(x + 1) = F(x) + 1$ for all $x \in \mathbb{R}$

F3. F is increasing

F4. The itineraries of f and F agree in the sense that

$$f^{(n-1)}(x) \ge z \;\Leftrightarrow\; \left\{ F^{(n-1)}\left(\frac{x - z}{t}\right) \right\} \in [0, F(0))$$

for all $n \in \mathbb{Z}_{++}$ and all $x \in [f(z), f(z^-))$.

Proof First, note that the function F is well-defined as $f(z) \in \mathbb{R}$ and $f(z) < z < f(z^-) < \infty$ so that $t \in \mathbb{R}_{++}$ and

$$\alpha = \frac{f(z^-) - z}{f(z^-) - f(z)} \in (0, 1). \tag{43}$$

It is easy to see that F1 and F2 hold. Indeed

$$F(0) = \frac{f(z) - z}{t} + 1 = \frac{f(z) - z + f(z^-) - f(z)}{f(z^-) - f(z)} = \alpha \in (0, 1)$$

and as the ratio in the definition of $F(x)$ only depends on $\{x\}$, we have

$$F(x + 1) - F(x) = \lfloor x + 1 \rfloor - \lfloor x \rfloor = 1 \qquad \text{for } x \in \mathbb{R}. \tag{44}$$

Now we show that F3 holds.

- If $x \in [0, \alpha)$ then $tx + z \in [z, f(z))$. But $f(\cdot)$ is increasing on $[z, f(z))$ and $t \in \mathbb{R}_{++}$. It follows that $F(x) = (f(tx + z) - z)/t$ is increasing for $x \in [0, \alpha)$.
- As $f(f(z^-)) \le f(f(z))$ and f is increasing on $[z, f(z^-))$, it follows that

$$F(\alpha^-) = \frac{\lim_{x \uparrow f(z^-)} f(u) - z}{t} + 1 \le \frac{f(f(z^-)) - z}{t} + 1 \le \frac{f(f(z)) - z}{t} + 1 = F(\alpha).$$

- If $x \in [\alpha, 1)$ then $tx - t + z \in [f(z^-), z)$. But $f(\cdot)$ is increasing on $[f(z^-), z)$ and $t \in \mathbb{R}_{++}$. It follows that $F(x) = (f(tx - t + z) - z)/t$ is increasing for $x \in [\alpha, 1)$.
- The definition of F gives

$$F(1^-) = \frac{f(z^-) - z}{t} + 1 = \alpha + 1 = F(1).$$

In summary, $F(x)$ is increasing for $x \in [0,1]$. But as $F(x+1) = F(x)+1$ for $x \in \mathbb{R}$ we conclude that F is increasing for $x \in \mathbb{R}$. Therefore F3 holds.

Now, we note that for any $m \in \mathbb{Z}_+$ and any $x \in [\alpha-1, \alpha)$ we have

$$\frac{f^{(m)}(tx+z)-z}{t} \in [\alpha-1, \alpha). \tag{45}$$

Indeed, for $m=0$ we have $(f^{(m)}(tx+z)-z)/t = x \in [\alpha-1,\alpha)$. Also if $y \in [f(z), f(z^-))$ then the assumptions about f show that $f(y) \in [f(z), f(z^-))$, while if $x \in [\alpha-1,\alpha)$ then the definitions of t and α show that $tx+z \in [f(z), f(z^-))$. Thus $(f^{(m)}(tx+z)-z)/t \in [\alpha-1,\alpha)$.

Now, we show by induction that for any for any $x \in [\alpha-1,\alpha)$ and $n \in \mathbb{Z}_+$ we have

$$F^{(n)}(\{x\}) - \frac{f^{(n)}(tx+z)-z}{t} \in \mathbb{Z}. \tag{46}$$

In the base case $n=0$ we have $F^{(0)}(\{x\}) - (f^{(0)}(tx+z)-z)/t = \{x\} - ((tx+z)-z)/t \in \mathbb{Z}$. For the inductive step, let $p := F^{(n)}(\{x\})$ and $q := (f^{(n)}(tx+z)-z)/t$ and suppose that $p-q \in \mathbb{Z}$. Then we have

$$F^{(n+1)}(\{x\}) = F(p) = F(q)+p-q$$

as F satisfies F2 and by the assumption that $p-q \in \mathbb{Z}$. Furthermore, $q \in [\alpha-1,\alpha)$, by (45) so that for some $k \in \mathbb{Z}$,

$$\begin{aligned}
F^{(n+1)}(\{x\}) &= p-q+\begin{cases} \dfrac{f(tq+z)-z}{t}+1 & \text{if } q \in [0,\alpha) \\ \dfrac{f(t(q+1)-t+z)-z}{t} & \text{if } q \in [\alpha-1,0) \end{cases} \\
&= \frac{f(tq+z)-z}{t}+k \\
&= \frac{f(t\frac{f^{(n)}(tx+z)-z}{t}-z)}{t}+k \\
&= \frac{f^{(n+1)}(tx+z)-z}{t}+k.
\end{aligned}$$

Therefore (46) is true.

From (45) and (46) it follows that for any $x \in [\alpha-1,\alpha)$ and $n \in \mathbb{Z}_+$ we have

$$\begin{aligned}
\{F^{(n-1)}(\{x\})\} \in [0,\alpha) &\Leftrightarrow \left\{\frac{f^{(n-1)}(tx+z)-z}{t}\right\} \in [0,\alpha) \\
&\Leftrightarrow \frac{f^{(n-1)}(tx+z)-z}{t} \in [0,\alpha) \\
&\Leftrightarrow f^{(n-1)}(tx+z) \geq z.
\end{aligned} \tag{47}$$

Therefore F4 holds. This completes the proof. ■

Lemma 62 *Suppose $F : \mathbb{R} \to \mathbb{R}$ satisfies Claims F1-F3 of Lemma 61. Let $\alpha := F(0)$ and for $x \in \mathbb{R}$ let $s(x)$ denote the infinite word with letters*

$$s_n(x) := \begin{cases} 1 & \text{if } \{F^{(n-1)}(x)\} \in [0,\alpha) \\ 0 & \text{if } \{F^{(n-1)}(x)\} \in [\alpha,1) \end{cases}$$

for $n \in \mathbb{Z}_{++}$. Then $s(x)$ is 1-balanced.

Proof For any $n \in \mathbb{Z}_{++}$ and any $x \in [0,1)$ we have

$$\begin{aligned}
&\lfloor F^{(n)}(x)-\alpha\rfloor - \lfloor F^{(n-1)}(x)-\alpha\rfloor \\
&\quad= \lfloor F(r+z)-\alpha\rfloor - \lfloor r+z-\alpha\rfloor && \text{for } r := \{F^{(n-1)}(x)\}, z := \lfloor F^{(n-1)}(x)\rfloor \\
&\quad= \lfloor F(r)+z-\alpha\rfloor - \lfloor r+z-\alpha\rfloor && \text{as } F(u)+1 = F(u) \text{ for } u \in \mathbb{R} \\
&\quad= \lfloor F(r)-\alpha\rfloor - \lfloor r-\alpha\rfloor && \text{as } \lfloor r+z\rfloor = \lfloor r\rfloor + z \text{ since } z \in \mathbb{Z} \\
&\quad= -\lfloor r-\alpha\rfloor && \text{as } \alpha = F(0) \leq F(r) < F(1) = \alpha+1 \\
&&& \text{since } F(\cdot) \text{ is increasing and } r \in [0,1) \\
&\quad= \lceil \alpha - r\rceil \\
&\quad= \begin{cases} 1 & \text{if } r \in [0,\alpha) \\ 0 & \text{if } r \in [\alpha,1) \end{cases} \\
&\quad= s_n(x).
\end{aligned}$$

Therefore, for any $m \in \mathbb{Z}_+$,

$$\sum_{k=1}^{m} s_k(x) = \sum_{k=1}^{m}\Big(\lfloor F^{(k)}(x)-\alpha\rfloor - \lfloor F^{(k-1)}(x)-\alpha\rfloor\Big) = \lfloor F^{(m)}(x)-\alpha\rfloor - \lfloor x-\alpha\rfloor. \tag{48}$$

Now we show by induction that for any $m \in \mathbb{Z}_+$,

$$F^{(m)}(z) - F^{(m)}(y) \in [0,1) \qquad \text{for any } y,z \in \mathbb{R} \text{ with } z-y \in [0,1). \tag{49}$$

The base case with $m=0$ reads $z-y \in [0,1)$ which is true. For the inductive step, let $z' := F^{(m-1)}(z)$ and $y' := F^{(m-1)}(y)$ for $m-1 \in \mathbb{Z}_+$ and assume that $z'-y' \in [0,1)$. Then $F(y') \leq F(z')$ as $y' \leq z'$ and F is increasing. Also $F(z') < F(y'+1) = F(y')+1$ as $z'-y' < 1$, as F is increasing and as $F(u+1) = F(u)+1$ for any $u \in \mathbb{R}$. Therefore $F^{(m)}(z) - F^{(m)}(y) = F(z') - F(y') \in [0,1)$.

For any choice of $m \in \mathbb{Z}_+$ and $x,y \in [0,1)$, we wish to prove that $|\Delta| \leq 1$ for

$$\begin{aligned}
\Delta &:= |s_1(x)s_2(x)\dots s_m(x)|_1 - |s_1(y)s_2(y)\dots s_m(y)|_1 \\
&= \sum_{k=1}^{m}(s_k(x)-s_k(y)) = \underbrace{\Big(\lfloor F^{(m)}(x)-\alpha\rfloor - \lfloor F^{(m)}(y)-\alpha\rfloor\Big)}_{=:A} - \underbrace{(\lfloor x-\alpha\rfloor - \lfloor y-\alpha\rfloor)}_{=:B}
\end{aligned}$$

using equation (48). But if $x \geq y$ then $x-y \in [0,1)$ so (49) shows that A and B are both of the form $\lfloor a\rfloor - \lfloor b\rfloor$ for some $a-b \in [0,1)$ and it follows that both A and B are in $\{0,1\}$.

Whereas if $x \leq y$ then $y - x \in [0, 1)$ so both A and B are in $\{-1, 0\}$. We conclude that $|\Delta| = |A - B| \leq |1 - 0| = 1$. ■

The following is Theorem 3.1 of Dulucq and Gouyou-Beauchamps (1990) and provides the missing link between 1-balanced words and mechanical words.

Lemma 63 *Suppose w is a word of length $n \in \mathbb{Z}_{++}$ with $|w|_0|w|_1 > 0$. Then w is 1-balanced if and only if $w_k = \left\lfloor \frac{pk+r}{q} \right\rfloor - \left\lfloor \frac{p(k-1)+r}{q} \right\rfloor$ for $k = 1, 2, \ldots, n$, where $p, q, r \in \mathbb{Z}$ satisfy $0 < p < q \leq n$, $0 \leq r < q \leq n$ and $\gcd(p, q) = 1$.*

Now we are ready to describe the itineraries of maps-with-gaps, for arbitrary initial states and thresholds.

Lemma 64 *Suppose ϕ_0, ϕ_1 satisfy A2 and that $n \in \mathbb{Z}_{++}$, $x \in \mathcal{I}$, $s \in \overline{\mathbb{R}}$ and $a \in \{0, 1\}$. Then*

$$\sigma(x|s^-)_{1:n} = l^m w$$

for some $l \in \{0, 1\}$, some $m \in \{0, 1, \ldots, n\}$, and some factor w of a lower mechanical word.

Proof First, note that if $a, b \in \mathcal{I}$, then Lemma 35 shows that

$$b > y_1 \Rightarrow \phi_{1^m}(a) < b \text{ for some } m \in \mathbb{Z}_{++} \tag{50}$$
$$b \leq y_0 \Rightarrow \phi_{0^m}(a) \geq b \text{ for some } m \in \mathbb{Z}_{++}. \tag{51}$$

We consider seven cases.

1. Say $s \leq y_1$ and $x \geq s$. Then $\sigma(x|s^-) = 1\sigma(\phi_1(x)|s^-)$. But if $x > y_1$ then Lemma 33 gives $\phi_1(x) > y_1 \geq s$, whereas if $x \leq y_1$ then $\phi_1(x) \geq x \geq s$. In both cases, $\sigma(\phi_1(x)|s^-) = 1\sigma(\phi_{11}(x)|s^-)$. Repeating this argument gives $\sigma(x|s^-) = 1^\infty$, so the claim is true.

2. Say $s \leq y_1$ and $x < s$. Then $\sigma(x|s^-)$ begins with 0, and as $s < y_0$ it follows from (51) that there is a least $m \in \mathbb{Z}_{++}$ with $\phi_{0^m}(x) \geq s$. Thus $\sigma(x|s^-) = 0^m\sigma(\phi_{0^m}(x)|s^-) = 0^m1^\infty$ by Case 1, so the claim is true.

3. Say $s \in (y_0, y_1)$ and $x \in [\phi_1(s), \phi_0(x))$. As ϕ_0, ϕ_1 satisfy A2, Lemmas 59, 61 and 62 together show that $\sigma(x|s^-)$ is a 1-balanced word. Thus Lemma 63 shows that $\sigma(x|s^-)_{1:n} = w$ for some factor w of a lower mechanical word, so the claim is true.

4. Say $s \in (y_0, y_1)$ and $x < \phi_1(s)$. Then Lemma 33 shows that $\phi_1(s) < s$. Thus $x < s$, so $\sigma(x|s^-)$ begins with 0, and $\phi_1(s) < y_0$, so (51) shows that there is a least $m \in \mathbb{Z}_{++}$ with $\phi_{0^m}(x) \geq \phi_1(s)$. Thus $\sigma(x|s^-) = 0^m\sigma(\phi_{0^m}(x)|s^-)$ where $\sigma(\phi_{0^m}(x)|s^-)_{1:n} = w$ for some factor w of a lower mechanical word, by Case 3, so the claim is true.

5. Say $s \in (y_0, y_1)$ and $x \geq \phi_0(s)$. Then arguing as in Case 4, but using (50), shows that $\sigma(x|s^-)_{1:n} = 1^m w$ for some $m \in \mathbb{Z}_{++}$ and some factor w of a lower mechanical word.

6. Say $s \geq y_0$ and $x < s$. Then arguing as in Case 1 shows that $\sigma(x|s^-) = 0^\infty$.

7. Say $s \geq y_0$ and $x \geq s$. Then arguing as in Case 2 shows that $\sigma(x|s^-) = 1^m0^\infty$ for some $m \in \mathbb{Z}_{++}$.

This completes the proof. ■

The following is an analogue of Lemma 52 in which only the threshold varies.

Lemma 65 *Suppose ϕ_0, ϕ_1 satisfy A2 and $x \in \mathcal{I}$. Then $\sigma(x|s^-)$ is a lexicographically non-increasing function of $s \in \overline{\mathbb{R}}$.*

Proof If $s, t \in \overline{\mathbb{R}}$ and $\sigma(x|s^-) \succ \sigma(x|t^-)$, then for some finite word u and words v, w,

$$\sigma(x|s^-) = u1v, \qquad \sigma(x|t^-) = u0w.$$

So the definition of the itinerary gives $t > \phi_u(x) \geq s$. This completes the proof. ■

The following is Theorem 17 and part of Corollary 18 of Mignosi (1991).

Lemma 66 *Let $\mathcal{A}_n$ be the set of factors of length $n \in \mathbb{Z}_{++}$ of lower mechanical words. Then*

$$\operatorname{card}(\mathcal{A}_n) = 1 + \sum_{i=1}^{n} (n - i + 1)\,\text{EulerTotient}(i) = \frac{2n^3}{\ell^2} + O(n^2 \log n).$$

Now we are ready to bound the number of discontinuities of the itinerary of a map-with-a-gap as a function of its threshold.

Lemma 67 *Suppose ϕ_0, ϕ_1 satisfy A2, that $t \in \mathbb{Z}_+$, $x \in \mathcal{I}$ and $a \in \{0, 1\}$. Then the mapping $s \mapsto A_{1:t}(x, a; s)$ for $s \in \overline{\mathbb{R}}$ has at most a polynomial number $p(t)$ of discontinuities.*

Proof Let $\mathcal{A}_n$ be the set of all factors of length n of lower mechanical words. Let $\mathcal{F}_t$ be the set of all words of the form $l^m w$ for some $l \in \{0, 1\}$, some $m \in \{0, 1, \ldots, n\}$ and some factor w of a lower mechanical word. By Lemma 64, the word $A_{1:t}(x, a; s)$ is in $\mathcal{F}_t$. Also, Lemma 65, shows that the mapping $s \mapsto A_{1:t}(x, a; s)$ is lexicographically non-increasing. Thus, the number of discontinuities of this mapping is at most

$$\operatorname{card}(\mathcal{F}_t) = \operatorname{card}\left(\{l^m w : l \in \{0, 1\}, m \in \mathbb{Z}_+, m \leq t, w \in \mathcal{A}_{t-m}\}\right) = 2\sum_{m=0}^{t} \operatorname{card}(\mathcal{A}_{t-m}).$$

Finally, Lemma 66 shows the right-hand side is $O(t^4)$. This completes the proof. ■

Proof [Proof of Theorem 14.] The theorem simply couples together Lemmas 64, 65 and 67. ■

Appendix B. Proof of Lemma 22

Demonstrating Lemma 22 by combining results in Marshall et al. (2010) requires as much text as a direct proof.

Proof Let $\mathcal{X}$ be the set of sequences X with components $X_k = \sum_{i=1}^{k} x_i$ for $k = 1, 2, \ldots, n$ where $x_1, x_2, \ldots, x_n$ is a non-decreasing sequence on $\mathbb{R}_{++}$. Let $g : \mathcal{X} \to \mathbb{R}$ be the function

$$g(X) := f_1(X_1) + \sum_{i=2}^{n} f_i(X_i - X_{i-1}).$$

Let $f_i'(\cdot)$ denote the (sub)gradient of $f_i(\cdot)$. For $i = 1, 2, \ldots, n-1$, the (sub)gradients of $g(\cdot)$ are

$$\frac{\partial g(X)}{\partial X_i} = f_i'(x_i) - f_{i+1}'(x_{i+1}) \leq f_{i+1}'(x_i) - f_{i+1}'(x_{i+1}) \leq 0$$

where the first inequality holds as $f_i'(x) \leq f_{i+1}'(x)$ for $x \in \mathbb{R}_{++}$ (by Hypothesis 4) and the second holds as $x_i \leq x_{i+1}$ and $f_{i+1}(\cdot)$ is convex (by Hypothesis 3). Also,

$$\frac{\partial g(X)}{\partial X_n} = f_n'(x_n) \leq 0$$

as $f_n(\cdot)$ is non-increasing (by Hypothesis 3). Therefore $g(\cdot)$ is non-increasing in all of its arguments. As the sequences A, B with components $A_k := \sum_{i=1}^{k} a_i, B_k := \sum_{i=1}^{k} b_k$ are in $\mathcal{X}$ (by Hypothesis 1) and $A_k \leq B_k$ for $k = 1, 2, \ldots, n$ (by Hypothesis 2), it follows that $g(A) \geq g(B)$. So the definition of $g(\cdot)$ gives

$$\sum_{i=1}^{n} f(a_i) = g(A) \geq g(B) = \sum_{i=1}^{n} f(b_i)$$

as claimed. ■

Appendix C. Proof of Lemma 24

We start by recalling the definition of the matrix $M(w)$ corresponding to a given finite word w, which corresponds to the composition of Kalman-Filter variance updates, and introducing some related matrices $K, S(w)$ and X. We then prove Claims 1 to 5 of Lemma 24 in turn.

Definition 68 *Let I be the 2-by-2 identity matrix. For $r \in (0, 1]$ and $0 \leq a \leq b$, let*

$$F := \begin{pmatrix} r & 1/r \\ ar & (a+1)/r \end{pmatrix}, \qquad G := \begin{pmatrix} r & 1/r \\ br & (b+1)/r \end{pmatrix}, \qquad K := \begin{pmatrix} r & 1/r \\ r - r^3 & -r \end{pmatrix}.$$

Let $M(\epsilon) = I, M(0) = F, M(1) = G$ and for any finite word w let

$$M(w) = M(w_{|w|}) \cdots M(w_2) M(w_1), \qquad S(w) = \sum_{i=0}^{|w|} M(w_{1:i}).$$

Let

$$X := \begin{pmatrix} -r/(1-r^2) & 0 \\ 0 & 1/r \end{pmatrix}.$$

Remark 69 *We use the the following facts repeatedly without mention. Clearly* $\det(F) = \det(G) = 1$, *so that* $\det(M(w)) = 1$ *for any word* w. *Also,* $KF = F^{-1}K, KG = G^{-1}K$ *and* $K^2 = I$. *Thus for* $A \in \{KF, KG, K\}$ *we have* $A^2 = I$, *so* A *is an involutory matrix. Thus* $KM(w)^{-1}K = M(w^R)$, *where* w^R *denotes the reverse* $w_n \ldots w_2 w_1$ *of a word* $w = w_1 w_2 \ldots w_n$.

Notation. For a vector $v \in \mathbb{R}^m$ where $m \in \mathbb{Z}_{++}$, we write $v > 0$ if $v_i > 0$ for $i = 1, 2, \ldots, m$ and $v \geq 0$ if $v_i \geq 0$ for $i = 1, 2, \ldots, m$. Similarly, for a matrix $P \in \mathbb{R}^{m \times n}$ where $m, n \in \mathbb{Z}_{++}$, we write $P > 0$ ($P \geq 0$) if $P_{ij} > 0$ ($P_{ij} \geq 0$) for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$.

C.1. Claim 1

We require one simple Lemma.

Lemma 70 *Suppose* $a \in \mathbb{R}_+$ *and* $r \in (0, 1]$. *Then the fixed point* y_0 *satisfies*

$$y_0 \leq \frac{1}{1 - r^2}.$$

Proof As the positive root of

$$y_0 = \frac{r^2 y_0 + 1}{a r^2 y_0 + 1 + a}$$

is a decreasing function of a for $a \in \mathbb{R}_+$, setting $a = 0$ gives an upper bound. This upper bound u satisfies $u = r^2 u + 1$, so that $u = 1/(1 - r^2)$. ■

Proof [Proof of Claim 1 of Lemma 24.] We prove the claim for x satisfying

$$\phi_p(0) \leq x \leq \frac{1}{r - r^2},$$

noting that

$$\phi_p\left(\frac{1}{1-r^2}\right) \leq \frac{1}{1-r^2} \leq \frac{1}{r-r^2}$$

where the first inequality follows from Lemma 33 (in Appendix A) as Lemma 70 gives $1/(1-r^2) \geq y_0 \geq y_p$, and the second inequality holds as $r \in (0, 1]$.

For any word w and for $k = 1, 2, \ldots, |w|$, let

$$\begin{pmatrix} u_k \\ v_k \end{pmatrix} := M(w_{1:k}) \begin{pmatrix} x \\ 1 \end{pmatrix}.$$

Clearly u_k, v_k are positive as $x \geq \phi_p(0) \geq 0$ and

$$M(w_{1:k}) \begin{pmatrix} x \\ 1 \end{pmatrix} \geq \begin{pmatrix} r & \frac{1}{r} \\ 0 & \frac{1}{r} \end{pmatrix}^k \begin{pmatrix} 0 \\ 1 \end{pmatrix} > 0.$$

Now, for any $0 \leq z \leq 1/(r-r^2)$ and $H = M(w_k)$, noting that $a, b, r \geq 0$ and $r \leq 1$ gives

$$\frac{H_{11}z + H_{12}}{H_{21}z + H_{22}} \leq r^2 z + 1 \leq \frac{r^2}{r-r^2} + 1 \leq \frac{1}{r-r^2}.$$

For $k = 1, 2, \ldots, |w|$, induction using this inequality proves that

$$\frac{u_k}{v_k} \leq \frac{1}{r-r^2}.$$

(For $k = 1$, put $z = x \leq 1/(r-r^2)$. For $k > 1$, assume that $z = u_{k-1}/v_{k-1} \leq 1/(r-r^2)$.) Thus

$$\begin{pmatrix} u_{k+1} - u_k \\ v_{k+1} - v_k \end{pmatrix} = (M(w_{k+1}) - I) \begin{pmatrix} \frac{u_k}{v_k} \\ 1 \end{pmatrix} v_k \geq \begin{pmatrix} r-1 & \frac{1}{r} \\ 0 & \frac{1}{r} - 1 \end{pmatrix} \begin{pmatrix} \frac{1}{r-r^2} \\ 1 \end{pmatrix} v_k \geq 0.$$

But both $a_k(x), b_k(x)$ are of the form u_k and $c_k(x), d_k(x)$ are of the form v_k for appropriate w. Thus $a_{1:m}(x), b_{1:m}(x), c_{1:m}(x)$ and $d_{1:m}(x)$ are non-decreasing and positive. ■

C.2. Claim 2

To prove Claim 2 we need two simple Lemmas.

Lemma 71 *If w is a word, then $M(w) - M(w^R) = tr(KM(w))K$.*

Proof For any $C \in \mathbb{R}^{2\times 2}$, direct calculation gives $C - K\text{adj}(C)K = \text{tr}(KC)K$. But $\det(M(w)) = 1$, so $M(w) - M(w^R) = M(w) - KM(w)^{-1}K = M(w) - K\text{adj}(M(w))K = \text{tr}(KM(w))K$. ■

Lemma 72 *Suppose p is a palindrome, $r \in (0,1]$ and $n \in \mathbb{Z}_+$. Then for some $x \geq 0$,*

$$M((10p)^n 10) - M((01p)^n 01) = xK.$$

Proof For any matrices $P, Q \in \mathbb{R}^{2\times 2}$ with $\det(P) = 1$ and $Q \geq 0$, direct calculation gives

$$\begin{aligned}&[QGFP]_{22}[QFGP]_{21} - [QFGP]_{22}[QGFP]_{21} \\ &\quad = (b-a)\det(P)((1-r^2+a+b+ab)Q_{22}^2 + (2+a+b)Q_{22}Q_{21} + Q_{21}^2) \geq 0\end{aligned}$$

as $1 \geq r^2, b \geq a \geq 0$. Therefore, for any words w, w',

$$\frac{M(w01w')_{22}}{M(w01w')_{21}} \geq \frac{M(w10w')_{22}}{M(w10w')_{21}}. \tag{52}$$

Let $A = M((10p)^n 10), B = M((01p)^n 01)$. Repeated application of (52) gives

$$\frac{B_{22}}{B_{21}} = \frac{M(01p01p\cdots 01)_{22}}{M(01p01p\cdots 01)_{21}} \geq \frac{M(10p01p\cdots 01)_{22}}{M(10p01p\cdots 01)_{21}} \geq \frac{M(10p10p\cdots 10)_{22}}{M(10p10p\cdots 10)_{21}} = \frac{A_{22}}{A_{21}}.$$

As $A, B \geq 0$ and Lemma 71 gives $A = B + xK$ for some $x \in \mathbb{R}$, it follows that

$$(B+xK)_{21}B_{22} \geq (B+xK)_{22}B_{21} \quad \Rightarrow \quad K_{21}B_{22}x \geq K_{22}B_{21}x.$$

Finally, the fact that $K_{22} \leq 0 < K_{21}$ and $B \geq 0$ gives $x \geq 0$. ■

Proof [Proof of Claim 2 of Lemma 24.] For some $t \geq 0$, Lemma 72 gives

$$\begin{aligned}(d/dx)(b_1(x) - a_1(x)) &= [M((10p)^n 1) - M((01p)^n 0)]_{11} \\ &= [(F^{-1} - G^{-1})M((01p)^n 01) + tF^{-1}K]_{11} \\ &= [(0,0)M((01p)^n 01)]_1 + t[KF]_{11} \\ &= t\,(rF_{11} + (1/r)F_{21}) \\ &\geq 0\end{aligned}$$

and

$$\begin{aligned}b_1(\phi_p(0)) - a_1(\phi_p(0)) &= [M(p(10p)^n 1) - M(p(01p)^n 0)]_{12} \\ &= [(F^{-1} - G^{-1})M(p(01p)^n 01) + tF^{-1}KM(p)]_{12} \\ &= t[KFM(p)]_{12} \\ &= t(rM(p0)_{12} + (1/r)M(p0)_{22}) \\ &\geq 0.\end{aligned}$$

Also, if $k > 1$ and $w = p_{1:(k-2)}$, then

$$\begin{aligned}(d/dx)(b_k(x) - a_k(x)) &= [M((10p)^n 10w) - M((01p)^n 01w)]_{11} \\ &= t[M(w)K]_{11} \\ &= t(M(w)_{11}r + M(w)_{12}r(1-r^2)) \\ &\geq 0\end{aligned}$$

and as p is a palindrome, $p = sw^R$ for some word s, so

$$\begin{aligned}b_k(\phi_p(0)) - a_k(\phi_p(0)) &= [M(p(10p)^n 10w) - M(p(01p)^n 01w)]_{12} \\ &= t[M(w)KM(p)]_{12} \\ &= t[KM(w^R)^{-1}M(w^R)M(s)]_{12} \\ &= t[KM(s)]_{12} \\ &= t(rM(s)_{12} + (1/r)M(s)_{22}) \\ &\geq 0.\end{aligned}$$

This completes the proof. ■

C.3. Claim 3

Claim 3 of Lemma 24 is more challenging than Claims 1 and 2. We begin with six Lemmas.

Lemma 73 *Suppose p is any palindrome and $r \in (0,1]$. Then for any $k \in \mathbb{Z}_+$,*

$$\Delta_k := [M(((10p)^\infty)_{1:k}) - M(((01p)^\infty)_{1:k})]_{21} \geq 0.$$

Proof If $k = 1$ then

$$\begin{aligned}\Delta_k &= [M(1) - M(0)]_{21} \\ &= [G - F]_{21} \\ &= (b-a)r \\ &\geq 0.\end{aligned}$$

If $k = (n+1)|01p| + 1$ for some $n \in \mathbb{Z}_+$ then Lemma 72 shows there is an $x \geq 0$ such that

$$\begin{aligned}\Delta_k &= [M((10p)^n 10p1) - M((01p)^n 01p0)]_{21} \\ &= [M(p1)(xK + M((01p)^n 01)) - M(p0)M((01p)^n 01)]_{21} \\ &= [xM(p1)K + (G-F)M((01p)^{n+1})]_{21} \\ &= x(M(p1)_{21}r + M(p1)_{22}(r - r^3)) \\ &\qquad + (b-a)(rM((01p)^{n+1})_{12} + (1/r)M((01p)^{n+1})_{22}) \\ &\geq 0.\end{aligned}$$

Otherwise, there is a prefix w of p and an $n \in \mathbb{Z}_+$ such that

$$\begin{aligned}\Delta_k &= [M((10p)^n 10w) - M((01p)^n 01w)]_{21} \\ &= [M(w)(M((10p)^n 10) - M((01p)^n 01))]_{21} \\ &= [xM(w)K]_{21} \qquad \text{for some } x \geq 0 \text{ by Lemma 72} \\ &= x(M(w)_{21}r + M(w)_{22}(r - r^3)) \\ &\geq 0.\end{aligned}$$

This completes the proof. ∎

Lemma 74 *Suppose $p = ws$ is a palindrome, $n \in \mathbb{Z}_+$ and $r \in (0,1]$. Then*

$$[M(p(10p)^n 10w) - M(p(01p)^n 01w)]_{22} \leq 0.$$

Proof First note that for any finite word u,

$$M(u)_{22} \geq M(u)_{21}. \tag{53}$$

Indeed, if $u = \epsilon$ then $M(\epsilon) = I$ so the inequality holds. Otherwise, for some $c \in \{a, b\}$

$$M(u)_{22} - M(u)_{21} = \left[M(u_{2:|u|})M(u_1)\begin{pmatrix}-1\\1\end{pmatrix}\right]_2 = \left[M(u_{2:|u|})\frac{1}{r}\begin{pmatrix}1-r^2\\1+(1-r^2)c\end{pmatrix}\right]_2 \geq 0$$

as the definition of $M(\cdot)$ assumes that $0 \leq a < b$ so that $c \geq 0$, and as $r \in (0,1]$ and $M(u_{2:|u|}) \geq 0$.

Also, by Lemma 72, for some $x \geq 0$

$$\begin{aligned}[M(p(10p)^n 10w) - M(p(01p)^n 01w)]_{22} &= [M(s)^{-1}(M((10p)^{n+1}) - M((01p)^{n+1}))M(p)]_{22} \\ &= x[M(s)^{-1}KM(p)]_{22} \\ &= x[KM(ps^R)]_{22} \\ &= xr\left((1-r^2)M(ps^R)_{21} - M(ps^R)_{22}\right) \\ &\leq xr\left((1-r^2)M(ps^R)_{22} - M(ps^R)_{22}\right) \\ &\leq 0\end{aligned}$$

where the penultimate line is (53). ∎

Lemma 75 *Suppose p is a palindrome, $n \in \mathbb{Z}_+$ and $r \in (0,1]$. Then*

$$\left[\Big(M((10p)^n 10)X - M((01p)^n 01)X + M((10p)^n 1) - M((01p)^n 0)\Big)M(p)\right]_{22} = 0.$$

Proof Let $P = M(p)$. Solving $KP = P^{-1}K$ shows that there exist $f, h \in \mathbb{R}$ such that

$$P = \begin{pmatrix} \dfrac{1 - f^2r^2 + fhr^2 + f^2r^4}{fr^2 + h} & f \\ \dfrac{-1 - fh + h^2 + fhr^2}{fr^2 + h}r^2 & h \end{pmatrix}.$$

Directly substituting this expression into

$$Q_n := \left[\Big(FG(PFG)^n X - GF(PGF)^n X + G(PFG)^n - F(PGF)^n\Big)P\right]_{22}$$

shows that $Q_0 = Q_1 = 0$. (Showing that $Q_1 = 0$ directly is algebraically tedious and we had to check this with computer algebra. The authors would be interested in a short demonstration that $Q_1 = 0$ as this may give insight into related problems.)

Lemma 71, then the fact that $\mathrm{tr}(K) = 0$, then the cyclic property of the trace give

$$\mathrm{tr}(PFG) = \mathrm{tr}(GFP + \mathrm{tr}(KPFG)K) = \mathrm{tr}(GFP) = \mathrm{tr}(PGF).$$

So PFG and PGF are 2-by-2 matrices with unit determinant whose traces are equal. For some matrices of eigenvectors U, V and some eigenvalue $\lambda \geq 1$, such matrices may be written in the form

$$PFG = U\Lambda U^{-1}, \qquad PGF = V\Lambda V^{-1}, \qquad \Lambda := \begin{pmatrix}\lambda & 0\\ 0 & 1/\lambda\end{pmatrix}.$$

Thus, for some $\alpha, \beta \in \mathbb{R}$,

$$Q_n = \alpha\lambda^n + \beta\lambda^{-n}$$

But $Q_0 = 0$ implies that $\beta = -\alpha$. Thus $Q_1 = \alpha(\lambda - 1/\lambda) = 0$ implies that either $\alpha = 0$ or $\lambda = 1$. In either case, $Q_n = \alpha(\lambda^n - \lambda^{-n}) = 0$ for all $n \geq 0$. ∎

Lemma 76 *Suppose p is a palindrome. Then $S(p)KM(p) = KS(p)$.*

Proof We proceed by induction on the length of p. In the base case, $S(\epsilon)KM(\epsilon) = IKI = KS(\epsilon)$ and if a is a letter then $S(a)KM(a) = (I + M(a))KM(a) = K(I + M(a)^{-1})M(a) = KS(a)$. Otherwise, say $p = aqa$ where a is a letter and $S(q)KM(q) = KS(q)$. Then

$$\begin{aligned}
S(p)KM(p) &= (I + M(a) + S(q)M(a) + M(aqa))KM(aqa) \\
&= (I + M(a) + M(aqa))KM(aqa) + S(q)KM(a)^{-1}M(a)M(q)M(a) \\
&= K(I + M(a)^{-1} + M(aqa)^{-1})M(aqa) + S(q)KM(q)M(a) \\
&= K(M(aqa) + M(aq) + I) + KS(q)M(a) \\
&= KS(p).
\end{aligned}$$

■

Lemma 77 *Suppose w is a finite word and $r \in (0, 1]$. Then $[K(S(w) - XM(w))]_{22} \geq 0$.*

Proof We proceed by induction on the length of w. In the base case,

$$[K(S(\epsilon) - XM(\epsilon))]_{22} = [K(I - X)]_{22} = 1 - r \geq 0.$$

Otherwise, say $w = ul$ where $|l| = 1$, $|u| < \infty$ and $[K(S(u) - XM(u))]_{22} \geq 0$. Then

$$\begin{aligned}
[K(S(w) - XM(w))]_{22} &= [K(S(u) + M(w) - X(M(w) - M(u)) - XM(u))]_{22} \\
&= [K(S(u) - XM(u))]_{22} + [K(I - X(I - M(l)^{-1}))M(w)]_{22} \\
&\geq [K(I - X(I - M(l)^{-1}))M(w)]_{22} \\
&= (1 - r)(r^2 M(w)_{12} + M(w)_{22}) \\
&\geq 0
\end{aligned}$$

where in the penultimate line we substituted the definitions of K and X, noting that $M(l) \in \{F, G\}$, and where the final inequality follows as $r \leq 1$ and $M(w) \geq 0$. ■

Lemma 78 *Suppose p is a palindrome and $r \in (0, 1]$. Then for any $n \in \mathbb{Z}_+$*

$$[(S(10p) - I)M(p(10p)^n) - (S(01p) - I)M(p(01p)^n)]_{22} \geq 0.$$

Proof Let $P := M(p)$. Then

$$\begin{aligned}
&[(S(10p) - I)M(p(10p)^n) - (S(01p) - I)M(p(01p)^n)]_{22} \\
&= [S(p)(FG(PFG)^n - GF(PGF)^n)P + (G(PFG)^n - F(PGF)^n)P]_{22} \\
&= [S(p)(FG(PFG)^n - GF(PGF)^n)P - (FG(PFG)^n - GF(PGF)^n)XP]_{22} \\
&= [S(p)xKP - xKXP]_{22} \\
&= [xK(S(p) - X)P]_{22}
\end{aligned}$$

$$\geq 0$$

where the second equality uses Lemma 75, the third holds for some $x \geq 0$ by Lemma 72, the fourth follows from Lemma 76 and the final inequality is Lemma 77. ■

Proof [Proof of Claim 3 of Lemma 24.] Say $1 \leq k \leq m$. Using Lemmas 73, 74 and 78 successively gives

$$\begin{aligned}
\sum_{i=1}^{k}(d_i(x) - c_i(x)) &= \sum_{i=1}^{k}[M((10p)^n(10p)_{1:i}) - M((01p)^n(01p)_{1:i})]_{21}x \\
&\quad + \sum_{i=1}^{k}[M((10p)^n(10p)_{1:i}) - M((01p)^n(01p)_{1:i})]_{22} \\
&\geq \sum_{i=1}^{k}[M((10p)^n(10p)_{1:i}) - M((01p)^n(01p)_{1:i})]_{21}\frac{M(p)_{12}}{M(p)_{22}} \\
&\quad + \sum_{i=1}^{k}[M((10p)^n(10p)_{1:i}) - M((01p)^n(01p)_{1:i})]_{22} \\
&= \frac{1}{M(p)_{22}}\sum_{i=1}^{k}[M(p(10p)^n(10p)_{1:i}) - M(p(01p)^n(01p)_{1:i})]_{22} \\
&\geq \frac{1}{M(p)_{22}}\sum_{i=1}^{m}[M(p(10p)^n(10p)_{1:i}) - M(p(01p)^n(01p)_{1:i})]_{22} \\
&= \frac{1}{M(p)_{22}}[(S(10p) - I)M(p(10p)^n) - (S(01p) - I)M(p(01p)^n)]_{22} \\
&\geq 0.
\end{aligned}$$

This completes the proof. ■

C.4. Claim 4

The proof of Claim 4 requires only one preparatory Lemma.

Lemma 79 *Suppose w is any finite word. Then*

$$[M(w)^{-1}]_{21} \leq 0 \leq [M(w)^{-1}]_{22}.$$

Proof We use induction on the length of w. In the base case, $M(\epsilon) = I$, for which

$$[I^{-1}]_{21} = 0 \leq 1 = [I^{-1}]_{22}.$$

Otherwise, suppose $w = 0u$ (the case $w = 1u$ is similar), let $U := M(u)^{-1}$ and assume that

$$U_{21} \leq 0 \leq U_{22}.$$

Then the induction assumption and and fact that $a, r \geq 0$ give

$$[M(w)^{-1}]_{21} = [UF^{-1}]_{21} = \frac{1}{r}((1+a)U_{21} - ar^2 U_{22}) \leq 0$$
$$[M(w)^{-1}]_{22} = [UF^{-1}]_{22} = \frac{1}{r}(U_{22}r^2 - U_{21}) \geq 0.$$

This completes the proof. ■

Proof [Proof of Claim 4 of Lemma 24.] Claim 3 of Lemma 24 applied for $k = 1$ shows that

$$c_1(x) \leq d_1(x).$$

For $k = 2, 3, \ldots, m$, as Lemma 73 shows that $c_k(x) - d_k(x)$ is a decreasing function of x, it suffices to prove that

$$\left[M((01p)^n(01p)_{1:k}) \begin{pmatrix} \phi_p\left(\frac{1}{1-r^2}\right) \\ 1 \end{pmatrix}\right]_2 \geq \left[M((10p)^n(10p)_{1:k}) \begin{pmatrix} \phi_p\left(\frac{1}{1-r^2}\right) \\ 1 \end{pmatrix}\right]_2.$$

The left-hand side minus the right-hand side, up to a positive factor, is

$$\begin{aligned}
&\left[\left(M((01p)^n(01p)_{1:k}) - M((10p)^n(10p)_{1:k})\right) M(p) \begin{pmatrix} 1 \\ 1-r^2 \end{pmatrix}\right]_2 \\
&\quad = -z\left[M(p_{1:(k-2)})KM(p)\begin{pmatrix} 1 \\ 1-r^2 \end{pmatrix}\right]_2 \quad \text{for some } z \geq 0 \text{ by Lemma 72} \\
&\quad = -z\left[M(p_{1:(|p|-k+2)})^{-1}K\begin{pmatrix} 1 \\ 1-r^2 \end{pmatrix}\right]_2 \\
&\quad = -z\left[M(p_{1:(|p|-k+2)})^{-1}\begin{pmatrix} \frac{1}{r} \\ 0 \end{pmatrix}\right]_2 \\
&\quad = -\frac{z}{r}[M(p_{1:(|p|-k+2)})^{-1}]_{21} \\
&\quad \geq 0
\end{aligned}$$

where the last line is Lemma 79. Therefore

$$c_k(x) \geq d_k(x) \quad \text{for } k = 2, 3, \ldots, m \text{ and } x \leq \phi_p\left(\frac{1}{1-r^2}\right).$$

This completes the proof. ■

C.5. Claim 5

Proof [Proof of Claim 5 of Lemma 24.] We show that

$$\phi_w(0) \leq y_{01w} < y_{10w} \leq \phi_w\left(\frac{1}{1-r^2}\right)$$

for any finite word w (not just for palindromes p).

The first inequality follows as

$$\phi_w(0) \leq \phi_w(\phi_{01}(0)) = \phi_{01w}(0) \leq y_{01w}$$

as $0 \leq \phi_{01}(0)$, as ϕ_w is increasing, as $0 \leq y_{01w}$ and by Lemma 33 (in Appendix A).

The second inequality holds as $\phi_{01}(x) < \phi_{10}(x)$ for $x \in \mathbb{R}_+$. Thus

$$y_{01w} = \phi_w(\phi_{01}(y_{01w})) < \phi_w(\phi_{10}(y_{01w})) = \phi_{10w}(y_{01w})$$

so applying Lemma 33 gives

$$y_{01w} < y_{10w}.$$

The third inequality holds as

$$y_{10w} = \phi_w(y_{w10}) < \phi_w(y_0) \leq \phi_w\left(\frac{1}{1-r^2}\right)$$

by definition of y_{10w}, as $y_{w10} < y_0$, as ϕ_w is increasing and by Lemma 70.

This completes the proof. ■

References

Kazuyuki Aihara and Hideyuki Suzuki. Theory of hybrid dynamical systems and its applications to biological and medical systems. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 368(1930):4893–4914, 2010.

Eitan Altman and Shaler Stidham Jr. Optimality of monotonic policies for two-action Markovian decision processes, with applications to control of queues with delayed information. *Queueing Systems*, 21(3-4):267–291, 1995.

Eitan Altman, Bruno Gaujal, and Arie Hordijk. Multimodularity, convexity, and optimization properties. *Mathematics of Operations Research*, 25(2):324–347, 2000.

Michael Athans. On the determination of optimal costly measurement strategies for linear stochastic systems. *Automatica*, 8(4):397–412, 1972.

Ashwinkumar Badanidiyuru, Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Streaming submodular maximization: Massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 671–680, 2014.

Mario Bernardo, Chris Budd, Alan R. Champneys, and Piotr Kowalczyk. *Piecewise-Smooth Dynamical Systems: Theory and Applications*. Springer Science and Business Media, 2008.

Jean Berstel, Aaron Lauve, Christophe Reutenauer, and Franco Saliola. *Combinatorics on Words: Christoffel Words and Repetitions in Words*. CRM Monograph Series, 2008.

Dimitris Bertsimas and José Niño-Mora. Conservation laws, extended polymatroids and multiarmed bandit problems; a polyhedral approach to indexable systems. *Mathematics of Operations Research*, 21(2):257–306, 1996.

Thierry Bousch and Jean Mairesse. Asymptotic height optimization for topical IFS, Tetris heaps, and the finiteness conjecture. *Journal of the American Mathematical Society*, 15 (1):77–111, 2002.

Claude Brezinski. Convergence acceleration during the 20th century. *Journal of Computational and Applied Mathematics*, 122(1):1–21, 2000.

Michael Carter and Bruce van Brunt. *The Lebesgue-Stieltjes Integral*. Springer, 2000.

Yuxin Chen, Hiroaki Shioi, Cesar Montesinos, Lian Pin Koh, Serge Wich, and Andreas Krause. Active detection via adaptive submodularity. In *Proceedings of The 31st International Conference on Machine Learning*, pages 55–63, 2014.

Erwin B. Christoffel. Observatio arithmetica. *Annali di Matematica Pura ed Applicata*, 6: 148–152, 1875.

Christopher R. Dance and Tomi Silander. When are Kalman-filter restless bandits indexable? In *Advances in Neural Information Processing Systems*, pages 1711–1719, 2015.

Christopher R. Dance and Tomi Silander. Optimal policies for observing time series and related restless bandit problems. *arXiv preprint arXiv:1703.10010*, 2017.

Ramez M. Daoud, Hassanein H. Amer, Hany M. Elsayed, and Y. Sallez. Ethernet-based car control network. In *Electrical and Computer Engineering, 2006. CCECE'06. Canadian Conference on*, pages 1031–1034. IEEE, 2006.

Abhimanyu Das and David Kempe. Algorithms for subset selection in linear regression. In *Proceedings of the 40th ACM Symposium on the Theory of Computing*, pages 45–54. ACM, 2008.

Geoffrey Davis, Stéphane Mallat, and Marco Avellaneda. Adaptive greedy approximations. *Constructive Approximation*, 13(1):57–98, 1997.

Robert Devaney. *An Introduction to Chaotic Dynamical Systems*. Westview Press, 2008.

Serge Dulucq and Dominique Gouyou-Beauchamps. Sur les facteurs des suites de Sturm. *Theoretical Computer Science*, 71(3):381–400, 1990.

John Gittins, Kevin Glazebrook, and Richard Weber. *Multi-Armed Bandit Allocation Indices*. John Wiley and Sons, 2011.

Kevin Glazebrook, Hodge David, and Kirkbride Chris. General notions of indexability for queueing control and asset management. *Annals of Applied Probability*, 21(3):876–907, 2011.

Amy Glen and Jacques Justin. Episturmian words: a survey. *RAIRO-Theoretical Informatics and Applications*, 43(3):403–442, 2009.

Ronald Graham, Donald Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Addison-Wesley, 1994.

Sudipto Guha, Kamesh Munagala, and Peng Shi. Approximation algorithms for restless bandit problems. *Journal of the ACM*, 58(1):3, 2010.

Vijay Gupta, Timothy H. Chung, Babak Hassibi, and Richard M. Murray. On a stochastic sensor selection algorithm with applications in sensor scheduling and sensor coverage. *Automatica*, 42(2):251–260, 2006.

Wassim M. Haddad, Vijay S. Chellaboina, and Sergey G. Nersesov. *Impulsive and Hybrid Dynamical Systems: Stability, Dissipativity, and Control*. Princeton University Press, 2014.

Onésimo Hernández-Lerma and Jean Bernard Lasserre. *Further Topics in Discrete-Time Markov Control Processes*. Springer, 1999.

Alfred O. Hero and Douglas Cochran. Sensor management: Past, present, and future. *IEEE Sensors Journal*, 11(12):3064–3075, 2011.

Sandra Hirche, Manuel Ferre, Jordi Barrio, Claudio Melchiorri, and Martin Buss. Bilateral control architectures for telerobotics. In *Advances in Telerobotics*, pages 163–176. Springer, 2007.

Stephen John Hogan, L. Higham, and T.C.L. Griffin. Dynamics of a piecewise linear map with a gap. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 463, pages 49–65. The Royal Society, 2007.

Rudolph E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

Thomas H. Kerr. Modeling and evaluating an empirical INS difference monitoring procedure used to sequence SSBN navaid fixes. *Navigation*, 28(4):263–285, 1981.

Leonard Kleinrock. A conservation law for a wide class of queueing disciplines. *Naval Research Logistics Quarterly*, 12(2):181–192, 1965.

Andrey N. Kolmogorov and Sergei V. Fomin. *Introductory Real Analysis, translated by Richard A. Silverman*. Dover, New York, 1975.

Victor S. Kozyakin. Sturmian sequences generated by order-preserving circle maps. Available as http://cool.iitp.ru/personal/kozyakin/koz-11-5.pdf., 2003.

Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(Feb):235–284, 2008.

Andreas Krause, Ram Rajagopal, Anupam Gupta, and Carlos Guestrin. Simultaneous optimization of sensor placements and balanced schedules. *IEEE Transactions on Automatic Control*, 56(10):2390–2405, 2011.

Julia Kuhn, Michel Mandjes, and Yoni Nazarathy. Exploration vs exploitation with partially observable gaussian autoregressive arms. In *Proceedings of the 8th International Conference on Performance Evaluation Methodologies and Tools*, pages 209–216. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2014.

Harold J. Kushner. On the optimum timing of observations for linear control systems with unknown initial state. *IEEE Transactions on Automatic Control*, 9(2):144–150, 1964.

Barbara La Scala and Bill Moran. Optimal target tracking with restless bandits. *Digital Signal Processing*, 16(5):479–487, 2006.

Maialen Larrañaga, Urtzi Ayesta, and Ina Verloop. Dynamic control of birth-and-death restless bandits: application to resource-allocation problems. *IEEE/ACM Transactions on Networking*, 24(6):3812–3825, 2016.

Jérôme Le Ny, Munther Dahleh, and Eric Feron. Multi-UAV dynamic routing with partial observations using restless bandit allocation indices. In *American Control Conference, 2008*, pages 4220–4225. IEEE, 2008.

Jérôme Le Ny, Eric Feron, and Munther Dahleh. Scheduling continuous-time Kalman filters. *IEEE Transactions on Automatic Control*, 56(6):1381–1394, 2011.

Keqin Liu and Qing Zhao. Indexability of restless bandit problems and optimality of Whittle index for dynamic multichannel access. *IEEE Transactions on Information Theory*, 56 (11):5547–5567, 2010.

M. Lothaire. *Algebraic Combinatorics on Words*. Cambridge University Press, 2002.

Oleg Makarenkov and Jeroen S. W. Lamb. Dynamics and bifurcations of nonsmooth systems: A survey. *Physica D: Nonlinear Phenomena*, 241(22):1826–1844, 2012.

Albert Marshall, Ingram Olkin, and Barry Arnold. *Inequalities: Theory of Majorization and its Applications*. Springer Science and Business Media, 2010.

Lewis Meier, John Peschon, and Robert Dressler. Optimal control of measurement subsystems. *IEEE Transactions on Automatic Control*, 12(5):528–536, 1967.

Filippo Mignosi. On the number of factors of Sturmian words. *Theoretical Computer Science*, 82(1):71–84, 1991.

Adam Molin and Sandra Hirche. On LQG joint optimal scheduling and control under communication constraints. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 5832–5838. IEEE, 2009.

William Moran, Sofia Suvorova, and Stephen Howard. Application of sensor scheduling concepts to radar. In *Foundations and Applications of Sensor Management*, pages 221–256. Springer, 2008.

Marston Morse and Gustav A. Hedlund. Symbolic dynamics II. Sturmian trajectories. *American Journal of Mathematics*, 62(1):1–42, 1940.

Anastasios I. Mourikis and Stergios I. Roumeliotis. Optimal sensor scheduling for resource-constrained localization of mobile robot formations. *IEEE Transactions on Robotics*, 22 (5):917–931, 2006.

José Niño-Mora. Restless bandits, partial conservation laws and indexability. *Advances in Applied Probability*, 33(1):76–98, 2001.

José Niño-Mora. Dynamic allocation indices for restless projects and queueing admission control: a polyhedral approach. *Mathematical Programming, Series A*, 93(3):361–413, 2002.

José Niño-Mora. Restless bandit marginal productivity indices, diminishing returns, and optimal control of make-to-order/make-to-stock M/G/1 queues. *Mathematics of Operations Research*, 31(1):50–84, 2006.

José Niño-Mora. A $(2/3)n^3$ fast-pivoting algorithm for the Gittins index and optimal stopping of a Markov chain. *INFORMS Journal on Computing*, 19(4):596–606, 2007.

José Niño-Mora. A dynamic page-refresh index policy for web crawlers. In *Proceedings of the 19th International Conference on Analytical and Stochastic Modeling Techniques and Applications, LNCS 8499*, pages 46–60. Springer, 2014.

José Niño-Mora. A verification theorem for indexability of discrete time real state discounted restless bandits. *arXiv preprint arXiv:1512.04403*, 2015.

José Niño-Mora. Whittle's index policy for multi-target tracking with jamming and nondetections. In *International Conference on Analytical and Stochastic Modeling Techniques and Applications*, pages 210–222. Springer, 2016.

José Niño-Mora. A verification theorem for threshold-indexability of real-state discounted restless bandits. *Mathematics of Operations Research*, 2019. Forthcoming.

José Niño-Mora and Sofía Villar. Multi-target tracking via restless bandit marginal productivity indices and Kalman filter in discrete time. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 2905–2910, 2009.

Yaakov Oshman. Optimal sensor selection strategy for discrete-time state estimators. *IEEE Transactions on Aerospace and Electronic Systems*, 30(2):307–314, 1994.

Christos H. Papadimitriou and John N. Tsitsiklis. The complexity of optimal queuing network control. *Mathematics of Operations Research*, 24(2):293–305, 1999.

Charles Chapman Pugh. *Real Mathematical Analysis*. Springer, 2002.

Bhooshan Rajpathak, Harish Pillai, and Santanu Bandyopadhyay. Analysis of stable periodic orbits in the one dimensional linear piecewise-smooth discontinuous map. *Chaos*, 22 (3):033126, 2012.

Halsey L. Royden and Patrick M. Fitzpatrick. *Real Analysis*. Pearson, fourth edition, 2010.

Peter Joseph Seiler. *Coordinated control of unmanned aerial vehicles*. PhD thesis, University of California, Berkeley, 2001.

Richard F. Serfozo. Monotone optimal policies for Markov decision processes. In *Stochastic Systems: Modeling, Identification and Optimization, II*, pages 202–215. Springer, 1976.

J. George Shanthikumar and David D. Yao. Multiclass queueing systems: Polymatroidal structure and optimal scheduling control. *Operations Research*, 40(3-supplement-2):S293–S299, 1992.

Thorvald Thiele. *Sur la compensation de quelques erreurs quasi-systématiques par la méthode des moindres carrés*. CA Reitzel, 1880.

Fabio Tramontana, Frank Westerhoff, and Laura Gardini. On the complicated price dynamics of a simple one-dimensional discontinuous financial market model with heterogeneous interacting traders. *Journal of Economic Behavior and Organization*, 74(3):187–205, 2010.

Ina Verloop. Asymptotically optimal priority policies for indexable and nonindexable restless bandits. *Annals of Applied Probability*, 26(4):1947–1995, 2016.

Sofía Villar. *Restless bandit index policies for dynamic sensor scheduling optimization*. PhD thesis, Statistics Department, Universidad Carlos III de Madrid, 2012.

Robert B. Washburn. Application of multi-armed bandits to sensor management. In *Foundations and Applications of Sensor Management*, pages 153–175. Springer, 2008.

Richard Weber and Gideon Weiss. On an index policy for restless bandits. *Journal of Applied Probability*, pages 637–648, 1990.

Jaap Wessels. Markov programming by successive approximations with respect to weighted supremum norms. *Journal of Mathematical Analysis and Applications*, 58(2):326–335, 1977.

Peter Whittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, pages 287–298, 1988.

Wei Wu and Ari Arapostathis. Optimal sensor querying: general Markovian model and LQG models with controlled observations. *IEEE Transactions on Automatic Control*, 53 (6):1392–1405, 2008.

Lin Zhao, Wei Zhang, Jianghai Hu, Alessandro Abate, and Claire J. Tomlin. On the optimal solutions of the infinite-horizon linear sensor scheduling problem. *IEEE Transactions on Automatic Control*, 59(10):2825–2830, 2014.

Journal of Machine Learning Research 20 (2019) 1-37 Submitted 5/17; Revised 12/18; Published 2/19

A New Approach to Laplacian Solvers and Flow Problems

Patrick Rebeschini patrick.rebeschini@stats.ox.ac.uk
Department of Statistics
University of Oxford
Oxford, OX1 3LB, UK

Sekhar Tatikonda sekhar.tatikonda@yale.edu
Department of Statistics and Data Science
Yale University
New Haven, CT 06511, USA

Editor: Francis Bach

Abstract

This paper investigates the behavior of the Min-Sum message passing scheme to solve systems of linear equations in the Laplacian matrices of graphs and to compute electric flows. Voltage and flow problems involve the minimization of quadratic functions and are fundamental primitives that arise in several domains. Algorithms that have been proposed are typically centralized and involve multiple graph-theoretic constructions or sampling mechanisms that make them difficult to implement and analyze. On the other hand, message passing routines are distributed, simple, and easy to implement. In this paper we establish a framework to analyze Min-Sum to solve voltage and flow problems. We characterize the error committed by the algorithm on general weighted graphs in terms of hitting times of random walks defined on the computation trees that support the operations of the algorithms with time. For d-regular graphs with equal weights, we show that the convergence of the algorithms is controlled by the total variation distance between the distributions of non-backtracking random walks defined on the original graph that start from neighboring nodes. The framework that we introduce extends the analysis of Min-Sum to settings where the contraction arguments previously considered in the literature (based on the assumption of walk summability or scaled diagonal dominance) can not be used, possibly in the presence of constraints.

Keywords: Min-Sum, message passing, decentralized algorithms, Laplacian solver, network flows.

1. Introduction

Voltage and flow problems. Laplacian matrices of graphs are fundamental. Solving linear systems of equations in these matrices constitutes the back-bone of many algorithms with applications to a wide variety of fields. See Spielman (2011); Vishnoi (2013). When a weighted graph is viewed as an electric circuit, where weighted edges represent resistors and vertices represent nodes to which some external current is applied, solving linear systems of equations in the Laplacian matrix of the graph corresponds to computing the voltages induced to each node by the external currents. For this reason, we refer to the Laplacian problem as the "voltage problem." Along with the problem of computing the voltages is the problem of computing the current induced across each resistor, which we refer to as the "flow problem." The two problems are related by Ohm's law—which says that electric flows are induced by voltage differences across the nodes—and they both can be cast in the form of quadratic optimization problems. The voltage problem is an unconstrained problem to find the voltage assignment that maximizes the quadratic Laplacian form. The flow problem is a constrained problem to find the flow assignment that minimizes the circuit energy and satisfies Kirchhoff's law. The voltage problem is the dual of the flow problem.

Previous literature on Laplacian solvers. The extensive literature on approximately solving the voltage and flow problems typically involves algorithms that rely on multiple graph-theoretic constructions or sampling mechanisms which make them difficult to implement and analyze. The first quasi-linear time solver was given in Spielman and Teng (2004, 2014), and it achieves a running time of $\tilde{O}(m \log^c n \log 1/\varepsilon)$ in the Laplacian-modified ℓ_2 norm. Here, n and m are, respectively, the number of vertices and edges in the graph, c is a constant close to 100, and the $\tilde{O}$ notation hides $\log \log n$ factors. Since this result, many algorithms have been proposed and showed to yield smaller values of c. The work in Koutis et al. (2011) has reduced the constant c to 1. Currently, the best running time is in Cohen et al. (2014) where $c = 1/2$. These algorithms all rely on the same general architecture, which combines the Chebyshev method or the conjugate gradient algorithm with the use of recursive preconditioning, spectral sparsifiers, low-stretch spanning trees, and expander graphs. Despite their theoretical guarantees, these algorithms present noticeable difficulties to practitioners due to the involved machinery they rely upon. Lately, new algorithms have been proposed that depart from this framework and use fewer graph-theoretic constructions. The method in Kelner et al. (2013), for instance, which operates in the flow domain and yields a fast solver with $c = 2$, using very little of the previous machinery—although it still requires low-stretch spanning trees. More recently, the work in Kyng and Sachdeva (2016), which relies on sparse Gaussian elimination for Laplacian matrices to yield a fast solver with $c = 3$, without the use of any graph-theoretic construction. While more appealing, these new algorithms still rely on random sampling so the convergence analysis is only performed in expectation, or with high probability. Moreover, all these algorithms are centralized. There has also been some work in parallelizing these solvers, with the first parallel solver with near linear work and poly-logarithmic depth presented in Peng and Spielman (2014). These solvers work in the shared memory model and are not completely distributed.

Min-Sum algorithm. The aforementioned limitations prompted us to investigate the behavior of the Min-Sum algorithm to solve both the voltage and the flow problem. Min-Sum is a distributed, deterministic (i.e., not randomized), general-purpose message passing algorithm that is used to optimize objective functions that are sums of component functions supported on a given graph. Min-Sum has emerged as a canonical procedure to address large scale problems in a variety of domains. Its success comes from the fact that the algorithm is simple and easy to implement, require minimal data structures and low communication per iteration, and in many applications it has been shown to converge to the problem solution. Interest in Min-Sum has been triggered by its success in solving certain classes of challenging combinatorial optimization problems, such as decoding low-density parity-check codes, see Berrou et al. (1993); Richardson and Urbanke (2001) for instance, and solving certain type of satisfiability problems, see Mezard et al. (2002); Braunstein et al. (2005), for instance.

Despite its successes, the convergence analysis of the Min-Sum algorithm remains limited. For quadratic optimization problems and, more generally, convex optimization prob-

©2019 Patrick Rebeschini and Sekhar Tatikonda.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-286.html.

lems, the Min-Sum algorithm has been shown to converge to the problem solution for *any* graph topology under the assumption that the objective function is walk-summable (Malioutov et al., 2006), or, equivalently, scaled diagonal dominant (Moallemi and Van Roy, 2010). These conditions allow the use of contraction arguments to establish exponential convergence rates. These conditions do not apply to quadratic objective functions in Laplacian matrices, hence to the voltage problem, and they do not apply to constrained optimization problems, hence to the flow problem. At the same time, on the one hand quadratic approximations of the Min-Sum algorithm leading to the Approximate Message Passing (AMP) algorithm are becoming increasingly popular in high-dimensional statistics and machine learning (Donoho et al., 2009; Montanari, 2012; Barbier et al., 2018). The AMP algorithm is typically analysed from an asymptotical point of view and this analysis is *not* based on contraction arguments. The recent success of the AMP algorithm prompts for improving the understanding of the convergence and correctness of the Min-Sum algorithm directly applied to quadratic models, where previously-considered contraction-based assumptions do not hold. On the other hand, while some work has been done to study the behavior of Min-Sum in constrained settings, this work seems to have focused on linear objective functions and a general theory is missing. See Gamarnik et al. (2012) and references therein.

Results. This paper investigates the convergence behavior of the Min-Sum algorithm applied to the voltage and flow problems and establishes a general framework for the analysis of message-passing algorithms that can accommodate for non contraction-based assumptions and for constrained-based optimization problems. The algorithms that we analyze—Algorithm 3 for the voltage problem and Algorithm 5 for the flow problem—are easy to implement, with a pseudo-code that fits two lines. We investigate the behavior of these algorithms on general weighted graphs, and we show that the error they incur can be characterized in terms of hitting times of ordinary diffusion random walks defined on the computation trees that are obtained by unraveling the operations of the algorithms with time (Weiss and Freeman, 2001). We specialize the analysis to the case of d-regular graphs. When $d = 2$, we show that the flow algorithm converges to the true solution in the ℓ_∞ norm with rate $O(1/t)$, where t is the iteration time, independently of the graph dimensions n and m, yielding a ε-approximate solution with a running time $O(m/\varepsilon)$, linear in m (Corollary 16). On the other hand, when $d = 2$, we show that the voltage algorithm does *not* converge, as it keeps oscillating (Corollary 10). For graphs with equal weights, when $d \geq 3$, we show that the error both algorithms commit is characterized by quantities involving the difference of non-backtracking random walks on the original graph that originate from neighbor nodes (Lemma 9). We show that the convergence rate of the algorithms (with respect to various norms) can be controlled by uniform bounds on the total variation distance between the distributions of these walks. For graphs where these bounds decay exponentially $O(e^{-t})$ (such as complete graphs, as we show numerically) or polynomially $O(1/t^\beta)$ with $\beta > 0$ (such as cycles and tori, where $\beta = 1/2$ as we show numerically), possibly under additional assumptions on the problem instance (depending on the norm chosen for the analysis), the Min-Sum algorithm yield ε-approximate solutions for both voltage and flow problem with time $O(m \log 1/\varepsilon)$ or $O(m/\varepsilon^{1/\beta})$, respectively, linear in m (Theorem 15 for voltages, and Theorem 18 for flows).

Novelty in the analysis of message passing. Our analysis of the Min-Sum algorithm presents two key contributions that go beyond the specifics of the problems here addressed.

1. Beyond contraction arguments. Typically, the convergence analysis of the Min-Sum scheme exploits a decay of correlation property in the computation tree established upon imposing *local* assumptions on the original problem, such as the assumption of scaled diagonal dominance. These conditions essentially yield a bound on the local interactions in the computation tree by a quantity strictly less than one. In this case, a contraction argument can be performed at each level of the tree to establish an *exponential* decay of correlation from root to leaves, which translates into an exponential convergence for the algorithm. On the other hand, the analysis of message passing that we present in this paper is based on a *global* analysis of the problem supported on the computation tree, via the analogy with electric circuits and the characterization with hitting times of random walks, which is not captured by contraction arguments based on worst-case bounds for local interactions. In fact, as mentioned above, for some graphs this analysis yields a convergence *polynomial* in time, not exponential.
2. Beyond unconstrained optimization. Typically, the convergence analysis of the Min-Sum algorithm relies on finding the fix point(s) of the message updates. For unconstrained optimization problems, the key insight to find fix point(s) is that the first order optimality conditions for the original problem are respected by the problem supported on the computation tree, for every node that is not a leaf (Weiss and Freeman, 2001; Moallemi and Van Roy, 2010). The convergence analysis that we present for the flow problem—a constrained problem—is based on the observation that the Karush-Kuhn-Tucker (KKT) optimality conditions for constrained problems are respected on the computation tree, for every non-leaf node. This insight allows to extend the analysis of the Min-Sum algorithm to more general constrained optimization problems.

Structure of the paper. The structure of the paper is as follows. In Section 2 we define the voltage and flow problems as optimization routines. In Section 3 we introduce the general-purpose Min-Sum algorithm, and we specialize it to the voltage and flow problem, respectively. In Section 4 we present the error characterization for the Min-Sum algorithms for d-regular graphs, and we present the main results on their convergence. In Section 5 we develop the general framework to investigate the Min-Sum algorithm, yielding the characterization of the error in terms of hitting times of random walks on the computation trees, and we present the proofs of the results given in the previous section. In Appendix A we establish a general connection between the inverse of restricted Laplacian matrices and hitting times of ordinary diffusion random walks on graphs—which is the main computational tool used in our analysis. Technical results are in Appendix B.

Remark 1 (Notation) *Given a matrix M, we denote by M^T its transpose, by M^{-1} its inverse, and by M^+ its Moore-Penrose pseudoinverse. Given a set $\mathcal{I}$ and $i \in \mathcal{I}$, we denote by i both the element $i \in \mathcal{I}$ and the subset $\{i\} \subseteq \mathcal{I}$ (e.g., we write $\mathcal{I}\backslash i$ to mean $\mathcal{I}\backslash\{i\}$). Given a function $f : \mathbb{R}^{\mathcal{I}} \to \mathbb{R}$, $i \in \mathcal{I}$, we write $f(x_{\mathcal{I}\backslash i}\,\cdot)$ to mean the function $z \in \mathbb{R} \to f(x_{\mathcal{I}\backslash i} z)$. We adopt the following taxonomy of notation:*

- *the arrow notation $\vec{}$ is used to either refer to quantities related to directed graphs or to refer to probability distributions related to non-backtracking random walks;*
- *the hat notation $\hat{}$ is used to refer to the output of algorithms;*
- *the bar notation $\bar{}$ is for quantities related to graphs where a node has been removed;*
- *the double-struck notation is for quantities related to computation trees, cf. Section 5.*

2. Voltage and Flow Problems

Throughout, let us consider a simple (i.e., no self-loops, and no multiple edges), undirected, weighted graph G given by the triple (V, E, W), where V is a set of n vertices, E is a set of m edges, and $W \in \mathbb{R}^{V\times V}$ is a symmetric matrix that assigns a positive weight to every edge, namely, $W_{vw} > 0$ if $\{v, w\} \in E$, and $W_{vw} = 0$ if $\{v, w\} \notin E$. The matrix W is called the weighted *adjacency matrix* of the weighed graph G. Define the weighted degree of a vertex v by $d_v := \sum_{w\in V} W_{vw}$, and let $D \in \mathbb{R}^{V\times V}$ be the diagonal matrix defined by $D_{vv} := d_v$. The Laplacian L of the graph G is the matrix defined as $L := D - W$. Without loss of generality, in what follows we assume that the graph G is connected, otherwise we can treat each connected component on its own. It can be verified that in this case the null space of the Laplacian matrix L is spanned by the all-ones vector $1 \in \mathbb{R}^V$, so that the range of L is spanned by the vectors orthogonal to 1.

Voltage problem. We are interested in solving linear equations in the Laplacian matrix L. That is, given b in the range of L, i.e., $1^T b = 0$, our goal is to find ν that satisfies

$$L\nu = b. \tag{1}$$

As it can immediately be checked from first order optimality conditions, the set of solutions to (1) coincides with the set of solutions to the "voltage problem," namely:

$$\text{maximize} \quad -\frac{1}{2}\nu^T L\nu + b^T\nu. \tag{2}$$

We are interested in the unique solution that is orthogonal to the all one vector, namely, $\nu^\star = L^+ b$, where L^+ is the Moore-Penrose pseudoinverse of L. Given an accuracy parameter $\varepsilon > 0$, the fast solvers mentioned in the introduction achieve a quasi-linear running time $\tilde{O}(m \log^c n \log 1/\varepsilon)$ to return an estimate $\hat{\nu}$ that satisfies $\|\nu^\star - \hat{\nu}\|_L \le \varepsilon\|\nu^\star\|_L$, where $\|\nu\|_L := \sqrt{\nu^T L\nu}$ is the Laplacian-modified ℓ_2 norm, also known as L-norm. It can be seen that $\|\cdot\|_L$ is a pseudo-norm, as $\|\nu\|_L = 0$ for any vector ν orthogonal to the all one vector. In many applications where Laplacian solvers are needed, the L-norm is the natural norm to run the analysis. See Vishnoi (2013), for instance. The L-norm is related to the Euclidean norm as follow: $\|\nu\|_L \le \kappa\|\nu\|_2$ and $\|\nu\|_2 \le \kappa\|\nu\|_L$, where κ is the ratio between the the maximum and minimum non-zero eigenvalues of L.

Flow problem. To introduce the flow problem, let us fix an arbitrary orientation of the edges in E. Denote this set of directed edges by $\vec{E}$, and let $\vec{G} = (V, \vec{E})$ be the corresponding directed graph. Let $A \in \mathbb{R}^{V\times\vec{E}}$ be the *signed vertex-edge incidence* matrix of the graph $\vec{G}$, defined as $A_{ve} := 1$ if edge e leaves node v, $A_{ve} := -1$ if edge e enters node v, $A_{ve} := 0$ otherwise. To each directed edge $e = (v, w)$ let $R \in \mathbb{R}^{\vec{E}\times\vec{E}}$ be the diagonal matrix defined by $R_{ee} := 1/W_{vw}$. It is easy to verify that the Laplacian L of the undirected graph G can be expressed as $L = AR^{-1}A^T$. Note that while A depends on the choice of the direction of the edges in $\vec{E}$, L does not. The "flow problem" is defined as follows:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}x^T Rx \\ \text{subject to} \quad & Ax = b. \end{aligned} \tag{3}$$

This problem can be interpreted as a minimal-energy electrical network problem. To each directed edge $e \in \vec{E}$ is associated a resistor of value R_{ee} (analogously, a conductor of value $1/R_{ee}$) through which a current x_e flows, with the convention that $x_e > 0$ if the current is in the direction of the edge, $x_e < 0$ if the current is in the direction opposite the edge. The energy of having current x_e on edge e is given by the $\frac{1}{2}R_{ee}x_e^2$. For each node $v \in V$, b_v represents a given external current: $b_v > 0$ represents a source where the current enters node v, whereas $b_v < 0$ represents a sink where the current exits node v. The constraint equations $Ax = b$ embody Kirchhoff's conservation law, by which at each vertex the net sum of the incoming and outgoing internal flows equals the external flow.

Connection between voltage and flow problems. From the KKT optimality conditions it can be shown (see Proposition 31 in Appendix B) that the unique solution of problem (3) reads $x^\star = R^{-1}A^T L^+ b = R^{-1}A^T\nu^\star$. That is, if $e = (v, w) \in \vec{E}$, we have $x_e^\star = W_{vw}(\nu_v^\star - \nu_w^\star)$ which represents Ohm's law. Finally, notice that problem (2) is the dual of (3). Recall that the Lagrangian of (3) is the function $\mathcal{L}$ from $\mathbb{R}^{\vec{E}} \times \mathbb{R}^V$ to $\mathbb{R}$ defined as

$$\mathcal{L}(x, \nu) := \frac{1}{2}x^T Rx + \nu^T(b - Ax). \tag{4}$$

The dual function is $q(\nu) = \min_{x\in\mathbb{R}^{\vec{E}}} \mathcal{L}(x, \nu) = -\frac{1}{2}\nu^T L\nu + b^T\nu$, and the dual problem reads $\max_{\nu\in\mathbb{R}^V} q(\nu)$, which is problem (2).

3. Min-Sum Algorithm

The Min-Sum algorithm is a distributed routine to optimize a cost function that has a graph structure. Let $\mathcal{V}$ and $\mathcal{F}$ be two finite sets, whose elements are respectively referred to as *variables* and *factors*, to be interpreted as two groups of vertices in a certain undirected bipartite graph with edge set $\mathcal{E}$, where each edge in $\mathcal{E}$ connects an element of $\mathcal{V}$ with an element of $\mathcal{F}$. For any variable node $i \in \mathcal{V}$, we use the notation ∂i to denote the set of neighbors of vertex i in the factor graph. Analogously, for any factor node $a \in \mathcal{F}$, we use the notation ∂a to denote the set of neighbors of vertex a in the factor graph. Consider the following optimization problem over $x \in \mathbb{R}^{\mathcal{V}}$:

$$\text{minimize} \quad g(x) := \sum_{i\in\mathcal{V}} g_i(x_i) + \sum_{a\in\mathcal{F}} g_a(x_{\partial a}), \tag{5}$$

for a given set of functions $g_i : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$, for each $i \in \mathcal{V}$, and $g_a : \mathbb{R}^{\partial a} \to \mathbb{R} \cup \{\infty\}$, for each $a \in \mathcal{F}$. The Min-Sum algorithm is an iterative algorithm that proceeds by updating a set of *messages* associated to each edge in the underlying factor graph. Messages are functions of the optimization variables x_i, for $i \in \mathcal{V}$, and each edge in the factor graph is associated with two messages at any given iteration of the algorithm. For any time $s \ge 0$, and $i \in \mathcal{V}, a \in \mathcal{F}$ so that $\{i, a\} \in \mathcal{E}$, $\mu_{i\to a}^s : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$ represents a message from variable to factor, and $\mu_{a\to i}^s : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$ represents a message from factor to variable.

In this work we consider the *synchronous* implementation of the Min-Sum algorithm, where at each iteration all messages are updated. For a given iteration step $t \ge 1$, the algorithm reads as in Algorithm 1.

The iterations of the Min-Sum algorithm correspond to the dynamic programming iterations to solve (5) when the underlying graph is a tree. In general, the Min-Sum algorithm

Algorithm 1: Min-Sum

Input: Initial messages $\{\mu^0_{a\to i}\}$, for each $\{i,a\}\in\mathcal{E}$;
for $s\in\{1,\dots,t\}$ **do**
  Compute, for each $\{i,a\}\in\mathcal{E}$,
  $\mu^s_{i\to a}=g_i+\sum_{b\in\partial i\setminus a}\mu^{s-1}_{b\to i}$, $\quad\mu^s_{a\to i}=\min_{x\in\mathbb{R}^\nu}\{g_a(x_{\partial a\setminus i}\,\cdot)+\sum_{j\in\partial a\setminus i}\mu^s_{j\to a}(x_j)\}$;
Compute $\mu^t_i=g_i+\sum_{a\in\partial i}\mu^t_{a\to i}$, for each $i\in\mathcal{V}$;
Output: $\hat{x}^t_i:=\arg\min_{z\in\mathbb{R}}\mu^t_i(z)$, for each $i\in\mathcal{V}$.

is not guaranteed to converge, and even when it converges it is not guaranteed that it converges to an optimal point of problem (5). The Min-Sum algorithm is suited for cooperative distributed multi-agents optimization. In this framework, to each variable $i\in\mathcal{V}$ is associated an agent who has access only to the local objective functions g_i and g_a, for each $a\in\partial i$. At each time step, each agent can only exchange information with his near-neighbors in the factor graph, i.e., the agents who share at least one factor with him. Agents' goal is to cooperate over time to solve problem (5). The Min-Sum algorithm is a mean to achieve this goal: iteratively, each agent i updates a set of messages $\{\mu^s_{i\to a}\}, a\in\partial i$, using the messages that he has received from his neighbors at the previous time step, i.e., $\{\mu^{s-1}_{j\to a}\}, a\in\partial j$, for each neighbor j of i, and using the locally known functions g_i and g_a, for each $a\in\partial i$.

Remark 2 (Belief Propagation) *The Min-Sum algorithm alternates minimizations and summations to compute the minimum of a sum of functions of the form in* (5)*. The main property that is used to derive the local updates of this algorithm (on a tree, where the algorithm computes the correct solution) is the fact that the minimum distributes over the summation. Closely related to the Min-Sum algorithm is the Sum-Product algorithm, where the goal is to compute the sum (or integral) of a product of functions of the following form:*

$$\prod_{i\in\mathcal{V}} g_i(x_i)\prod_{a\in\mathcal{F}} g_a(x_{\partial a}). \tag{6}$$

As the sum (or integral) distributes over the product, the Sum-Product algorithm takes the exact same form as Algorithm 1 with minima replaced by sums and sums replaced by products. In the machine learning community, the Sum-Product algorithm is usually referred to as Belief Propagation, and a classical application involves the computation of the normalization function of graphical models, where random variables $\{X_v\}_{v\in V}$ have a joint density that factorizes as in (6)*. While Min-Sum and Sum-Product are formally related, they are known to produce results that are qualitatively different. In particular, while there are sufficient conditions that guarantee the convergence and correctness of Min-Sum in graphs of arbitrary topology, the Sum-Product algorithm is known to yield correct results only on locally tree-like graphs (Weiss and Freeman, 2001).*

We now specialize the Min-Sum algorithm to the voltage and flow problem, respectively.

3.1. Min-Sum for the Voltage Problem

In order to apply the Min-Sum algorithm to solve the voltage problem (2), throughout the rest of this paper we consider the equivalent minimization form: minimize $\frac{1}{2}\nu^TL\nu-b^T\nu$. We

rewrite this problem in the form (5).[1] To this end, let us identify $\mathcal{V}\leftrightarrow V$ and $\mathcal{F}\leftrightarrow E$. In this context, the notation ∂v indicates the set of edges in E are connected to $v\in V$; analogously, the notation ∂e indicates the pair of vertices that are connected by edge e. For each $v\in V$, define the function $g_v(\nu_v)=-b_v\nu_v$, and for each $e\in E$ define the function $g_e(\nu_{\partial e}):=\frac{1}{2}W_{vw}(\nu_v-\nu_w)^2$, where $\partial e=\{v,w\}$. Recalling that $\nu^TL\nu=\sum_{\{v,w\}\in E}W_{vw}(\nu_v-\nu_w)^2$, the problem above is equivalent to the problem of minimizing the function $\sum_{v\in V}g_v(\nu_v)+\sum_{e\in E}g_e(\nu_{\partial e})$, which is of the form (5). As in the factor graph interpretation each factor node (i.e., edge in E) is adjacent to exactly two variable nodes (i.e., vertices in V), it is convenient to rewrite the Min-Sum algorithm in terms of updates of messages from factors to variables only. At each time step $t\geq 1$, to each edge $e=\{v,w\}\in E$ are associated two messages: $\mu^t_{e\to v}$ and $\mu^t_{e\to w}$. It is easy to verify that in the present setting Algorithm 1 becomes Algorithm 2.

Algorithm 2: Min-Sum, voltage problem

Input: Initial messages $\{\mu^0_{e\to v}\}$, for $e\in E, v\in\partial e$;
for $s\in\{1,\dots,t\}$ **do**
  Compute, for each $e\in E$, $v\in\partial e$, with $w=\partial e\setminus v$
  $\mu^s_{e\to v}=\min_{\nu_w\in\mathbb{R}}\{g_w(\nu_w)+g_e(\nu_w\,\cdot)+\sum_{f\in\partial w\setminus e}\mu^{s-1}_{f\to w}(\nu_w)\}$;
Compute $\mu^t_v=g_v+\sum_{e\in\partial v}\mu^t_{e\to v}$, for each $v\in V$;
Output: $\hat{\nu}^t_v:=\arg\min_{z\in\mathbb{R}}\mu^t_v(z)$, for each $v\in V$.

Algorithm 2 applies for any choice of the initial messages $\{\mu^0_{e\to v}\}$, for $e\in E, v\in\partial e$. If we choose initial messages that are quadratic functions, then the messages computed by the algorithm remain quadratic. The following Proposition presents the quadratic updates, without stating formulas for the constant terms as these terms do not influence the output of the Min-Sum algorithm, as we will show next.

Proposition 3 *For a given $s\geq 1$, for each $e\in E$, $v\in\partial e$ let $\mu^{s-1}_{e\to v}(z)=\frac{1}{2}W^{s-1}_{e\to v}z^2+w^{s-1}_{e\to v}z+c^{s-1}_{e\to v}$. Then, for each $e\in E$, $v\in\partial e$, we have $\mu^s_{e\to v}(z)=\frac{1}{2}W^s_{e\to v}z^2+w^s_{e\to v}z+c^s_{e\to v}$, where $\{c^s_{e\to v}\}$ is a certain set of constants, and (here $w=\partial e\setminus v$)*

$$W^s_{e\to v}=W_{vw}\frac{\sum_{f\in\partial w\setminus e}W^{s-1}_{f\to w}}{W_{vw}+\sum_{f\in\partial w\setminus e}W^{s-1}_{f\to w}},\qquad w^s_{e\to v}=W_{vw}\frac{\sum_{f\in\partial w\setminus e}w^{s-1}_{f\to w}-b_w}{W_{vw}+\sum_{f\in\partial w\setminus e}W^{s-1}_{f\to w}}.$$

Proof Fix $s\geq 1$, $e\in E$, $v\in\partial e$, and let $w=\partial e\setminus v$. The Min-Sum update rule reads $\mu^s_{e\to v}(\nu_v)=\min_{z\in\mathbb{R}}\{-b_wz+\frac{1}{2}W_{vw}(\nu_v-z)^2+\frac{1}{2}z^2\sum_{f\in\partial w\setminus e}W^{s-1}_{f\to w}+z\sum_{f\in\partial w\setminus e}w^{s-1}_{f\to w}+\sum_{f\in\partial w\setminus e}c^{s-1}_{f\to w}\}$. The solution is $z^\star=(W_{vw}\nu_v+b_w-\sum_{f\in\partial w\setminus e}w^{s-1}_{f\to w})/(W_{vw}+\sum_{f\in\partial w\setminus e}W^{s-1}_{f\to w})$, and the proof is immediately concluded by substitution. ■

From Proposition 3 it follows that if we initialize Algorithm 2 with quadratic messages of the form $\mu^0_{e\to v}(z)=\frac{1}{2}W^0_{e\to v}z^2+w^0_{e\to v}z$, then, modulo the constant terms, we can track the evolution of the quadratic messages computed by the algorithm by evaluating recursively

1. It is well known that different ways of mapping the original optimization problem to the form (5) yield different convergence results. See Moallemi and Roy (2009), for instance.

the set of parameters $\{W^s_{e\to v}\}, \{w^s_{e\to v}\}$ as prescribed by Proposition 3. The belief function associated to vertex v at time t then reads $\mu^t_v(z) = -b_v z + \sum_{e\in\partial v}(\frac{1}{2}W^t_{e\to v}z^2 + w^t_{e\to v}z + c^t_{e\to v})$, so that $\hat{\nu}^t_v = \arg\min_{z\in\mathbb{R}} \mu^t_v(z) = (b_v - \sum_{e\in\partial v} w^t_{e\to v})/(\sum_{e\in\partial v} W^t_{e\to v})$, independent of $\{c^t_{e\to v}\}$. The final procedure, in the case of quadratic initial messages, yields Algorithm 3.

Algorithm 3: Min-Sum, voltage problem, quadratic messages

Input: Initial messages $\{W^0_{e\to v}\}, \{w^0_{e\to v}\}$, $e \in E, v \in \partial e$;
for $s \in \{1, \dots, t\}$ **do**
  Compute, for each $e \in E$, $v \in \partial e$, with $w = \partial e \setminus v$
  $W^s_{e\to v} = W_{vw}\frac{\sum_{f\in\partial w\setminus e} W^{s-1}_{f\to w}}{W_{vw}+\sum_{f\in\partial w\setminus e} W^{s-1}_{f\to w}}$, $\quad w^s_{e\to v} = -W_{vw}\frac{b_w - \sum_{f\in\partial w\setminus e} w^{s-1}_{f\to w}}{W_{vw}+\sum_{f\in\partial w\setminus e} W^{s-1}_{f\to w}}$;
Output: $\hat{\nu}^t_v = \frac{b_v - \sum_{e\in\partial v} w^t_{e\to v}}{\sum_{e\in\partial v} W^t_{e\to v}}$, for each $v \in V$.

Algorithm 3, with the choice of initial conditions $W^0_{e\to v} = W_{vw}$ and $w^0_{e\to v} = 0$, for each $e = \{v, w\} \in E, v \in \partial e$, will be the focus of our investigation for the voltage problem. The convergence analysis will rely on the general formulation of Algorithm 2.

Remark 4 (Gaussian Belief Propagation) *When applied to quadratic objective functions, the Min-Sum algorithm (e.g., Algorithm 3) corresponds to the computation of the mean performed by the Belief Propagation algorithm applied to Gaussian graphical models (Weiss and Freeman, 2001). This connection is due to the fact that in the Gaussian case the mode of a distribution coincides with its mean. Min-Sum can be seen as computing the maximum a posteriori (MAP) estimate of the graphical model (i.e., the mode), which corresponds to the mean computed by Belief Propagation. Typically, the convergence and correctness of the mean of Gaussian Belief Propagation is investigated within the framework of walk-summability (Malioutov et al., 2006), or, equivalently, scaled diagonal dominance (Moallemi and Van Roy, 2010). As the Laplacian L is not positive definite, the existing framework can not be applied to investigate the convergence behavior of Algorithm 3. In this paper, we overcome this limitation by using a different approach based on the global analysis of the computation tree that supports the operations of the algorithm with time.*

3.2. Min-Sum for the Flow Problem

To apply the Min-Sum algorithm to solve the flow problem (3), we first rewrite this problem in the unconstrained form (5). To this end, let us identify $\mathcal{V} \leftrightarrow \vec{E}$ and $\mathcal{F} \leftrightarrow V$. In this context, the notation ∂v indicates the set of edges in $\vec{E}$ that either enter or leave node $v \in V$; analogously, the notation ∂e indicates the pair of vertices that are connected by edge e. For each $e \in \vec{E}$, define the function $g_e(x_e) = \frac{1}{2}R_{ee}x_e^2$, and for each $v \in V$ define the function $g_v(x_{\partial v}) := 0$ if $\sum_{e\in\vec{E}} A_{ve}x_e = b_v$, and $g_v(x_{\partial v}) := \infty$ otherwise. Clearly, problem (3) is equivalent to the problem of minimizing the function $\sum_{e\in\vec{E}} g_e(x_e) + \sum_{v\in V} g_v(x_{\partial v})$, which is of the form (5). As in the factor graph interpretation each variable node (i.e., edge in $\vec{E}$) is adjacent to exactly two factor nodes (i.e., vertices in V), it is convenient to rewrite the Min-Sum algorithm in terms of updates of messages from variables to factors only. At each time step $t \geq 1$, to each edge $e \in \vec{E}$ such that $\partial e = \{v, w\}$ (that is, either $e = (v, w)$ or $e = (w, v)$) are associated two messages: $\mu^t_{e\to v}$ and $\mu^t_{e\to w}$. It can be verified that in the present setting Algorithm 1 becomes Algorithm 4.[2]

Algorithm 4: Min-Sum, flow problem

Input: Initial messages $\{\mu^0_{e\to v}\}$, for $e \in \vec{E}, v \in \partial e$;
for $s \in \{1, \dots, t\}$ **do**
  Compute, for each $e \in \vec{E}$, $v \in \partial e$, with $w = \partial e \setminus v$
  $\mu^s_{e\to v} = g_e + \min_{x\in\mathbb{R}^{\vec{E}}} \left\{ g_w(x_{\partial w\setminus e}\,\cdot) + \sum_{f\in\partial w\setminus e} \mu^{s-1}_{f\to w}(x_f) \right\}$;
Compute $\mu^t_e = \mu^t_{e\to v} + \mu^t_{e\to w} - g_e$, for $e = (v, w) \in \vec{E}$;
Output: $\hat{x}^t_e := \arg\min_{z\in\mathbb{R}} \mu^t_e(z)$, for each $e \in \vec{E}$.

Note that if $\partial w \setminus e = \varnothing$, then the message $\mu^s_{e\to v}$ in Algorithm 4 reads $\mu^s_{e\to v}(z) = \frac{1}{2}R_{ee}b_w^2$ if $z = A_{we}b_w$, and $\mu^s_{e\to v}(z) = \infty$ otherwise, for any $s \geq 1$, irrespective of the choice of the initial messages. This is a consequence of the fact that if node $w \in V$ is a *leaf* of G (i.e., w has degree 1), then by Kirchhoff's law the value x_e of the flow on the unique edge e that is associated to w is $x_e = A_{we}b_w$. Without loss of generality, in what follows we assume that the graph G has no leaves. Otherwise, by removing one leaf at a time as described above, we can reduce the original problem to a new problem supported on a graph with no leaves (if G is a tree we can exactly compute all the components of the solution $x^\star$ in this way).

Algorithm 4 applies for any choice of the initial messages $\{\mu^0_{e\to v}\}$, for $e \in \vec{E}, v \in \partial e$. If we choose initial messages that are quadratic functions, then also the following messages computed by the algorithm remain quadratic. The next Lemma presents the quadratic updates, without stating formulas for the constant terms as these terms do not influence the output of the Min-Sum algorithm, as we will show next.

Proposition 5 *For a given $s \geq 1$, for each $e \in \vec{E}$, $v \in \partial e$ let $\mu^{s-1}_{e\to v}(z) = \frac{1}{2}R^{s-1}_{e\to v}z^2 + r^{s-1}_{e\to v}z + c^{s-1}_{e\to v}$. Then, if the graph G has no leaves, for each $e \in \vec{E}$, $v \in \partial e$, we have $\mu^s_{e\to v}(z) = \frac{1}{2}R^s_{e\to v}z^2 + r^s_{e\to v}z + c^s_{e\to v}$, where $\{c^s_{e\to v}\}$ is a set of constants, and (here $w = \partial e \setminus v$)*

$$R^s_{e\to v} = R_{ee} + \frac{1}{\sum_{f\in\partial w\setminus e} 1/R^{s-1}_{f\to w}}, \qquad r^s_{e\to v} = -A_{we}\frac{\sum_{f\in\partial w\setminus e} A_{wf}r^{s-1}_{f\to w}/R^{s-1}_{f\to w} + b_w}{\sum_{f\in\partial w\setminus e} 1/R^{s-1}_{f\to w}}.$$

Proof Fix $s \geq 1$, $e \in \vec{E}$, $v \in \partial e$, and let $w \in \partial e \setminus v$. As we assume that the graph G has no leaves, then $\partial w \setminus e \neq \varnothing$ and the Min-Sum update rule reads $\mu^s_{e\to v}(z) = g_e(z) + \min_{x\in\mathbb{R}^{\partial w\setminus e}: \tilde{A}x = b_w - A_{we}z} \sum_{f\in\partial w\setminus e} \mu^{s-1}_{f\to w}(x_f)$, where $\tilde{A} := A_{w,\partial w\setminus e}$ is the submatrix of A given by the w-th row and the columns indexed by the elements in $\partial w \setminus e$. The minimization in the definition of $\mu^s_{e\to v}$ can be written as

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}x^T\tilde{R}x + \tilde{r}^T x + 1^T\tilde{c} \\ \text{subject to} \quad & \tilde{A}x = b_w - A_{we}z \end{aligned} \tag{7}$$

2. For convenience of notation, we present Algorithm 4 with initial messages indexed by time 0. Messages are indexed by a time parameter that is shifted by one unit compared to Algorithm 1.

over $x \in \mathbb{R}^{\partial w \setminus e}$, with $\tilde{R} \in \mathbb{R}^{(\partial w \setminus e) \times (\partial w \setminus e)}$ diagonal with $\tilde{R}_{ff} := R^{s-1}_{f \to w}$, $\tilde{r} \in \mathbb{R}^{\partial w \setminus e}$ with $\tilde{r}_f := r^{s-1}_{f \to w}$, and $\tilde{c} \in \mathbb{R}^{\partial w \setminus e}$ with $\tilde{c}_f := c^{s-1}_{f \to w}$. By Proposition 31 in Appendix B the solution of (7) reads $x^\star = h - A_{we} z y$, with $h := b_w y - (I - y\tilde{A})\tilde{R}^{-1}\tilde{r}$, $y := \tilde{R}^{-1}\tilde{A}^T \tilde{L}^{-1}$, and $\tilde{L} := \tilde{A}\tilde{R}^{-1}\tilde{A}^T \in \mathbb{R}$. Substituting $x^\star$ in (7) we get $\mu^s_{e \to v}(z) = \frac{1}{2} R^s_{e \to v} z^2 + r^s_{e \to v} z + c^s_{e \to v}$, with $R^s_{e \to v} := R_{ee} + y^T \tilde{R} y$, and $r^s_{e \to v} := -A_{we}(y^T \tilde{R} h + \tilde{r}^T y)$. The proof follows as $\tilde{L} = \sum_{f \in \partial w \setminus e} 1/R^{s-1}_{f \to w}$, $y^T \tilde{R} y = \tilde{L}^{-1}$, $y^T \tilde{R} h = \tilde{L}^{-1} b_w$, $\tilde{r}^T y = \tilde{L}^{-1} \sum_{f \in \partial w \setminus e} A_{wf} r^{s-1}_{f \to w} / R^{s-1}_{f \to w}$. ■

From Proposition 5 it follows that if we initialize Algorithm 4 with quadratic messages $\mu^0_{e \to v}(z) = \frac{1}{2} R^0_{e \to v} z^2 + r^0_{e \to v} z$, then, modulo the constant terms, we can track the evolution of the quadratic messages computed by the algorithm by evaluating recursively the set of parameters $\{R^s_{e \to v}\}, \{r^s_{e \to v}\}$ as prescribed by Proposition 5. The belief function associated to edge e at time t then reads $\mu^t_e(z) = \frac{1}{2}(R^t_{e \to v} + R^t_{e \to w} - R_{ee}) z^2 + (r^t_{e \to v} + r^t_{e \to w}) z + c^t_e$, for a certain constant c^t_e, so that $\hat{x}^t_e = \arg\min_{z \in \mathbb{R}} \mu^t_e(z) = -(r^t_{e \to v} + r^t_{e \to w})/(R^t_{e \to v} + R^t_{e \to w} - R_{ee})$, independent of c^t_e. The final procedure so obtained, in the case of quadratic initial messages, for a graph G with no leaves, is given in Algorithm 5.

Algorithm 5: Min-Sum, flow problem, quadratic messages, no leaves

Input: Initial messages $\{R^0_{e \to v}\}, \{r^0_{e \to v}\}$, $e \in \vec{E}, v \in \partial e$;
for $s \in \{1, \dots, t\}$ **do**
  Compute, for each $e \in \vec{E}$, $v \in \partial e$, with $w = \partial e \setminus v$
  $R^s_{e \to v} = R_{ee} + \frac{1}{\sum_{f \in \partial w \setminus e} 1/R^{s-1}_{f \to w}}$, $\quad r^s_{e \to v} = -A_{we} \frac{\sum_{f \in \partial w \setminus e} A_{wf} r^{s-1}_{f \to w}/R^{s-1}_{f \to w} + b_w}{\sum_{f \in \partial w \setminus e} 1/R^{s-1}_{f \to w}}$;
Output: $\hat{x}^t_e = -\frac{r^t_{e \to v} + r^t_{e \to w}}{R^t_{e \to v} + R^t_{e \to w} - R_{ee}}$, for $e = (v, w) \in \vec{E}$.

Algorithm 5, with the choice of initial conditions $R^0_{e \to v} = R_{ee}$, $r^0_{e \to v} = 0$, for each $e \in \vec{E}, v \in \partial e$, will be the focus of our investigation for the flow problem. The convergence analysis will rely on the general formulation of Algorithm 4.

4. Results for d-Regular Graphs

A d-regular graph is a graph where each node has d neighbors. In this section, we first characterize the error committed by the Min-Sum algorithm to solve the voltage and flow problems on connected d-regular graphs in terms of the solution $\nu^\star = L^+ b$ of the voltage problem. Then, we develop convergence results with respect to various norms.

The proofs of the main results here presented—Lemma 6 and Lemma 9—are given in Section 5, where we develop the general machinery to analyze the behavior of the Min-Sum scheme to solve constrained optimization problems, and we present the characterization of the error committed by the voltage and flow algorithms on general weighted graphs in terms of hitting times of random walks on the computation tree (for the sake of simplicity, we postpone the results on general graphs to Section 5, as these results require the notion of computation tree). The proofs of the technical results used here are given in Appendix B.

Henceforth, let $\hat{\nu}^t$ be the outcome of Algorithm 3 at time t with the choice of initial conditions $W^0_{e \to v} = W_{vw}$ and $w^0_{e \to v} = 0$, for each $e = \{v, w\} \in E, v \in \partial e$. Analogously, let $\hat{x}^t$

be the outcome of Algorithm 5 at time t with the choice of initial conditions $R^0_{e \to v} = R_{ee}$, $r^0_{e \to v} = 0$, for each $e \in \vec{E}, v \in \partial e$. Throughout, given a weighted directed graph $\vec{G} = (V, \vec{E}, W)$ with no bi-directed edges (i.e., either $(v, w) \in \vec{E}$ or $(w, v) \in \vec{E}$), let $G = (V, E, W)$ be the weighted undirected graph naturally associated to $\vec{G}$, i.e., edges in E are obtained by removing the orientation of the edges in $\vec{E}$. Let $n := |V|$ and $m := |E|$ be the number of nodes and edges, respectively.

4.1. Error Characterization

We characterize the error committed by the Min-Sum algorithm as a function of time and as a function of the voltage solution $\nu^\star = L^+ b$.

The case $d = 2$, i.e., a cycle graph, allows a characterization for general weighted graphs.

Lemma 6 (Regular graphs, $d = 2$) *Let $\vec{G} = (V = \{0, \dots, n-1\}, \vec{E}, W)$ be a weighted directed cycle, and let $G = (V, E, W)$ be the corresponding undirected graph. Define the function $i \in \mathbb{N} \to \rho(i) := i \mod n$. For $v \in V$, $e = \{\rho(v), \rho(v+1)\} \in E$, $t \geq 2$, we have*

$$\nu^\star_v - \hat{\nu}^t_v = \alpha_{v,t} \, \nu^\star_{\rho(v-t-1)} + (1 - \alpha_{v,t}) \, \nu^\star_{\rho(v+t+1)}, \qquad \alpha_{v,t} := \frac{\sum_{k=v}^{v+t} 1/W_{\rho(k)\rho(k+1)}}{\sum_{k=v-t-1}^{v+t} 1/W_{\rho(k)\rho(k+1)}},$$

$$x^\star_e - \hat{x}^t_e = A_{ve} \, \beta_{e,t} \, (\nu^\star_{\rho(v-t)} - \nu^\star_{\rho(v+t+1)}), \qquad \beta_{e,t} := \frac{1}{\sum_{k=v-t}^{v+t} 1/W_{\rho(k)\rho(k+1)}}.$$

If each edge has the same weight ω, then $\alpha_{v,t} = 1/2$ and $\beta_{e,t} = \omega/(2t+1)$.

In the case of constant weights, this result shows that Min-Sum displays a non-convergent (oscillatory) behavior in the voltage setting (as $\alpha_{v,t}$ does not decay with time), while Min-Sum displays a convergent behavior in the flow setting (as $\beta_{v,t}$ decays with time). We will discuss this behavior more extensively in the following sections.

Remark 7 *The fact that the Min-Sum algorithm does not converge is not surprising. Gaussian BP, see Remark 4, is known to suffer from divergent behaviors even in the absence of degeneracy, i.e., even when the quadratic form of interest is defined by a positive definite matrix, as observed numerically in Rusmevichientong and Roy (2001).*

For $d \geq 3$ we can provide a characterization of the error in the case of graphs with equal weights. We begin by defining and bounding some quantities of interest that we will need to state our results. The proof of the following proposition is given in Appendix B.

Proposition 8 *Fix $d \geq 3$. For $s \geq 1$ let $h_s := \frac{1}{(d-1)^s - 1}$, and let $\delta_0 := \frac{1}{d}$, $\delta_1 := \frac{1}{d-1} + d - 1$, and $\delta_s := \frac{1}{d-1}(2 + \frac{(d-2)^2}{d-1}(1 + h_{s+2}))\delta_{s-1} - \frac{1}{(d-1)^2}\delta_{s-2}$ for $s \geq 2$. For any $t \geq 3$ define*

$$b_{d,t} := \frac{1}{(d-1)^2}\left(1 + (d-2)(1 + h_{t+1})\right)\delta_{t-2} - \frac{(d-2)(1+h_{t+1})}{(d-1)^3}\delta_{t-3},$$

$$c_{d,t} := \frac{1}{d-1}\left(1 + \frac{1}{(d-2)(1+h_t)}\right)\delta_{t-2} - \frac{1}{(d-1)^2}\delta_{t-3}.$$

Then we have $\frac{1}{2} \leq \frac{d-2}{d-1} \leq b_{d,t} \leq c_{d,t} \leq 1 + \varepsilon_d < 4$ and $c_{d,t} \geq 1$, where ε_d is a positive decreasing function of $d \geq 3$ so that $\varepsilon_d \to 0$ as $d \to \infty$ and $\varepsilon_3 < 3$.

For each $v \in V$, let $\vec{\mathbf{P}}_v$ be the law of a time-homogeneous *non-backtracking* random walk $Y_0, Y_1, Y_2, \ldots$ on the nodes of G with initial state $Y_0 = v$, which is defined by $\vec{\mathbf{P}}_v(Y_0 = w) := 1$ if $w = v$ and equal to 0 otherwise, by $\vec{\mathbf{P}}_v(Y_1 = w) := 1/d$ if $\{w, v\} \in E$ and equal to 0, and, for any $\{z, z'\} \in E$, by $\vec{\mathbf{P}}_v(Y_{t+1} = w | Y_t = z, Y_{t-1} = z') := \frac{1}{d-1}$ if $\{w, z\} \in E, w \neq z'$ and equal to 0 otherwise. For ease of notation, for $t \geq 0$, $w \in V$, define the row-stochastic matrices $P^{(t)}$ and $P^{(t,w)}$ as

$$P^{(t)}_{vz} := \vec{\mathbf{P}}_v(Y_t = z), \qquad P^{(t,w)}_{vz} := \vec{\mathbf{P}}_v(Y_t = z | Y_1 \neq w), \qquad v, z \in V. \tag{8}$$

We can now state the main result for d-regular graphs with $d \geq 3$ and equal weights, namely, $W_{vw} = \omega$ for any $\{v, w\} \in E$, for some $\omega > 0$.

Lemma 9 (Regular graphs, $d \geq 3$, equal weights) *Let $\vec{G} = (V, \vec{E}, W)$ be a connected directed d-regular graph, with $d \geq 3$, where each edge has the same weight $\omega > 0$, and let $G = (V, E, W)$ be the corresponding undirected graph. For $v \in V$, $e = (v, w) \in \vec{E}$, $t \geq 3$,*

$$\nu^\star_v - \hat{\nu}^t_v = \frac{1}{b_{d,t}} (P^{(t+1)} \nu^\star)_v, \qquad x^\star_e - \hat{x}^t_e = \frac{\omega}{c_{d,t}} \frac{d-1}{d} ((P^{(t,w)} \nu^\star)_v - (P^{(t,v)} \nu^\star)_w),$$

where $b_{d,t}$ and $c_{d,t}$ are defined and bounded as in Proposition 8.

This result characterizes the error committed by the Min-Sum algorithm in terms of the distribution of properly-defined non-backtracking random walks. Several papers have previously related the convergence properties of message-passing algorithms to non-backtracking random walks. For instance, we refer to Moallemi and Roy (2006) for an application to consensus optimization (this paper also analyzes d-regular graphs), and to Abbe (2018) for a more recent application to the Stochastic Block Model.

Given the error characterizations provided by Lemma 6 and Lemma 9, we now investigate and discuss the convergence properties of the Min-Sum algorithm.

4.2. Convergence Results for the Voltage Problem

We first analyze the convergence behavior of the Min-Sum algorithm when applied to the voltage problem: Algorithm 3. Following the literature on Laplacian solvers, as already noticed in Section 2, we perform the analysis in the L-norm, which is defined as $\|\nu\|_L := \sqrt{\nu^T L \nu}$ for any $\nu \in \mathbb{R}^V$, where $\nu^T L \nu = \sum_{\{v,w\} \in E} W_{vw} (\nu_v - \nu_w)^2$.

In the case $d = 2$, as previously discussed, the results of Lemma 6 attest that the output $\hat{\nu}^t$ of Algorithm 3 does *not* converge, as it keeps oscillating as a function of t. This is particularly evident in the case of equal weights, as the following corollary attests.

Corollary 10 (Regular graphs, $d = 2$) *Consider the setting of Lemma 6 with equal weights $W_{vw} = \omega$. For $t \geq 2$, $\|\nu^\star - \hat{\nu}^t\|^2_L = \frac{1}{2}\|\nu^\star\|^2_L + \frac{\omega}{2} \sum_{v=0}^{n-1} \nu^\star_v (2\nu^\star_{\rho(v+2t+2)} - \nu^\star_{\rho(v+2t+3)} - \nu^\star_{\rho(v+2t+1)})$.*

Proof The proof follows immediately from Lemma 6 noticing that $\|\nu^\star - \hat{\nu}^t\|^2_L$ equals $\omega \sum_{v=0}^{n-1} (\nu^\star_{v+1} - \hat{\nu}^t_{v+1} - (\nu^\star_v - \hat{\nu}^t_v))^2 = \frac{\omega}{4} \sum_{v=0}^{n-1} (\nu^\star_{\rho(v-t)} - \nu^\star_{\rho(v-t-1)} + \nu^\star_{\rho(v+t+2)} - \nu^\star_{\rho(v+t+1)})^2$, which reads $\frac{1}{2}\|\nu^\star\|^2_L + \frac{\omega}{2} \sum_{v=0}^{n-1} (\nu^\star_{\rho(v-t)} - \nu^\star_{\rho(v-t-1)})(\nu^\star_{\rho(v+t+2)} - \nu^\star_{\rho(v+t+1)})$. The result follows by grouping and relabeling the terms in the sum. ■

The oscillatory behavior of the Min-Sum algorithm on cycle graphs in the voltage problem marks a clear difference with what happens in the flow problem where, as we will see in Corollary 16 below, the algorithm *does* converge to the problem solution on cycle graphs.

Let us now consider the case $d \geq 3$. In this case, as Corollary 13 below attests, the convergence of the Min-Sum algorithm in the L-norm can be controlled by the total variation (TV) distance between the probability distributions of non-backtracking random walks that originate from neighbor nodes. In this framework, it is natural to also consider a modification of the L-norm normalized by the probability mass that is left out by the overlap of the above-mentioned random walks, which coincides with the TV distance between the walks. We give a general definition of this norm, and then present some of its properties.

Definition 11 (TV-normalized L-norm) *Given a row-stochastic matrix $M \in \mathbb{R}^{V \times V}$, define the pseudo-norm $\| \cdot \|_{L,M}$ as $\|\nu\|^2_{L,M} := \sum_{\{v,w\} \in E} W_{vw} \left(\frac{(M\nu)_v - (M\nu)_w}{\|M_v - M_w\|_{TV}} \right)^2$, for any $\nu \in \mathbb{R}^V$, where $\|M_v - M_w\|_{TV} := \frac{1}{2} \sum_{z \in V} |M_{vz} - M_{wz}|$ is the TV distance between the probability mass functions defined by the v-th and w-th rows of the matrix M, respectively.*

The fact that $\| \cdot \|_{L,M}$ is a pseudo-norm follows from the fact that $\| \cdot \|_L$ itself is a pseudo-norm. Indeed, note that $\|\nu\|_{L,M} = \|M\nu\|_{L'}$, where L' is the Laplacian of the weighted graph (V, E, W'), with $W'_{vw} := W_{vw} / \|M_v - M_w\|^2_{TV}$. By a well-known result (see Proposition 4.3 in Levin et al. (2009), for instance), we have $\|M_v - M_w\|_{TV} = \sum_{z \in V : M_{vz} > M_{wz}} (M_{vz} - M_{wz}) = \sum_{z \in V : M_{vz} < M_{wz}} (M_{vz} - M_{wz})$, which shows how the TV-normalized L-norm takes into account the overlap of the distributions defined by the rows of the matrix M.

Proposition 12 *For any row-stochastic matrix $M \in \mathbb{R}^{V \times V}$ and $\nu \in \mathbb{R}^V$ we have:*

(i) $\|\nu\|_{L,I} = \|\nu\|_L$, *where I is the identity matrix;*

(ii) $\|M\nu\|_L \leq \max_{\{v,w\} \in E} \|M_v - M_w\|_{TV} \, \|\nu\|_{L,M}$;

(iii) $\|\nu\|_L \leq (\sum_{\{v,w\} \in E} W_{vw})^{1/2} \max_{\{v,w\} \in E} |\nu_v - \nu_w| \leq (\sum_{\{v,w\} \in E} W_{vw})^{1/2} \operatorname{osc}(\nu)$;

(iv) $\|\nu\|_{L,M} \leq (\sum_{\{v,w\} \in E} W_{vw})^{1/2} \operatorname{osc}(\nu)$;

where the oscillation of ν is defined as $\operatorname{osc}(\nu) := \max_{v,w \in V} |\nu_v - \nu_w|$.

Proof Properties $(i), (ii)$ and (iii) are straightforward to verify. Property (iv) follows from $|(M\nu)_v - (M\nu)_w| \leq \|M_v - M_w\|_{TV} \operatorname{osc}(\nu)$. See Georgii (2011), Chapter 8, for instance. ■

We now state a convergence result for the Min-Sum algorithm $\hat{\nu}^t$ for the voltage problem, in the case of d-regular graphs, $d \geq 3$, with the same edge weights. As the L-norm involves a sum over neighbour nodes, and as Lemma 9 shows that the error of Min-Sum is characterized by the distribution of a (non-overlapping) random walk, we immediately see that Min-Sum for the voltage problem does not converge on bipartite graphs as in this case random walks that start at two neighbor locations never overlap. For this reason, we also consider an averaged version of the Min-Sum algorithm where the output of two consecutive iterations

is properly averaged: $\hat{\nu}^t_{ave} := \frac{b_{d,t-1}}{b_{d,t-1}+b_{d,t}}\hat{\nu}^{t-1} + \frac{b_{d,t}}{b_{d,t-1}+b_{d,t}}\hat{\nu}^t$. Let $Q^{(t)} := (P^{(t-1)} + P^{(t)})/2$ and define

$$\gamma(t) := \max_{\{v,w\}\in E} \|P_v^{(t)} - P_w^{(t)}\|_{TV}, \qquad \gamma_{\rm ave}(t) := \max_{\{v,w\}\in E} \|Q_v^{(t)} - Q_w^{(t)}\|_{TV}.$$

Corollary 13 (Regular graphs, $d \geq 3$, equal weights) *Setting of Lemma 9. For $t \geq 4$,*

$$\|\nu^\star - \hat{\nu}^t\|_L \leq 2\,\gamma(t+1)\,\|\nu^\star\|_{L,P^{(t+1)}} \leq 2\,\gamma(t+1)\,\sqrt{m}\,\mathrm{osc}(\nu^\star),$$
$$\|\nu^\star - \hat{\nu}^t_{\rm ave}\|_L \leq 2\,\gamma_{\rm ave}(t+1)\,\|\nu^\star\|_{L,Q^{(t+1)}} \leq 2\,\gamma_{\rm ave}(t+1)\,\sqrt{m}\,\mathrm{osc}(\nu^\star).$$

Proof From Lemma 9, $\nu^\star - \hat{\nu}^t = P^{(t+1)}\nu^\star/b_{d,t}$. Using property (ii) in Proposition 12 with $M = P^{(t+1)}$ we get $\|\nu^\star - \hat{\nu}^t\|_L^2 = \|P^{(t+1)}\nu^\star\|_L^2/b_{d,t}^2 \leq 4\,\gamma(t+1)^2\,\|\nu^\star\|^2_{L,P^{(t+1)}}$, as $b_{d,t} \geq 1/2$ by Proposition 8. As for $t \geq 4$ Lemma 9 yields $(b_{d,t-1}+b_{d,t})(\nu^\star - \hat{\nu}^t_{\rm ave}) = b_{d,t-1}(\nu^\star - \hat{\nu}^{t-1}) + b_{d,t}(\nu^\star - \hat{\nu}^t) = 2Q^{(t+1)}\nu^\star$, we get $\|\nu^\star - \hat{\nu}^t_{\rm ave}\|_L^2 = 4\|Q^{(t+1)}\nu^\star\|_L^2/(b_{d,t-1}+b_{d,t})^2 \leq 4\,\gamma_{\rm ave}(t+1)^2\,\|\nu^\star\|^2_{L,Q^{(t+1)}}$. The proof is concluded using Property (iv) in Proposition 12. ■

Corollary 13 shows that the convergence in the L-norm of the Min-Sum estimate $\hat{\nu}^t$ and its averaged version $\hat{\nu}^t_{\rm ave}$ is controlled by the behavior with time of the quantities $\gamma(t+1)$ and $\gamma_{\rm ave}(t+1)$, respectively, which represent uniform bounds on the TV distance between probability distributions of non-backtracking random walks that make $t+1$ steps originating from neighbor nodes. Proposition 32 in Appendix B yields a recursive formula to compute the matrices $P^{(t)}$'s in terms of the adjacency matrix of the undirected unweighted graph (V, E). This formula can then be used to compute the quantities $\gamma(t)$ and $\gamma_{\rm ave}(t)$. Here we present numerical results for a few classes of d-regular graphs (for convenience of presentation, we include also the numerical results related to the quantities $\widetilde{\gamma}(t)$ and $\widetilde{\gamma}_{\rm ave}(t)$ that control the convergence behavior of the Min-Sum algorithm in the flow problem and that will be defined later on in Section 4.3).

Example 1 (Complete graphs) *A n-complete graph is a graph with n nodes where every pair of nodes is connected by an edge. A n-complete graph is d-regular with degree $d = n-1$. Figure 1 (left). The quantity $\gamma(t)$ is seen to decay (albeit not monotonically) like an exponential function $\alpha(d)\exp(-\beta(d)t)$, where $\alpha(d) > 0$ is a decreasing function of d and $\beta(d) > 0$ is an increasing function of d. Figure 1 (center).*

Example 2 (Connected cycles) *A $d/2$-connected cycle, for $d \geq 4$ even, is a cycle graph where each node is connected to its d nearest nodes. Figure 2 (left). The quantity $\gamma(t)$ is seen to decay like a polynomial function $\alpha(d)/\sqrt{t}$, where $\alpha(d) > 0$ is an increasing function of d. The decay of $\gamma(t)$ is independent[3] of n and m. Figure 2 (center).*

Example 3 (Tori) *A $d/2$-dimensional torus, for $d \geq 4$ even, is a $d/2$-dimensional grid with periodic boundary conditions. Figure 3 (left). The quantity $\gamma_{\rm ave}(t)$ is seen to decay like a polynomial function $\alpha(d)/\sqrt{t}$, where $\alpha(d) > 0$ is an increasing function of d. The decay of $\gamma_{\rm ave}(t)$ is independent of n and m (the same considerations as in Example 2 hold). Figure 3 (center).*

3. It is implied that the decay with time is independent of n and m as long as time t satisfies $t \lesssim D/d$, where D is the diameter of the graph. Once the random walks reach the periodic boundary conditions the decay is faster (exponential instead of polynomial) due to the extra overlaps of the walks.

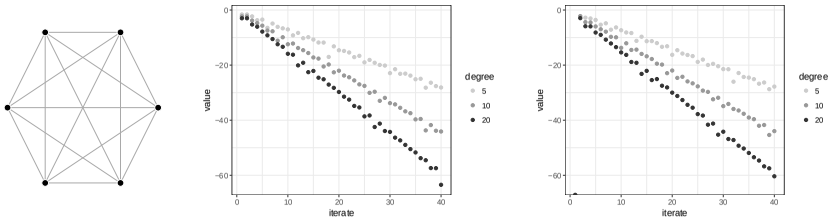


Figure 1: A 6-complete graph (left). Plot of $\log(\gamma(t))$ (center) and of $\log(\widetilde{\gamma}(t))$ (right) versus time t for various $(d+1)$-complete graphs.

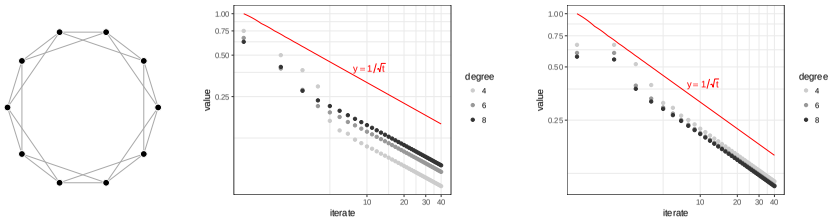


Figure 2: A 2-connected cycle (left). Plot in logarithmic scale of $\gamma(t)$ (center) and of $\widetilde{\gamma}(t)$ (right) versus time t for various $d/2$-connected cycles.

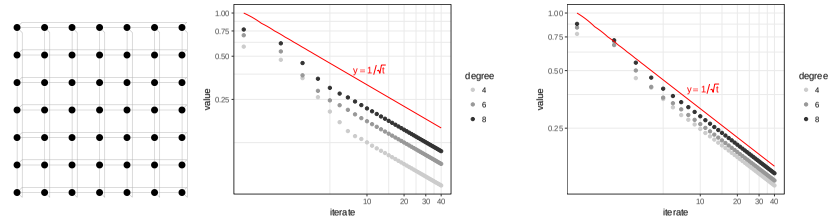


Figure 3: A 2-dimensional torus (left). Plot in logarithmic scale of $\gamma_{\rm ave}(t)$ (center) and of $\widetilde{\gamma}_{\rm ave}(t)$ (right) versus time t for various $d/2$-dimensional tori.

Remark 14 (Dobrushin's ergodicity coefficient) *For Markov chains with transition kernel P, there is a large literature investigating the contraction coefficient $\max_{v,w\in V}\|P_v^t - P_w^t\|_{TV}$ (a.k.a. Dobrushin's ergodicity coefficient), where t here represents the minimal time so that the coefficient is strictly less than 1. The Dobrushin's coefficient is fundamentally different from $\gamma(t)$. The former involves a worst-case bound over all nodes in the graph, while the latter involves a worst-case bound only over nearest-neighbors nodes. As a result,*

the Dobrushin's ergodicity coefficient typically depends on n and m, even when $\gamma(t)$ does not, such as in the examples above.

The fast solvers analyzed in the literature (see Introduction, Section 1, and Section 2) achieve a quasi-linear running time $\tilde{O}(m \log^c n \log 1/\varepsilon)$ to return an estimate $\hat{\nu}$ that satisfies $\|\nu^\star - \hat{\nu}\|_L \leq \varepsilon \|\nu^\star\|_L$. As per the results in Corollary 13, to compare the performance of the Min-Sum algorithm to these solvers we need to relate the TV-normalized L-norm to the ordinary L-norm. It is easy to construct examples where $\|\nu^\star\|_L < \|\nu^\star\|_{L,M}$ or where $\|\nu^\star\|_L > \|\nu^\star\|_{L,M}$, for a certain row-stochastic matrix M, while both norms satisfy $\|\nu^\star\|_L \leq \sqrt{m}\,\mathrm{osc}(\nu)$ (property (iii) in Proposition 12) and $\|\nu^\star\|_{L,M} \leq \sqrt{m}\,\mathrm{osc}(\nu)$ (property (iv) in Proposition 12). The following theorem summarizes the algorithmic complexity of the Min-Sum algorithm ν^t in the L-norm, under the assumption that the two norms of interest can be compared, and under the assumption that $\gamma(t)$ decays exponentially or polynomially, as attested numerically in the examples above (an analogous statement holds for the averaged Min-Sum algorithm ν^t_{ave}, replacing $P^{(t)}$ with $Q^{(t)}$, and $\gamma(t)$ with $\gamma_{\mathrm{ave}}(t)$).

Theorem 15 (Min-Sum voltage, running time, L-norm) *Setting of Lemma 9. Let $\alpha > 0, \beta > 0, \delta > 1$ independent of n, m. Assume that $\|\nu^\star\|_{L,P^{(t)}} \leq \delta \|\nu^\star\|_L$ for $t \geq 0$.*

1. *If $\gamma(t) \leq \alpha \exp(-\beta t)$ for $t \geq 0$, then the Min-Sum algorithm yields an estimate $\hat{\nu}$ such that $\|\nu^\star - \hat{\nu}\|_L \leq \varepsilon \|\nu^\star\|_L$ with running time $O(m \log 1/\varepsilon)$.*
2. *If $\gamma(t) \leq \alpha/t^\beta$ for $t \geq 0$, then the Min-Sum algorithm yields an estimate $\hat{\nu}$ such that $\|\nu^\star - \hat{\nu}\|_L \leq \varepsilon \|\nu^\star\|_L$ with running time $O(m/\varepsilon^{1/\beta})$.*

Proof The proof follows immediately from Corollary 13, recalling that each iteration of the Min-Sum algorithm requires a complexity that is linear in the number of edges, i.e., $O(m)$. If *1.* holds, then $\|\nu^\star - \hat{\nu}^t\|_L \leq \varepsilon \|\nu^\star\|_L$ is guaranteed by choosing t such that $2\alpha\delta \exp(-\beta(t+1)) \leq \varepsilon$; we can choose the smallest value $t = \lceil \log(2\alpha\delta/\varepsilon)/\beta - 1 \rceil \sim O(\log 1/\varepsilon)$. If *2.* holds, then $\|\nu^\star - \hat{\nu}^t\|_L \leq \varepsilon \|\nu^\star\|_L$ is guaranteed by choosing t such that $2\alpha\delta/(t+1)^\beta \leq \varepsilon$; we can choose the smallest value $t = \lceil (2\alpha\delta/\varepsilon)^{1/\beta} - 1 \rceil \sim O(1/\varepsilon^{1/\beta})$. ■

Theorem 15 shows that for certain classes of d-regular graphs, as characterized by the decaying properties of $\gamma(t)$, under the assumption that the TV-normalized L-norm of $\nu^\star$ is bounded by the L-norm of $\nu^\star$, the Min-Sum algorithm yields linear time solvers for the voltage problem, with respect to the L-norm.

4.3. Convergence Results for the Flow Problem

We now analyze the convergence behavior of the Min-Sum algorithm when applied to the flow problem: Algorithm 5. We present the results for the ℓ_∞ norm, defined as $\|x\|_\infty := \max_{e \in \vec{E}} x_e$ for any $x \in \mathbb{R}^{\vec{E}}$. The results for the ℓ_2 norm, defined as $\|x\|_2 := \sqrt{x^T x}$, are analogous to the results in the L-norm for the voltage case, as we describe in Remark 19.

For the case $d = 2$, Lemma 6 immediately yields the following result.

Corollary 16 (Regular graphs, $d = 2$) *Consider the setting of Lemma 6. Let $\omega_M := \max_{\{v,w\} \in E} W_{vw}$. Then, $\|x^\star - \hat{x}^t\|_\infty \leq \frac{2\omega_M}{2t+1} \|\nu^\star\|_\infty$. If ω_M is independent of n and m, then the Min-Sum algorithm yields $\hat{x}$ such that $\|x^\star - \hat{x}\|_\infty \leq \varepsilon \|\nu^\star\|_\infty$ with running time $O(m/\varepsilon)$.*

Proof From Lemma 6, by the triangle inequality, we have $|x^\star_e - \hat{x}^t_e| \leq \frac{\omega_M}{2t+1}(|\nu^\star_{\rho(v-t)}| + |\nu^\star_{\rho(v+t+1)}|)$, and the first bound follows by the triangle inequality for norms. The running time is immediate, as each iteration of the algorithm requires a complexity $O(m)$. ■

Corollary 16 shows that the flow Min-Sum algorithm on cycles converges in linear time to the problem solution, as opposed to the divergent behavior observed in the voltage case.

Let us now consider the case $d \geq 3$. As in the voltage problem, also in this case the convergence of the Min-Sum algorithm can be controlled by the TV distance between the probability distributions of properly-defined non-backtracking random walks that originate from neighbor nodes. As before, we also consider an averaged version of the Min-Sum algorithm where the output of two consecutive iterations is properly averaged: $\hat{x}^t_{\mathrm{ave}} := \frac{c_{d,t-1}}{c_{d,t-1}+c_{d,t}} \hat{x}^{t-1} + \frac{c_{d,t}}{c_{d,t-1}+c_{d,t}} \hat{x}^t$. Analogously to the voltage case, let $Q^{(t,w)} := (P^{(t-1,w)} + P^{(t,w)})/2$ and

$$\widetilde{\gamma}(t) := \max_{\{v,w\} \in E} \|P^{(t,w)}_v - P^{(t,v)}_w\|_{TV}, \qquad \widetilde{\gamma}_{\mathrm{ave}}(t) := \max_{\{v,w\} \in E} \|Q^{(t,w)}_v - Q^{(t,v)}_w\|_{TV}.$$

Corollary 17 (Regular graphs, $d \geq 3$, equal weights) *Setting of Lemma 9. For $t \geq 4$,*

$$\|x^\star - \hat{x}^t\|_\infty \leq 2\,\omega\,\frac{d-1}{d}\,\widetilde{\gamma}(t)\,\|\nu^\star\|_\infty, \qquad \|x^\star - \hat{x}^t_{\mathrm{ave}}\|_\infty \leq 2\,\omega\,\frac{d-1}{d}\,\widetilde{\gamma}_{\mathrm{ave}}(t)\,\|\nu^\star\|_\infty.$$

Proof For any $t \geq 1$, define the matrix $\widetilde{\Delta}^{(t)} \in \mathbb{R}^{\vec{E} \times V}$ as $\widetilde{\Delta}^{(t)}_{ez} := P^{(t,w)}_{vz} - P^{(t,v)}_{wz}, e = (v,w) \in \vec{E}$, $z \in V$. From Lemma 9 we have $x^\star - \hat{x}^t = \frac{\omega}{c_{d,t}} \frac{d-1}{d} \widetilde{\Delta}^{(t)} \nu^\star$, and the first bound follows from the sub-multiplicative property of the induced ℓ_∞ matrix norm, noticing that $\widetilde{\gamma}(t) = \|\widetilde{\Delta}^{(t)}\|_\infty/2$, and using that $c_{d,t} \geq 1$ from Proposition 8. The second bound follows analogously, as for any $t \geq 4$ we have $(c_{d,t-1} + c_{d,t})(x^\star - \hat{x}^t_{\mathrm{ave}}) = c_{d,t-1}(x^\star - \hat{x}^{t-1}) + c_{d,t}(x^\star - \hat{x}^t) = \omega \frac{d-1}{d}(\widetilde{\Delta}^{(t-1)} + \widetilde{\Delta}^{(t)})\nu^\star$, and $\widetilde{\gamma}_{\mathrm{ave}}(t) = \|(\widetilde{\Delta}^{(t-1)} + \widetilde{\Delta}^{(t)})/2\|_\infty/2$. ■

Corollary 17 shows that the convergence in the ℓ_∞ norm of the Min-Sum estimate $\hat{x}^t$ and its averaged version $\hat{x}^t_{\mathrm{ave}}$ is controlled by the behavior with time of the quantities $\widetilde{\gamma}(t)$ and $\widetilde{\gamma}_{\mathrm{ave}}(t)$, respectively, which represent uniform bounds on the TV distance between probability distributions of non-backtracking random walks that make t steps originating from neighbor nodes, conditioned on the event that the two walks avoid each others at the first step. Proposition 32 in Appendix B yields a recursive formula to compute the matrices $P^{(t,w)}$'s in terms of the adjacency matrix of the undirected unweighted graph (V, E). This formula can then be used to compute the quantities $\widetilde{\gamma}(t)$ and $\widetilde{\gamma}_{\mathrm{ave}}(t)$. Numerical results for a few classes of d-regular graphs have been presented in the previous section. See Figure 1 (right)[4], Figure 2 (right), and Figure 3 (right). The following theorem summarizes the algorithmic complexity of the Min-Sum algorithm x^t in the ℓ_∞ norm, under the assumption that $\widetilde{\gamma}(t)$ decays exponentially or polynomially, as attested numerically in the examples above (an analogous statement holds for the output x^t_{ave}, replacing $\widetilde{\gamma}(t)$ with $\widetilde{\gamma}_{\mathrm{ave}}(t)$).

Theorem 18 (Min-Sum flow, running time, ℓ_∞ norm) *Setting of Lemma 9. Let $\alpha > 0, \beta > 0$, and ω be independent of n and m.*

4. It is easy to see that $\widetilde{\gamma}(1) = 0$ for complete graphs, which explains the points at $t = 1$ in Figure 1 (right).

1. *If $\widetilde{\gamma}(t) \le \alpha \exp(-\beta t)$ for $t \ge 0$, then the Min-Sum algorithm yields an estimate $\hat{x}$ such that $\|x^\star - \hat{x}\|_\infty \le \varepsilon \|\nu^\star\|_\infty$ with running time $O(m \log 1/\varepsilon)$.*

2. *If $\widetilde{\gamma}(t) \le \alpha / t^\beta$ for $t \ge 0$, then the Min-Sum algorithm yields an estimate $\hat{x}$ such that $\|x^\star - \hat{x}\|_\infty \le \varepsilon \|\nu^\star\|_\infty$ with running time $O(m/\varepsilon^{1/\beta})$.*

Proof It follows from Corollary 17, analogously to the proof of Theorem 15. ■

Theorem 18 shows that for certain classes of d-regular graphs, as characterized by the decaying properties of $\widetilde{\gamma}(t)$, the Min-Sum algorithm yields linear time solvers for the flow problem, with respect to the ℓ_∞ norm. The following remark shows that an analogous result to Theorem 18 can be derived with respect to the ℓ_2 norm, with an additional assumption in the spirit of the one used in Theorem 15 for the voltage case.

Remark 19 (Analysis in ℓ_2 norm) *For d-regular graphs with $d \ge 3$, the analysis in the ℓ_2 norm for the flow problem follows the same steps as the analysis in the L-norm for the voltage problem. In fact, if one defines the matrices $\Delta^{(t)}$ and $\widetilde{\Delta}^{(t)} \in \mathbb{R}^{\vec{E} \times V}$ respectively as $\Delta^{(t)}_{ez} := P^{(t)}_{vz} - P^{(t)}_{wz}$ and $\widetilde{\Delta}^{(t)}_{ez} := P^{(t,w)}_{vz} - P^{(t,v)}_{wz}$, for any $e = (v,w) \in \vec{E}, z \in V$, then $\|x^\star - \hat{x}^t\|_2 = \frac{\omega}{c_{d,t}} \frac{d-1}{d} \|\widetilde{\Delta}^{(t)} \nu^\star\|_2$, while $\|\nu^\star - \hat{\nu}^t\|_L = \frac{1}{b_{d,t}} \|\Delta^{(t)} \nu^\star\|_2$. In analogy with Theorem 15, the results of Theorem 18 hold when replacing $\|x^\star - \hat{x}\|_\infty \le \varepsilon \|\nu^\star\|_\infty$ with $\|x^\star - \hat{x}\|_2 \le \varepsilon \|\nu^\star\|_L$, upon assuming that there exists $\delta > 1$ independent of n and m such that $\sum_{\{v,w\} \in E} W_{vw} \big(\frac{(P^{(t,w)}\nu)_v - (P^{(t,v)}\nu)_w}{\|P^{(t,w)}_v - P^{(t,v)}_w\|_{TV}} \big)^2 \le \delta^2 \|\nu^\star\|^2_L$, for $t \ge 0$, which corresponds to the assumption $\|\nu^\star\|_{L, P^{(t)}} \le \delta \|\nu^\star\|_L$ in Theorem 15.*

5. Results for General Weighted Graphs

This section develops the characterization of the error committed by the voltage and flow algorithms on general weighted graphs in terms of hitting times of ordinary diffusion random walks defined on the computation trees that are obtained by unraveling the operations of the algorithms with time (Theorem 23 and Lemma 24 for the flow case, and Theorem 28 for the voltage case). As applications, we specialize the general results here developed to prove the error characterization lemmas given in Section 4, i.e., Lemma 6 and Lemma 9.

While the main architecture of the results for the voltage problem (Algorithm 3) and the flow problem (Algorithm 5) is similar, there are some key differences that need to be taken into account. First, as the voltage algorithm updates functions (i.e., messages) on the vertices while the flow algorithm updates functions on the edges, the two algorithms give rise to different topologies for the computation tree. Second, as the voltage problem is an unconstrained optimization problem while the flow problem is an optimization problem with constraints, a different mechanism to find the fix point(s) of the algorithm is needed. As in the literature, to the best of our knowledge, previous analyses of the Min-Sum algorithm for quadratic and, more generally, convex problems only focused on the unconstrained case, we first present the full details of the results for the flow problem. Then, we give an overview of the results for the voltage problem, outlining in details only the parts where the analysis differs significantly from the one done for the flow problem.

5.1. Flow Problem, Algorithm 3

Our framework builds on the general scheme presented in Moallemi and Van Roy (2010) for unconstrained optimization problems, as we investigate the evolution of the Min-Sum algorithm under a set of linear perturbations. Henceforth, we consider Algorithm 5 when the initial messages are parametrized by $\{R^0_{e \to v} = R_{ee}\}, \{r^0_{e \to v} = p_{e \to v}\}$, for a certain set of real numbers $p = \{p_{e \to v}\}$. As far as the analysis is concerned, it is convenient (and more amenable to generalizations) to consider the form of Algorithm 4. For a given choice of perturbation parameters $p = \{p_{e \to v}\}$, define the initial messages as $\mu^0_{e \to v}(\,\cdot\,, p) : z \in \mathbb{R} \longrightarrow \mu^0_{e \to v}(z, p) = \frac{1}{2} R_{ee} z^2 + p_{e \to v} z$, and denote by $\{\mu^s_{e \to v}(\,\cdot\,, p)\}$, $s \ge 1$, the corresponding sequence of messages generated by the Min-Sum algorithm. Denote the estimates at time t as $\hat{x}^t(p)_e := \arg\min_{z \in \mathbb{R}} \mu^t_e(z, p)$, where $\mu^t_e(\,\cdot\,, p)$ is the corresponding belief function. We use the notation $\mathcal{N}(v) := \{w \in V : \{v, w\} \in E\}$ to denote the neighborhood of vertex v in G.

5.1.1. Computation Tree (Flow Problem)

To investigate the convergence behavior of the Min-Sum algorithm, we recall the concept of the computation tree, which is the graph obtained by unraveling the operations of the algorithm with time (Gallager, 1963; Weiss and Freeman, 2001; Tatikonda and Jordan, 2002). Given an edge $\tilde{e} = (\tilde{v}, \tilde{w}) \in \vec{E}$, the computation tree rooted at $\tilde{e}$ of depth t supports the optimization problem that is obtained by unfolding the computations involved in the Min-Sum estimate $\hat{x}^t(p)_{\tilde{e}}$. Formally, the computation tree is a directed tree $\vec{\mathbb{T}} = (\mathbb{V}, \vec{\mathbb{E}})$ (throughout, we use the double-struck notation to refer to quantities related to the computation tree) where each vertex in $\mathbb{V}$ is mapped to a vertex in V through a map $\sigma : \mathbb{V} \to V$ that preserves the edge structure, namely, if $\mathbb{e} = (\mathbb{v}, \mathbb{w}) \in \vec{\mathbb{E}}$ then $(\sigma(\mathbb{v}), \sigma(\mathbb{w})) \in \vec{E}$. Henceforth, we use the notation $\sigma(\mathbb{e}) := (\sigma(\mathbb{v}), \sigma(\mathbb{w}))$. The tree $\vec{\mathbb{T}}$ is defined iteratively. Initially, at level 0, the tree corresponds to a single root edge $\tilde{\mathbb{e}} = (\tilde{\mathbb{v}}, \tilde{\mathbb{w}}) \in \vec{\mathbb{E}}$ corresponding to $\tilde{e}$, i.e., $\sigma(\tilde{\mathbb{e}}) = \tilde{e}$. At this stage, $\tilde{\mathbb{v}}$ and $\tilde{\mathbb{w}}$ are the leaves of the tree. For the remaining t levels, the following procedure is repeated. The leaves in the tree are examined. Given a leaf $\mathbb{v}$ with $\sigma(\mathbb{v}) = v$ that is connected to a vertex $\mathbb{w}$ with $\sigma(\mathbb{w}) = w$, for any $z \in \mathcal{N}(v) \setminus w$, a vertex $\mathbb{z}$ with $\sigma(\mathbb{z}) = z$ and a directed edge $(\mathbb{v}, \mathbb{z})$ (resp. $(\mathbb{z}, \mathbb{v})$) are added to the next level of the tree if $(v, z) \in \vec{E}$ (resp. $(z, v) \in \vec{E}$). Figure 4 gives an example. We denote the set of vertices and edges in the k-th level of the tree respectively by $\mathbb{V}^k \subset \mathbb{V}$ and $\mathbb{E}^k \subset \mathbb{E}$. In what follows we also extend the usual neighborhood notation to vertices and edges in the graph $\mathbb{T}$, namely, $\partial \mathbb{v}$ is the set of edges in $\mathbb{T}$ that are connected to node $\mathbb{v}$, and $\partial \mathbb{e}$ is the set of vertices in $\mathbb{T}$ that are connected to edge $\mathbb{e}$.

The t-th (last) level of the tree $\vec{\mathbb{T}}$ is of particular relevance, as it supports the initial conditions of the Min-Sum algorithm, as we are now about to describe. Let $\mathbb{R} \in \mathbb{R}^{\vec{\mathbb{E}} \times \vec{\mathbb{E}}}$ be the diagonal matrix defined by $\mathbb{R}_{\mathbb{e}\mathbb{e}} := R_{\sigma(\mathbb{e})\sigma(\mathbb{e})}$. For a given choice of the perturbation parameters $p = \{p_{e \to v}\}$, define $\mathbb{p}(p) \in \mathbb{R}^{\vec{\mathbb{E}}}$ by $\mathbb{p}(p)_{\mathbb{e}} := p_{\sigma(\mathbb{e}) \to \sigma(\mathbb{v})}$ if $\mathbb{e} \in \vec{\mathbb{E}}^t, \mathbb{v} = \partial \mathbb{e} \cap \mathbb{V}^{t-1}$, and equals to 0 otherwise. Let $\bar{\mathbb{V}} := \mathbb{V} \setminus \mathbb{V}^t$ (throughout, we use the bar notation to denote quantities related to a graph in which a node has been removed). Define the matrix $\bar{\mathbb{A}} \in \mathbb{R}^{\bar{\mathbb{V}} \times \vec{\mathbb{E}}}$ as $\bar{\mathbb{A}}_{\mathbb{v}\mathbb{e}} := A_{\sigma(\mathbb{v})\sigma(\mathbb{e})}$ if $\mathbb{e} \in \partial \mathbb{v}$ and $\bar{\mathbb{A}}_{\mathbb{v}\mathbb{e}} := 0$ otherwise. Define the vector $\bar{\mathbb{b}} \in \mathbb{R}^{\bar{\mathbb{V}}}$ as $\bar{\mathbb{b}}_{\mathbb{v}} := b_{\sigma(\mathbb{v})}$.

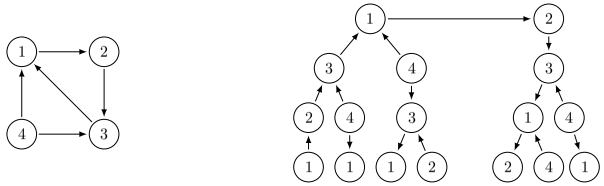


Figure 4: Graph $\vec{G} = (V = \{1, 2, 3, 4\}, \vec{E})$ (left) and computation tree $\vec{\mathbb{T}} = (\mathbb{V}, \vec{\mathbb{E}})$ of depth $t = 3$ with root edge corresponding to $(1, 2) \in \vec{E}$ (right). Vertices in $\vec{\mathbb{T}}$ are labeled by the corresponding vertices in $\vec{G}$.

Consider the following problem supported on the tree $\vec{\mathbb{T}}$, over $\mathbb{x} \in \mathbb{R}^{\vec{\mathbb{E}}}$:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\mathbb{x}^T \mathbb{R}\mathbb{x} + \mathbb{p}(p)^T \mathbb{x} \\ \text{subject to} \quad & \bar{\mathbb{A}}\mathbb{x} = \bar{\mathbb{b}}. \end{aligned} \tag{9}$$

Let $\mathbb{x}^\star(p)$ denote the unique optimal solution of this problem, as a function of the perturbation p. The relationship between the Min-Sum algorithm and the computation tree is made explicit by the following lemma, which can be easily established by inductively examining the operations of Algorithm 4.

Lemma 20 *Given the setting of this section, $\hat{x}^t(p)_{\tilde{e}} = \mathbb{x}^\star(p)_{\tilde{\mathbb{e}}}$.*

We now introduce a few quantities related to the computation tree $\vec{\mathbb{T}}$ and the problem (9) that will play a key role in our analysis. The signed vertex-edge incidence matrix $\mathbb{A} \in \mathbb{R}^{\mathbb{V} \times \vec{\mathbb{E}}}$ associated to the directed graph $\vec{\mathbb{T}}$ is defined for each $\mathbb{v} \in \mathbb{V}$, $\mathbb{e} \in \vec{\mathbb{E}}$ as $\mathbb{A}_{\mathbb{v}\mathbb{e}} := 1$ if edge $\mathbb{e}$ leaves node $\mathbb{v}$, $\mathbb{A}_{\mathbb{v}\mathbb{e}} := -1$ if edge $\mathbb{e}$ enters node $\mathbb{v}$, and $\mathbb{A}_{\mathbb{v}\mathbb{e}} := 0$ otherwise. It is easy to verify that $\bar{\mathbb{A}}$ is the submatrix of $\mathbb{A}$ that corresponds to the rows associated to $\bar{\mathbb{V}}$. Let $\mathbb{W} \in \mathbb{R}^{\mathbb{V} \times \mathbb{V}}$ be a symmetric matrix that assigns a positive weight to every edge in $\vec{\mathbb{E}}$, defined as $\mathbb{W}_{\mathbb{v}\mathbb{w}} := W_{\sigma(\mathbb{v})\sigma(\mathbb{w})}$ if either $(\mathbb{v}, \mathbb{w}) \in \vec{\mathbb{E}}$ or $(\mathbb{w}, \mathbb{v}) \in \vec{\mathbb{E}}$, and $\mathbb{W}_{\mathbb{v}\mathbb{w}} := 0$ otherwise. Henceforth, we think of $\vec{\mathbb{T}} = (\mathbb{V}, \vec{\mathbb{E}}, \mathbb{W})$ as the corresponding weighted graph, and $\mathbb{T} = (\mathbb{V}, \mathbb{E}, \mathbb{W})$ as the undirected graph naturally associated to $\vec{\mathbb{T}}$, where the edge set $\mathbb{E}$ is obtained by removing the orientation from the edges in $\vec{\mathbb{E}}$. As $\mathbb{T}$ is connected by assumption, it is easy to show that the range of $\mathbb{A}$ is spanned by the vectors orthogonal to the all-ones vector. It follows that the matrix $\bar{\mathbb{A}}$ has full row rank. Define the weighted degree of a vertex $\mathbb{v}$ by $\mathbb{d}_{\mathbb{v}} := \sum_{\mathbb{w} \in \mathbb{V}} \mathbb{W}_{\mathbb{v}\mathbb{w}}$, and let $\mathbb{D} \in \mathbb{R}^{\mathbb{V} \times \mathbb{V}}$ be the diagonal matrix defined by $\mathbb{D}_{\mathbb{v}\mathbb{v}} := \mathbb{d}_{\mathbb{v}}$. The Laplacian $\mathbb{L}$ of the graph $\vec{\mathbb{T}}$ is the matrix defined as $\mathbb{L} := \mathbb{D} - \mathbb{W}$. Let $\bar{\mathbb{L}} \in \mathbb{R}^{\bar{\mathbb{V}} \times \bar{\mathbb{V}}}$ denote the submatrix of $\mathbb{L}$ that corresponds to the rows and columns associated to $\bar{\mathbb{V}}$. Note that $\mathbb{R}_{\mathbb{e}\mathbb{e}} := 1/\mathbb{W}_{\mathbb{v}\mathbb{w}}$. It is easy to verify that the following equalities holds: $\mathbb{L} = \mathbb{A}\mathbb{R}^{-1}\mathbb{A}^T$ and $\bar{\mathbb{L}} = \bar{\mathbb{A}}\mathbb{R}^{-1}\bar{\mathbb{A}}^T$. We also define the transition matrix of the ordinary diffusion random walk on $\mathbb{T}$ as the matrix $\mathbb{P} := \mathbb{D}^{-1}\mathbb{W}$.

5.1.2. Fix Point (Flow Problem)

Clearly, the choice $p = 0$ for the linear perturbation parameters yields the original (unperturbed) algorithm. The next theorem below shows that for a particular choice of p the

Min-Sum algorithm yields the optimal solution at any time $t \geq 1$. The optimal choice of p is a function of the optimal Lagrangian multipliers for problem (3). Recall from (4) that the Lagrangian of this optimization problem is given by $\mathcal{L}(x, \nu) := \frac{1}{2}x^T Rx + \nu^T(b - Ax)$, where $\nu = (\nu_v)_{v \in V}$ is the vector formed by the Lagrangian multipliers. Define the function Ψ from $\mathbb{R}^{\vec{E}} \times \mathbb{R}^V$ to $\mathbb{R}^{\vec{E}} \times \mathbb{R}^V$ as $\Psi(x, \nu) := (\nabla_x \mathcal{L}(x, \nu), Ax - b) = (Rx - A^T\nu, Ax - b)$. Recall that $x^\star$ denotes the unique optimal solution of problem (3). As the constraints are linear, by the Lagrange multiplier theorem there exists $\nu^\star \in \mathbb{R}^V$ so that $\Psi(x^\star, \nu^\star) = 0$, i.e.,

$$\begin{cases} R_{ee}x^\star_e - (A^T\nu^\star)_e = 0 \text{ for } e \in \vec{E}, \\ \sum_{e \in \vec{E}} A_{ve}x^\star_e = b_v \text{ for } v \in V. \end{cases} \tag{10}$$

Note that the first line in (10) represents Ohm's law, and the second line represents Kirchhoff's law. Given the pair $(x^\star, \nu^\star)$, we are now ready to define the set of parameters $p^\star = \{p^\star_{e \to v}\}$ so that the Min-Sum algorithm yields the optimal solution at any time step.

Theorem 21 *For each $e \in \vec{E}$, $v \in \partial e$, let $w = \partial e \setminus v$ and define $p^\star_{e \to v} := -A_{we}\nu^\star_w$. Then, for any $\tilde{e} \in \vec{E}$, $t \geq 1$, we have $\hat{x}^t(p^\star)_{\tilde{e}} = x^\star_{\tilde{e}}$.*

Proof Fix $\tilde{e} \in \vec{E}$, $t \geq 1$. Consider the setting in Section 5.1.1, and let $\vec{\mathbb{T}}$ be the computation tree of depth t rooted at $\tilde{e}$. The Lagrangian of the problem (9) corresponding to the choice $p = p^\star$ is the function $\mathbb{L}$ from $\mathbb{R}^{\vec{\mathbb{E}}} \times \mathbb{R}^{\bar{\mathbb{V}}}$ to $\mathbb{R}$ defined by $\mathbb{L}(\mathbb{x}, \mathbb{v}) := \frac{1}{2}\mathbb{x}^T\mathbb{R}\mathbb{x} + \mathbb{p}(p^\star)^T\mathbb{x} + \mathbb{v}^T(\bar{\mathbb{b}} - \bar{\mathbb{A}}\mathbb{x})$, where $\mathbb{v} = (\mathbb{v}_{\mathbb{v}})_{\mathbb{v} \in \bar{\mathbb{V}}}$. Let Ψ from $\mathbb{R}^{\vec{\mathbb{E}}} \times \mathbb{R}^{\bar{\mathbb{V}}}$ to $\mathbb{R}^{\vec{\mathbb{E}}} \times \mathbb{R}^{\bar{\mathbb{V}}}$ be defined as $\Psi(\mathbb{x}, \mathbb{v}) := (\nabla_{\mathbb{x}}\mathbb{L}(\mathbb{x}, \mathbb{v}), \bar{\mathbb{A}}\mathbb{x} - \bar{\mathbb{b}}) = (\mathbb{R}\mathbb{x} + \mathbb{p}(p^\star) - \bar{\mathbb{A}}^T\mathbb{v}, \bar{\mathbb{A}}\mathbb{x} - \bar{\mathbb{b}})$. The Lagrangian multiplier theorem says for the unique minimizer $\mathbb{x}^\star \equiv \mathbb{x}^\star(p^\star) \in \mathbb{R}^{\vec{\mathbb{E}}}$ of (9) there exists $\mathbb{v}^\star \in \mathbb{R}^{\bar{\mathbb{V}}}$ so that $\Psi(\mathbb{x}^\star, \mathbb{v}^\star) = 0$, namely,

$$\begin{cases} \mathbb{R}_{\mathbb{e}\mathbb{e}}\mathbb{x}^\star_{\mathbb{e}} - (\bar{\mathbb{A}}^T\mathbb{v}^\star)_{\mathbb{e}} = 0, \mathbb{e} \in \vec{\mathbb{E}} \setminus \vec{\mathbb{E}}^t, \\ \mathbb{R}_{\mathbb{e}\mathbb{e}}\mathbb{x}^\star_{\mathbb{e}} + p^\star_{\sigma(\mathbb{e}) \to \sigma(\mathbb{v})} - (\bar{\mathbb{A}}^T\mathbb{v}^\star)_{\mathbb{e}} = 0, \mathbb{e} \in \vec{\mathbb{E}}^t, \mathbb{v} = \partial\mathbb{e} \cap \mathbb{V}^{t-1}, \\ \sum_{\mathbb{e} \in \vec{\mathbb{E}}} \bar{\mathbb{A}}_{\mathbb{v}\mathbb{e}}\mathbb{x}^\star_{\mathbb{e}} = \bar{\mathbb{b}}_{\mathbb{v}}, \mathbb{v} \in \bar{\mathbb{V}}. \end{cases}$$

Using (10) it is easy to check that the choice $\mathbb{x}^\star_{\mathbb{e}} = x^\star_{\sigma(\mathbb{e})}$ and $\mathbb{v}^\star_{\mathbb{v}} = \nu^\star_{\sigma(\mathbb{v})}$ satisfies the above system of equations. The proof is concluded by Lemma 20. ■

5.1.3. Error Characterization (Flow Problem)

The next lemma characterizes the sensitivity of the $\tilde{\mathbb{e}}$-th component (root edge of $\vec{\mathbb{T}}$) of the optimal solution of the optimization problem (9) with respect to perturbations of the parameters $p_{e \to v}$'s, which are supported on $\vec{\mathbb{E}}^t$.

Lemma 22 *Consider the setting in Section 5.1.1. Fix $\tilde{e} = (\tilde{v}, \tilde{w}) \in \vec{E}$, $t \geq 1$, and let $\vec{\mathbb{T}}$ be the computation tree of depth t rooted at $\tilde{e}$, with root edge $\tilde{\mathbb{e}} = (\tilde{\mathbb{v}}, \tilde{\mathbb{w}}) \in \vec{\mathbb{E}}$. For any $p = \{p_{e \to v}\}$, $e \in \vec{E}$ and $v \in \partial e$, we have*

$$\frac{\partial}{\partial p_{e \to v}}\mathbb{x}^\star(p)_{\tilde{\mathbb{e}}} = \frac{1}{R_{\tilde{e}\tilde{e}}} \sum_{\mathbb{v} \in \mathbb{V}^{t-1}} (\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} - \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}})(\mathbb{y}_{e \to v})_{\mathbb{v}},$$

where $\mathbb{y}_{e \to v} \in \mathbb{R}^{\mathbb{V}^{t-1}}$ is defined, for each $\mathbb{v} \in \mathbb{V}^{t-1}$, as $(\mathbb{y}_{e \to v})_{\mathbb{v}} := \frac{A_{ve}}{R_{ee}}\mathbf{1}_{\sigma(\mathbb{v}) = v}\mathbf{1}_{\sigma(\partial\mathbb{v} \cap \vec{\mathbb{E}}^t) = e}$, where $\mathbf{1}_S$ is the indicator function defined as $\mathbf{1}_S = 1$ if statement S is true, $\mathbf{1}_S = 0$ otherwise.

Proof By Proposition 31 in Appendix B, the solution of problem (9) reads $\mathbb{x}^\star(p) = \mathbb{R}^{-1}\bar{\mathbb{A}}^T\bar{\mathbb{L}}^{-1}\bar{\mathbb{b}} + (\mathbb{R}^{-1}\bar{\mathbb{A}}^T\bar{\mathbb{L}}^{-1}\bar{\mathbb{A}} - I)\mathbb{R}^{-1}\mathbb{p}(p)$, where $I \in \mathbb{R}^{\vec{\mathbb{E}}\times\vec{\mathbb{E}}}$ is the identity matrix. As $\mathbb{p}(p)$ is supported on the edges of $\mathbb{T}$ that are connected to its leaves and $\tilde{\mathbb{e}}$ is the root edge, we have $\frac{\partial}{\partial p_{e\to v}}\mathbb{x}^\star(p)_{\tilde{\mathbb{e}}} = \big(\mathbb{R}^{-1}\bar{\mathbb{A}}^T\bar{\mathbb{L}}^{-1}\bar{\mathbb{A}}\mathbb{R}^{-1}\frac{\partial}{\partial p_{e\to v}}\mathbb{p}(p)\big)_{\tilde{\mathbb{e}}} = \frac{1}{R_{\tilde{e}\tilde{e}}}\sum_{\mathbb{v}\in\mathbb{V}^{t-1}}(\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} - \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}})(\mathbb{y}_{e\to v})_{\mathbb{v}}$, where $(\mathbb{y}_{e\to v})_{\mathbb{v}} = (\bar{\mathbb{A}}\mathbb{R}^{-1}\frac{\partial}{\partial p_{e\to v}}\mathbb{p}(p))_{\mathbb{v}} = \frac{A_{ve}}{R_{ee}}\mathbf{1}_{\sigma(\mathbb{v})=v}\mathbf{1}_{\sigma(\partial\mathbb{v}\cap\vec{\mathbb{E}}^t)=e}$ for each $\mathbb{v}\in\mathbb{V}^{t-1}$. ■

As a consequence of Lemma 22, we immediately have the following characterization of the error committed by Algorithm 5 with initial conditions $\{R^0_{e\to v} = R_{ee}\}$, $\{r^0_{e\to v} = 0\}$.

Theorem 23 *Consider the setting in Section 5.1.1. Fix $\tilde{e} = (\tilde{v},\tilde{w}) \in \vec{E}$, $t \geq 1$, and let $\vec{\mathbb{T}}$ be the computation tree of depth t rooted at $\tilde{e}$, with root edge $\tilde{\mathbb{e}} = (\tilde{\mathbb{v}},\tilde{\mathbb{w}}) \in \vec{\mathbb{E}}$. We have*

$$x^\star_{\tilde{e}} - \hat{x}^t_{\tilde{e}} = \mathbb{W}_{\tilde{\mathbb{v}}\tilde{\mathbb{w}}}\sum_{\mathbb{v}\in\mathbb{V}^{t-1}}(\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} - \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}})\sum_{\mathbb{w}\in\mathbb{V}^t}\mathbb{W}_{\mathbb{v}\mathbb{w}}\nu^\star_{\sigma(\mathbb{w})}.$$

Proof From Theorem 21 and Lemma 22, by the fundamental theorem of calculus and the chain rule of differentiation, we get $x^\star_{\tilde{e}} - \hat{x}^t_{\tilde{e}} = \hat{x}^t(p^\star)_{\tilde{e}} - \hat{x}^t(0)_{\tilde{e}} = \int_0^1 d\theta\,\frac{d}{d\theta}\mathbb{x}^\star(\theta p^\star)_{\tilde{\mathbb{e}}} = \int_0^1 d\theta\sum_{e\in\vec{E},v\in\partial e}\frac{\partial}{\partial p_{e\to v}}\mathbb{x}^\star(\theta p^\star)_{\tilde{\mathbb{e}}}\,p^\star_{e\to v} = \mathbb{W}_{\tilde{\mathbb{v}}\tilde{\mathbb{w}}}\sum_{\mathbb{v}\in\mathbb{V}^{t-1}}(\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} - \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}})\sum_{e\in\vec{E},v\in\partial e}(\mathbb{y}_{e\to v})_{\mathbb{v}}\,p^\star_{e\to v}$. By Theorem 21 $p^\star_{e\to v} = -A_{we}\nu^\star_w$, $w = \partial e\setminus v$, and $\sum_{\substack{e\in\vec{E}\\ v\in\partial e}}(\mathbb{y}_{e\to v})_{\mathbb{v}}\,p^\star_{e\to v} = \sum_{\mathbb{w}\in\mathbb{V}^t}\mathbb{W}_{\mathbb{v}\mathbb{w}}\nu^\star_{\sigma(\mathbb{w})}$. ■

We now use the error characterization provided by Theorem 23 and the general connection between restricted Laplacians and random walks developed in Appendix A to prove the results in Section 4. Henceforth, for each $\mathbb{v}\in\mathbb{V}$, let $\mathbf{P}_{\mathbb{v}}$ be the law of a time homogeneous Markov chain $X_0, X_1, X_2, \ldots$ on $\mathbb{V}$ with transition matrix $\mathbb{P} := \mathbb{D}^{-1}\mathbb{W}$ and initial condition $X_0 = \mathbb{v}$. Analogously, denote by $\mathbf{E}_{\mathbb{v}}$ the expectation with respect to this law. The hitting time to a set $\mathbb{Z}\subseteq\mathbb{V}$ is defined as $T_{\mathbb{Z}} := \inf\{k\geq 0 : X_k\in\mathbb{Z}\}$. We also extend the notation $\mathcal{N}(\mathbb{v}) := \{\mathbb{w}\in\mathbb{V} : \{\mathbb{v},\mathbb{w}\}\in\mathbb{E}\}$ to denote the vertex-neighborhood of vertex $\mathbb{v}$ in $\mathbb{T}$.

The next lemma, which builds on the results developed in Appendix A, gives an expression for the restricted Laplacian in terms of hitting probabilities of random walks. This result represents the computational device that we will use repeatedly in what follows.

Lemma 24 *Consider the setting in Section 5.1.1. Fix $\tilde{e} = (\tilde{v},\tilde{w}) \in \vec{E}$, $t\geq 1$, and let $\vec{\mathbb{T}}$ be the computation tree of depth t rooted at $\tilde{e}$, with root edge $\tilde{\mathbb{e}} = (\tilde{\mathbb{v}},\tilde{\mathbb{w}}) \in \vec{\mathbb{E}}$. For each $\mathbb{v}\in\bar{\mathbb{V}}$, $\mathbb{w}\in\mathbb{V}^{t-1}$, with $\mathbb{z} = \mathcal{N}(\mathbb{w})\setminus\mathbb{V}^t$, we have*

$$\bar{\mathbb{L}}^{-1}_{\mathbb{v}\mathbb{w}} = \frac{\mathbf{P}_{\mathbb{v}}(T_{\mathbb{w}} < T_{\mathbb{V}^t})}{[1-\mathbf{P}_{\mathbb{z}}(T_{\mathbb{w}} < T_{\mathbb{V}^t})\,\mathbf{P}_{\mathbb{w}}(X_1 = \mathbb{z})]\,\mathbb{d}_{\mathbb{w}}}.$$

Proof Fix $\mathbb{v}\in\bar{\mathbb{V}}$, $\mathbb{w}\in\mathbb{V}^{t-1}$, and let $\mathbb{z} = \mathcal{N}(\mathbb{w})\setminus\mathbb{V}^t$. Define the random variable $A := \sum_{k=0}^{T_{\mathbb{V}^t}}\mathbf{1}_{X_k=\mathbb{w}}$. From Proposition 30 we have $\bar{\mathbb{L}}^{-1}_{\mathbb{v}\mathbb{w}} = \frac{1}{\mathbb{d}_{\mathbb{w}}}\mathbf{E}_{\mathbb{w}}[A]\,\mathbf{P}_{\mathbb{v}}(T_{\mathbb{w}} < T_{\mathbb{V}^t})$. By the Markov property, performing a first step analysis for the random walk, we get $\mathbf{E}_{\mathbb{w}}[A] = 1 + \sum_{\tilde{\mathbb{z}}\in\mathcal{N}(\mathbb{w})}\mathbf{E}_{\tilde{\mathbb{z}}}[A]\mathbf{P}_{\mathbb{w}}(X_1 = \tilde{\mathbb{z}}) = 1 + \mathbf{E}_{\mathbb{z}}[A]\mathbf{P}_{\mathbb{w}}(X_1 = \mathbb{z})$, where we used that $\mathbf{E}_{\tilde{\mathbb{z}}}[A] = 0$ for any $\tilde{\mathbb{z}}\in\mathbb{V}^t$. Conditioning on the event $\{T_{\mathbb{w}} < T_{\mathbb{V}^t}\}$ and its complementary, using that $\mathbf{E}_{\mathbb{z}}[A|T_{\mathbb{w}} > T_{\mathbb{V}^t}] = 0$, we also get $\mathbf{E}_{\mathbb{z}}[A] = \mathbf{E}_{\mathbb{z}}[A|T_{\mathbb{w}} < T_{\mathbb{V}^t}]\,\mathbf{P}_{\mathbb{z}}(T_{\mathbb{w}} < T_{\mathbb{V}^t})$. By the strong Markov property $\mathbf{E}_{\mathbb{z}}[A|T_{\mathbb{w}} < T_{\mathbb{V}^t}] = \mathbf{E}_{\mathbb{w}}[A]$, so that $\mathbf{E}_{\mathbb{w}}[A] = 1 + \mathbf{E}_{\mathbb{w}}[A]\,\mathbf{P}_{\mathbb{z}}(T_{\mathbb{w}} < T_{\mathbb{V}^t})\,\mathbf{P}_{\mathbb{w}}(X_1 = \mathbb{z})$ and $\mathbf{E}_{\mathbb{w}}[A] = 1/(1-\mathbf{P}_{\mathbb{z}}(T_{\mathbb{w}} < T_{\mathbb{V}^t})\,\mathbf{P}_{\mathbb{w}}(X_1 = \mathbb{z}))$. ■

5.1.4. Proofs of Results for d-Regular Graphs (Flow Problem)

We can now present the proofs of the results concerning the flow problem contained in Lemma 6 and Lemma 9 on the characterization of the error committed by the Min-Sum algorithm as a function of time and as a function of the voltage solution $\nu^\star = L^+b$.

Proof [Proof of Lemma 6, flow problem] Let $\vec{G} = (V = \{0,\ldots,n-1\},\vec{E},W)$ be a weighted directed cycle. Consider the setting in Section 5.1.1. Fix $v\in V$ and assume that $\tilde{e} = (\tilde{v} = \rho(v),\tilde{w} = \rho(v+1)) \in \vec{E}$ (the case $e = (\tilde{w} = \rho(v+1),\tilde{v} = \rho(v))$ yields to the same result with a minus sign in front). Fix $t\geq 2$, and let $\vec{\mathbb{T}}$ be the computation tree of depth t rooted at $\tilde{e}$, with root edge $\tilde{\mathbb{e}} = (\tilde{\mathbb{v}},\tilde{\mathbb{w}}) \in \vec{\mathbb{E}}$. Label the tree vertices as $\mathbb{V} = \{0,1,\ldots,2t+1\}$, as in Figure 5. Clearly, $\tilde{\mathbb{v}} = t$, $\tilde{\mathbb{w}} = t+1$, $\mathbb{V}^{t-1} = \{1,2t\}$, and $\mathbb{V}^t = \{0,2t+1\}$.



Figure 5: Cycle $G = (V,E)$ (left) and computation tree $\mathbb{T} = (\mathbb{V},\mathbb{E})$ of depth t with root edge $\{\tilde{\mathbb{v}} = t, \tilde{\mathbb{w}} = t+1\} \in \mathbb{E}$ (right).

We will apply Lemma 24 to compute the quantities $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} - \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}}$, for each $\mathbb{v}\in\mathbb{V}^{t-1}$. We write the computations for $\mathbb{v} = 1$, as the case $\mathbb{v} = 2t$ follows immediately by symmetry. For convenience of notation, we write $\mathbb{W}_s$ to indicate $\mathbb{W}_{s,s+1}$. For each $s\in\mathbb{V}$, define $f_s := \mathbf{P}_s(T_1 < T_{\mathbb{V}^t})$, $p_s := \mathbf{P}_s(X_1 = s+1) = \mathbb{W}_s/\mathbb{d}_s$, $\mathbb{d}_s := \mathbb{W}_{s-1} + \mathbb{W}_s$, and $q_s := \mathbf{P}_s(X_1 = s-1) = 1-p_s$. By a first step analysis of the random walk and the Markov property, we have

$$f_s = f_{s-1}q_s + f_{s+1}p_s, \quad s\in\{2,\ldots,2t\}, \tag{11}$$

with boundary conditions $f_1 = 1$, $f_{2t+1} = 0$. The above system of equations corresponds to the classical Gambler Ruin's problem, with final states 1 and $2t+1$, see Figure 6.

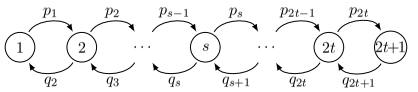


Figure 6: Gambler's Ruin.

To solve these equations, using that $1 = p_s + q_s$, we first rewrite them as $f_{s+1} - f_s = \frac{q_s}{p_s}(f_s - f_{s-1})$, for $s\in\{2,\ldots,2t\}$. Using that $q_s/p_s = \mathbb{W}_{s-1}/\mathbb{W}_s$ we get $f_{s+1} - f_s = \frac{\mathbb{W}_1}{\mathbb{W}_s}(f_2 - f_1)$. Summing these equations side by side, using that $f_1 = 1$, we get $f_s =$

$1+(f_2-1)\mathbb{W}_1(1/\mathbb{W}_1+1/\mathbb{W}_2+\cdots+1/\mathbb{W}_{s-1})$. Using that $f_{2t+1}=0$ we can solve for f_2 and finally obtain $f_s=\frac{1/\mathbb{W}_s+1/\mathbb{W}_{s+1}+\cdots+1/\mathbb{W}_{2t}}{1/\mathbb{W}_1+1/\mathbb{W}_2+\cdots+1/\mathbb{W}_{2t}}$ for any $s\in\{1,\dots,2t\}$. From Lemma 24, noticing that $\mathbb{z}=2$ in this case, we have $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}1}-\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}1}=(f_t-f_{t+1})/((1-f_2p_1)\mathbb{d}_1)$, which equals $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}1}-\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}1}=\frac{1/\mathbb{W}_t}{\mathbb{W}_0(1/\mathbb{W}_0+\cdots+1/\mathbb{W}_{2t})}$. By symmetry, we have $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}},2t}-\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}},2t}=\frac{-1/\mathbb{W}_t}{\mathbb{W}_{2t}(1/\mathbb{W}_0+\cdots+1/\mathbb{W}_{2t})}$. By Theorem 23, as $\mathbb{W}_s=W_{\sigma(s),\sigma(s+1)}$, we finally get $x^\star_{\tilde{e}}-\hat{x}^t_{\tilde{e}}=\frac{\nu^\star_{\sigma(0)}-\nu^\star_{\sigma(2t+1)}}{1/W_{\sigma(0)\sigma(1)}+\cdots+1/W_{\sigma(2t)\sigma(2t+1)}}$. The proof follows as the map $\sigma:\mathbb{V}\to V$ reads $\sigma(s)=\rho(s-t+v)$. ■

Proof [Proof of Lemma 9, flow problem] Let $\vec{G}=(V,\vec{E},W)$ be a connected d-regular graph, with $d\geq 3$, where each edge has the same weight ω. Consider the setting in Section 5.1.1. Fix $\tilde{e}=(\tilde{v},\tilde{w})\in\vec{E}$, $t\geq 3$, and let $\vec{\mathbb{T}}$ be the computation tree of depth t rooted at $\tilde{e}$, with root edge $\tilde{\mathbb{e}}=(\tilde{\mathbb{v}},\tilde{\mathbb{w}})\in\vec{\mathbb{E}}$. Fix $\mathbb{v}\in\mathbb{V}^{t-1}$, and assume that $\mathbb{v}$ is contained in the subtree with root vertex $\tilde{\mathbb{v}}$. For what follows, it is convenient to label some of the vertices in $\mathbb{V}$. Let $\mathbb{v}=0,1=\mathcal{N}(\mathbb{v})\setminus\mathbb{V}^t,2=\mathcal{N}(1)\setminus\mathbb{V}^{t-1},\dots,k+1=\mathcal{N}(k)\setminus\mathbb{V}^{t-k}$, until we reach $\tilde{\mathbb{v}}=t-1$. Label $\tilde{\mathbb{w}}=t$. See Figure 7. We will apply Lemma 24 to compute the quantity $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}}-\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}}$. To this

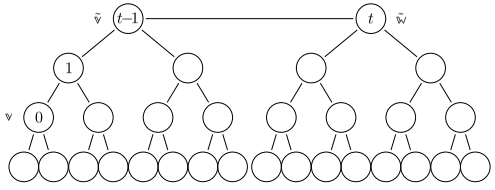


Figure 7: Computation tree $\mathbb{T}=(\mathbb{V},\mathbb{E})$ of depth $t=3$ associated to any 3-regular graph.

end, we need to compute the hitting probabilities $f_{\mathbb{w}}:=\mathbf{P}_{\mathbb{w}}(T_{\mathbb{v}}<T_{\mathbb{V}^t})$ for $\mathbb{w}\in\{1,t-1,t\}$. To do this, it is convenient to first reduce the network topology by invoking the equivalence between random walks and electrical circuits. See Peter G. Doyle (1984). The electrical circuit associated to the random walk X on $\mathbb{T}$ is constructed by assigning to each edge $\mathbb{e}=\{\mathbb{v},\mathbb{w}\}\in\mathbb{T}$ a conductor ω, which corresponds to a resistor $1/\omega$. Then, the quantities $f_{\mathbb{w}}$, $\mathbb{w}\in\mathbb{V}$, correspond to the voltages induced on $\mathbb{V}\setminus\{\mathbb{v}\cup\mathbb{V}^t\}$ when a voltage of intensity 1 is applied to $\mathbb{v}$ ($f_{\mathbb{v}}=1$), and a voltage of intensity 0 is applied to the sites in $\mathbb{V}^t$ ($f_{\mathbb{w}}=0$, $\mathbb{w}\in\mathbb{V}^t$). By repeatedly applying the standard network reduction operations for conductors in parallel and in series, it is easy to verify that, as long as the quantities $f_1,f_2,\dots,f_t$ are concerned, the original network is equivalent to the one represented in Figure 8.

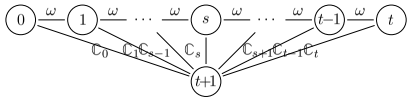


Figure 8: Reduced network.

In the reduced network, the conductance between the pairs of nodes s and $s+1$, for $s\in\{0,\dots,t-1\}$, remain ω as in the unreduced network, i.e., the computation tree. The

effect of all the nodes different from $0,\dots,t$ in the original graph is summarized in the reduced network by node $t+1$. For any $s\in\{0,\dots,t\}$, between node s and node $t+1$ there is a new conductor $\mathbb{C}_s$, which is obtained by reducing the subtree having root vertex s in the original graph. It is easy to verify that

$$\mathbb{C}_0=\omega(d-1),\qquad \mathbb{C}_s=\omega\frac{(d-2)^2}{d-1}(1+h_{s+1}),\ s\in\{1,\dots,t-1\},\qquad \mathbb{C}_t=\omega(d-2)(1+h_t),$$

with $h_s:=\frac{1}{(d-1)^s-1}$. Let $p_0:=\frac{\omega}{\omega+\mathbb{C}_0}, q_0:=1-p_0$, $p_s=q_s:=\frac{\omega}{2\omega+\mathbb{C}_s}$ for $s\in\{1,\dots,t-1\}$, $p_t:=\frac{\mathbb{C}_t}{\omega+\mathbb{C}_t}, q_t:=1-p_t$. By the Markov property we have

$$f_s=f_{s-1}q_s+f_{s+1}p_s,\quad s\in\{1,\dots,t\},\tag{12}$$

with $f_0=1, f_{t+1}=0$. This system of equations corresponds to a modification of the classical Gambler Ruin's problem, where states 0 and $t+1$ are the final states, and each intermediate state s has a probability $r_s:=1-p_s-q_s$ to reach state $t+1$. See Figure 9.

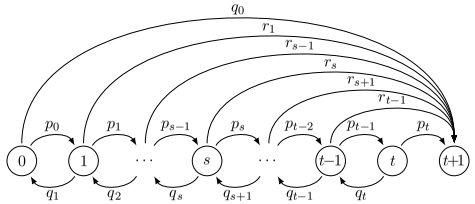


Figure 9: Gambler's Ruin on the reduced network.

Without loss of generality, we set $r_t=0$. We can easily write the solution of this system of equations in a compact form. Define $\alpha_1=\alpha_2:=1,\alpha_s:=\alpha_{s-1}-p_{s-2}q_{s-1}\alpha_{s-2}$ for $s\geq 3$. Define $\beta_1=\beta_2:=1$, $\beta_s:=\beta_{s-1}-p_{s-1}q_s\beta_{s-2}$ for $s\geq 3$. Then, as a function of f_1, we can write the solution of (12) as $f_s=\frac{1}{p_1\cdots p_{s-1}}(\alpha_sf_1-\beta_{s-1}q_1)$, for $s\in\{2,\dots,t+1\}$. In fact, it is easy to verify that this expression satisfies system (12). Using $f_{t+1}=0$ we find $f_1=q_1\beta_t/\alpha_{t+1}$ and $f_s=\frac{p_0q_1}{p_0\cdots p_{s-1}}(\frac{\alpha_s\beta_t}{\alpha_{t+1}}-\beta_{s-1})$ for $s\in\{1,\dots,t+1\}$, with $\beta_0:=0$. Lemma 24 yields the following, using that $\mathbb{d}_{\mathbb{v}}=\omega d$,

$$\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}}=\frac{f_{t-1}}{(1-f_1p_0)\mathbb{d}_{\mathbb{v}}}=\frac{1}{\omega d}\frac{q_1}{p_1\cdots p_{t-2}(\alpha_{t+1}-p_0q_1\beta_t)}(\alpha_{t-1}\beta_t-\alpha_{t+1}\beta_{t-2}),$$
$$\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}}=\frac{f_t}{(1-f_1p_0)\mathbb{d}_{\mathbb{v}}}=\frac{1}{\omega d}\frac{q_1}{p_1\cdots p_{t-2}(\alpha_{t+1}-p_0q_1\beta_t)}\frac{1}{p_{t-1}}(\alpha_t\beta_t-\alpha_{t+1}\beta_{t-1}).$$

Using the recursive formulas for α_{t+1} and β_t, and the fact that $p_t+q_t=1$, we get $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}}-\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}}=\frac{1}{\omega d}\frac{q_1}{p_1\cdots p_{t-2}(\alpha_{t+1}-p_0q_1\beta_t)}p_t(\alpha_{t-1}\beta_{t-1}-\alpha_t\beta_{t-2})$, which, as $\alpha_s\beta_s-\alpha_{s+1}\beta_{s-1}=p_{s-1}q_s(\alpha_{s-1}\beta_{s-1}-\alpha_s\beta_{s-2})$, becomes

$$\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}}-\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}}=\frac{p_t}{q_t}\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}}=\frac{1}{\omega d(d-1)^{t-1}}\frac{1}{\xi_{t-1}},\tag{13}$$

where $\xi_{t-1} := \frac{1}{(d-1)^{t-1}} \frac{\alpha_{t+1} - p_0 q_1 \beta_t}{q_1 \cdots q_{t-1} p_t}$ is defined by $\xi_s = \frac{p_s}{q_s p_{s+1}(d-1)} \xi_{s-1} - \frac{p_{s-1} p_s q_{s+1}}{q_{s-1} q_s p_{s+1} (d-1)^2} \xi_{s-2}$ for $s \in \{2, \dots, t-1\}$, with $\xi_0 := \frac{p_0 q_1}{p_1}$ and $\xi_1 = \frac{1}{d-1} \frac{1 - p_0 q_1 - p_1 q_2}{q_1 p_2}$. Plugging in the definitions of the transition probabilities, recalling the quantities defined in Proposition 8, it is immediate to verify that $\delta_s \equiv \xi_s$ for $s \in \{0, \dots, t-2\}$ and $c_{d,t} \equiv \xi_{t-1}$. By symmetry, it is clear that $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\tilde{\mathbb{v}}} - \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\tilde{\mathbb{w}}} = \frac{1}{\omega d (d-1)^{t-1}} \frac{1}{c_{d,t}}$ holds for *any* $\mathbb{v} \in \mathbb{V}^{t-1}$ that is contained in the subtree of $\mathbb{T}$ with root vertex $\tilde{\mathbb{v}}$. On the other hand, again by symmetry, for any $\mathbb{v} \in \mathbb{V}^{t-1}$ contained in the subtree of $\mathbb{T}$ with root vertex $\tilde{\mathbb{w}}$ we have $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} - \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}} = -\frac{1}{\omega d (d-1)^{t-1}} \frac{1}{c_{d,t}}$. Theorem 23 yields

$$x^\star_{\tilde{e}} - \hat{x}^t_{\tilde{e}} = \omega^2 \sum_{\mathbb{v} \in \mathbb{V}^{t-1}} (\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} - \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}}) \sum_{\mathbb{w} \in \mathbb{V}^t} \nu^\star_{\sigma(\mathbb{w})} = \frac{\omega}{d(d-1)^{t-1}} \frac{1}{c_{d,t}} \sum_{v \in V} \nu^\star_v (\#^t_{\tilde{v} \to v, \tilde{w}} - \#^t_{\tilde{w} \to v, \tilde{v}}),$$

where $\#^t_{\tilde{v} \to v, \tilde{w}}$ is the number of non-backtracking paths in the original undirected graph $G = (V, E)$ that connect node $\tilde{v}$ to node v in t time steps, where the first step is different from $\tilde{w}$. As the total number of non-backtracking paths after t steps is $d(d-1)^{t-1}$, we have $\vec{\mathbf{P}}_{\tilde{v}}(Y_1 \neq \tilde{w}, Y_t = v) = \#^t_{\tilde{v} \to v, \tilde{w}} / (d(d-1)^{t-1})$. The proof follows as $\vec{\mathbf{P}}_{\tilde{v}}(Y_1 \neq \tilde{w}, Y_t = v) = \vec{\mathbf{P}}_{\tilde{v}}(Y_t = v | Y_1 \neq \tilde{w}) \vec{\mathbf{P}}_{\tilde{v}}(Y_1 \neq \tilde{w})$, and $\vec{\mathbf{P}}_{\tilde{v}}(Y_1 \neq \tilde{w}) = (d-1)/d$. ■

5.2. Voltage Problem, Algorithm 5

We now give an overview of the results in the voltage case. The main framework for these results is the same as the one used in the previous section to investigate the behavior of the Min-Sum algorithm applied to the flow problem. For this reason, as outlined at the beginning of Section 5, here we only discuss in details the parts of the proofs that present major differences compared to the flow problem.

Henceforth we consider Algorithm 3 when the initial messages are parametrized by $\{W^0_{e \to v} = W_{vw}, e = \{v, w\}\}, \{h^0_{e \to v} = p_{e \to v}\}$, for a certain set of real numbers $p = \{p_{e \to v}\}$. Also in this case, as far as the analysis is concerned, it is convenient to consider the form of Algorithm 2. For a given choice of perturbation parameters $p = \{p_{e \to v}\}$, define the initial messages as $\mu^0_{e \to v}(\cdot, p) : z \in \mathbb{R} \longrightarrow \mu^0_{e \to v}(z, p) = \frac{1}{2} W_{vw} z^2 + p_{e \to v} z$, where $e = \{v, w\}$, and denote by $\{\mu^s_{e \to v}(\cdot, p)\}$, $s \geq 1$, the corresponding sequence of messages generated by the Min-Sum algorithm. Denote the estimates at time t as $\hat{\nu}^t_v = \arg\min_{z \in \mathbb{R}} \mu^t_v(z, p)$, where $\mu^t_v(\cdot, p)$ is the corresponding belief function.

5.2.1. Computation Tree (Voltage Problem)

The computation tree for the voltage problem differs from the one in the flow problem due to the fact that the Min-Sum algorithms for the voltage problem propagates messages that are supported on vertices, not on edges. For this reason, the computation tree that we will now define is rooted at a vertex, not at an edge as the one defined in Section 5.1.1.

Given a vertex $\tilde{v} \in V$, the computation tree rooted at $\tilde{v}$ of depth t supports the optimization problem that is obtained by unfolding the computations involved in the Min-Sum estimate $\hat{\nu}^t(p)_{\tilde{v}}$. Formally, the computation tree is an undirected tree $\mathbb{T} = (\mathbb{V}, \mathbb{E})$ where each vertex in $\mathbb{V}$ is mapped to a vertex in V through a map $\sigma : \mathbb{V} \to V$ that preserves the edge structure, namely, if $\mathbb{e} = \{\mathbb{v}, \mathbb{w}\} \in \mathbb{E}$ then $\sigma(\mathbb{e}) := \{\sigma(\mathbb{v}), \sigma(\mathbb{w})\} \in E$. The tree $\mathbb{T}$

is defined iteratively. Initially, at level -1, the tree is made by a single root vertex $\tilde{\mathbb{v}} \in \mathbb{V}$ corresponding to $\tilde{v}$, i.e., $\sigma(\tilde{\mathbb{v}}) = \tilde{v}$. Consecutively, for any $z \in \mathcal{N}(\tilde{v})$, a vertex $\mathbb{z}$ with $\sigma(\mathbb{z}) = z$ and an edge $\{\tilde{\mathbb{v}}, \mathbb{z}\}$ are added to the level 0 of the tree. For the remaining $t-1$ levels, the following procedure is repeated. The leaves in the tree are examined. Given a leaf $\mathbb{v}$ with $\sigma(\mathbb{v}) = v$ that is connected to a vertex $\mathbb{w}$ with $\sigma(\mathbb{w}) = w$, for any $z \in \mathcal{N}(v) \setminus w$, a vertex $\mathbb{z}$ with $\sigma(\mathbb{z}) = z$ and an edge $\{\mathbb{v}, \mathbb{z}\}$ are added to the next level of the tree. Figure 10 gives an example. We denote the set of vertices and edges in the k-th level of the tree respectively by $\mathbb{V}^k \subset \mathbb{V}$ and $\mathbb{E}^k \subset \mathbb{E}$. In what follows we also extend the usual neighborhood notation to vertices and edges in the graph $\mathbb{T}$, namely, $\partial \mathbb{v}$ is the set of edges in $\mathbb{T}$ that are connected to node $\mathbb{v}$, and $\partial \mathbb{e}$ is the set of vertices in $\mathbb{T}$ that are connected to edge $\mathbb{e}$.

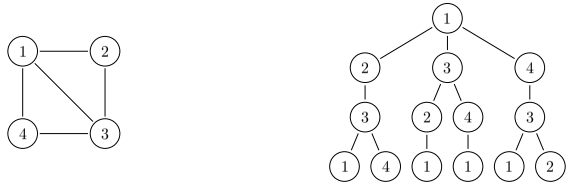


Figure 10: Graph $G = (V = \{1, 2, 3, 4\}, E)$ (left) and computation tree $\mathbb{T} = (\mathbb{V}, \mathbb{E})$ of depth $t = 2$ with root edge corresponding to $1 \in V$ (right). Vertices in $\mathbb{T}$ are labeled by the corresponding vertices in G.

The t-th (last) level of the tree is of particular relevance as it supports the initial conditions of the Min-Sum algorithm, as we are now about to describe. Let $\mathbb{W} \in \mathbb{R}^{\mathbb{V} \times \mathbb{V}}$ be the diagonal matrix defined by $\mathbb{W}_{\mathbb{v}\mathbb{w}} := W_{\sigma(\mathbb{v})\sigma(\mathbb{w})}$ if $\{\mathbb{v}, \mathbb{w}\} \in \mathbb{E}$ and $\mathbb{W}_{\mathbb{v}\mathbb{w}} := 0$ otherwise. Let $\bar{\mathbb{V}} := \mathbb{V} \setminus \mathbb{V}^t$. Define the weighted degree of a vertex $\mathbb{v}$ by $\mathbb{d}_{\mathbb{v}} := \sum_{\mathbb{w} \in \mathbb{V}} \mathbb{W}_{\mathbb{v}\mathbb{w}}$, and let $\mathbb{D} \in \mathbb{R}^{\mathbb{V} \times \mathbb{V}}$ be the diagonal matrix defined by $\mathbb{D}_{\mathbb{v}\mathbb{v}} := \mathbb{d}_{\mathbb{v}}$. The Laplacian $\mathbb{L}$ of the graph $\mathbb{T}$ is the matrix defined as $\mathbb{L} := \mathbb{D} - \mathbb{W}$. Let $\bar{\mathbb{L}} \in \mathbb{R}^{\bar{\mathbb{V}} \times \bar{\mathbb{V}}}$ denote the submatrix of $\mathbb{L}$ that corresponds to the rows and columns associated to $\bar{\mathbb{V}}$. For a given choice of the perturbation parameters $p = \{p_{e \to v}\}$, define $\bar{\mathbb{p}}(p) \in \mathbb{R}^{\bar{\mathbb{V}}}$ by

$$\bar{\mathbb{p}}(p)_{\mathbb{v}} := \begin{cases} \sum_{\mathbb{f} \in \partial \mathbb{v} \setminus \mathbb{e}} p_{\sigma(\mathbb{f}) \to \sigma(\mathbb{v})} & \text{if } \mathbb{v} \in \mathbb{V}^{t-1}, \mathbb{e} = \partial \mathbb{v} \setminus \mathbb{E}^t, \\ 0 & \text{otherwise,} \end{cases} \tag{14}$$

and let $\bar{\mathbb{b}} \in \mathbb{R}^{\bar{\mathbb{V}}}$ defined by $\bar{\mathbb{b}}_{\mathbb{v}} := b_{\sigma(\mathbb{v})}$ for each $\mathbb{v} \in \bar{\mathbb{V}}$.

Consider the following problem supported on the computation tree $\mathbb{T}$, over $\bar{\mathbb{v}} \in \mathbb{R}^{\bar{\mathbb{V}}}$:

$$\text{minimize} \quad \frac{1}{2} \bar{\mathbb{v}}^T \bar{\mathbb{L}} \bar{\mathbb{v}} + (\bar{\mathbb{p}}(p) - \bar{\mathbb{b}})^T \bar{\mathbb{v}}. \tag{15}$$

While the Laplacian $\mathbb{L}$ is positive semi-definite, the restricted Laplacian $\bar{\mathbb{L}}$ is positive definite (Spielman, 2010). Hence, for any choice of the perturbation p the objective function in problem (15) is strictly convex and it admits a unique solution $\bar{\mathbb{v}}^\star(p)$. The relationship between the Min-Sum algorithm and the tree is made explicit by the following lemma, which can be established by inductively examining the operations of Algorithm 2.

Lemma 25 *Given the setting of this section,* $\hat{\nu}^t(p)_{\tilde{v}} = \bar{\mathbb{v}}^\star(p)_{\tilde{\mathbb{v}}}$.

5.2.2. Fix Point (Voltage Problem)

The choice $p = 0$ for the linear perturbation parameters yields the original (unperturbed) algorithm. The next theorem below shows that for a particular choice of p the Min-Sum algorithm yields the optimal solution at any time $t \geq 1$. The optimal choice of p is a function of the optimal solution $\nu^\star$ for problem (2). Recall that, by definition, $\nu^\star$ is the unique vector that satisfy $L\nu^\star = b$ and $1^T\nu^\star = 0$. Hence, component-wise, for each $v \in V$,

$$d_v\nu^\star_v - \sum_{w\in\partial v} W_{vw}\nu^\star_w - b_v = 0, \tag{16}$$

where recall that $d_v := \sum_{w\in\partial v} W_{vw}$. Given $\nu^\star$, we are now ready to define the set of parameters $p^\star = \{p^\star_{e\to v}\}$ so that the Min-Sum algorithm yields the optimal solution at any time step.

Theorem 26 *For each $e \in E$, $v \in \partial e$, let $w = \partial e \setminus v$ and define $p^\star_{e\to v} := -W_{vw}\nu^\star_w$. Then, for any $\tilde v \in V$, $t \geq 1$, we have $\hat\nu^t(p^\star)_{\tilde v} = \nu^\star_{\tilde v}$.*

Proof Fix $\tilde v \in V$, $t \geq 1$. Consider the setting in Section 5.2.1, and let $\mathbb{T}$ be the computation tree of depth t rooted at $\tilde v$. From the first order optimality conditions for problem (15) corresponding to the choice $p = p^\star$, we know that $\bar v^\star \equiv \bar v^\star(p^\star)$ is the unique vector so that

$$\begin{cases} \mathbb{d}_{\mathbb{v}}\bar v^\star_{\mathbb{v}} - \sum_{\mathbb{w}\in\partial\mathbb{v}} \mathbb{W}_{\mathbb{v}\mathbb{w}}\bar v^\star_{\mathbb{w}} - \bar{\mathbb{b}}_{\mathbb{v}} = 0, \mathbb{v} \in \bar{\mathbb{V}} \setminus \mathbb{V}^{t-1}, \\ \mathbb{d}_{\mathbb{v}}\bar v^\star_{\mathbb{v}} - \mathbb{W}_{\mathbb{v}\mathbb{w}}\bar v^\star_{\mathbb{w}} + \bar{\mathbb{p}}(p^\star)_{\mathbb{v}} - \bar{\mathbb{b}}_{\mathbb{v}} = 0, \mathbb{v} \in \mathbb{V}^{t-1}, \{\mathbb{v},\mathbb{w}\} = \partial\mathbb{v} \setminus \mathbb{E}^t. \end{cases}$$

Using (16) it is easy to check that the choice $\bar v^\star_{\mathbb{v}} = \nu^\star_{\sigma(\mathbb{v})}$ satisfies the above system of equations. The proof is concluded by Lemma 25. ∎

5.2.3. Error Characterization (Voltage Problem)

The next lemma characterizes the sensitivity of the $\tilde{\mathbb{v}}$-th component (root vertex of $\mathbb{T}$) of the optimal solution of the optimization problem (15) with respect to perturbations of the parameters $p_{e\to v}$'s, which are supported on $\mathbb{V}^{t-1}$.

Lemma 27 *Consider the setting in Section 5.2.1. Fix $\tilde v \in V$, $t \geq 1$, and let $\mathbb{T}$ be the computation tree of depth t rooted at $\tilde v$, with root edge $\tilde{\mathbb{v}}$. For any $p = \{p_{e\to v}\}$, $e \in E$ and $v \in \partial e$, we have*

$$\frac{\partial}{\partial p_{e\to v}}\bar v^\star(p)_{\tilde{\mathbb{v}}} = -\sum_{\mathbb{v}\in\mathbb{V}^{t-1}} \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}}(\mathbb{y}_{e\to v})_{\mathbb{v}},$$

where $\mathbb{y}_{e\to v} \in \mathbb{R}^{\mathbb{V}^{t-1}}$ is defined, for each $\mathbb{v} \in \mathbb{V}^{t-1}$, as $(\mathbb{y}_{e\to v})_{\mathbb{v}} := \mathbf{1}_{\sigma(\mathbb{v})=v}\mathbf{1}_{\sigma(\partial\mathbb{v}\cap\vec{\mathbb{E}}^t)=e}$.

Proof From the first order optimality conditions, we know that the unique solution of problem (15) reads $\bar v^\star(p) = \bar{\mathbb{L}}^{-1}(\bar{\mathbb{b}} - \bar{\mathbb{p}}(p))$ so that $\frac{\partial}{\partial p_{e\to v}}\bar v^\star(p) = -\bar{\mathbb{L}}^{-1}\frac{\partial}{\partial p_{e\to v}}\bar{\mathbb{p}}(p)$. Using (14) it is easy to check that $(\mathbb{y}_{e\to v})_{\mathbb{v}} = (\frac{\partial}{\partial p_{e\to v}}\bar{\mathbb{p}}(p))_{\mathbb{v}}$. ∎

As a consequence of Lemma 27, we immediately have the following characterization of the error committed by Algorithm 3 with $\{W^0_{e\to v} = W_{vw}, e = \{v,w\}\}, \{h^0_{e\to v} = 0\}$.

Theorem 28 *Consider the setting in Section 5.2.1. Fix $\tilde v \in V$, $t \geq 1$, and let $\mathbb{T}$ be the computation tree of depth t rooted at $\tilde v$, with root vertex $\tilde{\mathbb{v}} \in \mathbb{V}$. We have*

$$\nu^\star_{\tilde v} - \hat\nu^t_{\tilde v} = \sum_{\mathbb{v}\in\mathbb{V}^{t-1}} \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} \sum_{\mathbb{w}\in\mathbb{V}^t} \mathbb{W}_{\mathbb{v}\mathbb{w}}\nu^\star_{\sigma(\mathbb{w})}.$$

Proof From Theorem 26 and Lemma 27, $\nu^\star_{\tilde v} - \hat\nu^t_{\tilde v} = \hat\nu^t(p^\star)_{\tilde v} - \hat\nu^t(0)_{\tilde v} = \int_0^1 d\theta\, \frac{d}{d\theta}\bar v^\star(\theta p^\star)_{\tilde{\mathbb{v}}} = \int_0^1 d\theta \sum_{e\in E, v\in\partial e} \frac{\partial}{\partial p_{e\to v}}\bar v^\star(\theta p^\star)_{\tilde{\mathbb{v}}}\, p^\star_{e\to v} = -\sum_{\mathbb{v}\in\mathbb{V}^{t-1}} \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} \sum_{e\in E, v\in\partial e}(\mathbb{y}_{e\to v})_{\mathbb{v}}\, p^\star_{e\to v}$, by the fundamental theorem of calculus and the chain rule of differentiation. From Theorem 26 we get $p^\star_{e\to v} := -W_{vw}\nu^\star_w$ for $e = \{v,w\}$, and $\sum_{e\in E, v\in\partial e}(\mathbb{y}_{e\to v})_{\mathbb{v}}\, p^\star_{e\to v} = -\sum_{\mathbb{w}\in\mathbb{V}^t} \mathbb{W}_{\mathbb{v}\mathbb{w}}\nu^\star_{\sigma(\mathbb{w})}$. ∎

Lemma 24 for the flow case holds also in the present case, upon referring the notation $\mathbf{P}_{\mathbb{v}}$, $\mathbf{E}_{\mathbb{v}}$, etc. to the Markov chain defined on the computation tree for the voltage case.

5.2.4. Proofs of Results for d-Regular Graphs (Voltage Problem)

We can now present the outline of the proofs for the results concerting the voltage problem in Lemma 6 and Lemma 9 in Section 4. Here we assume that the reader is already familiar with the proofs for the flow case.

Proof [Proof of Lemma 6, voltage problem] Let $G = (V = \{0,\dots,n-1\}, E, W)$ be a weighted cycle. Consider the setting in Section 5.2.1. Fix $v \in V$, $t \geq 2$, and let $\mathbb{T}$ be the computation tree of depth t rooted at $\tilde v$, with root vertex $\tilde{\mathbb{v}}$, i.e., $\sigma(\tilde{\mathbb{v}}) = \tilde v$. Label the vertices of the computation tree as $\mathbb{V} = \{0,1,\dots,2t+2\}$, as in Figure 11. Clearly, $\tilde{\mathbb{v}} = t+1$, $\mathbb{V}^{t-1} = \{1, 2t+1\}$, and $\mathbb{V}^t = \{0, 2t+2\}$.



Figure 11: Ring $G = (V, E)$ (left) and computation tree $\mathbb{T} = (\mathbb{V}, \mathbb{E})$ of depth t with root vertex $\tilde{\mathbb{v}} = t+1 \in \mathbb{V}$ (right).

Given this set up for the computation tree, the remaining of the proof follows exactly the steps of the proof in the flow case. Using the analogue of Lemma 24 for the voltage case, we can compute the quantities $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}}$, for each $\mathbb{v} \in \mathbb{V}^{t-1}$, by solving a system of equations like (11). If we write $\mathbb{W}_s$ to indicate $\mathbb{W}_{s,s+1}$, we find that $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}1} = \frac{1/\mathbb{W}_{t+1}+\cdots+1/\mathbb{W}_{2t+1}}{\mathbb{W}_0(1/\mathbb{W}_0+\cdots+1/\mathbb{W}_{2t+1})}$, and, by symmetry, $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}},2t+1} = \frac{1/\mathbb{W}_0+\cdots+1/\mathbb{W}_t}{\mathbb{W}_{2t+1}(1/\mathbb{W}_0+\cdots+1/\mathbb{W}_{2t+1})}$. By Theorem 28 we get $\nu^\star_{\tilde v} - \hat\nu^t_{\tilde v} = \sum_{\mathbb{v}\in\mathbb{V}^{t-1}} \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{v}}\mathbb{v}} \sum_{\mathbb{w}\in\mathbb{V}^t} \mathbb{W}_{\mathbb{v}\mathbb{w}}\nu^\star_{\sigma(\mathbb{w})} = \frac{1/\mathbb{W}_{t+1}+\cdots+1/\mathbb{W}_{2t+1}}{1/\mathbb{W}_0+\cdots+1/\mathbb{W}_{2t+1}}\nu^\star_{\sigma(0)} + \frac{1/\mathbb{W}_0+\cdots+1/\mathbb{W}_t}{1/\mathbb{W}_0+\cdots+1/\mathbb{W}_{2t+1}}\nu^\star_{\sigma(2t+2)}$, and the proof follows using $\mathbb{W}_s = W_{\sigma(s),\sigma(s+1)}$ and that $\sigma : \mathbb{V} \to V$ reads $\sigma(v) = \rho(s-t-1+\tilde v)$. ∎

Proof [Proof of Lemma 9, voltage problem] Let $G = (V, E, W)$ be a connected d-regular graph, with $d \geq 3$, where each edge has the same weight ω. Consider the setting in Section 5.2.1. Fix $\tilde{w} \in V$, $t \geq 3$, and let $\mathbb{T}$ be the computation tree of depth t rooted at $\tilde{w}$, with root vertex $\tilde{\mathbb{w}} \in \mathbb{V}$. Fix $\mathbb{v} \in \mathbb{V}^{t-1}$. For what follows, it is convenient to label some of the vertices in $\mathbb{V}$. Let $\mathbb{v} = 0, 1 = \mathcal{N}(\mathbb{v}) \setminus \mathbb{V}^t, 2 = \mathcal{N}(1) \setminus \mathbb{V}^{t-1}, \ldots, k+1 = \mathcal{N}(k) \setminus \mathbb{V}^{t-k}$, until we reach $\tilde{\mathbb{w}} = t$. See Figure 12. Given this set up for the computation tree, the remaining

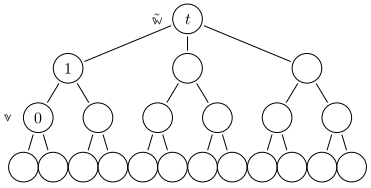


Figure 12: Computation tree $\mathbb{T} = (\mathbb{V}, \mathbb{E})$ of depth $t = 2$ associated to any given d-regular graph, with $d = 3$.

of the proof follows exactly the steps of the proof in the flow case. Using the analogue of Lemma 24 for the voltage case, we can compute $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{w}}$ by solving a system of equations of the form (12) supported on the a reduced network that is exactly like the one in Figure 8. The only difference is that now $\mathbb{C}_t = \omega(d-2)(1+h_{t+1})$, while all other effective conductances $\mathbb{C}_s$, for $s \in \{0, \ldots, t-1\}$, read exactly as for the flow problem. Rearranging (13) we get $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{w}} = \frac{1}{\omega d(d-1)^{t-1}} \frac{q_t}{p_t} \frac{1}{\xi_{t-1}}$, where $\xi_{t-1} := \frac{1}{(d-1)^{t-1}} \frac{\alpha_{t+1} - p_0 q_1 \beta_t}{q_1 \cdots q_{t-1} p_t}$ is defined as in the flow case by $\xi_s = \frac{p_s}{q_s p_{s+1}(d-1)} \xi_{s-1} - \frac{p_{s-1} p_s q_{s+1}}{q_{s-1} q_s p_{s+1}(d-1)^2} \xi_{s-2}$ for $s \in \{2, \ldots, t-1\}$, with $\xi_0 := \frac{p_0 q_1}{p_1}$ and $\xi_1 = \frac{1}{d-1} \frac{1 - p_0 q_1 - p_1 q_2}{q_1 p_2}$. Plugging in the definitions of the transition probabilities, noticing that p_t and q_t differ from the analogous quantities defined in the flow case due to the new definition of $\mathbb{C}_t$, recalling Proposition 8, it is immediate to verify that $\delta_s \equiv \xi_s$ for $s \in \{0, \ldots, t-2\}$ and $\frac{d-1}{(d-2)(1+h_{t+1})} b_{d,t} \equiv \xi_{t-1}$, so that $\bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{w}} = \frac{1}{\omega d(d-1)^t} \frac{1}{b_{d,t}}$. Theorem 28 yields

$$\nu^\star_{\tilde{w}} - \hat{\nu}^t_{\tilde{w}} = \sum_{\mathbb{v} \in \mathbb{V}^{t-1}} \bar{\mathbb{L}}^{-1}_{\tilde{\mathbb{w}}\mathbb{v}} \sum_{\mathbb{w} \in \mathbb{V}^t} \mathbb{W}_{\mathbb{v}\mathbb{w}} \nu^\star_{\sigma(\mathbb{w})} = \frac{1}{d(d-1)^t} \frac{1}{b_{d,t}} \sum_{v \in V} \nu^\star_v \, \#^{t+1}_{\tilde{w} \to v}.$$

where $\#^{t+1}_{\tilde{w} \to v}$ is the number of non-backtracking paths in the original undirected graph $G = (V, E)$ that connect node $\tilde{w}$ to node v in $t+1$ time steps. The proof is easily concluded noticing that the total number of non-backtracking paths after $t+1$ steps is $d(d-1)^t$, so that $\vec{\mathbf{P}}_{\tilde{w}}(Y_{t+1} = v) = \#^{t+1}_{\tilde{w} \to v} / (d(d-1)^t)$. ∎

Acknowledgments

We would like to acknowledge support for this project from the National Science Foundation (NSF grant ECCS-1609484).

Appendix A. Inverse of Restricted Laplacians and Hitting Times

In this appendix we establish a standalone general connection between the inverse of restricted Laplacian matrices—which are obtained by removing certain rows and columns from Laplacians matrices—and hitting times of random walks on graphs. This connection (Proposition 30 below) is used repeatedly in Section 5 to prove the error characterization of the min-sum algorithm.

Let $G = (V, E, W)$ be a simple (i.e., no self-loops, and no multiple edges), connected, undirected, weighted graph, where to each edge $\{v, w\} \in E$ is associated a positive weight $W_{vw} = W_{wv} > 0$, and $W_{vw} = 0$ if $\{v, w\} \notin E$. Let D be a diagonal matrix with entries $d_v = D_{vv} = \sum_{w \in V} W_{vw}$ for each $v \in V$. Let $L := D - W$ be the Laplacian matrix for G. Henceforth, for each $v \in V$, let $\mathbf{P}_v$ be the law of a time homogeneous Markov chain $X_0, X_1, X_2, \ldots$ on V with transition matrix $P = D^{-1}W$ and initial condition $X_0 = v$. Analogously, denote by $\mathbf{E}_v$ the expectation with respect to this law. The hitting time to a set $Z \subseteq V$ is defined as $T_Z := \inf\{k \geq 0 : X_k \in Z\}$. Let $Z \subseteq V$ be fixed, and define $\bar{W}$ and $\bar{D}$ as the matrix obtained by removing the rows and columns associated to Z from W and D, respectively. Let $\bar{V} := V \setminus Z$, $\bar{E} := E \setminus \{\{u, v\} \in E : u \in Z \text{ or } v \in Z\}$ and consider the graph $\bar{G} := (\bar{V}, \bar{E})$. The quantity $\bar{L} := \bar{D} - \bar{W}$ represents a restricted Laplacian—sometimes also called grounded Laplacian. Let $\bar{P} := \bar{D}^{-1}\bar{W}$ be the transition matrix of the transient part of the killed random walk that is obtained from X by adding cemeteries at the sites in Z. Creating cemeteries at the sites in Z means modifying the walk X so that each site in Z becomes a recurrent state, i.e., once the walk is in state $\bar{z} \in Z$ it will go back to $\bar{z}$ with probably 1. This is clearly done by replacing the $\bar{z}$-th row of P by a row with zeros everywhere but in the $\bar{z}$-th coordinate, where the entry is equal to 1.

The relation between the transition matrix $\bar{P}$ of the killed random walk and the law of the random walk X itself is made explicit in the next proposition.

Proposition 29 *For any $v, w \in \bar{V}$, $k \geq 0$, we have $\bar{P}^k_{vw} = \mathbf{P}_v(X_k = w, T_Z > k)$.*

Proof Proof is by induction. Clearly, for any $v, w \in \bar{V}$, we have $\mathbf{P}_v(X_0 = w, T_Z > 0) = \mathbf{P}_v(X_0 = w) = \mathbf{1}_{v=w} = \bar{P}^0_{vw}$, which proves the statement for $k = 0$ ($\mathbf{1}_S$ is the indicator function defined as $\mathbf{1}_S = 1$ if statement S is true, $\mathbf{1}_S = 0$ otherwise). Assume that the statement holds for any time $i \geq 0$ up to $k > 0$. By the properties of conditional expectation, as $\{T_Z > k+1\} = \{X_0 \notin Z, \ldots, X_{k+1} \notin Z\}$, we have $\mathbf{P}_v(X_{k+1} = w, T_Z > k+1) = \mathbf{E}_v[\mathbf{P}_v(X_{k+1} = w, T_Z > k+1 | X_0, \ldots, X_k)] = \mathbf{E}_v[\mathbf{1}_{\{X_0 \notin Z, \ldots, X_k \notin Z\}} \mathbf{P}_v(X_{k+1} = w, X_{k+1} \notin Z | X_0, \ldots, X_k)]$ for any $v, w \in \bar{V}$. By the Markov property, on the event $\{X_k \notin Z\}$, we have $\mathbf{P}_v(X_{k+1} = w, X_{k+1} \notin Z | X_0, \ldots, X_k) = \mathbf{P}_{X_k}(X_1 = w, X_1 \notin Z) = \bar{P}_{X_k w}$, so that by the induction hypothesis we have $\mathbf{P}_v(X_{k+1} = w, T_Z > k+1) = \mathbf{E}_v[\mathbf{1}_{\{T_Z > k\}} \bar{P}_{X_k w}] = \sum_{u \in V \setminus Z} \mathbf{P}_v(X_k = u, T_Z > k) \bar{P}_{uw} = \bar{P}^{k+1}_{vw}$, which proves the statement for $k+1$. ∎

We can now relate the inverse of the reduced Laplacian $\bar{L}$ with the hitting times of the original random walk X. The following result represents the main computational tool used in Section 5 to establish decay of correlation in the computation tree that supports the operations of the min-sum algorithm over time.

Proposition 30 *For $v, w \in \bar{V}$, $\bar{L}^{-1}_{vw} = \bar{L}^{-1}_{ww} \mathbf{P}_v(T_w < T_Z)$, and $\bar{L}^{-1}_{ww} = \frac{1}{d_w} \mathbf{E}_w \left[\sum_{k=0}^{T_Z} \mathbf{1}_{X_k = w} \right]$.*

Proof Let us first assume that $\bar{G}$ is connected. The matrix $\bar{P}$ is clearly sub-stochastic. Then $\bar{P}$ is irreducible (in the sense of Markov chains, i.e., for each $v, w \in \bar{V}$ there exists t to that $\bar{P}^k_{vw} \neq 0$) and the spectral radius of $\bar{P}$ is strictly less than 1 (see Corollary 6.2.28 in Horn and Johnson (1986), for instance), so that the Neumann series $\sum_{k=0}^{\infty} \bar{P}^k$ converges. The Neumann series expansion for $\bar{L}^{-1}$ yields $\bar{L}^{-1} = \sum_{k=0}^{\infty}(I - \bar{D}^{-1}\bar{L})^k\bar{D}^{-1} = \sum_{k=0}^{\infty} \bar{P}^k\bar{D}^{-1}$, or, $\bar{L}^{-1}_{vw} = \frac{1}{d_w}\sum_{k=0}^{\infty} \bar{P}^k_{vw}$. As $\bar{P}^k_{vw} = \mathbf{P}_v(X_k = w, T_Z > k)$ by Proposition 29, by the Monotone convergence theorem we can take the summation inside the expectation and get $\sum_{k=0}^{\infty} \bar{P}^k_{vw} = \sum_{k=0}^{\infty} \mathbf{E}_v[\mathbf{1}_{X_k=w}\mathbf{1}_{T_Z>k}] = \mathbf{E}_v[\sum_{k=0}^{T_Z-1} \mathbf{1}_{X_k=w}]$, which equals $\mathbf{E}_v[\sum_{k=0}^{T_Z} \mathbf{1}_{X_k=w}]$ as $X_{T_Z} \in Z$, and $w \notin Z$. Recall that if S is a stopping time for the Markov chain $X := X_0, X_1, X_2, \ldots$, then by the strong Markov property we know that, conditionally on $\{S < \infty\}$ and $\{X_S = w\}$, the chain $X_S, X_{S+1}, X_{S+2}, \ldots$ has the same law as the Markov chain $Y := Y_0, Y_1, Y_2, \ldots$ with transition matrix P and initial condition $Y_0 = w$, and Y is independent of $X_0, \ldots, X_S$. The hitting times T_w and T_Z are two stopping times for X, and so is their minimum $S := T_w \wedge T_Z$. As either $X_S = w$ or $X_S \in Z$, we have $\mathbf{E}_v[\sum_{k=0}^{T_Z} \mathbf{1}_{X_k=w}] = \mathbf{E}_v[\sum_{k=0}^{T_Z} \mathbf{1}_{X_k=w}|X_S = w]\mathbf{P}_v(X_S = w)$, where we used that, conditionally on $\{X_S \in Z\} = \{T_w > k_Z\}$, clearly $\sum_{k=0}^{T_Z} \mathbf{1}_{X_k=w} = 0$. Conditionally on $\{X_S = w\} = \{T_w < T_Z\} = \{S = T_w\}$, we have $T_Z = S + \inf\{k \geq 0 : X_{S+k} \in Z\}$, and the strong Markov property yields (note that the event $\{S < \infty\}$ has probability one from any starting point, as the graph G is connected by assumption so that the Markov chain will almost surely eventually hit either w or a site in Z) $\mathbf{E}_v[\sum_{k=0}^{T_Z} \mathbf{1}_{X_k=w}|X_S = w] = \mathbf{E}_v[\sum_{k=0}^{\inf\{k\geq 0:X_{S+k}\in Z\}} \mathbf{1}_{X_{S+k}=w}|X_S = w] = \mathbf{E}_w[\sum_{k=0}^{T_Z} \mathbf{1}_{X_k=w}]$. Putting everything together we have $\bar{L}^{-1}_{vw} = \frac{1}{d_w}\mathbf{E}_w[\sum_{k=0}^{T_Z} \mathbf{1}_{X_k=w}]\mathbf{P}_v(T_w < T_Z)$. As $\mathbf{P}_w(T_w < T_Z) = 1$, clearly $\bar{L}^{-1}_{ww} = \frac{1}{d_w}\mathbf{E}_w[\sum_{k=0}^{T_Z} \mathbf{1}_{X_k=w}]$ so that $\bar{L}^{-1}_{vw} = \bar{L}^{-1}_{ww}\mathbf{P}_v(T_w < T_Z)$. The argument just presented extends easily to the case when $\bar{G}$ is not connected. In this case the matrix $\bar{P}$ has a block structure, where each block corresponds to a connected component and to a sub-stochastic submatrix, so the argument above can be applied to each block. ■

Appendix B. Technical Results

This appendix contains technical results that are used in the paper. The following result on quadratic optimization has been used repeatedly in the main text (see Section 2 and the proofs of Proposition 5 and Lemma 22).

Proposition 31 *Given $R \in \mathbb{R}^{m\times m}$ symmetric positive definite, $r \in \mathbb{R}^m$, $A \in \mathbb{R}^{n\times m}$, $b \in \mathbb{R}^n$ in the range of A, the unique solution of the optimization problem*

$$\begin{aligned} &\textit{minimize} && \frac{1}{2}x^T Rx + r^T x \\ &\textit{subject to} && Ax = b \end{aligned}$$

over $x \in \mathbb{R}^m$ reads $x^\star = R^{-1}A^T L^+ b + (R^{-1}A^T L^+ A - I)R^{-1}r$ with $L := AR^{-1}A^T$.

Proof The KKT optimality equations for this problem read

$$\begin{pmatrix} R & -A^T \\ A & 0 \end{pmatrix}\begin{pmatrix} x \\ \nu \end{pmatrix} = \begin{pmatrix} -r \\ b \end{pmatrix}.$$

Solving the first equation for x we have $x = R^{-1}(A^T\nu - r)$, which, upon substitution into the second equation $Ax = b$ yields a solution $\nu = L^+(AR^{-1}r + b)$, and the proof follows. ■

We now present the proof of Proposition 8 in Section 4.

Proof [Proof of Proposition 8] Fix $d \geq 3$, $t \geq 3$. As $s \to h_s$ monotonically converges to 0 exponentially fast, we expect that the function $s \to \delta_s$, hence the values of both $b_{d,t}$ and $c_{d,t}$, can be controlled by the system that is obtained by replacing h_s with 0 for any $s \geq 0$. Thus, we define $\hat{\delta}_0 := \delta_0$, $\hat{\delta}_1 := \delta_1$, and $\hat{\delta}_s := \frac{1}{d-1}(2 + \frac{(d-2)^2}{d-1})\hat{\delta}_{s-1} - \frac{1}{(d-1)^2}\hat{\delta}_{s-2}, s \geq 2$. The characteristic equation of the dynamical system $\hat{\delta}_s$, $s \geq 0$, reads $(d-1)^2\lambda^2 - (d^2-2d+2)\lambda + 1 = 0$. This equation has two distinct real solutions: $\lambda_+ = 1$ and $\lambda_- = 1/(d-1)^2 < 1$. Imposing the initial conditions for $\hat{\delta}_0$ and $\hat{\delta}_1$, we find $\hat{\delta}_s = c_+ + c_-\lambda_-^s \geq 0$ with $c_+ = \frac{d(d-1)((d-1)^2+1)-1}{d((d-1)^2-1)} > 0$ and $c_- = 1/d - c_+ < 0$.

We begin by analyzing $c_{d,t}$. To this end, define $\hat{c}_{d,t} := \frac{1}{d-1}(1 + \frac{1}{(d-2)(1+h_t)})\hat{\delta}_{t-2} - \frac{1}{(d-1)^2}\hat{\delta}_{t-3}$. First, we prove that $c_{d,t} \geq 1$. By induction it is easy to show that $(d-1)(\delta_s - \hat{\delta}_s) \geq \delta_{s-1} - \hat{\delta}_{s-1} \geq 0$ for any $s \geq 1$, so we have $c_{d,t} \geq \hat{c}_{d,t}$. As $\hat{\delta}_s \geq c_+ + c_-\lambda_-$ for any $s \geq 1$ and $\hat{\delta}_s \leq c_+$ for any $s \geq 0$, we get $\hat{c}_{d,t} \geq \frac{1}{d-1}(1 + \frac{1}{(d-2)(1+h_1)})(c_+ + c_-\lambda_-) - \frac{1}{(d-1)^2}c_+ \geq 1$. Second, we prove that $c_{d,t} \leq 1 + \varepsilon_d$. Given $\alpha := 1 + \frac{1}{(d-1)^2}$, we first show that $(d-1)(\alpha\hat{\delta}_s - \delta_s) \geq \alpha\hat{\delta}_{s-1} - \delta_{s-1} \geq 0$ for any $s \geq 1$, so that $c_{d,t} \leq \alpha\hat{c}_{d,t}$. Define $\xi_s := \alpha\hat{\delta}_s - \delta_s$. We proceed by induction. Note that $(d-1)\xi_1 \geq \xi_0 \geq 0$. For a given $s \geq 3$, assume that $(d-1)\xi_r \geq \xi_{r-1}$ for all $r \leq s-1$. We will prove that $(d-1)\xi_s \geq \xi_{s-1}$. We have $(d-1)\xi_s - \xi_{s-1} = \frac{(d-1)\xi_{s-1} - \xi_{s-2}}{d-1} + \frac{(d-2)^2}{d-1}((1+h_{s+2})\xi_{s-1} - \alpha h_{s+2}\hat{\delta}_{s-1})$. As $\hat{\delta}_{s-1} \leq c_+$ and $(d-1)^{s-1}h_{s+2} \leq h_3$, with $c_+h_3 < 1$, by the induction hypothesis we have $\xi_{s-1} \geq \frac{\xi_1}{(d-1)^{s-2}} = \frac{(\alpha-1)\delta_1}{(d-1)^{s-2}}$ and $(d-1)\xi_s - \xi_{s-1} \geq \frac{(d-2)^2}{(d-1)^s}((d-1)(\alpha-1)\delta_1 - \alpha) = 0$, which proves the induction step. As $\frac{1}{d} = c_+ + c_- \leq \hat{\delta}_s \leq c_+$, we get $c_{d,t} \leq \alpha\hat{c}_{d,t} \leq \bar{\alpha} := \alpha(\frac{c_+}{d-1}(1 + \frac{1}{d-2}) - \frac{1}{d(d-1)^2}))$. The function $d \geq 3 \to \varepsilon_d := \bar{\alpha} - 1$ is decreasing, with $\varepsilon_d \to 0$ as $d \to \infty$, and $\varepsilon_3 < 3$.

By induction it is immediate to show that $(d-1)\delta_s \geq \delta_{s-1} \geq 0$ for any $s \geq 1$, so that a direct comparison yields $b_{d,t} \leq c_{d,t}$. Finally, we prove that $b_{d,t} \geq \frac{d-2}{d-1}$. Define $\hat{b}_{d,t} := \frac{1}{(d-1)^2}(1+(d-2)(1+h_{t+1}))\hat{\delta}_{t-2} - \frac{(d-2)(1+h_{t+1})}{(d-1)^3}\hat{\delta}_{t-3}$. As $(d-1)(\delta_s - \hat{\delta}_s) \geq \delta_{s-1} - \hat{\delta}_{s-1} \geq 0$ for $s \geq 1$, we have $b_{d,t} \geq \hat{b}_{d,t}$. As $\hat{\delta}_s \geq \hat{\delta}_3 = c_+ + c_-\lambda_-^3$ for $s \geq 3$ and $\hat{\delta}_s \leq c_+$ for $s \geq 0$, by the monotonicity of h_s we get $\hat{b}_{d,t} \geq \frac{1}{(d-1)^2}(1+(d-2))(c_+ + c_-\lambda_-^3) - \frac{(d-2)(1+h_1)}{(d-1)^3}c_+ \geq \frac{d-2}{d-1}$. ■

The next proposition yields a recursive formula to compute, in the case of d-regular graphs, the probability distributions $P_v^{(t)}$ and $P_v^{(t,w)}$ (interpreted as row vectors) given by the v-th row of the matrices $P^{(t)}$ and $P^{(t,w)}$ defined in (8), respectively. For any $v \in V$, let $\mathbf{1}_v \in \mathbb{R}^V$ be defined as $(\mathbf{1}_v)_z := 1$ if $v = z$, $(\mathbf{1}_v)_z := 0$ otherwise.

Proposition 32 *Let (V, E) be a connected d-regular graph with $d \geq 2$, and let $B \in \mathbb{R}^{V\times V}$ be its adjacency matrix, that is, the symmetric matrix defined by $B_{vw} := 1$ if $\{v, w\} \in E$,*

$B_{vw} := 0$ *otherwise. For any* $v \in V$ *we have*

$$P_v^{(1)} = \frac{\mathbf{1}_v^T B}{d}, \quad P_v^{(2)} = \frac{P_v^{(1)} B - \mathbf{1}_v^T}{d-1}, \quad P_v^{(t)} = \frac{P_v^{(t-1)} B - P_v^{(t-2)}}{d-1} \; t \geq 3,$$

Given $w \in V$ *such that* $\{v, w\} \in E$*, we have that* $P_v^{(t,w)}$ *satisfies the same recursion as above with initial condition* $P_v^{(1,w)} = (\mathbf{1}_v^T B - \mathbf{1}_w^T)/(d-1)$.

Proof We first prove the recursion for $P_v^{(t,w)}$. Fix $\{v, w\} \in E$, and let $\widetilde{\#}_z^t$ be the number of non-backtracking paths that connect node v to node z in t steps, where the first step is different from w. By definition, B_{vz} equals the number of paths that connect node v to node z in a single step. Clearly, $\widetilde{\#}_z^1 = B_{vz} - (\mathbf{1}_w)_z$, i.e., in row-vector form, $\widetilde{\#}^1 = \mathbf{1}_v^T B - \mathbf{1}_w^T$. It is immediate to verify that $\widetilde{\#}^2 = \widetilde{\#}^1 B - (d-1)\mathbf{1}_v^T$, as there are $d-1$ paths that can backtrack to v at time $t = 2$. For $t \geq 3$ we have $\widetilde{\#}^t = \widetilde{\#}^{t-1} B - (d-1)\widetilde{\#}^{t-2}$. In fact, $(\widetilde{\#}^{t-1} B)_z$ equals the number of paths that go from v to z in t steps such that the first move is different from w and the first $t-1$ steps are non-backtracking. To count only the paths that are non-backtracking for all t steps, we need to subtract from $(\widetilde{\#}^{t-1} B)_z$ the number of paths that backtrack at the t-th step, which are the ones that are at z both at step $t-2$ ant t. There are $(d-1)\widetilde{\#}_z^{t-2}$ of these paths, as vertex z has d neighbors but only $d-1$ moves are allowed at the $(t-1)$-th step due to the non-backtracking nature of the paths up to step $t-1$. The proof follows noticing that the total number of non-backtracking after t steps is $d(d-1)^{t-1}$, so that $\vec{\mathbf{P}}_v(Y_1 \neq w, Y_t = z) = \widetilde{\#}_z^t/(d(d-1)^{t-1})$ and $P_{vz}^{(t,w)} = \vec{\mathbf{P}}_v(Y_t = z | Y_1 \neq w) = \vec{\mathbf{P}}_v(Y_1 \neq w, Y_t = z)/\vec{\mathbf{P}}_v(Y_1 \neq w) = \widetilde{\#}_z^t/(d-1)^t$, $\vec{\mathbf{P}}_v(Y_1 \neq w) = (d-1)/d$. The recursion for $P_v^{(t)}$ is proved analogously. Let $\#_z^t$ be the number of non-backtracking paths that connect node v to node z in t steps. Clearly, $\#^1 = \mathbf{1}_v^T B$, $\#^2 = \#^1 B - d\mathbf{1}_v^T$, and $\#^t = \#^{t-1} B - (d-1)\#^{t-2}$. The proof follows as $P_{vz}^{(t)} = \vec{\mathbf{P}}_v(Y_t = z) = \#_z^t/(d(d-1)^{t-1})$. ■

References

Emmanuel Abbe. Community detection and stochastic block models. *Foundations and Trends® in Communications and Information Theory*, 14(1-2):1–162, 2018.

Jean Barbier, Florent Krzakala, Nicolas Macris, Léo Miolane, and Lenka Zdeborová. Optimal errors and phase transitions in high-dimensional generalized linear models. In *COLT*, volume 75 of *Proceedings of Machine Learning Research*, pages 728–731. PMLR, 2018.

C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo-codes. 1. In *Communications, 1993. ICC '93 Geneva. Technical Program, Conference Record, IEEE International Conference on*, volume 2, pages 1064–1070 vol.2, 1993.

A. Braunstein, M. Mézard, and R. Zecchina. Survey propagation: An algorithm for satisfiability. *Random Struct. Algorithms*, 27(2):201–226, 2005.

Michael B. Cohen, Rasmus Kyng, Gary L. Miller, Jakub W. Pachocki, Richard Peng, Anup B. Rao, and Shen Chen Xu. Solving sdd linear systems in nearly mlog1/2n time. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC '14, pages 343–352. ACM, 2014.

David L. Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009.

R.G. Gallager. *Low-density Parity-check Codes*. M.I.T. Press research monographs. M.I.T. Press, 1963.

David Gamarnik, Devavrat Shah, and Yehua Wei. Belief propagation for min-cost network flow: Convergence and correctness. *Operations Research*, 60(2):410–428, 2012.

Hans-Otto Georgii. *Gibbs measures and phase transitions*, volume 9 of *de Gruyter Studies in Mathematics*. Walter de Gruyter & Co., Berlin, second edition, 2011.

Roger A. Horn and Charles R. Johnson, editors. *Matrix Analysis*. Cambridge University Press, New York, NY, USA, 1986.

Jonathan A. Kelner, Lorenzo Orecchia, Aaron Sidford, and Zeyuan Allen Zhu. A simple, combinatorial algorithm for solving sdd systems in nearly-linear time. *CoRR*, abs/1301.6628, 2013.

I. Koutis, G. L. Miller, and R. Peng. A nearly-m log n time solver for sdd linear systems. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 590–598, 2011.

Rasmus Kyng and Sushant Sachdeva. Approximate Gaussian elimination for Laplacians - fast, sparse, and simple. *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 573–582, 2016.

David Asher Levin, Yuval Peres, and Elizabeth Lee Wilmer. *Markov chains and mixing times*. Providence, R.I. American Mathematical Society, 2009.

Dmitry M. Malioutov, Jason K. Johnson, and Alan S. Willsky. Walk-sums and belief propagation in gaussian graphical models. *J. Mach. Learn. Res.*, 7:2031–2064, 2006.

M. Mezard, G. Parisi, and Riccardo Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297:812 – 815, 2002.

C. C. Moallemi and B. Van Roy. Consensus propagation. *IEEE Transactions on Information Theory*, 52(11):4753–4766, Nov 2006.

C. C. Moallemi and B. Van Roy. Convergence of min-sum message passing for quadratic optimization. *IEEE Transactions on Information Theory*, 55(5):2413–2423, 2009.

Ciamac C. Moallemi and Benjamin Van Roy. Convergence of min-sum message-passing for convex optimization. *Information Theory, IEEE Transactions on*, 56(4):2041–2050, 2010.

Andrea Montanari. *Graphical models concepts in compressed sensing*, pages 394–438. Cambridge University Press, 2012.

Richard Peng and Daniel A. Spielman. An efficient parallel solver for sdd linear systems. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC '14, pages 333–342, New York, NY, USA, 2014. ACM.

J. Laurie Snell Peter G. Doyle. *Random Walks and Electric Networks*, volume 22. Mathematical Association of America, 1 edition, 1984.

T. J. Richardson and R. L. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Transactions on Information Theory*, 47(2):599–618, 2001.

P. Rusmevichientong and B. Van Roy. An analysis of belief propagation on the turbo decoding graph with gaussian densities. *IEEE Transactions on Information Theory*, 47 (2):745–765, 2001.

Daniel A Spielman. Algorithms, graph theory, and linear equations in laplacian matrices. In *Proceedings of the International Congress of Mathematicians*, volume 4, pages 2698–2722, 2010.

Daniel A. Spielman. *Algorithms, graph theory, and linear equations in Laplacian matrices.*, pages 2698–2722. Hackensack, NJ: World Scientific; New Delhi: Hindustan Book Agency, 2011.

Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 81–90. ACM, 2004.

Daniel A. Spielman and Shang-Hua Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM Journal on Matrix Analysis and Applications*, 35(3):835–885, 2014.

Sekhar C. Tatikonda and Michael I. Jordan. Loopy belief propagation and Gibbs measures. In *Proc. UAI*, volume 18, pages 493–500, 2002.

Nisheeth K. Vishnoi. Lx = b Laplacian Solvers and Their Algorithmic Applications. *Foundations and Trends® in Theoretical Computer Science*, 8(1–2):1–141, 2013.

Yair Weiss and William T. Freeman. Correctness of Belief Propagation in Gaussian Graphical Models of Arbitrary Topology. *Neural Computation*, 13(10):2173–2200, 2001.

Journal of Machine Learning Research 20 (2019) 1-59 Submitted 6/17; Revised 10/18; Published 2/19

A Well-Tempered Landscape for Non-convex Robust Subspace Recovery

Tyler Maunu MAUNUT@MIT.EDU
Department of Mathematics
Massachusetts Institute of Technology
Cambridge, MA 02139

Teng Zhang TENG.ZHANG@UCF.EDU
Department of Mathematics
University of Central Florida
Orlando, FL 32816

Gilad Lerman LERMAN@UMN.EDU
School of Mathematics
University of Minnesota
Minneapolis, MN 55455

Editor: Moritz Hardt

Abstract

We present a mathematical analysis of a non-convex energy landscape for robust subspace recovery. We prove that an underlying subspace is the only stationary point and local minimizer in a specified neighborhood under a deterministic condition on a dataset. If the deterministic condition is satisfied, we further show that a geodesic gradient descent method over the Grassmannian manifold can exactly recover the underlying subspace when the method is properly initialized. Proper initialization by principal component analysis is guaranteed with a simple deterministic condition. Under slightly stronger assumptions, the gradient descent method with a piecewise constant step-size scheme achieves linear convergence. The practicality of the deterministic condition is demonstrated on some statistical models of data, and the method achieves almost state-of-the-art recovery guarantees on the Haystack Model for different regimes of sample size and ambient dimension. In particular, when the ambient dimension is fixed and the sample size is large enough, we show that our gradient method can exactly recover the underlying subspace for any fixed fraction of outliers (less than 1).

Keywords: robust subspace recovery, non-convex optimization, dimension reduction, optimization on the Grassmannian

1. Introduction

Robust subspace recovery (RSR) involves estimating a low-dimensional linear subspace in a corrupted dataset. It assumes that a portion of the given dataset lies close to or on a subspace, which we will refer to as the "underlying subspace". The other portion of the dataset is assumed to be corrupted and may lie far from the underlying subspace. In this regime, noise and corruption are separate entities: corruption refers to large and potentially arbitrary changes to a data point, while noise is a small perturbation of a data point.

©2019 Tyler Maunu, Teng Zhang and Gilad Lerman.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-324.html.

A basic method for modeling data by a low-dimensional subspace is Principal Component Analysis (PCA) (Jolliffe, 2002). PCA is popular for both reducing noise and capturing low-dimensional structure within data. However, PCA is notoriously sensitive to corrupted data and does not perform well in many regimes of the RSR problem.

Many strategies have been proposed for the RSR problem, which are reviewed in Lerman and Maunu (2018b). However, despite nice progress, many of the proposed methods have inherent issues. Perhaps the largest flaw in many existing methods is computational time: many methods require calculation of a full covariance matrix or matrix inversion. If we have a dataset of N points in $\mathbb{R}^D$, these calculations typically have complexity $O(ND^2)$ or $O(D^3)$. On the other hand, the PCA d-dimensional subspace can be calculated with complexity $O(NDd)$. Some recent proposals for RSR have complexity that scales like $O(NDd)$, but they either do not have satisfying theoretical guarantees or have extra user specified parameters. Ideally, we would like algorithms that run in $O(NDd)$ because the set of d-dimensional subspaces has dimension $O(Dd)$.

The key point of our work is the development of a computationally efficient and provably accurate method for RSR. We desire a method that has complexity $O(NDd)$ and that does not sacrifice theoretical guarantees. The method we propose involves minimization of the robust least absolute deviations energy function. Minimizing this function involves solving a non-convex optimization problem that is NP-hard.

Even though the problem is NP-hard in general, we derive conditions that ensure the energy landscape is well-behaved in a substantial neighborhood of an underlying subspace. These conditions also ensure that a geodesic gradient descent method can locally recover an underlying subspace, and the convergence of this method is linear under some slightly stronger assumptions. This linear convergence implies that this $O(NDd)$ algorithm is very efficient. Furthermore, we give the most complete discussion of recovery under various regimes of corruption and relatively broad statistical models. In particular, we show that our method is robust to very high percentages of corruption under special generative models. To our knowledge, we give the strongest guarantees on a non-convex method for RSR to date and even obtain stronger results than some convex methods.

In the rest of this section, we give some necessary background for our method and an overview of this work. First, in Section 1.1, we give some essential background to understand our approach to RSR. Then, Section 1.2 outlines the main contributions of our work. Finally, Section 1.3 summarizes the structure of the paper, and Section 1.4 discusses necessary notation.

1.1. Essential Background

Here, we briefly summarize the necessary background to understand the primary contribution of this work. First, for the rest of this paper, we assume a linear subspace setting, which means that we only consider underlying subspaces that are linear. We leave the case of affine subspaces to future work. For simplicity of discussion, we advocate centering by the geometric median for real data when the center is not known.

The essential problem of RSR is an optimization problem over the Grassmannian, which is the set of linear subspaces of a fixed dimension. The Grassmannian is a non-convex set, which makes optimization over it hard. Frequently, this leads to NP-hard or SSE-hard

formulations (Hardt and Moitra, 2013; Clarkson and Woodruff, 2015). In this paper, we denote by $G(D,d)$ the Grassmannian of linear d-dimensional subspaces in $\mathbb{R}^D$ and refer to such subspaces as d-subspaces.

It is illuminating to first outline the PCA subspace problem, since it has a similar form to our methodology. The basic formulation for the PCA subspace problem can be cast as an optimization over $G(D,d)$. For a dataset $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \subset \mathbb{R}^D$ centered at the origin, the PCA d-subspace is the solution of the least squares problem

$$\min_{L \in G(D,d)} \sum_{i=1}^{N} \|\boldsymbol{x}_i - \boldsymbol{P}_L \boldsymbol{x}_i\|_2^2, \tag{1}$$

where $\boldsymbol{P}_L$ denotes the orthogonal projection matrix onto the subspace L. As has been noted in many past works, the least squares formulation is sensitive to corrupted data.

Some have attempted to make the PCA formulation robust by considering least absolute deviations:

$$\min_{L \in G(D,d)} \sum_{i=1}^{N} \|\boldsymbol{x}_i - \boldsymbol{P}_L \boldsymbol{x}_i\|_2. \tag{2}$$

This optimization can be thought of as estimating a geometric median subspace. Some have tried to directly optimize this problem (Ding et al., 2006; Lerman and Maunu, 2018a), while others have tried to solve convex relaxations of it (McCoy and Tropp, 2011; Xu et al., 2012; Zhang and Lerman, 2014; Lerman et al., 2015).

The goal of this work is to directly analyze the energy landscape of (2) and guarantee that a non-convex gradient descent method for this energy minimization can recover an underlying subspace. This gradient descent method leads to huge gains in speed over previous convex methods but does not sacrifice accuracy in subspace recovery.

In the RSR problem setup, it is common to refer to the uncorrupted portion of the dataset as inliers, which lie on or near the underlying subspace. The case where the inliers lie on the underlying subspace is referred to as the noiseless RSR setting, while the case where the inliers lie near the underlying subspace is referred to as the noisy RSR setting. The corrupted points in the dataset are referred to as outliers, which are assumed to lie somewhere in the ambient space. Exact recovery in the noiseless setting refers to when a method outputs the underlying subspace exactly. Near recovery in the noisy setting refers to when a method outputs a good approximation of the underlying subspace, where the goodness of approximation depends on the noise level.

1.2. Contribution of This Work

As mentioned, the goal of this work is to recover an underlying subspace by directly optimizing the non-convex function in (2). To motivate why such a procedure might work, Figure 1 demonstrates the landscape of the energy function in (2) for two simulated datasets. The novelty of this paper consists of the following observation in these and certain other datasets: despite non-convexity, the energy landscape appears to exhibit basins of attraction around the underlying subspaces. In other words, the energy function decreases over $G(D,d)$ in the direction of L_* within some neighborhood. Indeed, it appears that direct minimization of the energy in a local neighborhood would yield exact recovery of the underlying subspace.

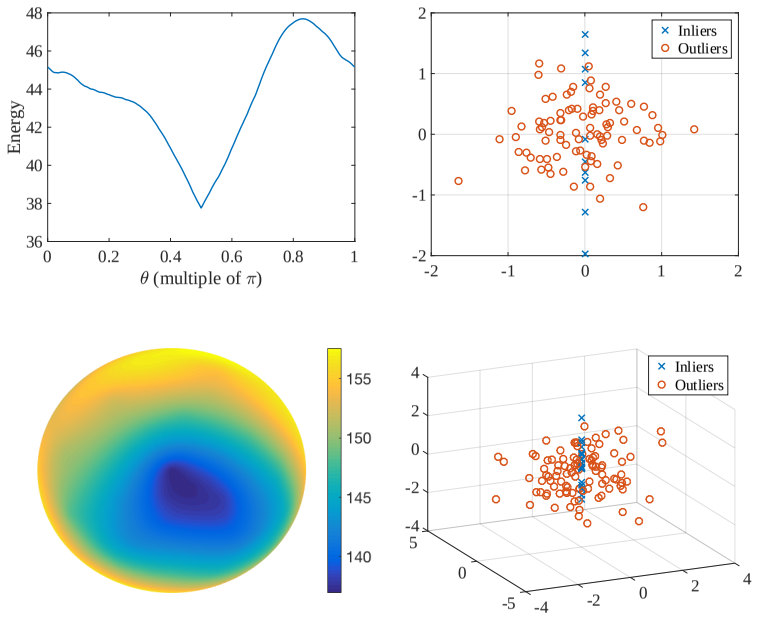


Figure 1: Demonstration of the energy landscape of (2) over $G(2,1)$ and $G(3,1)$ with simulated Gaussian data. The simulated datasets are demonstrated on the right and the corresponding energy landscape is depicted on the left. Top: In $\mathbb{R}^2$, 90 outliers are i.i.d. $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/2)$ and 10 inliers are i.i.d. $\mathcal{N}(0,1)$ along the y-axis. $G(2,1)$ is identified with a semicircle and parameterized by angle. The energy is depicted as a function of the parametrizing angle. Its global minimum is at $\pi/2$, which corresponds to the underlying line at the y-axis of $\mathbb{R}^2$. Bottom: In $\mathbb{R}^3$, 100 outliers are i.i.d. $\mathcal{N}(\boldsymbol{0}, 4\boldsymbol{I}_3/3)$ and 20 inliers are i.i.d. $\mathcal{N}(0,1)$ along the z-axis. $G(3,1)$ is identified with the top hemisphere which is flattened to a circle. The energy function is depicted by a heat map on that circle. The global minimum is at the center of the circle, which corresponds to the underlying line at the z-axis of $\mathbb{R}^3$.

It is important to emphasize that this phenomenon is inherently local. Looking at the energy plots in Figure 1, it appears that there may be other local minimizers far from the underlying subspace, and so proper initialization is quite important.

Our key contributions follow:

- We show that, under deterministic conditions, the robust energy landscape of (2) exhibits basins of attraction around an underlying subspace as seen in Figure 1. Theorems 1 and 2 formulate this result for the energy landscape of the noiseless and noisy RSR settings respectively.

- We propose a geodesic gradient descent algorithm in Section 4. Theorems 3 and 4 show that this algorithm exactly recovers the underlying subspace under the conditions of Theorem 1 with proper initialization. Theorem 3 guarantees sublinear convergence of this algorithm to the underlying subspace. With some additional, slightly stronger assumptions, Theorem 4 guarantees linear convergence to the underlying subspace using a piecewise constant step-size scheme. These results can be generalized to near recovery in the noisy RSR setting as well (see Remark 5).

- Lemma 9 guarantees that we can initialize in the correct local neighborhood using PCA under a similar deterministic condition. This yields a complete guarantee for geodesic gradient descent with PCA initialization.

- The deterministic guarantees are shown to hold for a variety of statistical models of data. In particular, we achieve competitive guarantees for recovery under the special Haystack Model. More specifically, we consider three different regimes of sample size. The first regime, $N = O(D)$, describes the scenario of a relatively small sample. The smallest ratio of inliers to outliers where exact recovery is still possible for this model among all algorithms is $d/(D-d)$, although this has only been established for algorithms of complexity $O(ND^2)$ at best (Hardt and Moitra, 2013; Zhang, 2016). We guarantee instead exact recovery for our $O(NDd)$ algorithm in this regime under a larger ratio of inliers to outliers, namely of order $O(d/\sqrt{D})$ (see Corollary 16). In the regime of larger samples, when $N = O(D^2)$, exact recovery with the even smaller ratio $O(d/\sqrt{D(D-d)})$ was previously obtained for a convex algorithm (Zhang and Lerman, 2014). Theorem 17 implies recovery with this same small fraction but in the regime $N = O(d(D-d)^2 \log(D))$. Except for this convex algorithm and our proposed algorithm, we are unaware of any other method that is guaranteed to obtain exact recovery under such a small fraction in the regime $N = O(d(D-d)^2 \log(D))$. Beyond this, in the regime of very large samples, we show that our method can exactly recover an underlying subspace with any fixed fraction of outliers, where N is at least some polynomial order of D and also depends on the ratio of inliers to outliers. This is the only efficient RSR method with such a guarantee.

We will close this section by briefly commenting on how to read the theoretical results of this paper. The first three bullets constitute the primary theoretical interest of this work, since they guarantee the usefulness of our non-convex RSR method under some fairly general conditions. In particular, the stability statistic developed in Section 3.1.1 lies at the core of much of our analysis, and many of the more complicated results are extensions of this statistic's analysis in Theorem 1. The discussion of statistical models as outlined in the fourth bullet above is merely included for interpretability of the general conditions we offer. With such models, we develop a heuristic understanding of these conditions, and, in particular, we gain insight into trade-offs between inliers and outliers. These models also allow for easier comparison of the theoretical guarantees of all RSR methods.

1.3. Paper Organization

First, in Section 2, we review previous work on the RSR problem. In Section 3, we describe deterministic conditions that ensure that the energy landscape of (2) behaves nicely around an underlying subspace. Then, in Section 4, we outline a geodesic gradient descent method on the set of subspaces. We show that this method can locally recover an underlying subspace for datasets satisfying the deterministic conditions and that the convergence rate is linear under some slightly stronger assumptions. We obtain exact recovery in the noiseless RSR setting and near recovery in the noisy RSR setting. Then, Section 5 shows that the conditions hold for certain statistical models of data. In particular, we obtain almost state-of-the-art results on recovery under the Haystack Model and also consider a range of other models. Next, Section 6 gives simulations that agree with the theoretical results of this paper. Finally, Section 7 concludes this work and discusses possible future directions.

1.4. Notation

Before presenting the results of this paper, we explain our commonly used notation. The letter L is used to refer to d-subspaces. We use bold upper case letters for matrices and bold lower case letters for column vectors. For a matrix $\boldsymbol{A}$, $\mathrm{Sp}(\boldsymbol{A})$ is the subspace spanned by the columns of $\boldsymbol{A}$, $\sigma_j(\boldsymbol{A})$ denotes the jth singular value of $\boldsymbol{A}$, and, if $\boldsymbol{A}$ is square, then $\lambda_j(\boldsymbol{A})$ denotes its jth eigenvalue. The spectral norm of $\boldsymbol{A}$ is $\|\boldsymbol{A}\|_2$, and the Euclidean 2-norm for vectors is denoted by $\|\cdot\|$. The notation $\widetilde{\boldsymbol{A}}$ denotes projection of the columns of $\boldsymbol{A}$ to the unit sphere. For $d \leq D$, the set of semi-orthogonal $D \times d$ matrices is denoted by $O(D,d) = \{\boldsymbol{V} \in \mathbb{R}^{D\times d} : \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_d\}$. For $\boldsymbol{V} \in O(D,d)$, we denote its columns by $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d$. We recall that $G(D,d)$ denotes the Grassmannian, that is, the set of d-dimensional linear subspaces of $\mathbb{R}^D$. The orthogonal projection matrix onto the subspace $L = \mathrm{Sp}(\boldsymbol{V})$ is denoted by $\boldsymbol{P}_L$, and we interchangeably use $\boldsymbol{P}_{\boldsymbol{V}} = \boldsymbol{P}_L$. The projection onto the orthogonal complement of L is $\boldsymbol{Q}_L = \boldsymbol{I} - \boldsymbol{P}_L$. We denote the largest principal angle between two subspaces L_1 and L_2 by $\theta_1(L_1, L_2)$, that is, $\theta_1(L_1, L_2) = \arccos(\sigma_d(\boldsymbol{P}_{L_1}\boldsymbol{P}_{L_2}))$. We say that an event occurs with high probability (w.h.p.) if the probability is bounded below by $1 - O(N^a)$, for some absolute constant $a > 0$. We say that an event occurs with overwhelming probability (w.o.p.) if the probability is bounded below by $1 - O(e^{-aN^b})$, for absolute constants $a, b > 0$. The notation $f(x) \lesssim g(x)$ is used to denote that $f(x) < Cg(x)$ for some absolute constant C (and the notation $\gtrsim$ is used in the same way).

2. Background and Review of Previous Work

In this section, we will review past work on the RSR problem and necessary background concepts for this work. First, Section 2.1 discusses past attempts to solve the RSR problem. Then, Section 2.2 gives the background concepts that are necessary to understand our later results.

2.1. Review of Previous Work

The most ubiquitous subspace modeling framework uses principal component analysis (Jolliffe, 2002). Its optimization problem, which is formulated in (1), is non-convex since $G(D,d)$

is non-convex. However, despite its non-convexity, this problem has a direct solution, which is calculated from the singular value decomposition of the data matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]$. This problem also has a nice energy landscape. Indeed, if the dth singular value of $\boldsymbol{X}$ is larger than the $(d+1)$st singular value, then the global minimum is unique, and there are no other local minima; otherwise, if the dth singular value is equal to the $(d+1)$st, then all local minima are globally optimal. Saddle points are also guaranteed to be sufficiently far from the global minimum, and they have explicit expressions. We discuss the PCA energy landscape further in Appendix A. These nice properties of the PCA subspace optimization are not shared by the algorithms for RSR that we discuss next.

Examples of works on RSR include Maronna (2005); Maronna et al. (2006); Ding et al. (2006); Zhang et al. (2009); Lerman and Zhang (2011); McCoy and Tropp (2011); Xu et al. (2012); Coudron and Lerman (2012); Hardt and Moitra (2013); Zhang and Lerman (2014); Goes et al. (2014); Lerman and Zhang (2014); Lerman et al. (2015); Zhang (2016); Lerman and Maunu (2018a); Cherapanamjeri et al. (2017). A comprehensive overview of this topic is given in Lerman and Maunu (2018b).

We note that this problem is distinct from what is typically called robust PCA (RPCA) (Candès et al., 2011; Zhou et al., 2010), where the corruptions occur element-wise throughout the whole data matrix rather than some samples being wholly corrupted. Algorithms for RPCA typically do not perform well in the RSR setting, and algorithms for RSR do not perform well in the RPCA setting.

RSR is inherently non-convex due to the non-convexity of $G(D, d)$. Robust versions of the PCA energy may have more complicated landscapes in general. One way of making PCA robust is to simply project the data to the unit sphere, S^{D-1}, before running PCA (Locantore et al., 1999; Maronna, 2005; Maronna et al., 2006). This deals with PCA's sensitivity to the scaling of the data and makes it easier to find directions that robustly capture variance. However, it is still not able to deal with correlated outlier directions and does not have good asymptotic guarantees even for simple models (Lerman and Maunu, 2018a).

As mentioned, another way to make PCA robust is to consider least absolute deviations (Ding et al., 2006; McCoy and Tropp, 2011; Xu et al., 2012; Zhang and Lerman, 2014; Lerman and Zhang, 2014; Lerman et al., 2015; Lerman and Maunu, 2018a). The first use of least absolute deviations in subspace modeling was the work on orthogonal regression by Osborne and Watson (1985). This was not extended to general subspace modeling until much later (Watson, 2001; Ding et al., 2006). Previous works have considered convex relaxation of this energy (McCoy and Tropp, 2011; Xu et al., 2012; Zhang and Lerman, 2014; Lerman et al., 2015). However, such convex relaxations are generally slow and may not approximate the underlying problem well. Indeed, most either have complexity $O(ND^2)$ or $O(D^3)$.

The works of Lerman and Zhang (2011, 2014) established under a certain model that an underlying subspace is recoverable by the minimizer of (2). However, they did not provide a guaranteed algorithm for minimizing this energy. The estimates of these works do not hold for small sample sizes: they only hold for large N. Lerman and Maunu (2018a) developed the FMS algorithm, which employs iteratively reweighted least squares to optimize (2). However, the FMS algorithm does not have deterministic guarantees of fast convergence or deterministic results on recovery of the underlying subspace. The FMS algorithm does

have theoretical guarantees of approximate recovery for a very special model of data, with relatively large samples. In contrast, we directly minimize (2) by gradient descent, and we provide deterministic guarantees of fast convergence and subspace recovery.

Another recent work on RSR was given by Cherapanamjeri et al. (2017), where they propose Thresholding based Outlier Robust PCA (TORP). TORP has analysis for arbitrary outliers and noise, as long as the percentage of outliers is known in advance. While the tolerance to very low percentages of arbitrary outliers is not that impressive, the noise analysis is somewhat novel. Under Gaussian noise, the authors are able to show similar sample complexity as that of PCA. One downside of this algorithm is that one must know the percentage of outliers as an input. Further, since the guarantees are only for adversarial models of outliers, there is no discussion of improved estimates when the outliers are not adversarial but instead obey a specific statistical model.

In the existing literature, only a few methods achieve the complexity bound of $O(TNDd)$, where T is iteration count. These include SPCA (Maronna, 2005), RANSAC and RandomizedFind (Hardt and Moitra, 2013; Arias-Castro and Wang, 2017), FMS (Lerman and Maunu, 2018a), and TORP (Cherapanamjeri et al., 2017). Among these, the algorithms either do not have sufficiently satisfying guarantees for recovery, or they do not have a good bound on T, or they require additional parameters. SPCA is the fastest out of these algorithms since it has $T = 1$. Here, we are slightly abusing the complexities and assuming the cost of running PCA is $O(NDd)$, despite the fact that PCA is also an iterative algorithm. We choose this convention due to the fact that many methods use PCA as a sub-routine. While SPCA is somewhat robust to arbitrary outliers, it cannot exactly recover subspaces in the presence of outliers. However, SPCA is nice since it is quite general and lacks the specialized assumptions of many methods. RANSAC requires a user to input specialized parameters, such as the consensus number and a consensus threshold. RANSAC can also only bound T in probability under certain conditions. This also goes for the analysis of RandomizedFind given by Arias-Castro and Wang (2017), along with their updated algorithm that has complexity $O(TDd)$. In many cases, though, this T can be very large. For both the RANSAC and RandomizedFind methods, recovery guarantees exist in the noiseless RSR setting under specialized assumptions, but there are no satisfying extensions of either method to noise. On the other hand, TORP (Cherapanamjeri et al., 2017) requires a user to input the percentage of outliers that is not known in general. TORP has a guarantee of linear convergence under certain conditions, but, as we mentioned earlier, it does not have satisfying guarantees for subspace recovery. FMS (Lerman and Maunu, 2018a) only has guarantees for rate of convergence and recovery for very special models of data.

One way to compare the theoretical guarantees of various methods is to assume a statistical model of data and then determine which algorithm performs best in this model. For example, one common choice of model in past works was the Haystack Model, which can be seen in Lerman et al. (2015). Another model was to assume spherically symmetric outliers, and inliers spherically symmetric on an underlying subspace (Lerman and Zhang, 2014; Lerman and Maunu, 2018a). Others have examined models with arbitrary outliers (Xu et al., 2012; Cherapanamjeri et al., 2017). In this work, after giving our general theoretical guarantees, we will show how they can be applied to a variety of statistical models of data.

This paper also fits in to the surge of recent work that has focused on non-convex optimization for many structured data problems (Dauphin et al., 2014; Hardt, 2014; Jain

et al., 2014; Ge et al., 2015; Lee et al., 2016; Arora et al., 2015; Mei et al., 2018; Ge et al., 2016; Boumal, 2016; Sun et al., 2015b,a; Ma et al., 2018). Some work has focused on non-convex optimization for robust PCA (Netrapalli et al., 2014; Yi et al., 2016; Zhang and Yang, 2017), which is a related but different problem than RSR. Others have attempted to solve non-convex versions of the RSR problem (Lerman and Maunu, 2018a; Cherapanamjeri et al., 2017), but, as we have discussed, these methods each have their own shortcomings.

This work is partially built on optimization on manifolds, and in particular there are important results on optimization over the Grassmannian manifold (Edelman et al., 1999; Absil et al., 2004). Edelman et al. (1999) develop gradient descent on the Grassmannian and give formulations for Newton's method and conjugate gradient for the Grassmannian. We discuss optimization on the Grassmannian in more detail in the next section.

Many other recent works have also focused on using optimization on the Grassmannian to solve various problems (Zhang et al., 2009; Goes et al., 2014; St. Thomas et al., 2014; Zhang and Balzano, 2016; Ye and Lim, 2016; Lim et al., 2016). The work of Zhang and Balzano (2016) examines a rank one geodesic gradient scheme for solving online PCA. Their setting is distinctly different from ours since they attempt to solve the PCA problem rather than RSR. They also only prove recovery of the PCA solution for a specific model of Gaussian noise, and no deterministic condition for global recovery is given. Further, while we assume centered data in this paper, St. Thomas et al. (2014) and Lim et al. (2016) consider estimation on the affine Grassmannian.

2.2. Review of Optimization over $G(D,d)$

The minimization in (2) involves optimization over the Grassmannian manifold. To understand the energy landscape, one must have a basic understanding of the geometry of $G(D,d)$ and how to calculate derivatives over it.

We can write the energy function in (2) in two equivalent ways. First, as a function over $G(D,d)$, we write

$$F(L;\mathcal{X}) = \sum_{\mathcal{X}} \|\boldsymbol{Q}_L \boldsymbol{x}_i\|. \tag{3}$$

On the other hand, we can represent points in $G(D,d)$ by equivalence classes of points in $O(D,d)$. For any $\boldsymbol{V} \in O(D,d)$, the subspace $\mathrm{Sp}(\boldsymbol{V})$ can be represented by the equivalence class $[\boldsymbol{V}] = \{\boldsymbol{V}\boldsymbol{R} : \boldsymbol{R} \in O(d,d)\}$. For $\boldsymbol{V} \in O(D,d)$, the energy (3) is equivalent to

$$F(\boldsymbol{V};\mathcal{X}) = \sum_{i=1}^{N} \|(\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^T)\boldsymbol{x}_i\|. \tag{4}$$

While both formulations are equivalent, we use (3) to formulate geodesic derivatives over $G(D,d)$ and the coordinate representation in (4) to calculate gradients. In the following, the geodesic derivative of (3) will be used to characterize the local landscape, and the gradient of (4) will be used to analyze the performance of the gradient descent algorithm we discuss later in Section 4.

One can measure the distance between subspaces in $G(D,d)$ using the principal angles. For a discussion of principal angles between subspaces, see Appendix B. Denoting the largest principal angle between L_0 and L_1 by $\theta_1(L_0,L_1)$, we can define a metric on $G(D,d)$

by $\mathrm{dist}(L_0,L_1) = \theta_1(L_0,L_1)$. We then define a ball on the metric space $G(D,d)$ by

$$B(L,\gamma) = \{L' \in G(D,d) : \ \theta_1(L',L) < \gamma\}.$$

We say that an element of $O(D,d)$ lies in the ball $B(L,\gamma)$ if the subspace spanned by its columns lies in $B(L,\gamma)$.

In the following, we frequently use a construction for geodesics on the Grassmannian. For a review of this construction and necessary terminology, see Appendix B or §3.2.1 of Lerman and Zhang (2014). Suppose that the interaction dimension between L_1 and L_2 is k, that is, $k = d - \dim(L_0 \cap L_1)$. Let $\theta_1,\dots,\theta_k$ be the nonzero principal angles for L_0 and L_1 (in decreasing order), and let the respective principal vectors for L_0 and L_1 be $\boldsymbol{v}_1,\dots,\boldsymbol{v}_k$ and $\boldsymbol{y}_1,\dots,\boldsymbol{y}_k$. Finally, let $\boldsymbol{u}_1,\dots,\boldsymbol{u}_k$ be a complementary orthogonal basis for L_1 with respect to L_0. We can use these to parameterize a geodesic $L(t)$ with $L(0) = L_0$ and $L(1) = L_1$, where the formula is given in (62) of Appendix B. Then, following Lerman and Zhang (2014) and Lerman and Maunu (2018a), we can calculate the directional geodesic subderivative of (3) at L_0 in the direction of L_1:

$$\frac{\mathrm{d}}{\mathrm{d}t} F(L(t);\mathcal{X})\Big|_{t=0} = -\sum_{\|\boldsymbol{Q}_0\boldsymbol{x}_i\|>0} \frac{\sum_{j=1}^{k} \theta_j (\boldsymbol{v}_j^T \boldsymbol{x}_i)(\boldsymbol{u}_j^T \boldsymbol{x}_i)}{\|\boldsymbol{Q}_{L_0}\boldsymbol{x}_i\|}. \tag{5}$$

A subderivative of (4) with respect to $\boldsymbol{V}$ is

$$\frac{\partial}{\partial \boldsymbol{V}} F(\boldsymbol{V};\mathcal{X}) = -\sum_{\|\boldsymbol{Q}_{\boldsymbol{V}}\boldsymbol{x}_i\|>0} \frac{\boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{V}}{\|\boldsymbol{Q}_{\boldsymbol{V}}\boldsymbol{x}_i\|}. \tag{6}$$

The definition of subderivative and subdifferential as we use them are given next. For a more in depth discussion of these concepts, see, for example, Clarke (1990) and Ledyaev and Zhu (2007). In both of the derivatives (5) and (6), the sum is taken over all points that do not lie in $\mathrm{Sp}(\boldsymbol{V})$. This restriction is what makes them both subderivatives. For any general function $f(x)$, a subderivative of f at x_0 is any number in the subdifferential $\partial f(x_0)$. In turn, the subdifferential of f at x_0 is the set of all numbers between the one-sided derivatives of f at x_0. For (5), the subdifferential is defined to be the set of all numbers between

$$a = \lim_{t\to 0^-} \frac{F(L(t);\mathcal{X}) - F(L(0);\mathcal{X})}{t} \text{ and } b = \lim_{t\to 0^+} \frac{F(L(t);\mathcal{X}) - F(L(0);\mathcal{X})}{t}.$$

In other words, the subdifferential is $[\min(a,b), \max(a,b)]$, which is the set of all instantaneous tangent slopes at $L(0)$. For the other case of (6), for any entry of $\boldsymbol{V}$, $\boldsymbol{V}_{ij}$, let $\boldsymbol{\Delta}$ be the matrix of all zeros except $\boldsymbol{\Delta}_{ij} = 1$. Then, the subdifferential of $F(\boldsymbol{V};\mathcal{X})$ for $\boldsymbol{V}_{ij}$ is all numbers between

$$a_{ij} = \lim_{t\to 0^-} \frac{F(\boldsymbol{V}+t\boldsymbol{\Delta};\mathcal{X}) - F(\boldsymbol{V};\mathcal{X})}{t} \text{ and } b_{ij} = \lim_{t\to 0^+} \frac{F(\boldsymbol{V}+t\boldsymbol{\Delta};\mathcal{X}) - F(\boldsymbol{V};\mathcal{X})}{t}.$$

This can be generalized to any direction $\boldsymbol{\Delta}$ with $\|\boldsymbol{\Delta}\|_F = 1$, where the subdifferential is the convex hull of the one sided derivatives. We say that the subdifferential is less than a number if all of its elements are bounded above by that number, that is,

$$\partial F(L(t);\mathcal{X})|_{t=0} < M \iff a < M \ \forall\ a \in \partial F(L(t);\mathcal{X})|_{t=0}. \tag{7}$$

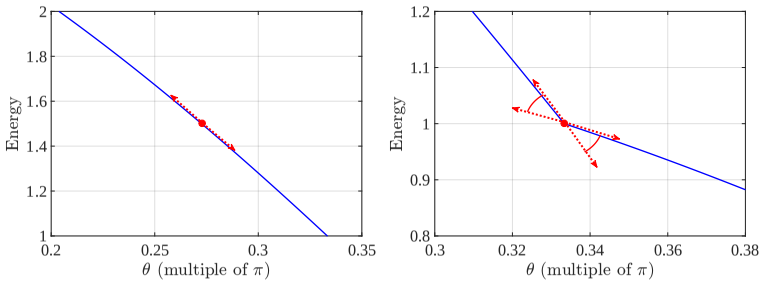


Figure 2: Demonstration of the derivative and subdifferential of the energy in (2). We assume that $d = 1$, $D = 2$ and identify $G(2,1)$ with the unit circle. The energy function in (2) is thus parameterized by angle and its graph is similar to the one in the top left sub-figure of Figure 1. In both images, the slope of the red two-sided arrow represents the magnitude of the directional geodesic derivative or subderivative over $G(2,1)$, which is the same as derivative or subderivative with respect to the representing angle. The first image shows a differentiable energy function on the given domain. In the second image, there is a value where the energy function is not differentiable. In this case, we use the subdifferential, which is the set of slopes of all lines between the acute angles formed by the two red two-sided arrows. Note that this subdifferential is bounded above by a negative number. We later prove that this property generally holds for the energy function of (2) under certain conditions.

To gain an intuition for these concepts, we display a visualization of the derivative and subdifferential for a simulated energy landscape in Figure 2. The derivative follows the standard definition from calculus on manifolds and is just the slope of the tangent line displayed on the left in Figure 2. On the other hand, at points where the function $F(L(t);\mathcal{X})$ is non-smooth at $t = 0$, we use the subdifferential instead. The extreme slopes for the subdifferential are displayed on the right in Figure 2.

In future sections, to save space, we will write the sums in (5) and (6) as $\sum_{\mathcal{X}}$ and leave the condition $\|\boldsymbol{Q}_{\boldsymbol{V}}\boldsymbol{x}_i\| > 0$ as implied. Following Section 2.5.3 of Edelman et al. (1999), to respect the geometry of the Grassmannian, the (sub)gradient of (4) is defined as

$$\nabla F(\boldsymbol{V};\mathcal{X}) = \boldsymbol{Q}_{\boldsymbol{V}}\frac{\partial}{\partial \boldsymbol{V}}F(\boldsymbol{V};\mathcal{X}). \tag{8}$$

3. A Well-Tempered Landscape for Least Absolute Deviations

We assume a dataset $\mathcal{X} = \{\boldsymbol{x}_1,\dots,\boldsymbol{x}_N\} \subset \mathbb{R}^D$ that can be partitioned into corrupted (outlier) and uncorrupted (inlier) parts. We refer to $\mathcal{X}$ as an inlier-outlier dataset, where in the coming sections, we will more rigorously define this notion in the noiseless and noisy RSR settings. We denote the subsets of inliers and outliers in $\mathcal{X}$ as $\mathcal{X}_{\text{in}}$ and $\mathcal{X}_{\text{out}}$,

respectively. The corresponding data matrices for $\mathcal{X}_{\text{in}}$ and $\mathcal{X}_{\text{out}}$ are $\boldsymbol{X}_{\text{in}}$ and $\boldsymbol{X}_{\text{out}}$, where columns represent data points.

As stated previously, the basic problem of RSR is to recover the subspace L_* from an inlier-outlier dataset. In the noiseless setting one can try to exactly recover this subspace, and in the noisy setting one may try to approximately recover it. In the latter case, this means that we wish to estimate it up to a specified approximation error, which may depend on the level of noise in the inliers. In order for this problem to be well-defined, basic assumptions must be made. Indeed, if all inliers lie at the origin, then any subspace would be a solution to the RSR problem. This issue, among others, was extensively discussed in §III-A of Lerman and Maunu (2018b). Our theoretical results for recovery will depend on a condition formulated later in this section that ensures the problem is well-defined.

First, Section 3.1 discusses some statistics that play a fundamental role in our analysis. Next, Section 3.2 uses these statistics to develop the deterministic conditions that ensure the energy landscape of (2) behaves nicely around an underlying subspace in both the noiseless and noisy RSR settings.

3.1. Landscape Statistics

Equipped with the notions laid out in Section 2.2, we are ready to define some important statistics for the landscape of (2). These statistics are inspired by those originally discussed in Lerman et al. (2015), and they are later used for our stability results in Theorems 1 and 2. We first discuss the noiseless RSR setting in Section 3.1.1 and then the noisy RSR setting in Section 3.1.2.

3.1.1. The Noiseless RSR Setting

For the noiseless setting, we assume that the inliers, $\mathcal{X}_{\text{in}} \subset \mathcal{X}$, lie on a low-dimensional linear subspace $L^* \in G(D,d)$, and the rest of the points, $\mathcal{X}_{\text{out}} = \mathcal{X} \setminus \mathcal{X}_{\text{in}}$, are in $\mathbb{R}^D \setminus \{L^*\}$. We call $\mathcal{X}$ defined in this way a noiseless inlier-outlier dataset.

The permeance of the inliers in a noiseless inlier-outlier dataset is defined as

$$\mathcal{P}(\mathcal{X}_{\text{in}}) = \lambda_d\left(\sum_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\frac{\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|}\right). \tag{9}$$

Here, $\lambda_d(\cdot)$ denotes the dth eigenvalue of a matrix. Notice that large values of $\mathcal{P}$ ensure that the inliers are well-distributed. In other words, they permeate throughout L_*.

We also define an alignment statistic for the noiseless inlier-outlier dataset $\mathcal{X}$. With some abuse of notation, we write $\nabla F(L;\mathcal{X})$ to refer to the gradient with respect to some basis of L, where the choice of basis does not matter. The alignment statistic of a set of outliers with respect to a subspace is

$$\mathcal{A}(\mathcal{X}_{\text{out}}, L) = \|\nabla F(L;\mathcal{X}_{\text{out}})\|_2. \tag{10}$$

It is not hard to show that (10) is invariant with respect to choice of basis for L. In effect, if this term is always small, then the outliers are not concentrated in any low-dimensional space. In other words, they are not aligned. In our later analysis, we use a simple and

illuminating bound for $\mathcal{A}$:

$$\mathcal{A}(\mathcal{X}_{\text{out}}, L) \leq \sqrt{N_{\text{out}}}\|\boldsymbol{X}_{\text{out}}\|_2. \tag{11}$$

The derivation for this bound is left to Appendix C. We note that this bound may be tight, but in most cases it is not.

We will define a stability statistic for a neighborhood of L_*, $B(L_*, \gamma)$. This neighborhood depends on a parameter γ, which fixes the maximum principal angle of subspaces in this neighborhood with L_*. Using the permeance and alignment defined in (9) and (10), the stability statistic of a noiseless inlier-outlier dataset is

$$\mathcal{S}(\mathcal{X}, L_*, \gamma) = \cos(\gamma)\mathcal{P}(\mathcal{X}_{\text{in}}) - \sup_{L \in B(L_*,\gamma)} \mathcal{A}(\mathcal{X}_{\text{out}}, L). \tag{12}$$

The simple condition required in most of our theoretical analysis is $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$. This essentially means that the amount that the inliers permeate the underlying subspace must be able to beat the alignment of the outliers with respect to any subspace.

Note that $\mathcal{S}(\mathcal{X}, L_*, 0) = \mathcal{P}(\mathcal{X}_{\text{in}}) - \mathcal{A}(\mathcal{X}_{\text{out}}, L_*)$ is a tighter stability condition than the one in (2.4) of Lerman et al. (2015). Indeed, the stability expression of Lerman et al. (2015), takes the form

$$\mathcal{S}_{\text{REAP}}(\mathcal{X}, L_*) = \frac{1}{4\sqrt{d}}\mathcal{P}_{\text{REAP}}(\mathcal{X}_{\text{in}}) - \mathcal{A}_{\text{REAP}}(\mathcal{X}_{\text{out}}, L_*). \tag{13}$$

Here, $\mathcal{P}_{\text{REAP}}$ and $\mathcal{A}_{\text{REAP}}$ are actually lower and upper bounds on the permeance and alignment defined in (9) and (10), respectively. This, together with the extra factor of $1/(4\sqrt{d})$, means $\mathcal{S}_{\text{REAP}}(\mathcal{X}, L_*)$ is not as tight as $\mathcal{S}(\mathcal{X}, L_*, \gamma)$ for $\gamma = 0$. The upside is that the REAPER alignment only needs to be examined at a single point L_*, whereas $\mathcal{S}(\mathcal{X}, L_*, \gamma)$ becomes hard to estimate as γ increases. It is not clear in general which statistic is tighter when $\gamma > 0$.

3.1.2. The Noisy RSR Setting

The noisy setting occurs when the inliers lie near the low-dimensional subspace rather than exactly on it. In this case, we need to be more careful with the statistics of our inlier points. For each inlier point, we write $\boldsymbol{x}_i = \boldsymbol{P}_{L_*}\boldsymbol{x}_i + \boldsymbol{\epsilon}_i$, where $\boldsymbol{P}_{L_*}\boldsymbol{x}_i \in L_*$ and $\boldsymbol{\epsilon}_i \in L_*^{\perp}$ is added noise. Then, $\mathcal{X}^{\text{dns}} = (\boldsymbol{P}_{L_*}\mathcal{X}_{\text{in}}) \cup \mathcal{X}_{\text{out}}$ is the corresponding noiseless inlier-outlier dataset (here, the $\cdot^{\text{dns}}$ superscript stands for "de-noised"). We assume that the noise in our data is uniformly bounded by ϵ, that is, $\|\boldsymbol{x}_i - \boldsymbol{P}_{L_*}\boldsymbol{x}_i\| < \epsilon$ for all $\boldsymbol{x}_i \in \mathcal{X}_{\text{in}}$.

Some small technical issues come up with noisy RSR datasets that make the conditions harder to interpret. However, the following discussion is just a generalization of the previous section on the noiseless case after dealing with these technicalities.

To write the stability statistic in the noisy RSR setting, we must define the following set-valued functions of $\mathcal{X}_{\text{in}}$. These are defined for a unit vector $\boldsymbol{w} \in L_* \cap S^{D-1}$ and small-projection cutoff δ. They are meant to distinguish between inliers who have a projection onto $\boldsymbol{w}$ with length bigger than δ, and inliers who have a projection onto $\boldsymbol{w}$ with length less than or equal to δ. These functions are defined as

$$\mathcal{F}_0(\mathcal{X}_{\text{in}}, \boldsymbol{w}, \delta) = \{\boldsymbol{x} \in \mathcal{X}_{\text{in}} : |\boldsymbol{w}^T\boldsymbol{x}| \leq \delta\}, \tag{14}$$

$$\mathcal{F}_1(\mathcal{X}_{\text{in}}, \boldsymbol{w}, \delta) = \{\boldsymbol{x} \in \mathcal{X}_{\text{in}} : |\boldsymbol{w}^T\boldsymbol{x}| > \delta\}. \tag{15}$$

Inliers in the first set are coined "small-projection inliers", and inliers in the latter set are coined "large-projection inliers".

With these sets, our noisy inlier-outlier stability statistic is

$$\mathcal{S}_n(\mathcal{X}, L_*, \epsilon, \delta, \gamma) = \frac{\cos(\gamma - 2\arctan(\epsilon/\delta))}{2} \min_{\boldsymbol{w} \in L_* \cap S^{D-1}} \left(\sum_{\boldsymbol{x}_i \in \mathcal{F}_1(\mathcal{X}_{\text{in}}, \boldsymbol{w}, \delta)} \frac{\boldsymbol{w}^T\boldsymbol{P}_{L_*}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_{L_*}\boldsymbol{w}}{\|\boldsymbol{P}_{L_*}\boldsymbol{x}\| + \sqrt{\epsilon^2 + \delta^2}} \right)$$
$$- \sqrt{\delta^2 + \epsilon^2} \max_{\boldsymbol{w} \in L_* \cap S^{D-1}} \#\left(\mathcal{F}_0(\mathcal{X}_{\text{in}}, \boldsymbol{w}, \delta)\right) - \sup_{B(L_*,\gamma)} \mathcal{A}(\mathcal{X}_{\text{out}}, L). \tag{16}$$

This statistic is somewhat similar to what we had in the noiseless RSR setting, although now we have separated our inlier terms into two parts. The first term behaves like the permeance from the noiseless RSR setting, with the addition that small-projection inliers are trimmed. The last term is again the alignment of the outliers. The middle term is quite technical, and it is meant to capture cases when inliers may have large angle with a fixed direction of L_*. If we take $\delta \to 0$ and $\epsilon/\delta \to 0$, then the stability almost becomes our original stability, with the added factor of $1/2$ on the permeance term. If $\mathcal{S}_n(\mathcal{X}, L_*, \epsilon, \delta, \gamma) > 0$, then we will demonstrate later that recovery is possible up to accuracy $\eta = 2\arctan(\epsilon/\delta)$.

An illustration of the small-projection cutoff δ, the noise bound ϵ, and the accuracy η for noisy inliers is given in Figure 3 to help ease understanding of our statistic. For simplicity, we show the case of a one-dimensional subspace in $\mathbb{R}^2$. Here, the vectors $\boldsymbol{w} \in L_* \cap S^{D-1}$ are $(1, 0)^T$ or $(-1, 0)^T$. Since these two vectors are equivalent for the two functions in (14) and (15), the large-projection inliers and small-projection inliers are only determined by the magnitude of δ. Thus, the cutoff defined by a certain choice of δ in (14) and (15) corresponds to separating small and large inliers by their x-value.

We note that the statistic is by no means tight and future work should analyze how accurate these methods can be with noise. As we will discuss in Section 7, one could also study RSR in settings with high noise, such as heavy tailed noise or under the spiked model. The main point of the noisy statistic is to show that our results yield ϵ/δ-approximate recovery when the noise is uniformly bounded by ϵ. Here, δ is constrained in that the stability condition, $\mathcal{S}_n(\mathcal{X}, L_*, \epsilon, \delta, \gamma) > 0$, must hold.

3.2. The Local Landscape under Stability

In this section, we will give results that prove the local stability of the energy landscape of (2). We begin with the theorem for the noiseless RSR setting in Section 3.2.1, and then prove an analogous result for the noisy RSR setting in Section 3.2.2.

3.2.1. Stability in the Noiseless Case

We show that positivity of the stability statistic given in (12) with $0 < \gamma < \pi/2$ implies stability of L_* as a minimizer in the neighborhood $B(L_*, \gamma)$. Stability of L_* means that it is the only critical point and minimizer in $B(L_*, \gamma)$, and, at all other points in this neighborhood, there exists a direction in $G(D, d)$ such that the energy landscape looks like one of the two cases displayed in Figure 2; in other words, there is a direction of decrease.

Theorem 1 (Stability of L_*) *Suppose that a noiseless inlier-outlier dataset with an underlying subspace L_* satisfies $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$, for some $0 < \gamma < \pi/2$. Then, all points in*

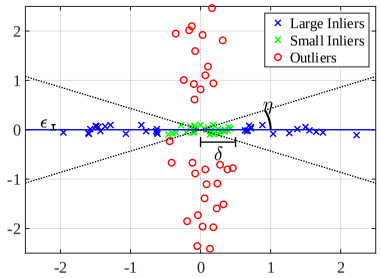


Figure 3: Demonstration of a noisy inlier-outlier dataset, where $d = 1$ and $D = 2$. The following parameters used in our analysis are also displayed: the small-projection cutoff δ, the noise bound ϵ, and the accuracy η. Note that, in this example, inliers within a δ-neighborhood of the origin are removed from the permeance calculation, and ϵ the maximum distance of the inliers to the X-axis. Our analysis guarantees recovery up to the accuracy η, which means that we would recover the x-axis within the acute angle formed by the dotted lines.

$B(L_*, \gamma) \setminus \{L_*\}$ have a subdifferential along a geodesic strictly less than $-\mathcal{S}(\mathcal{X}, L_*, \gamma)$, that is, it is a direction of decreasing energy. This implies that L_* is the only critical point and local minimizer in $B(L_*, \gamma)$.

Proof [Proof of Theorem 1] The main point behind the proof is the following statement. We show that, for any $L \in B(L_*, \gamma) \setminus \{L_*\}$, there is a geodesic $L(t)$ with $L(0) = L$, and an open interval around 0, $\mathcal{I} = (\theta_1(L, L_*) - \gamma, \delta(L))$, for some $\delta(L) > 0$, such that

$$\text{For } t \in \mathcal{I},\ \theta_1(L(t), L_*) \text{ is a strictly decreasing function;} \tag{17}$$

$$\text{For } t \in \mathcal{I},\ F(L(t); \mathcal{X}) \text{ is a strictly decreasing function.} \tag{18}$$

In simple words, the function $F(L(t); \mathcal{X})$ is decreasing as $L(t)$ moves closer to L_*. This implies that L_* is the only critical point and minimizer in $B(L_*, \gamma)$ by a perturbation argument, which we will explicitly state at the end of the proof.

Fix a subspace $L \in B(L_*, \gamma) \setminus \{L_*\}$, and let the principal angles between L and L_* be $\theta_1, \dots, \theta_d$. Also, choose a set of corresponding principal vectors $\boldsymbol{v}_1, \dots, \boldsymbol{v}_d$ and $\boldsymbol{y}_1, \dots, \boldsymbol{y}_d$ for L and L_*, respectively, and let $l \geq 1$ be the maximum index such that $\theta_1 = \cdots = \theta_l$. We let $\boldsymbol{u}_1, \dots, \boldsymbol{u}_l$ be complementary orthogonal vectors for $\boldsymbol{v}_1, \dots, \boldsymbol{v}_l$ and $\boldsymbol{y}_1, \dots, \boldsymbol{y}_l$. For $t \in [0, 1]$, we form the geodesic

$$L(t) = \mathrm{Sp}(\boldsymbol{v}_1 \cos(t) + \boldsymbol{u}_1 \sin(t), \dots, \boldsymbol{v}_l \cos(t) + \boldsymbol{u}_l \sin(t), \boldsymbol{v}_{l+1}, \dots, \boldsymbol{v}_d). \tag{19}$$

Notice that this geodesic is parameterized by arclength in terms of the metric defined in Section 2.2. This geodesic moves only the l furthest directions of $L(0)$ towards L_* and fixes those directions that are closer than θ_1. This geodesic certainly satisfies (17). We

have also removed dependence on θ_1, since this unnecessarily impacts the magnitude of the geodesic subderivative (5). Following (5) with no dependence on $\theta_1, \dots, \theta_l$ and then using (6) and (8), we have

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t} F(L(t); \mathcal{X})\Big|_{t=0} &= -\sum_{j=1}^{l} \sum_{\mathcal{X}} \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} \\
&= -\sum_{j=1}^{l} \left(\sum_{\mathcal{X}_{\text{in}}} \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} + \sum_{\mathcal{X}_{\text{out}}} \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} \right) \\
&\leq -\sum_{j=1}^{l} \left(\frac{\cos(\theta_1)\sin(\theta_1)}{\sin(\theta_1)} \sum_{\mathcal{X}_{\text{in}}} \frac{\boldsymbol{y}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{y}_j}{\|\boldsymbol{x}_i\|} - \sup_{\boldsymbol{V} \in B(L_*, \gamma)} \|\nabla F(\boldsymbol{V}; \mathcal{X}_{\text{out}})\|_2 \right) \\
&\leq -l\mathcal{S}(\mathcal{X}, L_*, \gamma) \leq -\mathcal{S}(\mathcal{X}, L_*, \gamma) < 0.
\end{aligned} \tag{20}$$

The third line is obtained by noting that the inliers and the vectors $\boldsymbol{y}_j$, $1 \leq j \leq l$, are contained L_*, the vectors $\boldsymbol{v}_j$, $1 \leq j \leq l$, are contained in L, and $\theta_1 = \ldots = \theta_l$. Therefore, for all $1 \leq j \leq l$ and all inliers $\boldsymbol{x}_i$, $\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j \geq \cos(\theta_1)\sin(\theta_1)(\boldsymbol{y}_j^T \boldsymbol{x}_i)^2$. In the third line we also used (6) and (8) for the outlier term and maximized it over $B(L_*, \gamma)$. Thus, (20) implies that every subspace in $\overline{B(L_*, \gamma)} \setminus \{L_*\}$ has a direction with negative local subderivative.

The argument above was for a special subderivative at $L(0)$. We now extend this argument to show that this implies that the subdifferential at every point is bounded above by $-\mathcal{S}(\mathcal{X}, L_*, \gamma)$. On the one hand, if $L(0)$ contains no points of $\mathcal{X}$, then the subderivative corresponds to the actual derivative, and we therefore have that $F(L(t); \mathcal{X})$ is decreasing, as desired.

On the other hand, assume that $L(0) \cap \mathcal{X}$ is non-empty. Since $\mathcal{X}$ is finite, there exists a neighborhood of 0, $(-\omega, \omega)$, for $0 < \omega < \pi/2$, such that $L(t)$ has empty intersection with $\mathcal{X} \setminus L(0)$, that is,

$$\{\boldsymbol{x} \in \mathcal{X} : \boldsymbol{x} \in L(t),\ \text{for some } t \in (-\omega, \omega)\} = L(0) \cap \mathcal{X}. \tag{21}$$

Notice that the changing directions in (19) form a geodesic along $G(D, l)$ while the other directions are fixed (i.e., $L(t) \supset \mathrm{Sp}(\boldsymbol{v}_{l+1}, \dots, \boldsymbol{v}_d)$ for all t). Thus, derivatives of $F(L(t); \mathcal{X})$ over $G(D, d)$ are equivalent to derivatives of $F(\tilde{L}(t); \boldsymbol{Q}_{\boldsymbol{v}_{l+1}, \dots, \boldsymbol{v}_d} \mathcal{X})$ over $G(D, l)$, where $\tilde{L}(t) = \mathrm{Sp}(\boldsymbol{v}_1 \cos(t) + \boldsymbol{u}_1 \sin(t), \dots, \boldsymbol{v}_l \cos(t) + \boldsymbol{u}_l \sin(t)))$. Here, $\boldsymbol{Q}_{\boldsymbol{v}_{l+1}, \dots, \boldsymbol{v}_d}$ is the projection onto the orthogonal complement of $\mathrm{Sp}(\boldsymbol{v}_{l+1}, \dots, \boldsymbol{v}_d)$, which is applied to all points of $\mathcal{X}$. By (21), this geodesic over $G(D, l)$ has the property

$$\tilde{L}(t) \cap \mathcal{X} = \emptyset,\ \forall\ t \in (-\omega, 0) \cup (0, \omega).$$

Consequently, $F(\tilde{L}(t); \boldsymbol{Q}_{\boldsymbol{v}_{l+1}, \dots, \boldsymbol{v}_d} \mathcal{X})$ is continuously differentiable on $(-\omega, 0)$ and $(0, \omega)$ as a function over $G(D, l)$, and it is apparent that the bounds in (20) also hold for all of these derivatives. Putting these facts together, by the continuity of the derivatives of $F(\tilde{L}(t); \boldsymbol{Q}_{\boldsymbol{v}_{l+1}, \dots, \boldsymbol{v}_d} \mathcal{X})$ for $t \in (-\omega, 0) \cup (0, \omega)$, the subdifferential of $F(L(t); \mathcal{X})$ at $t = 0$ is bounded above by $-\mathcal{S}(\mathcal{X}, L_*, \gamma)$, which in turn implies that (18) holds.

Finally, there are clearly no other critical points than L_* in $B(L_*, \gamma)$, because every point has a direction of decrease. To show that (17) and (18) imply that L_* is a local minimizer, consider a one-dimensional perturbation of L_*, L'. In other words, $\theta_1(L_*, L') > 0$

and $\theta_2(L_*, L') = 0$. Then, (17) and (18) imply that if $L(t)$ is the geodesic between L' and L_*, then $F(L(t); \mathcal{X})$ is decreasing for $t \in (\theta_1(L', L_*) - \gamma, \theta_1(L', L_*))$. The more general perturbation case is just an extension of this argument. Indeed, a d-dimensional perturbation may be written as a sequence of one-dimensional perturbations. ■

3.2.2. Stability with Small Noise

The following theorem generalizes Theorem 1 for the case of a noisy inlier-outlier dataset. The proof of this theorem is left to Appendix D.1, and it essentially follows that of the noiseless RSR setting with the altered stability statistic. Here, we only guarantee that there is a large region with no critical points up to a precision of $\eta = 2\arctan(\epsilon/\delta)$, which is determined by the noise level and inlier permeance.

Theorem 2 (Stability of $B(L_*, \eta)$ with Noise) *Assume a noisy inlier-outlier dataset, with an underlying subspace L_* and noise parameter $\epsilon > 0$, that satisfies for some $\delta > \epsilon > 0$ the stability condition $\mathcal{S}_n(\mathcal{X}, L_*, \epsilon, \delta, \gamma) > 0$. Let $\eta = 2\arctan(\epsilon/\delta)$ and assume further that $\eta < \gamma$. Then, all points in $B(L_*, \gamma) \setminus B(L_*, \eta)$ have a subdifferential along a geodesic strictly less than $-\mathcal{S}_n(\mathcal{X}, L_*, \epsilon, \delta, \gamma)$, that is, it is a direction of decreasing energy. This implies that the only local minimizers and saddle points in $B(L_*, \gamma)$ are in $B(L_*, \eta)$.*

4. A Geodesic Gradient Method for RSR and its Guarantees

In this section, we discuss a geodesic gradient descent method for minimizing (3). First, Section 4.1 gives the details of our algorithm. After laying out the algorithm, we discuss convergence to a local minimizer under deterministic conditions in Section 4.2. Then, Section 4.3 shows that the PCA d-subspace is a good initializer under a separate deterministic condition.

4.1. Minimization by Geodesic Gradient Descent

We use gradient descent to minimize (4). Denoting the singular value decomposition of the negative gradient by $-\nabla F(\boldsymbol{V}; \mathcal{X}) = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}^T$, Theorem 2.3 of Edelman et al. (1999) states that the geodesic starting at $\boldsymbol{V}(0) = \boldsymbol{V}$ with $\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{V}(t)|_{t=0} = -\nabla F(\boldsymbol{V}; \mathcal{X})$ is

$$\boldsymbol{V}(t) = \boldsymbol{V}\boldsymbol{W}\cos(\boldsymbol{\Sigma}t)\boldsymbol{W}^T + \boldsymbol{U}\sin(\boldsymbol{\Sigma}t)\boldsymbol{W}^T. \tag{22}$$

Here, sin and cos are the typical matrix sin and cos, defined by their corresponding power series.

We develop a geodesic gradient descent method using the construction in (22). At a point $\boldsymbol{V}^k$, we may choose a value of t and move along the geodesic to the next iterate. For a sequence of step-sizes $(t^k)_{k\in\mathbb{N}}$, the sequence of subspaces is defined recursively by $\boldsymbol{V}^{k+1} = \boldsymbol{V}^k(t^k)$. The full algorithm with a specific choice of step-size is given in Algorithm 1 and is referred to as GGD. The complexity of this algorithm is $O(TNDd)$, where T is the number of iterations. Note that we use a piecewise constant scheme for the step-sizes, which starts with step-size $t^1 = s$, and every K iterations the step-size is shrunk by a factor of $1/2$.

Algorithm 1 RSR by Geodesic Gradient Descent (GGD)

1: **Input:** dataset $\mathcal{X}$, subspace dimension d, initial step-size s, tolerance τ, constant step interval length K
2: **Output:** $\boldsymbol{V}^* \in O(D, d)$, whose columns span the robust subspace
3: $\boldsymbol{V}^1 = PCA(\boldsymbol{X}, d)$
4: $k = 1$, $s = 1$
5: **while** $\theta_1(\boldsymbol{V}^k, \boldsymbol{V}^{k-1}) > \tau$ or $k = 1$ **do**
6: Compute $\nabla F(\boldsymbol{V}^k; \mathcal{X})$ by (6) and (8)
7: Compute the SVD $\boldsymbol{U}^k\boldsymbol{\Sigma}^k\boldsymbol{W}^k = -\nabla F(\boldsymbol{V}^k; \mathcal{X})$
8: $s^k = s/2^{\lfloor k/K \rfloor}$
9: $\boldsymbol{V}^{k+1} = \boldsymbol{V}^k\boldsymbol{W}^k\cos(\boldsymbol{\Sigma}^k s^k)\boldsymbol{W}^{kT} + \boldsymbol{U}^k\sin(\boldsymbol{\Sigma}^k s^k)\boldsymbol{W}^{kT}$ ▷ (22)
10: $k = k + 1$
11: **end while**

The next section provides convergence guarantees for GGD with both the piecewise constant step-size of Algorithm 1 and with step-size $s/\sqrt{k}$. We later compare these two examples of step-size on a simulated dataset in Figure 6. The factor of $1/2$ in line 8 is called the shrink-factor and could be chosen to be any fraction. Later, in our experiments, we display the convergence properties of GGD with two choices of this factor.

4.2. Local Convergence of GGD Under Stability

In this section, we give convergence guarantees for GGD. Theorem 3 shows that the convergence is sublinear under the deterministic conditions of Theorem 1 for step-size $t^k = s/\sqrt{k}$. Then, Theorem 4 shows that the convergence of Algorithm 1 is linear under a slightly stronger assumption. These results are all for the noiseless RSR setting. However, all of these results can be extended to the noisy RSR setting in a simple fashion using the notions described in Section 3.1.2.

As a reminder, Theorem 1 implies that if $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ in the noiseless RSR setting then L_* is the only limit point in $B(L_*, \gamma)$ for GGD. In other words, there is no need to worry about saddle points or non-optimal critical points in this neighborhood of L_*. In the following theorem, we give a general sublinear convergence bound for GGD in the local neighborhood considered in Theorem 1. This implies that the algorithm can exactly recover the underlying subspace in the noiseless RSR problem. The step-size used in this theorem is $t^k = s/\sqrt{k}$ at iteration k instead of the piecewise constant scheme seen in Algorithm 1. The proof of this theorem is left to Appendix D.2.

Theorem 3 (Noiseless Sublinear Convergence) *Suppose that $\mathcal{X}$ is an inlier-outlier dataset with an underlying subspace L_*. Suppose also that there exists $0 < \gamma < \pi/2$ such that $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ and that the initial GGD iterate is $\boldsymbol{V}^1 \in B(L_*, \gamma)$. Then, for sufficiently small s as input (which may depend on $\mathcal{S}(\mathcal{X}, L_*, \gamma)$, d, D, N_{in}, and N_{out}), modified GGD with $t^k = s/\sqrt{k}$ converges to L_* with rate $\theta_1(L_k, L_*) < C/\sqrt{k}$, for some constant C.*

While the rate $O(1/\sqrt{k})$ matches typical results in non-smooth optimization, faster convergence is desirable. With the piecewise constant step-size given in Algorithm 1 and a further deterministic condition, GGD linearly converges to the underlying subspace L_*.

Theorem 4 (Noiseless Linear Convergence) *Suppose that $\mathcal{X}$ is an inlier-outlier dataset with an underlying subspace L_*. Suppose also that there exists $0 < \gamma < \pi/2$ such that $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ and that the initial GGD iterate is $\boldsymbol{V}^1 \in B(L_*, \gamma)$. Assume further that*

$$\inf_{L \in B(L_*,\gamma)\setminus\{L_*\}} \frac{1}{4}\left|\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\right| > \sup_{L\in B(L_*,\gamma)\setminus\{L_*\}} \sum_{\mathcal{X}\cap L} 2\|\boldsymbol{x}_i\|, \tag{23}$$

where $L(t)$ is a geodesic parameterized by arclength from L through L_*. Then, for sufficiently large K and sufficiently small s as input (which may depend on $\mathcal{S}(\mathcal{X}, L_*, \gamma)$, d, D, N_{in}, and N_{out}), the sequence generated by Algorithm 1 converges linearly to L_*.

Remark 5 *As mentioned earlier, Theorems 3 and 4 can also be extended to the noisy RSR setting with more complicated statements. Indeed, these extensions follow the same ideas as Theorem 2 (which extends Theorem 1 to the noisy RSR setting).*

We remark that the restriction in (23) can be weakened, although it results in a more complicated theorem statement: for clarity, we show the simpler version in the theorem statement. We refer to (23) as the strong gradient condition. In general, the sum in the right hand side of (23) only contains a few points when the inliers and outliers are not too linearly dependent. For example, consider the case of inliers lying in general position within the subspace and outliers lying in general position. This is the case if all D-subsets of $\mathcal{X}_{\text{out}}$ are linearly independent and all d-subsets of $\mathcal{X}_{\text{in}}$ are linearly independent. Under this assumption, the right hand side contains less than $2(d-1)$ points. On the other hand, the left hand side may be sufficiently large for a wide range of statistical models of data. For more interpretation of this condition for a specific model of data, we point the reader to Lemma 20, where we demonstrate a simple way in which the condition might hold.

We are not able to give an explicit bound on the rate of convergence factor, which depends on certain statistical characteristics of the data. While we do not currently have estimates of this factor and it may depend on N, D, and d, we expect this dependence to not be too bad, especially based on our numerical experiments.

Proof [Proof of Theorem 4] The proof of this theorem is a consequence of the following lemmas. The proof of these lemmas is deferred to Appendix D.3.

The first lemma locally bounds above the increase in cost around L_*.

Lemma 6 *If $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$, then*

$$F(L;\mathcal{X}) - F(L_*;\mathcal{X}) < 2\theta_1(L, L_*)\sum_{\mathcal{X}_{\text{in}}} \|\boldsymbol{x}_i\|, \ \forall L \in B(L_*, \gamma). \tag{24}$$

Notice that $\theta_1(L, L_*)$ is a measure of distance between L and L_*. The next lemma bounds the magnitude of the gradient in $B(L_*, \gamma) \setminus L_*$.

Lemma 7 *If $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$, then, for some $C_1 > 0$ (that depends on the data),*

$$\left|\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\right| > C_1 \sum_{\mathcal{X}_{\text{in}}} \|\boldsymbol{x}_i\|, \ \forall L \in B(L_*, \gamma) \setminus L_*, \tag{25}$$

where $L(t)$ is a geodesic parameterized by arclength from L through L_*.

Finally, the third lemma bounds the decrease in cost between consecutive iterates.

Lemma 8 *If $L_k \in B(L_*, \gamma)$ and (23) holds, then there exists $c_0 > 0$ such that for each step-size choice $t^k = c\theta_1(L_k, L_*)$ with $c < c_0$,*

$$F(L_k) - F(L_{k+1})) \geq \frac{c\theta_1(L_k, L_*)}{2}\left|\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\right|, \tag{26}$$

where $L(t)$ is a geodesic parameterized by arclength from L_k through L_*.

Choosing the step-size $t^k = c\theta_1(L_k, L_*)$, with c coming from Lemma 8, and combining the results of Lemmas 6, 7 and 8, we find that

$$\begin{aligned} F(L_{k+1};\mathcal{X}) - F(L_*;\mathcal{X}) &< F(L_k;\mathcal{X}) - F(L_*;\mathcal{X}) - \frac{c\theta_1(L_k, L_*)}{2}\left|\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\right| \\ &\leq F(L_k;\mathcal{X}) - F(L_*;\mathcal{X}) - \frac{C_1}{2}c\theta_1(L_k, L_*)\left(\sum_{\mathcal{X}_{\text{in}}}\|\boldsymbol{x}_i\|\right) \\ &\leq F(L_k;\mathcal{X}) - F(L_*;\mathcal{X}) - \frac{C_1 c}{4}\left(F(L_k;\mathcal{X}) - F(L_*;\mathcal{X})\right) \\ &\leq (1 - C_2)(F(L_k;\mathcal{X}) - F(L_*;\mathcal{X})), \end{aligned} \tag{27}$$

where $0 < C_2 < 1$. Here, $(1 - C_2)$ is the rate of convergence factor, which depends on d. Thus, if one could choose the step-size $t^k = c\theta_1(L_k, L_*)$, then the sequence of costs $F(L_k;\mathcal{X})$ would converge linearly to $F(L_*;\mathcal{X})$.

For all $L \in B(L_*, \gamma) \setminus \{L_*\}$, letting $a = \theta_1(L, L_*)$, we find that

$$\begin{aligned} F(L;\mathcal{X}) - F(L_*;\mathcal{X}) &= \int_{t=0}^{a} \frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})\mathrm{d}t \geq a \inf_{L'\in B(L_*,\gamma)}\left|\frac{\mathrm{d}}{\mathrm{d}t}F(L'(t);\mathcal{X})\right| \\ &\geq a\mathcal{S}(\mathcal{X}, L_*, \gamma), \end{aligned} \tag{28}$$

where $L(t)$ is the geodesic parameterized by arclength from L_* through L and $L'(t)$ is the geodesic parameterized by arclength from L_* through L'. The last inequality follows from the argument in Theorem 1, and in particular from (20). Thus, for $C_3 = 1/\mathcal{S}(\mathcal{X}, L_*, \gamma)$, where $0 < C_3 < \infty$, (28) implies that

$$\theta_1(L_{k+1}, L_*) \leq C_3(F(L_{k+1};\mathcal{X}) - F(L_*;\mathcal{X})). \tag{29}$$

This means that linear convergence of the energy sequence, $(F(L_k;\mathcal{X}))_{k\in\mathbb{N}}$, gives linear convergence of the iterates, $(L_k)_{k\in\mathbb{N}}$.

So far, we have shown that there exists a c such that choosing step-size $t^k = c\theta_1(L_k, L_*)$ leads to linear convergence of L_k to L_*. However, this choice of step-size is purely theoretical, since in practice we would not know $\theta_1(L_k, L_*)$ at each iteration. We must now rectify this choice of step-size with that used in Algorithm 1.

Suppose a constant step-size s and a constant c satisfying the above argument. Then, the sequence $(L_k)_{k\in\mathbb{N}}$ at least converges linearly to an element of the set $B(L_*, s/c)$ because (27) holds as long as $t^k = s < c\theta_1(L_k, L_*)$. If instead the constant step-size is $s/2$, we will get

linear convergence to an element of the set $B(L_*, s/2c)$, albeit at a slower rate. Notice that, at each shrinking step (i.e., switching to step-size $s/2$ from s), if the sequence has already reached $B(L_*, s/c)$, we can bound the rate of convergence factor $(1 - C_2)$ by

$$\left(1 - \frac{s/2}{c\theta_1(L_k, L_*)}\frac{C_1 c}{4}\right) \leq \left(1 - \frac{C_1 c}{8}\right). \tag{30}$$

Further, for all subsequent steps where $s/2 < c\theta_1(L_k, L_*)$, we have that the rate of convergence factor is strictly less than $1 - C_1 c/(8\sqrt{d}(d+1))$. Thus, as long as the time between shrinking is large enough, we are guaranteed to have linear convergence. If the first step-size is allowed to run long enough, we find that the number of steps m between shrinking needs to be at most

$$\left(1 - \frac{C_1 c}{8}\right)^m < \frac{1}{2}.$$

■

4.3. Complete Guarantee with PCA Initialization

Notice that the previous section does not give a complete guarantee because the results are local. In other words, they assume that we first initialize in $B(L_*, \gamma)$ and then run GGD. To make these results practical, we will show that it is possible to initialize in this neighborhood under a simple deterministic condition. As before with Theorems 3 and 4, we consider only the noiseless RSR setting here. Extensions to the noisy case do not require much more effort.

The result of this section shows that PCA initializes in $B(L_*, \gamma)$ under a similar deterministic condition to $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$. This is quantified in the following lemma.

Lemma 9 *Suppose that, for a noiseless inlier-outlier dataset,*

$$\sqrt{2}\sin(\gamma)\lambda_d(\boldsymbol{X}_{\text{in}}\boldsymbol{X}_{\text{in}}^T) - \|\boldsymbol{X}_{\text{out}}\|_2^2 > 0. \tag{31}$$

Then, $L_{PCA} \in B(L_*, \gamma)$.

Proof The proof of this lemma is a direct consequence of the Davis-Kahan $\sin\theta$ Theorem (Davis and Kahan, 1970), which has a nice formulation in Vu et al. (2013). Let $L_* \in G(D, d)$ span the principal d-subspace of $\boldsymbol{X}_{\text{in}}\boldsymbol{X}_{\text{in}}^T$ and $L_{PCA} \in G(D, d)$ span the principal d-subspace of $\boldsymbol{X}_{\text{out}}\boldsymbol{X}_{\text{out}}^T + \boldsymbol{X}_{\text{in}}\boldsymbol{X}_{\text{in}}^T$. Then, applying Corollary 3.1 of Vu et al. (2013) to these matrices yields

$$|\sin(\theta_1(L_*, L_{PCA}))| \leq \sqrt{2}\frac{\|\boldsymbol{X}_{\text{out}}\|^2}{\lambda_d(\boldsymbol{X}_{\text{in}}\boldsymbol{X}_{\text{in}}^T)}. \tag{32}$$

Thus, if (31) holds, then we are guaranteed that

$$|\sin(\theta_1(L_*, L_{PCA}))| < |\sin(\gamma)|. \tag{33}$$

■

The condition required in (31) bears some nice similarity to the earlier condition of Theorems 1 and 3, $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$. Here the first term in (31) is like the inlier permeance and the second term is like the outlier alignment.

In summary, if both of the conditions $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ and (31) hold, then GGD with step-size $s/\sqrt{k}$ exactly recovers L_* with convergence rate $O(1/\sqrt{k})$. If we additionally have that (23) holds, then GGD with the step-size in Algorithm 1 linearly converges to L_*.

5. Guarantees for Specific Statistical Models

In this section, we discuss some statistical models of data and determine when they satisfy the assumptions of our theorems. These models are meant to illustrate that our conditions are satisfied in a wide range of RSR examples, and they begin to explore the recovery limits of our algorithm. These results should provide some useful context and lead to easier interpretation of the general conditions given in Section 3 and Section 4.

The main idea behind the study of statistical models of data is to compare the theoretical guarantees of various algorithms with a given choice of metric. A natural metric for this purpose is the signal to noise ratio (SNR). The SNR is the ratio of inliers to outliers, $N_{\text{in}}/N_{\text{out}}$, and we are interested in the minimal SNR that allows exact (or sufficiently near) recovery of an RSR algorithm under a given set of assumptions on the data. In other words, under further assumptions on the data generating model, we want to derive a more interpretable condition than $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$.

Due to the properties of these models, different SNR bounds may arise for different regimes of sample size. A first common small sample regime assumes that $N = O(D)$, where the SNR bounds are usually higher to account for the increased variation in the data. Another regime of slightly larger samples is obtained when $N = O(D^p)$ and $p > 1$ is sufficiently small. Under some special statistical models, the SNR may decrease as p increases. That is, a larger fraction of outliers can be tolerated with larger orders of sample size. A third regime uses very large, and possibly arbitrarily large, N. We refer to the SNR bound of this third case as the very large N regime. Under some very special models of data, the SNR bound can go to zero in this regime as the sample size increases. However, in this case, the sample size must depend on the SNR itself.

In the following, not only will we show almost state-of-the-art results for GGD on the Haystack Model of Lerman et al. (2015), but we will also demonstrate how our convergence theorem holds for other more general models of data. This is an important step towards understanding how RSR algorithms perform outside of the simple Haystack Model. The general statistical models considered below assume that the inliers and outliers are sampled from probability distributions that obey certain assumptions. These models are the following:

- Assumptions on the outliers:
  - Bounded support distributions
  - Sub-Gaussian distributions
- Assumptions on the inliers:

– Continuous, bounded support distributions
– Sub-Gaussian distributions

In each of these cases, the inliers and outliers will be assumed to be i.i.d. samples from the given distributions. The precise definition of each of these distributions will be given later.

In Section 5.1, we bound the alignment of outliers under the above outlier models. Then, in Section 5.2, we bound the permeance of inliers under the above inlier models. The goal of these first two subsections is to understand how each part of $\mathcal{S}(\mathcal{X}, L_*, \gamma)$ behaves on its own. After this, in Section 5.3 we prove that $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ under certain conditions on these models of inliers and outliers. Next, we show in Section 5.4 that PCA can initialize in $B(L_*, \gamma)$ in a wide range of cases. Then, Section 5.5 gives an in depth discussion of the Haystack Model, where we show that GGD with step-size $s/\sqrt{k}$ has almost state-of-the-art guarantees. The discussion considers the previously mentioned three regimes of sample size. Finally, Section 5.6 gives an idea of how statistical models can also ensure that the strong gradient condition in Theorem 4 holds, which gives more evidence that the method may converge linearly in practice.

5.1. Outlier Distributions with Restricted Alignment

We explain the two assumptions on outliers listed above, which lead to bounds on the alignment. We first discuss bounded distributions in Section 5.1.1 and then discuss sub-Gaussian distributions in Section 5.1.2.

5.1.1. Bounded Support Distributions

We consider the case of outliers drawn from a distribution with bounded support. This assumption is needed because our bound on the alignment scales like the spectral norm of $\boldsymbol{X}_{\text{out}}$, which can be very large for even a single large outlier. An outlier distribution of this type has the form

$$\mathcal{X}_{\text{out}} \sim \mu, \ \mu(\mathbb{R}^D \setminus B(\boldsymbol{0}, M)) = 0, \tag{34}$$

where μ represents the probability measure and M is a uniform bound on the magnitude of the outliers. In this case, we have the worst-case bound

$$\|\boldsymbol{X}_{\text{out}}\|_2 < M\sqrt{N_{\text{out}}}. \tag{35}$$

In the special case where $\mathcal{X}_{\text{out}} \sim \text{Unif}(B(\boldsymbol{0}, M))$, the following bound was provided in Lemma 8.4 of Lerman et al. (2015):

$$\|\boldsymbol{X}_{\text{out}}\|_2 \leq M\left(\sqrt{\frac{N_{\text{out}}}{D-0.5}} + \sqrt{2} + \frac{t}{\sqrt{D-0.5}}\right), \text{ w.p. at least } 1 - 1.5e^{-t^2}. \tag{36}$$

We remark that (35) holds under any sampling from a bounded distribution and (36) holds under i.i.d. sampling of a special distribution.

From these bounds and (11), we get a sense of how the alignment scales for different types of outliers. When outliers are more adversarial but still bounded, the alignment scales like $O(N_{\text{out}})$. On the other hand, when outliers have the special distribution $\text{Unif}(B(\boldsymbol{0}, M))$, we can bound the alignment by $N_{\text{out}}/\sqrt{D}$. Later, in Theorem 17, we show how to improve this to $O(N_{\text{out}}/\sqrt{D(D-d)})$, due to the fact that the bound in (11) is not tight.

5.1.2. Sub-Gaussian Distributions

Rather than assume that the outliers are bounded, we can instead assume they come from a sub-Gaussian distribution. In this case, we have the following lemma.

Lemma 10 *Suppose that the outliers follow a sub-Gaussian distribution with covariance $\boldsymbol{\Sigma}_{\text{out}}/D_{\text{out}}$, which has rank D_{out}. Then,*

$$\|\boldsymbol{X}_{\text{out}}\|_2 \leq \left\|\boldsymbol{\Sigma}_{\text{out}}^{1/2}\right\|_2 \left(2\frac{\sqrt{N_{\text{out}}}}{\sqrt{D_{\text{out}}}} + C\right), \tag{37}$$

with probability at least $1 - e^{-cN_{\text{out}}}$. Here, c and C depend on the sub-Gaussian norm of $\boldsymbol{\Sigma}_{\text{out}}^{-1/2}\boldsymbol{x}$, where $\boldsymbol{x}$ follows the outlier distribution.

Proof Note that the transformed data, $\boldsymbol{\Sigma}_{\text{out}}^{-1/2}\boldsymbol{X}_{\text{out}}$, is isometric. Therefore, we bound the spectral norm of the outliers by

$$\|\boldsymbol{X}_{\text{out}}\|_2 = \|\boldsymbol{\Sigma}_{\text{out}}^{1/2}\boldsymbol{\Sigma}_{\text{out}}^{-1/2}\boldsymbol{X}_{\text{out}}\|_2 \leq \|\boldsymbol{\Sigma}_{\text{out}}^{1/2}\|_2\|\boldsymbol{\Sigma}_{\text{out}}^{-1/2}\boldsymbol{X}_{\text{out}}\|_2.$$

We can apply Theorem 5.39 of Vershynin (2012b) to the last term in this inequality. This yields the bound in (37). ■

5.2. Permeating Inlier Distributions

We will look at two assumptions that yield permeance of inliers. First, Section 5.2.1 examines continuous distribution inliers, and then Section 5.2.2 looks at sub-Gaussian inliers.

5.2.1. Bounded Continuous Inlier Distributions

An i.i.d. sample from a distribution with a continuous density lies in general position with probability 1. In the case of a continuous distribution on a subspace L_*, this means that no d-subset of them lies on a $d-1$-dimensional subspace. Some slightly stronger assumptions are also needed to easily prove that the inlier permeance has a nontrivial lower bound, which includes the distribution having bounded support, although other assumptions could be used. We will refer to these distributions as bounded continuous distributions. The proof of this proposition is given in Appendix D.4.1.

Proposition 11 *Suppose that the inliers are sampled from a distribution that has a continuous density with respect to the uniform measure on L_*. Suppose further that this distribution has mean zero and has support contained in $B(\boldsymbol{0}, M) \cap L_*$, for some constant M. Then,*

$$\mathcal{P}(\mathcal{X}_{\text{in}}) \gtrsim \frac{N_{\text{in}}}{M} \min_{\boldsymbol{v} \in L_* \cap S^{D-1}} \text{Var}(\boldsymbol{v}^T\boldsymbol{x}), \ \textit{w.h.p.} \tag{38}$$

Here, the probability goes to 1 and the permeance goes to ∞ as $N_{\text{in}} \to \infty$.

5.2.2. Sub-Gaussian Inlier Distributions

We show how the assumption of a sub-Gaussian distribution provides a lower bound for the permeance of inliers. The proof of this theorem is given in Appendix D.4.2.

Proposition 12 *Suppose that inliers are sampled i.i.d. from a sub-Gaussian distribution with covariance $\mathbf{\Sigma}_{\text{in}}/d$, which has rank d. Then, for $0 < a < 1$ satisfying $(1-a)^2 N_{\text{in}} > C_1^2 d$,*

$$\mathcal{P}(\mathcal{X}_{\text{in}}) \geq \frac{\lambda_d(\mathbf{\Sigma}_{\text{in}})}{\lambda_1(\mathbf{\Sigma}_{\text{in}})^{1/2}}(1-a)^2\frac{N_{\text{in}}}{d} + O\left(\sqrt{\frac{N_{\text{in}}}{d}}\right), \text{ w.p. at least } 1 - 4e^{-c_1 a^2 N_{\text{in}}}. \tag{39}$$

Here, c_1 and C_1 are constants that depend on the sub-Gaussian norms of $\mathbf{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}$ and $\widehat{\mathbf{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}}$, where $\boldsymbol{x}$ is a random vector that follows the inlier distribution.

Notice that the choice of a here affects both the bound and the probability. One could, in principal, choose $a = N^{-1/2+\epsilon}$ for some constant $\epsilon > 0$ and still achieve an overwhelming probability bound.

5.3. Combining Statistical Models to Enforce $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$

In this section, we explicitly compare the permeance and alignment bounds for these statistical models of data to see when we can expect to have $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$, which is the essential assumption in Theorems 1 and 3. Together with the result of the next section on PCA initialization, this implies that GGD exactly recovers L_* provided that the SNR is appropriately bounded from below in these models. First, Section 5.3.1 will look at the case of bounded outliers and bounded continuous inliers. Then, Section 5.3.2 will discuss the case of sub-Gaussian inliers and outliers in what we call the Generalized Haystack Model.

5.3.1. Bounded Outliers and Bounded Continuous Inliers

Under the assumption of bounded outliers and bounded continuous inliers, we can guarantee $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ for large enough SNR and sample sizes. This results in the following proposition.

Proposition 13 (Stability with Bounded Outliers and Continuous Bounded Inliers) *Suppose that the outliers follow a bounded distribution and the inliers follow a mean zero, bounded distribution with continuous density on L_*. Then, for a fixed parameter $0 < \gamma < \pi/2$, $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ w.h.p. for sufficiently large SNR and N.*

Proof First, the result of Proposition 11 bounds the permeance of inliers. On the other hand, the outliers follow a bounded distribution. This implies that

$$\max_{L \in G(D,d)} \mathcal{A}(\mathcal{X}, L) \leq M N_{\text{out}}. \tag{40}$$

Thus, comparing (38) and (40), for both $N_{\text{in}}/N_{\text{out}}$ and N sufficiently large, $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ w.h.p. ■

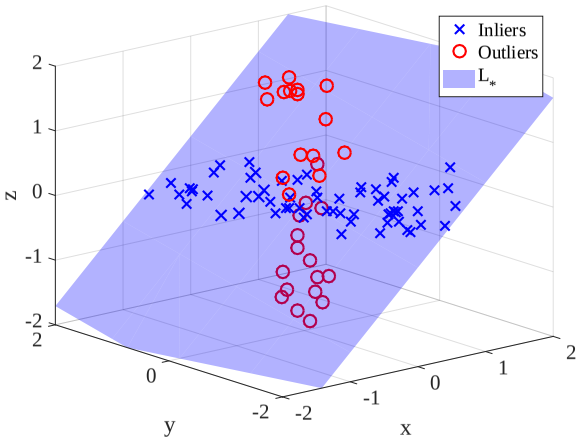


Figure 4: Example dataset drawn from a Generalized Haystack Model, where $d = 2$, $D = 3$, $N_{\text{in}} = 100$, and $N_{\text{out}} = 40$. Here, L_* is a random 2-dimensional subspace. Inliers are sampled i.i.d. from a normal distribution supported on L_* that has variance 4 and 0.09 in its principal directions. The outliers are sampled i.i.d. from a normal distribution with covariance $\mathbf{\Sigma}_{\text{out}} = \text{diag}(.04, .04, 2.25)$.

5.3.2. The Generalized Haystack Model: sub-Gaussian Inliers and Outliers

Next, we propose the Generalized Haystack Model as a special case of sub-Gaussian inliers and outliers. Fix a positive diagonal matrix $\mathbf{\Lambda}_{\text{in}} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{V}^* \in O(D,d)$, which spans $L_* \in G(D,d)$. Letting $\mathbf{\Sigma}_{\text{in}} = \boldsymbol{V}^* \mathbf{\Lambda}_{\text{in}} \boldsymbol{V}^{*T}$, we assume that N_{in} inliers are i.i.d. sampled from a sub-Gaussian distribution with covariance $\mathbf{\Sigma}_{\text{in}}/d$. Fix a symmetric positive semi-definite matrix $\mathbf{\Sigma}_{\text{out}} \in \mathbb{R}^{D\times D}$ and assume N_{out} outliers are i.i.d. sampled from a sub-Gaussian distribution with covariance $\mathbf{\Sigma}_{\text{out}}/D_{\text{out}}$, where D_{out} is the rank of $\mathbf{\Sigma}_{\text{out}}$. This specifies a Generalized Haystack Model with parameters N_{in}, $\mathbf{\Sigma}_{\text{in}}$, N_{out}, $\mathbf{\Sigma}_{\text{out}}$, D_{out}, and d. An example dataset drawn from a Generalized Haystack Model is given in Figure 4. This model generalizes the Haystack Model, which was proposed by Lerman et al. (2015) as a simple model with spherically symmetric Gaussian distributions of inliers and outliers. In the latter model, inliers are distributed i.i.d. $\mathcal{N}(\mathbf{0}, \sigma_{\text{in}}^2 \boldsymbol{P}_{L_*}/d)$, and outliers are distributed i.i.d. $\mathcal{N}(\mathbf{0}, \sigma_{\text{out}}^2 \boldsymbol{I}/D)$. This defines the Haystack Model with parameters N_{in}, σ_{in}, N_{out}, σ_{out}, D and d.

We can combine the previous results on sub-Gaussian inliers and outliers to yield a theoretical guarantee. We show that, under certain conditions on the Generalized Haystack Model, $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ with overwhelming probability.

Theorem 14 (Stability of the Generalized Haystack Model) *Suppose that the dataset $\mathcal{X}$ follows the Generalized Haystack Model with parameters N_{in}, $\mathbf{\Sigma}_{\text{in}}$, N_{out}, $\mathbf{\Sigma}_{\text{out}}$, D_{out}, and*

d. Suppose also that $0 < \gamma < \pi/2$ *and*

$$\mathrm{SNR} > \frac{1}{\cos(\gamma)} \frac{\lambda_1(\mathbf{\Sigma}_{\mathrm{in}})^{1/2}}{\lambda_d(\mathbf{\Sigma}_{\mathrm{in}})} \lambda_1(\mathbf{\Sigma}_{\mathrm{out}}^{1/2}) \frac{2}{(1-a)^2} \frac{d}{\sqrt{D_{\mathrm{out}}}} + o(1). \tag{41}$$

Then $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ *with probability at least* $1 - 4e^{-c_1 a^2 N_{\mathrm{in}}} - 2e^{-c_2 N_{\mathrm{out}}}$*, provided that* $(1-a)^2 N_{\mathrm{in}} > C_1^2 d$*, where* c_1*,* c_2 *and* C_1 *are constants depending on the sub-Gaussian norms of the inliers and outliers.*

Proof It is left to compare the bounds derived earlier in Lemma 10 and Proposition 12. Notice that (41) can be obtained by requiring the right hand side of (37) to be less than $\cos(\gamma)$ times the right hand side of (39). This results in precisely the statement in the theorem. ■

Scaling the inlier and outlier covariance matrices by d and D_{out}, respectively, ensures that in the spherically symmetric case (where $\mathbf{\Sigma}_{\mathrm{in}}$ and $\mathbf{\Sigma}_{\mathrm{out}}$ are orthogonal projections onto subspaces of $\mathbb{R}^D$ of dimensions d and D_{out} respectively) the inliers and outliers have the same typical length. Thus, with this normalization, differences in the traces of $\mathbf{\Sigma}_{\mathrm{in}}$ and $\mathbf{\Sigma}_{\mathrm{out}}$ translate into differences in typical scale between inliers and outliers. We emphasize that it is important to prove results for general sub-Gaussian distributions rather than just spherically symmetric Gaussians. This is due to the fact that simpler strategies, like running PCA and then filtering points far from the PCA subspace, can be applied to the symmetric case with great success. The Generalized Haystack Model allows for certain adversarial outliers: for example, outliers can be contained in a low-dimensional subspace as well. Nevertheless, since the Haystack Model has been addressed by several previous works and since it is easy to improve our estimates for it, we address it in Section 5.5.

5.4. PCA Initialization

The discussed models can also guarantee good initialization by PCA. This is an essential ingredient to actually have a practical algorithm. We demonstrate this on the specific case of the Generalized Haystack Model. However, this sort of argument can be extended to the other types of models discussed above as well (such as bounded distributions of outliers and bounded continuous inliers).

We must have a lower bound on the SNR that depends on the parameters of the sub-Gaussian distributions in order for the following proposition to hold. A short proof for this proposition is given in Appendix D.5. It essentially states that Lemma 9 holds with high probability under certain conditions on the Generalized Haystack Model.

Proposition 15 (PCA Initialization with Sub-Gaussian Models) *Suppose that the dataset* $\mathcal{X}$ *follows the Generalized Haystack Model with parameters* N_{in}*,* $\mathbf{\Sigma}_{\mathrm{in}}$*,* N_{out}*,* $\mathbf{\Sigma}_{\mathrm{out}}$*,* D_{out}*, and* d*. Suppose also that, for some* $0 < \gamma < \pi/2$*,*

$$\mathrm{SNR} \geq \frac{\sqrt{2}}{\sin(\gamma)} \frac{d}{D_{\mathrm{out}}} \frac{\lambda_1(\mathbf{\Sigma}_{\mathrm{out}})}{\lambda_d(\mathbf{\Sigma}_{\mathrm{in}})} + o(1). \tag{42}$$

Then, for large enough $N = N_{\mathrm{out}} + N_{\mathrm{in}}$*, the PCA* d*-subspace is contained in* $B(L_*, \gamma)$ *w.h.p.*

5.5. Performance of GGD Under the Haystack Model

We assume here the simpler Haystack Model and show that GGD performs almost as well as state-of-the-art methods on datasets drawn from this model. We compute results for three different regimes of sample size. These are the small sample regime $N = O(D)$, the larger sample regime $N = O(D^p)$, for $p > 1$ sufficiently small, and the very large N regime, where N must depend on the SNR as well. In the larger sample regime, GGD requires at least $N = O(d(D-d)^2 \log(D))$, which is not more than $N = O(D^{3+\epsilon})$. In the very large N regime, in addition to dependence of N on a power of D, it also depends on a negative power of the SNR and thus is very large for small SNR. In our case, the very large N regime considers sample sizes of the order $N_{\mathrm{out}} = O(\max(d^3 D^3 \log^3(N_{\mathrm{out}}), (dN_{\mathrm{out}}/N_{\mathrm{in}})^6))$. The big O notation is slightly abused here, as we are really indicating results for finite N and D: the order is meant to illustrate the relation between these finite values. We compare all of these results together in Table 1.

5.5.1. Bounds for Sample Size $N = O(D)$

We first translate the bounds obtained previously in Theorem 14 to this special model. We choose $a = 1/2$ and thus obtain the following corollary:

Corollary 16 *Suppose that the dataset* $\mathcal{X}$ *follows the Haystack Model with parameters* N_{in}*,* σ_{in}*,* N_{out}*,* σ_{out}*,* D*, and* d*. Then, if* $N_{\mathrm{in}} > 4C_1^2 d$ *and*

$$\mathrm{SNR} \geq 8 \frac{1}{\cos(\gamma)} \frac{\sigma_{\mathrm{out}}}{\sigma_{\mathrm{in}}} \frac{d}{\sqrt{D}} + o(1), \tag{43}$$

$\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ *with probability at least* $1 - 4e^{-c_1 N_{\mathrm{in}}/4} - 2e^{-c_2 N_{\mathrm{out}}}$*, where* c_1*,* c_2 *and* C_1 *are absolute constants.*

Notice that in this case, we obtain strong probabilistic estimates for even small sample sizes of $N = O(D)$. For the full theoretical guarantee, we also need to consider (42), and we must choose a value for γ. To balance between the $\sin(\gamma)$ and $\cos(\gamma)$ in (42) and (43), respectively, we fix $\gamma = \pi/4$. From these equations, for this fixed γ, we conclude that our theoretical SNR for the Haystack Model in the small sample regime is

$$\mathrm{SNR} \geq \max\left(8\sqrt{2}\frac{\sigma_{\mathrm{out}}}{\sigma_{\mathrm{in}}}\frac{d}{\sqrt{D}}, 2\frac{\sigma_{\mathrm{out}}^2}{\sigma_{\mathrm{in}}^2}\frac{d}{D}\right). \tag{44}$$

On the other hand, previous works (Hardt and Moitra, 2013; Lerman et al., 2015; Zhang, 2016) obtained optimal bounds for this model when $N = O(D)$ and the SNR is on the order of

$$\mathrm{SNR} \gtrsim \frac{\sigma_{\mathrm{out}}}{\sigma_{\mathrm{in}}} \frac{d}{(D-d)}. \tag{45}$$

We remark that the bound of Lerman et al. (2015) for the REAPER algorithm requires the assumption $d < (D-1)/2$ and its constant is relatively large. This is in contrast to Hardt and Moitra (2013) and Zhang (2016), who do not have restrictions on d and do not have dependence on $\sigma_{\mathrm{out}}/\sigma_{\mathrm{in}}$. In this regime, we are unable to establish sharp results like the ones of REAPER, Tyler's M-estimator, or RandomizedFind. These estimates are

better by a factor of $\sqrt{D}/(D-d)$ than our current estimate. Nevertheless, in this regime the complexity of our algorithm is $O(NDd)$, whereas the complexity of the mentioned algorithms is $O(ND^2)$ or $O(D^3)$.

5.5.2. Bounds for Sample Size $N = O(d(D-d)^2 \log(D))$

Zhang and Lerman (2014) obtained the following sharper bound for the GMS algorithm under the Haystack Model and the larger sample regime of $N = O(D^2)$:

$$\text{SNR} \gtrsim \frac{\sigma_{\text{out}}}{\sigma_{\text{in}}} \frac{d}{\sqrt{D(D-d)}}. \tag{46}$$

We remark that this is the sharpest bound for any similar sample regime under this model when $\sigma_{\text{out}} \approx \sigma_{\text{in}}$. While the bounds mentioned above (Hardt and Moitra, 2013; Lerman et al., 2015; Zhang, 2016) hold for any regime of sample size, they are worse by a factor of $\sqrt{D/(D-d)}$. We show that a similar bound in a similar regime holds for the GGD algorithm. Indeed, the primary deficiency in Corollary 16 is that we use the loose bound on the alignment, $\sqrt{N_{\text{out}}}\|\boldsymbol{X}_{\text{out}}\|_2$. However, one could instead operate using the precursor to this bound,

$$\mathcal{A}(\mathcal{X}_{\text{out}}, L) \leq \|\widetilde{\boldsymbol{Q}_L \boldsymbol{X}_{\text{out}}}\|_2 \|\boldsymbol{X}_{\text{out}}\|_2. \tag{47}$$

Using this bound instead, we have the following theorem, which shows that GGD achieves the optimal SNR bound under the Haystack Model in the region $N = O(d\,(D-d)^2\,\log(D))$, which is at worst $O(D^3 \log D)$.

Theorem 17 (Stability of the Haystack Model) *Suppose that the dataset $\mathcal{X}$ follows the Haystack Model with parameters N_{in}, σ_{in}, N_{out}, σ_{out}, D, and d. If*

$$\text{SNR} \geq \frac{\sigma_{\text{in}}}{\sigma_{\text{out}}} \frac{1}{\cos(\gamma)} \frac{5}{(1-a)^2} \frac{d}{\sqrt{D(D-d)}} + o(1), \tag{48}$$

then $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ with probability at least

$$1 - 2e^{-N_{\text{out}}/16} - e^{-N_{\text{out}}/4} - C_1 \exp\left(-\frac{N_{\text{out}}}{4(D-d)} + \frac{d(D-d)}{2} \log\left(\frac{D}{D-d}\right)\right) - 4e^{-c_1 a^2 N_{\text{in}}}. \tag{49}$$

Here, we can take a close to 0 (e.g., we can take $a = o(N_{\text{in}}^{-1/2})$), and so we ignore the factor of $1/(1-a)^2$ in (48). The full theoretical guarantee, which also considers (42), is established similarly to (44). We conclude that our theoretical SNR bound for the Haystack Model in this regime is

$$\text{SNR} \geq \max\left(5\sqrt{2}\frac{\sigma_{\text{out}}}{\sigma_{\text{in}}} \frac{d}{\sqrt{D(D-d)}}, 2\frac{\sigma_{\text{out}}^2}{\sigma_{\text{in}}^2}\frac{d}{D}\right). \tag{50}$$

The final results for the two different regimes, that is, (44) and (50), are for $\gamma = \pi/4$ and initialization by PCA. As a side note, if the SNR grows, we see that larger values of γ may be tolerated for GGD. In particular, for large sample sizes and sufficiently large SNRs, γ can be sufficiently close to $\pi/2$. In this case, random initialization of GGD are expected

to work as well as the PCA initialization. We quantify this claim more rigorously in the special case where $d < D/2$ and $d, D \to \infty$. Based on the analysis of extreme singular values of random Gaussian matrices (Rudelson and Vershynin, 2008), it can be shown that with high probability, a random initialization lies in $B(L_*, \gamma) \setminus \{L_*\}$, where $\cos(\gamma) = O(1/\sqrt{Dd})$. Therefore, GGD with random initialization succeeds with high probability under the given assumptions on d and D when $N_{\text{in}}/N_{\text{out}} \geq O\left(d\sigma_{\text{out}}/\sigma_{\text{in}}\right)$.

5.5.3. Bounds for Very Large N

In the large sample regime, one can prove something much stronger. Indeed, for any fraction of outliers, it is obvious that PCA asymptotically recovers the underlying subspace L_* (Lerman and Maunu, 2018a). Further, it was shown that FMS can asymptotically recover L_* for any fraction of outliers with better dependence on the sample size than PCA (Lerman and Maunu, 2018a) (while the result given for FMS is for the spherized Haystack Model, the result can be extended to the non-spherized version as well). It would be very surprising if GGD could not do something similar.

In fact, this type of result can be extended to GGD as well, which means that, as $N \to \infty$, we can take SNR$\to 0$. However, while the PCA and FMS subspace estimators converge to the underlying subspace as $N \to \infty$, the probability of either exactly recovering the underlying subspace for any fixed N is zero. In contrast, GGD can exactly recover the underlying subspace with overwhelming probability for finite, yet large, N. This result is stated in the following theorem, which is proved in Appendix D.7.

Theorem 18 *Suppose that $\mathcal{X}$ follows the Haystack Model with parameters N_{in}, σ_{in}, N_{out}, σ_{out}, D, and d. For any* SNR *lower bound $\alpha > 0$ (i.e.,* SNR $> \alpha$*) and for any N_{out} at least $O(\max(d^3 D^3 \log^3(N_{\text{out}}), (dN_{\text{out}}/N_{\text{in}})^6))$, GGD recovers L_* w.o.p. In particular, N_{out} is at least $O(d/\alpha^6)$,*

For this theorem to hold, we see that N needs to be quite large, especially for low SNRs. However, note that we still obtain strong probabilistic bounds for large enough finite N. On the other hand, taking $\alpha \to 0$ in this theorem requires $N_{\text{out}} \to \infty$, which implies asymptotic recovery for GGD for any SNR in this model.

5.5.4. Comparison of all Haystack Model Results

We compare the lowest SNR guarantees for a variety of RSR algorithms under the Haystack Model. Table 1 replicates Table 1 of Lerman and Maunu (2018b) with updated estimates. It both compares lower bounds on SNR under the Haystack Model and also briefly describes the actual data model that each algorithm has guarantees for. The algorithms include geodesic gradient descent (GGD), FMS (Lerman and Maunu, 2018a), REAPER (Lerman et al., 2015), GMS (Zhang and Lerman, 2014), OP (Xu et al., 2012) and LLD (McCoy and Tropp, 2011), HR-PCA (Xu et al., 2013), Tyler's M-estimator (TME) (Zhang, 2016), TORP (Cherapanamjeri et al., 2017), CP (Rahmani and Atia, 2016), SSC (Soltanolkotabi and Candès, 2012), HOSC (Arias-Castro et al., 2011), RANSAC (Arias-Castro and Wang, 2017), and RF (Hardt and Moitra, 2013). For each SNR bound, we also give the associated sample size, N, for each result to begin holding with high probability. Again, we note that the big O notation is slightly abused here since the results are really for non-asymptotic N and D.

PCA	$N_{\text{in}}/N_{\text{out}}$ " $\gtrsim$ "0, when $N \to \infty$, D fixed *No exact recovery and poor estimates for finite N.*
GGD	$N_{\text{in}}/N_{\text{out}} \geq \max\left(8\sqrt{2}\frac{\sigma_{\text{out}}}{\sigma_{\text{in}}}\frac{d}{\sqrt{D}}, 2\frac{\sigma_{\text{out}}^2}{\sigma_{\text{in}}^2}\frac{d}{D}\right)$, when $N = O(D)$ $N_{\text{in}}/N_{\text{out}} \geq \max\left(5\sqrt{2}\frac{\sigma_{\text{out}}}{\sigma_{\text{in}}}\frac{d}{\sqrt{D(D-d)}}, 2\frac{\sigma_{\text{out}}^2}{\sigma_{\text{in}}^2}\frac{d}{D}\right)$, when $N = O(d(D-d)^2\log(D))$ Any $N_{\text{in}}/N_{\text{out}} > 0$, when $N_{\text{out}} \gtrsim \max(d^3D^3\log^3(N), (dN_{\text{out}}/N_{\text{in}})^6)$ $\implies N_{\text{in}}/N_{\text{out}} \gtrapprox 0$, when $N \to \infty$, D fixed *Deterministic condition, results for a variety of data models.*
FMS	$N_{\text{in}}/N_{\text{out}}$ " $\gtrsim$ "0, when $N \to \infty$, D fixed *Approximate recovery for large samples from spherized Haystack or from two one-dimensional subspaces on the sphere. Much better estimates for finite N than PCA.*
REAPER	$N_{\text{in}}/N_{\text{out}} \geq 16\frac{\sigma_{\text{out}}}{\sigma_{\text{in}}}\frac{d}{D}$, when $N = O(D)$, $1 \leq d \leq (D-1)/2$ *Deterministic condition, results for Haystack where $d < (D-1)/2$.*
GMS	$N_{\text{in}}/N_{\text{out}} \geq 4\frac{\sigma_{\text{out}}}{\sigma_{\text{in}}}\frac{d}{\sqrt{(D-d)D}}$, when $N = O(D^2)$ *Deterministic condition, results for Haystack that extends to elliptical outliers.*
OP	$N_{\text{in}}/N_{\text{out}} \geq \frac{121d}{9}O(\max(1, \log(N)/d))$, when $N = O(D)$ *Deterministic condition (formulated for arbitrary outliers) with last term in above formula replaced by an inlier incoherence parameter μ.*
HR-PCA	$N_{\text{in}}/N_{\text{out}} \to \infty$, when $N \to \infty$, D fixed *Weak lower bound on the expressed variance, requires fraction of outliers as input.*
TME/(D)RF	$N_{\text{in}}/N_{\text{out}} > \frac{d}{D-d}$, when $N = O(D)$ *Result for "general-position" data, but does not extend to noise.*
TORP	$N_{\text{in}}/N_{\text{out}} \geq 128d\max(1, \log(N)/d)^2$, when $N = O(D)$ *Deterministic condition (formulated for arbitrary outliers) with last term replaced by an inlier incoherence parameter μ, requires fraction of outliers as input.*
CP	$N_{\text{in}}/N_{\text{out}} \geq d/(D-d^2)$ $(N = O(D), d < \sqrt{D})$ $N_{\text{in}}/N_{\text{out}} \gtrapprox 0$, when $N \to \infty$, $d < \sqrt{D}$, D fixed *Exact recovery for the spherized Haystack model with a random inlier subspace and $d < \sqrt{D}$, and also for a special model of outliers around a line.*
SSC	$N_{\text{in}}/N_{\text{out}} \geq d/D \cdot ((\frac{N_{\text{in}}-1}{d})^{\frac{cD}{d}-1} - 1)^{-1}$, when $N < e^{c\sqrt{D}}/D$, w.h.p. in D *Exact recovery for the spherized Haystack model with a random inlier subspace.*
HOSC	$N_{\text{in}}/N_{\text{out}} \geq \log(N)N^{-2(D-d)/(2D-d)}$ $\implies N_{\text{in}}/N_{\text{out}} \gtrapprox 0$, where $N \to \infty$, D fixed *Result for outliers uniformly sampled from $[0,1]^D$ and inliers uniformly sampled from the intersection of a d-subspace with $(0,1)^d$. Also extends to a union of manifolds and to settings with small noise.*

Table 1: Updated table from Lerman and Maunu (2018b), which compares the lower bounds on SNR for many RSR methods under the Haystack Model. Included here are theoretical guarantees along with the corresponding sample size requirements for the result to hold with probability close to 1.

Here, the symbol $\gtrapprox 0$ is used for the SNR lower bound for recovery when a method can exactly recover an underlying subspace for any fixed SNR and large enough N. Similarly, the symbol " $\gtrsim$ "0 is used when a method can *approximate* L_* to any accuracy for any fixed

SNR and large enough N. However, methods with " $\gtrsim$ "0 instead of $\gtrapprox 0$ cannot exactly recover the underlying subspace.

Among all algorithms, only PCA, GGD, TORP, RANSAC, and FMS run in $O(NDd)$ time. Since the strong gradient condition can be shown to hold for the Haystack model (see Corollary 21 in Section 5.6), GGD also achieves linear convergence. This means that it is theoretically guaranteed to be the fastest out of these $O(NDd)$ algorithms (TORP has a guarantee for linear convergence, but under very restrictive assumptions on the SNR). Furthermore, among all algorithms, GGD is the only one with guarantees close to state-of-the-art for the small sample size regime. Among all algorithms it has the state-of-the-art result for the regime $N = O(d(D-d)^2\log(D))$, although GMS obtains such a result for the smaller regime $N = O(D^2)$. GGD is also the fastest algorithm with a result for the large sample regime as well. We note that algorithms with worse complexity, such as CP, SSC, and HOSC, have guarantees in this setting as well. We have found it too complicated to compare the exact theoretical results of these various methods, and so we have instead opted to just show that they are guaranteed to have exact recovery for any percentage of outliers for large enough N.

5.6. A Note on the Strong Gradient Condition in Theorem 4

While it may be hard to interpret the strong gradient condition in Theorem 4, it is possible to show that it holds for a variety of the data models in this section. The brief discussion here is just meant to illustrate that this condition is, in fact, practical. It is not hard to extend the arguments below to other cases discussed earlier as well. This lemma relies on the following definition.

Definition 19 *Sets of inliers and outliers, $\mathcal{X}_{\text{in}}$ and $\mathcal{X}_{\text{out}}$, within an inlier-outlier dataset are said to lie in general position with respect to each other if*

$$\max_{G(D,d)\setminus\{L_*\}} \#(\mathcal{X} \cap L) \leq d. \tag{51}$$

With this definition, we prove the following lemma.

Lemma 20 *Suppose that the inliers and outliers follow distributions that satisfy* (34) *and lie in general position with respect to each other. Assume further that the inliers are drawn from a distribution such that $\mathcal{P}(\mathcal{X}_{\text{in}}) \to \infty$ as $N \to \infty$, and that*

$$\max_{L\in G(D,d)} \mathcal{A}(\mathcal{X}_{\text{out}}, L) < c\cos(\gamma)\mathcal{P}(\mathcal{X}_{\text{in}}), \tag{52}$$

for some $0 < c < 1$ for all N. Then, (23) *holds for N sufficiently large.*

The general position assumption implies that the amount of data points in any d-subspace that is not L_* is at most d. This general position-type assumption is also satisfied with probability 1 by any sample drawn from the Haystack Model, or by inliers drawn uniformly from $L_* \cap B(\mathbf{0}, M)$ and outliers drawn uniformly from $B(\mathbf{0}, M)$.

Proof Under the assumptions of the lemma, (23) reduces to

$$\inf_{L\in B(L_*,\gamma)\setminus\{L_*\}} \left|\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\right| > 2dM, \tag{53}$$

where $L(t)$ is a geodesic parameterized by arclength from L_k through L_*. Since we know that $\left|\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\right| \geq \mathcal{S}(\mathcal{X}, L_*, \gamma)$ (from the proof of Theorem 1), a sufficient condition for the strong gradient condition to hold is

$$\mathcal{S}(\mathcal{X}, L_*, \gamma) > 8dM. \tag{54}$$

The assumptions also imply that

$$\mathcal{S}(\mathcal{X}, L_*, \gamma) > \cos(\gamma)(1-c)\mathcal{P}(\mathcal{X}_{\text{in}}). \tag{55}$$

Therefore, the reduction of the strong gradient condition in (54) is satisfied for N sufficiently large, since $\mathcal{P}(\mathcal{X}_{\text{in}}) \to \infty$ as $N \to \infty$. ■

One can easily modify the proof of Theorem 18 and the proof of Lemma 20 to obtain linear convergence for the Haystack model. We state this result without proof in the following corollary.

Corollary 21 *Suppose that the inliers and outliers are distributed according to the Haystack Model with parameters N_{in}, σ_{in}, N_{out}, σ_{out}, D, and d. Then, (23) holds w.h.p. for N sufficiently large.*

6. Simulations

In this section, we run two simulations to verify the theory we proved earlier. All experiments are run using the Haystack Model. Some more comprehensive comparisons of the various RSR algorithms on synthetic and stylized datasets are contained in the review paper of Lerman and Maunu (2018b).

First, we attempt to demonstrate the stability condition $\mathcal{S}$ from (12). While we cannot explicitly evaluate the maximum within this expression, we can instead simulate the values achieved by

$$\cos(\gamma)\mathcal{P}(\mathcal{X}_{\text{in}}) - \mathcal{A}(\mathcal{X}_{\text{out}}, L), \tag{56}$$

for L in a small neighborhood of L_*.

The values achieved by (56) are simulated in Figure 5 for the noiseless RSR settings with the fixed value of $\gamma = \pi/4$. The dataset for this figure is generated according to the Haystack Model outlined in Section 5.5, with parameters $N_{\text{in}} = 200$, $\sigma_{\text{in}} = 1$, $N_{\text{out}} = 200$, $\sigma_{\text{out}} = 1$, $D = 200$, and $d = 10$. For this plot, the y-axis represents a distance from the underlying subspace L_* in terms of the maximum principal angle up to γ. The x-axis represents randomly generated subspaces at that distance from L_*. The color represents the value of (56). As we can see, the value of (56) is indeed positive within this large neighborhood of L_*.

We also simulate the convergence properties of the GGD method in Figure 6. The data was generated according to the Haystack Model with parameters $N_{\text{in}} = 200$, $\sigma_{\text{in}} = 1$, $N_{\text{out}} = 200$, $\sigma_{\text{out}} = 1$, $D = 100$, and $d = 5$. We compare different choices of step-size in accordance with the statements of Theorems 3 and 4. The green line denotes the convergence of the step-size $t^k = s/\sqrt{k}$ in GGD with $s = 1/D$, which is the modified GGD from Theorem 3. We also display three different types of piecewise constant step-size schemes

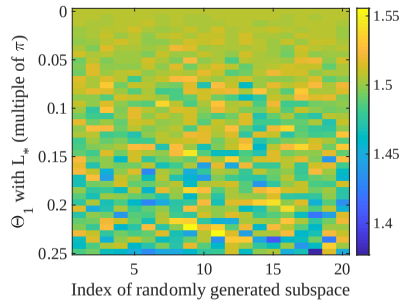


Figure 5: Simulation of the stability statistic. Data was generated for the Haystack Model with parameters $N_{\text{out}} = 200$, $\sigma_{\text{out}} = 1$, $D = 200$, and $d = 10$. The y-axis of each figure represents the maximum principal angle of a randomly generated subspace with L_*. For each value of the maximum principal angle, the x-axis represents the index in the set of 20 randomly generated subspaces with the specified maximum principal angle.

with initial step-size $1/D$. The blue and red lines represent Algorithm 1 with different choices of K and shrink-factors. In one case, we use $K = 20$ and a shrink-factor of $1/2$. In the other case, we use $K = 50$ and a shrink-factor of $1/10$. The pink line is a modification of this piecewise constant step-size scheme that decreases the cost monotonically. In this scheme, the algorithm only shrinks the step-size when the current constant step-size does not decrease the energy function. When this is the case, the step-size is shrunk by the largest factor $1/2^n$, for $n \in \mathbb{N}$, such that the energy decreases. While this set-up performs the best, we do not yet have a theoretical justification for this scheme. We see that all of the choices of piecewise constant step-size converge linearly to L_* (although the convergence is not monotonic). The small fluctuations around an error of 10^{-7} occur due to the machine precision limit in MATLAB.

7. Conclusion

We have presented a deterministic condition that ensures the landscape of (2) behaves nicely around an underlying subspace: if $\mathcal{S}(\gamma, 0, L_*) > 0$, the underlying subspace is the only minimizer and stationary point in $B(L_*, \gamma)$. The deterministic condition also ensures the convergence of a non-convex gradient method for RSR. The convergence of this method is linear under some slightly stronger assumptions on the data. We have shown that the condition $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ and the strong gradient condition in (23) hold for certain statistical models of data and have used these examples to understand the limits of recovery in various regimes. These models indicate the flexibility of the conditions to deal with a wide range of RSR datasets. GGD is even shown to have almost state-of-the-art SNR guarantees under the Haystack Model.

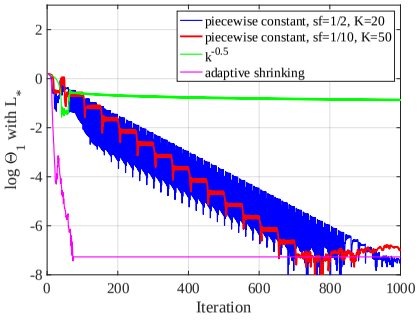


Figure 6: Convergence characteristics of the GGD algorithm with different step-size choices. Data was generated according to the Haystack Model with parameters $N_{\text{in}} = 200$, $\sigma_{\text{in}} = 1$, $N_{\text{out}} = 200$, $\sigma_{\text{out}} = 1$, $D = 100$, and $d = 5$.. The y-axis is the logarithm of the top principal angle with the underlying subspace L_*. For all algorithms, the initial step-size is $s = 1/D$. For the piecewise constant step-size scheme of Algorithm 1, we use two difference combinations of shrink-factor and time between shrinking, K. We also include an adaptive shrinking method, that only shrinks the constant step-size when the current step-size does not decrease the energy.

A main point of this work is to obtain exact recovery guarantees that are similar in spirit to the REAPER algorithm of Lerman et al. (2015) for a non-convex algorithm. We even manage to provide better estimates in large sample regimes. Indeed, the stability analysis of this work is inspired by the stability analysis of the REAPER algorithm, which is done with respect to a convex relaxation of the energy function considered here. Since we do not relax the non-convex problem, our stability is tighter than the REAPER stability when considering sufficiently small neighborhoods. However, for large enough neighborhoods, the REAPER stability might be tighter than ours. For example, a difference shows up in small sample regimes of the Haystack Model: the dependence on a neighborhood makes our result weaker in terms of the SNR regimes we can tolerate. Nevertheless, as far as we know, there is no non-convex RSR competitor for the types of estimates we have developed in this paper and no other competitor with computational complexity of order $O(NDd)$. Furthermore, in larger sample regimes we obtain a stronger result than REAPER's.

While there are many directions that future work can take, we only specify a couple here. One avenue for future work is to extend this result to other data models. For example, one may consider more adversarial models of corruption. This is pursued in a forthcoming work (Maunu and Lerman).

Another direction for future work involves study of subspace recovery in the presence of heavier noise. The only works that consider RSR in really noisy settings are Coudron and Lerman (2012), Minsker (2015), and Cherapanamjeri et al. (2017), although the area remains largely unexplored. The noise considered in our work involves small perturbations

from the underlying subspace. However, in general, more modern work in data science has focused on settings with heavier noise. For example, some have considered noise drawn from distributions with heavy tails (Minsker, 2015), while others have considered PCA in the spiked model (Johnstone, 2001; Baik et al., 2005). In the latter case, when the dimension is very large, the noisy inliers would most likely be very far from the underlying subspace. It is an interesting direction for future work to study RSR in both of these settings.

Future work may also consider proof of convergence for other methods, such as Newton's method and conjugate gradient (Edelman et al., 1999), or IRLS (Lerman and Maunu, 2018a), using the guarantees on the energy landscape of this paper. One could also consider using different frameworks for optimization over $G(D, d)$, such as a retraction based method like that specified in Absil et al. (2009). A quick heuristic argument indicates that the retraction formulation should agree with GGD up to first order, but we leave rigorous examination of this to future work.

Finally, one may also directly follow the ideas of Lim et al. (2016, 2018) and consider extensions to the affine Grassmannian, since in practice we cannot assume that the data is properly centered.

A supplementary web page with code will be made available at `https://twmaunu.github.io/WTL/`.

Acknowledgments

This work was supported by NSF awards DMS-14-18386 and DMS-18-21266, a UMII Mn-DRIVE graduate assistantship, and a UMN Doctoral Dissertation Fellowship. The authors would like to thank Chao Gao, Nati Srebro, and anonymous reviewers for helpful comments, and Moritz Hardt for the professional handling of this manuscript.

Appendix A. Landscape of the PCA energy

In this appendix, we discuss the landscape of the energy function described in (1). Let $\boldsymbol{X} \in \mathbb{R}^{D\times N}$ be a matrix with columns given by $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$, and assume that its singular values and left singular vectors are $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_D$ and $\boldsymbol{v}_1, \dots, \boldsymbol{v}_D \in \mathbb{R}^D$, respectively. Under a generic scenario when $\sigma_d > \sigma_{d+1}$ and $\sigma_{D-d} > \sigma_{D-d+1}$, the landscape of the energy in (1) has the following properties:

- There exists a unique local minimum given by $L = \text{span}(\boldsymbol{v}_1, \dots, \boldsymbol{v}_d)$, which is also the global minimizer; and a unique local maximum given by $L = \text{span}(\boldsymbol{v}_{D-d+1}, \dots, \boldsymbol{v}_D)$, which is also the global maximizer.
- The set of saddle points are given by the set of L such that $L = \text{span}(\boldsymbol{v}_{i_1}, \dots, \boldsymbol{v}_{i_d})$ for any d distinct integers $(i_1, \dots, i_d)$ between 1 and D, and $(i_1, \dots, i_d)$ can not be $(1, \dots, d)$ or $(D-d+1, \dots, D)$ (which correspond to the minimizer and the maximizer).

These properties can be derived by casting the PCA problem as a constrained optimization problem and then examining the corresponding Lagrangian. Since the number of saddle points is finite and all saddle points are orthogonal to the minimizer, there exists a local neighborhood around the minimizer such that the minimizer is the unique critical point

inside this neighborhood, which is a property similar to Theorem 1. Using a strategy similar to this work, if an initialization is chosen appropriately such that the initialization lies in a neighborhood around the global minimizer, a gradient descent algorithm would converge to the global minimizer of (1).

One other case of interest involves $\sigma_d = \sigma_{d+1} = \cdots = \sigma_k$, for $d < k \leq D$. In this case, the singular vectors corresponding to $\sigma_d, \ldots, \sigma_k$ would span a subspace, and any orthonormal basis for $\mathrm{Sp}(\boldsymbol{v}_d, \ldots, \boldsymbol{v}_k)$ would result in a valid set of singular vectors corresponding to the singular values $\sigma_d, \ldots, \sigma_k$. As an example of what happens in this case, if $\sigma_1 > \sigma_2 > \cdots > \sigma_d = \cdots = \sigma_k$ then the d-subspace spanned by $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{d-1}$, $\boldsymbol{v}_i \in \mathrm{Sp}(\boldsymbol{v}_d, \ldots, \boldsymbol{v}_k) \cap S^{D-1}$ are minimizers. One can construct other sorts of continuums by also considering equality of singular values between σ_1 and σ_d. All cases constructed in these ways would yield a continuum of minimizers for the PCA d-subspace energy, which all achieve the same energy. Analogously, one can consider the cases where $\sigma_k = ... = \sigma_{D-d+1}$ for $1 \leq k < D-d+1$, and show a continuum of maximizers. Or, one can also consider the cases where $\sigma_k = ... = \sigma_l$ where $d < k < l < D - d + 1$, and show a continuum of saddle points.

In addition, while it is unrelated to the focus of this work, we remark that the landscape of PCA also has the "strict saddle point property" discussed in Ge et al. (2015): the Hessian of every saddle point has a negative eigenvalue. Ge et al. (2015) propose algorithms with theoretical guarantees to minimize such energy functions. This property can be seen by again examining the Lagrangian formulation of the PCA problem.

Appendix B. Grassmannian Geodesics

In this appendix, we describe some basic geometric notions on the Grassmannian manifold, $G(D,d)$. Given two subspaces $L_1, L_2 \in G(D,d)$, the principal angles between the two subspaces are defined sequentially. The smallest angle, θ_d, is given by

$$\theta_d = \min_{\substack{\boldsymbol{v} \in L_1, \boldsymbol{y} \in L_2 \\ \|\boldsymbol{v}\| = \|\boldsymbol{y}\| = 1}} \arccos(|\boldsymbol{v}^T \boldsymbol{y}|). \tag{57}$$

The vectors $\boldsymbol{v}_d$ and $\boldsymbol{y}_d$ which achieve the minimum are the principal vectors corresponding to θ_d. The d principal angles are defined sequentially by

$$\theta_k = \min_{\substack{\boldsymbol{v} \in L_1, \|\boldsymbol{v}\| = 1, \boldsymbol{v} \perp \boldsymbol{v}_{k+1}, \ldots, \boldsymbol{v}_d \\ \boldsymbol{y} \in L_2, \|\boldsymbol{y}\| = 1, \boldsymbol{y} \perp \boldsymbol{y}_{d+1}, \ldots, \boldsymbol{y}_d}} \arccos(|\boldsymbol{v}^T \boldsymbol{y}|), \tag{58}$$

and the corresponding principal vectors are found in the same way. The ordering defined in (57) and (58) is the reverse of what is usually used for principal angles: here, θ_1 is the largest principal angle, while most other works denote the smallest principal angle with θ_1. Notice that if two principal angles are equal, the choice of principal vectors is not unique. Principal angles and vectors can be efficiently calculated: if $\boldsymbol{W}_1 \in O(D,d)$ spans L_1 and $\boldsymbol{W}_2 \in O(D,d)$ spans L_2, then we write the singular value decomposition

$$\boldsymbol{W}_1^T \boldsymbol{W}_2 = \boldsymbol{V}_{12} \boldsymbol{\Sigma}_{12} \boldsymbol{Y}_{12}^T. \tag{59}$$

The principal angles are given in reverse order by $\arccos(\mathrm{diag}(\boldsymbol{\Sigma}_{12}))$, and the corresponding principal vectors are given by the columns of $\boldsymbol{V}_{12}$ and $\boldsymbol{Y}_{12}$ in reverse order. Now let two

subspaces L_1 and L_2 have principal angles $\theta_1, \ldots, \theta_d$ and principal vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d$ and $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_d$, respectively. Let k be the largest index such that $\theta_k > 0$, which is also known as the interaction dimension (Lerman and Zhang, 2014), where $d - k$ is the dimension of $L_1 \cap L_2$. Then, we can define a complementary orthogonal basis $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k$ for L_2 with respect to L_1 as

$$\boldsymbol{u}_j \in \mathrm{Sp}(\boldsymbol{v}_j, \boldsymbol{y}_j),\ \boldsymbol{u}_j \perp \boldsymbol{v}_j,\ \boldsymbol{u}_j^T \boldsymbol{y}_j > 0. \tag{60}$$

As explained in Lerman and Zhang (2014), for any two subspaces L_1 and L_2 such that $\theta_1 = \arccos(|\boldsymbol{v}_1^T \boldsymbol{y}_1|) < \pi/2$, a unique geodesic on $G(D,d)$ with $L(0) = L_1$ and $L(1) = L_2$ can be parameterized by

$$L(t) = \mathrm{Sp}(\boldsymbol{v}_1 \cos(\theta_1 t) + \boldsymbol{u}_1 \sin(\theta_1 t), \ldots, \boldsymbol{v}_k \cos(\theta_k t) + \boldsymbol{u}_k \sin(\theta_k t), \boldsymbol{v}_{k+1}, \ldots, \boldsymbol{v}_d). \tag{61}$$

In the paper, we will frequently use a reparametrization of geodesics to prevent the angles from needlessly affecting the magnitude of the derivatives. In these cases, we reparametrize the geodesic in (62) by arclength (in terms of metric defined Section 2.2) by writing

$$\begin{aligned} L(t) = \mathrm{Sp}\Big(\boldsymbol{v}_1 \cos(t) + \boldsymbol{u}_1 \sin(t), \boldsymbol{v}_2 \cos\left(\frac{\theta_2}{\theta_1} t\right) + \boldsymbol{u}_2 \sin\left(\frac{\theta_2}{\theta_1} t\right), \ldots, \\ \boldsymbol{v}_k \cos\left(\frac{\theta_k}{\theta_1} t\right) + \boldsymbol{u}_k \sin\left(\frac{\theta_k}{\theta_1} t\right), \boldsymbol{v}_{k+1}, \ldots, \boldsymbol{v}_d\Big). \end{aligned} \tag{62}$$

Appendix C. Bound on the Alignment

Here we derive the simple bound on the alignment statistic seen in (11). Here, the $\widetilde{\cdot}$ notation denotes projection of the data points (columns of a data matrix) to the unit sphere, S^{D-1}. This bound is used to prove Theorem 17. The first step follows from (6), (8), and (10).

$$\begin{aligned} \mathcal{A}(\mathcal{X}_{\text{out}}, \boldsymbol{V}) &= \left\| \sum_{\boldsymbol{x}_i \in \mathcal{X}_{\text{out}}} \frac{\boldsymbol{Q}_{\boldsymbol{V}} \boldsymbol{x}_i}{\|\boldsymbol{Q}_{\boldsymbol{V}} \boldsymbol{x}_i\|} \boldsymbol{x}_i^T \boldsymbol{V} \right\|_2 = \left\| \widetilde{\boldsymbol{Q}_{\boldsymbol{V}} \boldsymbol{X}_{\text{out}}} \boldsymbol{X}_{\text{out}}^T \boldsymbol{V} \right\|_2 \\ &\leq \left\| \widetilde{\boldsymbol{Q}_{\boldsymbol{V}} \boldsymbol{X}_{\text{out}}} \right\|_2 \left\| \boldsymbol{X}_{\text{out}}^T \boldsymbol{V} \right\|_2 \leq \left\| \widetilde{\boldsymbol{Q}_{\boldsymbol{V}} \boldsymbol{X}_{\text{out}}} \right\|_F \|\boldsymbol{X}_{\text{out}}\|_2 \\ &\leq \sqrt{N_{\text{out}}} \|\boldsymbol{X}_{\text{out}}\|_2. \end{aligned} \tag{63}$$

Appendix D. Supplementary Proofs

In this appendix, we give the supplementary proofs for the various theorems, propositions, and lemmas given in the paper.

D.1. Proof of Theorem 2

The proof of this theorem proceeds much in the same way as the proof of Theorem 1.

We first define the following geodesic on $G(D,d)$: Fix a subspace $L \in B(L_*, \gamma) \setminus B(L_*, \eta)$, and let the principal angles between L and L_* be $\theta_1, \ldots, \theta_d$. Here, we define $\eta = 2 \arctan(\epsilon/\delta)$. Also, choose a set of corresponding principal vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d$

and $\boldsymbol{w}_1, \dots, \boldsymbol{w}_d$ for L and L_*, respectively, and let l be the maximum index such that $\theta_1 = \cdots = \theta_l$. We let $\boldsymbol{u}_1, \dots, \boldsymbol{u}_l$ be complementary orthogonal vectors for $\boldsymbol{v}_1, \dots, \boldsymbol{v}_l$ and $\boldsymbol{w}_1, \dots, \boldsymbol{w}_l$, which exist since $\theta_1(L, L_*) > 0$. For $t \in [0,1]$, we form the geodesic

$$L(t) = \mathrm{Sp}(\boldsymbol{v}_1 \cos(t) + \boldsymbol{u}_1 \sin(t), \dots, \boldsymbol{v}_l \cos(t) + \boldsymbol{u}_l \sin(t), \boldsymbol{v}_{l+1}, \dots, \boldsymbol{v}_d).$$

This geodesic moves only the furthest directions of $L(0)$ towards L_*, and we have removed dependence on θ_1, since this unnecessarily impacts the magnitude of the geodesic derivative (5).

We will first prove the inequality $\partial F(L(t); \mathcal{X}) < -\mathcal{S}_n(\mathcal{X}, L_*, \epsilon, \delta, \gamma)$. Using the derivative formula in (5), a subderivative of $F(L(t); \mathcal{X})$ at $t = 0$ is given by

$$\frac{\mathrm{d}}{\mathrm{d}t} F(L(t); \mathcal{X})\Big|_{t=0} = - \sum_{\substack{\boldsymbol{x}_i \in \mathcal{X} \\ \|\boldsymbol{Q}_L \boldsymbol{x}_i\| > 0}} \sum_{j=1}^{l} \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} \tag{64}$$

$$= - \sum_{j=1}^{l} \left(\sum_{\substack{\boldsymbol{x}_i \in \mathcal{F}_1(\mathcal{X}_{\mathrm{in}}, \boldsymbol{w}_j, \delta) \\ \|\boldsymbol{Q}_L \boldsymbol{x}_i\| > 0}} \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} + \sum_{\substack{\boldsymbol{x}_i \in \mathcal{F}_0(\mathcal{X}_{\mathrm{in}}, \boldsymbol{w}_j, \delta) \\ \|\boldsymbol{Q}_L \boldsymbol{x}_i\| > 0}} \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} + \sum_{\substack{\boldsymbol{x}_i \in \mathcal{X}_{\mathrm{out}} \\ \|\boldsymbol{Q}_L \boldsymbol{x}_i\| > 0}} \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} \right).$$

We examine the terms in (64) one by one. Using (6) and (8), we can bound the outlier term

$$- \sum_{\substack{\boldsymbol{x}_i \in \mathcal{X}_{\mathrm{out}} \\ \|\boldsymbol{Q}_L \boldsymbol{x}_i\| > 0}} \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} \leq \|\nabla F(L; \mathcal{X}_{\mathrm{out}})\|_2 . \tag{65}$$

We know that $|\boldsymbol{x}_i \cdot \boldsymbol{u}_j| \leq \|\boldsymbol{Q}_L \boldsymbol{x}_i\|$ for all i since $\boldsymbol{u}_j \in \mathrm{Sp}(\boldsymbol{Q}_L)$. We also know that, since $\theta_j > \eta$, $|\boldsymbol{v}_j^T \boldsymbol{x}_i| \leq |\boldsymbol{w}_j^T \boldsymbol{x}_i| \leq \sqrt{\delta^2 + \epsilon^2}$ for all $\boldsymbol{x}_i \in \mathcal{F}_0(\mathcal{X}_{\mathrm{in}}, \boldsymbol{w}_j, \delta)$. These two observations imply that

$$- \sum_{\substack{\boldsymbol{x}_i \in \mathcal{F}_0(\mathcal{X}_{\mathrm{in}}, \boldsymbol{w}_j, \delta) \\ \|\boldsymbol{Q}_L \boldsymbol{x}_i\| > 0}} \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} \leq \sqrt{\delta^2 + \epsilon^2} \max_{\boldsymbol{w} \in L_* \cap S^{D-1}} \#(\mathcal{F}_0(\mathcal{X}_{\mathrm{in}}, \boldsymbol{w}, \delta)). \tag{66}$$

Thus, we finally must deal with the inlier term. We have the inequalities

$$\boldsymbol{v}_j \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j \geq \cos(\theta_1 - \eta) \sin(\theta_1 - \eta), \tag{67}$$

$$\|\boldsymbol{Q}_L \boldsymbol{x}_i\| \leq \sin(\theta_1) \|\boldsymbol{P}_{L_*} \boldsymbol{x}_i\| + \epsilon. \tag{68}$$

Applying these to the inlier term, as long as $\theta_1 > \eta$, we find

$$\begin{aligned} - \sum_{\substack{\boldsymbol{x}_i \in \mathcal{F}_1(\mathcal{X}_{\mathrm{in}}, \boldsymbol{w}_j, \delta) \\ \|\boldsymbol{Q}_L \boldsymbol{x}_i\| > 0}} & \frac{\boldsymbol{v}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{u}_j}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} \\ & \leq - \frac{\cos(\theta_1 - \eta) \sin(\theta_1 - \eta)}{\sin(\theta_1)} \sum_{\substack{\boldsymbol{x}_i \in \mathcal{F}_1(\mathcal{X}_{\mathrm{in}}, \boldsymbol{w}_j, \delta) \\ \|\boldsymbol{Q}_L \boldsymbol{x}_i\| > 0}} \frac{\boldsymbol{w}_j^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{w}_j}{\|\boldsymbol{P}_{L_*} \boldsymbol{x}_i\| + \epsilon / \sin(\theta_1)} \\ & \leq - \frac{\cos(\gamma - \eta)}{2} \lambda_d \left(\sum_i \frac{\boldsymbol{P}_{L_*} \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{P}_{L_*}}{\|\boldsymbol{P}_{L_*} \boldsymbol{x}_i\| + \sqrt{\delta^2 + \epsilon^2}} \right). \end{aligned} \tag{69}$$

Putting together (65), (66), and (69), we find that

$$\partial F(L(t); \mathcal{X})\Big|_{t=0} \leq -l \mathcal{S}_n(\mathcal{X}, L_*, \epsilon, \delta, \gamma) < 0. \tag{70}$$

Thus, (70) implies that every subspace in $B(L_*, \gamma) \backslash B(L_*, \eta)$ has a direction with negative local subderivative. From here, the proof is the same as that of Theorem 1.

D.2. Proof of Theorem 3

We will first show that

$$\theta_1(\boldsymbol{V}^{k+1}, L_*) < \theta_1(\boldsymbol{V}^k, L_*), \tag{71}$$

for sufficiently small t^k. Let $\boldsymbol{V}^* \in O(D, d)$ span L_*. We will establish (71) by showing

$$\sigma_d \left(\boldsymbol{V}^{*T} \boldsymbol{V}^{k+1} \right) > \sigma_d \left(\boldsymbol{V}^{*T} \boldsymbol{V}^k \right).$$

Using (22) and the fact that

$$\cos(\boldsymbol{\Sigma}^k t^k) = \boldsymbol{I} - O((t^k)^2), \qquad \sin(\boldsymbol{\Sigma}^k t^k) = \boldsymbol{\Sigma}^k t^k - O((t^k)^3),$$

we can write

$$\begin{aligned} \sigma_d \left(\boldsymbol{V}^{*T} \boldsymbol{V}^{k+1} \right) &= \sigma_d \left(\boldsymbol{V}^{*T} \left(\boldsymbol{V}^k \boldsymbol{W}^k \cos(\boldsymbol{\Sigma}^k t^k) \boldsymbol{W}^{kT} + \boldsymbol{U}^k \sin(\boldsymbol{\Sigma}^k t^k) \boldsymbol{W}^{kT} \right) \right) \\ &= \sigma_d \left(\boldsymbol{V}^{*T} \left(\boldsymbol{V}^k - t^k \nabla F(\boldsymbol{V}^k; \mathcal{X}) + O((t^k)^2) \right) \right). \end{aligned} \tag{72}$$

Let $\boldsymbol{v}_1^k \in \mathrm{Sp}(\boldsymbol{V}^k)$ be a unit vector corresponding to the maximum principal angle with L_*, and let $\boldsymbol{u}_1^k$ be its complementary orthogonal vector. Define the unit vector $\boldsymbol{y}_1^k \in L_* \cap \mathrm{Sp}(\boldsymbol{v}_1^k, \boldsymbol{u}_1^k)$, and write $\theta_1^k = \theta_1(\mathrm{Sp}(\boldsymbol{V}^k), L_*)$. Suppose that $\sigma_d^k = \sigma_d(\boldsymbol{V}^{*T} \boldsymbol{V}^k)$ has multiplicity r, and let $\boldsymbol{\beta}_1,\ \boldsymbol{\beta}_2 \in O(d, r)$ be such that

$$\boldsymbol{\beta}_1^T \boldsymbol{V}^{*T} \boldsymbol{V}^k \boldsymbol{\beta}_2 = \mathrm{diag}(\sigma_d^k, \dots, \sigma_d^k). \tag{73}$$

We now apply Result 4.1 in Söderström (1999), which states the following. Suppose a matrix $\boldsymbol{A}$ has a singular value σ with multiplicity r, with corresponding left and right

singular vectors $\boldsymbol{U}$ and $\boldsymbol{V}$. Suppose that we perturb $\boldsymbol{A}$ by $\epsilon\boldsymbol{B}$. Then, $\boldsymbol{A}+\epsilon\boldsymbol{B}$ has r singular values $\sigma_1(\boldsymbol{A}+\epsilon\boldsymbol{B}),\dots,\sigma_r(\boldsymbol{A}+\epsilon\boldsymbol{B})$ which satisfy

$$\sigma_j(\boldsymbol{A}+\epsilon\boldsymbol{B}) = \sigma_j(\boldsymbol{A}) + \frac{\epsilon}{2}\lambda_j\left(\boldsymbol{V}^T\boldsymbol{B}^T\boldsymbol{U} + \boldsymbol{U}^T\boldsymbol{B}\boldsymbol{V}\right) + O(\epsilon^2).$$

Applying this to (72) yields

$$\begin{aligned}
&\sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k+1}\right) \geq \sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k}\right) + \frac{t^k}{2}\lambda_d\left(\boldsymbol{\beta}_1^T\boldsymbol{V}^{*T}\sum_{\mathcal{X}}\frac{\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{V}^k}{\|\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\|}\boldsymbol{\beta}_2\right) + O((t^k)^2) \qquad (74)\\
&= \sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k}\right) + t^k\left(\boldsymbol{y}_1^{kT}\sum_{\mathcal{X}}\frac{\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\|}\boldsymbol{v}_1^k\right) + O((t^k)^2)\\
&= \sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k}\right) + t^k\sin(\theta_1^k)\left(\boldsymbol{u}_1^{kT}\sum_{\mathcal{X}_{\text{in}}}\frac{\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\|}\boldsymbol{v}_1^k + \boldsymbol{u}_1^{kT}\sum_{\mathcal{X}_{\text{out}}}\frac{\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\|}\boldsymbol{v}_1^k\right) + O((t^k)^2).
\end{aligned}$$

Here, the $O((t^k)^2)$ term is bounded below by $-C_2(t^k)^2$, where C_2 does not depend on $\boldsymbol{V}^k$, which follows from compactness of $O(D,d)$.

Notice that the inlier term in (74) is positive and bounded below

$$\begin{aligned}
\boldsymbol{u}_1^{kT}\sum_{\mathcal{X}_{\text{in}}}\frac{\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\|}\boldsymbol{v}_1^k &\geq \frac{1}{\sin(\theta_1(\text{Sp}(\boldsymbol{V}^k),L_*))}\sum_{\mathcal{X}_{\text{in}}}\frac{\boldsymbol{u}_1^{kT}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{v}_1^k}{\|\boldsymbol{x}_i\|} \qquad (75)\\
&\geq \cos(\theta_1(\text{Sp}(\boldsymbol{V}^k),L_*)))\sum_{\mathcal{X}_{\text{in}}}\frac{\boldsymbol{y}_1^{kT}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{y}_1^k}{\|\boldsymbol{x}_i\|} \geq \cos(\gamma)\lambda_d\left(\sum_{\mathcal{X}_{\text{in}}}\frac{\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|}\right).
\end{aligned}$$

Using the fact that $\boldsymbol{u}_1^k \in \text{Sp}(\boldsymbol{Q}_{\boldsymbol{V}^k})$ and $\boldsymbol{v}_1^k \in \text{Sp}(\boldsymbol{V}^k)$, we can bound the outlier term in (74)

$$\left|\boldsymbol{u}_1^{kT}\sum_{\mathcal{X}_{\text{out}}}\frac{\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\|}\boldsymbol{v}_1^k\right| \leq \sigma_1\left(\sum_{\mathcal{X}_{\text{out}}}\frac{\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{Q}_{\boldsymbol{V}^k}\boldsymbol{x}_i\|}\boldsymbol{V}^k\right) = \sigma_1\left(\nabla F(\boldsymbol{V}^k;\mathcal{X}_{\text{out}})\right). \qquad (76)$$

Thus, from (75) and (76) we conclude

$$\begin{aligned}
&\sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k+1}\right) - \sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k}\right) \qquad (77)\\
&\geq t^k\sin(\theta_1^k)\left(\cos(\gamma)\lambda_d\left(\sum_{\mathcal{X}_{\text{in}}}\frac{\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|}\right) - \sigma_1\left(\nabla F(\boldsymbol{V}^k;\mathcal{X}_{\text{out}})\right)\right) - C_2(t^k)^2\\
&\geq t^k\sin(\theta_1^k)\left(\cos(\gamma)\lambda_d\left(\sum_{\mathcal{X}_{\text{in}}}\frac{\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|}\right) - \sup_{\boldsymbol{V}\in B(L_*,\gamma)}\sigma_1\left(\nabla F(\boldsymbol{V};\mathcal{X}_{\text{out}})\right)\right) - C_2(t^k)^2\\
&\geq t^k\sin(\theta_1^k)C_1 - C_2(t^k)^2 = t^k\left(\sin(\theta_1^k)C_1 - C_2t^k\right),
\end{aligned}$$

for positive constants C_1 and C_2 which do not depend on $\boldsymbol{V}^k$. Hence, for small enough t^k, we have that (71) holds.

It remains to show that the sequence with step-size $s/\sqrt{k}$ converges to L_* for sufficiently small s. Suppose that s satisfies

$$s < \min\left(\frac{C_1\sin(\gamma)}{2C_2}, \frac{1}{4\sqrt{C_2}}\right). \qquad (78)$$

Then, for any $\boldsymbol{V}^k$ with $\frac{\gamma}{2\sqrt{k}} \leq \theta_1(\boldsymbol{V}^k, L_*) \leq \gamma$, looking at (77) and the first term in (78), $s/\sqrt{k}$ decreases the principal angle by at least C_3/k, for some constant C_3. On the other hand, for any $\boldsymbol{V}^k$ such that $\theta_1(\boldsymbol{V}^k, L_*) < \frac{\gamma}{2\sqrt{k}}$, we have the bound

$$\sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k+1}\right) - \sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k}\right) > -C_2(t^k)^2. \qquad (79)$$

Note that (79) gives the inequality

$$\sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k+1}\right) > \sigma_d\left(\boldsymbol{V}^{*T}\boldsymbol{V}^{k}\right) - C_2(t^k)^2 \geq \cos\left(\frac{\gamma}{2\sqrt{k}}\right) - C_2(t^k)^2. \qquad (80)$$

It is straightforward to show that if

$$t^k < \frac{1}{4\sqrt{C_2}\sqrt{k}},$$

then the right hand side of (80) is greater than $\cos(\gamma/\sqrt{k})$. Thus, the second term in the minimum of (78) implies that if $\theta_1(\boldsymbol{V}^k, L_*) < \frac{\gamma}{2\sqrt{k}}$, then $\theta_1(\boldsymbol{V}^{k+1}, L_*) < \gamma/\sqrt{k}$.

We summarize this in the following way. For any k, either $\frac{\gamma}{2\sqrt{k}} \leq \theta_1(\boldsymbol{V}^k, L_*) \leq \gamma$ or $\theta_1(\boldsymbol{V}^k, L_*) < \frac{\gamma}{2\sqrt{k}}$. If the former holds, then $\theta_1(\boldsymbol{V}^{k+1}, L_*) < \theta_1(\boldsymbol{V}^k, L_*) - C_3/k$. If the latter holds, then we have the bound $\theta_1(\boldsymbol{V}^{k+1}, L_*) < \gamma/\sqrt{k}$. Thus, the maximum principal angle with L_* either decreases by C_3/k or the distance is bounded by $\gamma/\sqrt{k}$. Put together, these imply that $\boldsymbol{V}^k$ converges to L_* with $O(1/\sqrt{k})$ rate of convergence.

D.3. Proof of Lemmas in Theorem 4

This appendix contains proofs for the three lemmas in Theorem 4.

D.3.1. Proof of Lemma 6

For any fixed subspace $L \in B(L_*, \gamma)$, let the principle vectors and angles for L with respect to L_* be given by $\{\boldsymbol{v}_j\}_{j=1}^d$ and $\{\theta_j\}_{j=1}^d$. Then, writing $\boldsymbol{\Theta} = \text{diag}(\theta_1, \cdots, \theta_d)$, we know that $F(L_*; \mathcal{X}_{\text{in}}) = 0$ and

$$\begin{aligned}
F(L;\mathcal{X}_{\text{in}}) &= \sum_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\|\sin(\theta)\boldsymbol{V}^T\boldsymbol{x}\| < \sum_{\boldsymbol{x}\in\mathcal{X}_{\text{in}}}\|\boldsymbol{\Theta}\boldsymbol{V}^T\boldsymbol{x}\| \leq \|\boldsymbol{\Theta}\|_2\max_{\boldsymbol{V}\in O(D,d)}\sum_{\mathcal{X}_{\text{in}}}\|\boldsymbol{V}^T\boldsymbol{x}_i\| \qquad (81)\\
&\leq \theta_1\sum_{\mathcal{X}_{\text{in}}}\|\boldsymbol{x}_i\|.
\end{aligned}$$

Let $L(t)$ be the geodesic from L_* to L parameterized by arclength, and let $s = \theta_1$. The difference in energies for the outliers is bounded by

$$|F(L;\mathcal{X}_{\text{out}}) - F(L_*;\mathcal{X}_{\text{out}})| = \left|\int_{t=0}^{s} \frac{\mathrm{d}}{\mathrm{d}t} F(L(t);\mathcal{X}_{\text{out}})\mathrm{d}t\right| \tag{82}$$
$$\leq \theta_1 \sup_{L' \in B(L_*,\sin(\gamma)} (\sigma_1(\nabla F(L';\mathcal{X}_{\text{out}}))).$$

By the assumption $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$, we have by the definition in (12) that

$$\sup_{L' \in B(L_*,\sin(\gamma)} (\sigma_1(\nabla F(L';\mathcal{X}_{\text{out}}))) < \cos(\gamma)\lambda_d \left(\sum_{\mathcal{X}_{\text{in}}} \frac{\boldsymbol{x}_i \boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|}\right) \leq \max_{\boldsymbol{v}\in S^{D-1}} \left(\sum_{\mathcal{X}_{\text{in}}} \boldsymbol{v}^T \frac{\boldsymbol{x}_i \boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|}\boldsymbol{v}\right) \tag{83}$$
$$< \max_{\boldsymbol{v}\in S^{D-1}} \sum_{\mathcal{X}_{\text{in}}} |\boldsymbol{v}^T \boldsymbol{x}| \leq \sum_{\mathcal{X}_{\text{in}}} \|\boldsymbol{x}_i\|_2.$$

Thus, combining (81)-(83) yields

$$|F(L;\mathcal{X}) - F(L_*;\mathcal{X})| = |F(L;\mathcal{X}_{\text{in}}) + F(L;\mathcal{X}_{\text{out}}) - (F(L_*;\mathcal{X}_{\text{in}}) + F(L;\mathcal{X}_{\text{out}}))| \tag{84}$$
$$\leq |F(L;\mathcal{X}_{\text{in}})| + |F(L;\mathcal{X}_{\text{out}}) - F(L;\mathcal{X}_{\text{out}})|$$
$$\leq 2\theta_1 \sum_{\mathcal{X}_{\text{in}}} \|\boldsymbol{x}_i\|.$$

As a result, the lemma is proved.

D.3.2. Proof of Lemma 7

For any $L \in B(L_*,\gamma) \setminus \{L_*\}$ and geodesic $L(t)$ parameterized by arclength from L through L_*,

$$\left|\frac{\mathrm{d}}{\mathrm{d}t} F(L(t);\mathcal{X})|_{t=0}\right| = \left|\frac{\mathrm{d}}{\mathrm{d}t} F(L(t);\mathcal{X}_{\text{in}})|_{t=0} + \frac{\mathrm{d}}{\mathrm{d}t} F(L(t);\mathcal{X}_{\text{out}})|_{t=0}\right| \tag{85}$$
$$\geq \left|\frac{\mathrm{d}}{\mathrm{d}t} F(L(t);\mathcal{X}_{\text{in}})|_{t=0}\right| - \left|\frac{\mathrm{d}}{\mathrm{d}t} F(L(t);\mathcal{X}_{\text{out}})|_{t=0}\right|$$
$$\geq \cos(\gamma)\lambda_1 \left(\sum_{\mathcal{X}_{\text{in}}} \frac{\boldsymbol{x}_i \boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|}\right) - \sup_{L\in B(L_*,\gamma)} \sigma_1(\nabla F(L;\mathcal{X}_{\text{out}}))$$
$$\geq \mathcal{S}(\mathcal{X}, L_*, \gamma) \geq C_1 \sum_{\mathcal{X}_{\text{in}}} \|\boldsymbol{x}_i\|,$$

where, $C_1 = \mathcal{S}(\mathcal{X}, L_*, \gamma)/\sum_{\mathcal{X}_{\text{in}}} \|\boldsymbol{x}_i\|$.

D.3.3. Proof of Lemma 8

First, assume that c_0 is small enough such that

$$\inf_{L(0)\in B(L_*,\gamma)\setminus\{L_*\}} \frac{1}{4}\left|\frac{\mathrm{d}}{\mathrm{d}t} F(L(t);\mathcal{X})|_{t=0}\right| > \sum_{\mathcal{X}} 3\sqrt{c_0}\|\boldsymbol{x}_i\|, \tag{86}$$

where $L(t)$ is a geodesic parameterized by arclength between $L(0)$ and L_*. We are guaranteed that such a c_0 exists by Lemma 6.

Fix $c < c_0$ and define a geodesic line $L(t)$ on $G(D,d)$ such that $L(0) = L_k$ and $L(c\theta_1(L_k, L_*)) = L_{k+1}$. We first investigate the derivatives of the function $\operatorname{dist}(\boldsymbol{x}, L(t))$. We will then show, under the given assumptions on the data, that $\sum_i \operatorname{dist}(\boldsymbol{x}_i, L(t))$ is close to being Lipschitz.

Applying Lerman and Zhang (2014, (23)-(24)), and assuming that $\sum_{j=1}^d \theta_j^2 = 1$, we have

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\operatorname{dist}(\boldsymbol{x}, L(t)) \tag{87}$$
$$= \frac{\mathrm{d}}{\mathrm{d}t}\left[-\frac{\sum_{j=1}^d \theta_j((\cos(t\theta_j)\boldsymbol{v}_j + \sin(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})((-\sin(t\theta_j)\boldsymbol{v}_j + \cos(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})}{\operatorname{dist}(\boldsymbol{x}, L(t))}\right]$$
$$= -\frac{\left(\sum_{j=1}^d \theta_j((\cos(t\theta_j)\boldsymbol{v}_j + \sin(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})((-\sin(t\theta_j)\boldsymbol{v}_j + \cos(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})\right)^2}{\operatorname{dist}(\boldsymbol{x}, L(t))^3}$$
$$- \frac{\sum_{j=1}^d \theta_j^2((-\sin(t\theta_j)\boldsymbol{v}_j + \cos(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})((-\sin(t\theta_j)\boldsymbol{v}_j + \cos(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})}{\operatorname{dist}(\boldsymbol{x}, L(t))}$$
$$- \frac{\sum_{j=1}^d \theta_j^2((\cos(t\theta_j)\boldsymbol{v}_j + \sin(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})((-\cos(t\theta_j)\boldsymbol{v}_j - \sin(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})}{\operatorname{dist}(\boldsymbol{x}, L(t))}.$$

Using $|(\cos(t\theta_j)\boldsymbol{v}_j + \sin(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x}| \leq \|\boldsymbol{x}\|$ and the Cauchy-Schwarz inequality in this equation gives

$$\left(\sum_{j=1}^d \theta_j((\cos(t\theta_j)\boldsymbol{v}_j + \sin(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})((-\sin(t\theta_j)\boldsymbol{v}_j + \cos(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})\right)^2 \tag{88}$$
$$\leq \left(\sum_{j=1}^d \theta_j^2((\cos(t\theta_j)\boldsymbol{v}_j + \sin(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})^2\right)\left(\sum_{j=1}^d((-\sin(t\theta_j)\boldsymbol{v}_j + \cos(t\theta_j)\boldsymbol{u}_j)\cdot\boldsymbol{x})^2\right)$$
$$\leq \sum_{j=1}^d \theta_j^2\|\boldsymbol{x}\|^2 \operatorname{dist}(\boldsymbol{x}, L(t))^2 = \|\boldsymbol{x}\|^2 \operatorname{dist}(\boldsymbol{x}, L(t))^2.$$

Putting (87) and (88) together yields

$$\left|\frac{\mathrm{d}^2}{\mathrm{d}t^2}\operatorname{dist}(\boldsymbol{x}, L(t))\right| \leq \frac{\|\boldsymbol{x}\|^2 \operatorname{dist}(\boldsymbol{x}, L(t))^2}{\operatorname{dist}(\boldsymbol{x}, L(t))^3} + \frac{2\|\boldsymbol{x}\|^2 \sum_{j=1}^d \theta_j^2}{\operatorname{dist}(\boldsymbol{x}, L(t))} = \frac{3\|\boldsymbol{x}\|^2}{\operatorname{dist}(\boldsymbol{x}, L(t))}. \tag{89}$$

On the other hand, Lerman and Zhang (2014, Lemma 3.2) implies that

$$\left|\frac{\mathrm{d}}{\mathrm{d}t}\operatorname{dist}(\boldsymbol{x}, L(t))\right| \leq \|\boldsymbol{x}\|. \tag{90}$$

Then, define the set

$$\mathcal{G}(\mathcal{X}, L_*, L(t), c) = \left\{\boldsymbol{x} \in \mathcal{X} : \min_{t\in[0, c\theta_1(L, L_*)]} \frac{\operatorname{dist}(\boldsymbol{x}, L(t))}{\|\boldsymbol{x}\|} \leq \sqrt{c}\theta_1(L_k, L_*)\right\}. \tag{91}$$

Then, for all $0 \le s \le c\theta_1(L_k, L_*)$,

$$\begin{aligned}
&\left|\sum_{\boldsymbol{x}\in\mathcal{X}}\left(\frac{\mathrm{d}}{\mathrm{d}t}\operatorname{dist}(x, L(t))\Big|_{t=s} - \frac{\mathrm{d}}{\mathrm{d}t}\operatorname{dist}(x, L(t))\Big|_{t=0}\right)\right| \qquad (92)\\
&\quad\le \sum_{\boldsymbol{x}\in\mathcal{X}\setminus\mathcal{G}(\mathcal{X},L_*,L(t),c)}\int_{t=0}^{s}\left|\frac{\mathrm{d}^2}{\mathrm{d}t^2}\operatorname{dist}(\boldsymbol{x}, L(t))\right|\mathrm{d}t + \sum_{\boldsymbol{x}\in\mathcal{G}(\mathcal{X},L_*,L(t),c)} 2\|\boldsymbol{x}\|\\
&\quad\le \sum_{\boldsymbol{x}\in\mathcal{X}\setminus\mathcal{G}(\mathcal{X},L_*,L(t),c)} s\frac{3\|\boldsymbol{x}\|}{\sqrt{c}\theta_1(L_k, L_*)} + \sum_{\boldsymbol{x}\in\mathcal{G}(\mathcal{X},L_*,L(t),c)} 2\|\boldsymbol{x}\|\\
&\quad\le \sum_{\boldsymbol{x}\in\mathcal{X}\setminus\mathcal{G}(\mathcal{X},L_*,L(t),c))} 3\sqrt{c}\|\boldsymbol{x}\| + \sum_{\boldsymbol{x}\in\mathcal{G}(\mathcal{X},L_*,L(t),c)} 2\|\boldsymbol{x}\|.
\end{aligned}$$

In the last line of (92), the first term can be made as small as one would like by taking c_0 small enough. On the other hand, for c small enough, all points in $\mathcal{G}(\mathcal{X}, L_*, L(t), c)$ must be contained in a subspace $L \in B(L_*, \gamma) \setminus \{L_*\}$. Thus, for small enough c, the function $F(L(t); \mathcal{X})$ is approximately Lipschitz for $t \in [0, c\theta_1(L_k, L_*)]$.

As a result,

$$\begin{aligned}
F(L_k;\mathcal{X}) - F(L_{k+1};\mathcal{X})) &= F(L(0);\mathcal{X}) - F(L(c\theta_1(L_k,L_*));\mathcal{X})\\
&= -\int_{t=0}^{c\theta_1(L_k,L_*)}\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})\mathrm{d}t\\
&\ge -c\theta_1(L_k,L_*)\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\\
&\quad - c\theta_1(L_k,L_*)\left(\sum_{\boldsymbol{x}\in\mathcal{X}\setminus\mathcal{G}(L_k,L_{k+1})} 3\sqrt{c}\|\boldsymbol{x}\| + \sum_{\boldsymbol{x}\in\mathcal{G}(L_k,L_{k+1})} 2\|\boldsymbol{x}\|\right).
\end{aligned}$$

Finally, noting that $\frac{\mathrm{d}}{\mathrm{d}t}F(L(t))|_{t=0} < 0$, we use (86) and the assumption in (23) to find

$$\begin{aligned}
F(L_k) - F(L_{k+1})) &\ge \left|c\theta_1(L_k,L_*)\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\right| - \frac{c\theta_1(L_k,L_*)}{2}\left|\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\right|\\
&\ge \frac{c\theta_1(L_k,L_*)}{2}\left|\frac{\mathrm{d}}{\mathrm{d}t}F(L(t);\mathcal{X})|_{t=0}\right|.
\end{aligned}$$

D.4. Proof of Inlier Permeance Bounds

D.4.1. Proof of Proposition 11

The continuous distribution assumption implies that all directions in $S^{D-1}\cap L_*$ have nonzero probability. Let $\boldsymbol{x}$ be a random variable following the inlier distribution. By the central limit theorem, we have that

$$\frac{1}{N_{\text{in}}}\sum_{\mathcal{X}_{\text{in}}}\left(\frac{|\boldsymbol{v}^T\boldsymbol{x}_i|^2}{\|\boldsymbol{x}_i\|} - \operatorname{Var}\left(\frac{\boldsymbol{v}^T\boldsymbol{x}}{\|\boldsymbol{x}\|^{1/2}}\right)\right) \xrightarrow{d} N(0,1).$$

We also have that $\max_i \|\boldsymbol{x}_i\| < M$ by the bounded-support assumption assumption. Thus, for large enough N_{in} and a covering argument, we have for some absolute constant C that

$$\min_{\boldsymbol{v}\in L_*\cap S^{D-1}}\sum_{\mathcal{X}_{\text{in}}}\frac{|\boldsymbol{v}^T\boldsymbol{x}_i|^2}{\|\boldsymbol{x}_i\|} \ge C\frac{N_{\text{in}}}{M}\min_{\boldsymbol{v}\in L_*\cap S^{D-1}}\operatorname{Var}\left(\frac{\boldsymbol{v}^T\boldsymbol{x}}{\|\boldsymbol{x}\|^{1/2}}\right), \text{ w.h.p.}$$

This can be done by using, e.g., Hoeffding's inequality on the bounded random variable $\boldsymbol{v}^T\boldsymbol{x}/\|\boldsymbol{x}\|^{1/2}$.

D.4.2. Proof of Proposition 12

For the inliers, we need to bound $\mathcal{P}(\mathcal{X}_{\text{in}})$. Using Theorem 3.1 in Lu and Pearce (2000) and letting $\widetilde{\cdot}$ denote the spherization operator,

$$\lambda_d\left(\sum_{\mathcal{X}_{\text{in}}}\frac{\boldsymbol{x}_i\boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|}\right) = \sigma_d(\widetilde{\boldsymbol{X}_{\text{in}}}\boldsymbol{X}_{\text{in}}^T) \ge \sigma_d(\widetilde{\boldsymbol{X}_{\text{in}}})\sigma_d(\boldsymbol{X}_{\text{in}}). \qquad (93)$$

We proceed by bounding the last term in (93). Notice that $\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{X}_{\text{in}}$ has directional variance $1/d$ in all directions of L_* (i.e., these transformed inliers are isotropic with variance $1/d$). We again apply Theorem 3.1 in Lu and Pearce (2000) and then apply Theorem 5.39 in Vershynin (2012b) to $\sigma_d\left(\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{X}_{\text{in}}\right)$, where we scale by the standard deviation $1/\sqrt{d}$ and choose $t = a\sqrt{N_{\text{in}}}$:

$$\begin{aligned}
\sigma_d\left(\boldsymbol{X}_{\text{in}}\right) &= \sigma_d\left(\boldsymbol{\Sigma}_{\text{in}}^{1/2}\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{X}_{\text{in}}\right) \ge \sigma_d\left(\boldsymbol{\Sigma}_{\text{in}}^{1/2}\right)\sigma_d\left(\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{X}_{\text{in}}\right) \qquad (94)\\
&> \lambda_d\left(\boldsymbol{\Sigma}_{\text{in}}^{1/2}\right)\left((1-a)\sqrt{\frac{N_{\text{in}}}{d}} - C_1\right), \text{ w.p. at least } 1 - 2e^{-c_1a^2N_{\text{in}}}.
\end{aligned}$$

Here, c_1 and C_1 are constants that depend on the sub-Gaussian norm of $\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}$, where $\boldsymbol{x}$ follows the inlier distribution, and a must be chosen such that $(1-a)^2N_{\text{in}} > C_1^2d$. On the other hand, $\widetilde{\boldsymbol{X}}_{\text{in}}$ is still sub-Gaussian and $\widetilde{\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{X}_{\text{in}}}$ is isotropic sub-Gaussian. Therefore, by using (94) and applying Theorem 3.1 in Lu and Pearce (2000), we have

$$\begin{aligned}
\sigma_d\left(\widetilde{\boldsymbol{X}_{\text{in}}}\right) &= \sigma_d\left(\left[\frac{\boldsymbol{x}_1}{\|\boldsymbol{x}_1\|},\ldots,\frac{\boldsymbol{x}_{N_{\text{in}}}}{\|\boldsymbol{x}_{N_{\text{in}}}\|}\right]\right) \qquad (95)\\
&= \sigma_d\left(\boldsymbol{\Sigma}_{\text{in}}^{1/2}\left[\frac{\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}_1}{\|\boldsymbol{\Sigma}_{\text{in}}^{1/2}\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}_1\|},\ldots,\frac{\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}_{N_{\text{in}}}}{\|\boldsymbol{\Sigma}_{\text{in}}^{1/2}\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}_{N_{\text{in}}}\|}\right]\right)\\
&\ge \sigma_d\left(\frac{\boldsymbol{\Sigma}_{\text{in}}^{1/2}}{\|\boldsymbol{\Sigma}_{\text{in}}^{1/2}\|_2}\left[\frac{\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}_1}{\|\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}_1\|},\ldots,\frac{\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}_{N_{\text{in}}}}{\|\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}_{N_{\text{in}}}\|}\right]\right)\\
&\ge \frac{\lambda_d(\boldsymbol{\Sigma}_{\text{in}}^{1/2})}{\lambda_1(\boldsymbol{\Sigma}_{\text{in}}^{1/2})}\sigma_d\left(\widetilde{\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{X}_{\text{in}}}\right)\\
&> \frac{\lambda_d(\boldsymbol{\Sigma}_{\text{in}}^{1/2})}{\lambda_1(\boldsymbol{\Sigma}_{\text{in}}^{1/2})}\left((1-a)\sqrt{\frac{N_{\text{in}}}{d}} - C_1'\right), \text{ w.p. at least } 1 - 2e^{-c_1'a^2N_{\text{in}}}.
\end{aligned}$$

Here, c_1' and C_1' are constants that depend on the sub-Gaussian norm of $\widetilde{\boldsymbol{\Sigma}_{\text{in}}^{-1/2}\boldsymbol{x}}$, where $\boldsymbol{x}$ is a random vector that follows the inlier distribution, and a must chosen such that $(1-a)^2 N_{\text{in}} > C_1'^2 d$. Therefore, for a to satisfy both requirements, we need $(1-a)^2 N_{\text{in}} > \max(C_1, C_1)'^2 d$. Abusing notation, we let c_1 be the minimum of c_1 and c_1', and we let C_1 be the maximum of C_1 and C_1'. Putting (94) and (95) together, we find

$$\lambda_d\left(\sum_{\mathcal{X}_{\text{in}}} \frac{\boldsymbol{x}_i \boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|}\right) > \frac{\lambda_d(\boldsymbol{\Sigma}_{\text{in}})}{\lambda_1(\boldsymbol{\Sigma}_{\text{in}})^{1/2}}\left((1-a)\sqrt{\frac{N_{\text{in}}}{d}} - C_1\right)^2, \text{ w.p. at least } 1 - 4e^{-c_1 a^2 N_{\text{in}}}. \quad (96)$$

D.5. Proof of Proposition 15

To prove this proposition, we only need to show that PCA initializes in $B(L_*, \gamma)$. Note that the covariance matrix for the data $\mathcal{X}$ is $\boldsymbol{\Sigma} = N_{\text{out}}\boldsymbol{\Sigma}_{\text{out}}/(ND_{\text{out}}) + N_{\text{in}}\boldsymbol{\Sigma}_{\text{in}}/(Nd)$. We want to see how close the sample covariance approximates $\boldsymbol{\Sigma}_{\text{in}}$, since the PCA subspace is spanned by the top d eigenvectors of the sample covariance. Denoting the sample covariance by $\boldsymbol{\Sigma}_N$, let $\boldsymbol{V}_{PCA}$ be its top d eigenvectors. Also, let $\boldsymbol{V}^*$ be the top d eigenvectors of $\boldsymbol{\Sigma}_{\text{in}}$ and let $\boldsymbol{V}$ be the top d eigenvectors of $\boldsymbol{\Sigma}$. We note that

$$|\sin(\theta_1(\boldsymbol{V}_{PCA}, \boldsymbol{V}^*))| \le |\sin(\theta_1(\boldsymbol{V}_{PCA}, \boldsymbol{V}))| + |\sin(\theta_1(\boldsymbol{V}, \boldsymbol{V}^*))|. \quad (97)$$

To deal with the last term in (97), the Davis-Kahan $\sin\theta$ Theorem (Davis and Kahan, 1970), or more precisely Corollary 3.1 of Vu et al. (2013), gives

$$|\sin(\theta_1(\boldsymbol{V}, \boldsymbol{V}^*)| \le \sqrt{2}\frac{\lambda_1(N_{\text{out}}\boldsymbol{\Sigma}_{\text{out}}/(ND_{\text{out}}))}{\lambda_d(N_{\text{in}}\boldsymbol{\Sigma}_{\text{in}}/(Nd))} = \sqrt{2}\frac{N_{\text{out}}}{N_{\text{in}}}\frac{d}{D_{\text{out}}}\frac{\lambda_1(\boldsymbol{\Sigma}_{\text{out}})}{\lambda_d(\boldsymbol{\Sigma}_{\text{in}})}.$$

Thus,

$$\sin(\gamma) > \sqrt{2}\frac{N_{\text{out}}}{N_{\text{in}}}\frac{d}{D_{\text{out}}}\frac{\lambda_1(\boldsymbol{\Sigma}_{\text{out}})}{\lambda_d(\boldsymbol{\Sigma}_{\text{in}})} \quad (98)$$

is a sufficient condition for $|\sin(\theta_1(\boldsymbol{V}, \boldsymbol{V}^*)| < \sin(\gamma)$.

On the other hand, similar to the derivation of (94), for the first term in the right hand side of (97), we must bound how close the sample covariance is to the true covariance. Proposition 2.1 of Vershynin (2012a) implies that, for every $\delta > 0$,

$$\|\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_N\|_2 \le \epsilon(\delta, \boldsymbol{\Sigma}_{\text{in}}, \boldsymbol{\Sigma}_{\text{out}})\left(\frac{D_{\text{out}}}{N}\right)^{\frac{1}{2}}, \quad (99)$$

with probability at least $1-\delta$, where $\epsilon(\delta, \boldsymbol{\Sigma}_{\text{in}}, \boldsymbol{\Sigma}_{\text{out}})$ is a constant depending on δ, $\boldsymbol{\Sigma}_{\text{in}}$, and $\boldsymbol{\Sigma}_{\text{out}}$. By assumption, $\boldsymbol{\Sigma}$ has a positive dth eigengap. Thus, another application of the Davis-Kahan $\sin\theta$ Theorem yields

$$|\sin(\theta_1(\boldsymbol{V}_{PCA}, \boldsymbol{V}))| \le \frac{\|\boldsymbol{\Sigma} - \boldsymbol{\Sigma}_N\|_2}{\lambda_d(\boldsymbol{\Sigma}) - \lambda_{d+1}(\boldsymbol{\Sigma})} \le \epsilon_2(\delta, \boldsymbol{\Sigma}_{\text{in}}, \boldsymbol{\Sigma}_{\text{out}})\left(\frac{dD}{N}\right)^{\frac{1}{2}},$$

with probability $1-\delta$. For large enough N, we have

$$|\sin(\theta_1(\boldsymbol{V}_{PCA}, \boldsymbol{V})| \le \sin(\gamma) - |\sin(\theta_1(\boldsymbol{V}, \boldsymbol{V}^*)|. \quad (100)$$

Rearranging (100) and using the triangle inequality in (97) yields

$$|\sin(\theta_1(\boldsymbol{V}_{PCA}, \boldsymbol{V}^*))| \le \sin(\gamma),$$

with probability $1-\delta$.

D.6. Proof of Theorem 17

We begin by bounding the outlier term. The last term on the right hand side of (47) is optimally bounded in the proof of Theorem 14. On the other hand, in the following proposition, we obtain tighter bounds for the first term on the right hand side of this inequality.

Proposition 22 *Suppose that $\mathcal{X}$ is drawn from the Haystack Model with parameters N_{in}, σ_{in}, N_{out}, σ_{out}, D, and d. Then, for $D - d \ge 3$*

$$\max_{L\in G(D,d)} \left\|\widetilde{\boldsymbol{Q}_L \boldsymbol{X}_{\text{out}}}\right\|_2 \le \frac{7}{2}\sqrt{\frac{N_{\text{out}}}{D-d}} + \sqrt{2}, \quad (101)$$
$$\text{w.p. at least } 1 - \exp\left(-\frac{N_{\text{out}}}{4}\right) - C_1\exp\left(-\frac{N_{\text{out}}}{4(D-d)} + \frac{d(D-d)\log(D)}{2}\right),$$

where C_1 is an absolute constant.

Proof [Proof of Proposition 22] First, for a single L, Lemma 8.4 in Lerman et al. (2015) yields

$$\left\|\widetilde{\boldsymbol{Q}_L \boldsymbol{X}_{\text{out}}}\right\|_2 \le \left(\sqrt{\frac{N_{\text{out}}}{D-d-0.5}} + \sqrt{2} + \frac{t}{\sqrt{D-d-0.5}}\right), \text{ w.p. at least } 1 - 1.5e^{-t^2}. \quad (102)$$

We set $t = \sqrt{N_{\text{out}}}/2$ to obtain the desired bound for a single subspace L. However, in order to cover all of $G(D,d)$, we much use a more complicated argument.

Assume that we have two subspaces L_0 and L_1 such that $\theta_1(L_0, L_1) < 1/(2\sqrt{D})$. First, due to the triangle inequality,

$$\left\|\widetilde{\boldsymbol{Q}_{L_0}\boldsymbol{X}_{\text{out}}} - \widetilde{\boldsymbol{Q}_{L_1}\boldsymbol{X}_{\text{out}}}\right\|_2 < \left\|\widetilde{\boldsymbol{Q}_{L_0}\boldsymbol{X}^0_{\text{out}}} - \widetilde{\boldsymbol{Q}_{L_1}\boldsymbol{X}^0_{\text{out}}}\right\|_2 + \left\|\widetilde{\boldsymbol{Q}_{L_0}\boldsymbol{X}^1_{\text{out}}} - \widetilde{\boldsymbol{Q}_{L_1}\boldsymbol{X}^1_{\text{out}}}\right\|_2. \quad (103)$$

Here, we have partitioned the outliers into two parts. The following set-valued functions define these parts. Let $L(t)$ be the geodesic between L_0 and L_1 such that $L(0) = L_0$ and $L(1) = L_1$. Then, we define

$$\mathcal{X}^0_{\text{out}} = \left\{\boldsymbol{x} \in \mathcal{X}_{\text{out}} : \min_{t\in[0,1]} \angle(\boldsymbol{x}, L(t)) < \frac{1}{2\sqrt{D}}\right\}, \quad (104)$$

$$\mathcal{X}^1_{\text{out}} = \left\{\boldsymbol{x} \in \mathcal{X}_{\text{out}} : \min_{t\in[0,1]} \angle(\boldsymbol{x}, L(t)) \ge \frac{1}{2\sqrt{D}}\right\}. \quad (105)$$

Notice that

$$\mathcal{X}^0_{\text{out}} \subset \left\{\boldsymbol{x} \in \mathcal{X}_{\text{out}} : \angle(\boldsymbol{x}, L_0) < \frac{1}{\sqrt{D}}\right\}. \quad (106)$$

With these datasets, their data matrices are $\boldsymbol{X}^0_{\text{out}}$ and $\boldsymbol{X}^1_{\text{out}}$.

For the last term in (103), since $\theta_1(L_0, L_1) < 1/(2\sqrt{D})$, we have

$$\left\| \widetilde{\boldsymbol{Q}_{L_0} \boldsymbol{X}^1_{\text{out}}} - \widetilde{\boldsymbol{Q}_{L_1} \boldsymbol{X}^1_{\text{out}}} \right\|_2 < \frac{1}{2}\sqrt{\frac{N_{\text{out}}}{D}}. \tag{107}$$

On the other hand, we must look at the concentration of points around subspaces for the second to last term in (103). For any given L, we have the following concentration lemma.

Lemma 23 *If $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, then*

$$\Pr\left(\angle(\boldsymbol{x}, L) < \frac{1}{2}\sqrt{\frac{D-d}{D}} \right) < \exp\left(-\frac{D-d}{2} \right).$$

Proof This is a direct consequence of Lemma 2.2 in Dasgupta and Gupta (2003). ■

For $D - d \geq 3$, we have

$$\Pr\left(\angle(\boldsymbol{x}, L) < \frac{1}{2}\sqrt{\frac{D-d}{D}} \right) \leq \exp\left(-\frac{D-d}{2} \right) \leq \frac{1}{D-d}. \tag{108}$$

Using (108) together with a loose Chernoff bound for the concentration of binomial random variables (Mitzenmacher and Upfal, 2005), we know that

$$\Pr\left(\#\left(\left\{ \boldsymbol{x} \in \mathcal{X}_{\text{out}} : \angle(\boldsymbol{x}, L) < \frac{1}{2}\sqrt{\frac{D-d}{D}} \right\} \right) > \frac{3}{2}\frac{N_{\text{out}}}{D-d} \right) < \exp\left(-\frac{N_{\text{out}}}{4(D-d)} \right). \tag{109}$$

Thus, we use (109) on the second to last term in (103) to find

$$\left\| \widetilde{\boldsymbol{Q}_{L_0} \boldsymbol{X}^0_{\text{out}}} - \widetilde{\boldsymbol{Q}_{L_1} \boldsymbol{X}^0_{\text{out}}} \right\|_2 < \sqrt{\#(\mathcal{X}^0_{\text{out}})} \leq \sqrt{\frac{3}{2}\frac{N_{\text{out}}}{D-d}}, \tag{110}$$
$$\text{w.p. at least } 1 - \exp\left(-\frac{N_{\text{out}}}{4(D-d)} \right).$$

Notice that also, by construction, this is true for all $L_0', L_1' \in B(L(t), \frac{1}{4}\sqrt{(D-d)/D})$.

We finish by completing the covering argument on $G(D, d)$. By remark 8.4 of (Szarek, 1983), $G(D, d)$ can be covered by $(C)^{d(D-d)}/(\gamma_1)^{d(D-d)}$ balls of radius γ_1. This means that, using a union bound and taking (102), (103), (107), and (110) together,

$$\Pr\left(\max_{L \in G(D,d)} \|\widetilde{\boldsymbol{Q}_L \boldsymbol{X}_{\text{out}}}\| < \frac{7}{2}\sqrt{\frac{N_{\text{out}}}{D-d-0.5}} + \sqrt{2} \right) > \tag{111}$$
$$1 - C_1 \exp\left(-\frac{N_{\text{out}}}{4(D-d)} + C_2 d(D-d)\log\left(\frac{D}{D-d} \right) \right) - \exp\left(-\frac{N_{\text{out}}}{4} \right),$$

where C_1 and C_2 are absolute constants. Thus, we see that N must be on the order $N = O(d(D-d)^2 \log(D))$, and we have have the desired statement. ■

We now continue with the proof of Theorem 17. In Proposition 22, We see that N needs to be on the order of $d(D-d)^2 \log(D)$, which is not the case of the small sample regime. On the other hand, Theorem 5.39 of Vershynin (2012b) states that

$$\|\boldsymbol{X}_{\text{out}}\|_2 \leq \sigma_{\text{out}} \left[\frac{5}{4}\sqrt{\frac{N_{\text{out}}}{D}} + 1 \right], \text{ w.p. at least } 1 - 2e^{-N_{\text{out}}/16}. \tag{112}$$

We combine the result of Proposition 22 with (112) to find that

$$\|\widetilde{\boldsymbol{Q}_L \boldsymbol{X}_{\text{out}}}\|_2 \|\boldsymbol{X}_{\text{out}}\|_2 \leq \frac{7}{2}\frac{N_{\text{out}}}{\sqrt{D(D-d)}} + O\left(\sigma_{\text{out}}\sqrt{\frac{N_{\text{out}}}{D}} \right), \tag{113}$$
$$\text{w.p. at least } 1 - 2e^{-N_{\text{out}}/16} - e^{-N_{\text{out}}/4} - C_1 \exp\left(-\frac{N_{\text{out}}}{4(D-d)} + C_2 d(D-d)\log\left(\frac{D}{D-d} \right) \right).$$

We can also improve the results for the inlier permeance bound since they are already isometric. In this case, we have that (94) still holds. On the other hand, $\widetilde{\boldsymbol{X}_{\text{in}}}$ is isotropic, which implies that

$$\sigma_d\left(\widetilde{\boldsymbol{X}_{\text{in}}} \right) > \left((1-a)\sqrt{\frac{N_{\text{in}}}{d}} - C_1' \right), \text{ w.p. at least } 1 - 2e^{-c_1' a^2 N_{\text{in}}}, \tag{114}$$

where as before we must have $(1-a)^2 N_{\text{in}}^2 > C_1'^2 d$. Abusing notation again (as we did in the proof of Proposition 12), we let c_1 be the minimum of c_1 and c_1' and C_1 be the max of C_1 and C_1' and we find that

$$\lambda_d\left(\sum_{\mathcal{X}_{\text{in}}} \frac{\boldsymbol{x}_i \boldsymbol{x}_i^T}{\|\boldsymbol{x}_i\|} \right) > \sigma_{\text{in}}\left((1-a)\sqrt{\frac{N_{\text{in}}}{d}} - C_1 \right)^2, \text{ w.p. at least } 1 - 4e^{-c_1 a^2 N_{\text{in}}}. \tag{115}$$

This results in the following exact statement for the Haystack Model. We combine (113) with (115) to find that if (48) holds, then $\mathcal{S}(\mathcal{X}, L_*, \gamma) > 0$ with probability at least (49).

D.7. Proof of Theorem 18

Here, we will show that GGD can recover the underlying subspace for any fixed fraction of outliers, provided that N_{out} is large enough. We denote the percentage of inliers by α_1 and the percentage of outliers by α_0. Then, the SNR is given by α_1/α_0.

Assume for simplicity that $\sigma_{\text{out}} = 1$, since the general case follows from the same logic. First, we will bound the maximum norm of the set of outlier points. Each $\boldsymbol{x}_i$ is i.i.d. $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/D)$, we can bound its norm by

$$\Pr\left(\|\boldsymbol{x}_i\| \leq 1 + \frac{t}{\sqrt{D}} \right) \geq 1 - \exp\left(-ct^2 \right), \tag{116}$$

where c is just some universal constant. Thus, applying a union bound to (116) yields

$$\Pr\left(\max_i \|\boldsymbol{x}_i\| \leq 1 + \frac{t}{\sqrt{D}} \right) \geq 1 - N_{\text{out}} \exp\left(-ct^2 \right). \tag{117}$$

We can use a value of $t = \sqrt{D}N_{\text{out}}^{1/6}$ to find

$$\Pr\left(\max_i \|\boldsymbol{x}_i\| \leq 1 + N^{1/6}\right) \geq 1 - N_{\text{out}} \exp\left(-cDN_{\text{out}}^{1/3}\right). \tag{118}$$

Next, consider the alignment for a single fixed subspace $L \neq L_*$:

$$\mathcal{A}(\mathcal{X}_{\text{out}}, L) = \left\| \sum_{\mathcal{X}_{\text{out}}} \frac{\boldsymbol{Q}_L \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{P}_L}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} \right\|_2. \tag{119}$$

Taking an expectation within the norm in (119) yields

$$\mathbb{E} \sum_{\mathcal{X}_{\text{out}}} \frac{\boldsymbol{Q}_L \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{P}_L}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} = \boldsymbol{0}. \tag{120}$$

This follows by the symmetry of the outlier distribution, $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/D)$. Fix arbitrary vectors $\boldsymbol{u} \in \text{Sp}(\boldsymbol{Q}_L) \cap S^{D-1}$ and $\boldsymbol{v} \in \text{Sp}(\boldsymbol{P}_L) \cap S^{D-1}$. Then, (120) implies that

$$\mathbb{E} \sum_{\mathcal{X}_{\text{out}}} \frac{\boldsymbol{u}^T \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{v}}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|} = 0. \tag{121}$$

In the following, we will continue to use $\boldsymbol{u}$ and $\boldsymbol{v}$ defined in this way.

To this end, define the following random variable

$$J(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) = \frac{\boldsymbol{u}^T \boldsymbol{x} \boldsymbol{x}^T \boldsymbol{v}}{\|\boldsymbol{Q}_L \boldsymbol{x}\|}, \tag{122}$$

where $\boldsymbol{x}$ is distributed $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/D)$ (i.e., it is an outlier in the Haystack model). We will first give a concentration bound for $\sum_{\mathcal{X}_{\text{out}}} J(\boldsymbol{x}_i, \boldsymbol{u}, \boldsymbol{v})$, which appears in (121). Notice that $J(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})$ is a mean zero random variable and is bounded by

$$|J(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})| \leq |\boldsymbol{x}^T \boldsymbol{v}|. \tag{123}$$

Here, $\boldsymbol{x}^T \boldsymbol{v}$ is Gaussian with variance $1/D$. Thus, $J(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})$ is sub-Gaussian with variance proxy $1/D$. This implies that

$$\Pr\left(\left|\sum_{\mathcal{X}_{\text{out}}} J(\boldsymbol{x}_i, \boldsymbol{u}, \boldsymbol{v})\right| > N_{\text{out}} t\right) \leq 2 \exp\left(-\frac{t^2 D N_{\text{out}}}{2}\right). \tag{124}$$

Letting $t = N_{\text{out}}^{-1/3}$ we find that

$$\Pr\left(\left|\sum_{\mathcal{X}_{\text{out}}} J(\boldsymbol{x}_i, \boldsymbol{u}, \boldsymbol{v})\right| > N_{\text{out}}^{2/3}\right) \leq 2 \exp\left(-\frac{D N_{\text{out}}^{1/3}}{2}\right). \tag{125}$$

Notice that $J(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v})$ is continuous as a function of $\boldsymbol{u}$ and $\boldsymbol{v}$. Using (123), between two points $\boldsymbol{v}_1, \boldsymbol{v}_2 \in S^{D-1}$, we can bound the deviation in $\sum_{\mathcal{X}_{\text{out}}} J(\boldsymbol{x}_i, \boldsymbol{u}, \cdot)$ by

$$\left|\sum_{\mathcal{X}_{\text{out}}} J(\boldsymbol{x}_i, \boldsymbol{u}, \boldsymbol{v}_1) - \sum_{\mathcal{X}_{\text{out}}} J(\boldsymbol{x}_i, \boldsymbol{u}, \boldsymbol{v}_2)\right| \leq \sum_{\mathcal{X}_{\text{out}}} |(\boldsymbol{v}_1 - \boldsymbol{v}_2)^T \boldsymbol{x}_i| \leq \|\boldsymbol{v}_1 - \boldsymbol{v}_2\| N_{\text{out}} \max_i \|\boldsymbol{x}_i\|. \tag{126}$$

Combining (118), (125) and (126) with $\|\boldsymbol{v}_1 - \boldsymbol{v}_2\| < N_{\text{out}}^{-1/3}$ yields

$$\Pr\left(\max_{\boldsymbol{v}_2 : \|\boldsymbol{v}_2 - \boldsymbol{v}_1\| \leq N_{\text{out}}^{-1/3}} \left|\sum_i J(\boldsymbol{x}_i, \boldsymbol{u}, \boldsymbol{v}_2)\right| \leq 3N_{\text{out}}^{5/6}\right) \geq 1 - N_{\text{out}} \exp\left(-cDN_{\text{out}}^{1/3}\right) - 2\exp\left(-\frac{DN_{\text{out}}^{1/3}}{2}\right). \tag{127}$$

Since $L \cap S^{D-1}$ can be covered by $\left(2N_{\text{out}}^{1/3} + 1\right)^d$ balls of radius $N_{\text{out}}^{-1/3}$, we use a covering argument with (127) to find

$$\Pr\left(\max_{\boldsymbol{v} \in L \cap S^{D-1}} \left|\sum_{\mathcal{X}_{\text{out}}} J(\boldsymbol{x}_i, \boldsymbol{u}, \boldsymbol{v})\right| \leq 3N_{\text{out}}^{5/6}\right) \geq 1 - N_{\text{out}} \exp\left(-cDN_{\text{out}}^{1/3}\right) - 2\left(2N_{\text{out}}^{1/3} + 1\right)^d \exp\left(-\frac{DN_{\text{out}}^{1/3}}{2}\right). \tag{128}$$

Notice that we do not include the probability from (118) in the covering, since this probability holds independently of the choice of $\boldsymbol{v} \in L \cap S^{D-1}$.

The previous argument was actually independent of choice of $\boldsymbol{u} \in L^\perp \cap S^{D-1}$. Thus, we finally have

$$\Pr\left(\left\|\sum_{\mathcal{X}_{\text{out}}} \frac{\boldsymbol{Q}_L \boldsymbol{x}_i \boldsymbol{x}_i^T \boldsymbol{P}_L}{\|\boldsymbol{Q}_L \boldsymbol{x}_i\|}\right\|_2 \leq 3N_{\text{out}}^{5/6}\right) \geq 1 - N_{\text{out}} \exp\left(-cDN_{\text{out}}^{1/3}\right) - 2\left(2N_{\text{out}}^{1/3} + 1\right)^d \exp\left(-\frac{DN_{\text{out}}^{1/3}}{2}\right). \tag{129}$$

Now that we have covered a single $L \in G(D, d)$, we must extend this to all of $G(D, d)$ by another covering argument. From Lemma 2.2 of Dasgupta and Gupta (2003), we have, for each $\boldsymbol{x}_i \in \mathcal{X}_{\text{out}}$

$$\Pr\left(\angle(\boldsymbol{x}_i, L) < \frac{\pi}{2}\sqrt{\frac{\beta(D-d)}{D}}\right) \leq \exp\left(\frac{D-d}{2}(1 + \log(\beta))\right). \tag{130}$$

If we choose $\beta < N_{\text{out}}^{-2/3}$, then it is not hard to show that

$$\Pr\left(\angle(\boldsymbol{x}_i, L) < \frac{\pi}{2}\sqrt{\frac{\beta(D-d)}{D}}\right) \leq N_{\text{out}}^{-1/3}. \tag{131}$$

Define the cone around a subspace L as

$$\mathcal{C}(L, \xi) = \{\boldsymbol{x} \in \mathbb{R}^D : \angle(\boldsymbol{x}, L) < \xi\}. \tag{132}$$

Using a loose Chernoff bound for the concentration of binomial random variables (Mitzenmacher and Upfal, 2005), we have

$$\Pr\left(\left|\#\left(\mathcal{X}_{\text{out}}\cap\mathcal{C}\left(L,N_{\text{out}}^{-1/3}\frac{\pi}{2}\sqrt{\frac{(D-d)}{D}}\right)\right)-\delta N_{\text{out}}^{2/3}\right|\geq N_{\text{out}}^{2/3}\right)\leq\exp\left(-\frac{N_{\text{out}}^{2/3}\delta^2}{3}\right). \tag{133}$$

Choosing $\delta = N_{\text{out}}^{-1/6}$ yields

$$\Pr\left(\left|\#\left(\mathcal{X}_{\text{out}}\cap\mathcal{C}\left(L,N_{\text{out}}^{-1/3}\frac{\pi}{2}\sqrt{\frac{(D-d)}{D}}\right)\right)-N_{\text{out}}^{1/2}\right|\geq N_{\text{out}}^{2/3}\right)\leq\exp\left(-\frac{N_{\text{out}}^{1/3}}{3}\right). \tag{134}$$

For any $L_0 \in G(D,d)$, and $L_1 \in B(L_0,\xi)$, we can separate the alignment term into two parts: those $\boldsymbol{x}_i$ that are close to L_0 and L_1 and those that lie further away. The idea behind this is that points that are far away from L_0 and L_1 will contribute similar amounts to the alignment. On the other hand, those that are very close can contribute at most 2 times their norm to the alignment.

Rigorously, we write

$$\begin{aligned}
&\left\|\sum_{\mathcal{X}_{\text{out}}}\frac{\boldsymbol{Q}_{L_0}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_{L_0}}{\|\boldsymbol{Q}_{L_0}\boldsymbol{x}_i\|_2}-\frac{\boldsymbol{Q}_{L_1}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_{L_1}}{\|\boldsymbol{Q}_{L_1}\boldsymbol{x}_i\|_2}\right\|_2 \\
&\qquad\leq\left\|\sum_{\boldsymbol{x}_i\in\mathcal{X}_{\text{out}}\cap\mathcal{C}(L_0,\xi)}\frac{\boldsymbol{Q}_{L_0}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_{L_0}}{\|\boldsymbol{Q}_{L_0}\boldsymbol{x}_i\|_2}-\frac{\boldsymbol{Q}_{L_1}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_{L_1}}{\|\boldsymbol{Q}_{L_1}\boldsymbol{x}_i\|_2}\right\|_2+ \\
&\qquad\qquad\left\|\sum_{\boldsymbol{x}_i\in\mathcal{X}_{\text{out}}\cap\mathcal{C}(L_0,\xi)^C}\frac{\boldsymbol{Q}_{L_0}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_{L_0}}{\|\boldsymbol{Q}_{L_0}\boldsymbol{x}_i\|_2}-\frac{\boldsymbol{Q}_{L_1}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_{L_1}}{\|\boldsymbol{Q}_{L_1}\boldsymbol{x}_i\|_2}\right\|_2 \\
&\qquad\leq\sum_{\boldsymbol{x}_i\in\mathcal{X}_{\text{out}}\cap\mathcal{C}(L_0,\xi)}2\|\boldsymbol{x}_i\|+2\xi\sum_{\boldsymbol{x}_i\in\mathcal{X}_{\text{out}}\cap\mathcal{C}(L_0,\xi)^C}\|\boldsymbol{x}_i\|_2.
\end{aligned} \tag{135}$$

We will examining (135) term by term. For the first term, if we choose $\xi = N_{\text{out}}^{-1/3}$ and combine (118) and (134), we find that

$$\sum_{\boldsymbol{x}_i\in\mathcal{X}_{\text{out}}\cap\mathcal{C}(L_0,\xi)}2\|\boldsymbol{x}_i\|\leq 2\#(\mathcal{X}_{\text{out}}\cap\mathcal{C}(L_0,\xi))\max_{\mathcal{X}_{\text{out}}}\|\boldsymbol{x}_i\|\leq 2\left(1+N_{\text{out}}^{1/6}\right)N_{\text{out}}^{2/3}, \tag{136}$$

$$\text{w.p. at least } 1-N_{\text{out}}\exp\left(-cDN_{\text{out}}^{1/3}\right)-\exp\left(-\frac{N_{\text{out}}^{1/3}}{3}\right).$$

For the second term, choosing $\xi = N_{\text{out}}^{-1/3}$ and again using (118) yields

$$2\xi\sum_{\boldsymbol{x}_i\in\mathcal{X}_{\text{out}}\cap\mathcal{C}(L_0,\xi)^C}\|\boldsymbol{x}_i\|_2\leq 2\left(1+N_{\text{out}}^{1/6}\right)N_{\text{out}}^{2/3}+o(1), \tag{137}$$

$$\text{w.p. at least } 1-N_{\text{out}}\exp\left(-cDN_{\text{out}}^{1/3}\right).$$

Putting (136) and (137) yields

$$\left\|\sum_{\mathcal{X}_{\text{out}}}\frac{\boldsymbol{Q}_{L_0}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_{L_0}}{\|\boldsymbol{Q}_{L_0}\boldsymbol{x}_i\|_2}-\frac{\boldsymbol{Q}_{L_1}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_{L_1}}{\|\boldsymbol{Q}_{L_1}\boldsymbol{x}_i\|_2}\right\|_2\leq 8N_{\text{out}}^{5/6}, \tag{138}$$

$$\text{w.p. at least } 1-N_{\text{out}}\exp\left(-cDN_{\text{out}}^{1/3}\right)-\exp\left(-\frac{N_{\text{out}}^{1/3}}{3}\right).$$

As already mentioned in Appendix D.6 (following Szarek (1983)), $G(D,d)$ can be covered by $(C_1)^{d(D-d)}/(\gamma_1)^{d(D-d)}$ balls of radius γ_1. Thus, by a union bound with (129) and (138),

$$\left\|\sum_{\mathcal{X}_{\text{out}}}\frac{\boldsymbol{Q}_L\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{P}_L}{\|\boldsymbol{Q}_L\boldsymbol{x}_i\|}\right\|_2\leq 11N_{\text{out}}^{5/6},\ \forall\ L\in G(D,d), \tag{139}$$

$$\text{w.p. at least } 1-N_{\text{out}}\exp\left(-cDN_{\text{out}}^{1/3}\right)-2\left((C_1)^{d(D-d)}(N_{\text{out}}^{1/3})^{d(D-d)}\right)\cdot$$
$$\left(\left(2N_{\text{out}}^{1/3}+1\right)^d\exp\left(-\frac{DN_{\text{out}}^{1/3}}{2}\right)+\exp\left(-\frac{N_{\text{out}}^{1/3}}{3}\right)\right).$$

As $N_{\text{out}}\to\infty$, we see that this probability goes to 1.

On the other hand, we recall the bound on the permeance of inliers from (115). Combining this with (139) and adding back in the scale factor σ_{out}, we have that if

$$\cos(\gamma)\sigma_{\text{in}}\left((1-a)\sqrt{\frac{N_{\text{in}}}{d}}-C_1\right)^2\geq 11\sigma_{\text{out}}N_{\text{out}}^{5/6}, \tag{140}$$

then $\mathcal{S}(\mathcal{X},L_*,\gamma)>0$ w.o.p. This equates to

$$\text{SNR}\geq\frac{11d\sigma_{\text{out}}}{\cos(\gamma)N_{\text{out}}^{1/6}\sigma_{\text{in}}(1-a)^2}+O\left(\frac{\sqrt{N_{\text{in}}}}{N_{\text{out}}}\right), \tag{141}$$

which goes to 0 as $N\to\infty$ for any fixed fraction of outliers. We see, in terms of dependence on parameters in (139) and (141), N_{out} must be at least on the order of $O(\max(d^3D^3\log^3(N_{\text{out}}),(dN_{\text{out}}/N_{\text{in}})^6))$. This is due to the fact that (141) must hold and the limiting probability in (139) is

$$2\left((C_1)^{d(D-d)}(N_{\text{out}}^{1/3})^{d(D-d)}\right)\exp\left(-\frac{N_{\text{out}}^{1/3}}{3}\right),$$

which is only close to zero when the number of outliers is more than the specified regime.

References

P.-A. Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of Grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematica*, 80(2):199–220, 2004.

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

E. Arias-Castro and J. Wang. RANSAC algorithms for subspace recovery and subspace clustering. *ArXiv e-prints*, November 2017.

E. Arias-Castro, G. Chen, and G. Lerman. Spectral clustering based on local linear approximations. *Electron. J. Statist.*, 5:1537–1587, 2011.

S. Arora, R. Ge, T. Ma, and A. Moitra. Simple, efficient, and neural algorithms for sparse coding. In *COLT*, 2015.

J. Baik, G. Ben Arous, and S. Péché. Phase transition of the largest eigenvalue for nonnull complex sample covariance matrices. *The Annals of Probability*, 33(5):1643–1697, 2005.

N. Boumal. Nonconvex phase synchronization. *SIAM Journal on Optimization*, 26(4): 2355–2377, 2016.

E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

Y. Cherapanamjeri, P. Jain, and P. Netrapalli. Thresholding based outlier robust PCA. In *COLT*, pages 593–628, 2017.

F. H. Clarke. *Optimization and nonsmooth analysis*, volume 5. SIAM, 1990.

K. L. Clarkson and D. P. Woodruff. Input sparsity and hardness for robust subspace approximation. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 310–329. IEEE, 2015.

M. Coudron and G. Lerman. On the sample complexity of robust PCA. In *NIPS*, pages 3221–3229. 2012.

S. Dasgupta and A. Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems*, pages 2933–2941, 2014.

C. Davis and W. M. Kahan. The rotation of eigenvectors by a perturbation. iii. *SIAM J. on Numerical Analysis*, 7:1–46, 1970.

C. Ding, D. Zhou, X. He, and H. Zha. R1-PCA: rotational invariant L_1-norm principal component analysis for robust subspace factorization. In *ICML*, pages 281–288. ACM, 2006.

A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, 20(2):303–353 (electronic), 1999. ISSN 0895-4798.

R. Ge, F. Huang, C. Jin, and Y. Yuan. Escaping from saddle pointsonline stochastic gradient for tensor decomposition. In *COLT*, pages 797–842, 2015.

R. Ge, J. D. Lee, and T. Ma. Matrix completion has no spurious local minimum. In *NIPS*, pages 2973–2981, 2016.

J. Goes, T. Zhang, R. Arora, and G. Lerman. Robust stochastic principal component analysis. *JMLR W&CP*, page 266274, 2014.

M. Hardt. Understanding alternating minimization for matrix completion. In *FOCS*, pages 651–660. IEEE, 2014.

M. Hardt and A. Moitra. Algorithms and hardness for robust subspace recovery. In *COLT*, pages 354–375, 2013.

P. Jain, A. Tewari, and P. Kar. On iterative hard thresholding methods for high-dimensional m-estimation. In *NIPS*, pages 685–693, 2014.

I. M. Johnstone. On the distribution of the largest eigenvalue in principal components analysis. *Annals of statistics*, pages 295–327, 2001.

I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer, 2nd edition, 2002.

Y. Ledyaev and Q. Zhu. Nonsmooth analysis on smooth manifolds. *Transactions of the American Mathematical Society*, 359(8):3687–3732, 2007.

J. D. Lee, M. Simchowitz, M. I. Jordan, and B. Recht. Gradient descent only converges to minimizers. In *COLT*, pages 1246–1257, 2016.

G. Lerman and T. Maunu. Fast, robust and non-convex subspace recovery. *Information and Inference: A Journal of the IMA*, 7(2):277–336, 2018a.

G. Lerman and T. Maunu. An overview of robust subspace recovery. *Proceedings of the IEEE*, 106(8):1380–1410, Aug 2018b. ISSN 0018-9219. doi: 10.1109/JPROC.2018.2853141.

G. Lerman and T. Zhang. Robust recovery of multiple subspaces by geometric l_p minimization. *Ann. Statist.*, 39(5):2686–2715, 2011.

G. Lerman and T. Zhang. l_p-recovery of the most significant subspace among multiple subspaces with outliers. *Constructive Approximation*, 40(3):329–385, 2014.

G. Lerman, M. B. McCoy, J. A. Tropp, and T. Zhang. Robust computation of linear models by convex relaxation. *Foundations of Computational Mathematics*, 15(2):363–410, 2015.

L. Lim, K. S. Wong, and K. Ye. Statistical estimation and the affine Grassmannian. *arXiv preprint arXiv:1607.01833*, 2016.

L. Lim, K. S. Wong, and K. Ye. The Grassmannian of affine subspaces. *arXiv preprint arXiv:1807.10883*, 2018.

N. Locantore, J. S. Marron, D. G. Simpson, N. Tripoli, J. T. Zhang, and K. L. Cohen. Robust principal component analysis for functional data. *Test*, 8(1):1–73, 1999.

L.-Z. Lu and C. E. M. Pearce. Some new bounds for singular values and eigenvalues of matrix products. *Annals of Operations Research*, 98(1-4):141–148, 2000.

C. Ma, K. Wang, Y. Chi, and Y. Chen. Implicit regularization in nonconvex statistical estimation: Gradient descent converges linearly for phase retrieval and matrix completion. In *PMLR*, volume 80, pages 3345–3354, 10–15 Jul 2018.

R. A. Maronna. Principal components and orthogonal regression based on robust scales. *Technometrics*, 47:264–273, 2005. ISSN 1537-2723.

R. A. Maronna, R. D. Martin, and V. J. Yohai. *Robust statistics: Theory and methods*. Wiley Series in Probability and Statistics. John Wiley & Sons Ltd., Chichester, 2006. ISBN 978-0-470-01092-1; 0-470-01092-4.

T. Maunu and G. Lerman. Robust subspace recovery with adverserial outliers. In preparation.

M. McCoy and J. A Tropp. Two proposals for robust PCA using semidefinite programming. *Electronic Journal of Statistics*, 5:1123–1160, 2011.

S. Mei, Y. Bai, and A. Montanari. The landscape of empirical risk for nonconvex losses. *The Annals of Statistics*, 46(6A):2747–2774, 2018.

S. Minsker. Geometric median and robust estimation in banach spaces. *Bernoulli*, 21(4): 2308–2335, 2015.

M. Mitzenmacher and E. Upfal. *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge university press, 2005.

P. Netrapalli, U. N. Niranjan, S. Sanghavi, A. Anandkumar, and P. Jain. Non-convex robust PCA. In *NIPS*, pages 1107–1115, 2014.

M. R. Osborne and G. A. Watson. An analysis of the total approximation problem in separable norms, and an algorithm for the total l_1 problem. *SIAM journal on scientific and statistical computing*, 6(2):410–424, 1985.

M. Rahmani and G. K. Atia. Coherence pursuit: Fast, simple, and robust principal component analysis. *IEEE Transactions on Signal Processing*, 65(23):6260–6275, 2016.

M. Rudelson and R. Vershynin. The Littlewood–Offord problem and invertibility of random matrices. *Advances in Mathematics*, 218(2):600 – 633, 2008. ISSN 0001-8708.

T. Söderström. *Perturbation results for singular values*. Institutionen för informationsteknologi, Uppsala universitet, 1999.

M. Soltanolkotabi and E. J. Candès. A geometric analysis of subspace clustering with outliers. *Ann. Stat.*, 40(4):2195–2238, 2012. doi: 10.1214/12-AOS1034.

B. St. Thomas, L. Lin, L. Lim, and S. Mukherjee. Learning subspaces of different dimension. *arXiv preprint arXiv:1404.6841*, 2014.

J. Sun, Q. Qu, and J. Wright. Complete dictionary recovery over the sphere. In *SAMPTA*, pages 407–410, May 2015a.

J. Sun, Q. Qu, and J. Wright. When are nonconvex problems not scary? *arXiv preprint arXiv:1510.06096*, 2015b.

S. J. Szarek. The finite-dimensional basis problem with an appendix on nets of Grassmann manifolds. *Acta Math.*, 151(3-4):153–179, 1983. ISSN 0001-5962.

R. Vershynin. How close is the sample covariance matrix to the actual covariance matrix? *Journal of Theoretical Probability*, 25(3):655–686, 2012a.

R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. In *Compressed sensing*, pages 210–268. Cambridge Univ. Press, Cambridge, 2012b.

V. Q. Vu, J. Cho, J. Lei, and K. Rohe. Fantope projection and selection: A near-optimal convex relaxation of sparse PCA. In *NIPS*, pages 2670–2678, 2013.

G. A. Watson. *Some Problems in Orthogonal Distance and Non-Orthogonal Distance Regression*. Defense Technical Information Center, 2001. URL `http://books.google.com/books?id=WKKWGwAACAAJ`.

H. Xu, C. Caramanis, and S. Sanghavi. Robust PCA via outlier pursuit. *IEEE Trans. Information Theory*, 58(5):3047–3064, 2012.

H. Xu, C. Caramanis, and S. Mannor. Outlier-robust PCA: the high-dimensional case. *IEEE Trans. Information Theory*, 59(1):546–572, 2013.

K. Ye and L. Lim. Schubert varieties and distances between subspaces of different dimensions. *SIAM Journal on Matrix Analysis and Applications*, 37(3):1176–1197, 2016.

X. Yi, D. Park, Y. Chen, and C. Caramanis. Fast algorithms for robust PCA via gradient descent. In *NIPS*, pages 4152–4160, 2016.

D. Zhang and L. Balzano. Global convergence of a Grassmannian gradient descent algorithm for subspace estimation. In *AISTATS*, pages 1460–1468, 2016.

T. Zhang. Robust subspace recovery by Tyler's M-estimator. *Information and Inference*, 5 (1):1–21, 2016.

T. Zhang and G. Lerman. A novel M-estimator for robust PCA. *Journal of Machine Learning Research*, 15(1):749–808, 2014.

T. Zhang and Y. Yang. Robust principal component analysis by manifold optimization. *arXiv preprint arXiv:1708.00257*, 2017.

T. Zhang, A. Szlam, and G. Lerman. Median K-flats for hybrid linear modeling with many outliers. In *International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 234–241, Kyoto, Japan, 2009.

Z. Zhou, X. Li, J. Wright, E. Candès, and Y. Ma. Stable principal component pursuit. In *International Symposium on Information Theory Proceedings (ISIT)*, pages 1518–1522. IEEE, 2010.

Journal of Machine Learning Research 20 (2019) 1-27 Submitted 07/17; Revised 01/18; Published 02/18

Approximation Hardness for A Class of Sparse Optimization Problems

Yichen Chen YICHENC@PRINCETON.EDU
Department of Computer Science, Princeton University
Princeton, NJ 08544, USA

Yinyu Ye YINYU-YE@STANFORD.EDU
Department of Management Science and Engineering, Stanford University
Stanford, CA 94304, USA

Mengdi Wang MENGDIW@PRINCETON.EDU
Department of Operations Research and Financial Engineering, Princeton University
Princeton, NJ 08544, USA

Editor: David Wipf

Abstract

In this paper, we consider three typical optimization problems with a convex loss function and a nonconvex sparse penalty or constraint. For the sparse penalized problem, we prove that finding an $\mathcal{O}(n^{c_1}d^{c_2})$-optimal solution to an $n \times d$ problem is strongly NP-hard for any $c_1, c_2 \in [0, 1)$ such that $c_1 + c_2 < 1$. For two constrained versions of the sparse optimization problem, we show that it is intractable to approximately compute a solution path associated with increasing values of some tuning parameter. The hardness results apply to a broad class of loss functions and sparse penalties. They suggest that one cannot even approximately solve these three problems in polynomial time, unless P = NP.

Keywords: nonconvex optimization, computational complexity, variable selection, NP-hardness, sparsity

1. Introduction

Sparsity is a prominent modeling tool for extracting useful information from high-dimensional data. A practical goal is to minimize the empirical loss using as few variables/features as possible. In this paper, we consider three typical optimization problems arising from sparse machine learning. The first problem takes the form of empirical risk minimization with an additive sparse penalty.

Problem 1 Given the loss function $\ell : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}^+$, penalty function $p : \mathbb{R} \mapsto \mathbb{R}^+$, and regularization parameter $\lambda > 0$, consider the problem

$$\min_{x \in \mathbb{R}^d} \sum_{i=1}^{n} \ell\left(a_i^T x, b_i\right) + \lambda \sum_{j=1}^{d} p\left(|x_j|\right),$$

where $A = (a_1, \ldots, a_n)^T \in \mathbb{R}^{n \times d}$, $b = (b_1, \ldots, b_n)^T \in \mathbb{R}^n$ are input data.

©2019 Yichen Chen, Yinyu Ye, and Mengdi Wang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-373.html.

We also consider two constrained versions of sparse optimization, which are given by Problems 2 and 3. Such problems arise from sparse estimation (Shen et al., 2012) and sparse recovery (Natarajan, 1995; Bruckstein et al., 2009).

Problem 2 Given the loss function $\ell : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}^+$, penalty function $p : \mathbb{R} \mapsto \mathbb{R}^+$, consider the problem

$$\min_{x \in \mathbb{R}^d} \sum_{i=1}^{n} \ell\left(a_i^T x, b_i\right) \quad \text{s.t.} \sum_{j=1}^{d} p\left(|x_j|\right) \leq K,$$

where $A = (a_1, \ldots, a_n)^T \in \mathbb{R}^{n \times d}$, $b = (b_1, \ldots, b_n)^T \in \mathbb{R}^n$ and the sparsity parameter K are input data.

Problem 3 Given the loss function $\ell : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}^+$, penalty function $p : \mathbb{R} \mapsto \mathbb{R}^+$, consider the problem

$$\min_{x \in \mathbb{R}^d} \sum_{j=1}^{d} p\left(|x_j|\right) \quad \text{s.t.} \sum_{i=1}^{n} \ell\left(a_i^T x, b_i\right) \leq \eta,$$

where $A = (a_1, \ldots, a_n)^T \in \mathbb{R}^{n \times d}$, $b = (b_1, \ldots, b_n)^T \in \mathbb{R}^n$ and the error tolerance parameter $\eta \geq 0$ are input data.

For a given sparsity level K, the optimal solution to Problem 2 is the best K-sparse solution that fits the data set (A, b). To select the best sparsity level that fits the data, one usually needs to solve a sequence of instances of Problem 2, corresponding to different values of K. Similarly for Problem 3, one often needs to compute the solution path that is associated with a sequence of values of η.

We are interested in the computational complexity of Problems 1, 2 and 3 under general conditions of the loss function ℓ and the sparse penalty p. In particular, we focus on the case where ℓ is a convex loss function and p is a nonconvex function with a unique minimizer at 0. These problems naturally arise from feature selection, compressive sensing, and sparse approximation. For some special cases of Problem 1, it has been shown that finding an *exact solution* is strongly NP-hard (Huo and Chen, 2010; Chen et al., 2014). However, these results have not excluded the possibility of the existence of polynomial-time algorithms with small approximation error. The technical note by Chen and Wang (2016) established the hardness of approximately solving Problems 1, 2 for the special case where p is the L_0 norm.

In this paper, we prove that it is strongly NP-hard to approximately solve Problems 1, 2 and 3 within certain levels of suboptimality. For Problem 1, we show that there exists a worst-case lower bound on the suboptimality error that can be achieved by any tractable deterministic algorithm. For Problems 2 and 3, we show that there does not exist any pseudo polynomial-time algorithm that can approximately compute a solution path where K or η increases at a certain speed. Our results apply to a variety of optimization problems in estimation and machine learning. Examples include sparse classification, sparse logistic regression and many more. The strong NP-hardness of approximation is one of the strongest forms of complexity result for continuous optimization. To our best knowledge, this is the first work that gives the proof of the approximation hardness for sparse optimization under general conditions on ℓ and p. A preliminary conference version of this paper (Chen et al.,

2017) focused solely on Problem 1. The current journal version extends the analysis to sparse constrained optimization and establishes hardness results for Problems 2 and 3.

Our results on optimization complexity provide new insights into the complexity of sparse feature selection (Zhang et al., 2014; Foster et al., 2015). In the case of sparse regression for linear models, our result on Problem 1 shows that the lower bound of approximation error is significantly larger than the desired small statistical error (although our lower bound is worst-case). In the case where practitioners wish to choose the best sparsity level, our result on Problem 2 shows that it is impossible to know how much the loss function would improve when increasing the sparsity level. These observations provide strong evidences for the hardness of variable selection.

Our main contributions are four-folded.

1. We prove the strong NP-hardness for Problems 1, 2 and 3 with a general loss function $\ell(\cdot)$, which are no longer limited to L_2 or L_p functions. These are the first results that apply to the broad class of problems including but not limited to: least square regression, linear model with Laplacian noise, robust regression, Poisson regression, logistic regression, inverse Gaussian models and the generalized linear model under the exponential distributions.

2. We present a general condition on the penalty function $p(\cdot)$ such that Problems 1, 2 and 3 are strongly NP-hard. Our condition is a slightly weaker version of strict concavity. It only requires the penalty function be concave while ruling out the possibility of linear penalty function (i.e., the LASSO) which is concave but also convex. It is satisfied by typical penalty functions such as the L_p norm ($p \in [0, 1)$), clipped L_1 norm, smoothly clipped absolute deviation, etc. To the best of our knowledge, this is the most general condition on the penalty function in the literature.

3. We prove that finding an $\mathcal{O}\left(\lambda n^{c_1} d^{c_2}\right)$-optimal solution to Problem 1 is strongly NP-hard, for any $c_1, c_2 \in [0, 1)$ such that $c_1 + c_2 < 1$. Here the $\mathcal{O}(\cdot)$ hides parameters that depend on the penalty function p, which is to be specified later. Our proof provides a first unified analysis that deals with a broad class of problems taking the form of Problem 1.

4. We prove that it is strongly NP-hard to distinguish the optimal values of instances of Problem 2 (or Problem 3) that are associated with increasing values of the sparsity parameter K (or the error tolerance parameter η). This is the first hardness result for sparsity-constrained optimization with general loss and penalty functions. It implies that it is hard to approximately compute a solution path for the purpose of parameter tuning.

Section 2 reviews the background of sparse optimization and related literatures in the complexity theory. Section 3 presents the key assumptions and illustrates examples of loss and penalty functions that satisfy the assumptions. Section 4 gives the main results. Section 5 discusses the implications of our hardness results. The full proofs are given in Section 6.

2. Background and Related Works

Sparse optimization problems are common in machine learning, estimation, and signal processing. The sparse penalty p plays the important role of variable/feature selection. A common technique of imposing sparsity is to penalize the objective with a penalty function, leading to Problem 1. A well known example is the LASSO where p is the L_1 norm penalty and ℓ is the regression objective (Tibshirani, 1996). Nonconvex choices of p have been extensively studied in order to provide stronger statistical guarantee to the optimal soluion. Fan and Li (2001) proposed the smoothly clipped absolute deviation (SCAD) penalty which forces the solution of Problem 1 to be unbiased, sparse and stable in certain statistical sense. Frank and Friedman (1993) proposed the bridge estimator which use the L_p norm ($0 < p < 1$) as its penalty function. Other related works include exact reconstruction of sparse signals by Candes et al. (2008) and Chartrand (2007), high-dimensional variable selection by Fan and Lv (2010), sparse Ising model by Xue et al. (2012) and regularized M-estimators by Loh and Wainwright (2013), etc.

Problem 2 finds applications in sparse estimation and feature selection. The work by Shen et al. (2012) proposed a statistically optimal estimator as the solution to the maximimum likelihood problem with L_0 sparsity constraint

$$\min_{x \in \mathbb{R}^d} -\sum_{i=1}^{n} \log g(x; a_i, b_i) \quad \text{s.t.} \sum_{i=1}^{d} \|x_i\|_0 \leq K.$$

This is a special case of Problem 2. The work by Fang et al. (2015) proposed an L_0 constrained optimization problem for sparse estimation of large-scale graphical models, which is also a special case of Problem 2. Another related problem is sparse recovery, which is to find the sparsest solution to a system of equations within an error tolerance. For example, Natarajan (1995) considered the problem

$$\min_{x \in \mathbb{R}^d} \sum_{i=1}^{d} \|x_i\|_0 \quad \text{s.t.} \sum_{i=1}^{n} (a_i^T x - b_i)^2 \leq \delta,$$

which is a special case of Problem 3. See the work by Bruckstein et al. (2009) for more examples of Problem 3.

Within the mathematical programming community, the complexity of Problem 1 has been considered in a few works. Huo and Chen (2010) first proved the hardness result for problems with a relaxed family of penalty functions $\min_{x \in \mathbb{R}^d} \|Ax - b\|_2^2 + \lambda \sum_{i=1}^{d} p(|x_i|)$. They show that for the penalties in ℓ_0, hard-thresholded (Antoniadis and Fan, 2001) and SCAD (Fan and Li, 2001), the above optimization problem is NP-hard. Our result (Theorem 1) requires weaker conditions on $p(\cdot)$ than theirs. In particular, our results applies to the $L_p (0 < p < 1)$ penalization and the clipped L_1 penalty function specified in Section 3.1 which do not satisfy the conditions in their paper. Moreover, our result applies to a broad class of ℓ functions and obtains *strong NP-hardness*. A problem is *strongly NP-hard* if every problem in NP can be polynomially reduced to it in a way such that input in the reduced instance are written in unary (Vazirani, 2001). It is a stronger notion than NP-hardness where NP-hard problems might still be fast to solve in practice using pseudo-polynomial algorithms if the coding size is small (Garey and Johnson, 1978). On the contrary, a

strongly NP-hard problem doesn't have such a pseudo-polynomial algorithm. Chen et al. (2014) showed that the L_2-L_p minimization $\min_{x\in\mathbb{R}^d} \|Ax-b\|_2^2 + \lambda\sum_{i=1}^d |x_i|^p$ is strongly NP-hard when $p \in (0,1)$. At the same time, Bian and Chen (2014) proved the strong NP-hardness for another class of penalty functions. Their result requires $p(t)$ to be at least locally strictly concave, while ours does not. In particular, among the examples listed in Section 3.1, their results do not apply to L_0 penalization and clipped L_1 penalty function. To the best of our knowledge, our results are the most general ones up to today, which contains as special cases a broad class of penalty functions including ℓ_0, hard-thresholded, SCAD, L_p penalization ($p \in (0,1)$), folded concave penalty family (Fan et al., 2014) etc.

Within the theoretical computer science community, there have been several early works on the complexity of sparse recovery, beginning with the work by Arora et al. (1993). Amaldi and Kann (1998) proved that the problem $\min\{\|x\|_0 \mid Ax = b\}$ is not approximable within a factor $2^{\log^{1-\epsilon} d}$ for any $\epsilon > 0$. Natarajan (1995) showed that, given $\epsilon > 0, A$ and b, the problem $\min\{\|x\|_0 \mid \|Ax-b\|_2 \le \epsilon\}$ is NP-hard. Davis et al. (1997) proved a similar result that for some given $\epsilon > 0$ and $M > 0$, it is NP-complete to find a solution x such that $\|x\|_0 \le M$ and $\|Ax-b\| \le \epsilon$. More recently, Foster et al. (2015) studied sparse linear recovery and sparse linear regression with *subgaussian noises*. Assuming that the true solution is K-sparse, it showed that no polynomial-time (randomized) algorithm can find a $K \cdot 2^{\log^{1-\delta} d}$-sparse solution x with $\|Ax-b\|_2^2 \le d^{C_1} n^{1-C_2}$ *with high probability*, where δ, C_1, C_2 are arbitrary positive scalars. Another work (Zhang et al., 2014) showed that under the Gaussian linear model, there exists a gap between the mean square loss that can be achieved by polynomial-time algorithms and the statistically optimal mean squared error. These two works focus on the estimation of linear models and impose distributional assumptions regarding the input data. For comparison with our results, theirs are stronger in the sense that they exclude the existence of any tractable randomized algorithm that succeds with high probability, while ours apply to deterministic algorithms. In the mean time, their results are less general than ours in the sense that they assume specific data distributions and specific loss functions, while ours are concerned with a much more general setting. In short, existing complexity results on sparse recovery are different in nature with our results on sparse computational optimization.

There remain several open questions. First, existing results do not apply to general loss functions ℓ or sparse penalties p. Existing analyses rely on specific properties of the L_q loss functions, such as the linear shift property $\|ax\|^q = a^q\|x\|^q$ and the property that L_q has sufficiently large second-order derivative around its minimum. However, these nice properties are lost in a majority of estimation problems, such as logistic regression and poisson regression. Second, the existing results from mathematical programming community apply only to the unconstrained Problem 1. The computational complexity of Problems 2 and 3 remain under-investigated in the community of optimization. Third, the results from computer science community apply to Problem 2 when the penalty function is L_0. These results work for specific loss functions and some of them impose distributional assumption about the input (Foster et al., 2015). In this paper, we focus on the worst-case complexity without making any distributional assumption regarding the input data. In this setting, the complexity of Problems 2 and 3 with penalty functions other than L_0 is yet to be established.

3. Assumptions

In this section, we state the two critical assumptions that lead to the strong NP-hardness results: one for the penalty function p, the other one for the loss function ℓ. We argue that these assumptions are essential and very general. They apply to a broad class of loss functions and penalty functions that are commonly used.

3.1. Assumption On The Sparse Penalty

Throughout this paper, we make the following assumption regarding the sparse penalty function $p(\cdot)$.

Assumption 1 *The penalty function $p(\cdot)$ satisfies the following conditions:*

(i) (Monotonicity) $p(\cdot)$ is non-decreasing on $[0,+\infty)$.

(ii) (Concavity) There exists $\tau > 0$ such that $p(\cdot)$ is concave but not linear on $[0,\tau]$.

In words, condition (ii) means that the concave penalty $p(\cdot)$ is nonlinear. Assumption 1 is the most general condition on penalty functions in the existing literature of sparse optimization. Below we present a few such examples.

1. In variable selection problems, the L_0 penalization $p(t) = I_{\{t\neq 0\}}$ arises naturally as a penalty for the number of factors selected.

2. A natural generalization of the L_0 penalization is the L_p penalization $p(t) = t^p$ where $(0 < p < 1)$. The corresponding minimization problem is called the bridge regression problem (Frank and Friedman, 1993).

3. To obtain a hard-thresholding estimator, Antoniadis and Fan (2001) use the penalty functions $p_\gamma(t) = \gamma^2 - ((\gamma - t)^+)^2$ with $\gamma > 0$, where $(x)^+ := \max\{x, 0\}$ denotes the positive part of x.

4. Any penalty function that belongs to the folded concave penalty family (Fan et al., 2014) satisfies the conditions in Assumption 1. Examples include the SCAD (Fan and Li, 2001) and the MCP (Zhang, 2010a), whose derivatives on $(0,+\infty)$ are $p'_\gamma(t) = \gamma I_{\{t\le\gamma\}} + \frac{(a\gamma-t)^+}{a-1} I_{\{t>\gamma\}}$ and $p'_\gamma(t) = (\gamma - \frac{t}{b})^+$, respectively, where $\gamma > 0$, $a > 2$ and $b > 1$.

5. The conditions in Assumption 1 are also satisfied by the clipped L_1 penalty function (Antoniadis and Fan, 2001; Zhang, 2010b) $p_\gamma(t) = \gamma \cdot \min(t, \gamma)$ with $\gamma > 0$. This is a special case of the piecewise linear penalty function:

$$p(t) = \begin{cases} k_1 t & \text{if } 0 \le t \le a \\ k_2 t + (k_1 - k_2)a & \text{if } t > a \end{cases}$$

where $0 \le k_2 < k_1$ and $a > 0$.

6. Another family of penalty functions which bridges the L_0 and L_1 penalties are the fraction penalty functions $p_\gamma(t) = \dfrac{(\gamma+1)t}{\gamma + t}$ with $\gamma > 0$ (Lv and Fan, 2009).

7. The family of log-penalty functions:

$$p_\gamma(t) = \frac{1}{\log(1+\gamma)} \log(1+\gamma t)$$

with $\gamma > 0$, also bridges the L_0 and L_1 penalties (Candes et al., 2008).

3.2. Assumption On The Loss Function

We state our assumption about the loss function ℓ.

Assumption 2 *Let M be an arbitrary constant. For any interval $[\tau_1, \tau_2]$ where $0 < \tau_1 < \tau_2 < M$, there exists $k \in \mathbb{Z}^+$ and $b \in \mathbb{Q}^k$ such that $h(y) := \sum_{i=1}^k \ell(y, b_i)$ has the following properties:*

(i) $h(y)$ is convex and Lipschitz continuous on $[\tau_1, \tau_2]$.

(ii) $h(y)$ has a unique minimizer y^* in (τ_1, τ_2).

(iii) There exists $N \in \mathbb{Z}^+, \bar{\delta} \in \mathbb{Q}^+$ and $C \in \mathbb{Q}^+$ such that when $\delta \in (0, \bar{\delta})$, we have

$$\frac{h(y^* \pm \delta) - h(y^*)}{\delta^N} \geq C.$$

(iv) $h(y^*)$, $\{b_i\}_{i=1}^k$ can be represented in $\mathcal{O}(\log \frac{1}{\tau_2 - \tau_1})$ bits.

Assumption 2 is a critical, but very general, assumption regarding the loss function $\ell(y, b)$. Condition (i) requires convexity and Lipschitz continuity within a neighborhood. Conditions (ii), (iii) essentially require that, given an interval $[\tau_1, \tau_2]$, one can artificially pick $b_1, \ldots, b_k$ to construct a function $h(y) := \sum_{i=1}^k \ell(y, b_i)$ such that h has its unique minimizer in $[\tau_1, \tau_2]$ and has enough curvature near the minimizer. This property ensures that a bound on the minimal value of $h(y)$ can be translated to a meaningful bound on the distance to the minimizer y^*. The conditions (i), (ii), (iii) are typical properties that a loss function usually satisfies. Condition (iv) is a technical condition that is used to avoid dealing with infinitely-long irrational numbers. It can be easily verified for almost all common loss functions.

We will show that Assumptions 2 is satisfied by a variety of loss functions. An (incomplete) list is given below.

1. In the least squares regression, the loss function has the form

$$\sum_{i=1}^n \left(a_i^T x - b_i\right)^2.$$

Using our notation, the corresponding loss function is $\ell(y, b) = (y - b)^2$. For all τ_1, τ_2, we choose an arbitrary $b' \in [\tau_1, \tau_2]$. We can verify that $h(y) = \ell(y, b')$ satisfies all the conditions in Assumption 2.

2. In the linear model with Laplacian noise, the negative log-likelihood function is

$$\sum_{i=1}^n \left|a_i^T x - b_i\right|.$$

So the loss function is $\ell(y, b) = |y - b|$. As in the case of least squares regression, the loss function satisfy Assumption 2. Similar argument also holds when we consider the L_q loss $|\cdot|^q$ with $q \geq 1$.

3. In robust regression, we consider the Huber loss (Huber, 1964) which is a mixture of L_1 and L_2 norms. The loss function takes the form

$$L_\delta(y, b) = \begin{cases} \frac{1}{2}|y - b|^2 & \text{for } |y - b| \leq \delta, \\ \delta(|y - b| - \frac{1}{2}\delta) & \text{otherwise.} \end{cases}$$

for some $\delta > 0$ where $y = a^T x$. We then verify that Assumption 2 is satisfied. For any interval $[\tau_1, \tau_2]$, we pick an arbitrary $b \in [\tau_1, \tau_2]$ and let $h(y) = \ell(y, b)$. We can see that $h(y)$ satisfies all the conditions in Assumption 2.

4. In Poisson regression (Cameron and Trivedi, 2013), the negative log-likelihood minimization is

$$\min_{x \in \mathbb{R}^d} -\log L(x; A, b) = \min_{x \in \mathbb{R}^d} \sum_{i=1}^n (\exp(a_i^T x) - b_i \cdot a_i^T x).$$

We now show that $\ell(y, b) = e^y - b \cdot y$ satisfies Assumption 2. For any interval $[\tau_1, \tau_2]$, we choose q and r such that $q/r \in [e^{\tau_1}, e^{\tau_2}]$. Note that $e^{\tau_2} - e^{\tau_1} = e^{\tau_1 + \tau_2 - \tau_1} - e^{\tau_1} \geq \tau_2 - \tau_1$. Also, e^{τ_2} is bounded by e^M. Thus, q, r can be chosen to be polynomial in $\lceil 1/(\tau_2 - \tau_1) \rceil$ by letting $r = \lceil 1/(\tau_2 - \tau_1) \rceil$ and q be some number less than $r \cdot e^M$. Then, we choose $k = r$ and $b \in \mathbb{Z}^k$ such that $h(y) = \sum_{i=1}^k \ell(y, b_i) = r \cdot e^y - q \cdot y$. Let us verify Assumption 2. (i), (iv) are straightforward by our construction. For (ii), note that $h(y)$ take its minimum at $\ln(q/r)$ which is inside $[\tau_1, \tau_2]$ by our construction. To verify (iii), consider the second order Taylor expansion of $h(y)$ at $\ln(q/r)$,

$$h(y + \delta) - h(y) = \frac{r \cdot e^y}{2} \cdot \delta^2 + o(\delta^2) \geq \frac{\delta^2}{2} + o(\delta^2),$$

We can see that (iii) is satisfied. Therefore, Assumption 2 is satisfied.

5. In logistic regression, the negative log-likelihood function minimization is

$$\min_{x \in \mathbb{R}^d} \sum_{i=1}^n \log(1 + \exp(a_i^T x)) - \sum_{i=1}^n b_i \cdot a_i^T x.$$

We claim that the loss function $\ell(y, b) = \log(1 + \exp(y)) - b \cdot y$ satisfies Assumption 2. By a similar argument as the one in Poisson regression, we can verify that $h(y) = \sum_{i=1}^r \ell(y, b_i) = r \log(1 + \exp(y)) - qy$ where $q/r \in [\frac{e^{\tau_1}}{1+e^{\tau_1}}, \frac{e^{\tau_2}}{1+e^{\tau_2}}]$ and q, r are polynomial in $\lceil 1/(\tau_2 - \tau_1) \rceil$ satisfies all the conditions in Assumption 2. For (ii), observe that $\ell(y, b)$

take its minimum at $y = \ln \frac{q/r}{1-q/r}$. To verify (iii), we consider the second order Taylor expansion at $y = \ln \frac{q/r}{1-q/r}$, which is

$$h(y+\delta) - h(y) = \frac{q}{2(1+e^y)}\delta^2 + o(\delta^2)$$

where $y \in [\tau_1, \tau_2]$. Note that e^y is bounded by e^M, which can be computed beforehand. As a result, (iii) holds as well.

6. In the mean estimation of inverse Gaussian models (McCullagh, 1984), the negative log-likelihood function minimization is

$$\min_{x\in\mathbb{R}^d} \sum_{i=1}^n \frac{(b_i \cdot \sqrt{a_i^T x} - 1)^2}{b_i}.$$

Now we show that the loss function $\ell(y,b) = \frac{(b\cdot\sqrt{y}-1)^2}{b}$ satisfies Assumption 2. By setting the derivative to be zero with regard to y, we can see that y take its minimum at $y = 1/b^2$. Thus for any $[\tau_1, \tau_2]$, we choose $b' = q/r \in [1/\sqrt{\tau_2}, 1/\sqrt{\tau_1}]$. We can see that $h(y) = \ell(y, b')$ satisfies all the conditions in Assumption 2.

7. In the estimation of generalized linear model under the exponential distribution (McCullagh, 1984), the negative log-likelihood function minimization is

$$\min_{x\in\mathbb{R}^d} -\log L(x; A, b) = \min_{x\in\mathbb{R}^d} \frac{b_i}{a_i^T x} + \log(a_i^T x).$$

By setting the derivative to 0 with regard to y, we can see that $\ell(y,b) = \frac{b}{y} + \log y$ has a unique minimizer at $y = b$. Thus by choosing $b' \in [\tau_1, \tau_2]$ appropriately, we can readily show that $h(y) = \ell(y, b')$ satisfies all the conditions in Assumption 2.

To sum up, the combination of *any* loss function given in Section 3.1 and *any* penalty function given in Section 3.2 results in a strongly NP-hard sparse optimization problem. We will provide formal statements and proof of these results in subsequent sections.

4. Main Results

In this paper, we aim to clarify the complexity for a broader class of sparse optimization problems taking the form of Problems 1, 2 and 3. Given an optimization problem $\min_{x\in X} f(x)$, we say that a solution $\bar{x}$ is ϵ-optimal if $\bar{x} \in X$ and $f(\bar{x}) \le \inf_{x\in X} f(x) + \epsilon$.

Theorem 1 (Strong NP-Hardness of Problem 1) *Let Assumptions 1 and 2 hold, and let $c_1, c_2 \in [0,1)$ be arbitrary such that $c_1 + c_2 < 1$. Then it is strongly NP-hard to find a $\lambda \cdot \kappa \cdot n^{c_1} d^{c_2}$-optimal solution of Problem 1, where d is the dimension of variable space and $\kappa = \min_{t\in[\tau/2,\tau]}\{\frac{2p(t/2)-p(t)}{t}\}$.*

The non-approximable error in Theorem 1 involves the constant κ which is determined by the sparse penalty function p. In the case where p is the L_0 norm function, we can take $\kappa = 1$. In the case of piecewise linear L_1 penalty, we have $\kappa = (k_1 - k_2)/4$. In the case of SCAD penalty, we have $\kappa = \Theta(\gamma^2)$.

According to Theorem 1, the non-approximable error $\lambda \cdot \kappa \cdot n^{c_1} d^{c_2}$ is determined by three factors: (i) properties of the regularization penalty $\lambda \cdot \kappa$; (ii) data size n; and (iii) dimension or number of variables d. This result illustrates a fundamental gap that can not be closed by any polynomial-time deterministic algorithm. This gap scales up when either the data size or the number of variables increases. In Section 5.1, we will see that this gap is substantially larger than the desired estimation precision in a special case of sparse linear regression.

Next we study the complexity of the sparsity-constrained Problem 2. We denote by $\ell_n(x)$ the normalized loss function:

$$\ell_n(x) = \frac{1}{n}\sum_{i=1}^n \ell\left(a_i^T x, b_i\right),$$

and denote by x_K^* the best K-sparse solution:

$$x_K^* \in \operatorname{argmin}\left\{ \ell_n(x) \,\middle|\, \sum_{j=1}^d p(|x_j|) \le K \right\}.$$

We obtain the following result.

Theorem 2 (Strong NP-Hardness of Problem 2) *Let Assumptions 1 and 2 hold, and let $c_1, c_2 \in [0,1)$ be arbitrary such that $c_1 + c_2 < 1$. Let $\hat{x}_K$ be the approximate solution to Problem 2 with sparsity parameter K. Then there does not exist a pseudo polynomial-time algorithm that takes the input of Problem 2 and outputs a sequence of approximate solutions satisfying*

$$\ell_n(\hat{x}_{K+\kappa n^{c_1} d^{c_2}}) \le \ell_n(x_K^*),$$

for all $K = 0, \kappa n^{c_1} d^{c_2}, 2\kappa n^{c_1} d^{c_2}, \ldots,$ unless P=NP, where $\kappa = \min_{t\in[\tau/2,\tau]}\{\frac{2p(t/2)-p(t)}{t}\}$.

Let us interpret the results of Theorem 2 in a practical setting. Suppose that we want to solve a sequence of sparsity-constrained problems with different values of the sparsity parameter K. The aim is to compare the corresponding empirical losses $\{\ell_n(x_K^*)\}$ and tune the parameter K.

Theorem 2 suggests that we cannot decide whether and how much the objective value will change by increasing the sparsity level from K to $K + \kappa n^{c_1} d^{c_2}$. Even if $\ell_n(x_K^*)$ is known as a benchmark, we can not find a better approximation of $\ell_n(x^*_{K+\kappa n^{c_1} d^{c_2}})$ in polynomial time. In short, Theorem 2 tells us that it is computationally intractable to differentiate the minimal empirical losses that correspond to different values of K, unless P=NP. This implies that tuning the parameter K is computationally intractable.

Our last result concerns the error-constrained Problem 3.

Theorem 3 *Let Assumptions 1 and 2 hold, and let $c \in [0,1)$ be arbitrary. Let $\hat{x}_\eta$ be the approximate solution to Problem 3 with error tolerance η and let x^*_η be the corresponding optimal solution. There does not exist a pseudo polynomial-time algorithm that takes the input of Problem 3 and outputs a sequence of approximate solutions satisfying*

$$\sum_{j=1}^{d} p\left(\left(\hat{x}_{\eta+\kappa n^{c_1} d^{c_2}}\right)_j\right) \leq \sum_{j=1}^{d} p\left(\left(x^*_\eta\right)_j\right),$$

for all $\eta = 0, \kappa n^{c_1} d^{c_2}, 2\kappa n^{c_1} d^{c_2}, \ldots,$ unless P=NP. Here, $(x)_j$ is the j-th component of vector x and $\kappa = \min_{t\in[\tau/2,\tau]}\{\frac{2p(t/2)-p(t)}{t}\}$.

Theorems 1, 2 and 3 are closely related to one another. Recall that the goal of sparse optimization is to make both the loss function and sparsity level small. Theorem 2 and Theorem 3 suggest that it is not possible to approximate the solution path, where either the loss tolerance or the sparsity level varies, in polynomial time. In contrast, Theorem 1 proves the approximation hardness for the sum between the loss tolerance and the sparsity level, when a fixed λ is used.

Theorems 1, 2 and 3 validate the long-lasting belief that optimization involving nonconvex penalty is hard. They provide worst-case lower bounds for the optimization error that can be achieved by any polynomial-time algorithm. This is one of the strongest forms of hardness result for continuous optimization.

5. Implications of The Hardness Results

In this section, we interpret the strong NP-hardness results in the contexts of linear regression with SCAD penalty (which is a special case of Problem 1) and sparsity parameter tuning (which is related to Problem 2). We give a few remarks on the implication of our hardness results.

5.1. Hardness of Regression with SCAD Penalty

Let us try to understand how significant is the non-approximable error of Problem 1. We consider the special case of linear models with SCAD penalty. Let the input data (A, b) be generated by the linear model $A\bar{x} + \varepsilon = b$, where $\bar{x}$ is the unknown *true* sparse coefficients and ε is a zero-mean multivariate subgaussian noise. Given the data size n and variable dimension d, we follow the work by Fan and Li (2001) and obtain a special case of Problem 1, given by

$$\min_x \frac{1}{2}\|Ax - b\|_2^2 + n\sum_{j=1}^{d} p_\gamma(|x_j|), \tag{1}$$

where $\gamma = \sqrt{\log d/n}$. Fan and Li (2001) showed that the optimal solution x^* of problem (1) has a small statistical error, i.e., $\|\bar{x} - x^*\|_2^2 = \mathcal{O}\left(n^{-1/2} + a_n\right)$, where $a_n = \max\{p'_\lambda(|x^*_j|) : x^*_j \neq 0\}$. Fan et al. (2015) further showed that we only need to find a $\sqrt{n\log d}$-optimal solution to (1) to achieve such a small estimation error.

However, Theorem 2 tells us that it is not possible to compute an $\epsilon_{d,n}$-optimal solution for problem (1) in polynomial time, where $\epsilon_{d,n} = \lambda\kappa n^{1/2} d^{1/3}$ (by letting $c_1 = 1/2, c_2 = 1/3$).

In the special case of problem (1), we can verify that $\lambda = n$ and $\kappa = \Omega(\gamma^2) = \Omega(\log d/n)$. As a result, we see that

$$\epsilon_{d,n} = \Omega(n^{1/2} d^{1/3}) \gg \sqrt{n\log d},$$

for high values of the dimension d. According to Theorem 2, it is strongly NP-hard to approximately solve problem (1) within the required statistical precision $\sqrt{n\log d}$, where there is no distributional assumption on the data.

This gap is due to that the positive statistical properties of SCAD rely on strong distributional assumptions, while our hardness result does not. This illustrates a sharp contrast between the desirable statistical properties of sparse optimization under distributional assumptions and the worst-case computational complexity. In short, there does not exist a general-purpose polynomial algorithm.

5.2. Hardness of Tuning the Sparsity Level with L_0 Penalty

Suppose that we are given the input data set (A, b) with d variables/features and n samples. Now we want to find a sparse solution x that approximately minimize the empirical loss $L_n(x) = \frac{1}{n}\sum_{i=1}^{n} \ell(a_i^T x, b_i)$. A practical problem is to find the right sparsity level for the approximate solution. This is essentially a model selection problem.

Finding the sparsity level requires computing the K-sparse solutions

$$x^*_K \in \operatorname{argmin}\left\{L_n(x) \mid \|x\|_0 \leq K\right\},$$

for a range of values of K. This can be translated into solving a sequence of L_0 constrained problems (of the form Problem 2) with K ranging from 1 to d. Regardless of the specific model selection procedure, it is inevitable to compute x^*_K for many values of K's, and to compare their empirical losses such as $L_n(x^*_K)$ and $L_n(x^*_{K+1})$.

Now let us interpret the results of Theorem 3 in the setting of tuning parameter K. Theorem 3 can be translated as follows. There exists some sparsity level K such that: even if the exact K-sparse solution x^*_K is known, the non-approximable optimization error for the $(K+1)$-sparse problem is at least

$$L_n(x^*_K) - L_n(x^*_{K+1}) > 0.$$

The minimal empirical loss using K features is the best possible approximation to the minimal loss using $K+1$ features. In other words, we cannot decide whether and how much the objective value will change by increasing the sparsity level from K to $K+1$. Even if $L_n(x^*_K)$ is known as a benchmark, we can not find a better approximation of $L_n(x^*_{K+1})$ in polynomial-time. In summary, Theorem 2 tells us that it is computationally intractable to differentiate between the sparsity levels K and $K+1$, unless P=NP. This implies that selection of the sparsity level is computationally intractable.

5.3. Remarks on the NP-Hardness Results

As illustrated by the preceding analysis, the non-approximibility of Problems 1, 2 and 3 suggests that computing the sparse estimator and tuning the sparsity parameter are hard *in the worst case*. Although the results seem negative, they should not discourage researchers from studying computational perspectives of sparse optimization. We make the following remarks:

1. Theorems 1, 2 and 3 are *worst-case* complexity results. They suggest that one cannot find a tractable solution to the sparse optimization problems, without making any additional assumption to rule out the worst-case instances. It is possible that the worst-case instances are highly unlikely to occur in practical situations.

2. Our results do not exclude the possibility that, under more stringent modeling and distributional assumptions, the problem would be tractable with high probability or on average.

In short, the sparse optimization Problems 1, 2 and 3 are fundamentally hard from a purely computational perspective. This paper together with the prior related works provide a complete answer to the computational complexity of sparse optimization.

6. Technical Proofs

In this section, we prove the hardness of approximation of Problem 1, 2 and 3 for general loss function ℓ and penalty function p. We develop the reduction proof through a series of preliminary lemmas.

6.1. Preliminary Lemmas

Our first lemma gives us a key fact about the nonconvex penalty function p. We use $B(\theta, \delta)$ to denote the interval $(\theta - \delta, \theta + \delta)$.

Lemma 4 *For any penalty function p that satisfies Assumption 1, we have*

(i) *$p(t)$ is continuous on $(0, \tau]$.*

(ii) *For any $t_1, ..., t_l \geq 0$, if $\sum_{i=1}^n t_i \leq \tau$, then $\sum_{i=1}^l p(t_i) \geq p(\sum_{i=1}^l t_i)$.*

(iii) *There exists $a \in [1/2, 1)$ such that when $\sum_{i=1}^l t_i \in [a\tau, \tau]$, the above inequality holds as equality if and only if $t_i = t^*$ for some i while $t_j = 0$ for $j \neq i$.*

(iv) *Denote $\kappa = \min_{t \in [a\tau, \tau]} \{\frac{2p(t/2) - p(t)}{t}\}$. For the constant a given in (iii), we have that $\forall \delta > 0, t_1, \cdots, t_l \in \mathbb{R}, \forall \epsilon \leq \kappa\delta$: if $\sum_{i=1}^l t_i = t^* \in [a\tau, \tau]$ and $p(\sum_{i=1}^l t_i) + \epsilon \geq \sum_{i=1}^l p(t_i)$, then there is at most one i such that $t_i \notin B(0, \delta)$.*

Proof As (i), (ii) and (iii) are proved by Ge et al. (2015), we prove (iv) here. We first prove the lemma when $t_1, \cdots, t_l \geq 0$. We start by proving the case when $l = 2$. By (iii), there exists a such that when $t^* \in [a\tau, \tau]$ and $p(t^*) \geq p(t_1) + p(t_2)$, we have $t_1 = 0$ or $t_2 = 0$. It follow that when $t_1 \neq 0, t_2 \neq 0$ and $t^* \in [a\tau, \tau]$, we have $p(t_1 + t_2) < p(t_1) + p(t_2)$. Without loss of genearlity, we assume that $t_1 \leq t_2$. Then, we have

$$\frac{p(t^*) - p(t^* - t_1)}{t_1} < \frac{p(t_1)}{t_1}.$$

Notice that the right term is non-increasing with the increment of t_1 as p is a concave function and the left term is non-decreasing with the increment of t_1 when t^* is fixed. As

$t_1 \leq t^*/2$, we have $\frac{p(t_1)}{t_1} \geq k_1(t^*) \coloneqq \frac{p(t^*/2)}{t^*/2}$ and $\frac{p(t^*) - p(t^* - t_1)}{t_1} \leq k_2(t^*) \coloneqq \frac{p(t^*) - p(t^*/2)}{t^*/2}$. As p is not linear on $[0, t^*]$, we have $k_1(t^*) > k_2(t^*)$.

On the other hand, we can see that when $p(t_1 + t_2) + \epsilon \geq p(t_1) + p(t_2)$,

$$\frac{p(t_1 + t_2) - p(t_2)}{t_1} + \frac{\epsilon}{t_1} \geq \frac{p(t_1)}{t_1}.$$

Assume $t_1 < t_2$, we have $k_2(t^*) + \epsilon/t_1 \geq k_1(t^*)$ [1]. As a result $t_1 \leq \frac{\epsilon}{k_1(t^*) - k_2(t^*)}$. Note that k_1 and k_2 are defined on a closed interval $[a\tau, \tau]$ by (iii), giving us that $\min_{t \in [a\tau, \tau]}(k_1(t) - k_2(t)) > 0$. Therefore, $\exists a \in (0, 1), \forall \delta > 0, \exists \epsilon_0 = \min_{t \in [a\tau, \tau]}(k_1(t) - k_2(t)) \cdot \delta, \forall \epsilon < \epsilon_0$, if $t_1 + t_2 = t^* \in [a\tau, \tau]$ and $p(t_1 + t_2) + \epsilon \geq p(t_1) + p(t_2)$, then $t_1 \leq \frac{\epsilon}{k_1(t^*) - k_2(t^*)} \leq \delta$. Therefore, there is at most one i such that $t_i \notin B(0, \delta)$.

Now consider the case when $l > 2$ and $t_1, \ldots, t_l \geq 0$. If there are more than one i such that $t_i \notin B(0, \delta)$, assume t_1 and t_2 are two of them. By (ii), we have

$$\sum_{i=1}^l p(t_i) \geq p(t_1) + p\left(\sum_{i=2}^l t_i\right).$$

If $t_1 + \sum_{i=2}^n t_i \in [a\tau, \tau]$ and $p(t_1 + \sum_{i=2}^l t_i) + \epsilon \geq \sum_{i=1}^l p(t_i) \geq p(t_1) + p(\sum_{i=2}^l t_i)$, either t_1 or $\sum_{i=2}^n t_i$ should be inside $B(0, \delta)$. This is contradictory to our assumption that both t_1 and t_2 are outside $B(0, \delta)$. To this point, we prove (iv) when $t_1, \cdots, t_l \geq 0$.

Next, we prove the lemma when $t_1, \cdots, t_l$ could be smaller than 0. Suppose $t^* = \sum_{i=1}^l t_i \in [a\tau, \tau]$ and $p(t^*) + \epsilon \geq \sum_{i=1}^l p(t_i)$. We consider two cases separately. In the first case, assume that there is one $t_i \leq -\delta$. Without loss of generality, we assume that $t^* > 0$. Then we can choose $\alpha = \delta, \beta = t^* - \alpha$ and get

$$p(\alpha + \beta) + \epsilon = p(t^*) + \epsilon \geq \sum_{i \in \{j : t_j < 0\}} p(t_i) + \sum_{i \in \{j : t_j > 0\}} p(t_i) \geq p(\alpha) + p(\beta),$$

which is a contradiction to the previous proof that only one of α, β could be outside of $B(0, \delta)$ as δ is smaller than $t^*/2$ by our choice and $\sum_{i \in \{j : t_j > 0\}} t_i > t^* > t^* - \alpha$. We then proceed to the case when there is one $t_i \geq \delta$ and one $t_j \geq \delta$. Suppose that $\alpha = t_i \geq t_j = \beta$. If $\alpha + \beta > t^*$, we set $\alpha' = \delta + \frac{t^* - 2\delta}{\alpha + \beta - 2\delta} \cdot (\alpha - \delta)$ and $\beta' = \delta + \frac{t^* - 2\delta}{\alpha + \beta - 2\delta} \cdot (\beta - \delta)$. It is easy to verify that

$$p(\alpha' + \beta') + \epsilon = p(t^*) + \epsilon \geq \sum_{i=1}^l p(t_i) \geq p(\alpha) + p(\beta) \geq p(\alpha') + p(\beta'),$$

which is a contradiction. If $\alpha + \beta < t^*$, we can verify that

$$p(\alpha + \beta + t^* - \alpha - \beta) + \epsilon = p(t^*) + \epsilon \geq \sum_{i=1}^l p(t_i) \geq p(\alpha) + p(\beta) + p(t^* - \alpha - \beta),$$

which is also a contradiction. To this point, we prove the case that $t_1, \cdots, t_l$ could be smaller than 0, which completes the proof of the lemma.

1. For the case when $t_1 = 0$, (iv) holds trivially.

Remark. In the proof of (iv), our choice of ϵ is linear to δ given δ. However, in the case of L_0, ϵ could be any constant smaller than 1 no matter what δ is. This property of L_0 has wide applications in statistical problems. Actually, suppose that penalty function is indexed by δ and p_δ satisfies

$$p_\delta(\delta) - p_\delta(a\tau) + p_\delta(a\tau - \delta) \geq C$$

for some constant C, then $\forall \delta > 0$ and $\epsilon \leq C$, the proposition stated in (iv) holds. To prove this, just note that if $p(t_1+t_2)-p(t_2)+\epsilon > p(t_1)$ and $t_1 > \delta$, then $p(t_1)-p(t_1+t_2)+p(t_2) > p(\delta)-p(a\tau)+p(a\tau-\delta) \geq C$ which is a contradiction to that ϵ should be smaller than C. ■

Lemma 4 states the key properties of the penalty function p. Property (iv) is of special interest. It indicates that if we can manipulate the sum of non-negative variables to let it lie within $[a\tau, \tau]$ while minimizing the penalty function, we can be sure that only one variable has large absolute value.

Our second lemma explores the relationship between the penalty function p and the loss function ℓ.

Lemma 5 *Let Assumption 1 hold. Let $f(\cdot)$ be a convex function with a unique minimizer $\hat{\tau} \in (a\tau, \tau)$ and $\frac{f(\hat{\tau}\pm x)-f(\hat{\tau})}{x^N} \geq C (0 < x < \bar{\delta})$ for some $N \in \mathbb{Z}^+, \bar{\delta} \in \mathbb{R}^+, C \in \mathbb{R}^+$. Define*

$$g_\mu(t) = p(|t|) + \mu \cdot f(t),$$

where $\mu > 0$. Let $h(\mu)$ be the minimum value of $g_\mu(\cdot)$. We have $\forall \delta < \bar{\delta}, \mu_\delta > \frac{p(|\hat{\tau}|)2^N}{C\delta^N}, \exists \epsilon_0 = \mu_\delta \cdot C \cdot \left(\frac{\delta}{2}\right)^N - p(|\hat{\tau}|)$: if t satisfies $h(\mu_\delta) + \epsilon_0 \geq g_{\mu_\delta}(t) \geq h(\mu_\delta)$, then $t \in [\hat{\tau} - \delta/2, \hat{\tau} + \delta/2]$.

Proof First, we can see that when $t > \hat{\tau} + \delta/2$, we have

$$\begin{aligned} g_{\mu_\delta}(t) &\geq p(|\hat{\tau}|) + \mu_\delta \cdot f(t) > p(|\hat{\tau}|) + \mu_\delta \cdot f(\hat{\tau} + \delta/2) \geq p(|\hat{\tau}|) + \mu_\delta \cdot f(\hat{\tau}) + \mu_\delta \cdot C \cdot \left(\frac{\delta}{2}\right)^N \\ &= g_{\mu_\delta}(\hat{\tau}) + \mu_\delta \cdot C \cdot \left(\frac{\delta}{2}\right)^N \geq h(\mu_\delta) + \mu_\delta \cdot C \cdot \left(\frac{\delta}{2}\right)^N \geq h(\mu_\delta) + \epsilon_0, \end{aligned}$$

by the definition of $f(\cdot)$. When $t < \hat{\tau} - \delta/2$, we have

$$\begin{aligned} g_{\mu_\delta}(t) &\geq \mu_\delta \cdot f(t) > \mu_\delta \cdot f(\hat{\tau} - \delta/2) \geq \mu_\delta \cdot f(\hat{\tau}) + \mu_\delta \cdot C \cdot \left(\frac{\delta}{2}\right)^N \\ &= \mu_\delta \cdot f(\hat{\tau}) + \frac{p(|\hat{\tau}|)2^N}{C\delta^N} \cdot C \cdot \left(\frac{\delta}{2}\right)^N + \left(\mu_\delta - \frac{p(|\hat{\tau}|)2^N}{C\delta^N}\right) \cdot C \cdot \left(\frac{\delta}{2}\right)^N \\ &\geq h(\mu_\delta) + \mu_\delta \cdot C \cdot \left(\frac{\delta}{2}\right)^N - p(|\hat{\tau}|). \end{aligned}$$

Therefore, when we choose $\epsilon_0 = \mu_\delta \cdot C \cdot \left(\frac{\delta}{2}\right)^N - p(|\hat{\tau}|)$, point t satisfying $h(\mu_\delta) + \epsilon_0 \geq g_{\mu_\delta}(t) \geq h(\mu_\delta)$ must lie in $[\hat{\tau} - \delta/2, \hat{\tau} + \delta/2]$. ■

Lemma 6 *Let Assumption 1 hold and let $f(\cdot)$ be a convex function with a unique minimizer $\hat{\tau} \in (a\tau, \tau)$ and $\frac{f(\hat{\tau}\pm x)-f(\hat{\tau})}{x^N} \geq C_1 (0 < x < \bar{\delta})$ for some $N \in \mathbb{Z}^+, \bar{\delta} \in \mathbb{R}^+, C_1 \in \mathbb{R}^+$. Let $h(\mu)$ be the minimum value of $g_\mu(x) = p(|x|) + \mu \cdot f(x)$, then we have*

(i) $\forall \mu \in \mathbb{Z}^+, t_1, ..., t_n \in \mathbb{R} : \sum_{j=1}^n p(|t_j|) + \mu \cdot f\left(\sum_{j=1}^n t_j\right) \geq h(\mu)$.

(ii) $\exists \kappa = \min_{t \in [a\tau, \tau]} \{\frac{2p(t/2)-p(t)}{t}\}, \forall \delta \leq \min\{\bar{\delta}, 4\tau - 4\hat{\tau}, 4\hat{\tau} - 4a\tau\}, \exists \mu = \frac{p(|\hat{\tau}|)4^{N+1}}{C_1\delta^N}, \epsilon_0 = \kappa \cdot \frac{\delta}{n}, \forall \theta \in [\hat{\tau} - \delta/4, \hat{\tau} + \delta/4]$: *if $t_1, ..., t_n \in \mathbb{R}$ satisfy*

$$h(\mu) + \epsilon_0 \geq \sum_{j=1}^n p(|t_j|) + \mu \cdot f\left(\sum_{j=1}^n t_j\right) \geq h(\mu), \tag{2}$$

then $t_i \in B(\theta, \delta)$ for one i and $t_j \in B(0, \delta)$ for all $j \neq i$.

Proof

We first prove (i). We consider two cases separately. In the first case, we suppose that $|\sum_{j=1}^n t_j| > \tau$. Then we have

$$\sum_{j=1}^n p(|t_j|) \geq \sum_{j=1}^n p\left(\frac{\tau}{\sum_{k=1}^n |t_k|} \cdot |t_j|\right) \geq p\left(\sum_{j=1}^n \frac{\tau}{\sum_{k=1}^n |t_k|} \cdot |t_j|\right) \geq p(\tau),$$

where the first inequality is inferred by the monotonicity of p and the second inequality is due to (ii) of Lemma 4. Thus, we have

$$\sum_{j=1}^n p(|t_j|) + \mu \cdot f\left(\sum_{j=1}^n t_j\right) > \min\{p(\tau) + \mu \cdot f(\tau), p(\tau) + \mu \cdot f(-\tau)\} \geq h(\mu).$$

As a result, we can see that (i) holds when $|\sum_{j=1}^n t_j| > \tau$. In the second case, we suppose $|\sum_{j=1}^n t_j| \leq \tau$ and obtain

$$\sum_{j=1}^n p(|t_j|) \geq \sum_{j=1}^n p\left(\frac{|\sum_{k=1}^n t_k|}{\sum_{k=1}^n |t_k|} |t_j|\right) \geq p\left(\sum_{j=1}^n \frac{|\sum_{k=1}^n t_k|}{\sum_{k=1}^n |t_k|} |t_j|\right) \geq p\left(\left|\sum_{j=1}^n t_j\right|\right),$$

where the second inequality is due to (ii) of Lemma 4. It follows that

$$\sum_{j=1}^n p(|t_j|) + \mu \cdot f\left(\sum_{j=1}^n t_j\right) \geq p\left(\left|\sum_{j=1}^n t_j\right|\right) + \mu \cdot f\left(\sum_{j=1}^n t_j\right) = g_\mu\left(\sum_{j=1}^n t_j\right) \geq h(\mu). \tag{3}$$

which completes our proof of (i).

We then prove (ii). Assume equation (2) holds. If $\sum_{j=1}^n t_j > \tau$, we can see that by choosing $\epsilon_0 \leq g_\mu(\tau) - g_\mu(\hat{\tau})$, we have

$$\sum_{j=1}^n p(|t_j|) + \mu \cdot f\left(\sum_{j=1}^n t_j\right) > g_\mu(\tau) = g_\mu(\hat{\tau}) + g_\mu(\tau) - g_\mu(\hat{\tau}) \geq h(\mu) + \epsilon_0.$$

We will show later that our choice of ϵ_0 is indeed smaller than $g_\mu(\tau) - g_\mu(\hat{\tau})$. We will also show later that equation (2) cannot hold when $\sum_{j=1}^n t_j < -\tau$ under our choice of parameters. Thus, if equation (2) holds, then $|\sum_{j=1}^n t_j| \leq \tau$, which implies that

$$p\left(\left|\sum_{j=1}^n t_j\right|\right) + \mu \cdot f\left(\sum_{j=1}^n t_j\right) \leq h(\mu) + \epsilon_0, \tag{4}$$

by equation (2) and the first inequality of (3), and

$$\sum_{j=1}^n p(|t_j|) \leq p\left(\left|\sum_{j=1}^n t_j\right|\right) + \epsilon_0, \tag{5}$$

due to equation (2) and equation (3). Note that we just need to prove the case when δ is sufficiently small. Thus, we assume in the following paper that δ is smaller than $\bar{\delta}, 4\tau - 4\hat{\tau}, 4\hat{\tau} - 4a\tau$.

Consider the case when equation (4) holds. By Lemma 6, if we choose $\mu = \frac{p(|\hat{\tau}|)4^{N+1}}{C\delta^N}$ and $\epsilon_1 = 3p(|\hat{\tau}|)$, then all of the points t such that $h(\mu)+\epsilon_1 \geq g_\mu(t) \geq h(\mu)$ lie in $[\hat{\tau}-\delta/4, \hat{\tau}+\delta/4]$. Thus, we have $\sum_{j=1}^n t_j \in [a\tau, \tau]$ and $\sum_{j=1}^n t_j \in B(\theta, \frac{\delta}{2})$ for all $\theta \in [\hat{\tau} - \delta/4, \hat{\tau} + \delta/4]$. Note that $g_\mu(t)$ is non-increasing when $t < 0$, meaning that equation (2) cannot hold under our choice of ϵ_1 when $\sum_{j=1}^n t_j \leq -\tau$.

On the other hand, if equation (2) holds, equation (5) should also hold. By (iv) of Lemma 4, for the same δ, $\exists \epsilon_2 = \min_{t\in[a\tau,\tau]}(k_1(t) - k_2(t)) \cdot \frac{\delta}{2n-2}$, there is at most one i such that $t_i \notin B(0, \frac{\delta}{2n-2})$. As $\sum_{j=1}^n t_j \in B(\theta, \frac{\delta}{2})$, we have $t_i \in B(\theta, \delta)$ for all $i = 1, \cdots, n$. Observe that $g_\mu(\tau) - g_\mu(\hat{\tau})$ is always larger than ϵ_1. Also, $\epsilon_1 > \epsilon_2$ if δ is sufficiently small. Therefore, $\exists \kappa = \min_{t\in[a\tau,\tau]}(k_1(t) - k_2(t))/2$, $\forall \delta \leq \min\{\bar{\delta}, 4\tau - 4\hat{\tau}, 4\hat{\tau} - 4a\tau\}, \exists \mu = \frac{p(|\hat{\tau}|)4^{N+1}}{C\delta^N}, \epsilon = \kappa \cdot \frac{\delta}{n}, \forall \theta \in [\hat{\tau} - \delta/4, \hat{\tau} + \delta/4]$: if $h(\mu) + \epsilon \geq g_\mu(\sum_{j=1}^n t_i)$, then $t_i \in B(\theta, \delta)$ for some i while $t_j \in B(0, \delta)$ for all $j \neq i$. ■

6.2. Proof of Theorem 1

Now we are ready to prove Theorem 1.

Proof Suppose that we are given the input to the 3-partition problem, i.e., $3m$ positive integers $s_1, ..., s_{3m}$. Assume *without loss of generality* that all s_i's are upper bounded by some polynomial function $M(m)$. This restriction on the input space does not weaken our result, because the 3-partition problem is strongly NP-hard.

In what follows, we construct a reduction from the 3-partition problem to Problem 1. We assume without loss of generality that $\frac{1}{4m}\sum_{j=1}^{3m} s_j < s_i < \frac{1}{2m}\sum_{j=1}^{3m} s_j$ for all $i = 1, \ldots, n$. Such condition can always be satisfied by adding a sufficiently large integer to all s_i's.

Step 1: The Reduction. The reduction is developed through the following steps.

1. For the interval $[a\tau, \tau]$, we choose $\{b_{1i}\}_{i=1}^{k_1}$ such that $\ell_1(y) = \frac{1}{\lambda}\sum_{i=1}^{k_1} \ell(y, b_{1i})$ satisfies Assumption 1 with constants $C, N, \bar{\delta}$ and has a unique minimizer $\hat{\tau}$ inside the interval $(a\tau, \tau)$. Let $\kappa = \min_{t\in[a\tau,\tau]}\{\frac{2p(t/2)-p(t)}{t}\}$. Let $\delta \leq \{\frac{a\tau}{9m\cdot M(m)}, \bar{\delta}, 4\tau - 4\hat{\tau}, 4\hat{\tau} - 4a\tau\}$, $\mu \geq \frac{p(|\hat{\tau}|)4^{N+1}}{C_1\delta^N}$ and $\epsilon = \kappa \cdot \frac{\delta}{3m}$ such that Lemma 6 is satisfied. Note that $\epsilon \geq \frac{C_3}{m^2\cdot M(m)}$ for some constant C_3 by our construction.

2. For the μ and ϵ chosen in the previous step, all the minimizers of $g_\mu(x) = p(|x|) + \mu \cdot \ell_1(x)$ lie in $[\hat{\tau}-\delta/4, \hat{\tau}+\delta/4]$ by Lemma 6. By the Lipschitz continuity of $p(|x|), f(x)$ and thus $g_\mu(x)$ on $[a\tau, \tau]$, there exists $\delta_\epsilon = \frac{\epsilon}{6mK}$ (K is the Lipschitz constant) such that we can find in polynomial time an interval $[\theta_1, \theta_2]$ where $\theta_2 - \theta_1 = \delta_\epsilon$ and $g_\mu(x) - g_\mu(t^*) < \frac{\epsilon}{6m}$ for $x \in [\theta_1, \theta_2]$. This interval can be find in polynomial time as $g_\mu(x)$ is Lipschitz continuous.

3. By Assumption 2, for the interval $[\theta_1, \theta_2]$, we choose $\{b_{2i}\}_{i=1}^{k_2}$ to construct a loss function $\ell_2 : \mathbb{R} \mapsto \mathbb{R}$ in polynomial time with regard to $1/\delta_\epsilon$ such that $\ell_2(y) = \frac{1}{\lambda}\sum_{i=1}^{k_2} \ell(y, b_{2i})$ has a unique minimizer at $\tilde{t} \in [\theta_1, \theta_2]$. We choose

$$\nu = \left\lceil \epsilon / \max\left(\ell_2(\tilde{t} + 2\delta m) - \ell_2(\tilde{t}), \ell_2(\tilde{t} - 2\delta m) - \ell_2(\tilde{t})\right)\right\rceil + 1,$$

and construct function $f : \mathbb{R}^{3m\times m} \mapsto \mathbb{R}$ where

$$f(x) = \lambda \cdot \sum_{i=1}^{3m}\sum_{j=1}^{m} p\left(|x_{ij}|\right) + \lambda\mu \cdot \sum_{i=1}^{3m} \ell_1\left(\sum_{j=1}^{m} x_{ij}\right) + \lambda\nu \cdot \sum_{j=1}^{m} \ell_2\left(\sum_{i}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} x_{ij}\right). \tag{6}$$

Note that by (iii) of Assumption 2, ν is polynomial in $\max(\lceil \frac{1}{\delta_\epsilon} \rceil, \lceil \theta_2 \rceil)$. In the rest of the paper, we ignore the $\lceil \theta_2 \rceil$ term in the bound as it can be upperbounded by τ, which can be taken as a constant in the reduction.

4. Let $\Phi_1 = 3m \cdot p(|\tilde{t}|) + \mu \cdot 3m \cdot \ell_1(\tilde{t}) - \frac{\epsilon}{2}$ and $\Phi_2 = \nu \cdot m \cdot \ell_2(\tilde{t})$. We claim that

   (i) If there exists z such that

   $$\Phi_1 + \Phi_2 + \epsilon \geq \frac{1}{\lambda} f(z) \geq \Phi_1 + \Phi_2,$$

   then we obtain a feasible assignment for the 3-partition problem as follows: If $z_{ij} \in B(\tilde{t}, \delta)$, we assign number i to subset j.

   (ii) If the 3-partition problem has a solution, we have $\frac{1}{\lambda}\min_x f(x) \leq \Phi_1 + \Phi_2 + \frac{\epsilon}{2}$.

5. Choose $r = \left\lceil \left(\frac{2(3m\cdot\lambda\cdot\mu\cdot k_1 + m\cdot\lambda\cdot\nu\cdot k_2)^{c_1}(3m^2)^{c_2}}{\epsilon/\kappa}\right)^{1/(1-c_1-c_2)} \right\rceil$ where c_1 and c_2 are two arbitrary constants that $c_1 + c_2 < 1$. Construct the following instance of Problem 1:

$$\begin{aligned}\min_{x^{(1)},\cdots,x^{(r)}\in\mathbb{R}^{3m\times m}} \sum_{q=1}^{r} f(x^{(q)}) = \min_{x^{(1)},\cdots,x^{(r)}\in\mathbb{R}^{3m\times m}} \lambda \cdot \sum_{q=1}^{r}\sum_{i=1}^{3m}\sum_{j=1}^{m} p(|x_{ij}^{(q)}|) + \\ \lambda\mu \sum_{q=1}^{r}\sum_{i=1}^{3m}\sum_{t=1}^{k_1} \ell\left(\sum_{j=1}^{m} x_{ij}^{(q)}, b_{1t}\right) + \lambda\nu \sum_{q=1}^{r}\sum_{j=1}^{m}\sum_{t=1}^{k_2} \ell\left(\sum_{i=1}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} x_{ij}^{(q)}, b_{2t}\right),\end{aligned} \tag{7}$$

where the input data are coefficients of x and the values $b_{11}, \ldots, b_{1t}, b_{21}, \ldots, b_{2t}$. The variable dimension d is $r \cdot 3m^2$ and the sample size n is $\lambda \cdot \mu \cdot r \cdot 3m \cdot k_1 + \lambda \cdot \nu \cdot r \cdot m \cdot k_2$. The input size is polynomial with respect to m. Our choice of r is the solution to $\epsilon r = 2\kappa n^{c_1} d^{c_2}$ where $\kappa = \min_{t \in [a\tau, \tau]} \{\frac{2p(t/2) - p(t)}{t}\}$.

The parameters μ, ν, δ, r, d are bounded by polynomial functions of m. Computing their values also takes polynomial time. The parameter k_1 and k_2 is a constant determined by the loss function ℓ and is not related to m. As a result, the reduction is polynomial.

6. Let $z^{(1)}, \cdots z^{(r)} \in \mathbb{R}^{3m \times m}$ be a $\lambda \cdot \kappa \cdot n^{c_1} d^{c_2}$-optimal solution to problem (13) such that $\sum_{i=1}^r f(z^{(i)}) \le \min_{x^{(1)}, \cdots, x^{(r)}} \sum_{i=1}^r f(x^{(i)}) + \lambda \cdot \kappa \cdot n^{c_1} d^{c_2}$. We claim that

   (iii) If the approximate solution $z^{(1)}, \cdots z^{(r)}$ satisfies

   $$\frac{1}{\lambda} \sum_{i=1}^{r} f(z^{(i)}) \le r\Phi_1 + r\Phi_2 + 2\kappa n^{c_1} d^{c_2}, \tag{8}$$

   we can choose one $z^{(i)}$ such that $\Phi_1 + \Phi_2 + \epsilon \ge \frac{1}{\lambda} f(z^{(i)}) \ge \Phi_1 + \Phi_2$ and obtain a feasible assignment: If $z_{ij}^{(i)} \in B(\tilde{t}, \delta)$, we assign number i to subset j. If the $\lambda \cdot \kappa \cdot n^{c_1} d^{c_2}$-optimal solution $z^{(1)}, \cdots z^{(r)}$ does not satisfy (8), the 3-partition problem has no feasible solution.

We have constructed a polynomial reduction from the 3-partition problem to finding an $\lambda \cdot \kappa \cdot n^{c_1} d^{c_2}$-optimal solution to problem (13). In what follows, we prove that the reduction works.

Step 2: Proof of Claim (i). We begin with the proof (i). By our choice of μ and Lemma 6(i), we can see that for all $x \in \mathbb{R}^{3m \times m}$,

$$\sum_{i=1}^{3m} \sum_{j=1}^{m} p(|x_{ij}|) + \mu \cdot \sum_{i=1}^{3m} \ell_1 \left(\sum_{j=1}^{m} x_{ij} \right) \ge 3m \cdot p(|t^*|) + \mu \cdot 3m \cdot \ell_1(t^*) \ge \Phi_1,$$

where the last inequality is due to that $g_\mu(\tilde{t}) - g_\mu(t^*) < \frac{\epsilon}{6m}$. By the fact $\tilde{t} = \operatorname{argmin}_t \ell_2(t)$, we have for all $x \in \mathbb{R}^{3m \times m}$ that

$$\nu \cdot \sum_{j=1}^{m} h \left(\sum_{i=1}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} x_{ij} \right) \ge \nu \cdot m \cdot \ell_2(\tilde{t}) = \Phi_2.$$

Thus we always have $\min_z \frac{1}{\lambda} f(z) \ge \Phi_1 + \Phi_2$. Now if there exists z such that $\Phi_1 + \Phi_2 + \epsilon \ge \frac{1}{\lambda} f(z) \ge \Phi_1 + \Phi_2$, we must have

$$\Phi_1 + \epsilon \ge \sum_{i=1}^{3m} \sum_{j=1}^{m} p(|z_{ij}|) + \mu \cdot \sum_{i=1}^{3m} h \left(\sum_{j=1}^{m} z_{ij} \right) \ge \Phi_1, \tag{9}$$

and

$$\Phi_2 + \epsilon \ge \nu \cdot \sum_{j=1}^{m} h \left(\sum_{i=1}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} z_{ij} \right) \ge \Phi_2. \tag{10}$$

In order for equation (9) to hold, we have that for all i,

$$p(|\tilde{t}|) + \mu \cdot \ell_1(\tilde{t}) + \frac{\epsilon}{2} \ge \sum_{j=1}^{m} p(|z_{ij}|) + \mu \cdot \ell_1 \left(\sum_{j=1}^{m} z_{ij} \right) \ge p(|t^*|) + \mu \cdot \ell_1(t^*).$$

Consider an arbitrary i. By Lemma 6(ii) and $g_\mu(\tilde{t}) - g_\mu(t^*) < \frac{\epsilon}{6m}$, we have $z_{ij} \in B(\tilde{t}, \delta)$ for one j while $z_{ik} = 0$ for all $k \ne j$. If $z_{ij} \in B(\tilde{t}, \delta)$, we assign number i to subset j. As $\delta < a\tau/2 \le \tilde{t}/2$, $B(\tilde{t}, \delta)$ and $B(0, \delta)$ are not overlapping. Thus each number index i is assigned to exactly one subset index j. Therefore the assignment is feasible.

We claim that every subset sum must equal to $\sum_{i=1}^{3m} s_i/m$. Assume that the jth subset sum is greater than or equal to $\sum_{i=1}^{3m} s_i/m + 1$. Let $I_j = \{i \mid z_{ij} \in B(\tilde{t}, \delta)\}$. Thus, $\sum_{i \in I_j} s_i \ge \sum_{i=1}^{3m} s_i/m + 1$. As a result, we have

$$\begin{aligned} \sum_{i=1}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} z_{ij} &\ge \sum_{i \in I_1} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} (\tilde{t} - \delta) + \sum_{i \in I_2} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} (-\delta) \\ &\ge \frac{\sum_{i=1}^{3m} s_i/m + 1}{\sum_{i=1}^{3m} s_i/m} \tilde{t} - \delta m = \tilde{t} + \frac{\tilde{t}}{\sum_{i=1}^{3m} s_i/m} - \delta m. \end{aligned}$$

Because $s_i \le M(m)$ for all i and $\delta = \frac{a\tau}{9m \cdot M(m)}$, we have

$$\frac{\tilde{t}}{\sum_{i=1}^{3m} s_i/m} - \delta m \ge \frac{a\tau}{3m \cdot M(n)} m - \delta m = 2\delta m > 0.$$

Since h is a convex function with minimizer y^*, we apply the preceding inequalities and further obtain

$$\ell_2 \left(\sum_{i=1}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} z_{ij} \right) \ge \ell_2(\tilde{t} + 2\delta m).$$

By our construction of ν and Assumption 1(iii), we further have

$$\nu \cdot \left(\ell_2 \left(\sum_{i=1}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} z_{ij} \right) - \ell_2(\tilde{t}) \right) \ge \nu \cdot \big(\ell_2(\tilde{t} + 2\delta m) - \ell_2(\tilde{t}) \big) > \epsilon. \tag{11}$$

However, in order for equation (10) to hold, we have that for all j,

$$\nu \cdot \ell_2(\tilde{t}) + \epsilon \ge \nu \cdot \ell_2 \left(\sum_{i=1}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} z_{ij} \right) \ge \nu \cdot \ell_2(\tilde{t}),$$

yielding a contradiction to (11). We could prove similarly that it is not possible for any subset sum to be strictly smaller than $\frac{1}{m} \sum_{i=1}^{3m} s_i$. Therefore, the sum of every subset equals to $\sum_{i=1}^{3m} s_i/m$. Finally, using the assumption that $\frac{1}{4m} \sum_{i=1}^{3m} s_i < s_i < \frac{1}{2m} \sum_{i=1}^{3m} s_i$, each subset has exactly three components. Therefore the assignment is indeed a solution to the 3-partition problem.

Step 3: Proof of Claim (ii). Suppose we have a solution to the 3-partition problem. Now we construct z to the optimization problem such that $f(z) \le \Phi_1 + \Phi_2 + \frac{\epsilon}{2}$. For all

$1 \le i \le 3m$, if number i is assigned to subset j, let $z_{ij} = \tilde{t}$ and $z_{ik} = 0$ for all $k \neq j$. We can easily verify that

$$\sum_{i=1}^{3m}\sum_{j=1}^{m} p\left(|z_{ij}|\right) + \mu \cdot \sum_{i=1}^{3m} \ell_1\left(\sum_{j=1}^{m} z_{ij}\right) = 3m \cdot \left(p(\tilde{t}) + \mu \cdot \ell_1(\tilde{t})\right) = \Phi_1 + \frac{\epsilon}{2},$$

Also, we have

$$\nu \cdot \sum_{j=1}^{m} \ell_2\left(\sum_{i=1}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} z_{ij}\right) = \nu \cdot m \cdot \ell_2(\tilde{t}) = \Phi_2.$$

Therefore,

$$\frac{1}{\lambda} f(z) \le \Phi_1 + \Phi_2 + \frac{\epsilon}{2}. \tag{12}$$

which completes the proof of (ii).

Step 4: Proof of Claim (iii). Suppose that the $\lambda \cdot \kappa \cdot n^{c_1} d^{c_2}$-optimal solution satisfies (8), i.e., $\frac{1}{\lambda}\sum_{i=1}^{r} f(z^{(i)}) \le r\Phi_1 + r\Phi_2 + 2\kappa n^{c_1} d^{c_2}$. It follows that there exists at least one term $z^{(i)}$ such that

$$\frac{1}{\lambda} f(z^{(i)}) \le \Phi_1 + \Phi_2 + \frac{2\kappa n^{c_1} d^{c_2}}{r} \le \Phi_1 + \Phi_2 + \epsilon.$$

where the second inequality equality uses $\epsilon r = 2\kappa n^{c_1} d^{c_2}$. Therefore, by claim (ii), we can find a solution to the 3-partition problem.

Suppose that the 3-partition problem has a solution. By claim (ii), there exists z such that $\frac{1}{\lambda} f(z) \le \Phi_1 + \Phi_2 + \frac{\epsilon}{2}$. Thus we have

$$\min_{x^{(1)},\cdots,x^{(r)}} \frac{1}{\lambda}\sum_{i=1}^{r} f(x^{(i)}) \le \frac{r}{\lambda} f(z) \le r\Phi_1 + r\Phi_2 + \kappa n^{c_1} d^{c_2}.$$

Thus if $z^{(1)}, \cdots z^{(r)}$ is a $\lambda \cdot \kappa \cdot n^{c_1} d^{c_2}$-optimal solution to (13), we have

$$\frac{1}{\lambda}\sum_{i=1}^{r} f(z^{(i)}) \le \min_{x^{(1)},\cdots,x^{(r)}} \frac{1}{\lambda}\sum_{i=1}^{r} f(x^{(i)}) + \kappa n^{c_1} d^{c_2} \le r\Phi_1 + r\Phi_2 + 2\kappa n^{c_1} d^{c_2}$$

implying that the relation (8) must hold. If (8) is not satisfied, the 3-partition problem has no solution. ■

Remark. When the loss function is L_2 loss, we can move $\lambda\mu$ and $\lambda\nu$ of equation (13) into the loss. Specifically, we have

$$\begin{aligned}
&\min_{x^{(1)},\cdots,x^{(r)}\in\mathbb{R}^{3m\times m}} \sum_{q=1}^{r} f(x^{(q)}) = \min_{x^{(1)},\cdots,x^{(r)}\in\mathbb{R}^{3m\times m}} \lambda \cdot \sum_{q=1}^{r}\sum_{i=1}^{3m}\sum_{j=1}^{m} p(|x_{ij}^{(q)}|) + \\
&\sum_{q=1}^{r}\sum_{i=1}^{3m}\left(\sum_{j=1}^{m}\sqrt{\lambda\mu}x_{ij}^{(q)} - \sqrt{\lambda\mu}b_1\right)^2 + \sum_{q=1}^{r}\sum_{j=1}^{m}\left(\sum_{i=1}^{3m}\frac{\sqrt{\lambda\nu}s_i}{\sum_{i'=1}^{3m} s_{i'}/m}x_{ij}^{(q)} - \sqrt{\lambda\nu}b_2\right)^2,
\end{aligned} \tag{13}$$

where μ, ν is chosen such that $\sqrt{\lambda\mu}, \sqrt{\lambda\nu}$ are rational numbers. In this case, the variable dimension is $r \cdot 3m^2$ and the sample size n is $4r \cdot m$. Our choice of r is the solution to $\epsilon r = 2\kappa n^{c_1} d^{c_2}$ which is $r = \left\lceil \left(\frac{2(4m)^{c_1}(3m^2)^{c_2}}{\epsilon/\kappa}\right)^{1/(1-c_1-c_2)} \right\rceil$. The value of r doesn't depend on λ and p, which means that we can plug in any λ, p and the reduction is still polynomial in m. It means that for any choice of λ and p, it is strongly NP hard to find a $\lambda\kappa n^{c_1} d^{c_2}$-optimal solution.

6.3. Proof of Theorem 2

Next we study the complexity of Problem 2. The proof uses a basic duality between Problem 1 and Problem 2.

Proof We will use a reduction from the 3-partition problem to prove the theorem. The reduction is developed through the following steps. We first constructed a polynomial reduction from the 3-partition problem to finding an approximate solution to Problem 2. We then prove that the reduction works.

1. Given the input to the 3-partition problem, we conduct the first three steps of the reduction in Theorem 1 to compute $\mu, \nu, \tilde{t}$ and ϵ with $\lambda = 1$. The nuance is that we pick $\delta_\epsilon = \frac{\epsilon}{12mK}$ in step 2 so that $g_\mu(\tilde{t}) - g_\mu(t^*) \le \frac{\epsilon}{12m}$ where $g_\mu(x) = p(|x|) + \mu \cdot \ell_1(x)$. Denote $f(x) = \mu \cdot \sum_{i=1}^{3m}\sum_{t=1}^{k_1} \ell\left(\sum_{j=1}^{m} x_{ij}, b_{1t}\right) + \nu \cdot \sum_{j=1}^{m}\sum_{t=1}^{k_2} \ell\left(\sum_{i=1}^{3m} \frac{s_i}{\sum_{i'=1}^{3m} s_{i'}/m} x_{ij}, b_{2t}\right)$ and $q(x) = \sum_{i=1}^{3m}\sum_{j=1}^{m} p\left(|x_{ij}|\right)$.

2. Choose $r = \left\lceil \left(\frac{4(3m\cdot\mu\cdot k_1 + m\cdot\nu\cdot k_2)^{c_1}(3m^2)^{c_2}}{\epsilon/\kappa}\right)^{1/(1-c_1-c_2)} \right\rceil$ where c_1 and c_2 are two arbitrary constants that $c_1 + c_2 < 1$. Note that $\kappa n^{c_1} d^{c_2} = \frac{\epsilon r}{4}$ by our choice of r. Construct the following instance of Problem 2:

$$\min_{x^{(1)},x^{(2)},\ldots,x^{(r)}\in\mathbb{R}^{3m\times m}} \sum_{i=1}^{r} f(x^{(i)}) \quad \text{s.t.} \quad \sum_{i=1}^{r} q(x^{(i)}) \le \bar{K}, \tag{14}$$

   where $\bar{K} \in [3m \cdot r \cdot p(\tilde{t}), 3m \cdot r \cdot p(\tilde{t}) + \epsilon r/4)$. The coding size of $\bar{K}$ is bounded by a polynomial function of m because $\epsilon r/4$ and $3m \cdot r \cdot p(\tilde{t})$ are both bounded by a polynomial functions of n. Denote the minimizer of the minimization problem (14) to be $x_{\bar{K}}^{(1)}, \cdots x_{\bar{K}}^{(r)}$.

3. Let $\Phi_1 = 3m \cdot p(|\tilde{t}|) + \mu \cdot 3m \cdot \ell_1(\tilde{t}) - \frac{\epsilon}{4}$ and $\Phi_2 = \nu \cdot m \cdot \ell_2(\tilde{t})$. We claim that if the 3-partition problem has a solution, then

   (i) $\sum_{i=1}^{r} f(x_{\bar{K}}^{(i)}) + \sum_{i=1}^{r} q(x_{\bar{K}}^{(i)}) \le r\Phi_1 + r\Phi_2 + \frac{\epsilon r}{2}$.

   (ii) $\sum_{i=1}^{r} q(x_{\bar{K}}^{(i)}) \ge 3m \cdot r \cdot p(\tilde{t}) - \epsilon r/4$.

4. Suppose we have approximate solutions satisfying $\sum_{i=1}^{r} f(\hat{x}_{\bar{K}+\kappa n^{c_1} d^{c_2}}^{(i)}) \le \sum_{i=1}^{r} f(x_{\bar{K}}^{(i)})$, we claim that

(iii) If the approximate solutions satisfy

$$\sum_{i=1}^{r} f(\hat{x}^{(i)}_{\bar{K}+\kappa n^{c_1}d^{c_2}}) + \sum_{i=1}^{r} q(\hat{x}^{(i)}_{\bar{K}+\kappa n^{c_1}d^{c_2}}) \le r\Phi_1 + r\Phi_2 + 4\kappa n^{c_1}d^{c_2}, \tag{15}$$

we can choose one index k such that $\Phi_1+\Phi_2+\epsilon \ge f(\hat{x}^{(k)}_{\bar{K}+\kappa n^{c_1}d^{c_2}}) \ge \Phi_1+\Phi_2$ and obtain a feasible assignment: If $\left(\hat{x}^{(k)}_{\bar{K}+\kappa n^{c_1}d^{c_2}}\right)_{ij} \in B(\tilde{t},\delta)$, we assign number i to subset j. If the approximate solutions do not satisfy (15), the 3-partition problem has no feasible solution.

We begin with the proof of (i). By the condition that the 3-partition problem has a solution, we construct $x^* \in \mathbb{R}^{3m\times m}$ as follows. If number i is assigned to subset j, let $x^*_{ij} = \tilde{t}$ and $x^*_{ik} = 0$ otherwise. We can see that $x^{(1)} = \cdots = x^{(r)} = x^*$ satisfy the constraint of (14) with sparsity level $\bar{K}$ and $\sum_{i=1}^r q(x^{(i)}_{\bar{K}}) \le 3m\cdot r\cdot p(\tilde{t}) + \epsilon r/4$. Thus,

$$\begin{aligned}\sum_{i=1}^{r} f(x^{(i)}_{\bar{K}}) + \sum_{i=1}^{r} q(x^{(i)}_{\bar{K}}) &\le r\cdot f(x^*) + 3m\cdot r\cdot p(\tilde{t}) + \frac{\epsilon r}{4} = r\cdot 3m\cdot g_\mu(\tilde{t}) + r\Phi_2 + \frac{\epsilon r}{4}\\ &= r\cdot 3m\cdot\left(g_\mu(\tilde{t}) - \frac{\epsilon}{12m}\right) + r\Phi_2 + \frac{\epsilon r}{2} = r\Phi_1 + r\Phi_2 + \frac{\epsilon r}{2},\end{aligned} \tag{16}$$

where $g_\mu(x) = p(|x|) + \mu\cdot\ell_1(x)$. To prove (ii), we just need to notice that if $\sum_{i=1}^r q(x^{(i)}_{\bar{K}}) < 3m\cdot r\cdot p(\tilde{t}) - \epsilon r/4$, we would have $\sum_{i=1}^r f(x^{(i)}_{\bar{K}}) + \sum_{i=1}^r q(x^{(i)}_{\bar{K}}) < r\Phi_1 + r\Phi_2$ by the same reasoning of equation (16), yielding a contradiction as $\sum_{i=1}^r f(x^{(i)}_{\bar{K}}) + \sum_{i=1}^r q(x^{(i)}_{\bar{K}})$ will always be greater than or equal to $r\Phi_1 + r\Phi_2$.

Now we prove (iii). To prove the first half of the claim, we only need to use $4\kappa n^{c_1}d^{c_2} = r\epsilon$ and apply the proof of Theorem 1 to get the result. To prove the second half of the claim, assume that we have an algorithm that outputs $\hat{x}_{\bar{K}+\kappa n^{c_1}d^{c_2}}$ satisfying $\sum_{i=1}^r f(\hat{x}^{(i)}_{\bar{K}+\kappa n^{c_1}d^{c_2}}) \le \sum_{i=1}^r f(x^{(i)}_{\bar{K}})$. Suppose that the 3-partition problem has a solution. Then we have

$$\sum_{i=1}^{r} f(\hat{x}^{(i)}_{\bar{K}+\kappa n^{c_1}d^{c_2}}) + \sum_{i=1}^{r} q(\hat{x}^{(i)}_{\bar{K}+\kappa n^{c_1}d^{c_2}}) \le \sum_{i=1}^{r} f(x^{(i)}_{\bar{K}}) + \sum_{i=1}^{r} q(x^{(i)}_{\bar{K}}) + \frac{\epsilon r}{2} \le r\Phi_1 + r\Phi_2 + \epsilon r,$$

where the first inequality is due to (ii) and the second inequality is due to (i). It means that the approximate solutions satisfy (15). To this point, we have finished the proof of Theorem 2. ■

Remark. Note that K is the input of Problem 2, which means that for all ℓ and p, we only need to find a K that makes Problem 2 hard to solve. A natural question to ask is whether or not K could be the parameter of Problem 2 such that the hardness result still holds. Unfortunately, we have the following counterexample. Assume p is L_0 norm and $K = 1$. Then the constraint is $\sum_{j=1}^d L_0(x_j) \le 1$, which means that there is at most one component of x that is not equal to 0. In this case, we could solve the optimization problem in polynomial time by searching for the nonzero component of x.

6.4. Proof of Theorem 3

Using a similar argument, we can prove the last part of our main result.

Proof The proof is analogous to the proof of Theorem 2. We will use a reduction from the 3-partition problem to prove the theorem. The reduction is developed through the following steps.

1. Given the input to the 3-partition problem, we conduct the first reduction step of Theorem 2 to compute $\mu, \nu, \tilde{t}$ and ϵ with $\lambda = 1$. Let $f(x) = \mu\cdot\sum_{i=1}^{3m}\sum_{t=1}^{k_1}\ell\left(\sum_{j=1}^m x_{ij}, b_{1t}\right) + \nu\cdot\sum_{j=1}^m\sum_{t=1}^{k_2}\ell\left(\sum_{i=1}^{3m}\frac{s_i}{\sum_{i'=1}^{3m}s_{i'}/m}x_{ij}, b_{2t}\right)$ and $q(x) = \sum_{i=1}^{3m}\sum_{j=1}^m p(|x_{ij}|)$.

2. Choose $r = \left\lceil\left(\frac{4(3m\cdot\mu\cdot k_1 + m\cdot\nu\cdot k_2)^{c_1}(3m^2)^{c_2}}{\epsilon/\kappa}\right)^{1/(1-c_1-c_2)}\right\rceil$ where c_1 and c_2 are two arbitrary constants that $c_1 + c_2 < 1$. Note that $\kappa n^{c_1}d^{c_2} = \frac{\epsilon r}{4}$ by our choice of r. Construct the following instance of Problem 3:

$$\min_{x^{(1)},x^{(2)},\ldots,x^{(r)}\in\mathbb{R}^{3m\times m}} \sum_{i=1}^{r} q(x^{(i)}) \quad \text{s.t.} \quad \sum_{i=1}^{r} f(x^{(i)}) \le \bar{\eta}, \tag{17}$$

where $\bar{\eta} \in [\mu\cdot 3m\cdot\ell_1(\tilde{t}) + \nu\cdot m\cdot\ell_2(\tilde{t}), \mu\cdot 3m\cdot\ell_1(\tilde{t}) + \nu\cdot m\cdot\ell_2(\tilde{t}) + \epsilon r/4)$. Note that the parameters $\mu, \nu, \delta, m, r, d$ and $\bar{\eta}$ are bounded by polynomial functions of n. Computing their values also takes polynomial time. Given the sparsity level $\bar{\eta}$, denote the minimizer of (17) to be $x^{(1)}_{\bar{\eta}}, \cdots x^{(r)}_{\bar{\eta}}$.

3. Let $\Phi_1 = 3m\cdot p(|\tilde{t}|) + \mu\cdot 3m\cdot\ell_1(\tilde{t}) - \frac{\epsilon}{4}$ and $\Phi_2 = \nu\cdot m\cdot\ell_2(\tilde{t})$. We claim that if the 3-partition problem has a solution, then

   (i) $\sum_{i=1}^r f(x^{(i)}_{\bar{\eta}}) + \sum_{i=1}^r q(x^{(i)}_{\bar{\eta}}) \le r\Phi_1 + r\Phi_2 + \frac{\epsilon r}{2}$.

   (ii) $\sum_{i=1}^r f(x^{(i)}_{\bar{\eta}}) \ge \mu\cdot 3m\cdot\ell_1(\tilde{t}) + \nu\cdot m\cdot\ell_2(\tilde{t}) - \epsilon r/4$.

4. Suppose we have an approximate solution satisfying $\sum_{i=1}^r f(\hat{x}^{(i)}_{\bar{\eta}+\kappa n^{c_1}d^{c_2}}) \le \sum_{i=1}^r f(x^{(i)}_{\bar{\eta}})$, we claim that

   (iii) If the approximate solution satisfies

$$\sum_{i=1}^{r} f(\hat{x}^{(i)}_{\bar{\eta}+\kappa n^{c_1}d^{c_2}}) + \sum_{i=1}^{r} q(\hat{x}^{(i)}_{\bar{\eta}+\kappa n^{c_1}d^{c_2}}) \le r\Phi_1 + r\Phi_2 + 4\kappa n^{c_1}d^{c_2}, \tag{18}$$

   we can choose the index k such that $\Phi_1+\Phi_2+\epsilon \ge f(\hat{x}^{(k)}_{\bar{\eta}+\kappa n^{c_1}d^{c_2}}) \ge \Phi_1+\Phi_2$ and obtain a feasible assignment: If $\left(\hat{x}^{(k)}_{\bar{\eta}+\kappa n^{c_1}d^{c_2}}\right)_{ij} \in B(\tilde{t},\delta)$, we assign number i to subset j. If the approximate solutions do not satisfy (18), the 3-partition problem has no feasible solution.

We have constructed a polynomial reduction from the 3-partition problem to finding an approximate solution to Problem 3. We then prove that the reduction works. We begin

with the proof of (i). By the condition that the 3-partition problem has a solution, we construct $x^* \in \mathbb{R}^{3m\times m}$ as follows. If number i is assigned to subset j, let $x^*_{ij} = \tilde{t}$ and $x^*_{ik} = 0$ otherwise. We can see that $x^{(1)} = \cdots = x^{(r)} = x^*$ satisfy the constraint of (14) with error tolerance $\bar{\eta}$ and $\sum_{i=1}^r f(x^{(i)}_{\bar{\eta}}) \le \mu \cdot 3m \cdot \ell_1(\tilde{t}) + \nu \cdot m \cdot \ell_2(\tilde{t}) + \epsilon r/4$. Thus,

$$\begin{aligned}\sum_{i=1}^r f(x^{(i)}_{\bar{\eta}}) + \sum_{i=1}^r q(x^{(i)}_{\bar{\eta}}) &\le r \cdot q(x^*) + \mu \cdot 3m \cdot \ell_1(\tilde{t}) + \nu \cdot m \cdot \ell_2(\tilde{t}) + \epsilon r/4 \\ &\le r \cdot 3m \cdot \Big(g_\mu(t^*) - \frac{\epsilon}{12m}\Big) + r\Phi_2 + \frac{\epsilon r}{2} = r\Phi_1 + r\Phi_2 + \frac{\epsilon r}{2},\end{aligned} \tag{19}$$

where $g_\mu(x) = p(|x|) + \mu \cdot \ell_1(x)$ and the last inequality is due to the choice of $\tilde{t}$ in step 1 of the reduction.

To prove (ii), we just need to notice that if $\sum_{i=1}^r f(x^{(i)}_{\bar{\eta}}) \le \mu\cdot 3m\cdot\ell_1(\tilde{t}) + \nu\cdot m\cdot\ell_2(\tilde{t}) - \epsilon r/4$, we would have $\sum_{i=1}^r f(x^{(i)}_{\bar{\eta}}) + \sum_{i=1}^r q(x^{(i)}_{\bar{\eta}}) < r\Phi_1 + r\Phi_2$ by the same reasoning of equation (16), yielding a contradiction as $\sum_{i=1}^r f(x^{(i)}_{\bar{\eta}}) + \sum_{i=1}^r q(x^{(i)}_{\bar{\eta}})$ will always be greater than or equal to $r\Phi_1 + r\Phi_2$.

Now we prove (iii). To prove the first half of the claim, we only need to use $4\kappa n^{c_1} d^{c_2} = r\epsilon$ and apply the proof of Theorem 1 to get the result. To prove the second half of the claim, assume that we have an algorithm that outputs $\hat{x}_\eta$ satisfying $\sum_{i=1}^r f(\hat{x}^{(i)}_{\eta+\kappa n^{c_1} d^{c_2}}) \le \sum_{i=1}^r f(x^{(i)}_\eta)$. Suppose that the 3-partition problem has a solution. Replacing η by $\bar{\eta}$ gives us

$$\sum_{i=1}^r f(\hat{x}^{(i)}_{\bar{\eta}+\kappa n^{c_1} d^{c_2}}) + \sum_{i=1}^r q(\hat{x}^{(i)}_{\bar{\eta}+\kappa n^{c_1} d^{c_2}}) \le \sum_{i=1}^r f(x^{(i)}_{\bar{\eta}}) + \sum_{i=1}^r q(x^{(i)}_{\bar{\eta}}) + \frac{\epsilon r}{2} \le r\Phi_1 + r\Phi_2 + \epsilon r,$$

where the first inequality is due to (ii) and the second inequality is due to (i). It means that the approximate solutions satisfy (18). To this point, we have finished the proof of Theorem 3. ■

References

E. Amaldi and V. Kann. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209(1):237–260, 1998.

A. Antoniadis and J. Fan. Regularization of wavelet approximations. *Journal of the American Statistical Association*, 96(455):939–967, 2001.

S. Arora, L. Babai, J. Stern, and Z. Sweedy. The hardness of approximate optima in lattices, codes, and systems of linear equations. In *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on*, pages 724–733. IEEE, 1993.

W. Bian and X. Chen. Optimality conditions and complexity for Non-Lipschitz constrained optimization problems. *Preprint*, 2014.

A. M. Bruckstein, D. L. Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review*, 51(1):34–81, 2009.

A. C. Cameron and P. K. Trivedi. *Regression analysis of count data*, volume 53. Cambridge university press, 2013.

E. Candes, M. Wakin, and S. Boyd. Enhancing sparsity by reweighted L_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5-6):877–905, 2008.

R. Chartrand. Exact reconstruction of sparse signals via nonconvex minimization. *Signal Processing Letters, IEEE*, 14(10):707–710, 2007.

X. Chen, D. Ge, Z. Wang, and Y. Ye. Complexity of unconstrained $L_2 - L_p$ minimization. *Mathematical Programming*, 143(1-2):371–383, 2014.

Y. Chen and M. Wang. Hardness of approximation for sparse optimization with L_0 norm. *Technical Report*, 2016.

Y. Chen, D. Ge, M. Wang, Z. Wang, Y. Ye, and H. Yin. Strong NP-hardness for sparse optimization with concave penalty functions. *ICML*, 2017.

G. Davis, S. Mallat, and M. Avellaneda. Adaptive greedy approximations. *Constructive approximation*, 13(1):57–98, 1997.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

J. Fan and J. Lv. A selective overview of variable selection in high dimensional feature space. *Statistica Sinica*, 20(1):101–148, 2010.

J. Fan, L. Xue, and H. Zou. Strong oracle optimality of folded concave penalized estimation. *The Annals of Statistics*, 42(3):819–849, 2014.

J. Fan, H. Liu, Q. Sun, and T. Zhang. TAC for sparse learning: Simultaneous control of algorithmic complexity and statistical error. *arXiv preprint arXiv:1507.01037*, 2015.

E. X. Fang, H. Liu, and M. Wang. Blessing of massive scale: Spatial graphical model inference with a total cardinality constraint. *working paper*, 2015.

D. Foster, H. Karloff, and J. Thaler. Variable selection is hard. In *COLT*, pages 696–709, 2015.

L. E. Frank and J. H. Friedman. A statistical view of some chemometrics regression tools. *Technometrics*, 35(2):109–135, 1993.

M. R. Garey and D. S. Johnson. "Strong" NP-Completeness results: Motivation, examples, and implications. *Journal of the ACM*, 25(3):499–508, 1978.

D. Ge, Z. Wang, Y. Ye, and H. Yin. Strong NP-hardness result for regularized L_q-minimization problems with concave penalty functions. *arXiv preprint arXiv:1501.00622*, 2015.

P. J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

X. Huo and J. Chen. Complexity of penalized likelihood estimation. *Journal of Statistical Computation and Simulation*, 80(7):747–759, 2010.

P.-L. Loh and M. J. Wainwright. Regularized M-estimators with nonconvexity: Statistical and algorithmic theory for local optima. In *Advances in Neural Information Processing Systems*, pages 476–484, 2013.

J. Lv and Y. Fan. A unified approach to model selection and sparse recovery using regularized least squares. *The Annals of Statistics*, 37(6A):3498–3528, 2009.

P. McCullagh. Generalized linear models. *European Journal of Operational Research*, 16 (3):285–292, 1984.

B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

X. Shen, W. Pan, and Y. Zhu. Likelihood-based selection and sharp parameter estimation. *Journal of the American Statistical Association*, 107(497):223–232, 2012.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

V. V. Vazirani. *Approximation Algorithms*. Springer Science & Business Media, 2001.

L. Xue, H. Zou, T. Cai, et al. Nonconcave penalized composite conditional likelihood estimation of sparse ising models. *The Annals of Statistics*, 40(3):1403–1429, 2012.

C.-H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 38(2):894–942, 2010a.

T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11:1081–1107, 2010b.

Y. Zhang, M. J. Wainwright, and M. I. Jordan. Lower bounds on the performance of polynomial-time algorithms for sparse linear regression. In *COLT*, 2014.

Journal of Machine Learning Research 20 (2019) 1-40 Submitted 8/17; Revised 1/19; Published 2/19

A Bootstrap Method for Error Estimation in Randomized Matrix Multiplication

Miles E. Lopes MELOPES@UCDAVIS.EDU
Department of Statistics
University of California at Davis
Davis, CA 95616, USA

Shusen Wang SHUSEN.WANG@STEVENS.EDU
Department of Computer Science
Stevens Institute of Technology
Hoboken, NJ 07030, USA

Michael W. Mahoney MMAHONEY@STAT.BERKELEY.EDU
International Computer Science Institute and Department of Statistics
University of California at Berkeley
Berkeley, CA 94720, USA

Editor: Hui Zou

Abstract

In recent years, randomized methods for numerical linear algebra have received growing interest as a general approach to large-scale problems. Typically, the essential ingredient of these methods is some form of randomized dimension reduction, which accelerates computations, but also creates random approximation error. In this way, the dimension reduction step encodes a tradeoff between cost and accuracy. However, the exact numerical relationship between cost and accuracy is typically unknown, and consequently, it may be difficult for the user to precisely know (1) how accurate a given solution is, or (2) how much computation is needed to achieve a given level of accuracy. In the current paper, we study randomized matrix multiplication (sketching) as a prototype setting for addressing these general problems. As a solution, we develop a bootstrap method for *directly estimating* the accuracy as a function of the reduced dimension (as opposed to deriving worst-case bounds on the accuracy in terms of the reduced dimension). From a computational standpoint, the proposed method does not substantially increase the cost of standard sketching methods, and this is made possible by an "extrapolation" technique. In addition, we provide both theoretical and empirical results to demonstrate the effectiveness of the proposed method.

Keywords: matrix sketching, randomized matrix multiplication, bootstrap methods

1. Introduction

The development of randomized numerical linear algebra (RNLA or RandNLA) has led to a variety of efficient methods for solving large-scale matrix problems, such as matrix multiplication, least-squares approximation, and low-rank matrix factorization, among others (Halko et al., 2011; Mahoney, 2011; Woodruff, 2014; Drineas and Mahoney, 2016). A general feature of these methods is that they apply some form of randomized dimension reduction to an input matrix, which reduces the cost of subsequent computations. In exchange for the reduced cost, the randomization leads to some error in the resulting solution, and consequently, there is a tradeoff between cost and accuracy.

For many canonical matrix problems, the relationship between cost and accuracy has been the focus of a growing body of theoretical work, and the literature provides many performance guarantees for RNLA methods. In general, these guarantees offer a good qualitative description of how the accuracy depends on factors such as problem size, number of iterations, condition numbers, and so on. Yet, it is also the case that such guarantees tend to be overly pessimistic for any particular problem instance — often because the guarantees are formulated to hold in the worst case among a large class of possible inputs. Likewise, it is often impractical to use such guarantees to determine precisely how accurate a given solution is, or precisely how much computation is needed to achieve a desired level of accuracy.

In light of this situation, it is of interest to develop efficient methods for estimating the exact relationship between the cost and accuracy of RNLA methods on a *problem-specific basis*. Since the literature has been somewhat quiet on this general question, the aim of this paper is to analyze randomized matrix multiplication as a prototype setting, and propose an approach that may be pursued more broadly. (Extensions are discussed at the end of the paper in Section 6.)

1.1. Randomized matrix multiplication

To describe our problem setting, we briefly review the rudiments of randomized matrix multiplication, which is often known as matrix sketching (Drineas et al., 2006a; Mahoney, 2011; Woodruff, 2014). If $\mathbf{A} \in \mathbb{R}^{n\times d}$ and $\mathbf{B} \in \mathbb{R}^{n\times d'}$ are fixed input matrices, then sketching methods are commonly used to approximate $\mathbf{A}^T\mathbf{B}$ in the the regime where $\max\{d, d'\} \ll n$. For instance, this regime corresponds to "big data" applications where $\mathbf{A}$ and $\mathbf{B}$ are data matrices with very large numbers of observations.

As a way of reducing the cost of ordinary matrix multiplication, the main idea of sketching is to compute the product $\tilde{\mathbf{A}}^T\tilde{\mathbf{B}}$ of smaller matrices $\tilde{\mathbf{A}} \in \mathbb{R}^{t\times d}$ and $\tilde{\mathbf{B}} \in \mathbb{R}^{t\times d'}$, for some choice of $t \ll n$. These smaller matrices are referred to as "sketches", and they are generated randomly according to

$$\tilde{\mathbf{A}} := \mathbf{S}\mathbf{A} \quad \text{and} \quad \tilde{\mathbf{B}} := \mathbf{S}\mathbf{B}, \tag{1}$$

where $\mathbf{S} \in \mathbb{R}^{t\times n}$ is a random "sketching matrix" satisfying the condition

$$\mathbb{E}[\mathbf{S}^T\mathbf{S}] = \mathbf{I}_n, \tag{2}$$

with $\mathbf{I}_n$ being the identity matrix. In particular, the relation (2) implies that the sketched product is an unbiased estimate, $\mathbb{E}[\tilde{\mathbf{A}}^T\tilde{\mathbf{B}}] = \mathbf{A}^T\mathbf{B}$. Most commonly, the matrix $\mathbf{S}$ can be interpreted as acting on $\mathbf{A}$ and $\mathbf{B}$ by sampling their rows, or by randomly projecting their columns. In Section 2, we describe some popular examples of sketching matrices to be considered in our analysis.

1.2. Problem formulation

When sketching is implemented, the choice of the sketch size t plays a central role, since it directly controls the relationship between cost and accuracy. If t is small, then the sketched

©2019 Miles E. Lopes, Shusen Wang, and Michael W. Mahoney.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-451.html.

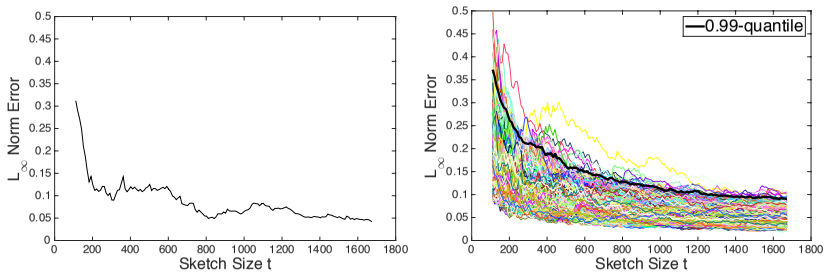


Figure 1: **Left panel:** The curve shows how ε_t fluctuates with varying sketch size t, as rows are added to $\mathbf{S}$, with $\mathbf{A}$ and $\mathbf{B}$ held fixed. (Each row of $\mathbf{A} \in \mathbb{R}^{8,124\times 112}$ is a feature vector of the Mushroom dataset (Frank and Asuncion, 2010), and we set $\mathbf{B} = \mathbf{A}$.) The rows of $\mathbf{S}$ were generated randomly from a Gaussian distribution (see Section 2), and the matrix $\mathbf{A}$ was scaled so that $\|\mathbf{A}^T\mathbf{A}\|_\infty = 1$. **Right panel:** There are 1,000 colored curves, each arising from a repetition of the simulation in the left panel. The thick black curve represents $q_{0.99}(t)$.

product $\tilde{\mathbf{A}}^T\tilde{\mathbf{B}}$ may be computed quickly, but it is unlikely to be a good approximation to $\mathbf{A}^T\mathbf{B}$. Conversely, if t is large, then the sketched product is more expensive to compute, but it is more likely to be accurate. For this reason, we will parameterize the relationship between cost and accuracy in terms of t.

Conventionally, the error of an approximate matrix product is measured with a norm, and in particular, we will consider error as measured by the ℓ_∞-norm,

$$\varepsilon_t \;:=\; \left\|\mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B} - \mathbf{A}^T\mathbf{B}\right\|_\infty, \tag{3}$$

where $\|\mathbf{C}\|_\infty := \max_{i,j}|c_{ij}|$ for a matrix $\mathbf{C} = [c_{ij}]$. (Further background on analysis of ℓ_∞-norm or entry-wise error for matrix multiplication may be found in (Higham, 2002; Drineas et al., 2006a; Demmel et al., 2007; Pagh, 2013), among others.) In the context of sketching, it is crucial to note that ε_t is a random variable, due to the randomness in $\mathbf{S}$. Consequently, it is natural to study the quantiles of ε_t, because they specify the *tightest possible bounds* on ε_t that hold with a prescribed probability. More specifically, for any $\alpha \in (0,1)$, the $(1-\alpha)$-quantile of ε_t is defined as

$$q_{1-\alpha}(t) \;:=\; \inf\left\{q \in [0,\infty) \,\middle|\, \mathbb{P}\big(\varepsilon_t \le q\big) \;\ge\; 1-\alpha\right\}. \tag{4}$$

For example, the quantity $q_{0.99}(t)$ is the tightest upper bound on ε_t that holds with probability at least 0.99. Hence, for any fixed α, the function $q_{1-\alpha}(t)$ represents a precise *tradeoff curve* for relating cost and accuracy. Moreover, the function $q_{1-\alpha}(t)$ is specific to the input matrices $\mathbf{A}$ and $\mathbf{B}$.

To clarify the interpretation of $q_{1-\alpha}(t)$, it is helpful to plot the fluctuations of ε_t. In the left panel of Figure 1, we illustrate a simulation where randomly generated rows are incrementally added to a sketching matrix $\mathbf{S}$, with $\mathbf{A}$ and $\mathbf{B}$ held fixed. Each time a row is added to $\mathbf{S}$, the sketch size t increases by 1, and we plot the corresponding value of ε_t as t ranges from 100 to 1,700. (Note that the user is typically unable to observe such a curve in practice.) In the right panel, we display 1,000 repetitions of the simulation, with each colored curve corresponding to one repetition. (The variation is due only to the different draws of $\mathbf{S}$.) In particular, the function $q_{0.99}(t)$ is represented by the thick black curve, delineating the top 1% of the colored curves at each value of t.

In essence, the right panel of Figure 1 shows that if the user had knowledge of the (unknown) function $q_{1-\alpha}(t)$, then two important purposes could be served. First, for any fixed value t, the user would have a sharp problem-specific bound on ε_t. Second, for any fixed error tolerance ϵ, the user could select t so that that "just enough" computation is spent in order to achieve $\varepsilon_t \le \epsilon$ with probability at least $1-\alpha$.

The estimation problem. The challenge we face is that a naive computation of $q_{1-\alpha}(t)$ by generating samples of ε_t would defeat the purpose of sketching. Indeed, generating samples of ε_t by brute force would require running the sketching method many times, and it would also require computing the entire product $\mathbf{A}^T\mathbf{B}$. Consequently, the technical problem of interest is to develop an efficient way to estimate $q_{1-\alpha}(t)$, without adding much cost to a *single run* of the sketching method.

1.3. Contributions

From a conceptual standpoint, the main novelty of our work is that it bridges two sets of ideas that are ordinarily studied in distinct communities. Namely, we apply the statistical technique of bootstrapping to enhance algorithms for numerical linear algebra. To some extent, this pairing of ideas might seem counterintuitive, since bootstrap methods are sometimes labeled as "computationally intensive", but it will turn out that the cost of bootstrapping can be managed in our context. Another reason our approach is novel is that we use the bootstrap to quantify error in the output of a randomized algorithm, rather than for the usual purpose of quantifying uncertainty arising from data. In this way, our approach harnesses the versatility of bootstrap methods, and we hope that our results in the "use case" of matrix multiplication will encourage broader applications of bootstrap methods in randomized computations. (See also Section 6, and note that in concurrent work, we have pursued similar approaches in the contexts of randomized least-squares and classification algorithms (Lopes et al., 2018b; Lopes, 2019).)

From a technical standpoint, our main contributions are a method for estimating the function $q_{1-\alpha}(t)$, as well as theoretical performance guarantees. Computationally, the proposed method is efficient in the sense that its cost is comparable to a single run of standard sketching methods (see Section 2). This efficiency is made possible by an "extrapolation" technique, which allows us to bootstrap small "initial" sketches with t_0 rows, and inexpensively estimate $q_{1-\alpha}(t)$ at larger values $t \gg t_0$. The empirical performance of the extrapolation technique is also quite encouraging, as discussed in Section 5. Lastly, with regard to theoretical analysis, our proofs circumvent some technical restrictions occurring in the analysis of related bootstrap methods in the statistics literature.

1.4. Related work

Several works have considered the problem of error estimation for randomized matrix computations—mostly in the context of low-rank approximation (Woolfe et al., 2008; Liberty et al., 2007; Halko et al., 2011), least squares (Lopes et al., 2018b), or matrix multiplication (Ar et al., 1993; Sarlós, 2006). With attention to matrix multiplication, the latter two papers offer methods for estimating high-probability bounds on the error $\eta_t := \|\tilde{\mathbf{A}}^T\tilde{\mathbf{B}} - \mathbf{A}^T\mathbf{B}\|$, where $\|\cdot\|$ is either the maximum absolute row sum norm, or the Frobenius norm. At a high level, all of the mentioned papers rely on a common technique, which is to randomly generate a sequence of "test-vectors", say $\mathbf{v}_1, \mathbf{v}_2, \ldots$, and then use the matrix-vector products $\mathbf{w}_i := \tilde{\mathbf{A}}^T\tilde{\mathbf{B}}\mathbf{v}_i - \mathbf{A}^T(\mathbf{B}\mathbf{v}_i)$ to derive an estimated bound, say $\hat{\eta}_t$, for η_t. The origin of this technique may be traced to the classical works (Dixon, 1983; Freivalds, 1979).

Our approach differs from the "test-vector approach" in some essential ways. One difference arises because the bounds on $\hat{\eta}_t$ are generally constructed from the vectors $\{\mathbf{w}_i\}$ using conservative inequalities. By contrast, our approach avoids this conservativeness by *directly estimating* $q_{1-\alpha}(t)$, which is an optimal bound on ε_t in the sense of equation (4).

A second difference deals with computational demands. For example, in order to compute the vectors $\{\mathbf{w}_i\}$ in the test-vector approach, it is necessary to access the full matrices $\mathbf{A}$ and $\mathbf{B}$. On the other hand, our method does not encounter this difficulty, because it only requires access to the much smaller sketches $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$. Also, in the test-vector approach, the cost to compute each vector $\mathbf{w}_i$ is proportional to the large dimension n, while the cost to compute $\hat{q}_{1-\alpha}(t)$ with our method is *independent* of n. Finally, the test-vector approach can only be used to check if the product $\tilde{\mathbf{A}}^T\tilde{\mathbf{B}}$ is accurate after it has been computed, whereas our approach can be used to dynamically "predict" an appropriate sketch size t from a small "initial" sketching matrix (see Section 3.3).

With regard to the statistics literature, our work builds upon a line of research dealing with "multiplier bootstrap methods" in high-dimensional problems (Chernozhukov et al., 2013, 2014, 2017). Such methods are well-suited to approximating the distributions of statistics such as $\|\bar{\mathbf{x}}\|_\infty$, where $\bar{\mathbf{x}} \in \mathbb{R}^p$ denotes the sample average of n independent mean-zero vectors, with $n \ll p$. More recently, this approach has been substantially extended to other "max type" statistics arising from sample covariance matrices (Chang et al., 2016; Chen, 2018). Nevertheless, the strong results in these works do not readily translate to our context, either because the statistics are substantially different from the ℓ_∞-norm (Chang et al., 2016), or because of technical assumptions (Chen, 2018). For instance, if the results in the latter work are applied to a sample covariance matrix of the form $\frac{1}{n}\sum_{i=1}^n(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top$, where $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ are mean-zero i.i.d. vectors, with $\mathbf{x}_1 = (X_{11}, \ldots, X_{1p})$, then it is necessary to make assumptions such as $\min_{j,k} \mathsf{var}(X_{1j}X_{1k}) \geq c$, for some constant $c > 0$. As this relates to the sketching context, note that the sketched product may be written as $\tilde{\mathbf{A}}^T\tilde{\mathbf{B}} = \frac{1}{t}\sum_{i=1}^t \mathbf{A}^T\mathbf{s}_i\mathbf{s}_i^T\mathbf{B}$, where $\mathbf{s}_1, \ldots, \mathbf{s}_n \in \mathbb{R}^t$ are the rows of $\sqrt{t}\mathbf{S}$. It follows that analogous variance assumptions would lead to conditions on the matrices $\mathbf{A}$ and $\mathbf{B}$ that could be violated if any column of $\mathbf{A}$ or $\mathbf{B}$ has many small entries, or is sparse. By contrast, our results do not rely on such variance assumptions, and we allow the matrices $\mathbf{A}$ and $\mathbf{B}$ to be unrestricted.

At a more technical level, the ability to avoid restrictions on $\mathbf{A}$ and $\mathbf{B}$ comes from our use of the Lévy-Prohorov metric for distributional approximations — which differs from the Kolmogorov metric that has been predominantly used in previous works on multiplier bootstrap methods. More specifically, analyses based on the Kolmogorov metric typically rely on "anti-concentration inequalities" (Chernozhukov et al., 2013, 2015), which ultimately lead to the mentioned variance assumptions. On the other hand, our approach based on the Lévy-Prohorov metric does not require the use of anti-concentration inequalities. Finally it should be mentioned that the techniques used to control the LP metric are related to those that have been developed for bootstrap approximations via coupling inequalities as in Chernozhukov et al. (2016).

Outline. This paper is organized as follows. Section 2 introduces some technical background. Section 3 describes the proposed bootstrap algorithm. Section 4 establishes the main theoretical results, and then numerical performance is illustrated in Section 5. Lastly, conclusions and extensions of the method are presented in Section 6, and all proofs are given in the appendices.

2. Preliminaries

Notation and terminology. The set $\{1, \ldots, n\}$ is denoted as $[n]$. The ith standard basis vector is denoted as $\mathbf{e}_i$. If $\mathbf{C} = [c_{ij}]$ is a real matrix, then $\|\mathbf{C}\|_F = (\sum_{i,j} c_{ij}^2)^{1/2}$ is the Frobenius norm, and $\|\mathbf{C}\|_2$ is the spectral norm (maximum singular value). If X is a random variable and $p \geq 1$, we write $\|X\|_p = (\mathbb{E}[|X|^p])^{1/p}$ for the usual L_p norm. If $\psi : [0, \infty) \to [0, \infty)$ is a non-decreasing convex function with $\psi(0) = 0$, then the ψ-Orlicz norm of X is defined as $\|X\|_\psi := \inf\{r > 0 \mid \mathbb{E}[\psi(|X|/r)] \leq 1\}$. In particular, we define $\psi_p(x) := \exp(x^p) - 1$ for $p \geq 1$, and we say that X is sub-Gaussian when $\|X\|_{\psi_2} < \infty$, or sub-exponential when $\|X\|_{\psi_1} < \infty$. In Appendix F, Lemma 9 summarizes the facts about Orlicz norms that will be used.

We will use c to denote a positive absolute constant that may change from line to line. The matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{S}$ are viewed as lying in a sequence of matrices indexed by the tuple (d, d', t, n). For a pair of generic functions f and g, we write $f(d, d', t, n) \lesssim g(d, d', t, n)$ when there is a positive absolute constant c so that $f(d, d', t, n) \leq c\, g(d, d', t, n)$ holds for all large values of d, d', t, and n. Furthermore, if a and b are two quantities that satisfy both $a \lesssim b$ and $b \lesssim a$, then we write $a \asymp b$. Lastly, we do not use the symbols $\lesssim$ or $\asymp$ when relating random variables.

Examples of sketching matrices. Our theoretical results will deal with three common types of sketching matrices, reviewed below.

- *Row sampling.* If $(p_1, \ldots, p_n)$ is a probability vector, then $\mathbf{S} \in \mathbb{R}^{t\times n}$ can be constructed by sampling its rows i.i.d. from the set $\{\frac{1}{\sqrt{tp_1}}\mathbf{e}_1, \ldots, \frac{1}{\sqrt{tp_n}}\mathbf{e}_n\} \subset \mathbb{R}^n$, where the vector $\frac{1}{\sqrt{tp_i}}\mathbf{e}_i$ is selected with probability p_i. Some of the most well known choices for the sampling probabilities include *uniform sampling*, with $p_i \equiv 1/n$, *length sampling* (Drineas et al., 2006a; Magen and Zouzias, 2011), with

$$p_i = \frac{\|\mathbf{e}_i^T\mathbf{A}\|_2\|\mathbf{e}_i^T\mathbf{B}\|_2}{\sum_{j=1}^n \|\mathbf{e}_j^T\mathbf{A}\|_2\|\mathbf{e}_j^T\mathbf{B}\|_2}, \tag{5}$$

and *leverage score sampling*, for which further background may be found in the papers (Drineas et al., 2006b, 2008, 2012).

- *Sub-Gaussian projection.* Gaussian projection is the most well-known random projection method, and is sometimes referred to as the Johnson-Lindenstrauss (JL) transform (Johnson and Lindenstrauss, 1984). In detail, if $\mathbf{G} \in \mathbb{R}^{t\times n}$ is a standard Gaussian matrix, with entries that are i.i.d. samples from $\mathcal{N}(0,1)$, then $\mathbf{S} = \frac{1}{\sqrt{t}}\mathbf{G}$ is a Gaussian projection matrix. More generally, the entries of $\mathbf{G}$ can be drawn i.i.d. from a zero-mean sub-Gaussian distribution, which often leads to similar performance characteristics in RNLA applications.

- *Subsampled randomized Hadamard transform (SRHT).* Let n be a power of 2, and define the Walsh-Hadamard matrix $\mathbf{H}_n$ recursively*

$$\mathbf{H}_n := \begin{pmatrix} \mathbf{H}_{n/2} & \mathbf{H}_{n/2} \\ \mathbf{H}_{n/2} & -\mathbf{H}_{n/2} \end{pmatrix} \quad \text{with} \quad \mathbf{H}_2 := \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

  Next, let $\mathbf{D}_n^\circ \in \mathbb{R}^{n\times n}$ be random diagonal matrix with independent ± 1 Rademacher variables along the diagonal, and let $\mathbf{P} \in \mathbb{R}^{t\times n}$ have rows uniformly sampled from $\{\frac{1}{\sqrt{t/n}}\mathbf{e}_1, \ldots, \frac{1}{\sqrt{t/n}}\mathbf{e}_n\}$. Then, the $t \times n$ matrix

$$\mathbf{S} = \mathbf{P}(\tfrac{1}{\sqrt{n}}\mathbf{H}_n)\mathbf{D}_n^\circ \tag{6}$$

  is called an SRHT matrix. This type of sketching matrix was introduced in the seminal paper (Ailon and Chazelle, 2006), and additional details regarding implementation may be found in the papers (Drineas et al., 2011; Wang, 2015). (The factor $\frac{1}{\sqrt{n}}$ is used so that $\frac{1}{\sqrt{n}}\mathbf{H}_n$ is an orthogonal matrix.) An important property of SRHT matrices is that they can be multiplied with any $n \times d$ matrix in $\mathcal{O}(n \cdot d \cdot \log t)$ time (Ailon and Liberty, 2009), which is faster than the $\mathcal{O}(n \cdot d \cdot t)$ time usually required for a dense sketching matrix.

3. Methodology

Before presenting our method in algorithmic form, we first explain the underlying intuition.

3.1. Intuition for multiplier bootstrap method

If the row vectors of $\sqrt{t}\mathbf{S}$ are denoted $\mathbf{s}_1, \ldots, \mathbf{s}_t \in \mathbb{R}^n$, then $\mathbf{S}^T\mathbf{S}$ may be conveniently expressed as a sample average

$$\mathbf{S}^T\mathbf{S} = \tfrac{1}{t}\textstyle\sum_{i=1}^t \mathbf{s}_i\mathbf{s}_i^T. \tag{7}$$

For row sampling, Gaussian projection, and SRHT, these row vectors satisfy $\mathbb{E}[\mathbf{s}_i\mathbf{s}_i^T] = \mathbf{I}_n$. Consequently, if we define the random $d \times d'$ rank-1 (dyad) matrix

$$\mathbf{D}_i \;=\; \mathbf{A}^T\mathbf{s}_i\mathbf{s}_i^T\mathbf{B}, \tag{8}$$

*The restriction that n is a power of 2 can be relaxed with variants of SRHT matrices (Avron et al., 2010; Boutsidis and Gittens, 2013).

then $\mathbb{E}[\mathbf{D}_i] = \mathbf{A}^T\mathbf{B}$, and it follows that the difference between the sketched and unsketched products can be viewed as a sample average of zero-mean random matrices

$$\mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B} - \mathbf{A}^T\mathbf{B} \;=\; \tfrac{1}{t}\textstyle\sum_{i=1}^t(\mathbf{D}_i - \mathbf{A}^T\mathbf{B}). \tag{9}$$

Furthermore, in the cases of length sampling and Gaussian projection, the matrices $\mathbf{D}_1, \ldots, \mathbf{D}_t$ are independent, and in the case of SRHT sketches, these matrices are "nearly" independent. So, in light of the central limit theorem, it is natural to suspect that the random matrix (9) will be well-approximated (in distribution) by a matrix with Gaussian entries. In particular, if we examine the (j_1, j_2) entry, then we may expect that $\mathbf{e}_{j_1}^T(\mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B} - \mathbf{A}^T\mathbf{B})\mathbf{e}_{j_2}$ will approximately follow the distribution $\mathcal{N}(0, \frac{1}{t}\sigma_{j_1,j_2}^2)$, where the unknown parameter σ_{j_1,j_2}^2 can be estimated with

$$\hat{\sigma}_{j_1,j_2}^2 \;:=\; \tfrac{1}{t}\textstyle\sum_{i=1}^t \big(\mathbf{e}_{j_1}^T(\mathbf{D}_i - \mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B})\mathbf{e}_{j_2}\big)^2.$$

Based on these considerations, the idea of the proposed bootstrap method is to generate a random matrix whose (j_1, j_2) entry is sampled from $\mathcal{N}(0, \frac{1}{t}\hat{\sigma}_{j_1,j_2}^2)$. It turns out that an efficient way of generating such a matrix is to sample i.i.d. random variables $\xi_1, \ldots, \xi_t \sim \mathcal{N}(0,1)$, independent of $\mathbf{S}$, and then compute

$$\tfrac{1}{t}\textstyle\sum_{i=1}^t \xi_i(\mathbf{D}_i - \mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B}). \tag{10}$$

In other words, if $\mathbf{S}$ is conditioned upon, then the distribution of the (j_1, j_2) entry of the above matrix is exactly $\mathcal{N}(0, \frac{1}{t}\hat{\sigma}_{j_1,j_2}^2)$.† Hence, if the matrix (10) is viewed as an "approximate sample" of $\mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B} - \mathbf{A}^T\mathbf{B}$, then it is natural to use the ℓ_∞-norm of the matrix (10) as an approximate sample of $\varepsilon_t = \|\mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B} - \mathbf{A}^T\mathbf{B}\|_\infty$. Likewise, if we define the bootstrap sample

$$\varepsilon_t^\star \;:=\; \Big\|\tfrac{1}{t}\textstyle\sum_{i=1}^t \xi_i\Big(\mathbf{D}_i - \mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B}\Big)\Big\|_\infty, \tag{11}$$

then the bootstrap algorithm will generate i.i.d. samples of $\varepsilon_t^\star$, conditionally on $\mathbf{S}$. In turn, the $(1-\alpha)$-quantile of the bootstrap samples, say $\hat{q}_{1-\alpha}(t)$, can be used to estimate $q_{1-\alpha}(t)$.

3.2. Multiplier bootstrap algorithm

We now explain how proposed method can be implemented in just a few lines. This description also reveals the important fact that the algorithm only requires access to the sketches $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$ (rather than the full matrices $\mathbf{A}$ and $\mathbf{B}$). Although the formula for generating samples of $\varepsilon_t^\star$ given below may appear different from equation (11), it is straightforward to check that these are equivalent. Lastly, the choice of the number of bootstrap samples B will be discussed at the end of subsection 3.3.

†It is also possible to show that the *joint* distribution of the entries in the matrix (10) mimics that of $\mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B} - \mathbf{A}^T\mathbf{B}$, but we omit such details to simplify the discussion.

Algorithm 1 (Multiplier bootstrap for ε_t).
Input: *the number of bootstrap samples B, and the sketches $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$.*
For $b = 1, \ldots, B$ **do**

1. *Draw an i.i.d. sample $\xi_1, \ldots, \xi_t$ from $\mathcal{N}(0,1)$, independent of $\mathbf{S}$;*
2. *Compute the bootstrap sample $\varepsilon^\star_{t,b} := \left\| \bar{\xi} \cdot (\tilde{\mathbf{A}}^T\tilde{\mathbf{B}}) - \tilde{\mathbf{A}}^T \mathbf{\Xi} \tilde{\mathbf{B}} \right\|_\infty$, where $\bar{\xi} := \frac{1}{t}\sum_{i=1}^t \xi_i$ and $\mathbf{\Xi} := \mathsf{diag}(\xi_1, \ldots, \xi_t)$.*

Return: $\hat{q}_{1-\alpha}(t) \longleftarrow$ *the $(1-\alpha)$-quantile of the values $\varepsilon^\star_{t,1}, \ldots, \varepsilon^\star_{t,B}$.*

3.3. Saving on computation with extrapolation

In its basic form, the cost of Algorithm 1 is $\mathcal{O}(B \cdot t \cdot d \cdot d')$, which has the favorable property of being independent of the large dimension n. Also, the computation of the samples $\varepsilon^\star_{t,1}, \ldots, \varepsilon^\star_{t,B}$ is embarrassingly parallel, with the cost of each sample being $\mathcal{O}(t \cdot d \cdot d')$. Moreover, due to the way that the quantile $q_{1-\alpha}(t)$ scales with t, it is possible to reduce the cost of Algorithm 1 even further — via the technique of extrapolation (also called Richardson extrapolation) (Sidi, 2003; Brezinski and Zaglia, 2013).

The essential idea of extrapolation is to carry out Algorithm 1 for a modest "initial" sketch size t_0, and then use an initial estimate $\hat{q}_{1-\alpha}(t_0)$ to "look ahead" and predict a larger value t for which $q_{1-\alpha}(t)$ is small enough to satisfy the user's desired level of accuracy. The immediate benefit of this approach is that Algorithm 1 only needs to applied to small "initial versions" of $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$, each with t_0 rows, which reduces the cost of the algorithm to $\mathcal{O}(B \cdot t_0 \cdot d \cdot d')$. Furthermore, this means that if Algorithm 1 is run in parallel, then it is only necessary to communicate copies of the small initial sketching matrices. (To illustrate the small size of the initial sketching matrices, our experiments include several examples where the ratio t_0/n is approximately 1/100 or less.)

From a theoretical viewpoint, our use of extrapolation is based on the approximation $q_{1-\alpha}(t) \approx \frac{\kappa}{\sqrt{t}}$, where t is sufficiently large, and $\kappa = \kappa(\mathbf{A}, \mathbf{B}, \alpha)$ is an unknown number. A formal justification for this approximation can be made using Proposition 3 in Appendix A, but it is simpler to give an intuitive explanation here. Recall from Section 3.1 that as t becomes large, the (j_1, j_2) entry $[\tilde{\mathbf{A}}^T\tilde{\mathbf{B}} - \mathbf{A}^T\mathbf{B}]_{j_1,j_2}$ should be well-approximated in distribution by a Gaussian random variable of the form $\frac{1}{\sqrt{t}} G_{j_1,j_2}$. In turn, this suggests that ε_t should be well-approximated in distribution by $\frac{1}{\sqrt{t}} \max_{j_1,j_2} |G_{j_1,j_2}|$, which has quantiles that are proportional to $\frac{1}{\sqrt{t}}$.

In order to take advantage of the theoretical scaling $q_{1-\alpha}(t) \approx \frac{\kappa}{\sqrt{t}}$, we may use Algorithm 1 to compute $\hat{q}_{1-\alpha}(t_0)$ with an initial sketch size t_0, and then approximate the value $q_{1-\alpha}(t)$ for $t \gg t_0$ with the following extrapolated estimator

$$\hat{q}^{\,\mathrm{ext}}_{1-\alpha}(t) := \tfrac{\sqrt{t_0}}{\sqrt{t}} \hat{q}_{1-\alpha}(t_0). \tag{12}$$

Hence, if the user would like to determine a sketch size t so that $q_{1-\alpha}(t) \leq \epsilon$, for some tolerance ϵ, then t should be selected so that $\hat{q}^{\,\mathrm{ext}}_{1-\alpha}(t) \leq \epsilon$, which is equivalent to

$$t \geq \left(\tfrac{\sqrt{t_0}}{\epsilon}\, \hat{q}_{1-\alpha}(t_0) \right)^2. \tag{13}$$

In our experiments in Section 5, we illustrate some examples where an accurate estimate of $q_{1-\alpha}(t)$ at $t = 10{,}000$ can be obtained from the rule (13) using an initial sketch size $t_0 \approx 500$, yielding a roughly 20-fold speedup on the basic version of Algorithm 1.

Comparison with the cost of sketching. Given that the purpose of Algorithm 1 is to enhance sketching methods, it is important to understand how the added cost of the bootstrap compares to the cost of running sketching methods in the standard way. As a point of reference, we compare with the cost of computing $\mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B}$ when $\mathbf{S}$ is chosen to be an SRHT matrix, since this is one of the most efficient sketching methods. If we temporarily assume for simplicity that $\mathbf{A}$ and $\mathbf{B}$ are both of size $n \times d$, then it follows from Section 2 that computing $\mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B}$ has a cost of order $\mathcal{O}(t \cdot d^2 + n \cdot d \cdot \log(t))$. Meanwhile, the cost of running Algorithm 1 with the extrapolation speedup based on an initial sketch size t_0 is $\mathcal{O}(B \cdot t_0 \cdot d^2)$. Consequently, the extra cost of the bootstrap does not exceed the stated cost of sketching when the number of bootstrap samples satisfies

$$B = \mathcal{O}(\tfrac{t}{t_0} + \tfrac{n \log(t)}{d\, t_0}), \tag{14}$$

and in fact, this could be improved further if parallelization of Algorithm 1 is taken into account. It is also important to note that rather small values of B are shown to work well in our experiments, such as $B = 20$. Hence, as long t_0 remains fairly small compared to t, then the condition (14) may be expected to hold, and this is borne out in our experiments. The same reasoning also applies when $n \log(t) \gg d \cdot t_0$, which conforms with the fact that sketching methods are intended to handle situations where n is very large.

3.4. Relation with the non-parametric bootstrap

For readers who are more familiar with the "non-parametric bootstrap" (based on sampling with replacement), the purpose of this short subsection is to explain the relationship with the multiplier bootstrap in Algorithm 1. Indeed, an understanding of this relationship may be helpful, since the non-parametric bootstrap might be viewed as more intuitive, and perhaps easier to generalize to more complex situations. However, it turns out that Algorithm 1 is technically more convenient to analyze, and that is why the paper focuses primarily on Algorithm 1. Meanwhile, from a practical point of view, there is little difference between the two approaches, since both have the same order of computational cost, and in our experience, we have observed essentially the same performance in simulations. Also, the extrapolation technique can be applied to both algorithms in the same way.

To spell out the connection, the only place where Algorithm 1 needs to be changed is in step 1. Rather than choosing the multiplier variables $\xi_1, \ldots, \xi_t$ to be i.i.d. $\mathcal{N}(0,1)$ as in Algorithm 1, the non-parametric bootstrap chooses $\xi_i = \zeta_i - 1$, where $(\zeta_1, \ldots, \zeta_t)$ is a sample from a multinomial distribution, based on tossing t balls into t equally likely bins, where ζ_i is the number of balls in bin i. Hence, the mean and variance of each ξ_i are nearly the same as before, with $\mathbb{E}[\xi_i] = 0$ and $\mathsf{var}(\xi_i) = 1 - 1/t$, but the variables $\xi_1, \ldots, \xi_t$ are no longer independent.

From a more algorithmic viewpoint, it is simple to check that the choice of $\xi_1, \ldots, \xi_t$ based on the multinomial distribution is equivalent to sampling with replacement from the rows of $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$. The underlying intuition for this approach is based on the fact that for many types of sketching matrices, the rows of $\mathbf{S}$ are i.i.d., which makes the rows of $\tilde{\mathbf{A}}$ i.i.d.,

and likewise for $\tilde{\mathbf{B}}$. Hence, if $\mathbf{S}$ is conditioned upon, then sampling with replacement from the rows of $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$ imitates the random mechanism that originally generated $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$.

Algorithm 2 (Non-parametric bootstrap for ε_t).

Input: *the number of samples B, and the sketches $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$.*

For $b = 1, \ldots, B$ **do**

1. *Draw a vector $(i_1, \ldots, i_t)$ by sampling t numbers with replacement from $\{1, \ldots, t\}$.*
2. *Form matrices $\tilde{\mathbf{A}}^* \in \mathbb{R}^{t \times d}$ and $\tilde{\mathbf{B}}^* \in \mathbb{R}^{t \times d'}$ by selecting (respectively) the rows from $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$ that are indexed by $(i_1, \ldots, i_t)$.*
3. *Compute the bootstrap sample $\varepsilon^*_{t,b} := \big\|(\tilde{\mathbf{A}}^*)^T(\tilde{\mathbf{B}}^*) - \tilde{\mathbf{A}}^T\tilde{\mathbf{B}}\big\|_\infty$.*

Return: $\hat{q}_{1-\alpha}(t) \longleftarrow$ *the $(1-\alpha)$-quantile of the values $\varepsilon^*_{t,1}, \ldots, \varepsilon^*_{t,B}$.*

4. Main results

Our main results quantify how well the estimate $\hat{q}_{1-\alpha}(t)$ from Algorithm 1 approximates the true value $q_{1-\alpha}(t)$, and this will be done by analyzing how well the distribution of a bootstrap sample $\varepsilon^\star_{t,1}$ approximates the distribution of ε_t. For the purposes of comparing distributions, we will use the Lévy-Prohorov metric, defined below.

Lévy-Prohorov (LP) metric. Let $\mathcal{L}(U)$ denote the distribution of a random variable U, and let $\mathscr{B}$ denote the collection of Borel subsets of $\mathbb{R}$. For any $A \in \mathscr{B}$, and $\delta > 0$, define the δ-neighborhood $A^\delta := \big\{x \in \mathbb{R} \,\big|\, \inf_{y \in A} |x - y| \le \delta\big\}$. Then, for any two random variables U and V, the d_{LP} metric between their distributions is given by

$$d_{\mathrm{LP}}(\mathcal{L}(U), \mathcal{L}(V)) \;:=\; \inf\Big\{\delta > 0 \,\Big|\, \mathbb{P}(U \in A) \le \mathbb{P}(V \in A^\delta) + \delta \;\text{ for all }\; A \in \mathscr{B}\Big\}.$$

The d_{LP} metric is a standard tool for comparing distributions, due to the fact that convergence with respect to d_{LP} is equivalent to convergence in distribution (Huber and Ronchetti, 2009, Theorem 2.9).

Approximating quantiles. An important property of the d_{LP} metric is that if two distributions are close in this metric, then their quantiles are close in the following sense. Recall that if F_U is the distribution function of a random variable U, then the $(1-\alpha)$-quantile of U is the same as the generalized inverse $F_U^{-1}(1-\alpha) := \inf\{q \in [0, \infty) \,|\, F_U(q) \ge 1 - \alpha\}$. Next, suppose that two random variables U and V satisfy

$$d_{\mathrm{LP}}\big(\mathcal{L}(U),\, \mathcal{L}(V)\big) \;\le\; \epsilon,$$

for some $\epsilon \in (0, \alpha)$ with $\alpha \in (0, 1/2)$. Then, the quantiles of U and V are close in the sense that

$$\big|F_U^{-1}(1-\alpha) - F_V^{-1}(1-\alpha)\big| \;\le\; \psi_\alpha(\epsilon), \tag{15}$$

where the function $\psi_\alpha(\epsilon) := F_U^{-1}(1-\alpha+\epsilon) - F_U^{-1}(1-\alpha-\epsilon) + \epsilon$ is strictly monotone, and satisfies $\psi_\alpha(0) = 0$. (For a proof, see Lemma 15 of Appendix F.) In light of this fact, it will be more convenient to express our results for approximating $q_{1-\alpha}(t)$ in terms of the d_{LP} metric.

4.1. Statements of results

Our main assumption involves three separate cases, corresponding to different choices of the sketching matrix $\mathbf{S}$.

Assumption 1 *The dimensions d and d' satisfy $d \asymp d'$. Also, there is a positive absolute constant $\kappa \ge 1$ such that $d^{1/\kappa} \lesssim t \lesssim d^\kappa$, which is to say that neither d nor t grows exponentially with the other. In addition, one of the following sets of conditions holds, involving the parameter $\nu(\mathbf{A}, \mathbf{B}) := \sqrt{\|\mathbf{A}^T\mathbf{A}\|_\infty \|\mathbf{B}^T\mathbf{B}\|_\infty}$.*

(a) (Sub-Gaussian case). The entries of the matrix $\mathbf{S} = [S_{i,j}]$ are zero-mean i.i.d. sub-Gaussian random variables, with $\mathbb{E}[S_{i,j}^2] = \frac{1}{t}$, and $\max_{i,j} \|\sqrt{t} S_{i,j}\|_{\psi_2} \lesssim 1$. Furthermore, $t \gtrsim \nu(\mathbf{A}, \mathbf{B})^{2/3} (\log d)^5$.

(b) (Length sampling case). The matrix $\mathbf{S}$ is generated by length sampling, with the probabilities in equation (5), and also, $t \gtrsim (\|\mathbf{A}\|_F \|\mathbf{B}\|_F)^{2/3} (\log d)^5$.

(c) (SRHT case). The matrix $\mathbf{S}$ is an SRHT matrix as defined in equation (6), and also, $t \gtrsim \nu(\mathbf{A}, \mathbf{B})^{2/3} (\log n)^2 (\log d)^5$.

Clarifications on bootstrap approximation. Before stating our main results below, it is worth clarifying a few technical items. First, since our analysis involves central limit type approximations of $\tilde{\mathbf{A}}^T\tilde{\mathbf{B}} - \mathbf{A}^T\mathbf{B}$ as a sum of t independent matrices, we will rescale the error variables by a factor of $\sqrt{t}$, obtaining

$$Z_t := \sqrt{t}\varepsilon_t, \tag{16}$$

as well as its bootstrap analogue,

$$Z_t^\star := \sqrt{t}\varepsilon_t^\star. \tag{17}$$

With regard to the original problem of estimating the quantile $q_{1-\alpha}(t)$ for ε_t, this rescaling makes no essential difference, since quantiles are homogenous with respect to scaling, and in particular, the $(1-\alpha)$-quantile of Z_t is simply $\sqrt{t} q_{1-\alpha}(t)$.

As a second clarification, recall that the bootstrap method generates samples $\varepsilon_t^\star$ based upon a particular realization of $\mathbf{S}$. For this reason, the bootstrap approximation to $\mathcal{L}(Z_t)$ is the conditional distribution $\mathcal{L}(Z_t^\star|\mathbf{S})$. Consequently, it should be noted that $\mathcal{L}(Z_t^\star|\mathbf{S})$ is a random probability measure, and $d_{\mathrm{LP}}(\mathcal{L}(Z_t)\,,\, \mathcal{L}(Z_t^\star|\mathbf{S}))$ is a random variable, since they both depend on the random matrix $\mathbf{S}$.

Theorem 1 *Let $h(x) = x^{1/2} + x^{3/4}$ for $x \geq 0$. If Assumption 1 (a) holds, then there is an absolute constant $c > 0$ such that the following bound holds with probability at least $1 - \frac{1}{t} - \frac{1}{dd'}$,*

$$d_{\mathrm{LP}}\Big(\mathcal{L}(Z_t)\,,\,\mathcal{L}(Z_t^\star|\mathbf{S})\Big) \;\leq\; \frac{c\cdot h(\nu(\mathbf{A},\mathbf{B}))\cdot\sqrt{\log(d)}}{t^{1/8}}.$$

If Assumption 1 (b) holds, then there is an absolute constant $c > 0$ such that the following bound holds with probability at least $1 - \frac{1}{t} - \frac{1}{dd'}$,

$$d_{\mathrm{LP}}\Big(\mathcal{L}(Z_t)\,,\,\mathcal{L}(Z_t^\star|\mathbf{S})\Big) \;\leq\; \frac{c\cdot h(\|\mathbf{A}\|_F\|\mathbf{B}\|_F)\cdot\sqrt{\log(d)}}{t^{1/8}}.$$

Remarks. A noteworthy property of the bounds is that they are *dimension-free* with respect to the large dimension n. Also, they have a very mild logarithmic dependence on d. With regard to the dependence on t, there are two other important factors to keep in mind. First, the practical performance of the bootstrap method (shown in Section 5) is much better than what the $t^{-1/8}$ rate suggests. Second, the problem of finding the optimal rates of approximation for multiplier bootstrap methods is a largely open problem — even in the simpler setting of bootstrapping the coordinate-wise maximum of vectors (rather than matrices). In the vector context, the literature has focused primarily on the Kolmogorov metric (rather than the LP metric), and some quite recent improvements beyond the $t^{-1/8}$ rate have been developed in Chernozhukov et al. (2017) and Lopes et al. (2018a). However, these works also rely on model assumptions that would lead to additional restrictions on the matrices $\mathbf{A}$ and $\mathbf{B}$ in our setup. Likewise, the problem of extending our results to achieve faster rates or handle other metrics is a natural direction for future work.

The SRHT case. For the case of SRHT matrices, the analogue of Theorem 1 needs to be stated in a slightly different way for technical reasons. From a qualitative standpoint, the results for SRHT and sub-Gaussian matrices turn out to be similar.

The technical issue to be handled is that the rows of an SRHT matrix are not independent, due to their common dependence on the matrix $\mathbf{D}_n^\circ$. Fortunately, this inconvenience can be addressed by conditioning on $\mathbf{D}_n^\circ$. Theoretically, this simplifies the analysis of the bootstrap, since it "decouples" the rows of the SRHT matrix. Meanwhile, if we let $\tilde{q}_{1-\alpha}(t)$ denote the $(1-\alpha)$-quantile of the distribution $\mathcal{L}(\varepsilon_t|\mathbf{D}_n^\circ)$,

$$\tilde{q}_{1-\alpha}(t) \;:=\; \inf\Big\{q\in[0,\infty)\Big|\;\mathbb{P}(\varepsilon_t\leq q\big|\mathbf{D}_n^\circ)\geq 1-\alpha\Big\},$$

then it is simple to check that $\tilde{q}_{1-\alpha}(t)$ acts as a "surrogate" for $q_{1-\alpha}(t)$, since[‡]

$$\begin{aligned}\mathbb{P}(\varepsilon_t\leq\tilde{q}_{1-\alpha}(t)) &= \mathbb{E}\big[\mathbb{P}\big(\varepsilon_t\leq\tilde{q}_{1-\alpha}(t)\big|\mathbf{D}_n^\circ\big)\big]\\ &\geq \mathbb{E}[1-\alpha]\\ &= 1-\alpha.\end{aligned}\tag{18}$$

[‡]It is also possible to show that $\tilde{q}_{1-\alpha}(t)$ fluctuates around $q_{1-\alpha}(t)$. Indeed, if we define the random variable $V := \mathbb{P}(\varepsilon_t \leq q_{1-\alpha}(t)|\mathbf{D}_n^\circ)$, it can be checked that the event $V \geq 1-\alpha$ is equivalent to the event $\tilde{q}_{1-\alpha}(t) \leq q_{1-\alpha}(t)$. Furthermore, if we suppose that $1-\alpha$ lies in the range of the c.d.f. of ε_t, then $\mathbb{E}[V] = 1-\alpha$. In turn, it follows that the event $\tilde{q}_{1-\alpha}(t) \leq q_{1-\alpha}(t)$ occurs when $V \geq \mathbb{E}[V]$, and conversely, the event $\tilde{q}_{1-\alpha}(t) > q_{1-\alpha}(t)$ occurs when $V < \mathbb{E}[V]$.

For this reason, we will view $\tilde{q}_{1-\alpha}(t)$ as the new parameter to estimate (instead of $q_{1-\alpha}(t)$), and accordingly, the aim of the following result is to quantify how well the bootstrap distribution $\mathcal{L}(Z_t^\star|\mathbf{S})$ approximates the conditional distribution $\mathcal{L}(Z_t|\mathbf{D}_n^\circ)$.

Theorem 2 *Let $h(x) = x^{1/2} + x^{3/4}$ for $x \geq 0$. If Assumption 1 (c) holds, then there is an absolute constant $c > 0$ such that the following bound holds with probability at least $1 - \frac{1}{t} - \frac{1}{dd'} - \frac{c}{n}$,*

$$d_{\mathrm{LP}}\Big(\mathcal{L}(Z_t|\mathbf{D}_n^\circ)\,,\,\mathcal{L}(Z_t^\star|\mathbf{S})\Big) \;\leq\; \frac{c\cdot h(\nu(\mathbf{A},\mathbf{B})\log(n))\cdot\sqrt{\log(d)}}{t^{1/8}}.$$

Remarks. Up to a factor involving $\log(n)$, the bound for SRHT matrices matches that for sub-Gaussian matrices. Meanwhile, from a more practical standpoint, our empirical results will show that the bootstrap's performance for SRHT matrices is generally similar to that for both sub-Gaussian and length-sampling matrices.

Further discussion of results. To comment on the role of $\nu(\mathbf{A},\mathbf{B})$ and $\|\mathbf{A}\|_F\|\mathbf{B}\|_F$ in Theorems 1 and 2, it is possible to interpret them as problem-specific "scale parameters". Indeed, it is natural that the bounds on d_{LP} should increase with the scale of $\mathbf{A}$ and $\mathbf{B}$ for the following reason. Namely, if $\mathbf{A}$ or $\mathbf{B}$ is multiplied by a scale factor $\kappa > 0$, then it can be checked that the quantile error $|\hat{q}_{1-\alpha}(t) - q_{1-\alpha}(t)|$ will also change by a factor of κ, and furthermore, the inequality (15) demonstrates a monotone relationship between the sizes of the quantile error and the d_{LP} error. For this reason, the bootstrap may still perform well in relation to the scale of the problem when the magnitudes of the parameters $\nu(\mathbf{A},\mathbf{B})$ and $\|\mathbf{A}\|_F\|\mathbf{B}\|_F$ are large. Alternatively, this idea can be seen by noting that the d_{LP} bounds can be made arbitrarily small by simply changing the units used to measure the entries of $\mathbf{A}$ and $\mathbf{B}$.

Beyond these considerations, it is still of interest to compare the results for different sketching matrices once a particular scaling has been fixed. For concreteness, consider a scaling where the spectral norms of $\mathbf{A}$ and $\mathbf{B}$ satisfy $\|\mathbf{A}\|_2 \asymp \|\mathbf{B}\|_2 \asymp 1$. (As an example, if we view $\mathbf{A}^\top\mathbf{A}$ as a sample covariance matrix, then the condition $\|\mathbf{A}\|_2 \asymp 1$ simply means that the largest principal component score is of order 1.) Under this scaling, it is simple to check that $\nu(\mathbf{A},\mathbf{B}) = \mathcal{O}(1)$, and $\|\mathbf{A}\|_F\|\mathbf{B}\|_F = \mathcal{O}(\sqrt{\mathsf{r}(\mathbf{A})\mathsf{r}(\mathbf{B})})$, where $\mathsf{r}(\mathbf{A}) := \|\mathbf{A}\|_F^2/\|\mathbf{A}\|_2^2$ is the "stable rank". In particular, note that if $\mathbf{A}$ and $\mathbf{B}$ are approximately low rank, as is common in applications, then $\mathsf{r}(\mathbf{A}) \ll d$, and $\mathsf{r}(\mathbf{B}) \ll d'$. Accordingly, we may conclude that if the conditions of Theorems 1 and 2 hold, then bootstrap consistency occurs under the following limits

$$\sqrt{\log(d)}/t^{1/8} = o(1) \quad \text{in the sub-Gaussian case,} \tag{19}$$

$$(\mathsf{r}(\mathbf{A})\mathsf{r}(\mathbf{B}))^{3/8}\sqrt{\log(d)}/t^{1/8} = o(1) \quad \text{in the length-sampling case,} \tag{20}$$

$$\log(n)^{3/4}\sqrt{\log(d)}/t^{1/8} = o(1) \quad \text{in the SRHT case,} \tag{21}$$

where we have used the simplifying assumption that $d \asymp d'$.

5. Experiments

This section outlines a set of experiments for evaluating the performance of Algorithm 1 with the extrapolation speed-up described in Section 3.3. The experiments involved both synthetic and natural matrices, as described below.

Synthetic matrices. In order to generate the matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ synthetically, we selected the factors of its singular value decomposition $\mathbf{A} = \mathbf{U}\mathsf{diag}(\boldsymbol{\sigma})\mathbf{V}^T$ in the following ways, fixing $n = 30,000$ and $d = 1,000$. In previous work, a number of other experiments in randomized matrix computations have been designed along these lines (Ma et al., 2014; Yang et al., 2016).

The factor $\mathbf{U} \in \mathbb{R}^{n\times d}$ was selected as the Q factor from the reduced QR factorization of a random matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$. The rows of $\mathbf{X}$ were sampled i.i.d. from a multivariate t-distribution, $\boldsymbol{t}_2(\boldsymbol{\mu}, \mathbf{C})$, with 2 degrees of freedom, mean $\boldsymbol{\mu} = \mathbf{0}$, and covariance $c_{ij} = 2 \times 0.5^{|i-j|}$ where $\mathbf{C} = [c_{ij}]$. (This choice causes the matrix $\mathbf{A}$ to have high row-coherence, which is of interest, since this is a challenging case for sampling-based sketching matrices.) Next, the factor $\mathbf{V} \in \mathbb{R}^{d\times d}$ was selected as the Q factor from a QR factorization of a $d \times d$ matrix with i.i.d. $\mathcal{N}(0, 1)$ entries. For the singular values $\boldsymbol{\sigma} \in \mathbb{R}^d_+$, we chose two options, leading to either a low or high stable rank $\mathsf{r}(\mathbf{A}) = \frac{\|\mathbf{A}\|_F^2}{\|\mathbf{A}\|_2^2}$. In the low stable rank case, we put $\sigma_i = 10^{\kappa_i}$ for a set of equally spaced values κ_i between 0 and -6, yielding $\mathsf{r}(\mathbf{A}) = 36.7$. Alternatively, in the high stable rank case, the entries of $\boldsymbol{\sigma}$ were equally spaced between 0.1 and 1, yielding $\mathsf{r}(\mathbf{A}) = 370.1$. Finally, to make all numerical comparisons on a common scale, we normalized $\mathbf{A}$ so that $\|\mathbf{A}^T\mathbf{A}\|_\infty = 1$.

Natural matrices. We also conducted experiments on five natural data matrices $\mathbf{A}$ from the LIBSVM repository Chang and Lin (2011), named 'Connect', 'DNA', 'MNIST', 'Mushrooms', and 'Protein', with the same normalization that was used for the synthetic matrices. These datasets are briefly summarized in Table 1.

Table 1: A summary of the natural datasets.

Dataset	Connect	DNA	MNIST	Mushrooms	Protein
n	67,557	2,000	60,000	8,124	17,766
d	126	180	780	112	356

5.1. Design of experiments

For each matrix $\mathbf{A}$, natural or synthetic, we considered the task of estimating the quantile $q_{0.99}(t)$ for the random sketching error $\varepsilon_t = \left\|\mathbf{A}^T\mathbf{A} - \mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{A}\right\|_\infty$. The sketching matrix $\mathbf{S} \in \mathbb{R}^{t\times n}$ was allowed to be one of three types: Gaussian projection, length-sampling, and SRHT, as described in Section 2.

Ground truth values. The ground truth values for $q_{0.99}(t)$ were constructed in the following way. For each matrix $\mathbf{A}$, a grid of t values was specified, ranging from $d/2$ up to a larger number as high as $10d$ or $20d$, depending on $\mathbf{A}$. Next, for each t value, and for each type of sketching matrix, we used 1,000 realizations of $\mathbf{S} \in \mathbb{R}^{t\times n}$, yielding 1,000 realizations of the random variable ε_t. In turn, the 0.99 sample quantile of the 1,000 realizations of ε_t was treated as the true value of $q_{0.99}(t)$, and this appears as the black curve in all plots.

Extrapolated estimates. With regard to the bootstrap extrapolation method in Section 3.3, we fixed the value $t_0 = d/2$ as the initial sketch size to extrapolate from. For each $\mathbf{A}$, and each type of sketching matrix, we applied Algorithm 1 to each of the 1,000 realizations of $\tilde{\mathbf{A}} = \mathbf{S}\mathbf{A} \in \mathbb{R}^{t_0\times d}$ generated previously. Each time Algorithm 1 was run, we used the modest choice of $B = 20$ for the number of bootstrap samples. From each set of 20 bootstrap samples, we used the 0.99 sample quantile as the estimate $\hat{q}_{0.99}(t_0)$.[§] Hence, there were 1,000 realizations of $\hat{q}_{0.99}(t_0)$ altogether. Next, we used the scaling rule in equation (12) to obtain 1,000 realizations of the extrapolated estimate $\hat{q}^{\text{ext}}_{0.99}(t)$ for values $t \geq t_0$.

In order to illustrate the variability of the estimate $\hat{q}^{\text{ext}}_{0.99}(t)$ over the 1,000 realizations, we plot three different curves as a function of t. The blue curve represents the average value of $\hat{q}^{\text{ext}}_{0.99}(t)$, while the green and yellow curves respectively correspond to the estimates ranking 100th an 900th out of the 1,000 realizations.

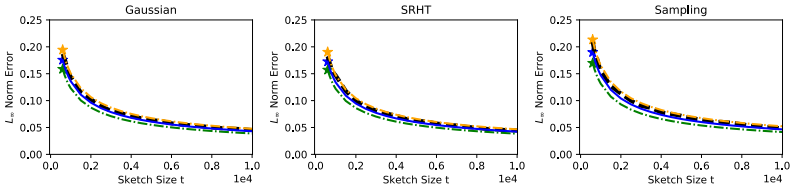
5.2. Comments on numerical results

Overall, the numerical results for the bootstrap extrapolation method are quite encouraging, and to a large extent, the method is accurate across many choices of $\mathbf{A}$ and $\mathbf{S}$. Given that the blue curves representing $\mathbb{E}[\hat{q}^{\text{ext}}_{0.99}(t)]$ are closely aligned with the black curves for $q_{0.99}(t)$, we see that the extrapolated estimate is essentially unbiased. Moreover, the variance of the estimate is fairly low, as indicated by the small gap between the green and yellow curves. The low variance is also notable when considered in light of the fact that only $B = 20$ bootstrap samples are used to construct $\hat{q}^{\text{ext}}_{0.99}(t)$, since the variance should decrease as B becomes larger.

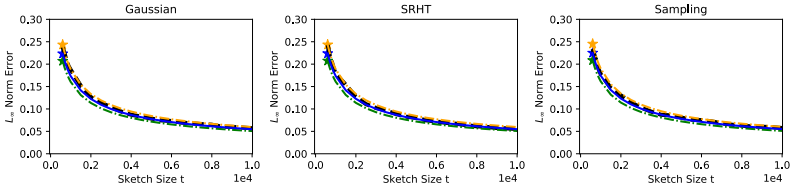
With attention to the extrapolation rule (12), there are two main points to note. First, the plots show that the extrapolation may be initiated at fairly low values of t_0, which are much less than the sketch sizes needed to achieve a small sketching error ε_t. Second, we see that $\hat{q}^{\text{ext}}_{0.99}(t)$ remains accurate for t much larger than t_0, well up to $t = 10,000$ and perhaps even farther. Consequently, the results show that the extrapolation technique is capable of saving quite a bit of computation without much detriment to statistical performance.

To consider the relationship between theory and practice, one basic observation is that all three types of sketching matrices obey roughly similar bounds in Theorems 1 and 2, and indeed, we also see generally similar numerical performance among the three types. At a more fine-grained level however, the Gaussian and SRHT sketching matrices tend to produce estimates $\hat{q}^{\text{ext}}_{0.99}(t)$ with somewhat higher variance than in the case of length sampling. Another difference between theory and simulation, is that the actual performance of the method seems to be better than what the theory suggests — since the estimates are accurate at values of t_0 that are much smaller than what would be expected from the rates in Theorems 1 and 2.

[§]Note that since $19/20 = 0.95$ and $20/20 = 1$, the 0.99 quantile was obtained by an interpolation rule.



(a) Low stable rank data.



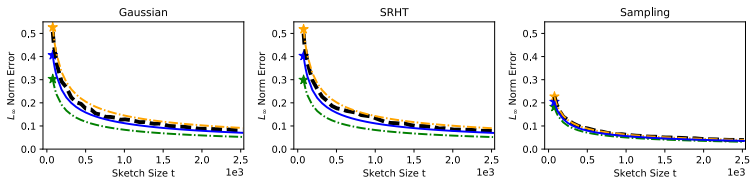
(b) High stable rank data.

Figure 2: **Results for synthetic matrices.** The black line represents $q_{0.99}(t)$ as a function of t. The blue star is the average bootstrap estimate at the initial sketch size $t_0 = d/2 = 500$, and the blue line represents the average extrapolated estimate $\mathbb{E}[\hat{q}^{\text{ext}}_{0.99}(t)]$ derived from the starting value t_0. To display the variability of the estimates, the green and yellow curves correspond to the 100th and 900th largest among the 1,000 realizations of $\hat{q}^{\text{ext}}_{0.99}(t)$ at each t.

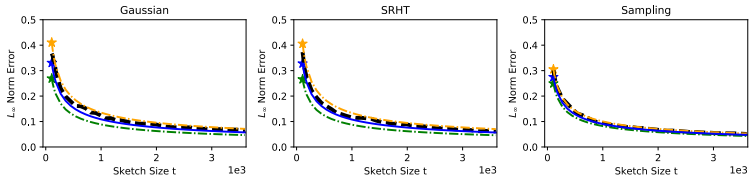
6. Conclusions and extensions

In this paper, we have focused on estimating the quantile $q_{1-\alpha}(t)$ as a way of addressing two fundamental issues in randomized matrix multiplication: (1) knowing how accurate a given sketched product is, and (2) knowing how much computation is needed to achieve a specified degree of accuracy. With regard to methodology, our approach is relatively novel in that it uses the statistical technique of bootstrapping to serve a computational purpose — by quantifying the error of a randomized sketching algorithm. A second important component of our method is the extrapolation technique, which ensures that the cost of estimating $q_{1-\alpha}(t)$ does not substantially increase the overall cost of standard sketching methods. Furthermore, our numerical results show that the extrapolated estimate is quite accurate in a variety of different situations, suggesting that our method may offer a general way to enhance sketching algorithms in practice.

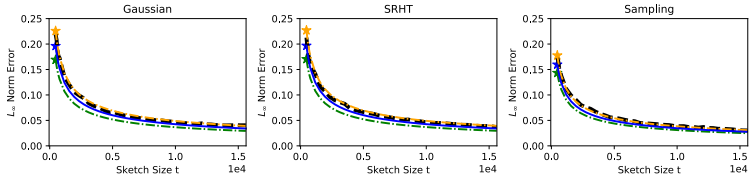
Extensions. More generally, the problems we have addressed for randomized matrix multiplication arise for many other large-scale matrix computations. Hence, it is natural to consider extensions of our approach to more complex settings, and in the remainder of this section, we briefly mention a few possibilities for future study.



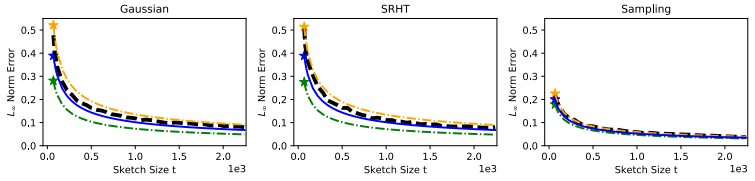
(a) Connect ($n = 67,557$ and $d = 126$).



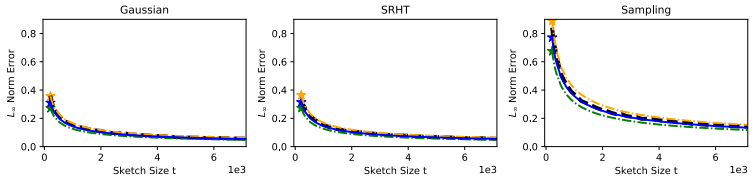
(b) DNA ($n = 2,000$ and $d = 180$).



(c) MNIST ($n = 60,000$ and $d = 780$).



(d) Mushrooms ($n = 8,124$ and $d = 112$).



(e) Protein ($n = 17,766$ and $d = 356$).

Figure 3: **Results for natural matrices.** The results for the natural matrices are plotted in the same way as described in the caption for the results on the synthetic matrices.

At a high level, each of the applications below deals with an object, say Θ, that is difficult to compute, as well as a randomized approximation, say $\widetilde{\Theta}$, that is built from a sketching matrix $\mathbf{S}$ with t rows. Next, if we consider the random error variable

$$\varepsilon_t = \|\widetilde{\Theta} - \Theta\|,$$

for an unspecified norm $\|\cdot\|$, then the problem of estimating the relationship between accuracy and computation can again be viewed as the problem of estimating the quantile function $q_{1-\alpha}(t)$ associated with ε_t. In turn, this leads to the question of how to develop a new bootstrap procedure that can generate approximate samples of ε_t, yielding an estimate $\hat{q}_{1-\alpha}(t)$. However, instead of starting from the multiplier bootstrap (Algorithm 1) as before, it may be conceptually easier to extend the non-parametric bootstrap (Algorithm 2) — because the latter bootstrap can viewed as a "plug-in" procedure that replaces $\mathbf{A}^T\mathbf{B}$ with $\tilde{\mathbf{A}}^T\tilde{\mathbf{B}}$, and replaces $\tilde{\mathbf{A}}^T\tilde{\mathbf{B}}$ with $(\tilde{\mathbf{A}}^*)^T(\tilde{\mathbf{B}}^*)$.

- *Linear regression.* Consider a multi-response linear regression problem, where the rows of $\mathbf{B} \in \mathbb{R}^{n\times d'}$ are response vectors, and the rows of $\mathbf{A} \in \mathbb{R}^{n\times d}$ are input observations. The optimal solution to ℓ_2-regression is given by

$$\mathbf{W}_{\text{opt}} \;=\; \underset{\mathbf{W}\in\mathbb{R}^{d\times d'}}{\text{argmin}} \; \|\mathbf{AW} - \mathbf{B}\|_F^2 \;=\; (\mathbf{A}^T\mathbf{A})^\dagger\mathbf{A}^T\mathbf{B},$$

  which has $\mathcal{O}(nd^2 + ndd')$ cost. In the case where $\max\{d, d'\} \ll n$, the matrix multiplications are a computational bottleneck, and an approximate solution can be obtained via

$$\widetilde{\mathbf{W}}_{\text{opt}} = (\tilde{\mathbf{A}}^T\tilde{\mathbf{A}})^\dagger(\tilde{\mathbf{A}}^T\tilde{\mathbf{B}}),$$

  which has a cost $\mathcal{O}(td^2 + tdd') + C_{\text{sketch}}$, where C_{sketch} is cost of matrix sketching (Drineas et al., 2006b, 2011, 2012; Clarkson and Woodruff, 2013). In order to estimate the quantile function associated with the error variable $\varepsilon_t = \|\widetilde{\mathbf{W}}_{\text{opt}} - \mathbf{W}_{\text{opt}}\|$, we could consider generating bootstrap samples of the form $\varepsilon_t^* = \|\widetilde{\mathbf{W}}^*_{\text{opt}} - \widetilde{\mathbf{W}}_{\text{opt}}\|$, where $\widetilde{\mathbf{W}}^*_{\text{opt}} = ((\tilde{\mathbf{A}}^*)^T(\tilde{\mathbf{A}}^*))^\dagger(\tilde{\mathbf{A}}^*)^T(\tilde{\mathbf{B}}^*)$. For recent results in the case where $\mathbf{W}$ is a vector, we refer to the paper (Lopes et al., 2018b).

- *Functions of covariance matrices.* If the rows of the matrix $\mathbf{A}$ are viewed as a sample of observations, then inferences on the population covariance structure are often based on functions of the form $\psi(\mathbf{A}^T\mathbf{A})$. For instance, the function $\psi(\mathbf{A}^T\mathbf{A})$ could be the top eigenvector, a set of eigenvalues, the condition number, or a test statistic. In any of these cases, if $\psi(\tilde{\mathbf{A}}^T\tilde{\mathbf{A}})$ is used as a fast approximation (Dasarathy et al., 2015), then the sketching error $\varepsilon_t = \|\psi(\tilde{\mathbf{A}}^T\tilde{\mathbf{A}}) - \psi(\mathbf{A}^T\mathbf{A})\|$ might be bootstrapped using $\varepsilon_t^* = \|\psi((\tilde{\mathbf{A}}^*)^T(\tilde{\mathbf{A}}^*)) - \psi(\tilde{\mathbf{A}}^T\tilde{\mathbf{A}})\|$.

- *Approximate Newton methods.* In large-scale applications, Newton's method is often impractical, since it involves the costly processing of a Hessian matrix. As an example, consider an optimization problem arising in binary classification, where the rows of $\mathbf{X} \in \mathbb{R}^{n\times d}$ are observations $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, and $y_1, \ldots, y_n \in \{0, 1\}$ are labels. If an ℓ_2-regularized logistic classifier is used, this leads to minimizing the objective function $f(\mathbf{w}) = \sum_{i=1}^n \log\left(1 + e^{-y_i\mathbf{w}^T\mathbf{x}_i}\right) + \frac{\gamma}{2}\|\mathbf{w}\|_2^2$ over coefficient vectors $\mathbf{w}$ in $\mathbb{R}^d$. The associated Newton step, with step size κ, is

$$\mathbf{w} \leftarrow \mathbf{w} - \kappa\,\mathbf{H}^{-1}\nabla f,$$

  involving the Hessian

$$\mathbf{H} = \mathbf{A}^T\mathbf{A} + \gamma\mathbf{I}_d, \quad \text{where} \quad \mathbf{A} = \mathsf{diag}\big(1 + e^{y_1\mathbf{w}^T\mathbf{x}_1}, \ldots, 1 + e^{y_n\mathbf{w}^T\mathbf{x}_n}\big)^{-1}\mathbf{X}.$$

  If $d \ll n$, the cost of Newton's method is dominated by the formation of $\mathbf{H}$ at each iteration, and the Hessian matrix can be approximated by the sketched version $\tilde{\mathbf{H}} = \tilde{\mathbf{A}}^T\tilde{\mathbf{A}} + \gamma\mathbf{I}_d$, which reduces the per-iteration cost from $\mathcal{O}(nd^2)$ to $\mathcal{O}(td^2 + nd) + C_{\text{sketch}}$ (Pilanci and Wainwright, 2017; Roosta-Khorasani and Mahoney, 2016; Xu et al., 2016). In this context, the quality of the approximate Newton step could be assessed in terms of the error

$$\varepsilon_t = \|\tilde{\mathbf{H}}^{-1}\nabla f \;-\; \mathbf{H}^{-1}\nabla f\|,$$

  and in turn, this might be bootstrapped using $\varepsilon_t^* = \|(\tilde{\mathbf{H}}^*)^{-1}\nabla f \;-\; \tilde{\mathbf{H}}^{-1}\nabla f\|$, where $\tilde{\mathbf{H}}^* = (\tilde{\mathbf{A}}^*)^T(\tilde{\mathbf{A}}^*) + \gamma\mathbf{I}_d$.

Acknowledgments

We thank the anonymous reviewers for their helpful suggestions. MEL thanks the National Science Foundation for partial support under grant DMS-1613218. MWM would like to thank the National Science Foundation, the Army Research Office, and the Defense Advanced Research Projects Agency for providing partial support of this work.

Appendices

Outline of appendices. Appendix A explains the main conceptual ideas underlying the proofs of Theorems 1 and 2. In particular, the proofs of these theorems will be decomposed into two main results: Propositions 3 and 4, which are given in Appendix A.

Appendix B will prove the sub-Gaussian case of Proposition 3, and Appendix C will prove the sub-Gaussian case of Proposition 4. Later on, Appendices D and E, will explain how the arguments can be changed to handle the length-sampling and SRHT cases.

Conventions used in proofs. If either of the matrices $\mathbf{A}$ or $\mathbf{B}$ are $\mathbf{0}$, then ε_t has a trivial point-mass distribution at 0. In this degenerate case, it is simple to check that the bootstrap produces an exact approximation. So, without loss of generality, all proofs are written under the assumption that $\mathbf{A}$ and $\mathbf{B}$ are non-zero. Next, since Assumption 1 is formulated using the $\lesssim$ notation, there is no loss of generality in carrying out calculations under the assumption that all the numbers t, n, d, d' are at least 8, which will ensure that quantities such as $\log(d)$ are greater than 2. Lastly, if a numbered lemma is invoked in the middle of a proof, the lemma may be found in Appendix F.

Appendix A. Gaussian and bootstrap approximations

Section A.1 introduces some notation that helps us to analyze the rescaled sketching error Z_t from the viewpoint of empirical processes. Next, in Section A.2, Theorem 1 will be decomposed into two propositions that compare Z_t and $Z_t^\star$ with the maximum of a suitable Gaussian process. The proofs of these propositions may be found in Appendices B and C.

A.1. Making a link between empirical processes and sketching error

The main idea of our analysis is to view Z_t as the maximum of an empirical process, which we now define. Recall the notation

$$\mathbf{D}_i = \mathbf{A}^T\mathbf{s}_i\mathbf{s}_i^T\mathbf{B} \quad \text{and} \quad \mathbf{M}_i \;=\; \mathbf{A}^T(\mathbf{s}_i\mathbf{s}_i^T - \mathbf{I}_n)\mathbf{B}.$$

Let $\mathbb{G}_t(\cdot)$ be the empirical process that acts on *linear* functions $f : \mathbb{R}^{d\times d'} \to \mathbb{R}$, according to

$$\mathbb{G}_t(f) \;:=\; \frac{1}{\sqrt{t}}\sum_{i=1}^{t}\Big(f(\mathbf{D}_i) - f(\mathbf{A}^T\mathbf{B})\Big) \;=\; \frac{1}{\sqrt{t}}\sum_{i=1}^{t} f(\mathbf{M}_i).$$

For future reference, we also define the corresponding bootstrap process

$$\mathbb{G}_t^\star(f) \;:=\; \frac{1}{\sqrt{t}}\sum_{i=1}^{t}\xi_i\cdot\Big(f(\mathbf{D}_i) - f\big(\mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B}\big)\Big),$$

where $\xi_1,\ldots,\xi_t$ are i.i.d. $\mathcal{N}(0,1)$ and independent of $\mathbf{S}$.

Next, we define a certain collection $\mathscr{F}$ of linear functions from $\mathbb{R}^{d\times d'}$ to $\mathbb{R}$. Let $j_1 \in [d]$, $j_2 \in [d']$, $s \in \{-1,1\}$, and $\mathbf{j} := (j_1, j_2, s)$. Then, for any matrix $\mathbf{W} \in \mathbb{R}^{d\times d'}$, we put

$$f_{\mathbf{j}}(\mathbf{W}) \;:=\; s\cdot\mathrm{tr}\big(\mathbf{C}_{\mathbf{j}}^T\mathbf{W}\big),$$

where $\mathbf{C}_{\mathbf{j}} := s\,\mathbf{e}_{j_1}\mathbf{e}_{j_2}^T \in \mathbb{R}^{d\times d'}$ and $\mathbf{e}_{j_1} \in \mathbb{R}^d$, $\mathbf{e}_{j_2} \in \mathbb{R}^{d'}$ are standard basis vectors. In words, the function $f_{\mathbf{j}}$ merely picks out the (j_1, j_2) entry of $\mathbf{W}$, and multiplies by a sign s. Likewise, let $\mathscr{J}$ be the collection of all the triples $\mathbf{j}$, and define the class of linear functions

$$\mathscr{F} \;:=\; \big\{f_{\mathbf{j}} \mid \mathbf{j}\in\mathscr{J}\big\}.$$

Clearly, $\mathsf{card}(\mathscr{F}) = 2dd'$. Under this definition, it is simple to check that Z_t and $Z_t^\star$, defined in equations (16) and (17), can be expressed as

$$Z_t \;=\; \max_{f_{\mathbf{j}}\in\mathscr{F}} \mathbb{G}_t(f_{\mathbf{j}}), \qquad \text{and} \qquad Z_t^\star \;=\; \max_{f_{\mathbf{j}}\in\mathscr{F}} \mathbb{G}_t^\star(f_{\mathbf{j}}).$$

A.2. Statements of the approximation results

Theorems 1 and 2 are obtained by combining the following two results (Propositions 3 and 4) via the triangle inequality. In essence, these results are based on a comparison with the maximum of a certain Gaussian process. More specifically, let $\mathbb{G} : \mathscr{F} \to \mathbb{R}$ be a zero-mean Gaussian process whose covariance structure is defined according to

$$\begin{aligned}\mathbb{E}\big[\mathbb{G}(f_{\mathbf{j}})\,\mathbb{G}(f_{\mathbf{k}})\big] \;&=\; \mathrm{cov}\big(f_{\mathbf{j}}(\mathbf{D}_1),\, f_{\mathbf{k}}(\mathbf{D}_1)\big)\\ &=\; \mathbb{E}\Big[f_{\mathbf{j}}(\mathbf{D}_1)\,f_{\mathbf{k}}(\mathbf{D}_1)\Big] - f_{\mathbf{j}}(\mathbf{A}^T\mathbf{B})\,f_{\mathbf{k}}(\mathbf{A}^T\mathbf{B}),\end{aligned} \tag{22}$$

for all $\mathbf{j},\mathbf{k} \in \mathscr{J}$. In turn, define the following random variable as the the maximum of this Gaussian process,

$$Z \;:=\; \max_{f_{\mathbf{j}}\in\mathscr{F}} \mathbb{G}(f_{\mathbf{j}}).$$

In order to handle the case of SRHT matrices, define another zero-mean Gaussian process $\tilde{\mathbb{G}} : \mathscr{F} \to \mathbb{R}$ (conditionally on a fixed realization of $\mathbf{D}_n^\circ$) to have its covariance structure given by

$$\begin{aligned}\mathbb{E}\big[\tilde{\mathbb{G}}(f_{\mathbf{j}})\,\tilde{\mathbb{G}}(f_{\mathbf{k}})\big|\mathbf{D}_n^\circ\big] \;&=\; \mathrm{cov}\big(f_{\mathbf{j}}(\mathbf{D}_1),\, f_{\mathbf{k}}(\mathbf{D}_1)\big|\mathbf{D}_n^\circ\big)\\ &=\; \mathbb{E}\Big[f_{\mathbf{j}}(\mathbf{D}_1)\,f_{\mathbf{k}}(\mathbf{D}_1)\Big|\mathbf{D}_n^\circ\Big] - f_{\mathbf{j}}(\mathbf{A}^T\mathbf{B})\,f_{\mathbf{k}}(\mathbf{A}^T\mathbf{B}),\end{aligned} \tag{23}$$

and let $\tilde{Z}$ denote the maximum of the process $\tilde{\mathbb{G}}$,

$$\tilde{Z} \;:=\; \max_{f_{\mathbf{j}}\in\mathscr{F}} \tilde{\mathbb{G}}(f_{\mathbf{j}}).$$

We are now in position to state the approximation results.

Proposition 3 (Gaussian approximation) *Under Assumption 1 (a), the following bound holds,*

$$d_{\mathrm{LP}}\big(\mathcal{L}(Z_t)\,,\,\mathcal{L}(Z)\big) \;\le\; \frac{c\cdot\nu(\mathbf{A},\mathbf{B})^{3/4}\cdot\sqrt{\log(d)}}{t^{1/8}}.$$

Under Assumption 1 (b), the following bound holds,

$$d_{\mathrm{LP}}\big(\mathcal{L}(Z_t)\,,\,\mathcal{L}(Z)\big) \;\le\; \frac{c\cdot(\|\mathbf{A}\|_F\|\mathbf{B}\|_F)^{3/4}\cdot\sqrt{\log(d)}}{t^{1/8}}.$$

Under Assumption 1 (c), the following bound holds with probability at least $1 - c/n$

$$d_{\mathrm{LP}}\big(\mathcal{L}(Z_t)\,,\,\mathcal{L}(\tilde{Z}|\mathbf{D}_n^\circ)\big) \;\le\; \frac{c\cdot\nu(\mathbf{A},\mathbf{B})^{3/4}\cdot(\log(n))^{3/4}\cdot\sqrt{\log(d)}}{t^{1/8}}.$$

Proposition 4 (Bootstrap approximation) *If Assumption 1 (a) holds, then the following bound holds with probability at least $1 - \frac{1}{t} - \frac{1}{dd'}$,*

$$d_{\mathrm{LP}}\big(\mathcal{L}(Z)\,,\,\mathcal{L}(Z_t^\star|\mathbf{S})\big) \;\le\; \frac{c\cdot\nu(\mathbf{A},\mathbf{B})^{1/2}\sqrt{\log(d)}}{t^{1/8}}.$$

If Assumption 1 (b) holds, then the following bound holds with probability at least $1-\frac{1}{t}-\frac{1}{dd'}$,

$$d_{\mathrm{LP}}\big(\mathcal{L}(Z)\,,\,\mathcal{L}(Z_t^\star|\mathbf{S})\big) \;\le\; \frac{c\cdot(\|\mathbf{A}\|_F\|\mathbf{B}\|_F)^{1/2}\sqrt{\log(d)}}{t^{1/8}}.$$

If Assumption 1 (c) holds, then the following bound holds with probability at least $1-\frac{1}{t}-\frac{1}{dd'}-\frac{c}{n}$,

$$d_{\mathrm{LP}}\big(\mathcal{L}(\tilde{Z}|\mathbf{D}_n^\circ)\,,\,\mathcal{L}(Z_t^\star|\mathbf{S})\big) \;\le\; \frac{c\cdot\nu(\mathbf{A},\mathbf{B})^{1/2}\cdot\log(n)^{1/2}\cdot\sqrt{\log(d)}}{t^{1/8}}.$$

Appendix B. Proof of Proposition 3, part (a)

Let $A \subset \mathbb{R}$ be a Borel set. Due to Theorem 3.1 from the paper Chernozhukov et al. (2016), we have for any $\delta > 0$,

$$\mathbb{P}(Z_t \in A) \leq \mathbb{P}(Z \in A^{c\delta}) + \tfrac{c\log^2(d)}{\delta^3\sqrt{t}}\Big(L_t + K_t(\delta) + J_t(\delta))\Big), \tag{24}$$

where we define the following non-random quantities

$$L_t := \max_{f_{\mathbf{j}} \in \mathscr{F}} \frac{1}{t}\sum_{i=1}^{t} \mathbb{E}\Big[\big|f_{\mathbf{j}}(\mathbf{M}_i)\big|^3\Big], \tag{25}$$

$$K_t(\delta) := \mathbb{E}\Bigg[\max_{f_{\mathbf{j}} \in \mathscr{F}} |f_{\mathbf{j}}(\mathbf{M}_1)|^3 \cdot 1\Big\{\max_{f_{\mathbf{j}} \in \mathscr{F}} |f_{\mathbf{j}}(\mathbf{M}_1)| > \tfrac{\delta\sqrt{t}}{\log(\mathrm{card}(\mathscr{F}))}\Big\}\Bigg], \tag{26}$$

$$J_t(\delta) := \mathbb{E}\Bigg[\max_{f_{\mathbf{j}} \in \mathscr{F}} |\mathbb{G}(f_{\mathbf{j}})|^3 \cdot 1\Big\{\max_{f_{\mathbf{j}} \in \mathscr{F}} |\mathbb{G}(f_{\mathbf{j}})| > \tfrac{\delta\sqrt{t}}{\log(\mathrm{card}(\mathscr{F}))}\Big\}\Bigg]. \tag{27}$$

The remainder of the proof consists in bounding each of these quantities, and we will establish the following two bounds for all $\delta > 0$,

$$L_t \leq c\,\nu(\mathbf{A},\mathbf{B})^3, \tag{28}$$

$$K_t(\delta) + J_t(\delta) \leq c\Big(\tfrac{\delta\sqrt{t}}{\log(d)} + \log(d)\,\nu(\mathbf{A},\mathbf{B})\Big)^3 \cdot \exp\Big(-\tfrac{\delta\sqrt{t}}{c\;\nu(\mathbf{A},\mathbf{B})\log^2(d)}\Big). \tag{29}$$

Recall also that $\mathrm{card}(\mathscr{F}) = 2dd'$, and $d \asymp d'$ under Assumption 1.

For the moment, we set aside the task of proving these bounds, and consider the choice of δ. There are two constraints that we would like δ to satisfy. First, we would like to choose δ so that the bounds on L_t and $(K_t(\delta) + J_t(\delta))$ are of the same order. In particular, we desire

$$K_t(\delta) + J_t(\delta) \;\leq\; c\,\nu(\mathbf{A},\mathbf{B})^3. \tag{30}$$

Second, with regard to line (24) we would like δ to solve the equation

$$\delta \;=\; \tfrac{1}{\delta^3}\tfrac{\log^2(d)\,\nu(\mathbf{A},\mathbf{B})^3}{\sqrt{t}}, \tag{31}$$

so that the second term in line (24) is of order δ. The idea is that if δ satisfies both of the conditions (30) and (31), then the definition of the d_{LP} metric and line (24) imply

$$d_{\mathrm{LP}}(\mathcal{L}(Z), \mathcal{L}(Z_t)) \;\leq\; c\,\delta.$$

To proceed, consider the choice

$$\delta_0 \;:=\; \tfrac{\log^{1/2}(d)\,\nu(\mathbf{A},\mathbf{B})^{3/4}}{t^{1/8}},$$

which clearly satisfies line (31). Futhermore, it can be checked that δ_0 also satisfies the constraint (30) under Assumption 1 (a). (The details of verifying this are somewhat tedious and are given in Lemma 16 in Appendix F.)

To finish the proof, it remains to establish the bounds (28) and (29). To handle L_t, note that[¶]

$$\begin{aligned}
\mathbb{E}[|f_{\mathbf{j}}(\mathbf{M}_i)|^3] &= \|f_{\mathbf{j}}(\mathbf{M}_1)\|_3^3 \\
&\leq c\|f_{\mathbf{j}}(\mathbf{M}_1)\|_{\psi_1}^3, && \text{(Lemma 9)} \qquad (32)\\
&\leq c\Big(\tfrac{\|\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\|_F^2}{\|\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\|_2}\Big)^3, && \text{(Lemma 14)} \\
&= c\Big(\|\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\|_F\Big)^3 && \text{(since } \|\mathbf{H}\|_2 = \|\mathbf{H}\|_F \text{ when } \mathbf{H} \text{ is rank-1)} \\
&= c\Big(\mathrm{tr}\big(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\mathbf{A}\mathbf{C}_{\mathbf{j}}\mathbf{B}^T\big)\Big)^{3/2} \\
&= c\Big(\mathbf{e}_{j_1}^T\mathbf{A}^T\mathbf{A}\mathbf{e}_{j_1} \cdot \mathbf{e}_{j_2}^T\mathbf{B}^T\mathbf{B}\mathbf{e}_{j_2}\Big)^{3/2} \\
&\leq c\;\nu(\mathbf{A},\mathbf{B})^3, && (33)
\end{aligned}$$

which proves the claimed bound in line (28).

Next, regarding $K_t(\delta)$, let us consider the random variable

$$\eta := \max_{f_{\mathbf{j}} \in \mathscr{F}} |f_{\mathbf{j}}(\mathbf{M}_1)|.$$

It follows from Lemma 9 (part 4) and Lemma 13 in Appendix F that $K_t(\delta)$ can be bounded in terms of the Orlicz norm $\|\eta\|_{\psi_1}$,

$$K_t(\delta) \leq c\Big(\tfrac{\delta\sqrt{t}}{\log(\mathsf{card}(\mathscr{F}))} + \|\eta\|_{\psi_1}\Big)^3 \cdot \exp(-\tfrac{\delta\sqrt{t}}{\|\eta\|_{\psi_1}\log(\mathsf{card}(\mathscr{F})}).$$

To handle $\|\eta\|_{\psi_1}$, it follows from Lemma 9 (part 3), that

$$\|\eta\|_{\psi_1} \leq c\log(\mathsf{card}(\mathscr{F})) \cdot \max_{f_{\mathbf{j}} \in \mathscr{F}} \|f_{\mathbf{j}}(\mathbf{M}_1)\|_{\psi_1}. \tag{34}$$

Furthermore, due to the earlier calculation starting at line (32) above,

$$\|f_{\mathbf{j}}(\mathbf{M}_1)\|_{\psi_1} \leq c\;\nu(\mathbf{A},\mathbf{B}). \tag{35}$$

Combining the last few steps, we conclude that

$$K_t(\delta) \leq c\Big(\tfrac{\delta\sqrt{t}}{\log(\mathrm{card}(\mathscr{F}))} + \log(\mathrm{card}(\mathscr{F}))\nu(\mathbf{A},\mathbf{B})\Big)^3 \cdot \exp\Big(-\tfrac{\delta\sqrt{t}}{c\nu(\mathbf{A},\mathbf{B})\log^2(\mathrm{card}(\mathscr{F}))}\Big). \tag{36}$$

Lastly, we turn to bounding $J_t(\delta)$. Fortunately, much of the argument for bounding $K_t(\delta)$ can be carried over. Specifically, consider the random variable

$$\zeta := \max_{f_{\mathbf{j}} \in \mathscr{F}} |\mathbb{G}(f_{\mathbf{j}})|.$$

[¶]In this step, we use the assumption that $\|\sqrt{t}S_{i,j}\|_{\psi_2} \leq c$ for all i and j.

Lemma 13 in Appendix F shows that $J_t(\delta)$ can be bounded in terms of $\|\zeta\|_{\psi_1}$,

$$J_t(\delta) \leq c\Big(\tfrac{\delta\sqrt{t}}{\log(\mathsf{card}(\mathscr{F}))} + \|\zeta\|_{\psi_1}\Big)^3 \cdot \exp\Big(-\tfrac{\delta\sqrt{t}}{\|\zeta\|_{\psi_1}\,\log(\mathsf{card}(\mathscr{F}))}\Big).$$

Proceeding in a way that is similar to the bound for $K_t(\delta)$, it follows from part (3) of Lemma 9 that

$$\|\zeta\|_{\psi_1} \leq c\log(\mathsf{card}(\mathscr{F})) \cdot \max_{f_{\mathbf{j}}\in\mathscr{F}} \|\mathbb{G}(f_{\mathbf{j}})\|_{\psi_1}.$$

Furthermore, for every $f_{\mathbf{j}} \in \mathscr{F}$, the facts in Lemma 9 imply

$$\begin{aligned}
\|\mathbb{G}(f_{\mathbf{j}})\|_{\psi_1} &\leq c\,\|\mathbb{G}(f_{\mathbf{j}})\|_{\psi_2} \\
&\leq c\sqrt{\mathsf{var}(\mathbb{G}(f_{\mathbf{j}}))} \\
&= c\sqrt{\mathsf{var}(f_{\mathbf{j}}(\mathbf{D}_1))} \quad \text{(by definition of } \mathbb{G}) \\
&\leq c\|f_{\mathbf{j}}(\mathbf{D}_1)\|_2 \\
&\leq c\|f_{\mathbf{j}}(\mathbf{D}_1)\|_{\psi_1} \qquad (37) \\
&\leq c\,\|f_{\mathbf{j}}(\mathbf{D}_1) - \mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)\|_{\psi_1} + c\,|\mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)]| \\
&= c\|f_{\mathbf{j}}(\mathbf{M}_1)\|_{\psi_1} + c\,|\mathrm{tr}(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A})| \\
&\leq c\,\nu(\mathbf{A},\mathbf{B}), \qquad (38)
\end{aligned}$$

where the last step follows from the bounds (32) through (33), and the fact that $|\mathrm{tr}(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A})| \leq \nu(\mathbf{A},\mathbf{B})$. Consequently, up to a constant factor, $J_t(\delta)$ satisfies the same bound as $K_t(\delta)$ given in line (36), and this proves the claim in line (29).

■

Appendix C. Proof of Proposition 4, part (a)

We will show there is a set of "good" sketching matrices $\mathscr{S} \subset \mathbb{R}^{t\times n}$ with the following two properties. First, a randomly drawn sketching matrix $\mathbf{S}$ is likely to fall in $\mathscr{S}$. Namely,

$$\mathbb{P}(\mathbf{S} \in \mathscr{S}) \geq 1 - \tfrac{1}{t}. \qquad (39)$$

Second, whenever the event $\{\mathbf{S} \in \mathscr{S}\}$ occurs, we have the following bound for any $\delta > 0$ and any Borel set $A \subset \mathbb{R}$,

$$\mathbb{P}\Big(\max_{f_{\mathbf{j}}\in\mathscr{F}} \mathbb{G}_t^\star(f_{\mathbf{j}}) \in A \Big| \mathbf{S}\Big) \;\leq\; \mathbb{P}\Big(\max_{f_{\mathbf{j}}\in\mathscr{F}} \mathbb{G}(f_{\mathbf{j}}) \in A^\delta\Big) + \tfrac{c\,\nu(\mathbf{A},\mathbf{B})\cdot\log(\mathsf{card}(\mathscr{F}))}{\delta\, t^{1/4}}. \qquad (40)$$

If we set δ to the particular choice $\delta_0 := t^{-1/8}\sqrt{\nu(\mathbf{A},\mathbf{B})\cdot\log(\mathsf{card}(\mathscr{F}))}$, then δ_0 solves the equation

$$\delta_0 \;=\; \tfrac{\nu(\mathbf{A},\mathbf{B})\cdot\log(\mathsf{card}(\mathscr{F}))}{\delta_0\, t^{1/4}}.$$

Consequently, by the definition of the d_{LP} metric, this implies that whenever the event $\{\mathbf{S} \in \mathscr{S}\}$ occurs, we have

$$d_{\mathrm{LP}}\big(\mathcal{L}(Z_t^\star|\mathbf{S})\,,\,\mathcal{L}(Z)\big) \;\leq\; c\,t^{-1/8}\,\sqrt{\nu(\mathbf{A},\mathbf{B})\cdot\log(\mathsf{card}(\mathscr{F}))}, \qquad (41)$$

and this implies the statement of Proposition 4.

To proceed with the main argument of constructing $\mathscr{S}$ and demonstrating the two properties (39) and (40), it is helpful to think of $\mathbb{G}_t^\star$ (conditionally on $\mathbf{S}$) and $\mathbb{G}$ as Gaussian vectors of dimension $\mathsf{card}(\mathscr{F}) = 2dd'$. From this point of view, we can compare the maxima of these vectors using a result due to Chernozhukov et al. (2016, Theorem 3.2). Under our assumptions, this result implies that for any realization of $\mathbf{S}$, any number $\delta > 0$, and any Borel set $A \subset \mathbb{R}$, we have

$$\mathbb{P}\Big(\max_{f_{\mathbf{j}}\in\mathscr{F}} \mathbb{G}_t^\star(f_{\mathbf{j}}) \in A \Big| \mathbf{S}\Big) \;\leq\; \mathbb{P}\Big(\max_{f_{\mathbf{j}}\in\mathscr{F}} \mathbb{G}(f_{\mathbf{j}}) \in A^\delta\Big) + \tfrac{c\sqrt{\Delta_t(\mathbf{S})\log(\mathsf{card}(\mathscr{F}))}}{\delta},$$

where we define the following function of $\mathbf{S}$,

$$\Delta_t(\mathbf{S}) \;:=\; \max_{(f_{\mathbf{j}},f_{\mathbf{k}})\in\mathscr{F}\times\mathscr{F}} \Big|\mathbb{E}\big[\mathbb{G}_t^\star(f_{\mathbf{j}})\mathbb{G}_t^\star(f_{\mathbf{k}})\big|\mathbf{S}\big] - \mathbb{E}\big[\mathbb{G}(f_{\mathbf{j}})\mathbb{G}(f_{\mathbf{k}})\big]\Big|. \qquad (42)$$

When referencing Theorem 3.2 from the paper Chernozhukov et al. (2016), note that $\mathbb{E}[\mathbb{G}(f_{\mathbf{j}})] = 0$ and $\mathbb{E}[\mathbb{G}_t^\star(f_{\mathbf{j}})|\mathbf{S}] = 0$ for all $f_{\mathbf{j}} \in \mathscr{F}$. To interpret $\Delta_t(\mathbf{S})$, it may be viewed as the ℓ_∞-distance between the covariance matrices associated with $\mathbb{G}_t^\star$ (conditionally on $\mathbf{S}$) and $\mathbb{G}$.

Using the above notation, we define the set of sketching matrices $\mathscr{S} \subset \mathbb{R}^{n\times t}$ according to

$$\mathbf{S} \in \mathscr{S} \qquad \text{if and only if} \qquad \Delta_t(\mathbf{S}) \leq \tfrac{c}{\sqrt{t}}\cdot\nu(\mathbf{A},\mathbf{B})^2\cdot\log(\mathsf{card}(\mathscr{F})). \qquad (43)$$

Based on this definition, it is simple to check that the proof is reduced to showing that the event $\{\mathbf{S} \in \mathscr{S}\}$ occurs with probability at least $1 - \frac{1}{t} - \frac{1}{dd'}$. This is guaranteed by the lemma below.

■

Lemma 5 *Suppose Assumption 1 (a) holds. Then, the event*

$$\Delta_t(\mathbf{S}) \;\leq\; \frac{c}{\sqrt{t}}\cdot\nu(\mathbf{A},\mathbf{B})^2\cdot\log\big(\mathsf{card}(\mathscr{F})\big)$$

occurs with probability at least $1 - \frac{1}{t} - \frac{1}{dd'}$.

Proof We begin by bounding $\Delta_t(\mathbf{S})$ with two other quantities (to be denoted $\Delta_t'(\mathbf{S})$, $\Delta_t''(\mathbf{S})$) that are easier to bound. Using the fact that $\frac{1}{t}\sum_{i=1}^t \mathbf{D}_i = \mathbf{A}^T\mathbf{S}^T\mathbf{S}\mathbf{B}$ it can be checked that

$$\mathbb{E}\big[\mathbb{G}_t^\star(f_{\mathbf{j}})\mathbb{G}_t^\star(f_{\mathbf{k}})\big|\mathbf{S}\big] \;=\; \Big(\tfrac{1}{t}\textstyle\sum_{i=1}^t f_{\mathbf{j}}(\mathbf{D}_i)f_{\mathbf{k}}(\mathbf{D}_i)\Big) - \Big(\tfrac{1}{t}\textstyle\sum_{i=1}^t f_{\mathbf{j}}(\mathbf{D}_i)\Big)\cdot\Big(\tfrac{1}{t}\textstyle\sum_{i=1}^t f_{\mathbf{k}}(\mathbf{D}_i)\Big).$$

Similarly, recall from line (22) that

$$\mathbb{E}[\mathbb{G}(f_{\mathbf{j}})\mathbb{G}(f_{\mathbf{k}})] \;=\; \mathbb{E}\Big[f_{\mathbf{j}}(\mathbf{D}_1)f_{\mathbf{k}}(\mathbf{D}_1)\Big] - \mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)]\cdot\mathbb{E}[f_{\mathbf{k}}(\mathbf{D}_1)].$$

From looking at the last two lines, it is natural to define the following *zero-mean* random variables for any triple $i, \mathbf{j}, \mathbf{k}$,[‖]

$$Q_{i\,\mathbf{j},\mathbf{k}} := f_{\mathbf{j}}(\mathbf{D}_i) f_{\mathbf{k}}(\mathbf{D}_i) - \mathbb{E}\Big[f_{\mathbf{j}}(\mathbf{D}_i) f_{\mathbf{k}}(\mathbf{D}_i)\Big],$$

and

$$R_{t,\mathbf{j}} := \tfrac{1}{t}\textstyle\sum_{i=1}^{t}\big(f_{\mathbf{j}}(\mathbf{D}_i) - \mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_i)]\big).$$

Then, some algebra shows that

$$\begin{aligned}\mathbb{E}\big[\mathbb{G}_t^*(f_{\mathbf{j}})\,\mathbb{G}_t^*(f_{\mathbf{k}})\,\big|\,\mathbf{S}\big] - \mathbb{E}\big[\mathbb{G}(f_{\mathbf{j}})\,\mathbb{G}(f_{\mathbf{k}})\big] = \Big(\tfrac{1}{t}\textstyle\sum_{i=1}^{t} Q_{i\,\mathbf{j},\mathbf{k}}\Big) - R_{t,\mathbf{j}}\cdot R_{t,\mathbf{k}} \\ - \mathbb{E}\big[f_{\mathbf{j}}(\mathbf{D}_1)\big]\cdot R_{t,\mathbf{k}} - \mathbb{E}\big[f_{\mathbf{k}}(\mathbf{D}_1)\big]\cdot R_{t,\mathbf{j}}.\end{aligned}$$

So, if we define the quantities

$$\Delta_t'(\mathbf{S}) \;:=\; \max_{(\mathbf{j},\mathbf{k})\in\mathscr{J}\times\mathscr{J}}\Big|\tfrac{1}{t}\sum_{i=1}^{t} Q_{i,\mathbf{j},\mathbf{k}}\Big|,$$

$$\Delta_t''(\mathbf{S}) \;:=\; \max_{\mathbf{j}\in\mathscr{J}}\big|R_{t,\mathbf{j}}\big|,$$

then

$$\Delta_t(\mathbf{S}) \;\le\; \Delta_t'(\mathbf{S}) + \Delta_t''(\mathbf{S})^2 + 2\nu(\mathbf{A},\mathbf{B})\cdot\Delta_t''(\mathbf{S}),$$

where we have made use of the simple bound $|\mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)]| \le \|\mathbf{A}^T\mathbf{B}\|_\infty \le \nu(\mathbf{A},\mathbf{B})$. The following lemma establishes tail bounds for $\Delta_t'(\mathbf{S})$ and $\Delta_t''(\mathbf{S})$, which lead to the statement of Proposition 4. ■

Lemma 6 *Suppose Assumption 1 (a) holds. Then, the event*

$$\Delta_t'(\mathbf{S}) \;\le\; \frac{c}{\sqrt{t}}\cdot\nu(\mathbf{A},\mathbf{B})^2\cdot\log\big(\mathsf{card}(\mathscr{F})\big) \qquad \text{(i)}$$

occurs with probability at least $1-\frac{1}{t}$*, and the event*

$$\Delta_t''(\mathbf{S}) \;\le\; \frac{c}{\sqrt{t}}\cdot\nu(\mathbf{A},\mathbf{B})\cdot\sqrt{\log\big(\mathsf{card}(\mathscr{F})\big)} \qquad \text{(ii)}$$

occurs with probability at least $1-\frac{1}{dd'}$.

[‖] Note that $Q_{i,\mathbf{j},\mathbf{k}}$ is a multivariate polynomial of degree-4 in the variables $S_{i,j}$, and so techniques based on moment generating functions, like Chernoff bounds, are not generally applicable to controlling $Q_{i\,\mathbf{j},\mathbf{k}}$. For instance, if $X\sim\mathcal{N}(0,1)$, then the variable X^4 does not have a moment generating function. Handling this obstacle is a notable aspect of our analysis.

Proof of Lemma 6 (i). Let $p>2$. Due to part (3) of Lemma 9 in Appendix F, we have

$$\|\Delta_t'(\mathbf{S})\|_p \;\le\; \big(\mathsf{card}(\mathscr{F})^2\big)^{1/p}\cdot\max_{(\mathbf{j},\mathbf{k})\in\mathscr{J}\times\mathscr{J}}\Big\|\tfrac{1}{t}\textstyle\sum_{i=1}^{t} Q_{i,\mathbf{j},\mathbf{k}}\Big\|_p. \qquad (44)$$

Note that each variable $Q_{i,\mathbf{j},\mathbf{k}}$ has moments of all orders, and when $\mathbf{j}$ and $\mathbf{k}$ are held fixed, the sequence $\{Q_{i,\mathbf{j},\mathbf{k}}\}_{1\le i\le t}$ is i.i.d. For this reason, it is natural to use Rosenthal's inequality to bound the L_p norm of the right side of the previous line. Specifically, the version of Rosenthal's inequality[**] stated in Lemma 10 in Appendix F leads to

$$\Big\|\tfrac{1}{t}\textstyle\sum_{i=1}^{t} Q_{i,\mathbf{j},\mathbf{k}}\Big\|_p \le c\cdot\frac{p/\log(p)}{t}\cdot\max\Big\{\Big\|\textstyle\sum_{i=1}^{t} Q_{i,\mathbf{j},\mathbf{k}}\Big\|_2,\,\big(\textstyle\sum_{i=1}^{t}\|Q_{i,\mathbf{j},\mathbf{k}}\|_p^p\big)^{1/p}\Big\}. \qquad (45)$$

The L_2 norm on the right side of Rosenthal's inequality (45) satisfies the bound

$$\begin{aligned}\Big\|\textstyle\sum_{i=1}^{t} Q_{i,\mathbf{j},\mathbf{k}}\Big\|_2 &= \sqrt{\mathsf{var}\big(\textstyle\sum_{i=1}^{t} Q_{i,\mathbf{j},\mathbf{k}}\big)} \\ &= \sqrt{t}\sqrt{\mathsf{var}(Q_{1,\mathbf{j},\mathbf{k}})} \\ &= \sqrt{t}\sqrt{\mathsf{var}\big(f_{\mathbf{j}}(\mathbf{D}_1)\,f_{\mathbf{k}}(\mathbf{D}_1)\big)} \\ &\le \sqrt{t}\Big\|f_{\mathbf{j}}(\mathbf{D}_1)\,f_{\mathbf{k}}(\mathbf{D}_1)\Big\|_2 \\ &\le \sqrt{t}\big\|f_{\mathbf{j}}(\mathbf{D}_1)\big\|_4\cdot\big\|f_{\mathbf{k}}(\mathbf{D}_1)\big\|_4 \qquad \text{(Cauchy-Schwarz)} \\ &\le c\sqrt{t}\big\|f_{\mathbf{j}}(\mathbf{D}_1)\big\|_{\psi_1}\cdot\big\|f_{\mathbf{k}}(\mathbf{D}_1)\big\|_{\psi_1} \qquad \text{(Lemma 9)} \\ &\le c\,\sqrt{t}\nu(\mathbf{A},\mathbf{B})^2,\end{aligned}$$

where the last step follows from the fact

$$\|f_{\mathbf{j}}(\mathbf{D}_1)\|_{\psi_1} \le c\,\nu(\mathbf{A},\mathbf{B}), \qquad (46)$$

obtained in the bounds (32) through (33).

Next, to handle the L_p norms in the bound (45), observe that

$$\begin{aligned}\|Q_{1,\mathbf{j},\mathbf{k}}\|_p &\le \big\|f_{\mathbf{j}}(\mathbf{D}_1)f_{\mathbf{k}}(\mathbf{D}_1)\big\|_p + \big|\mathbb{E}\big[f_{\mathbf{j}}(\mathbf{D}_1)f_{\mathbf{k}}(\mathbf{D}_1)\big]\big| \\ &\le 2\big\|f_{\mathbf{j}}(\mathbf{D}_1)\big\|_{2p}\cdot\big\|f_{\mathbf{k}}(\mathbf{D}_1)\big\|_{2p} \qquad \text{(Cauchy-Schwarz)} \\ &\le c p^2\big\|f_{\mathbf{j}}(\mathbf{D}_1)\big\|_{\psi_1}\cdot\big\|f_{\mathbf{k}}(\mathbf{D}_1)\big\|_{\psi_1} \qquad \text{(Lemma 9 in Appendix F)} \\ &\le c p^2\,\nu(\mathbf{A},\mathbf{B})^2 \qquad \text{(inequality (46))}.\end{aligned}$$

Hence, the second term in the Rosenthal bound (45) satisfies

$$\big(\textstyle\sum_{i=1}^{t}\|Q_{i,\mathbf{j},\mathbf{k}}\|_p^p\big)^{1/p} \;\le\; c\cdot p^2\cdot t^{1/p}\cdot\nu(\mathbf{A},\mathbf{B})^2,$$

[**] Here we are using the version of Rosenthal's inequality with the optimal dependence on p. It is a notable aspect of our argument that it makes essential use of this scaling in p.

and as long as the first term in the Rosenthal bound dominates[††], i.e.

$$p^2\, t^{1/p} \lesssim t^{1/2} \tag{47}$$

then we conclude that for any $\mathbf{j}$ and $\mathbf{k}$,

$$\Big\| \tfrac{1}{t} \textstyle\sum_{i=1}^t Q_{i,\mathbf{j},\mathbf{k}} \Big\|_p \;\leq\; \tfrac{c\cdot(p/\log(p))\cdot \nu(\mathbf{A},\mathbf{B})^2}{\sqrt{t}}.$$

Since the previous bound does not depend on $\mathbf{j}$ or $\mathbf{k}$, combining it with the first step in line (44) leads to

$$\big\|\Delta_t'(\mathbf{S})\big\|_p \;\leq\; c\cdot(p/\log(p))\cdot \mathsf{card}(\mathscr{F})^{2/p}\cdot \tfrac{\nu(\mathbf{A},\mathbf{B})^2}{\sqrt{t}}.$$

Next, we convert this norm bound into a tail bound. Specifically, if we consider the value

$$x_p \;:=\; c\cdot(p/\log(p))\cdot \mathsf{card}(\mathscr{F})^{2/p}\cdot \tfrac{\nu(\mathbf{A},\mathbf{B})^2}{\sqrt{t}}\cdot t^{1/p}$$

then Markov's inequality gives

$$\mathbb{P}\big(\Delta_t'(\mathbf{S}) \;\geq\; x_p\big) \;\leq\; \tfrac{\|\Delta_t'(\mathbf{S})\|_p^p}{x_p^p} \leq \tfrac{1}{t}.$$

Considering the choice of p given by

$$p = \log(\mathsf{card}(\mathscr{F})),$$

and noting that $\mathsf{card}(\mathscr{F})^{1/p} = e$, it follows that under this choice of p,

$$x_p \;\leq\; \Big(\tfrac{c\cdot\nu(\mathbf{A},\mathbf{B})^2\cdot\log(\mathsf{card}(\mathscr{F}))}{\sqrt{t}}\Big)\cdot\Big(\tfrac{t^{1/p}}{\log(p)}\Big).$$

Moreover, as long as $t \lesssim \mathsf{card}(\mathscr{F})^\kappa$ for some absolute constant $\kappa \geq 1$ (which holds under Assumption 1), then the last factor on the right satisfies

$$\big(\tfrac{t^{1/p}}{\log(p)}\big) \leq \tfrac{(\mathsf{card}(\mathscr{F})^{1/p})^\kappa}{\log(p)} = \tfrac{e^\kappa}{\log(\log(\mathsf{card}(\mathscr{F}))} \lesssim 1.$$

So, combining the last few steps, there is an absolute constant c such that

$$\mathbb{P}\Big(\Delta_t'(\mathbf{S}) \;\geq\; \tfrac{c\cdot\nu(\mathbf{A},\mathbf{B})^2\cdot\log(\mathsf{card}(\mathscr{F}))}{\sqrt{t}}\Big) \;\leq\; \tfrac{1}{t},$$

as needed. ■

[††]Under the choice of $p = \log(\mathsf{card}(\mathscr{F})) = \log(2dd')$ that will be made at the end of this argument, it is straightforward to check that the condition (47) holds under Assumption 1.

Proof of Lemma 6 (ii). Note that for each $i \in [t]$ and $\mathbf{j} \in \mathscr{J}$, we have

$$f_{\mathbf{j}}(\mathbf{D}_i) - \mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_i)] \;=\; f_{\mathbf{j}}(\mathbf{M}_i) \;=\; \mathbf{s}_i^T(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T)\mathbf{s}_i - \mathrm{tr}(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T), \tag{48}$$

which is a centered sub-Gaussian quadratic form. Due to the bound (35), we have

$$\Big\| f_{\mathbf{j}}(\mathbf{D}_i) - \mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_i)] \Big\|_{\psi_1} \;\leq\; c\,\nu(\mathbf{A},\mathbf{B}). \tag{49}$$

Furthermore, this can be combined with a standard concentration bound for sums of independent sub-exponential random variables (Lemma 12) to show that for any $r \geq 0$,

$$\mathbb{P}\Big(\Big|\tfrac{1}{t}\textstyle\sum_{i=1}^t f_{\mathbf{j}}(\mathbf{D}_i) - \mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_i)]\Big| \;\geq\; r\,\nu(\mathbf{A},\mathbf{B})\Big) \;\leq\; 2\exp\Big(-c\cdot t\cdot\min(r^2,r)\Big). \tag{50}$$

Hence, taking a union bound over all $\mathbf{j}$ gives

$$\mathbb{P}\Big(\Delta_t''(\mathbf{S}) \geq\; r\,\nu(\mathbf{A},\mathbf{B})\Big) \;\leq\; 2\exp\Big(\log(\mathsf{card}(\mathscr{F})) - c\cdot t\cdot\min(r^2,r)\Big). \tag{51}$$

Regarding the choice of r, note that by Assumption 1, we have $\frac{1}{\sqrt{t}}\sqrt{\log(\mathsf{card}(\mathscr{F}))} \lesssim 1$. It follows that there is a sufficiently large absolute constant $c_1 > 0$ such that if we put

$$r = \tfrac{c_1}{\sqrt{t}}\sqrt{\log(\mathsf{card}(\mathscr{F}))},$$

then

$$c\,t\,\min(r^2,r) \geq 2\log(\mathsf{card}(\mathscr{F})),$$

where c is the same as in the bound (51). In turn, this implies

$$\mathbb{P}\Big(\Delta_t''(\mathbf{S}) \;\geq\; \tfrac{c_1}{\sqrt{t}}\sqrt{\log(\mathsf{card}(\mathscr{F}))}\cdot\nu(\mathbf{A},\mathbf{B})\Big) \;\leq\; 2\exp(-\log(\mathsf{card}(\mathscr{F})) \;=\; \frac{1}{dd'}, \tag{52}$$

as desired. ■

Appendix D. Proof of Propositions 3 and 4 in case (b) (length sampling)

In order to carry out the proof Propositions 3 and 4 in the case of length sampling (Assumption 1 (b)), there are only two bounds that need to be updated. Namely, we must derive new bounds on $\|f_{\mathbf{j}}(\mathbf{D}_1) - \mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)]\|_{\psi_1}$ and $\|f_{\mathbf{j}}(\mathbf{D}_1)\|_{\psi_1}$ in order to account for the new distributional assumptions in case (b). Both of the new bounds will turn out to be of order $\|\mathbf{A}\|_F\|\mathbf{B}\|_F$, and consequently, the result of the propositions in case (b) will have the same form as in case (a), but with $\|\mathbf{A}\|_F\|\mathbf{B}\|_F$ replacing $\nu(\mathbf{A},\mathbf{B})$.

To derive the bound on $\|f_{\mathbf{j}}(\mathbf{D}_1) - \mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)]\|_{\psi_1}$, first note that

$$\begin{aligned}|\mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)]| &= |\mathrm{tr}(\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\mathbf{B})| \\ &\leq \|\mathbf{A}\|_F\|\mathbf{B}\|_F.\end{aligned}$$

Consequently,

$$\begin{aligned}\|f_{\mathbf{j}}(\mathbf{D}_1) - \mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)]\|_{\psi_1} &\leq \|f_{\mathbf{j}}(\mathbf{D}_1)\|_{\psi_1} + \|\mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)]\|_{\psi_1} \\ &\leq \|f_{\mathbf{j}}(\mathbf{D}_1)\|_{\psi_1} + c\|\mathbf{A}\|_F\|\mathbf{B}\|_F.\end{aligned} \tag{53}$$

Hence, it remains to show that $\|f_{\mathbf{j}}(\mathbf{D}_1)\|_{\psi_1} \leq c\|\mathbf{A}\|_F\|\mathbf{B}\|_F$, which is the content of Lemma 7 below. ■

Lemma 7 *If* $\mathbf{S}$ *is generated by length sampling with the probabilities in line* (5)*, then for any* $\mathbf{j} \in \mathscr{J}$*, we have the bound*

$$\big\|f_{\mathbf{j}}(\mathbf{D}_1)\big\|_{\psi_1} \leq 2\|\mathbf{A}\|_F\|\mathbf{B}\|_F. \tag{54}$$

Proof By the definition of the ψ_1-Orlicz norm, it suffices to find a value of $r > 0$ so that $\mathbb{E}\big[\exp\big(|f_{\mathbf{j}}(\mathbf{D}_1)|/r\big)\big]$ is at most 2. Due to the Cauchy-Schwarz inequality, the non-zero length-sampling probabilities p_l satisfy

$$\frac{1}{p_l} \leq \frac{\sqrt{\sum_{j=1}^n \|\mathbf{e}_j^T\mathbf{A}\|_2^2}\sqrt{\sum_{j=1}^n \|\mathbf{e}_j^T\mathbf{B}\|_2^2}}{\|\mathbf{e}_l^T\mathbf{A}\|_2\|\mathbf{e}_l^T\mathbf{B}\|_2} = \frac{\|\mathbf{A}\|_F\|\mathbf{B}\|_F}{\|\mathbf{e}_l^T\mathbf{A}\|_2\|\mathbf{e}_l^T\mathbf{B}\|_2}.$$

Consequently, for each $r > 0$ we have

$$\begin{aligned}\mathbb{E}\Big[\exp\Big(\tfrac{|f_{\mathbf{j}}(\mathbf{D}_1)|}{r}\Big)\Big] &= \sum_{l\in[n]:\, p_l>0} p_l \cdot \exp\Big(\tfrac{1}{r}\big|f_{\mathbf{j}}(\tfrac{1}{p_l}\mathbf{A}^T\mathbf{e}_l\mathbf{e}_l^T\mathbf{B})\big|\Big) \\ &\leq \max_{l\in[n]:\, p_l>0} \exp\Big(\tfrac{1}{r}\tfrac{1}{p_l}\big|f_{\mathbf{j}}(\mathbf{A}^T\mathbf{e}_l\mathbf{e}_l^T\mathbf{B})\big|\Big) \\ &\leq \max_{l\in[n]} \exp\Big(\tfrac{1}{r}\|\mathbf{A}\|_F\|\mathbf{B}\|_F\big|\tfrac{\mathbf{e}_l^T\mathbf{B}}{\|\mathbf{e}_l^T\mathbf{B}\|_2}\mathbf{C}_{\mathbf{j}}^T\tfrac{\mathbf{A}^T\mathbf{e}_l}{\|\mathbf{A}^T\mathbf{e}_l\|_2}\big|\Big) \\ &\leq \exp\Big(\tfrac{1}{r}\|\mathbf{A}\|_F\|\mathbf{B}\|_F\ \|\mathbf{C}_{\mathbf{j}}\|_2\Big) \\ &= \exp\Big(\tfrac{1}{r}\|\mathbf{A}\|_F\|\mathbf{B}\|_F\Big).\end{aligned}$$

Hence, if we take $r = 2\|\mathbf{A}\|_F\|\mathbf{B}\|_F$, then the right hand side is at most $e^{1/2} \leq 2$. ■

Appendix E. Proof of Propositions 3 and 4 in case (c) (SRHT)

The steps needed to extend the propositions in the case of SRHT matrices follows the same pattern as in case (b). However, there is a small subtlety insofar as all of the analysis is done conditionally on the matrix of signs $\mathbf{D}_n^\circ$ in the product $\mathbf{S} = \mathbf{P}_n(\frac{1}{\sqrt{n}}\mathbf{H}_n)\mathbf{D}_n^\circ$. Hence, it suffices to bound the ψ_1 Orlicz norm of $f_{\mathbf{j}}(\mathbf{D}_1)$ conditionally on $\mathbf{D}_n^\circ$, as well as the conditional expectation $|\mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)|\mathbf{D}_n^\circ]$. Regarding the conditional expectation, it can be checked that $\mathbb{E}[\mathbf{S}^T\mathbf{S}|\mathbf{D}_n^\circ] = \mathbf{I}_n$, and it follows that

$$|\mathbb{E}[f_{\mathbf{j}}(\mathbf{D}_1)|\mathbf{D}_n^\circ]| = |\mathrm{tr}(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T)| \leq \nu(\mathbf{A},\mathbf{B}).$$

Since we are not aware of a standard notation for a conditional Orlicz norm, we define

$$\big\|f_{\mathbf{j}}(\mathbf{D}_1)\big|\mathbf{D}_n^\circ\big\|_{\psi_1} := \inf\Big\{r > 0 \;\Big|\; \mathbb{E}\big[\psi_1\big(|f_{\mathbf{j}}(\mathbf{D}_1)|/r\big)\big|\mathbf{D}_n^\circ\big] \leq 1\Big\},$$

which is a random variable, since it is a function of $\mathbf{D}_n^\circ$. The following lemma provides a bound on this quantity, which turns out to be of order $\log(n)\,\nu(\mathbf{A},\mathbf{B})$. For this reason, the SRHT case (c) of Propositions 3 and 4 will have the same form as case (a), but with $\log(n)\,\nu(\mathbf{A},\mathbf{B})$ replacing $\nu(\mathbf{A},\mathbf{B})$. ■

Lemma 8 *If* $\mathbf{S}$ *is an SRHT matrix, then the following bound holds with probability at least* $1 - c/n$*,*

$$\big\|f_{\mathbf{j}}(\mathbf{D}_1)\big|\mathbf{D}_n^\circ\big\|_{\psi_1} \leq c \cdot \log(n) \cdot \nu(\mathbf{A},\mathbf{B}). \tag{55}$$

Proof By the definition of the conditional ψ_1-Orlicz norm, it suffices to find a value of $r > 0$ so that $\mathbb{E}\big[\exp\big(|f_{\mathbf{j}}(\mathbf{D}_1)|/r\big)\big|\mathbf{D}_n^\circ\big]$ is at most 2 (with the stated probability).

For an SRHT matrix $\mathbf{S} = \mathbf{P}\frac{1}{\sqrt{n}}\mathbf{H}_n\mathbf{D}_n^\circ$, recall that the rows of $\sqrt{t}\mathbf{P}$ are sampled uniformly at random from the set $\{\frac{1}{\sqrt{1/n}}\mathbf{e}_1, \ldots, \frac{1}{\sqrt{1/n}}\mathbf{e}_n\}$. It follows that

$$\begin{aligned}\mathbb{E}\Big[\exp\Big(\tfrac{|f_{\mathbf{j}}(\mathbf{D}_1)|}{r}\Big)\Big|\mathbf{D}_n^\circ\Big] &= \mathbb{E}\Big[\exp\Big(\tfrac{1}{r}\big|\mathrm{tr}(\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\mathbf{s}_1\mathbf{s}_1^T\mathbf{B})\big|\Big)\Big|\mathbf{D}_n^\circ\Big] \\ &= \frac{1}{n}\sum_{l=1}^n \exp\Big(\tfrac{1}{r}\big|\mathbf{e}_l^T(\mathbf{H}_n\mathbf{D}_n^\circ)\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T(\mathbf{D}_n^\circ\mathbf{H}_n^T)\mathbf{e}_l\big|\Big).\end{aligned}$$

Next, let $\boldsymbol{\varepsilon}_l \in \mathbb{R}^n$ be the lth row of $\mathbf{H}_n\mathbf{D}_n^\circ$, which gives

$$\begin{aligned}\mathbb{E}\Big[\exp\Big(\tfrac{|f_{\mathbf{j}}(\mathbf{D}_1)|}{r}\Big)\Big|\mathbf{D}_n^\circ\Big] &= \frac{1}{n}\sum_{l=1}^n \exp\Big(\big|\tfrac{\boldsymbol{\varepsilon}_l^T(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T)\boldsymbol{\varepsilon}_l}{r}\big|\Big) \\ &\leq \exp\Big(\max_{l\in[n]}\big|\tfrac{\boldsymbol{\varepsilon}_l^T(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T)\boldsymbol{\varepsilon}_l}{r}\big|\Big).\end{aligned} \tag{56}$$

Recalling that $\mathbf{D}_n^\circ = \mathsf{diag}(\boldsymbol{\varepsilon})$ where $\boldsymbol{\varepsilon} \in \mathbb{R}^n$ is a vector of i.i.d. Rademacher variables, and that all entries of $\mathbf{H}_n$ are ± 1, it follows that $\boldsymbol{\varepsilon}_l$ has the same distribution as $\boldsymbol{\varepsilon}$ for each $l \in [n]$. Consequently, each quadratic form $\boldsymbol{\varepsilon}_l^T(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T)\boldsymbol{\varepsilon}_l$ concentrates around $\mathrm{tr}(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T)$, and we can use a union bound to control the maximum of these quadratic forms. Note also that the matrix $\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T$ is rank-1, and so $\|\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\|_2^2 = \|\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\|_F^2$, Hence, by choosing the parameter u to be proportional to $\log(n)\cdot\|\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\|_F$ in the Hanson-Wright inequality (Lemma 11), and using a union bound, there is an absolute constant $c > 0$ such that

$$\mathbb{P}\Big(\max_{l\in[n]}\ \big|\boldsymbol{\varepsilon}_l^T(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T)\boldsymbol{\varepsilon}_l\big| \ \geq\ \mathrm{tr}(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T) + c\log(n)\|\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\|_F\Big) \ \leq\ \tfrac{c}{n}. \tag{57}$$

Furthermore, noting that $\mathrm{tr}(\mathbf{B}\mathbf{C}_j^T\mathbf{A}^T)$ and $\|\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T\|_F$ are both at most $\nu(\mathbf{A},\mathbf{B})$, we have

$$\mathbb{P}\Big(\max_{l\in[n]}\ \big|\boldsymbol{\varepsilon}_l^T(\mathbf{B}\mathbf{C}_{\mathbf{j}}^T\mathbf{A}^T)\boldsymbol{\varepsilon}_l\big| \ \geq\ 2c\log(n)\nu(\mathbf{A},\mathbf{B})\Big) \ \leq\ \tfrac{c}{n}. \tag{58}$$

Finally, this means that if we take $r = 4c\log(n)\nu(\mathbf{A},\mathbf{B})$ in the bound (56), then the event

$$\mathbb{E}\big[\exp\big(|f_{\mathbf{j}}(\mathbf{D}_1)|/r\big)\big|\mathbf{D}_n^{\circ}\big] \leq e^{1/2}$$

holds with probability at least $1-\frac{c}{n}$, which completes the proof, since $e^{1/2} \leq 2$. ■

Appendix F. Technical Lemmas

Lemma 9 (Facts about Orlicz norms) *Orlicz norms have the following properties, where c, c_1, and c_2 are positive absolute constants.*

1. *For any random variable X, and any $p \geq 1$,*

$$\|X\|_p \leq cp\|X\|_{\psi_1} \tag{59}$$

$$\|X\|_p \leq c\sqrt{p}\|X\|_{\psi_2} \tag{60}$$

$$\|X\|_{\psi_1} \leq c\|X\|_{\psi_2}. \tag{61}$$

2. *If $X \sim \mathcal{N}(0,\sigma^2)$, then $\|X\|_{\psi_2} \leq c\,\sigma$.*

3. *Let $p \geq 1$. For any sequence of random variables $X_1,\ldots,X_d$,*

$$\Big\|\max_{1\leq j\leq d} X_j\Big\|_p \leq d^{1/p}\max_{1\leq j\leq d}\|X_j\|_p$$

and

$$\Big\|\max_{1\leq j\leq d} X_j\Big\|_{\psi_1} \leq c\log(d)\max_{1\leq j\leq d}\|X_j\|_{\psi_1}$$

4. *Let X be any random variable. Then, for any $x>0$ and $p\geq 1$, we have*

$$\mathbb{P}\big(|X|\geq x\big) \leq \Big(\tfrac{\|X\|_p}{x}\Big)^p.$$

and

$$\mathbb{P}\big(|X|\geq x\big) \leq c_1 e^{-c_2 x/\|X\|_{\psi_1}}.$$

Proof In part 1, line (59) follows from line 5.11 of Vershynin (2012), line (60) follows from definition 5.13 of Vershynin (2012), and line (61) follows from p.94 of van der Vaart and Wellner (1996). Next, part 2 follows from the definition of the ψ_2-Orlicz norm and the moment generating function for $\mathcal{N}(0,\sigma^2)$. Part 3 is due to Lemma 2.2.2 of van der Vaart and Wellner (1996). Lastly, part 4 follows from Markov's inequality and line 5.14 of Vershynin (2012). ■

Lemma 10 (Rosenthal's inequality with best constants) *Fix any number $p>2$. Let $Y_1,\ldots,Y_t$ be independent random variables with $\mathbb{E}[Y_i]=0$ and $\mathbb{E}\big[|Y_i|^p\big]<\infty$ for all $1\leq i\leq t$. Then,*

$$\big\|\textstyle\sum_{i=1}^t Y_i\big\|_p \leq c\big(\frac{p}{\log(p)}\big)\cdot\max\Big\{\big\|\textstyle\sum_{i=1}^t Y_i\big\|_2,\ \big(\textstyle\sum_{i=1}^t \big\|Y_i\|_p^p\big)^{1/p}\Big\}. \tag{62}$$

Proof See the paper Johnson et al. (1985). The statement above differs slightly from the Theorem 4.1 in the paper Johnson et al. (1985), which requires symmetric random variables, but the remark on p.247 of that paper explains why the variables $Y_1,\ldots,Y_t$ need not be symmetric as long as they have mean 0. ■

Lemma 11 (Hanson-Wright inequality) *Let $\mathbf{x}=(X_1,\ldots,X_n)$ be a vector of independent sub-Gaussian random variables with $\mathbb{E}[X_j]=0$, and $\|X_j\|_{\psi_2}\leq\kappa$ for all $1\leq j\leq n$. Also, let $\mathbf{H}\in\mathbb{R}^{n\times n}$ be any fixed non-zero matrix. Then, there is an absolute constant $c>0$ such that for any $u\geq 0$,*

$$\mathbb{P}\Big(\big|\mathbf{x}^T\mathbf{H}\mathbf{x}-\mathbb{E}[\mathbf{x}^T\mathbf{H}\mathbf{x}]\big| \ \geq\ u\Big) \ \leq\ 2\exp\Big(-c\cdot\min\Big\{\tfrac{u^2}{\kappa^2\|\mathbf{H}\|_F^2},\tfrac{u}{\kappa\|\mathbf{H}\|_2}\Big\}\Big). \tag{63}$$

Proof See the paper Rudelson and Vershynin (2013). ■

Lemma 12 (Bernstein inequality for sub-exponential variables) *Let $Y_1,\ldots,Y_t$ be independent random variables with $\mathbb{E}[Y_i]=0$ and $\|Y_i\|_{\psi_1}\leq\kappa$ for all $1\leq i\leq t$. Then, there is an absolute constant $c>0$, such that for any $u\geq 0$,*

$$\mathbb{P}\Big(\Big|\tfrac{1}{t}\textstyle\sum_{i=1}^t Y_i\Big|\geq\kappa\cdot u\Big) \ \leq\ 2\exp\Big(-c\cdot t\cdot\min(u^2,u)\Big). \tag{64}$$

Proof See Proposition 16 in Vershynin (2012). ■

Lemma 13 ((Chernozhukov et al., 2016)) *If η is a non-negative random variable, and there are numbers $a,b>0$ such that*

$$\mathbb{P}(\eta>x)\leq ae^{-x/b},$$

for all $x>0$, then the following bound holds for all $r>0$,

$$\mathbb{E}[\eta^3\cdot 1\{\eta>r\}]\leq 6a(r+b)^3e^{-r/b}.$$

Proof See Lemma 6.6 in Chernozhukov et al. (2016). ■

Remark. The following lemma may be of independent interest, since it provides an explicit bound on the ψ_1-Orlicz norm of a centered sub-Gaussian quadratic form. Although this bound follows from the Hanson-Wright inequality, we have not seen it stated in the literature.

Lemma 14 *Let $\mathbf{x} = (X_1, \ldots, X_n)$, be independent random variables satisfying $\mathbb{E}[X_j] = 0$ and $\|X_j\|_{\psi_2} \leq \kappa$ for all $1 \leq j \leq n$. Also, let $\mathbf{H} \in \mathbb{R}^{n \times n}$ be a non-zero fixed matrix. Then, there is an absolute constant $c > 0$ such that*

$$\Big\| \mathbf{x}^T\mathbf{H}\mathbf{x} - \mathbb{E}[\mathbf{x}^T\mathbf{H}\mathbf{x}] \Big\|_{\psi_1} \;\leq\; c\kappa^2 \tfrac{\|\mathbf{H}\|_F^2}{\|\mathbf{H}\|_2}.$$

Proof Define the random variable $Q := \mathbf{x}^T\mathbf{H}\mathbf{x} - \mathbb{E}[\mathbf{x}^T\mathbf{H}\mathbf{x}]$. By the definition of the ψ_1-Orlicz norm, it suffices to find a value $r > 0$ such that $\mathbb{E}\big[\exp(|Q|/r)\big] \leq 2$. Using the tail-sum formula, and the change of variable $v = e^{u/r}$, we have

$$\begin{aligned}\mathbb{E}\big[\exp(|Q|/r)\big] &\leq 1 + \int_1^\infty \mathbb{P}(\exp(|Q|/r) > v) dv \\ &= 1 + \tfrac{1}{r}\int_0^\infty \mathbb{P}(|Q| > u) \cdot e^{u/r} du.\end{aligned}$$

Next, we employ the Hanson-Wright inequality (Lemma 11). By considering the "threshold" $u^* := \kappa^2 \frac{\|\mathbf{H}\|_F^2}{\|\mathbf{H}\|_2}$, it is helpful to note that the quantities in the exponent of the Hanson-Wright inequality satisfy $\frac{u^2}{\kappa^4\|\mathbf{H}\|_F^2} \leq \frac{u}{\kappa^2\|\mathbf{H}\|_2}$ if and only if $u \leq u^*$. Hence,

$$\begin{aligned}\mathbb{E}\big[\exp(|Q|/r)\big] &\leq 1 + \tfrac{1}{r}\int_0^\infty \exp\Big\{ - c\min\big(\tfrac{u^2}{\kappa^4\|\mathbf{H}\|_F^2}, \tfrac{u}{\kappa^2\|\mathbf{H}\|_2}\big)\Big\} \cdot e^{u/r} du \\ &\leq 1 + \tfrac{1}{r}\int_0^{u^*} e^{u/r} du + \tfrac{1}{r}\int_{u^*}^\infty \exp\Big\{ - u\big(\tfrac{c}{\kappa^2\|\mathbf{H}\|_2} - \tfrac{1}{r}\big)\Big\} du.\end{aligned}$$

Evaluating the last two integrals directly, if we let $C' := \frac{c}{\kappa^2\|\mathbf{H}\|_2} - \frac{1}{r}$ and choose r so that $C' > 0$, then

$$\begin{aligned}\mathbb{E}\big[\exp(|Q|/r)\big] &\leq e^{u^*/r} + \tfrac{1}{rC'} e^{-u^*C'}, \\ &\leq e^{u^*/r} + \tfrac{1}{\frac{cr}{\kappa^2\|\mathbf{H}\|_2} - 1}.\end{aligned}$$

Note that the condition $C' > 0$ means that it is necessary to have $r > \frac{1}{c}\kappa^2\|\mathbf{H}\|_2$. To finish the argument, we further require that r is large enough so that (say)

$$\tfrac{u^*}{r} \leq \tfrac{1}{4} \quad \text{and} \quad \tfrac{cr}{\kappa^2\|\mathbf{H}\|_2} \geq 3, \tag{65}$$

which ensures

$$\mathbb{E}\big[\exp(|Q|/r)\big] \leq e^{1/4} + \tfrac{1}{2} < 2,$$

as desired. Note that the constraints (65) are the same as

$$r \geq 4\kappa^2 \tfrac{\|\mathbf{H}\|_F^2}{\|\mathbf{H}\|_2} \quad \text{and} \quad r \geq \tfrac{3}{c}\kappa^2\|\mathbf{H}\|_2.$$

Due to the basic fact that $\|\mathbf{H}\|_2 \leq \|\mathbf{H}\|_F$ for all matrices $\mathbf{H}$, it follows that whenever $r \geq \max(4, \frac{3}{c})\kappa^2 \frac{\|\mathbf{H}\|_F^2}{\|\mathbf{H}\|_2}$, we have $\mathbb{E}\big[\exp(|Q|/r)\big] < 2$. ■

Remark. The following lemma is a basic fact about the d_{LP} metric, but may not be widely known, and so we give a proof. Recall also that we use the generalized inverse $F_V^{-1}(\alpha) := \inf\{z \in \mathbb{R} \,|\, F_V(z) \geq \alpha\}$, where F_V denotes the c.d.f. of V.

Lemma 15 *Fix $\alpha \in (0, 1/2)$ and suppose there is some $\epsilon \in (0, \alpha)$ such that random variables U and V satisfy*

$$d_{\text{LP}}(\mathcal{L}(U), \mathcal{L}(V)) \leq \epsilon.$$

Then, the quantiles of U and V satisfy

$$\big|F_U^{-1}(1-\alpha) - F_V^{-1}(1-\alpha)\big| \;\leq\; \psi_\alpha(\epsilon), \tag{66}$$

where the right side is defined as

$$\psi_\alpha(\epsilon) := F_U^{-1}(1-\alpha+\epsilon) - F_U^{-1}(1-\alpha-\epsilon) + \epsilon.$$

Proof Consider the Lévy metric, defined as

$$d_{\text{L}}(\mathcal{L}(U), \mathcal{L}(V)) := \inf\Big\{\epsilon > 0 \,\Big|\, F_U(x-\epsilon) - \epsilon \leq F_V(x) \leq F_U(x+\epsilon) + \epsilon \text{ for all } x \in \mathbb{R}\Big\}.$$

It is a fact that this metric is always dominated by the d_{LP} metric in the sense that

$$d_{\text{L}}(\mathcal{L}(U), \mathcal{L}(V)) \;\leq\; d_{\text{LP}}(\mathcal{L}(U), \mathcal{L}(V)),$$

for all scalar random variables U and V (Huber and Ronchetti, 2009, p.36). Based on the definition of the d_{L} metric, it is straightforward to check that the following inequalities hold under the assumption of the lemma,

$$F_U^{-1}(1-\alpha-\epsilon) - \epsilon \;\leq\; F_V^{-1}(1-\alpha) \;\leq\; F_U^{-1}(1-\alpha+\epsilon) + \epsilon.$$

(Specifically, consider the choices $x = F_V^{-1}(1-\alpha)$ and $x = F_U^{-1}(1-\alpha+\epsilon) + \epsilon$.) Next, if we subtract $F_U^{-1}(1-\alpha)$ from each side of the inequalities above, and note that $F_U^{-1}(\cdot)$ is non-decreasing, it follows that if we put $a = F_U^{-1}(1-\alpha+\epsilon) - F_U^{-1}(1-\alpha) + \epsilon$ and $b = F_U^{-1}(1-\alpha) - F_U^{-1}(1-\alpha-\epsilon) + \epsilon$, then

$$|F_V^{-1}(\alpha) - F_U^{-1}(\alpha)| \;\leq\; \max\{a, b\} \;\leq\; \psi_\alpha(\epsilon),$$

as needed. ■

Lemma 16 *Under Assumption 1 (a), the quantity*

$$\delta_0 = t^{-1/8}\log^{1/2}(d)\nu(\mathbf{A}, \mathbf{B})^{3/4}$$

satisfies conditions (30) *and* (31).

Proof Consider the number

$$\delta_1(r) := \tfrac{\log^2(d)\nu(\mathbf{A},\mathbf{B})}{\sqrt{t}} \cdot r,$$

where $r \geq 1$ is a free parameter to be adjusted. Based on the bound (29), it is easy to check that plugging $\delta_1(r)$ into $K_t(\cdot)$ and $J_t(\cdot)$ leads to

$$K_t(\delta_1(r)) + J_t(\delta_1(r)) \leq c \cdot \nu(\mathbf{A},\mathbf{B})^3 \cdot \log(d)^3 \cdot r^3 \cdot \exp(-r/c)$$

and if we take $r \geq c\log(\log(d)^4)$, then

$$K_t(\delta_1(r)) + J_t(\delta_1(r)) \leq c\,\nu(\mathbf{A},\mathbf{B})^3,$$

as desired in (30). Hence, as long as there is a choice of r satisfying

$$r \geq c\log(\log(d)^4) \qquad \text{and} \qquad \delta_1(r) = \delta_0,$$

then $\delta_1(r)$ will satisfy both of the desired constraints (30) and (31). Solving the equation $\delta_1(r) = \delta_0$ gives

$$r = t^{3/8} \cdot \log^{-3/2}(d) \cdot \nu(\mathbf{A},\mathbf{B})^{-1/4},$$

and then the condition $r \geq c\log(\log(d)^4)$ is the same as

$$\begin{aligned} t &\geq \Big(c\,\nu(\mathbf{A},\mathbf{B})^{1/4}\log(d)^{3/2} \cdot \log(\log(d)^4)\Big)^{8/3} \\ &= c\,\nu(\mathbf{A},\mathbf{B})^{2/3}\log(d)^4 \cdot \log(\log(d)^4)^{8/3}, \end{aligned} \tag{67}$$

which holds under Assumption 1 (a). ■

References

N. Ailon and B. Chazelle. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *Annual ACM Symposium on Theory of Computing (STOC)*, 2006.

N. Ailon and E. Liberty. Fast dimension reduction using Rademacher series on dual BCH codes. *Discrete & Computational Geometry*, 42(4):615–630, 2009.

S. Ar, M. Blum, B. Codenotti, and P. Gemmell. Checking approximate computations over the reals. In *Annual ACM Symposium on Theory of Computing (STOC)*, 1993.

H. Avron, P. Maymounkov, and S. Toledo. Blendenpik: Supercharging lapack's least-squares solver. *SIAM Journal on Scientific Computing*, 32(3):1217–1236, 2010.

C. Boutsidis and A. Gittens. Improved matrix algorithms via the subsampled randomized hadamard transform. *SIAM Journal on Matrix Analysis and Applications*, 34(3):1301–1340, 2013.

C. Brezinski and M. R. Zaglia. *Extrapolation methods: theory and practice*. Elsevier, 2013.

C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011. URL `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`.

J. Chang, W. Zhou, W.-X. Zhou, and L. Wang. Comparing large covariance matrices under weak conditions on the dependence structure and its application to gene clustering. *Biometrics*, 2016.

X. Chen. Gaussian and bootstrap approximations for high-dimensional u-statistics and their applications. *The Annals of Statistics*, 46(2):642–678, 2018.

V. Chernozhukov, D. Chetverikov, and K. Kato. Gaussian approximations and multiplier bootstrap for maxima of sums of high-dimensional random vectors. *The Annals of Statistics*, 41(6):2786–2819, 2013.

V. Chernozhukov, D. Chetverikov, and K. Kato. Gaussian approximation of suprema of empirical processes. *The Annals of Statistics*, 42(4):1564–1597, 2014.

V. Chernozhukov, D. Chetverikov, and K. Kato. Comparison and anti-concentration bounds for maxima of Gaussian random vectors. *Probability Theory and Related Fields*, 162(1-2):47–70, 2015.

V. Chernozhukov, D. Chetverikov, and K. Kato. Empirical and multiplier bootstraps for suprema of empirical processes of increasing complexity, and related Gaussian couplings. *Stochastic Processes and their Applications*, 2016.

V. Chernozhukov, D. Chetverikov, and K. Kato. Central limit theorems and bootstrap in high dimensions. *The Annals of Probability*, 45(4):2309–2352, 2017.

K. L. Clarkson and D. P. Woodruff. Low rank approximation and regression in input sparsity time. In *Annual ACM Symposium on theory of computing (STOC)*, 2013.

G. Dasarathy, P. Shah, B. Narayan Bhaskar, and R. D. Nowak. Sketching sparse matrices, covariances, and graphs via tensor products. *IEEE Transactions on Information Theory*, 61(3):1373–1388, 2015.

J. Demmel, I. Dumitriu, O. Holtz, and R. Kleinberg. Fast matrix multiplication is stable. *Numerische Mathematik*, 106(2):199–224, 2007.

J. D. Dixon. Estimating extremal eigenvalues and condition numbers of matrices. *SIAM Journal on Numerical Analysis*, 20(4):812–814, 1983.

P. Drineas and M. W. Mahoney. RandNLA: randomized numerical linear algebra. *Communications of the ACM*, 59(6):80–90, 2016.

P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006a.

P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Sampling algorithms for ℓ_2 regression and applications. In *Annual ACM-SIAM Symposium on Discrete Algorithm (SODA)*, 2006b.

P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, September 2008.

P. Drineas, M. W. Mahoney, S. Muthukrishnan, and T. Sarlós. Faster least squares approximation. *Numerische Mathematik*, 117(2):219–249, 2011.

P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13: 3475–3506, 2012.

A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

R. Freivalds. Fast probabilistic algorithms. *Mathematical Foundations of Computer Science*, pages 57–69, 1979.

N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

N. J. Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.

P. J. Huber and E. M. Ronchetti. *Robust Statistics*. Wiley, 2009.

W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics*, 26(189-206), 1984.

W. B. Johnson, G. Schechtman, and J. Zinn. Best constants in moment inequalities for linear combinations of independent and exchangeable random variables. *The Annals of Probability*, pages 234–253, 1985.

E. Liberty, F. Woolfe, P.-G. Martinsson, V. Rokhlin, and M. Tygert. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 104(51):20167–20172, 2007.

M. E. Lopes. Estimating the algorithmic variance of randomized ensembles via the bootstrap. *The Annals of Statistics*, 47(2):1088–1112, 2019.

M. E. Lopes, Z. Lin, and H.-G. Mueller. Bootstrapping max statistics in high dimensions: Near-parametric rates under weak variance decay and application to functional data analysis. *arXiv:1807.04429*, 2018a.

M. E. Lopes, S. Wang, and M. W. Mahoney. Error estimation for randomized least-squares algorithms via the bootstrap. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018b.

P. Ma, M. W. Mahoney, and B. Yu. A statistical perspective on algorithmic leveraging. In *International Conference on Machine Learning (ICML)*, 2014.

A. Magen and A. Zouzias. Low rank matrix-valued Chernoff bounds and approximate matrix multiplication. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2011.

M. W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011.

R. Pagh. Compressed matrix multiplication. *ACM Transactions on Computation Theory*, 5(3):9, 2013.

M. Pilanci and M. J. Wainwright. Newton sketch: a near linear-time optimization algorithm with linear-quadratic convergence. *SIAM Journal on Optimization*, 27(1):205–245, 2017.

F. Roosta-Khorasani and M. W. Mahoney. Sub-sampled Newton methods II: local convergence rates. *arXiv:1601.04738*, 2016.

M. Rudelson and R. Vershynin. Hanson-Wright inequality and sub-Gaussian concentration. *Electronic Communications in Probability*, 18:9 pp., 2013.

T. Sarlós. Improved approximation algorithms for large matrices via random projections. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006.

A. Sidi. *Practical Extrapolation Methods: Theory and Applications*. Cambridge University Press, 2003.

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes*. Springer, 1996.

R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. In *Compressed Sensing, Theory and Applications*. Cambridge University Press, 2012.

S. Wang. A practical guide to randomized matrix computations with MATLAB implementations. *arXiv:1505.07570*, 2015.

D. P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014.

F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert. A fast randomized algorithm for the approximation of matrices. *Applied and Computational Harmonic Analysis*, 25(3):335–366, 2008.

P. Xu, J. Yang, F. Roosta-Khorasani, C. Ré, and M. W. Mahoney. Sub-sampled Newton methods with non-uniform sampling. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3000–3008, 2016.

J. Yang, X. Meng, and M. W. Mahoney. Implementing randomized matrix algorithms in parallel and distributed environments. *Proceedings of the IEEE*, 104(1):58–92, 2016.

Journal of Machine Learning Research 20 (2019) 1-47 Submitted 9/17; Revised 2/19; Published 3/19

Stochastic Modified Equations and Dynamics of Stochastic Gradient Algorithms I: Mathematical Foundations

Qianxiao Li LIQIX@IHPC.A-STAR.EDU.SG
Institute of High Performance Computing
Agency for Science, Technology and Research
1 Fusionopolis Way, Connexis North, Singapore 138632

Cheng Tai CHENGTAI@PKU.EDU.CN
Beijing Institute of Big Data Research
and Peking University
Beijing, China, 100080

Weinan E WEINAN@MATH.PRINCETON.EDU
Princeton University
Princeton, NJ 08544, USA
Beijing Institute of Big Data Research
Beijing, China, 100080

Editor: Francis Bach

Abstract

We develop the mathematical foundations of the stochastic modified equations (SME) framework for analyzing the dynamics of stochastic gradient algorithms, where the latter is approximated by a class of stochastic differential equations with small noise parameters. We prove that this approximation can be understood mathematically as an weak approximation, which leads to a number of precise and useful results on the approximations of stochastic gradient descent (SGD), momentum SGD and stochastic Nesterov's accelerated gradient method in the general setting of stochastic objectives. We also demonstrate through explicit calculations that this continuous-time approach can uncover important analytical insights into the stochastic gradient algorithms under consideration that may not be easy to obtain in a purely discrete-time setting.

Keywords: stochastic gradient algorithms, modified equations, stochastic differential equations, momentum, Nesterov's accelerated gradient

1. Introduction

Stochastic gradient algorithms are often used to solve optimization problems of the form

$$\min_{x\in\mathbb{R}^d} \quad f(x) := \mathbb{E}f_\gamma(x) \tag{1.1}$$

where $\{f_r : r \in \Gamma\}$ is a family of functions from $\mathbb{R}^d$ to $\mathbb{R}$ and γ is a Γ-valued random variable, with respect to which the expectation is taken (these notions will be made precise in the following sections). For empirical loss minimization in supervised learning applications, γ is usually a uniform random variable taking values in $\Gamma = \{1, 2, \ldots, n\}$. In this case, f is the total empirical loss function and f_r, $r \in \Gamma$ are the loss function due to the r^{th} training sample. In this paper, we shall consider the general situation of a expectation over arbitrary index sets and distributions.

Solving (1.1) using the standard gradient descent (GD) on x gives the iteration scheme

$$x_{k+1} = x_k - \eta\nabla\mathbb{E}f_\gamma(x_k), \tag{1.2}$$

for $k \geq 0$ and η is a small positive step-size known as the learning rate. Note that this requires the evaluation of the gradient of an expectation, which can be costly (in this empirical risk minimization case, this happens when n is large). In its simplest form, the stochastic gradient descent (SGD) algorithm replaces the expectation of the gradient with a sampled gradient, i.e.

$$x_{k+1} = x_k - \eta\nabla f_{\gamma_k}(x_k), \tag{1.3}$$

where each γ_k is an independent and identically distributed (i.i.d.) random variable with the same distribution as γ. Under mild conditions, we then have $\mathbb{E}[\nabla f_{\gamma_k}(x_k)|x_k] = \nabla\mathbb{E}f(x_k)$. In other words, (1.3) is a sampled version of (1.2).

In the literature, many convergence results are available for SGD and its variants (Shamir and Zhang, 2013; Moulines and Bach, 2011; Needell et al., 2014; Xiao and Zhang, 2014; Shalev-Shwartz and Zhang, 2014; Bach and Moulines, 2013; Défossez and Bach, 2015). However, it is often the case that different analysis techniques must be adopted for different variants of the algorithms and there generally lacked a systematic approach to study their precise dynamical properties. In Li et al. (2015), a general approach was introduced to address this problem, in which discrete-time stochastic gradient algorithms are approximated by continuous-time stochastic differential equations with the noise term depending on a small parameter (the learning rate). This can be viewed as a generalization of the method of modified equations (Hirt, 1968; Noh and Protter, 1960; Daly, 1963; Warming and Hyett, 1974) to the stochastic setting, and allows one to employ tools from stochastic calculus to systematically analyze the dynamics of stochastic gradient algorithms. The stochastic modified equations (SME) approach was further developed in Li et al. (2017), where a weak approximation result for the SGD was proved in a finite-sum-objective setting.

The present series of papers builds on the earlier work of Li et al. (2015, 2017) and aims to establish the framework of stochastic modified equations and their applications in greater generality and depth, and highlight the advantages of this systematic framework for studying stochastic gradient algorithms using continuous-time methods. As the first in the series, this paper will focus on mathematical aspects, namely the main approximation theorems relating stochastic gradient algorithms to stochastic modified equations in the form of weak approximations. These generalize the approximation results in Li et al. (2017) in various aspects. In a subsequent paper in the series, we will discuss the application of this formalism to adaptive stochastic gradient algorithms and related problems.

The organization of this paper is as follows. We first discuss related work in Sec. 2, especially in the context of continuous-time approximations. Next, we motivate the SME approach and set up the precise mathematical framework in Sec. 3.2. We then prove in Sec. 4 a central result relating discrete stochastic algorithms and continuous stochastic processes, which allows us to derive SMEs for stochastic gradient descent and variants. In Sec. 5, the SME approach is used to analyze the dynamics of stochastic gradient algorithms when applied to optimize a simple yet non-trivial objective. Lastly, we conclude with some discussion

©2019 Qianxiao Li, Cheng Tai and Weinan E.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-526.html.

of our results in Sec. 6. The longer proofs of the results used in the paper are organized in the appendix. These are essentially self-contained, but basic knowledge of stochastic calculus and probability theory are assumed. Unfamiliar readers may refer to standard introductory texts, such as Durrett (2010) and Oksendal (2013).

1.1. Notation

In this paper, we adhere wherever possible to the following notation. Dimensional indices are written as subscripts with a bracket to avoid confusion with other sequential indices (e.g. time, iteration number), which do not have brackets. When more than one indices are present, we separate them with a comma, e.g. $x_{k,(i)}$ is the i-th coordinate of the vector x_k, the k^{th} member of a sequence. We adopt the Einstein's summation convention, where repeated (spatial) indices are summed, i.e. $x_{(i)}x_{(i)} := \sum_{i=1}^{d} x_{(i)}x_{(i)}$. For a matrix A, we denote by $\lambda(A) = \{\lambda_1(A), \lambda_2(A), \dots\}$ the set of eigenvalues of A. If A is Hermitian, then the eigenvalues are ordered so that $\lambda_1(A)$ denotes a maximum eigenvalue. We denote the usual Euclidean norm by $|\cdot|$ and for higher rank tensors, we use the same notation to denote the flattened vector norms (e.g. for matrices it will be the Frobenius norm). The $\wedge$ symbols denotes the minimum operator, i.e. $a \wedge b := \min(a, b)$.

For a probability space (or generally, a measure space) $(\Omega, \mathcal{F}, \mathbb{P})$, the symbol $\mathcal{L}(\Omega, \mathcal{F}, \mathbb{P})$, $p \in (1, \infty)$ denotes the usual Lebesgue spaces, i.e. $u \in \mathcal{L}^p(\Omega, \mathcal{F}, \mathbb{P})$ if

$$\|u\|^p_{\mathcal{L}^p(\Omega,\mathcal{F},\mathbb{P})} := \int_\Omega |u(\omega)|^p d\mathbb{P}(\omega) \equiv \mathbb{E}|u|^p < \infty.$$

When the underlying probability space is obvious, we use the shorthand $\mathcal{L}^p(\Omega) \equiv \mathcal{L}(\Omega, \mathcal{F}, \mathbb{P})$. In addition, when $\Omega = \mathbb{R}^d$, we also write the local $\mathcal{L}^p$ spaces as $\mathcal{L}^p_{\text{loc}}(\mathbb{R}^d)$, which contains u for which $|u|^p$ is integrable on compact subsets of $\mathbb{R}^d$.

Finally, we note that in the proofs of various results, we typically use the letter C (whose value may change across results) to denote a generic positive constant. This is usually independent of the learning rate η, but if not explicitly stated otherwise, it may depend on e.g. Lipschitz constants, ambient dimensions, etc.

2. Related work

In this section, we discuss several related works on analyzing discrete-time algorithms using continuous-time approaches. The idea of approximating discrete-time stochastic algorithms by continuous equations dates back to the large body of work known as stochastic approximation theory (Kushner and Yin, 2003; Ljung et al., 2012). These typically establish law of large numbers type results where the limiting equation is an ODE, which can then be used to prove powerful convergence results for the stochastic algorithms under consideration. A notion of convergence in distribution, similar to a central limit theorem, was also studied for the purpose of estimating the rate of convergence of the ODE methods (Kushner, 1978; Kushner and Shwartz, 1984; Kushner and Clark, 2012), where connections between leading order perturbations and Ornstein-Uhlenbeck (OU) processes are established. However, these estimates are not used to systematically study the dynamics of stochastic gradient descent and their variants, beyond convergence analysis for the discrete-time algorithms.

As far as the authors are aware, the first work on using stochastic differential equations to study the precise dynamical properties of stochastic gradient descent are the independent works of Li et al. (2015) and Mandt et al. (2015). In Li et al. (2015), a systematic framework of SDE approximation of SGD and SGD with momentum are derived and applied to study dynamical properties of the stochastic algorithms as well as adaptive parameter tuning schemes. These go beyond OU process approximations and this distinction is important since the OU process is not always the appropriate stochastic approximation in general settings (See Sec. 4.2 of this paper). In Mandt et al. (2015), a similar procedure is employed to derive a SDE approximation for the SGD, from which issues such as choice of learning rates are studied. Although the concrete analysis in Mandt et al. (2015) is on the restricted case of constant diffusion matrices leading to OU processes, the essential ideas on the general leading order approximation are also discussed. We also mention the work of Raginsky and Bouvrie (2012) and Mertikopoulos and Staudigl (2017), where the authors considered a continuous-time variant of mirror descent in the form of a coupled SDE and studied its convergence properties. In Krichene and Bartlett (2017), a SDE approximation of a general class of accelerated mirror descent algorithms is introduced. Under appropriate choices of the mirror map, the approximations considered there can be linked with the order-1 approximations derived in this paper. In contrast with the present paper which studies the precise sense in which these approximations hold, these works are primarily concerned with investigating the convergence of the optimization algorithms themselves using the approximate continuous-time equations.

It is important to note that the approximation arguments in the aforementioned works, including both Li et al. (2015) and Mandt et al. (2015), are heuristic from a mathematical point of view. In Li et al. (2017), the SME approximation is rigorously proved in the finite-sum-objective case with strong regularity conditions, and further asymptotic analysis and tuning algorithms are studied. The SME approach has subsequently been utilized to study variants of stochastic gradient algorithms, including those in the distributed optimization setting (An et al., 2018). The work of Mandt et al. (2015) is further developed in Mandt et al. (2016, 2017), with applications such as the development scalable MCMC algorithms.

The present paper builds on the earlier work of Li et al. (2015, 2017), but focuses on extending and solidifying the mathematical aspects. In particular, we present an entirely rigorous and self-contained mathematical formulation of the SME framework that applies to more general algorithms (including momentum SGD and stochastic Nesterov's accelerated gradient method) and more general objectives (expectation over random functions, instead of just a finite-sum). Moreover, various regularity conditions in Li et al. (2017) have been relaxed. The main approximation procedure is inspired by the seminal works of Milstein (1986, 1975) in numerical analysis of stochastic differential equations, but lower regularity conditions are required in our case due to the presence of the small noise parameter, which allows for better truncation of Itô-Taylor expansions. The mathematical analysis of the SME-type approximation for the SGD was also performed in Feng et al. (2017); Hu et al. (2017) using semi-group approaches, although the smoothness requirements presented there are greater than those established using the current methods. Lastly, the Nesterov's accelerated gradient SME we derive in Sec. 4.4 can be viewed as a generalization of the ODE approach in Su et al. (2014) to stochastic gradients, and we show that the presence of noise gives additional features to the dynamics. Finally, we note that continuous-time approx-

imations that establish links between optimization, calculus of variations and symplectic integration has been studied in Wibisono et al. (2016); Betancourt et al. (2018).

3. Stochastic modified equations

We now introduce the stochastic modified equations framework. The starting motivation is the observation that GD iterations is a (Euler) discretization of the continuous-time, ordinary differential equation

$$\frac{dx}{dt} = -\nabla f(x), \tag{3.1}$$

and studying (3.1) can give us important insights to the dynamics of the discrete-time algorithm for small enough learning rates. The natural question when extending this to SGD is, what is the right continuous-time equation to consider? Below, we begin with some heuristic considerations.

3.1. Heuristic motivations

we rewrite the SGD iteration (1.3) as

$$x_{k+1} = x_k - \eta \nabla f(x_k) + \sqrt{\eta} V_k(x_k, \gamma_k), \tag{3.2}$$

where $V_k(x_k, \gamma_k) = \sqrt{\eta}(\nabla f(x_k) - \nabla f_{\gamma_k}(x_k))$ is a d-dimensional random vector. A straightforward calculation shows that

$$\begin{aligned} &\mathbb{E}[V_k|x_k] = 0 \\ &\mathrm{cov}[V_k, V_k|x_k] = \eta \Sigma(x_k), \\ &\Sigma(x_k) := \mathbb{E}[(\nabla f_{\gamma_k}(x_k) - \nabla f(x_k))(f_{\gamma_k}(x_k) - \nabla f(x_k))^T|x_k], \end{aligned} \tag{3.3}$$

i.e. conditional on x_k, $V_k(x_k)$ has 0 mean and covariance $\eta\Sigma(x_k)$. Here, Σ is simply the conditional covariance of the stochastic gradient approximation ∇f_γ of ∇f.

Now, consider a time-homogeneous Itô stochastic differential equation (SDE) of the form

$$dX_t = b(X_t)dt + \sqrt{\eta}\sigma(X_t)dW_t, \tag{3.4}$$

where $X_t \in \mathbb{R}^d$ for $t \geq 0$ and W_t is a standard d-dimensional Wiener process. The function $b : \mathbb{R}^d \to \mathbb{R}^d$ is known as the drift and $\sigma : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ is the diffusion matrix. The key observation is that if we apply the Euler discretization with step-size η to (3.4), approximating $X_{k\eta}$ by $\hat{X}_k$, we obtain the following discrete iteration for the latter:

$$\hat{X}_{k+1} = \hat{X}_k + \eta b(\hat{X}_k) + \eta\sigma(\hat{X}_k)Z_k, \tag{3.5}$$

where $Z_k := W_{(k+1)\eta} - W_{k\eta}$ are d-dimensional i.i.d. standard normal random variables. Comparing with (3.2), if we set $b = -\nabla f$, $\sigma(x) = \Sigma(x)^{1/2}$ and identify t with $k\eta$, we then have matching first and second conditional moments. Hence, this motivates the approximating equation

$$dX_t = -\nabla f(X_t)dt + (\eta\Sigma(X_t))^{1/2}dW_t. \tag{3.6}$$

Note that as this heuristic argument shows, the presence of the small parameter $\sqrt{\eta}$ on the diffusion term is necessary to model the fact that when learning rate decreases, the fluctuations to the SGA iterates must also decrease.

The immediate mathematical question is then: in what sense is an SDE like (3.6) an approximation of (1.3)? Let us now establish the precise mathematical framework in which we can answer this question.

3.2. The mathematical framework

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a sufficiently rich probability space and $(\Gamma, \mathcal{F}_\Gamma)$ be a measure space representing the index space for our random objectives. Let $\gamma : \Omega \to \Gamma$ be a random variable and $(r, x) \mapsto f_r(x)$ a measurable mapping from $\Gamma \times \mathbb{R}^d$ to $\mathbb{R}$. Hence, for each x, $f_\gamma(x)$ is a random variable. Throughout this paper, we assume the follow facts about $f_\gamma(x)$:

Assumption 3.1 *The random variable $f_\gamma(x)$ satisfies*

(i) $f_\gamma(x) \in \mathcal{L}^1(\Omega)$ *for all* $x \in \mathbb{R}^d$

(ii) $f_\gamma(x)$ *is continuously differentiable in* x *almost surely and for each* $R > 0$*, there exists a random variable* $M_{R,\gamma}$ *such that* $\max_{|x|\leq R} |\nabla f_\gamma(x)| \leq M_{R,\gamma}$ *almost surely, with* $\mathbb{E}|M_{R,\gamma}| < \infty$

(iii) $\nabla f_\gamma(x) \in \mathcal{L}^2(\Omega)$ *for all* $x \in \mathbb{R}^d$

Note that in the empirical risk minimization case where Γ is finite, the conditions above are often trivially satisfied. Condition (i) in Assumption 3.1 allows us to define the total objective function we would like to minimize as the expectation

$$f(x) := \mathbb{E}f_\gamma(x) \equiv \int_\Omega f_{\gamma(\omega)}(x)d\mathbb{P}(\omega). \tag{3.7}$$

Moreover, Assumption 3.1 (ii) implies via the dominated convergence theorem that $\mathbb{E}\nabla f_\gamma = \nabla\mathbb{E}f_\gamma \equiv \nabla f$. Now, let $\{\gamma_k : k = 0, 1, \dots\}$ be a sequence of i.i.d. Γ-valued random variables with the same distribution as γ. Let $x_0 \in \mathbb{R}^d$ be fixed and define the generalized stochastic gradient iteration as the stochastic process

$$x_{k+1} = x_k + \eta h(x_k, \gamma_k, \eta) \tag{3.8}$$

for $k \geq 0$, where $h : \mathbb{R}^d \times \Gamma \times \mathbb{R} \to \mathbb{R}^d$ is a measurable function and $\eta > 0$ is the learning rate. In the simple case of SGD, we have $h(x, r, \eta) = -\nabla f_r(x)$, but we shall consider the generalized version above so that modified equations for SGD variants can also be derived from our approximation theorems.

Next, let us define the class of approximating continuous stochastic processes, which we call stochastic modified equations. Consider the time-homogeneous Itô diffusion process $\{X_t : t \geq 0\}$ represented by the following stochastic differential equation (SDE)

$$dX_t = b(X_t, \eta)dt + \sqrt{\eta}\sigma(X_t, \eta)dW_t, \qquad X_0 = x_0 \tag{3.9}$$

where $\{W_t : t \geq 0\}$ is a standard d-dimensional Wiener process independent of $\{\gamma_k\}$, $b : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$ is the approximating drift vector and $\sigma : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^{d\times d}$ is the approximating

diffusion matrix. In the following, we will need to pick b, σ appropriately so that (3.8) is approximated by (3.9), the sense of which we now describe.

First, notice that the stochastic process $\{x_k\}$ induces a probability measure on the product space $\mathbb{R}^d \times \mathbb{R}^d \times \cdots$, whereas $\{X_t\}$ induces a probability measure on $\mathcal{C}^0([0,\infty), \mathbb{R}^d)$. To compare them, one can form a piece-wise linear interpolation of the former. Alternatively, as we do in this work, we sample a discrete number of points from the latter. Second, the process $\{x_k\}$ is adapted to the filtration generated by $\{\gamma_k\}$ (e.g. in the case of SGD, this is the random sampling of functions in $\{f_r\}$), whereas the process $\{X_t\}$ is adapted to an independent, Wiener filtration. Hence, it is not appropriate to compare individual sample paths. Rather, we define below a sense of weak approximations by comparing the distributions of the two processes.

Definition 1 *Let G denote the set of continuous functions $\mathbb{R}^d \to \mathbb{R}$ of at most polynomial growth, i.e. $g \in G$ if there exists positive integers $\kappa_1, \kappa_2 > 0$ such that*

$$|g(x)| \leq \kappa_1(1 + |x|^{2\kappa_2}),$$

for all $x \in \mathbb{R}^d$. Moreover, for each integer $\alpha \geq 1$ we denote by G^α the set of α-times continuously differentiable functions $\mathbb{R}^d \to \mathbb{R}$ which, together with its partial derivatives up to and including order α, belong to G. Note that each G^α is a subspace of $\mathcal{C}^\alpha$, the usual space of α-times continuously differentiable functions. Moreover, if g depends on additional parameters, we say $g \in G^\alpha$ if the constants κ_1, κ_2 are independent of these parameters, i.e. $g \in G^\alpha$ uniformly. Finally, the definition generalizes to vector-valued functions coordinate-wise in the co-domain.

Definition 2 *Let $T > 0$, $\eta \in (0, 1 \wedge T)$, and $\alpha \geq 1$ be an integer. Set $N = \lfloor T/\eta \rfloor$. We say that a continuous-time stochastic process $\{X_t : t \in [0,T]\}$ is an order α weak approximation of a discrete stochastic process $\{x_k : k = 0, \ldots, N\}$ if for every $g \in G^{\alpha+1}$, there exists a positive constant C, independent of η, such that*

$$\max_{k=0,..,N} |\mathbb{E}g(x_k) - \mathbb{E}g(X_{k\eta})| \leq C\eta^\alpha. \tag{3.10}$$

Let us discuss briefly the notion of *weak approximation* as introduced above. These are approximations of the distribution of sample paths, instead of the sample paths themselves. This is enforced by requiring that the expectations of the two processes $\{X_t\}$ and $\{x_k\}$ over a sufficiently large class of test functions to be close. In our definition, the test function class $G^{\alpha+1}$ is quite large, and in particular it includes all polynomials. Thus, Eq. (3.10) implies in particular that all moments of the two processes become close at the rate of η^α, and hence so must their distributions. The notion of weak approximation must be contrasted with that of *strong approximations*, where one would for example require (in the case of *mean-square approximations*)

$$\left[\mathbb{E}|x_k - X_{k\eta}|^2\right]^{1/2} \leq C\eta^\alpha.$$

The above forces the actual sample-paths of the two processes to be close, per realization of the random process, which severely limits its application. In fact, one important advantage of

weak approximations is that the continuous-time process X_t can in fact approximate discrete stochastic processes whose step-wise driving noise is not Gaussian, as long as appropriate moments are matched (see e.g. the content of Theorem 3). This additional flexibility is useful as it allows the treatment of more general classes of stochastic gradient iterations.

4. The approximation theorems

We now present the main approximation theorems. The derivation is based on the following two-step process:

1. We establish a connection between one-step approximation and approximation on a finite time interval.
2. We construct a one-step approximation that is of order $\alpha+1$, and so the approximation on a finite interval is of order α.

4.1. Relating one-step to N-step approximations

Let us consider generally the question of the relationship between one-step approximations and approximations on a finite interval. Let $T > 0$, $\eta \in (0, 1 \wedge T)$ and $N = \lfloor T/\eta \rfloor$ and recall the general SGA iterations

$$x_{k+1} = x_k + \eta h(x_k, \gamma_k, \eta), \quad x_0 \in \mathbb{R}^d, \quad k = 0, \ldots, N. \tag{4.1}$$

and the general candidate family of approximating SDEs

$$dX_t^{\eta,\epsilon} = b(X_t^{\eta,\epsilon}, \eta, \epsilon)dt + \sqrt{\eta}\sigma(X_t^{\eta,\epsilon}, \eta, \epsilon)dW_t, \quad X_0 = x_0, \quad t \in [0,T], \tag{4.2}$$

where $\epsilon \in (0,1)$ is a mollification parameter, whose role will become apparent later. To reduce notational clutter and improve readability, unless some limiting procedure is considered, we shall not explicitly write the dependence of $X_t^{\eta,\epsilon}$ on η, ϵ and simply denote by X_t the solution of the above SDE. Let us also denote for convenience $\widetilde{X}_k := X_{k\eta}$. Further, let $\{X_t^{x,s} : t \geq s\}$ denote the stochastic process obeying the same equation (4.2), but with the initial condition $X_s^{x,s} = x$. We similarly write $\widetilde{X}_k^{x,l} := X_{k\eta}^{x,l\eta}$ and denote by $\{x_k^{x,l} : k \geq l\}$ the stochastic process satisfying (4.1) but with $x_l = x$.

Throughout this section, we assume the following conditions:

Assumption 4.1 *The functions $b : \mathbb{R}^d \times (0, 1 \wedge T) \times (0,1) \to \mathbb{R}^d$ and $\sigma : \mathbb{R}^d \times (0, 1 \wedge T) \times (0,1) \to \mathbb{R}^{d\times d}$ satisfy:*

1. *Uniform linear growth condition*

$$|b(x,\eta,\epsilon)|^2 + |\sigma(x,\eta,\epsilon)|^2 \leq L^2(1 + |x|^2)$$

for all $x, y \in \mathbb{R}^d$, $\eta \in (0, 1 \wedge T)$, $\epsilon \in (0,1)$.

2. *Uniform Lipschitz condition*

$$|b(x,\eta,\epsilon) - b(y,\eta,\epsilon)| + |\sigma(x,\eta,\epsilon) - \sigma(y,\eta,\epsilon)| \leq L|x - y|$$

for all $x, y \in \mathbb{R}^d$, $\eta \in (0, 1 \wedge T)$, $\epsilon \in (0,1)$.

Note that 2 implies 1 if there is at least one x where the supremum of b, σ over η, ϵ is finite. In particular, these conditions imply via Thm. 18 that there exists a unique solution to Eq. 4.2.

Now, let us denote the one-step changes

$$\Delta(x) := x_1^{x,0} - x, \qquad \widetilde{\Delta}(x) := \widetilde{X}_1^{x,0} - x. \tag{4.3}$$

We prove the following result which relates one-step approximations with approximations on a finite time interval.

Theorem 3 *Let $T > 0$, $\eta \in (0, 1 \wedge T)$, $\epsilon \in (0,1)$ and $N = \lfloor T/\eta \rfloor$. Let $\alpha \geq 1$ be an integer. Suppose further that the following conditions hold:*

(i) There exists a function $\rho : (0,1) \to \mathbb{R}_+$ and $K_1 \in G$ independent of η, ϵ such that

$$\left| \mathbb{E} \prod_{j=1}^{s} \Delta_{(i_j)}(x) - \mathbb{E} \prod_{j=1}^{s} \widetilde{\Delta}_{(i_j)}(x) \right| \leq K_1(x)(\eta\rho(\epsilon) + \eta^{\alpha+1}),$$

for $s = 1, 2, \ldots, \alpha$ and

$$\mathbb{E} \prod_{j=1}^{\alpha+1} \left| \Delta_{(i_j)}(x) \right| \leq K_1(x)\eta^{\alpha+1},$$

for all $i_j \in \{1, \ldots, d\}$.

(ii) For each $m \geq 1$, the $2m$-moment of $x_k^{x,0}$ is uniformly bounded with respect to k and η, i.e. there exists a $K_2 \in G$, independent of η, k, such that

$$\mathbb{E}|x_k^{x,0}|^{2m} \leq K_2(x),$$

for all $k = 0, \ldots, N \equiv \lfloor T/\eta \rfloor$.

Then, for each $g \in G^{\alpha+1}$, there exists a constant $C > 0$, independent of η, ϵ, such that

$$\max_{k=0,\ldots,N} |\mathbb{E}g(x_k) - \mathbb{E}g(X_{k\eta})| \leq C(\eta^\alpha + \rho(\epsilon))$$

The proof of Thm. 3 requires a number of technical results that we defer to the appendix. Below, we demonstrate the main ingredients of the proof and refer to the appendix where the proofs of the auxiliary results are fully presented.

Proof In this proof, since there are many conditioning on the initial condition, to prevent nested superscripts we shall introduce the alternative notation $X_t(x,s) \equiv X_t^{x,s}$, and similarly for $\tilde{X}_k$ and x_k. Fix $g \in G^{\alpha+1}$ and $1 \leq k \leq N$. We have

$$\mathbb{E}g(X_{k\eta}) = \mathbb{E}g(\widetilde{X}_k) = \mathbb{E}g(\widetilde{X}_k(\widetilde{X}_1, 1)) - \mathbb{E}g(\widetilde{X}_k(x_1, 1)) + \mathbb{E}g(\widetilde{X}_k(x_1, 1)).$$

If $k > 1$, by noting that $\widetilde{X}_k(x_1, 1) = \widetilde{X}_k(\widetilde{X}_2(x_1, 1), 2)$, we get

$$\mathbb{E}g(\widetilde{X}_k(x_1, 1)) = \mathbb{E}g(\widetilde{X}_k(\widetilde{X}_2(x_1, 1), 2)) - \mathbb{E}g(\widetilde{X}_k(x_2, 2)) + \mathbb{E}g(\widetilde{X}_k(x_2, 2))$$

Continuing this process, we then have

$$\begin{aligned}\mathbb{E}g(\widetilde{X}_k) =& \sum_{l=1}^{k-1} \mathbb{E}g(\widetilde{X}_k(\widetilde{X}_l(x_{l-1}, l-1), l)) - \mathbb{E}g(\widetilde{X}_k(x_l, l)) \\ &+ \mathbb{E}g(\widetilde{X}_k(x_{k-1}, k-1))\end{aligned}$$

and hence by subtracting $\mathbb{E}g(x_k) \equiv \mathbb{E}g(x_k(x_{k-1}, k-1))$ we get

$$\begin{aligned}\mathbb{E}g(\widetilde{X}_k) - \mathbb{E}g(x_k) =& \sum_{l=1}^{k-1} \mathbb{E}g(\widetilde{X}_k(\widetilde{X}_l(x_{l-1}, l-1), l)) - \mathbb{E}g(\widetilde{X}_k(x_l, l)) \\ &+ \mathbb{E}g(\widetilde{X}_k(x_{k-1}, k-1)) - \mathbb{E}g(x_k(x_{k-1}, k-1))\end{aligned}$$

and so

$$\mathbb{E}g(\widetilde{X}_k) - \mathbb{E}g(x_k) = \sum_{l=1}^{k-1} \mathbb{E}\mathbb{E}\left[g(\widetilde{X}_k(\widetilde{X}_l(x_{l-1}, l-1), l)) \middle| \widetilde{X}_l(x_{l-1}, l-1) \right] - \mathbb{E}\mathbb{E}\left[g(\widetilde{X}_k(x_l, l)) \middle| x_l \right]$$

$$+ \mathbb{E}g(\widetilde{X}_k(x_{k-1}, k-1)) - \mathbb{E}g(x_k(x_{k-1}, k-1)),$$

Now, let $u(x,s) = \mathbb{E}g(X_{k\eta}(x,s))$. Then, we have

$$\begin{aligned}|\mathbb{E}g(\widetilde{X}_k) - \mathbb{E}g(x_k)| \leq& \sum_{l=1}^{k-1} |\mathbb{E}u(\widetilde{X}_l(x_{l-1}, l-1), l\eta) - \mathbb{E}u(x_l(x_{l-1}, l-1), l\eta)| \\ &+ |\mathbb{E}g(\widetilde{X}_k(x_{k-1}, k-1)) - \mathbb{E}g(x_k(x_{k-1}, k-1))| \\ \leq& \sum_{l=1}^{k-1} \mathbb{E}|\mathbb{E}[u(\widetilde{X}_l(x_{l-1}, l-1), l\eta)|x_{l-1}] - \mathbb{E}[u(x_l(x_{l-1}, l-1), l\eta)|x_{l-1}]| \\ &+ \mathbb{E}|\mathbb{E}[g(\widetilde{X}_k(x_{k-1}, k-1))|x_{k-1}] - \mathbb{E}[g(x_k(x_{k-1}, k-1))|x_{k-1}]|.\end{aligned}$$

Using Prop. 25, $u(\cdot, s) \in G^{\alpha+1}$ uniformly in s, t, η and ϵ. Thus, by Assumption (i) and Lem. 27,

$$\begin{aligned}|\mathbb{E}g(x_k) - \mathbb{E}g(\widetilde{X}_k)| \leq& (\eta\rho(\epsilon) + \eta^{\alpha+1}) \left(\sum_{l=1}^{k-1} \mathbb{E}K_{l-1}(x_{l-1}) + \mathbb{E}K_{k-1}(x_{k-1}) \right) \\ \leq& (\eta\rho(\epsilon) + \eta^{\alpha+1}) \sum_{l=0}^{N} \kappa_{l,1}(1 + \mathbb{E}|x_l|^{2\kappa_{l,2}}),\end{aligned}$$

where in the last line we used moment estimates from Thm. 19. Finally, using Assumption (ii) and the fact that $N \leq T/\eta$, we have

$$|\mathbb{E}g(x_k) - \mathbb{E}g(X_{k\eta})| = |\mathbb{E}g(x_k) - \mathbb{E}g(\widetilde{X}_k)| \leq C(\rho(\epsilon) + \eta^\alpha).$$

■

4.2. SME for stochastic gradient descent

Thm. 3 allows us to prove the main approximation results for the current paper. In particular, in this section we derive a second-order accurate weak approximation for the simple SGD iterations (1.3), from which a simpler, first-order accurate approximation also follows. As seen in Thm. 3, we need only verify the conditions (i)-(ii) in order to prove the weak approximation result. These conditions mostly involve moment estimates, which we now perform. To simplify presentation, we introduce the following shorthand. Whenever we write

$$\psi(x) = \psi_0(x) + \eta\psi_1(x) + \mathcal{O}(r(\eta,\epsilon)),$$

for some remainder term $r(\eta,\epsilon)$, we mean: there exists $K \in G$ independent of η, ϵ such that

$$|\psi(x) - \psi_0(x) - \eta\psi_1(x)| \leq K(x)r(\eta,\epsilon).$$

Now, let us set in (4.2)

$$\begin{aligned} b(x,\eta,\epsilon) &= b_0(x,\epsilon) + \eta b_1(x,\epsilon) \\ \sigma(x,\eta,\epsilon) &= \sigma_0(x,\epsilon), \end{aligned}$$

where b_0, b_1, σ_0 are functions to be determined. We have the following moment estimate.

Lemma 4 *Let $\widetilde{\Delta}(x)$ be defined as in (4.3). Suppose further that with $b_0, b_1, \sigma_0 \in G^3$. Then we have*

(i) $\mathbb{E}\widetilde{\Delta}_{(i)}(x) = b_0(x,\epsilon)_{(i)}\eta + [\frac{1}{2}b_0(x,\epsilon)_{(j)}\partial_{(j)}b_0(x,\epsilon)_{(i)} + b_1(x,\epsilon)_{(i)}]\eta^2 + \mathcal{O}(\eta^3)$,

(ii) $\mathbb{E}\widetilde{\Delta}_{(i)}(x)\widetilde{\Delta}_{(j)}(x) = [b_0(x,\epsilon)_{(i)}b_0(x,\epsilon)_{(j)} + \sigma_0(x,\epsilon)_{(i,k)}\sigma_0(x,\epsilon)_{(j,k)}]\eta^2 + \mathcal{O}(\eta^3)$,

(iii) $\mathbb{E}\prod_{j=1}^3 |\widetilde{\Delta}_{(i_j)}(x)| = \mathcal{O}(\eta^3)$.

Proof To obtain (i)-(iii), we simply apply Lem. 28 with $\psi(z) = \prod_{j=1}^s (z_{(i_j)} - x_{(i_j)})$ for $s = 1, 2, 3$ respectively. ■

Next, we estimate the moments of the SGA iterations below.

Lemma 5 *Let $\Delta(x)$ be defined as in (4.3) with the SGD iterations, i.e. $h(x,r,\eta) = -\nabla f_r(x)$. Suppose that for each $x \in \mathbb{R}^d$, $f \in G^1$. Then,*

(i) $\mathbb{E}\Delta_{(i)}(x) = -\partial_{(i)}f(x)\eta$,

(ii) $\mathbb{E}\Delta_{(i)}(x)\Delta_{(j)}(x) = \partial_{(i)}f(x)\partial_{(j)}f(x)\eta^2 + \Sigma(x)_{(i,j)}\eta^2$,

(iii) $\mathbb{E}\prod_{j=1}^3 |\Delta_{(i_j)}(x)| = \mathcal{O}(\eta^3)$,

where $\Sigma(x) := \mathbb{E}(\nabla f_\gamma(x) - \nabla f(x))(\nabla f_\gamma(x) - \nabla f(x))^T$.

Proof We have $\Delta(x) = -\eta\nabla f_{\gamma_0}(x)$. Taking expectations, the results then follow. ■

We now prove the main approximation theorem for the simple SGD. Before presenting the statement and proof, we shall note a few technical issues that prevents the direct application of Thm. 3 with the moment estimates in Lem.4 and 5. The latter suggest ignoring ϵ and setting

$$b_0(x,\epsilon) = -\nabla f(x), \quad b_1(x,\epsilon) = --\tfrac{1}{4}\nabla|\nabla f(x)|^2, \quad \sigma_0(x,\epsilon) = \Sigma(x)^{\frac{1}{2}}.$$

Then, we would see from Lem.4 and 5 that the SGD and the SDE have matching moments up to $\mathcal{O}(\eta^3)$. The first issue with this approach is that even if $\Sigma(x)$ is sufficiently smooth (which may follow from the regularity of ∇f_γ), the smoothness of $\Sigma(x)^{1/2}$ cannot be guaranteed unless $\Sigma(x)$ is positive-definite, which is often too strong an assumption in practice and excludes interesting cases where $\Sigma(x)$ is a singular diffusion matrix. However, the results in Sec. 4.1 require smoothness. Second, we would like to consider functions f_γ that may not have higher strong derivatives required by the Lemmas, beyond those required to define the modified equation itself. To fix both of these issues, we will use a simple mollifying technique. This is the reason for the inclusion of the ϵ parameter in the results in Sec. 4.1.

Definition 6 *Let us denote by $\nu : \mathbb{R}^d \to \mathbb{R}$, $\nu \in \mathcal{C}_c^\infty(\mathbb{R}^d)$ the standard mollifier*

$$\nu(x) := \begin{cases} C\exp(-\frac{1}{1-|x|^2}) & |x| < 1 \\ 0 & |x| \geq 1, \end{cases}$$

where $C := (\int_{\mathbb{R}^d}\nu(y)dy)^{-1}$ is chosen so that the integral of ν is 1. Further, define $\nu^\epsilon(x) = \epsilon^{-d}\nu(x/\epsilon)$. Let $\psi \in \mathcal{L}^1_{loc}(\mathbb{R}^d)$ be locally integrable, then we may define its mollification by

$$\psi^\epsilon(x) := (\nu^\epsilon * \psi)(x) = \int_{\mathbb{R}^d}\nu^\epsilon(x-y)\psi(y)dy = \int_{\mathcal{B}(0,\epsilon)}\nu^\epsilon(y)\psi(x-y)dy,$$

where $\mathcal{B}(z,\epsilon)$ is the d-dimensional ball of radius ϵ centered at z. The mollification of vector (or matrix) valued functions are defined element-wise.

The mollifier has very useful properties. In particular, we will use the following well-known facts (see e.g. Evans (2010) for proof)

(i) If $\psi \in \mathcal{L}^1_{\text{loc}}(\mathbb{R}^d)$, then $\psi^\epsilon \in \mathcal{C}^\infty(\mathbb{R}^d)$

(ii) $\psi^\epsilon(x) \to \psi(x)$ as $\epsilon \to 0$ for almost every $x \in \mathbb{R}^d$ (with respect to the Lebesgue measure)

(iii) If ψ is continuous, then $\psi^\epsilon(x) \to \psi(x)$ as $\epsilon \to 0$ uniformly on compact subsets of $\mathbb{R}^d$

Next, we make use of the idea of weak derivatives.

Definition 7 *Let $\Psi \in \mathcal{L}^1_{loc}(\mathbb{R}^d)$ and J be a multi-index of order $|J|$. Suppose that there exists a $\psi \in \mathcal{L}^1_{loc}(\mathbb{R}^d)$ such that*

$$\int_{\mathbb{R}^d}\Psi(x)\nabla^J\phi(x)dx = (-1)^{|J|}\int_{\mathbb{R}^d}\psi(x)\phi(x)dx$$

for all $\phi \in \mathcal{C}_c^\infty$. Then, we call ψ the order J weak derivative of Ψ and write $D^J\Psi = \psi$. Note that when it exists, the weak derivative is unique almost everywhere and if Ψ is differentiable, $\nabla^J\Psi = D^J\Psi$ almost everywhere (Evans, 2010).

The introduction of weak derivatives motivates the definition of the weak version of the function spaces G^α.

Definition 8 *For $\alpha \geq 1$, we define the space G_w^α to be the subspace of $\mathcal{L}^1_{loc}(\mathbb{R}^d)$ such that if $g \in G_w^\alpha$, then g has weak derivatives up to order α and for each multi-index J with $|J| \leq \alpha$, there exists positive integers κ_1, κ_2 such that*

$$|D^J g(x)| \leq \kappa_1(1 + |x|^{2\kappa_2}) \text{ for a.e. } x \in \mathbb{R}^d.$$

As in Def. 1, if g depends on additional parameters, we say that $g \in G_w^\alpha$ if the above constants do not depend on the additional parameters. Also, vector-valued g are defined as above element-wise in the co-domain. Note that G_w^α is a subspace of the Sobolev space $W^{\alpha,1}_{loc}$.

Theorem 9 *Let, $T > 0$, $\eta \in (0, 1 \wedge T)$ and set $N = \lfloor T/\eta \rfloor$. Let $\{x_k : k \geq 0\}$ be the SGD iterations defined in (1.3). Suppose the following conditions are met:*

(i) $f \equiv \mathbb{E}f_\gamma$ is twice continuously differentiable, $\nabla|\nabla f|^2$ is Lipschitz, and $f \in G_w^4$.

(ii) ∇f_γ satisfies a Lipschitz condition:

$$|\nabla f_\gamma(x) - \nabla f_\gamma(y)| \leq L_\gamma|x - y| \quad a.s.$$

for all $x, y \in \mathbb{R}^d$, where L_γ is a random variable which is positive a.s. and $\mathbb{E}L_\gamma^m < \infty$ for each $m \geq 1$.

Define $\{X_t : t \in [0, T]\}$ as the stochastic process satisfying the SDE

$$dX_t = -\nabla(f(X_t) + \tfrac{1}{4}\eta|\nabla f(X_t)|^2)dt + \sqrt{\eta}\Sigma(X_t)^{1/2}dW_t \qquad X_0 = x_0, \tag{4.4}$$

with $\Sigma(x) = \mathbb{E}(\nabla f_\gamma(x) - \nabla f(x))(\nabla f_\gamma(x) - \nabla f(x))^T$. Then, $\{X_t : t \in [0, T]\}$ is an order-2 weak approximation of the SGD, i.e. for each $g \in G^3$, there exists a constant $C > 0$ independent of η such that

$$\max_{k=0,..,N} |\mathbb{E}g(x_k) - \mathbb{E}g(X_{k\eta})| \leq C\eta^2.$$

Proof First, we check that Eq. (4.4) admits a unique solution, which amounts to checking the conditions in Thm. 18. Note that the Lipschitz condition (ii) implies ∇f is Lipschitz with constant $\mathbb{E}L_\gamma$. To see that $\Sigma(x)^{1/2}$ is also Lipschitz, observe that $u(x) := \nabla f_\gamma(x) - \nabla f(x)$ is Lipschitz (in the sense of (ii), with constant at most $L_\gamma + \mathbb{E}L_\gamma$), and

$$\begin{aligned}|\Sigma(x)^{1/2} - \Sigma(y)^{1/2}| &= \left| \|[u(x)u(x)^T]^{1/2}\|_{\mathcal{L}^2(\Omega)} - \|[u(y)u(y)^T]^{1/2}\|_{\mathcal{L}^2(\Omega)} \right| \\ &\leq \|[u(x)u(x)^T]^{1/2} - [u(y)u(y)^T]^{1/2}\|_{\mathcal{L}^2(\Omega)}.\end{aligned}$$

Moreover, observe that for vectors $u \in \mathbb{R}^d$ the mapping $u \mapsto (uu^T)^{1/2} = uu^T/|u|$ is Lipschitz, which implies

$$|\Sigma(x)^{1/2} - \Sigma(y)^{1/2}| \leq L'\|u(x) - u(y)\|_{\mathcal{L}^2(\Omega)} \leq L''|x - y|.$$

The Lipschitz conditions on the drift and the diffusion matrix imply uniform linear growth, so by Thm. 18, Eq. (4.4) admits a unique solution.

For each $\epsilon \in (0, 1)$, define the mollified functions

$$b_0(x, \epsilon) = -\nu^\epsilon * \nabla f(x), \quad b_1(x, \epsilon) = -\tfrac{1}{4}\nu^\epsilon * (\nabla|\nabla f(x)|^2), \quad \sigma_0(x, \epsilon) = \nu^\epsilon * \Sigma(x)^{1/2}.$$

Observe that $b_0 + \eta b_1, \sigma_0$ satisfies a Lipschitz condition in x uniformly in η, ϵ. To see this, note that for any Lipschitz function ψ with constant L, we have

$$|\nu^\epsilon * \psi(x) - \nu^\epsilon * \psi(y)| \leq \int_{\mathcal{B}(0,\epsilon)} \nu^\epsilon(z)|\psi(x - z) - \psi(y - z)|dz \leq L|x - y|,$$

which proves $b_0 + \eta b_1$ and σ_0 are uniformly Lipschitz. Similarly, the linear growth condition follows. Hence, we may define a family of stochastic processes $\{X_t^\epsilon : \epsilon \in (0, 1)\}$ satisfying

$$dX_t^\epsilon = b_0(X_t^\epsilon, \epsilon) + \eta b_1(X_t^\epsilon, \epsilon) + \sqrt{\eta}\sigma_0(X_t^\epsilon, \epsilon)dW_t \qquad X_0^\epsilon = x_0,$$

which each admits a unique solution by Thm. 18. Now, we claim that $b_0(\cdot, \epsilon), b_1(\cdot, \epsilon), \sigma_0(\cdot, \epsilon) \in G^3$ uniformly in ϵ. To see this, simply observe that mollifications are smooth, and moreover, the polynomial growth is satisfied since $\nu^\epsilon * D^J\psi = \nabla^J(\nu^\epsilon * \psi)$ and furthermore, if $\psi \in G$, then we have

$$\begin{aligned}|\psi^\epsilon(x)| \leq& \int_{\mathcal{B}(0,\epsilon)} \nu^\epsilon(y)|\psi(x - y)|dy \\ \leq& \kappa_1\left(1 + 2^{2\kappa_2 - 1}|x|^{2\kappa_2} + 2^{2\kappa_2 - 1}\tfrac{1}{\epsilon^d}\int_{\mathcal{B}(0,\epsilon)} |y|^{2\kappa_2}dy\right)\end{aligned}$$

But $\int_{\mathcal{B}(0,\epsilon)} |y|^{2\kappa_2}dy \leq \text{Vol}(\mathcal{B}(0, \epsilon)) = C\epsilon^d$, where C is independent of ϵ. This shows that $\psi^\epsilon \in G$ uniformly in ϵ. This immediately implies that $b_0(\cdot, \epsilon), b_1(\cdot, \epsilon), \sigma_0(\cdot, \epsilon) \in G^3$.

Now, since $b_0(x, \epsilon) \to b_0(x, 0)$ (and similarly for b_1, σ_0), and the limits are continuous, by Lem. 4, 5, 29, 30,, all conditions of Thm. 3 are satisfied, and hence we conclude that for each $g \in G^3$, we have,

$$\max_{k=0,..,N} |\mathbb{E}g(X_{k\eta}^\epsilon) - \mathbb{E}g(x_k)| \leq C(\eta^2 + \rho(\epsilon)),$$

where C is independent of η and ϵ and $\rho(\epsilon) \to 0$ as $\epsilon \to 0$. Moreover, since $b_0(x, \epsilon) \to b_0(x, 0)$ (and similarly for b_1, σ_0) uniformly on compact sets, we may apply Thm. 20 to conclude that

$$\sup_{t \in [0,T]} \mathbb{E}|X_t^\epsilon - X_t|^2 \to 0 \text{ as } \epsilon \to 0.$$

Thus, we have

$$\begin{aligned}&|\mathbb{E}g(X_{k\eta}) - \mathbb{E}g(x_k)|\\ \leq&|\mathbb{E}g(X^\epsilon_{k\eta}) - \mathbb{E}g(x_k)| + |\mathbb{E}g(X^\epsilon_{k\eta}) - \mathbb{E}g(X_{k\eta})|\\ \leq& C(\eta^2 + \rho(\epsilon)) + \left(\mathbb{E}|X^\epsilon_{k\eta} - X_{k\eta}|^2\right)^{1/2}\\ &\times\left(\int_0^1 \mathbb{E}|\nabla^2 g(\lambda X^\epsilon_{k\eta} + (1-\lambda)X_{k\eta})|^2 d\lambda\right)^{1/2}\end{aligned}$$

Using Thm. 19 and assumption that $\nabla^2 g \in G$, the last expectation is finite and hence taking the limit $\epsilon \to 0$ yields our result. ■

By going for a lower order approximation, we of course have the following:

Corollary 10 *Assume the same conditions as in Thm. 9, except that we replace (i) with*

(i)' $f \equiv \mathbb{E}f_\gamma$ is continuously differentiable, and $f \in G^3_w$.

Define $\{X_t : t \in [0,T]\}$ as the stochastic process satisfying the SDE

$$dX_t = -\nabla f(X_t)dt + \sqrt{\eta}\Sigma(X_t)^{1/2}dW_t \qquad X_0 = x_0, \tag{4.5}$$

with $\Sigma(x) = \mathbb{E}(\nabla f_\gamma(x) - \nabla f(x))(\nabla f_\gamma(x) - \nabla f(x))^T$. Then, $\{X_t : t \in [0,T]\}$ is an order-1 weak approximation of the SGD, i.e. for each $g \in G^2$, there exists a constant $C > 0$ independent of η such that

$$\max_{k=0,\ldots,N} |\mathbb{E}g(X_{k\eta}) - \mathbb{E}g(x_k)| \leq C\eta.$$

Remark 11 *In the above results, the most restrictive condition is probably the Lipschitz condition on ∇f_γ. Such Lipschitz conditions are important to ensure that the SMEs admit unique strong solutions and the SGA having uniformly bounded moments. Note that following similar techniques in SDE analysis (e.g. Kloeden and Platen (2011)), these global conditions may be relaxed to their respective local versions if we assume in addition a uniform global linear growth condition on ∇f_γ. Finally, for applications, typical loss functions have inward pointing gradients for all sufficiently large x, meaning that the SGD iterates will be uniformly bounded almost surely. Thus, we may simply modify the loss functions for large x (without affecting the SGA iterates) to satisfy the conditions above.*

Remark 12 *The constant C does not depend on η, but as evidenced in the proof of the theorem, it generally depends on g, T, d and the various Lipschitz constants. For the fairly general situation we are consider, we do not derive tight estimates of these dependencies.*

4.3. SME for stochastic gradient descent with momentum

Let us discuss the corresponding SME for a popular variant of the SGD called the *momentum SGD* (MSGD). The momentum SGD augments the usual SGD iterations with a "memory" term. In the usual form, we have the iterations

$$\begin{aligned}\hat{v}_{k+1} &= \hat{\mu}\hat{v}_k - \hat{\eta}\nabla f_{\gamma_k}(x_k)\\ x_{k+1} &= x_k + \hat{v}_{k+1}\end{aligned}$$

where $\hat{\mu} \in (0,1)$ (typically close to 1) is called the *momentum parameter* and $\hat{\eta}$ is the learning rate. Let us consider a rescaled version of the above that is easier to analyze via continuous-time approximations. We redefine

$$\eta := \sqrt{\hat{\eta}}, \qquad v_k := \hat{v}_k/\sqrt{\hat{\eta}}, \qquad \mu := (1-\hat{\mu})/\sqrt{\hat{\eta}} \tag{4.6}$$

to obtain

$$\begin{aligned}v_{k+1} &= v_k - \mu\eta v_k - \eta\nabla f_{\gamma_k}(x_k)\\ x_{k+1} &= x_k + \eta v_{k+1}.\end{aligned} \tag{4.7}$$

In view of the rescaling, the range of momentum parameters we consider becomes $\mu \in (0, \eta^{-1/2})$, which we may replace by $(0,\infty)$ for simplicity.

Let us now derive the SME satisfied by the iterations (4.7). Observe that this is again a special case of (4.1) with x now replaced by (v,x) and

$$h(v,x,\gamma,\eta) = (-\mu v - \nabla f_\gamma(x), v - \eta\mu v - \eta\nabla f_\gamma(x))$$

In view of Thm. 14 and the results in Sec. 4.2, in order to derive the SMEs we simply match moments up to order 3. As in Sec. 4.2, let us define the one step difference

$$\Delta(v,x) := (v_1^{v,x,0} - v, x_1^{v,x,0} - x). \tag{4.8}$$

The following moment expansions are immediate.

Lemma 13 *Let $\Delta(x,v)$ be defined as in (4.8). We have*

(i) $\mathbb{E}\Delta_{(i)}(v,x) = \eta(-\mu v_{(i)} - \partial_{(i)}f(x), v) + \eta^2(0, -\mu v_{(i)} - \partial_{(i)}f(x)),$

(ii) $\mathbb{E}\Delta_{(i)}(v,x)\Delta_{(j)}(v,x) =$

$$\eta^2\begin{pmatrix}\mu^2 v_{(i)}v_{(j)} + \mu v_{(i)}\partial_{(j)}f(x) + \mu v_{(j)}\partial_{(i)}f(x)\\ \quad +\Sigma(x)_{(i,j)} + \partial_{(i)}\partial_{(j)}f(x) & -\mu v_{(i)}v_{(j)} - v_{(i)}\partial_{(j)}f(x)\\ -\mu v_{(i)}v_{(j)} - v_{(j)}\partial_{(i)}f(x) & v_{(i)}v_{(j)}\end{pmatrix}$$
$$+\mathcal{O}(\eta^3),$$

(iii) $\mathbb{E}\prod_{j=1}^3 |\Delta_{(i_j)}(v,x)| = \mathcal{O}(\eta^3),$

where $\Sigma(x) := \mathbb{E}(\nabla f_\gamma(x) - \nabla f(x))(\nabla f_\gamma(x) - \nabla f(x))^T$.

Proof The proof follows from direct calculation of the moments. ■

Hence, proceeding exactly as in Sec. 4.2 and using Lem.4, 13, we see that we may set

$$\begin{aligned}b_0(v,x) &= (-\mu v - \nabla f(x), v)\\ b_1(v,x) &= -\tfrac{1}{2}\left(\mu[\mu v + \nabla f(x)] - \nabla^2 f(x)v, \mu v + \nabla f(x)\right)\\ \sigma_0(v,x) &= \begin{pmatrix}\Sigma(x)^{1/2} & 0\\ 0 & 0\end{pmatrix}\end{aligned}$$

in order to match the moments. By similar mollification and limiting arguments as in Thm. 9, we arrive at the following approximation theorem, where we can see that the SME for MSGD takes the form of a Langevin equation.

Theorem 14 *Assume the same conditions as in Thm. 9 and that the drift terms in the following SDE are Lipschitz. Let $\mu > 0$ be fixed and define $\{V_t, X_t : t \in [0,T]\}$ as the stochastic process satisfying the SDE*

$$\begin{aligned} dV_t &= -[(\mu I + \tfrac{1}{2}\eta[\mu^2 I - \nabla^2 f(X_t)])V_t + (1+\tfrac{1}{2}\eta\mu)\nabla f(X_t)]dt + \sqrt{\eta}\Sigma(X_t)^{1/2}dW_t \quad V_0 = v_0, \\ dX_t &= [(1-\tfrac{1}{2}\eta\mu)V_t - \tfrac{1}{2}\eta\nabla f(X_t)]dt \quad X_0 = x_0, \end{aligned} \tag{4.9}$$

with $\Sigma(x)$ as defined in Thm. 9. Then, $\{(V_t, X_t) : t \in [0,T]\}$ is an order-2 weak approximation of the MSGD.

Moreover, if we relax the assumptions to Cor. 10, we have the order-1 weak approximation

$$\begin{aligned} dV_t &= -[\mu V_t + \nabla f(X_t)]dt + \sqrt{\eta}\Sigma(X_t)^{1/2}dW_t \quad V_0 = v_0, \\ dX_t &= V_t dt \quad X_0 = x_0. \end{aligned} \tag{4.10}$$

Note that by inverting the scaling (4.6), the order-1 SME (4.10) is the formal equation derived in Li et al. (2015). As discussed in Remark 11, using standard techniques the restrictive global Lipschitz conditions can be relaxed to a local Lipschitz condition together with a linear growth condition.

4.4. SME for a momentum variant: Nesterov accelerated gradient

It follows from the calculation above that we can also obtain the SME for the stochastic gradient version of the Nesterov accelerated gradient (NAG) method (Nesterov, 1983), which we refer to as SNAG. In the non-stochastic case, the NAG method has been analyzed using the ODE approach (Su et al., 2014). Therefore, the derivations in this section can be viewed as a stochastic parallel. The NAG method is sometimes used with stochastic gradients, and hence it is useful to analyze its properties in this setting and compare it to MSGD.

The unscaled NAG iterations are

$$\begin{aligned} \hat{v}_{k+1} &= \hat{\mu}_k \hat{v}_k - \hat{\eta}\nabla f_{\gamma_k}(x_k + \hat{\mu}_k \hat{v}_k) \\ x_{k+1} &= x_k + \hat{v}_{k+1} \end{aligned}$$

with $\hat{v}_0 = 0$, which differs from the momentum iterations as the gradient is now evaluated at a "predicted" position $x_k + \hat{\mu}_k\hat{v}_k$, instead of the original position x_k. Moreover, the momentum parameter $\hat{\mu}_k$ is now allowed to vary as k increases, and in fact, the usual choice of

$$\hat{\mu}_k = \tfrac{k-1}{k+2} \tag{4.11}$$

this has important links to stability and acceleration in the deterministic case (Nesterov, 1983; Su et al., 2014). In particular, it achieves $\mathcal{O}(1/k^2)$ convergence rate for general convex functions. On the other hand, a constant $\hat{\mu}_k$ is suggested for strongly convex functions (Nesterov, 2013). In the following, we shall first consider the case of constant momentum parameter with $\hat{\mu}_k \equiv \hat{\mu}$, and then the choice (4.11) subsequently.

Constant momentum. Using the same rescaling in (4.6), we have

$$\begin{aligned} v_{k+1} &= v_k - \mu\eta v_k - \eta\nabla f_{\gamma_k}(x_k + \eta(1-\mu\eta)v_k) \\ x_{k+1} &= x_k + \eta v_{k+1}. \end{aligned} \tag{4.12}$$

which is again (4.1) with

$$h(v,x,\gamma,\eta) = (-\mu v - \nabla f_\gamma(x+\eta(1-\mu\eta)v), v - \eta\mu v - \eta\nabla f_\gamma(x+\eta(1-\mu\eta)v))$$

Hence, we have the following moment expansion.

Lemma 15 *Let $\Delta(x,v) := (v_1^{v,x,0} - v, x_1^{v,x,0} - x)$. We have*

(i) $\mathbb{E}\Delta_{(i)}(v,x) = \eta(-\mu v_{(i)} - \partial_{(i)}f(x), v)$
$+ \eta^2(\partial_{(i)}\partial_{(j)}f(x)v_{(j)}, -\mu v_{(i)} - \partial_{(i)}f(x+v)) + \mathcal{O}(\eta^3)$,

(ii) $\mathbb{E}\Delta_{(i)}(v,x)\Delta_{(j)}(v,x) =$
$$\eta^2\begin{pmatrix} \mu^2 v_{(i)}v_{(j)} + \mu v_{(i)}\partial_{(j)}f(x+v) + \mu v_{(j)}\partial_{(i)}f(x+v) & \\ \quad +\Sigma(x+v)_{(i,j)} + \partial_{(i)}\partial_{(j)}f(x+v) & -\mu v_{(i)}v_{(j)} - v_{(i)}\partial_{(j)}f(x+v) \\ -\mu v_{(i)}v_{(j)} - v_{(j)}\partial_{(i)}f(x+v) & v_{(i)}v_{(j)} \end{pmatrix}$$
$+ \mathcal{O}(\eta^3)$,

(iii) $\mathbb{E}\prod_{j=1}^3 |\Delta_{(i_j)}(v,x)| = \mathcal{O}(\eta^3)$,

where $\Sigma(x) := \mathbb{E}(\nabla f_\gamma(x) - \nabla f(x))(\nabla f_\gamma(x) - \nabla f(x))^T$.

Proof The proof follows from direct calculation of the moments and Taylor's expansion. ■

Hence, we may match moments by setting

$$\begin{aligned} b_0(v,x) &= (-\mu v - \nabla f(x), v) \\ b_1(v,x) &= -\tfrac{1}{2}\left(\mu[\mu v + \nabla f(x)] + \nabla^2 f(x)v, \mu v + \nabla f(x)\right) \\ \sigma_0(v,x) &= \begin{pmatrix} \Sigma(x)^{\frac{1}{2}} & 0 \\ 0 & 0 \end{pmatrix} \end{aligned}$$

from which we obtain the following approximation theorem for SNAG.

Theorem 16 *Assume the same conditions as in Thm. 14. Define $\{V_t, X_t : t \in [0,T]\}$ as the stochastic process satisfying the SDE*

$$\begin{aligned} dV_t &= -[(\mu I + \tfrac{1}{2}\eta[\mu^2 I + \nabla^2 f(X_t)])V_t + (1+\tfrac{1}{2}\eta\mu)\nabla f(X_t)]dt + \sqrt{\eta}\Sigma(X_t)^{1/2}dW_t \quad V_0 = v_0, \\ dX_t &= [(1-\tfrac{1}{2}\eta\mu)V_t - \tfrac{1}{2}\eta\nabla f(X_t)]dt \quad X_0 = x_0, \end{aligned} \tag{4.13}$$

with Σ as defined in Thm. 14. Then, $\{(V_t, X_t) : t \in [0,T]\}$ is an order-2 weak approximation of SNAG. Moreover, the same order-1 weak approximation of MSGD in (4.10) holds for the SNAG.

The result above shows that for constant momentum parameters, the modified equations for MSGD and the SNAG are equivalent at leading order, but differ when we consider the second order modified equation. Let us now discuss the case where the momentum parameter is allowed to vary.

Varying momentum. Now let us take $\hat{\mu}$ as in (4.11). Then, using the same rescaling arguments, we arrive at

$$\begin{aligned} v_{k+1} &= v_k - \mu_k \eta v_k - \eta \nabla f_{\gamma_k}(x_k + \eta(1-\mu_k\eta)v_k) \\ x_{k+1} &= x_k + \eta v_{k+1}. \end{aligned} \tag{4.14}$$

with $\mu_k = 3/(2\eta + k\eta)$. Now, in order to apply our theoretical results to deduce the SME, simply notice that we may introduce an auxiliary scalar variable

$$z_{k+1} = z_k + \eta, \qquad z_0 = 0.$$

Then, $\mu_k = 3/(2\eta + z_k)$, and hence all terms are now not explicitly k-independent, thus we may proceed formally as in the previous sections to arrive at the order-1 SME for SNAG with varying momentum

$$\begin{aligned} dV_t &= -[\tfrac{3}{t}V_t + \nabla f(X_t)]dt + \sqrt{\eta}\Sigma(X_t)^{1/2}dW_t \quad V_0 = 0, \\ dX_t &= V_t dt \quad X_0 = x_0. \end{aligned} \tag{4.15}$$

This result is formal because the term $3/t$ does not satisfy our global Lipschitz conditions, unless we restrict our interval to some $[t_0, T]$ with $t_0 > 0$, in which case the above result becomes rigorous. Alternatively, some limiting arguments have to be used to establish well-posedness of the equation on $[0, T]$ individually. We shall omit these analyses in the current paper, and only consider (4.15) on some interval $[t_0, T]$, where initial conditions are then replaced by (v_{t_0}, x_{t_0}). As a point of comparison, (4.15) reduces to the ODE derived in Su et al. (2014) if $\Sigma(x) \equiv 0$ (i.e. the gradients are non-stochastic).

5. Applications of the SMEs to the analysis of SGA

In this section, we apply the SME framework developed to analyze the dynamics of the three stochastic gradient algorithm variants discussed above, namely SGD, MSGD and SNAG. We shall focus on simple but non-trivial models where to a large extent, analytical computations using SME are tractable, giving us key insights into the algorithms that are otherwise difficult to obtain without appealing to the continuous formalism presented in this paper. We consider primarily the following model:

Model: Let $H \in \mathbb{R}^{d\times d}$ be a symmetric, positive definite matrix. Define the sample objective

$$\begin{aligned} f_\gamma(x) &:= \tfrac{1}{2}(x-\gamma)^T H(x-\gamma) - \tfrac{1}{2}\mathrm{Tr}(H) \\ \gamma &\sim \mathcal{N}(0, I) \end{aligned} \tag{5.1}$$

which gives the total objective $f(x) \equiv \mathbb{E} f_\gamma(x) = \frac{1}{2}x^T H x$.

5.1. SME analysis of SGD

We first derive the SME associated with (5.1). For simplicity, we will only consider the order-1 SME (4.5). A direct computation shows that $\Sigma(x) = H^2$ and so the SME for SGD applied to model (5.1) is

$$dX_t = -HX_t dt + \sqrt{\eta} H dW_t,$$

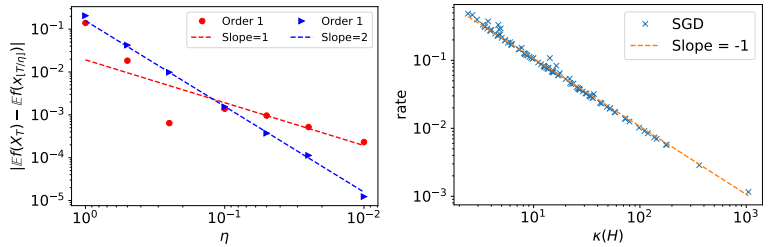
This is a multi-dimensional Ornstein-Uhlenbeck (OU) process and admits the explicit solution

$$X_t = e^{-tH}\left(x_0 + \sqrt{\eta}\int_0^t e^{sH} H dW_s\right).$$

Observe that for each $t \geq 0$, the distribution of X_t is Gaussian. Using Itô's isometry, we then deduce the dynamics of the objective function

$$\begin{aligned} \mathbb{E} f(X_t) =& \tfrac{1}{2}x_0^T H e^{-2tH} x_0 + \tfrac{1}{2}\eta \int_0^t \mathrm{Tr}(H^3 e^{-2(t-s)H}) ds \\ =& \tfrac{1}{2}x_0^T H e^{-2tH} x_0 + \tfrac{1}{4}\eta \sum_{i=1}^n \lambda_i^2(H)(1 - e^{-2t\lambda_i(H)}). \end{aligned} \tag{5.2}$$

The first term decays linearly with asymptotic rate $2\lambda_d(H)$, and the second term is induced by noise, and its asymptotic value is proportional to the learning rate η. This is the well-known two-phase behavior of SGD under constant learning rates: an initial descent phase induced by the deterministic gradient flow and an eventual fluctuation phase dominated by the variance of the stochastic gradients. In this sense, the SME makes the same predictions, and in fact we can see that it approximates the SGD iterations well as η decreases (Fig. 5.1(a)), according to the rates we derived in Thm. 9 and Cor. 10.



(a) (b)

Figure 5.1: SME prediction vs SGD dynamics. (a) SME as a weak approximation of the SGD. We compute the weak error with test function g equal to f (see Thm. 9). As predicted by our analysis, the order-2 SME (4.4) (order-1 SME (4.5)) should give a slope $= 2$ (1) decrease in error as η decreases (note that the x-axis is flipped). The SME solution is computed using an exact formula derived by the application of Itô isometry and the SGD expectation is averaged over 1e6 runs. We took $T = 2.0$. We see that the predictions of Thm. 9 and Cor.10 hold. (b) Descent rate vs condition number. H is generated with different condition numbers, and the resulting descent rate of SGD is approximately $\propto \kappa(H)^{-1}$, as predicted by the SME.

Moreover, notice that by the identification $t = k\eta$ (k is the SGD iteration number), the SME analysis tells us that the asymptotic linear convergence rate (in k, i.e. rate $\sim$

$-\log[\mathbb{E}f(x_k)]/k)$ in the descent phase of the SGD is $2\lambda_d(H)\eta$. For numerical stability (even in the non-stochastic case), we usually require $\eta \propto 1/\lambda_1(H)$, thus the maximal descent rate is inversely proportional to the condition number $\kappa(H) = \lambda_1(H)/\lambda_d(H)$. We validate this observation by generating a collection of H's with varying condition numbers and applying SGD with $\eta \propto 1/\lambda_1(H)$. In Fig 5.1(b), we plot the initial descent rates versus the condition number of H and we observe that we indeed have rate $\propto \kappa(H)^{-1}$.

Alternate model. Now, we consider a slight variation of the model (5.1). The goal is show that the dynamics of SGD (and the corresponding SME) is not always Gaussian-like and thus using the OU process to model the SGD is not always valid. Given the same positive-definite matrix H, we diagonalize it in the form $H = QDQ^T$ where Q is an orthogonal matrix and D is a diagonal matrix of eigenvalues. We then define the sample objective

$$\begin{aligned} &f_\gamma(x) := \tfrac{1}{2}(Q^T x)^T[D + \operatorname{diag}(\gamma)](Q^T x) \\ &\gamma \sim \mathcal{N}(0, I) \end{aligned} \tag{5.3}$$

which gives the same total objective $f(x) \equiv \mathbb{E}f_\gamma(x) = \frac{1}{2}x^T H x$. However, we have a different expression for $\Sigma(x)$

$$\Sigma(x) = Q\operatorname{diag}(Qx)^2 Q^T,$$

which gives the SME

$$\begin{aligned} dX_t &= -HX_t dt + \sqrt{\eta}Q|\operatorname{diag}(Q^T X_t)|Q^T dW_t \\ &\overset{\text{in distribution}}{=} -HX_t dt + \sqrt{\eta}Q\operatorname{diag}(Q^T x)Q^T dW_t. \end{aligned}$$

We can rewrite the above as

$$dX_t = -HX_t dt + \sqrt{\eta}\sum_{l=1}^{d} Q^{(l)} X_t dW_{(l),t},$$

where $Q^{(l)} = Q\operatorname{diag}(Q_{(l,\cdot)})Q^T$ and $Q_{(l,\cdot)}$ denotes the l^{th} row of Q. By observing that every pair of $\{H, Q^{(1)}, \ldots, Q^{(d)}\}$ commute, we have the explicit solution

$$X_t = e^{-\frac{1}{2}\eta t + \sqrt{\eta}\sum_{l=1}^{d} Q^{(l)} W_{(l),t}} e^{-Ht} x_0.$$

which is a multi-dimensional Black-Scholes (Black and Scholes, 1973) type of stochastic process. In particular, the distribution is not Gaussian of any $t > 0$. Nevertheless, we may take expectation to obtain

$$\mathbb{E}f(X_t) = \tfrac{1}{2}e^{\eta t}x_0^T H e^{-2Ht}x_0.$$

This immediately implies the following interesting behavior: if $\eta < 2\lambda_d(H)$, then $2H - \eta I$ is positive definite and so $\mathbb{E}f(X_t) \to 0$ exponentially at constant, non-zero η; Otherwise, depending on initial condition x_0, the objective may not converge to 0. In particular, if $\eta > 2\lambda_d(H)$ (which happens quite often if the condition number of H is large) and x_0

is in general position, then we have asymptotic exponential divergence. This is a variance-induced divergence typically observed in Black-Scholes and geometric Brownian motion type of stochastic processes. The term "variance-induced" is important here since the deterministic part of the evolution equation is mean-reverting and in fact is identical to the stable OU process studied earlier. In Fig. 5.2(a), (b), we show the correspondence of the SME findings with the actual dynamics of the SGD iterations. In particular, we see in Fig. 5.2(c) that for small η, we have exponential convergence of the SGD at constant learning rates, whereas for $\eta > 2\lambda_d(H)$, the SGD iterates start to oscillate wildly and its mean value is dominated by few large values and diverges approximately at the rate predicted by the SME. Note that this divergence is predicted to be at a finite η, and from the theory developed so far we cannot conclude that the SME approximation always holds accurately at this regime (but the approximation is guaranteed for η sufficiently small). Nevertheless, we observe at least in this model that the variance-induced divergence of the SGD happens as predicted by the SME.

5.2. SME analysis of MSGD

Let us now use the SME to analyze MSGD applied to model (5.1). We have shown earlier that $\Sigma(x) = H$. Thus, according to Thm. 14, the order-1 SME for MSGD is

$$\begin{aligned} dV_t &= -[\mu V_t + HX_t]dt + \sqrt{\eta}H dW_t, \\ dX_t &= V_t dt, \end{aligned} \tag{5.4}$$

with $X_0 = x_0$ and $V_0 = 0$. If we set $Y_t := (V_t, X_t) \in \mathbb{R}^{2d}$, U_t a $2d$-dimensional Brownian motion with first d coordinates equal to W_t, and define block matrices

$$A := \begin{pmatrix} \mu I & H \\ -I & 0 \end{pmatrix}, \qquad B := \begin{pmatrix} H & 0 \\ 0 & 0 \end{pmatrix}, \tag{5.5}$$

we can then write (5.4) as

$$dY_t = -AY_t + \sqrt{\eta}B dU_t, \qquad Y_0 = (0, x_0),$$

which admits the explicit solution

$$Y_t = e^{-At}\left(Y_0 + \sqrt{\eta}\int_0^t e^{As}B dU_s.\right).$$

By Itô's isometry, we have

$$\mathbb{E}f(X_t) = \tfrac{1}{2}\left[|\operatorname{diag}(0,H)^{1/2}e^{-At}Y_0|^2 + \eta\int_0^t |\operatorname{diag}(0,H)^{1/2}e^{-(t-s)A}B|^2 ds\right], \tag{5.6}$$

One can see immediately that a similar two-phase behavior is present, but the property of the descent phase now hinges on the spectral properties of the matrix A (instead of H). Before proceeding, we first observe that the eigenvalues of A can be written as

$$\lambda(A) := \{\Lambda_+, \Lambda_-\}, \qquad \Lambda_{\pm,i} = \tfrac{1}{2}\left(\mu \pm \sqrt{\mu^2 - 4\lambda_i(H)}\right), \qquad i = 1, 2, \ldots, d. \tag{5.7}$$

In particular, $\Re\lambda_i(A) > 0$ for all i as long as $\mu > 0$. We also need the following simple result concerning the decay of the norm of e^{-tA} if all eigenvalues of A have positive real part.

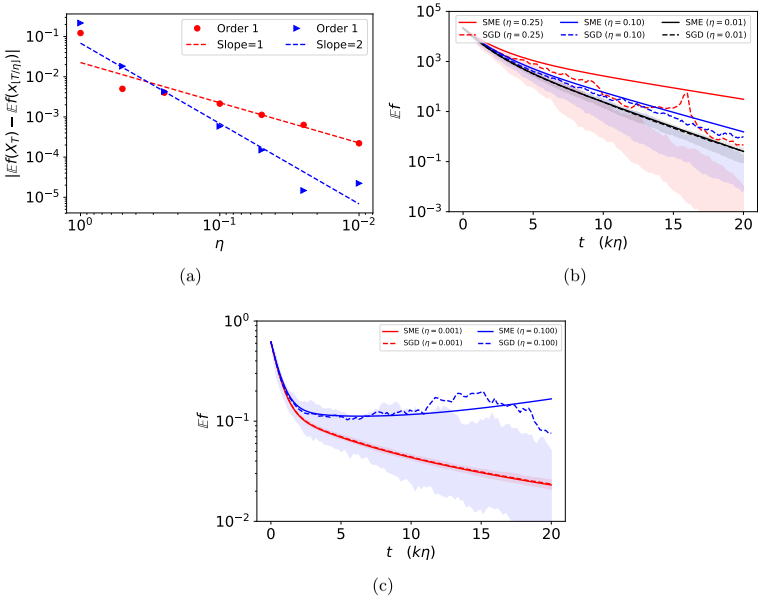


Figure 5.2: SME prediction vs SGD dynamics for the model variant (5.3). (a) Order of convergence of the SME to the SGD. We use the same setup as in Fig. 5.1(a). Observe that our analysis again predicts the correct rate of weak error decay as η decreases. (b) SGD paths vs order-1 SME prediction. Solid lines are SME exact solutions and dotted lines are means of SGD paths over 500 runs, and the 25-75 percentiles are shaded. We observe convergence of $\mathbb{E}f$ at constant η, and that the sample mean is dominated by few large values, as observed by the deviation of the percentiles from the mean. (b) Variance-induced explosion. As predicted by the SME analysis, if $\eta > 2\lambda_d(H)$ (Here, $\lambda_d(H) = 0.01$), variance-induced instability sets in.

Lemma 17 *Let A be a real square matrix such that all eigenvalues have positive real part. Then,*

(i) For each $\epsilon > 0$, there exists a constant $C_\epsilon > 0$ independent of t but depends on ϵ, such that

$$|e^{-tA}| \leq C_\epsilon e^{-t(\min_i \Re\lambda_i(A) - \epsilon)}$$

(ii) If in addition A is diagonalizable, then there exists a constant $C > 0$ independent of t such that

$$|e^{-tA}| \leq Ce^{-t\min_i \Re\lambda_i(A)}$$

Proof See Appendix E. ■

With the above results, we can now characterize the decay of the objective under momentum SGD. From expression (5.7), we see that as long as $\mu^2 \neq 4\lambda_i$ for any $i = 1, \ldots, d$, A has $2d$ distinct eigenvalues and is hence diagonalizable. We shall hereafter assume that μ is in general position such that this is the case. Using Lem. 17 and expression (5.6), we arrive at the estimate

$$\mathbb{E}f(X_t) \leq \tfrac{1}{2}C^2|x_0|^2\lambda_1(H)e^{-2t\min_i \Re\lambda_i(A)} + \tfrac{1}{2}\frac{\eta C^2\lambda_1(H)^3}{\min_i \Re\lambda_i(A)}(1 - e^{-2t\min_i \Re\lambda_i(A)}). \tag{5.8}$$

This result tells us that the convergence rate of the descent phase is now controlled by the minimum real part of the eigenvalues of A, instead of the minimum eigenvalue of H. In particular, we achieve the best linear convergence rate by maximizing the smallest real part of the eigenvalues of A. This leads to the following optimization problem for the optimal convergence rate:

$$\sup_{\mu\in(0,\infty)} \min_{i=1,\ldots,d} \min_{s\in\{+1,-1\}} \left\{\Re\left[\mu + s\sqrt{\mu^2 - 4\lambda_i(H)}\right]\right\}$$

Since H is positive definite, the supremum is attained at $\mu^* = 2\sqrt{\lambda_d(H)}$ with the rate also equal to $2\sqrt{\lambda_d(H)}$. However, note that if we take $\mu = \mu^*$ exactly, one can check that A is no longer diagonalizable and by Lem. 17, the rate is slightly diminished, thus technically we can take μ as close to μ^* as we like (i.e. the rate is as close to $2\sqrt{\lambda_d(H)}$ as we like), but exact equality is not technically deducible from current results. In Fig. 5.3(c), we demonstrate the optimal choice of μ and its effect on the convergence rate. Moreover, observe that as μ increases, the number of complex eigenvalues start to decrease, and the magnitudes of the imaginary parts of the complex eigenvalues also decrease. This signifies that increasing μ causes oscillations to decreases in magnitude and frequency. This is again corroborated by numerical experiments (Fig. 5.3(c)).

Another interesting observation is that by the identification $t = \eta k$, the descent rate (in terms of k) is $2\sqrt{\lambda_d(H)}\eta$. As before, if we choose the maximal stable learning rate we would have $\hat{\eta} \propto 1/\lambda_1(H)$ ($\hat{\eta} = \eta^2$ according to the scaling introduced in (4.6)). Thus, for the MSGD iterates we have its descent rate $\propto \kappa(H)^{-1/2}$, which is a huge improvement over SGD, whose rate is $\propto \kappa(H)^{-1}$, especially for badly conditioned matrices where $\kappa(H) \gg 1$. In Fig. 5.3(d), we plot the MSGD initial descent rates for varying condition numbers of H. Again, we observe that the SME analysis gives the correct characterization of the precise dynamics and recovers the square-root relationship with condition number.

Finally, let us discuss the effect of adding momentum to the asymptotic fluctuations due to noisy gradients. Note that it is not correct to conclude, using Eq. (5.8), that taking $\mu \approx \mu^*$ also gives the lowest fluctuations. This is because the constant C depends on μ as well, as is evidenced in the proof of Lem. 17, which shows that C depends on the conditioning of the

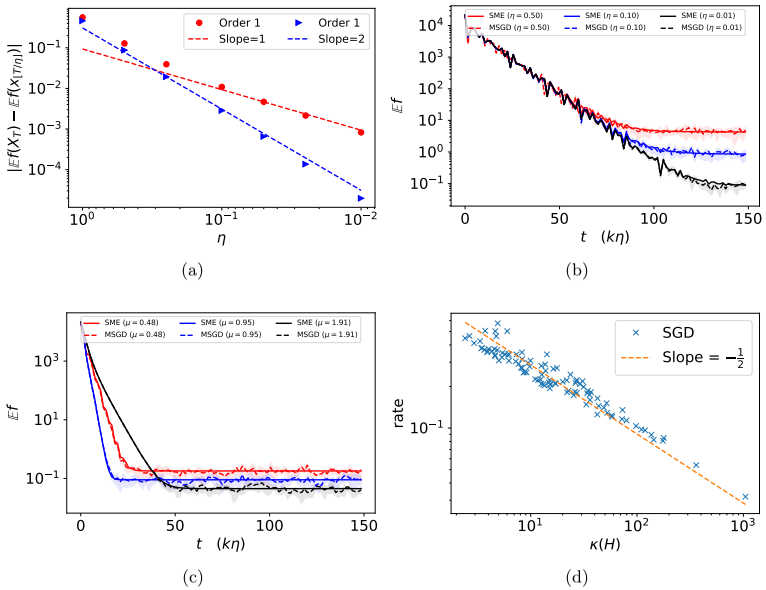


Figure 5.3: SME prediction vs MSGD dynamics. (a) and (b) SME vs MSGD dynamics at $\mu = 0.1$ for different learning rates η. As before, the SME prediction gets better as η decreases according to the predicted order. Notice also the presence of oscillations, due to the complex eigenvalues of A. (c) Optimal descent rate of the SGD is achieved by the SME prediction $\mu = \mu^*$, which is 0.95 in this case. Notice that exactly as predicted by the SME, increasing μ decreases the oscillation frequency and magnitude (due to having fewer complex eigenvalues and smaller imaginary parts), as well as the asymptotic fluctuations (due to formula (5.9)). (d) Descent rate vs condition number. H is generated with different condition numbers, and the descent rate of MSGD is $\propto \kappa(H)^{-1/2}$, as predicted by the SME, which for badly conditioned H gives a large improvement.

eigenvector matrix of A. To proceed, we do not use the bounds (5.8). Instead, we explicitly diagonalize A and after some computations, we arrive at the exact expression for $\mathbb{E}f(X_t)$

$$\mathbb{E}f(X_t) = \tfrac{1}{2}|\operatorname{diag}(0,H)^{1/2}e^{-At}Y_0|^2 \tag{5.9}$$

$$+ \tfrac{1}{2}\eta\sum_{i=1}^{d}\frac{\lambda_i^3}{|\mu^2-4\lambda_i|}\left[\frac{1-e^{-2t\Re\Lambda_{+,i}}}{2\Re\Lambda_{+,i}} + \frac{1-e^{-2t\Re\Lambda_{-,i}}}{2\Re\Lambda_{-,i}} - 2R(t,\mu,\lambda_i(H))\right] \tag{5.10}$$

where

$$R(t,\mu,\lambda) = \begin{cases} \frac{1-e^{-t\mu}}{\mu} & \mu \geq 2\sqrt{\lambda} \\ \frac{\mu+\sqrt{4\lambda-\mu^2}e^{-\mu t}\sin(t\sqrt{4\lambda-\mu^2})-\mu e^{-\mu t}\cos(t\sqrt{4\lambda-\mu^2})}{4\lambda} & \mu < 2\sqrt{\lambda} \end{cases}. \tag{5.11}$$

In particular, the asymptotic loss value induced by noise is

$$\lim_{t\to\infty}\mathbb{E}f(X_t) = \tfrac{1}{2}\eta\sum_{i=1}^{d}\frac{\lambda_i(H)^3}{|\mu^2-4\lambda_i(H)|}\left[\frac{1}{2\Re\Lambda_{+,i}} + \frac{1}{2\Re\Lambda_{-,i}} - 2\min\left\{\frac{\mu}{4\lambda_i(H)}, \frac{1}{\mu}\right\}\right] \tag{5.12}$$

Observe that this function (in fact, each term in the sum) is monotone-decreasing in μ, and for $\mu \ll 1$ it scales like μ^{-1}, and for $\mu \gg 1$ it scales like μ^{-3}. Thus, increasing the momentum parameter decreases the asymptotic noise in the iterates, i.e. decreases the asymptotic value of $\mathbb{E}f$, which should be 0 in the absence of noise. This again agrees with the actual MSGD dynamics (Fig. 5.3(b)). Consequently, to obtain "optimal" tradeoff between descent and noise, we would like a momentum schedule that equals μ^* in the descent phase and increases to infinity (in the original scaling this corresponds to $\hat{\mu} \to 0$) as we approach the optimum. Finding this optimal schedule can be cast as an optimal control problem (Li et al., 2017), and a rigorous investigation of these approaches will be considered in subsequent work.

5.3. SME analysis of SNAG

Finally, let us see what we can say, using the SME approach, about the difference between MSGD and SNAG in this stochastic setting. Let us first consider the case of constant momentum. From Thm. 16, we know that the order-1 SMEs are identical, so we must consider higher order SMEs. A straightforward computation yields the following order-2 SMEs for MSGD and SNAG (again we let $Y_t = (V_t, X_t)$)

$$\begin{aligned} &\text{MSGD:} && dY_t = -A_1 Y_t + \sqrt{\eta}B dU_t, && Y_0 = (0, x_0), \\ &\text{SNAG:} && dY_t = -A_2 Y_t + \sqrt{\eta}B dU_t, && Y_0 = (0, x_0), \end{aligned}$$

where $A_i = A + \frac{1}{2}\eta E_i$ with A, B as defined in (5.5) and

$$E_1 := \begin{pmatrix} \mu^2 I - H & \mu H \\ \mu I & H \end{pmatrix}, \qquad E_2 := \begin{pmatrix} \mu^2 I + H & \mu H \\ \mu I & H \end{pmatrix}.$$

From the analysis in Sec. 4.3, the descent rate is dominated by the minimal real parts of the eigenvalues of A_i, which are respectively

$$\lambda(A_1) = \left\{\tfrac{1}{4}\left(\mu(\eta\mu+2) \pm \sqrt{\mu^2(\eta\mu+2)^2 + 4\eta^2\lambda_i(H)^2 - 8\lambda_i(H)(\eta\mu+2)}\right), \quad i = 1,\dots,d\right\}$$

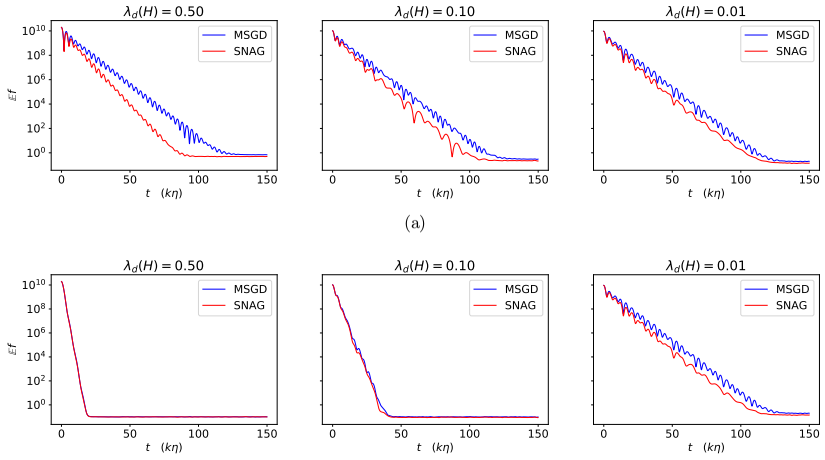
$$\lambda(A_2) = \left\{\tfrac{1}{4}\left(\mu(\eta\mu+2) + 2\eta\lambda_i(H) \pm \sqrt{\eta\mu+2}\sqrt{\mu^2(\eta\mu+2) + 4\lambda_i(H)(\eta\mu-2)}\right), \quad i = 1,\dots,d\right\}$$

We observe that for small μ (i.e. $\hat{\mu} \approx 1$ in the usual MSGD scaling), the terms in square-roots are negative and hence for the same small μ, the convergence rate of SNAG (in terms of initial descent rate) is $\frac{1}{2}\eta\lambda_d(H)$ larger than that of MSGD. This says in particular that

for H with larger $\lambda_d(H)$, the acceleration is more pronounced. Moreover, recall that the asymptotic fluctuations is given by

$$\eta \lim_{t\to\infty} \int_0^t |\operatorname{diag}(0,H)^{1/2} e^{-(t-s)(A+\frac{1}{2}\eta E_i)} B|^2 ds.$$

Without performing tedious computations, we can see that since $E_2 - E_1$ is positive definite, the exponential for the SNAG case decays more rapidly, and hence the eventual fluctuations are expected to be lower. These observations from the SME are again consistent with the behavior of their SGA counter-parts, as shown in Fig. 5.4(a). On the other hand, if we pick μ for each case by separately maximizing the smallest real part of the eigenvalues (as in Sec. 4.3), we obtain similar convergence rates up to η^2. In other words, if we tune μ well, there would be no difference between MSGD and SNAG in terms of descent rate (Fig. 5.4(b)).



(a)

(b)

Figure 5.4: MSGD vs SNAG (with constant momentum) dynamics for $\eta = 0.1$ and different $\lambda_d(H)$. (a) Dynamics at fixed $\mu = 0.2$. We observe that as predicted by the SME analysis, SNAG enjoys a faster linear convergence rate in the descent phase, as well as lower asymptotic fluctuations. The acceleration is indeed more pronounced for larger $\lambda_d(H)$. (b) When, μ for each case is chosen optimally for the descent (by maximizing the minimal real part of the eigenvalues of A_1, A_2 respectively), the dynamics becomes similar.

Now, let us discuss the varying momentum case. According to (4.15), for some small $t_0 > 0$ we have the order-1 SME for $t \in [t_0, T]$

$$dY_t = -A_t Y_t + \sqrt{\eta} B dU_t, \qquad Y_{t_0} = (v_{t_0}, x_{t_0}) \qquad A_t := \begin{pmatrix} \frac{3}{t} I & H \\ -I & 0 \end{pmatrix},$$

and B is defined as in (5.5). This admits the explicit solution

$$Y_t = e^{-(t-t_0)\tilde{A}_t} \left(Y_{t_0} + \sqrt{\eta} \int_{t_0}^t e^{s\tilde{A}_s} B dU_s. \right), \qquad \tilde{A}_t := \begin{pmatrix} 3\frac{\log(t/t_0)}{t-t_0} I & H \\ -I & 0 \end{pmatrix}.$$

The eigenvalues of $\tilde{A}_t$ are

$$\lambda(\tilde{A}_t) = \left\{ \frac{1}{2} \left(3\frac{\log(t/t_0)}{t-t_0} \pm \sqrt{9[\frac{\log(t/t_0)}{t-t_0}]^2 - 4\lambda_i(H)} \right), \qquad i = 1, \dots, d \right\}.$$

Since there is no lower-bound on the minimal real part, the descent rate before the onset of fluctuations is sub-linear. This is expected because the $\mathcal{O}(1/t)$ momentum schedule is suited for non-strongly-convex functions, whereas constant momentum is more appropriate for strong-convex functions (Nesterov, 2013). Furthermore, we observe that since the real parts of all eigenvalues of $\tilde{A}_t$ converge to 0 as $t \to \infty$, according to the analysis in Sec. 4.3, the asymptotic fluctuations due to noise should be large. Fig. 5.5 confirms these observations and further suggests that in the case of stochastic gradient methods, more careful momentum schedules must be derived in order to balance descent and fluctuations, e.g. using the optimal control framework presented in Li et al. (2017).

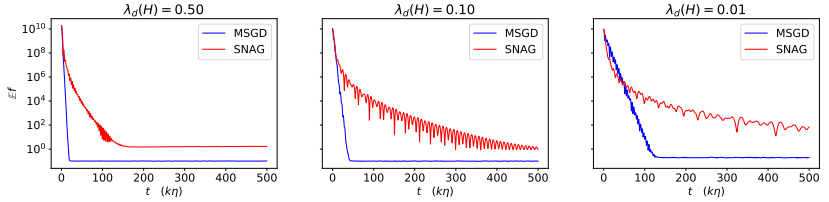


Figure 5.5: MSGD vs SNAG (with dynamic momentum according to Nesterov's choice (4.11)) dynamics for $\eta = 0.1$ and different $\lambda_d(H)$. We see that the convergence is indeed sub-linear, and moreover, the asymptotic fluctuations are large compared with MSGD, in which case μ here is picked to achieve optimal descent rate.

6. Conclusion

In this paper, we developed the general mathematical foundation of the stochastic modified equations framework for analyzing stochastic gradient algorithms. In particular, we demonstrate that this approach is (1) rigorous, (2) flexible and (3) useful. Indeed, the technique of weak approximations provides a precise mathematical framework for analyzing the relationship between stochastic gradient algorithms and stochastic differential equations. This should be contrasted with strong approximations in the numerical analysis of SDEs, where approximations are required to hold path-wise, say in the mean-square sense (Kloeden and Platen, 2011). The weak formulation greatly increases the flexibility of modelling different

type of stochastic gradient algorithms, as we have demonstrated in Sec. 4. In fact, the main result relating discrete-time algorithms and continuous-time SDEs (Thm. 3) is proved in a fairly general setting that allows one to derive a variety of SMEs for different variations of the SGAs (Sec. 4). Finally, in Sec. 5, we demonstrated the usefulness of the SME approach through explicit calculations. This is enabled by the precise approximation nature of the SMEs and the application of stochastic calculus tools. In particular, we uncovered interesting behaviors of SGAs when applied to a simple yet non-trivial setting, including the tradeoff of descent and fluctuations, the relationship with condition numbers and the subtle differences of MSGD and SNAG in the stochastic setting. In the subsequent work in the series, we will focus on applications, where we extend the SME formalism to study adaptive algorithms and related topics.

Acknowledgements

QL is supposed by the Agency for Science, Technology and Research (A*STAR), Singapore. WE is supported by the Office of Naval Research, USA, ONR N00014-13-1-0338.

Appendix A. General existence, uniqueness and moment estimates for SDEs

In this section, we establish general existence, uniqueness and moment estimates for the stochastic differential equations that we encounter in this paper. The results here will be used throughout the subsequent proofs. We note that although similar well-posedness results are well-known, here we require a slightly more general version (where the drift and diffusion terms are themselves random functions) in order to deal with the analysis in Appendix B. Moreover, we need uniform estimates with respect to parameters (η, ϵ), which warrants the following standard but necessary analysis.

Let $T > 0$ and Q be a subset of a Euclidean space. For $(x, t, q) \in \mathbb{R}^d \times [0, T] \times Q$, let $B(x, t, q)$ be a d-dimensional random vector and $S(x, t, q)$ be a $d \times d$-dimensional random matrix. Throughout this section we assume:

Assumption A.1 *The random functions B, S satisfy the following:*

(i) B, S are W_t-adapted and continuous in $(x, t) \in \mathbb{R}^d \times [0, T]$ almost surely

(ii) B, S satisfy a uniform linear growth condition, i.e. there exists a non-random constant $L > 0$ such that

$$|B(x,t,q)|^2 + |S(x,t,q)|^2 \le L^2(1+|x|^2) \quad a.s.$$

for all $x, y \in \mathbb{R}^d$, $t \in [s, T]$, $q \in Q$.

(iii) B, S satisfy a uniform Lipschitz condition in x, i.e.

$$|B(x,t,q) - B(y,t,q)| + |S(x,t,q) - S(y,t,q)| \le L|x-y| \quad a.s.$$

for all $t \in [s, T]$, $q \in Q$.

Theorem 18 *Let $s \in [0, T)$ and for each $q \in Q$, let $\{\phi_t^q : t \in [s, T]\}$ be a $\mathbb{R}^d$-valued, W_t-adapted random process that is continuous in $t \in [s, T]$ almost surely, with*

$$\sup_{q\in Q} \mathbb{E} \sup_{t\in[s,T]} |\phi_t^q|^2 < \infty. \tag{A.1}$$

Then, for each $q \in Q$ the stochastic differential equation

$$\xi_t^q = \phi_t^q + \int_s^t B(\xi_v^q, v, q)dv + \int_s^t S(\xi_v^q, v, q)dW_v \tag{A.2}$$

admits a unique solution $\{\xi_t^q : t \in [s, T]\}$ which is continuous for $t \in [s, T]$ a.s. and satisfies

$$\sup_{q\in Q} \mathbb{E} \sup_{t\in[s,T]} |\xi_t^q|^2 \le C\left(1 + \sup_{q\in Q} \mathbb{E} \sup_{t\in[s,T]} |\phi_t^q|^2\right) \tag{A.3}$$

for some constant $C > 0$ that depends only on L, T.

Proof For each $q \in Q$, let us define the recursion

$$\xi_t^{q,0} = \phi_t^q$$
$$\xi_t^{q,m+1} = \phi_t^q + \int_s^t B(\xi_v^{q,m}, v, \eta)dv + \int_s^t S(\xi_v^{q,m}, v, \eta)dW_v, \quad m \geq 0.$$

Note that Assumption A.1 implies each $\xi_t^{q,m}$ is well-defined. Now, let $m \geq 1$. By Itô's isometry, we have

$$|\xi_t^{q,m+1} - \xi_t^{q,m}|^2 \leq 2\left|\int_s^t B(\xi_v^{q,m}, v, q) - B(\xi_v^{q,m-1}, v, q)dv\right|^2 \tag{A.4}$$
$$+ 2\left|\int_s^t S(\xi_v^{q,m}, v, q) - S(\xi_v^{q,m-1}, v, q)dW_v\right|^2 \tag{A.5}$$
$$\leq 2T\int_s^t |B(\xi_v^{q,m}, v, q) - B(\xi_v^{q,m-1}, v, q)|^2 dv \tag{A.6}$$
$$+ 2\int_s^t |S(\xi_v^{q,m}, v, q) - S(\xi_v^{q,m-1}, v, q)|^2 dv. \tag{A.7}$$

Thus, applying the Lipschitz assumption A.1 (iii) and taking expectations, we get

$$\mathbb{E}|\xi_t^{q,m+1} - \xi_t^{q,m}|^2 \leq 2L^2(1+T)\int_s^t \mathbb{E}|\xi_v^{q,m} - \xi_v^{q,m-1}|^2 dv. \tag{A.8}$$

Now, for $m = 0$, Assumption A.1 (ii) together with (A.1) gives

$$\mathbb{E}|\xi_t^{q,1} - \xi_t^{q,0}|^2 \leq C\int_s^t \left(1 + \sup_{q\in Q}\mathbb{E}|\phi_v^q|^2\right)dv \leq C'(t-s). \tag{A.9}$$

Combining (A.8) and (A.9), we have

$$\mathbb{E}|\xi_t^{q,m+1} - \xi_t^{q,m}|^2 \leq \tfrac{[C(t-s)]^{m+1}}{(m+1)!}, \qquad m \geq 0 \tag{A.10}$$

for some $C > 0$ that only depends on T, L and $C_\phi := \sup_{q\in Q}\mathbb{E}\sup_{t\in[s,T]}|\phi_t^q|^2$. Moreover, Eq. (A.4) implies

$$\mathbb{E}\sup_{t\in[s,T]}|\xi_t^{q,m+1} - \xi_t^{q,m}|^2 \leq 2L^2T\int_s^T \mathbb{E}|\xi_t^{q,m} - \xi_t^{q,m-1}|^2 dt$$
$$+ 2\mathbb{E}\sup_{t\in[s,T]}\left|\int_s^t S(\xi_v^{q,m}, v, q) - S(\xi_v^{q,m-1}, v, q)dW_v\right|^2.$$

Estimate (A.10) implies the last stochastic integral is a martingale, and hence using Doob's maximal inequality and Itô's isometry, we have

$$\mathbb{E}\sup_{t\in[s,T]}|\xi_t^{q,m+1} - \xi_t^{q,m}|^2 \leq 2L^2T\int_s^T \mathbb{E}|\xi_t^{q,m} - \xi_t^{q,m-1}|^2 dt + 8L^2\int_s^T \mathbb{E}|\xi_t^{q,m} - \xi_t^{q,m-1}|^2 dt$$
$$\leq 2L^2(T+4)\tfrac{C^m T^{m+1}}{(m+1)!} = 2L^2(T+4)\tfrac{C^m T^{m+1}}{(m+1)!}$$

Applying Markov's inequality,

$$\sum_{m\geq 0}\mathbb{P}\left[\sup_{t\in[s,T]}|\xi_t^{q,m+1} - \xi_t^{q,m}| > 2^{-m}\right] \leq 2L^2(T+4)\sum_{m\geq 0}2^{2m}\tfrac{C^m T^{m+1}}{(m+1)!} < \infty.$$

Thus, by the Borel-Cantelli lemma,

$$\mathbb{P}\left[\sup_{t\in[s,T]}|\xi_t^{q,m+1} - \xi_t^{q,m}| > 2^{-m} \text{ infinitely often}\right] = 0,$$

which immediately implies

$$\xi_t^{q,k} = \xi_t^{q,0} + \sum_{m=0}^{k-1}(\xi_t^{q,m+1} - \xi_t^{q,m}) \to \xi_t^q \quad a.s.$$

uniformly in $t \in [s,T]$, for some limiting process ξ_t^q which is necessarily continuous almost surely and W_t-adapted. Moreover, we also have convergence in $\mathcal{L}^2(\Omega)$ uniformly in t. To see this, for each $k > l$ we observe that

$$\sup_{t\in[s,T]}\left(\mathbb{E}|\xi_t^k - \xi_t^l|^2\right)^{1/2} \leq \sup_{t\in[s,T]}\sum_{m=l}^{k-1}\left(\mathbb{E}|\xi_t^{q,m+1} - \xi_t^{q,m}|^2\right)^{1/2}$$
$$\leq \sum_{m=l}^{\infty}\sqrt{2L^2(T+4)\tfrac{C^m T^{m+1}}{(m+1)!}} \xrightarrow{l\to\infty} 0.$$

And hence $\xi_t^{q,k}$ converges uniformly in $\mathcal{L}^2(\Omega)$ to ξ_t^q as $k \to \infty$ (the limit is the same as the a.s. limit since a sub-sequence of it must converge a.s.). This immediately implies via the Lipschitz condition and Itô's isometry that

$$\mathbb{E}\left|\int_s^T B(\xi_t^{q,k}, t, \eta) - B(\xi_t^q, t, \eta)dt\right|^2 \leq T^2L^2\sup_{t\in[s,T]}\mathbb{E}\left|\xi_t^{q,k} - \xi_t^q\right|^2 \to 0,$$
$$\mathbb{E}\left|\int_s^T S(\xi_t^{q,k}, t, \eta) - S(\xi_t^q, t, \eta)dW_t\right|^2 \leq TL^2\sup_{t\in[s,T]}\left|\xi_t^{q,k} - \xi_t^q\right|^2 \to 0.$$

Thus, ξ_t^q satisfies (A.2).

We now show the estimate (A.3). From Eq. (A.2), we have by Itô's isometry,

$$\mathbb{E}|\xi_t^q|^2 \leq 3\mathbb{E}|\phi_t^q|^2 + 3\mathbb{E}\left|\int_s^T B(\xi_v^q, v, q)dv\right|^2 + 3\mathbb{E}\left|\int_s^t S(\xi_v^q, v, q)dW_v\right|^2$$
$$\leq 3C_\phi + 3T^2L^2\int_s^T \mathbb{E}(1 + |\xi_v^q|^2)dv + 3TL^2\int_s^T \mathbb{E}(1 + |\xi_v^q|^2)dv$$

Thus, by Gronwall's lemma, we have

$$\mathbb{E}|\xi_t^q|^2 \leq C(1 + C_\phi) \tag{A.11}$$

for some $C > 0$ depending only on T, L. Consequently, we have

$$\mathbb{E} \sup_{t\in[s,T]} |\xi_t^q|^2 \leq 3C_\phi + 3T^2L^2\mathbb{E}\int_s^T 1 + |\xi_s^q|^2 dt \tag{A.12}$$

$$+ 3\mathbb{E} \sup_{t\in[s,T]} \left|\int_s^t S(\xi_t^q, t, q)dW_t\right|^2. \tag{A.13}$$

Assumption A.1 (ii) and (A.11) implies the last stochastic integral is a martingale, and so by Doob's maximal inequality,

$$\left(\mathbb{E} \sup_{t\in[s,T]} \left|\int_s^t S(\xi_t, t, \eta)dW_t\right|\right)^2 \leq \int_s^T 4L^2(1 + \mathbb{E}|\xi_t|^2)dt. \tag{A.14}$$

Combining (A.12) and (A.14), we arrive at (A.3).

Finally, we show uniqueness. Suppose that ξ_t, ξ_t' are two solutions to (A.2). The same calculation as before shows that

$$\mathbb{E}|\xi_t - \xi_t'|^2 \leq 2L^2T(1+T)\int_s^t \mathbb{E}|\xi_v - \xi_v'|^2 dv.$$

and Gronwall's lemma implies

$$\mathbb{E}|\xi_t - \xi_t'|^2 \leq e^{2L^2T(1+T)}\mathbb{E}|\xi_s - \xi_s'|^2 = 0.$$

■

Theorem 19 *Let us assume the same conditions as in Thm. 18 and for each $q \in Q$, let ξ_t^q be the unique solution of* (A.2)*. Let $m \geq 1$ and suppose $\sup_{q\in Q}\mathbb{E}\sup_{t\in[s,T]}|\phi_t^q|^{2m} < \infty$. Then, there exists a constant $C > 0$ depending only on L, T, m such that*

$$\sup_{q\in Q}\mathbb{E} \sup_{t\in[s,T]} |\xi_t^q|^{2m} \leq C\left(1 + \sup_{q\in Q}\mathbb{E} \sup_{t\in[s,T]} |\phi_t^q|^{2m}\right).$$

Proof We have

$$|\xi_t^q|^{2m} \leq 3^{2m-1}|\phi_t^q|^{2m} + (3(t-s))^{2m-1}\int_s^t |B(\xi_v^q, v, q)|^{2m}dv$$
$$+ 3^{2m-1}\left|\int_s^t S(\xi_v^q, v, q)dW_v\right|^{2m}$$

Taking expectations, using Itô's isometry (inequality version) and Gronwall's inequality, we obtain

$$\mathbb{E}|\xi_t^q|^{2m} \leq 3^{2m-1}\sup_{q\in Q}\mathbb{E} \sup_{t\in[s,T]} |\phi_t^q|^{2m} + (3(t-s))^{2m-1}L^{2m}\int_s^t (1 + \mathbb{E}|\xi_v^q|^{2m})dv$$
$$+ 3^{2m-1}(m(2m-1))^m(t-s)^{m-1}L^{2m}\int_s^t (1 + \mathbb{E}|\xi_v^q|^{2m})dv$$
$$\leq C\left(1 + \sup_{q\in Q}\mathbb{E} \sup_{t\in[s,T]} |\phi_t^q|^{2m}\right) < \infty,$$

with C depending only on L, T, m. Next,

$$\mathbb{E} \sup_{t\in[s,T]} |\xi_t^q|^{2m} \leq 3^{2m-1}\sup_{q\in Q}\mathbb{E} \sup_{t\in[s,T]} |\phi_t^q|^{2m}$$
$$+ 3T^{2m-1}\mathbb{E}\int_s^T |B(\xi_v^q, v, q)|^{2m}dv$$
$$+ 3^{2m-1}\mathbb{E} \sup_{t\in[s,T]} \left|\int_s^t S(\xi_v^q, v, q)dW_v\right|^{2m}$$

Now, in the last term, the stochastic integral is a local martingale and so its absolute value is a submartingale, and hence the last term is bounded by

$$\mathbb{E} \sup_{t\in[s,T]} \left|\int_s^t S(\xi_v^q, v, q)dW_v\right|^{2m} \leq \left|\int_s^T S(\xi_v^q, v, q)dW_v\right|^{2m}$$
$$\leq C\int_s^T |S(\xi_v^q, v, q)|^{2m}dv.$$

Thus, using Thm. 19 and the linear growth condition, we conclude that

$$\sup_{q\in Q}\mathbb{E} \sup_{t\in[s,T]} |\xi_t^q|^{2m} \leq C\left(1 + \sup_{q\in Q}\mathbb{E} \sup_{t\in[s,T]} |\phi_t^q|^{2m}\right)$$

■

Finally, we examine some limiting behavior of solutions ξ_t^q as $q \to q^*$ for some $q^* \in Q$.

Theorem 20 *Let us assume the same conditions as in Thm. 18 and let $q^* \in Q$ be fixed. Suppose further that the following holds for any $t \in [s,T]$, $R > 0$ and $\epsilon > 0$:*

(i) $\lim_{q\to q^*}\mathbb{P}\left[\sup_{|x|\leq R}|B(x,t,q) - B(x,t,q^*)| > \epsilon\right] = 0$

(ii) $\lim_{q\to q^*}\mathbb{P}\left[\sup_{|x|\leq R}|S(x,t,q) - S(x,t,q^*)| > \epsilon\right] = 0$

(iii) $\lim_{q\to q^*}\sup_{t\in[s,T]}\mathbb{E}|\phi_t^q - \phi_t^{q^*}|^2 = 0$

Then, the solutions ξ_t^q of (A.2) *satisfy*

$$\lim_{q\to q^*}\sup_{t\in[s,T]}\mathbb{E}|\xi_t^q - \xi_t^{q^*}|^2 = 0,$$

i.e. $\xi_t^q \to \xi_t^{q^*}$ in $\mathcal{L}^2(\Omega)$ uniformly in $t \in [s,T]$.

Proof We have

$$\xi_t^q - \xi_t^{q^*} = \zeta_t^q + \int_s^t B(\xi_v^q, v, q) - B(\xi_v^{q^*}, v, q)dv$$
$$+ \int_s^t S(\xi_v^q, v, q) - S(\xi_v^{q^*}, v, q)dW_v,$$

where

$$\zeta_t^q := \phi_t^q - \phi_t^{q^*} + \int_s^t B(\xi_v^{q^*}, v, q) - B(\xi_v^{q^*}, v, q^*)dv + \int_s^t S(\xi_v^{q^*}, v, q) - S(\xi_v^{q^*}, v, q^*)dW_v.$$

Using the Lipschitz conditions,

$$\mathbb{E}|\xi_t^q - \xi_t^{q^*}|^2 \le 3\mathbb{E}|\zeta_t^q|^2 + 6L^2 \int_s^t \mathbb{E}|\xi_v^q - \xi_v^{q^*}|^2 dv,$$

which by Gronwall's lemma implies

$$\sup_{t\in[s,T]} \mathbb{E}|\xi_t^q - \xi_t^{q^*}|^2 \le 3e^{6L^2T} \sup_{t\in[s,T]} \mathbb{E}|\zeta_t^q|^2.$$

Thus, it remains to show that $\sup_{t\in[s,T]} \mathbb{E}|\zeta_t^q|^2 \to 0$ as $h \to 0$. Now,

$$\sup_{t\in[s,T]} \mathbb{E}|\zeta_t^q|^2 \le 3 \sup_{t\in[s,T]} \mathbb{E}|\phi_t^q - \phi_t^{q^*}|^2 + 3T \int_s^T \mathbb{E}|B(\xi_v^{q^*}, v, q) - B(\xi_v^{q^*}, v, q^*)|^2 dv + 3\int_s^T \mathbb{E}|S(\xi_v^{q^*}, v, q) - S(\xi_v^{q^*}, v, q^*)|^2 dv.$$

For each $v \in [s,T]$, the assumption (i) together with the a.s. continuity of B implies $B(\xi_v^{q^*}, v, q) \to B(\xi_v^{q^*}, v, q^*)$ in probability. Moreover, by Assumption A.1 (ii) the last integrand is bounded by $2L^2(1+\sup_{v\in[s,T]}|\xi_v^{q^*}|^2)$, which is integrable. By the dominated convergence theorem, the integral vanishes in the limit $h \to 0$. A similar calculation shows the last integral also vanishes in the same limit. Together with (iii), we arrive at our assertion. ■

Appendix B. Derivatives with respect to initial condition

Let us denote by $\{X_t^{x,s,q} : t \ge 0\}$ the stochastic process defined by the SDE

$$dX_t^{x,s,q} = b(X_t^{x,s,q}, q)dt + \sigma(X_t^{x,s,q}, q)dW_t, \quad t \in [s,T], \qquad X_s^{x,s,q} = x. \tag{B.1}$$

As in the previous section, $q \in Q$ where Q is a subset of a Euclidean space. Throughout this section, we assume the following:

Assumption B.1 *The (non-random) functions b, σ satisfy*

1. *Uniform linear growth condition*

$$|b(x,q)|^2 + |\sigma(x,q)|^2 \le L^2(1+|x|^2)$$

for all $x \in \mathbb{R}^d$, $q \in Q$.

2. *Uniform Lipschitz condition*

$$|b(x,q) - b(y,q)| + |\sigma(x,q) - \sigma(y,q)| \le L|x-y|$$

for all $x, y \in \mathbb{R}^d$, $q \in Q$.

With the above assumptions, by Thm. 18 the SDE (B.1) admits a unique solution. The focus of this section is to derive the SDEs that characterize the derivatives of $X_t^{x,s,q}$ with respect to x, the initial condition. In doing so, we will make use the results proved in Sec. A.

Definition 21 *Let $\Psi : \mathbb{R}^d \to \mathbb{R}$ and $\psi : \mathbb{R}^d \to \mathbb{R}^d$ be random functions and suppose for each $i = 1, \dots, d$,*

$$\lim_{h\to 0} \mathbb{E}\left|\tfrac{1}{h}[\Psi(x_{(1)}, \dots, x_{(i-1)}, x_{(i)} + h, x_{(i+1)}, \dots, x_{(d)}) - \psi_i(x_{(1)}, \dots, x_{(d)})] - \psi_{(i)}(x)\right|^2 = 0.$$

Then, we call ψ the derivative (in the $\mathcal{L}^2(\Omega)$ sense) of Ψ and write $\partial_{(i)}\Psi = \psi_{(i)}$, or $\nabla\Psi = \psi$. For multidimensional Ψ, we similarly define the derivative element-wise. Note that the derivative is almost surely unique, if it exists.

Lemma 22 *Let $s \in [0,T)$, $q \in Q$ and suppose that b and σ are continuously differentiable with respect to x. Then, $\nabla X_t^{x,s,q}$ exists and if we write $\xi_{(i,j),t}^{x,s,q} := \partial_{(j)} X_{(i),t}^{x,s,q}$, then it satisfies the linear random-coefficient stochastic differential equation*

$$\xi_{(i,j),t}^{x,s,q} = \delta_{(i,j)} + \int_s^t \xi_{(k,j),v}^{x,s,q} \partial_{(k)} b(X_v^{x,s,q}, v, q)_{(i)} dv + \int_s^t \xi_{(k,j),v}^{x,s,q} \partial_{(k)} \sigma(X_v^{x,s,q}, v, q)_{(i,l)} dW_{(l),v}, \tag{B.2}$$

where δ is the usual Kronecker delta. Moreover, we have

$$\sup_{q\in Q} \mathbb{E} \sup_{t\in[s,T]} |\xi_t^{x,s,q}|^{2m} < \infty$$

for all $m \ge 1$.

Proof Let j be fixed and h^j be a d-dimensional vector of 0's except the j^{th} coordinate where it is equal $h_{(j)}^j = h \ne 0$. Then, we have

$$\tfrac{1}{h}(X_{(i),t}^{x+h^j,s,q} - X_{(i),t}^{x,s,q}) = \delta_{(i,j)} + \tfrac{1}{h}\int_s^t b(X_v^{x+h^j,s,q}, q)_{(i)} - b(X_v^{x,s,q}, q)_{(i)} dv + \tfrac{1}{h}\int_s^t \sigma(X_v^{x+h^j,s,q}, q)_{(i,l)} - \sigma(X_v^{x,s,q}, q)_{(i,l)} dW_{(l),v}.$$

But,

$$\tfrac{1}{h}\int_s^t b(X_v^{x+h^j,(i),s}, q)_{(i)} - b(X_v^{x,s,q}, q)_{(i)} dv = \int_0^1 \int_s^t \tfrac{1}{h}(X_{(k),v}^{x+h^j,s,q} - X_{(k),v}^{x,s,q}) \partial_{(k)} b(\lambda X_v^{x+h^j,s,q} + (1-\lambda)X_v^{x,s,q})_{(i)} dv d\lambda,$$

and similarly,

$$\begin{aligned}&\frac{1}{h}\int_s^t \sigma(X_v^{x+h^j,s,q},q)_{(i,l)} - \sigma(X_v^{x,s,q},q)_{(i,l)}dW_{(l),v}\\ =&\int_0^1\int_s^t \frac{1}{h}(X_{(k),v}^{x+h^j,s,q} - X_{(k),v}^{x,s,q})\partial_{(k)}\sigma(\lambda X_v^{x+h^j,s,q} + (1-\lambda)X_v^{x,s,q})_{(i,l)}dW_{(l),v}d\lambda.\end{aligned}$$

Therefore, $\xi_t^{x,s,q,h} := \frac{1}{h}(X_t^{x+h^j,s,q} - X_t^{x,s,q})$ satisfies (A.2) with $Q\times[0,1]$ in place of Q, $\phi_{t,(i,j)}^{q,h} = \delta_{(i,j)}$ and

$$\begin{aligned}B(z,t,q,h)_{(i)} &= z_{(k)}\int_0^1\int_s^t \partial_{(k)}b(\lambda X_v^{x+h^j,s,q} + (1-\lambda)X_v^{x+h^j,s,q},q)_{(i)}dvd\lambda,\\ S(z,t,q,h)_{(i)} &= z_{(k)}\int_0^1\int_s^t \partial_{(k)}\sigma(\lambda X_v^{x+h^j,s,q} + (1-\lambda)X_v^{x+h^j,s,q},q)_{(i,l)}dW_{(l),v}d\lambda,\end{aligned}$$

if $h>0$. If $h<0$ we simply consider $-h$ on the left hand side instead and the proof is identical. Furthermore, the uniform Lipschitz conditions on b,σ implies bounded derivatives and so we may apply Thm. 18 to conclude that there is a process $\xi_t^{0,q}$ satisfying (A.2) with $h=0$, i.e. satisfies (B.2).

It remains to show $\xi_t^{q,h}\to\xi_t^{q,0}$ in $\mathcal{L}^2(\Omega)$ uniformly in $t\in[s,T]$, which amounts to checking conditions (i)-(iii) in Thm. 20, with $q^*=(q,0)$. The last condition (iii) is trivially satisfied. As for the first two, it is enough to show that $X_t^{x+h^j,s,q}\to X_t^{x,s,q}$ in $\mathcal{L}^2(\Omega)$ as $h\to 0$, uniformly in x, which follows from the straightforward estimate

$$\begin{aligned}\mathbb{E}|X_t^{x,s,q} - X_t^{x+h^j,s,q}|^2 \le& 3h^2 + C\int_s^t \mathbb{E}|X_v^{x,s,q} - X_v^{x+h^j,s,q}|^2dv\\ \le& C'h^2.\end{aligned}$$

Now, we may apply Thm. 20 to deduce the satisfaction of the SDE. Finally, the last moment estimate follows from Thm. 19. ∎

Let us now extend the above result to higher order derivatives. As before, we denote the order α partial derivative of Ψ in the $\mathcal{L}^2(\Omega)$ sense by

$$\partial_{(J)}^{\alpha}\Psi \equiv \partial_{(j_1,\dots,j_\alpha)}^{\alpha}\Psi$$

where J is an order α multi-index.

Lemma 23 *Suppose that $b,\sigma\in G^2$. Then, for each $i,j_1,j_2\in\{1,\dots,d\}$, the derivative $\xi_{(i,j_1,j_2),t}^{2,x,s,q} := \partial_{(j_1,j_2)}^2 X_{(i),t}^{x,s,q}$ exists and is the unique solution of the linear random-coefficient*

stochastic differential equation

$$\xi_{(i,j_1,j_2),t}^{2,x,s,q} = \int_s^t \partial_{(k_1,k_2)}^2 b(X_v^{x,s,q},q)_{(i)}\xi_{(k_1,j_1),v}^{1,x,s,q}\xi_{(k_2,j_2),v}^{1,x,s,q}dv \tag{B.3}$$

$$+\int_s^t \partial_{(k_1,k_2)}^2\sigma(X_v^{x,s,q},q)_{(i,l)}\xi_{(k_1,j_1),v}^{1,x,s,q}\xi_{(k_2,j_2),v}^{1,x,s,q}dW_{(l),v} \tag{B.4}$$

$$+\int_s^t \partial_{(k)}b(X_v^{x,s,q},q)_{(i)}\xi_{(k,j_1,j_2),v}^{2,x,s,q}dv \tag{B.5}$$

$$+\int_s^t \partial_{(k)}\sigma(X_v^{x,s,q},q)_{(i,l)}\xi_{(k,j_1,j_2),v}^{2,x,s,q}dW_{(l),v} \tag{B.6}$$

where $\xi_{(i,j),t}^{1,x,s,q} := \partial_{(j)}X_{(i),t}^{x,s,q}$ is the first derivative. Moreover, for each $m\ge 1$, we have $\mathbb{E}\sup_{t\in[s,T]}|\xi_t^{2,x,s,q}|^{2m}\in G$, i.e.

$$\sup_{q\in Q,s\in[0,T]}\mathbb{E}\sup_{t\in[s,T]}|\xi_t^{2,x,s,q}|^{2m}\le\kappa_1(1+|x|^{2\kappa_2}) \tag{B.7}$$

Proof Let us denote

$$\begin{aligned}\phi_{(i,j_1,j_2),t}^{x,s,q} =& \int_s^t \partial_{(k_1,k_2)}^2 b(X_v^{x,s,q},q)_{(i)}\xi_{(k_1,j_1),v}^{1,x,s,q}\xi_{(k_2,j_2),v}^{1,x,s,q}dv\\ &+\int_s^t \partial_{(k_1,k_2)}^2\sigma(X_v^{x,s,q},q)_{(i,l)}\xi_{(k_1,j_1),v}^{1,x,s,q}\xi_{(k_2,j_2),v}^{1,x,s,q}dW_{(l),v}.\end{aligned}$$

Note that by Lem. 22, $\mathbb{E}\sup_{t\in[s,T]}|\xi_t^{1,x,s,q}|^{2m}$ is finite for any $m\ge1$. Then, proceeding as in the proof of Lem. 22, we have

$$\begin{aligned}&\mathbb{E}\sup_{t\in[s,T]}|\phi_t^{x,s,q}|^2\\ \le& C\mathbb{E}\sup_{t\in[s,T]}\left(|\nabla^2 b(X_t^{x,s,q},q)|^2 + |\nabla^2\sigma(X_t^{x,s,q},q)|^2\right)|\xi_t^{1,x,s,q}|^4\\ \le& C\left[\mathbb{E}\sup_{t\in[s,T]}\left(|\nabla^2 b(X_t^{x,s,q},q)|^2 + |\nabla^2\sigma(X_t^{x,s,q},q)|^2\right)^2\right]^{1/2}\left[\mathbb{E}\sup_{t\in[s,T]}|\xi_t^{1,x,s,q}|^8\right]^{1/2}\end{aligned}$$

Here, C is independent of q and s. From the above, using the assumption that $b,\sigma\in G^2$, and the moment estimate in Thm. 19 on $X_t^{x,s,q}$, we conclude that

$$\sup_{q\in Q,s\in[0,T]}\mathbb{E}\sup_{t\in[s,T]}|\phi_t^{x,s,q}|^2\le\kappa_1(1+|x|^{2\kappa})$$

thus (B.3) admits a unique solution by Thm. 18, and the solution $\xi_t^{2,x,s,q}$ satisfies the same estimate. Moreover, the estimate above holds for any $2m$ power for $m\ge1$ by a similar calculation, which shows that $\mathbb{E}\sup_{t\in[s,T]}|\xi_t^{2,x,s,q}|^{2m}\in G$.

Finally, To show that $\xi_t^{2,x,s,q}$ is the second derivative of $X_t^{x,s,q}$ with respect to x, we proceed analogously as in the proof of 22, thanks to estimate (B.7) and polynomial growth

conditions, all the estimates required for interchanging the derivative and the integral signs are satisfied, so the equation for $\xi_t^{2,x,s,q}$ is obtained by formally differentiating under the integral sign with respect to x, which is precisely (B.3). ■

Lemma 24 *For each $\alpha \geq 1$, suppose that $b, \sigma \in G^{\alpha+1}$, then, the derivative $\nabla^{\alpha+1} X_t^{x,s,q}$ exists and is the unique a.s. continuous solution of the linear random-coefficient SDE*

$$\xi_{(i,J),t}^{\alpha+1,x,s,q} = \phi_{(i,J),t}^{x,s,q} + \int_s^t \partial_{(k)} b(X_v^{x,s,q}, q)_{(i)} \xi_{(k,J),t}^{\alpha+1,x,s,q} dv \tag{B.8}$$

$$+ \int_s^t \partial_{(k)} \sigma(X_v^{x,s,q}, q)_{(i,l)} \xi_{(k,J),t}^{\alpha+1,x,s,q} dW_{(l),v}, \tag{B.9}$$

where J is a multi-index of order $\alpha+1$ and $\phi_t^{x,s,q}$ is an a.s. continuous stochastic process satisfying $\mathbb{E}\sup_{t\in[s,T]} |\phi_t^{x,s,q}|^{2m} \in G$ for all $m \geq 1$. In fact, (B.8) *is obtained by formally differentiating* (B.2) *under the integral sign α times. Moreover, we have $\mathbb{E}\sup_{t\in[s,T]} |\xi_t^{\alpha+1,x,s,q}|^{2m} \in G$ for all $m \geq 1$.*

Proof The proof is identical to the $\alpha = 1$ case in Lem. 23. We omit writing out the whole proof here. ■

We now prove the following useful result, which imparts polynomial growth conditions onto expectations functionals.

Proposition 25 *Let $s \in [0,T]$ and $g \in G^{\alpha+1}$ for some $\alpha \geq 1$. For $t \in [s,T]$, define*

$$u(x,s,q,t) := \mathbb{E} g(X_t^{x,s,q})$$

Then, $u(\cdot, s, q, t) \in G^{\alpha+1}$ uniformly in s, q, t.

Proof Consider first the case $\alpha = 1$. We shall use the results in Lem. 22-24 to show that

$$\partial_{(i)} u(x,s,q,t) = \mathbb{E} \partial_{(k)} g(X_t^{x,s,q}) \partial_{(i)} X_{(k),t}^{x,s,q}$$

and that $\partial_{(i)} u(x,s,q,t) \in G$. Let h^j be defined as in the proof of 22, we have

$$\begin{aligned}
&\tfrac{u(x+h^j,s,q,t)-u(x,s,q,t)}{h} \\
=&\mathbb{E} \int_0^1 \tfrac{1}{h} \tfrac{d}{d\lambda} g(\lambda X_t^{x+h^j,s,q} + (1-\lambda) X_t^{x,s,q}) d\lambda \\
=&\mathbb{E} \int_0^1 \partial_{(k)} g(\lambda X_t^{x+h^j,s,q} + (1-\lambda) X_t^{x,s,q}) d\lambda \tfrac{X_{(k),t}^{x+h^j,s,q} - X_{(k),t}^{x,s,q}}{h}.
\end{aligned}$$

Now, $\frac{1}{h}(X_t^{x+h^j,s} - X_t^{x,s,q}) \to \partial_{(j)} X_t^{x,s,q}$ in $\mathcal{L}^2(\Omega)$. Moreover, set

$$I_h := \int_0^1 \partial_{(k)} g(\lambda X_t^{x+h^j,s,q} + (1-\lambda) X_t^{x,s,q}) d\lambda.$$

Since ∇g is continuous, $|I_h - \partial_{(k)} g(X_t^{x,s,q})|^2 \to 0$ in probability. Moreover,

$$\mathbb{E}|I_h - \partial_{(k)} g(X_t^{x,s,q})|^4 < \infty$$

by the assumption that $g \in G^1$. Thus, $\{|I_h - \partial_{(k)} g(X_t^{x,s,q})|^2 : h \in [0,1]\}$ is uniformly integrable and so $I_h \to \partial_{(k)} g(X_t^{x,s,q})$ in $\mathcal{L}^2(\Omega)$. We have thus arrived at

$$\partial_{(i)} u(x,s,q,t) = \mathbb{E} \partial_{(k)} g(X_t^{x,s,q}) \partial_{(i)} X_{(k),t}^{x,s,q},$$

and in particular,

$$|\nabla u(x,s,q,t)|^2 \leq \mathbb{E}|\nabla g(X_t^{x,s,q})|^2 \mathbb{E}|\nabla X_t^{x,s,q}|^2 \in G,$$

where we have used Thm. 19 and 22. The proof for higher order derivatives follow accordingly by the above procedure, using Lem. 24. ■

Appendix C. Auxiliary results for the proof of Thm. 3

Lemma 26 *Let $\alpha \geq 1$ and suppose b, σ satisfy Assumption B.1. Then, there exists a $K \in G$, independent of η and ϵ, such that*

$$\mathbb{E} \prod_{j=1}^{\alpha+1} \left| \widetilde{\Delta}_{(i_j)} \right| \leq K(x) \eta^{\alpha+1}.$$

where $i_j \in \{1, \ldots, d\}$ and $C > 0$ is independent of η.

Proof We have

$$\begin{aligned}
\mathbb{E}|\widetilde{\Delta}(x)|^{\alpha+1} \leq& 2^\alpha \mathbb{E} \left| \int_0^\eta b(X_s^{x,0}, \eta, \epsilon) ds \right|^{\alpha+1} + 2^\alpha \eta^{\frac{\alpha+1}{2}} \mathbb{E} \left| \int_0^\eta \sigma(X_s^{x,0}, \eta, \epsilon) dW_s \right|^{\alpha+1} \\
\leq& 2^\alpha \eta^\alpha \int_0^\eta \mathbb{E}|b(X_s^{x,0}, \eta, \epsilon)|^{\alpha+1} ds + 2^\alpha \eta^{\frac{\alpha+1}{2}} \left| \int_0^\eta \sigma(X_s^{x,0}, \eta, \epsilon) dW_s \right|^{\alpha+1}
\end{aligned}$$

Using Cauchy-Schwarz inequality, Itô's isometry, we get

$$\begin{aligned}
\mathbb{E} \left| \int_0^\eta \sigma(X_s^{x,0}, \eta, \epsilon) dW_s \right|^{\alpha+1} \leq& \left(\mathbb{E} \left| \int_0^\eta \sigma(X_s^{x,0}, \eta, \epsilon) dW_s \right|^{2\alpha+2} \right)^{1/2} \\
\leq& C \eta^{\alpha/2} \left(\int_0^\eta \mathbb{E}|\sigma(X_s^{x,0}, \eta, \epsilon)|^{2\alpha+2} ds \right)^{1/2}
\end{aligned}$$

where C depends only on α. Now, using the linear growth condition (B.1 (i)) and the moment estimates in Thm. 19, we obtain the result. ■

Lemma 27 *Suppose $u \in G^{(\alpha+1)}$ for some $\alpha \geq 1$. Let assumption (i) in Thm. 3 hold. Then, there exists some $K \in G$, independent of η, ϵ, such that*

$$\left|\mathbb{E}u(x_1^{x,0}) - \mathbb{E}u(\widetilde{X}_1^{x,0})\right| \leq K(x)(\eta\rho(\epsilon) + \eta^{\alpha+1})$$

Proof Using Taylor's theorem with the Lagrange form of the remainder, we have

$$\begin{aligned}
u(x_1^{x,0}) - u(\widetilde{X}_1^{x,0}) =& \sum_{s=1}^{\alpha} \tfrac{1}{s!} \sum_{i_1,\ldots,i_j=1}^{d} \prod_{j=1}^{s} [\Delta_{(i_j)}(x) - \widetilde{\Delta}_{(i_j)}(x)] \tfrac{\partial^s u}{\partial x_{(i_1)},\ldots x_{(i_j)}}(x) \\
&+ \tfrac{1}{(\alpha+1)!} \sum_{i_1,\ldots,i_j=1}^{d} \prod_{j=1}^{\alpha+1} [\Delta_{(i_j)}(x) - \widetilde{\Delta}_{(i_j)}(x)] \\
&\times \left[\tfrac{\partial^{(\alpha+1)} u}{\partial x_{(i_1)},\ldots x_{(i_j)}}(x + a\Delta(x)) - \tfrac{\partial^{(\alpha+1)} u}{\partial x_{(i_1)},\ldots x_{(i_j)}}(x + \widetilde{a}\widetilde{\Delta}(x)) \right],
\end{aligned}$$

where $a, \widetilde{a} \in [0,1]$. Taking expectations, using assumption (i) of Thm. 4.1 and Lem. 26, we get

$$|\mathbb{E}u(x_1^{x,0}) - \mathbb{E}u(\widetilde{X}_1^{x,0})| \leq K(x)(\eta\rho(\epsilon) + \eta^{\alpha+1}).$$

■

Appendix D. Auxiliary results for the proof of Thm. 9

Set in (4.2)

$$\begin{aligned}
b(x,\eta,\epsilon) &= b_0(x,\epsilon) + \eta b_1(x,\epsilon) \\
\sigma(x,\eta,\epsilon) &= \sigma_0(x,\epsilon),
\end{aligned}$$

We prove the following Itô-Taylor expansion.

Lemma 28 *Let $\psi : \mathbb{R}^d \to \mathbb{R}$ be a sufficiently smooth function and define the operators*

$$\begin{aligned}
A_{\epsilon,0}\psi(x) :=& b_0(x,\epsilon)_{(i)} \partial_{(i)} \psi(x) \\
A_{\epsilon,1}\psi(x) :=& b_1(x,\epsilon)_{(i)} \partial_{(i)} \psi(x) + \tfrac{1}{2} \sigma_0(x,\epsilon)_{(i,k)} \sigma_0(x,\epsilon)_{(j,k)} \partial^2_{(i,j)} \psi(x) \\
[\Lambda_{\epsilon,0} g(x)]_{(l)} :=& \sigma_0(x,\epsilon)_{(i,l)} \partial_{(i)} \psi(x), \qquad l = 1,\ldots,d.
\end{aligned}$$

Suppose further that $b_0, b_1, \sigma_0 \in G^3$. Then, we have

$$\mathbb{E}\psi(X_\eta^{x,0}) = \psi(x) + \eta A_{\epsilon,0}\psi(x) + \eta^2(\tfrac{1}{2}A^2_{\epsilon,0} + A_{\epsilon,1})\psi(x) + \mathcal{O}(\eta^3).$$

Proof Using Itô's formula, we have

$$\begin{aligned}
\psi(X_\eta^{x,0}) =& \psi(x) + \int_0^\eta A_{\epsilon,0}\psi(X_s^{x,0})ds + \eta \int_0^\eta A_{\epsilon,1}\psi(X_s^{x,0})ds \\
&+ \sqrt{\eta} \int_0^\eta \Lambda_{\epsilon,0}\psi(X_s^{x,0}) dW_s
\end{aligned}$$

By further application of the above formula to $A_{\epsilon,0}\psi$ and $A_{\epsilon,1}\psi$, we have

$$\begin{aligned}
\psi(X_\eta^{x,0}) =& \psi(x) + \eta A_{\epsilon,0}\psi(x) + \eta^2(\tfrac{1}{2}A^2_{\epsilon,0} + A_{\epsilon,1})\psi(x) \\
&+ \eta \int_0^\eta \int_0^s (A_{\epsilon,1}A_{\epsilon,0} + A_{\epsilon,0}A_{\epsilon,1})\psi(X_v^{x,0}) dv ds \\
&+ \int_0^\eta \int_0^s \int_0^v A^3_{\epsilon,0}\psi(X_r^{x,0}) dr dv ds \\
&+ \eta^2 \int_0^\eta \int_0^s A^2_{\epsilon,1}\psi(X_v^{x,0}) dv ds \\
&+ \eta \int_0^\eta \int_0^s \int_0^v A_{\epsilon,1}A^2_{\epsilon,0}\psi(X_r^{x,0}) dr dv ds \\
&+ \sqrt{\eta} \int_0^\eta \Lambda_{\epsilon,0}\psi(X_s^{x,0}) dW_s \\
&+ \sqrt{\eta} \int_0^\eta \int_0^s \Lambda_{\epsilon,0}A_{\epsilon,0}\psi(X_v^{x,0}) dW_v ds \\
&+ \sqrt{\eta} \int_0^\eta \int_0^s \int_0^v \Lambda_{\epsilon,0}A^2_{\epsilon,0}\psi(X_r^{x,0}) dW_r dv ds \\
&+ \eta^{3/2} \int_0^\eta \int_0^s \Lambda_{\epsilon,0}A_{\epsilon,1}\psi(X_v^{x,0}) dW_v ds
\end{aligned}$$

Taking expectations of the above, it remains to show that each of the terms in the integral either vanishes, or is $\mathcal{O}(\eta^3)$. This follows immediately from the assumption that $b_0, b_1, \sigma_0 \in G^3$ and $\psi \in G^4$. Indeed, observe that all the integrands have at most 3 derivatives in b_0, b_1, σ_0 and 4 derivatives in ψ, which by our assumptions all belong to G. Thus, the expectation of each integrand is bounded by $\kappa_1(1 + \sup_{t\in[0,\eta]} \mathbb{E}|X_t^{x,0}|^{2\kappa_2})$ for some κ_1, κ_2, which by Thm. 19 must be finite. Thus, the last 3 stochastic integrals are martingales and their expectation vanish, and the expectations of the other integrals are $\mathcal{O}(\eta^3)$ by the polynomial growth assumption and moment estimates in Thm. 19. ■

We also prove a general moment estimate for the generalized SGA iterations 4.1.

Lemma 29 *Let $\{x_k : k \geq 0\}$ be the generalized SGA iterations defined in 4.1. Suppose*

$$|h(x,\gamma,\eta)| \leq L_\gamma(1 + |x|)$$

for some random variable $L_\gamma > 0$ a.s. and $\mathbb{E}{L_\gamma}^m < \infty$ for all $m \geq 1$. Then, for fixed $T > 0$ and any $m \geq 1$, $\mathbb{E}|x_k|^m$ exists and is uniformly bounded in η and $k = 0,\ldots,N \equiv \lfloor T/\eta \rfloor$.

Proof For each $k \geq 0$, we have

$$|x_{k+1}|^m \leq |x_k|^l + \sum_{l=1}^{m} \binom{m}{l} |x_k|^{m-l} \eta^l |h(x_k,\gamma_k,\eta)|^{m-l}$$

Now, for $1 \leq l \leq m$,

$$\begin{aligned}
\mathbb{E}|x_k|^{m-l}|h(x_k,\gamma_k,\eta)|^l =& \mathbb{E}|x_k|^{m-l}\mathbb{E}(|h(x_k,\gamma_k,\eta)|^l|x_k) \\
\leq& \mathbb{E}(L_\gamma^l)\mathbb{E}|x_k|^{m-l}(1 + |x_k|^l) \\
\leq& 2\mathbb{E}(L_\gamma^l)(1 + \mathbb{E}|x_k|^m).
\end{aligned}$$

Hence, if we let $a_k := \mathbb{E}|x_k|^m$, we have

$$a_{k+1} \le (1 + C\eta)a_k + C'\eta$$

where $C, C' > 0$ are independent of η and k, which immediately implies

$$\begin{aligned} a_k \le & (a_0 + C'/C)(1 + C\eta)^k - C'/C \\ \le & (|x_0|^m + C'/C)e^{(T/\eta)\log(1+C\eta)} - C'/C \\ \le & (|x_0|^m + C'/C)e^{CT} - C'/C. \end{aligned}$$

■

We also need the following result concerning mollified functions.

Lemma 30 *Let $\epsilon \in (0, 1)$ and ψ be continuous with its weak derivative $D\psi$ belonging to G_w. Denote by $\psi^\epsilon = \nu^\epsilon * \psi$ the mollification of ψ. Then, there exists a $K \in G$ independent of ϵ such that*

$$|\psi^\epsilon(x) - \psi(x)| \le \epsilon K(x)$$

Proof We have for almost every x,

$$\begin{aligned} |\psi^\epsilon(x) - \psi(x)| \le & \int_{\mathcal{B}(0,\epsilon)} \nu^\epsilon(y)|\psi(x - y) - \psi(x)|dy \\ = & \int_{\mathcal{B}(0,\epsilon)} \nu^\epsilon(y) \left| \int_0^1 D\psi(x - \lambda y) \cdot y d\lambda \right| dy \\ \le & \epsilon \int_{\mathcal{B}(0,\epsilon)} \int_0^1 \nu^\epsilon(y)|D\psi(x - \lambda y)|d\lambda dy \\ \le & \epsilon \int_{\mathcal{B}(0,\epsilon)} \nu^\epsilon(y)\kappa_1[1 + \kappa_2(|x| + |y|)]dy \\ \le & \epsilon K(x). \end{aligned}$$

Since ψ is continuous, the above equality holds for all $x \in \mathbb{R}^d$. ■

Appendix E. Auxiliary results for computations in Sec. 5

Lemma 31 *Let A be a real square matrix such that all eigenvalues have positive real part. Then,*

(i) For each $\epsilon > 0$, there exists a constant $C_\epsilon > 0$ independent of t but depends on ϵ, such that

$$|e^{-tA}| \le C_\epsilon e^{-t(\min_i \Re\lambda_i(A) - \epsilon)}$$

(ii) If in addition A is diagonalizable, then there exists a constant $C > 0$ independent of t such that

$$|e^{-tA}| \le Ce^{-t \min_i \Re\lambda_i(A)}$$

Proof (i) We know that A is similar to a Jordan block matrix J so that $e^{-At} = Pe^{-Jt}P^{-1}$. Hence, $|e^{-At}| \le |P||P^{-1}||e^{-Jt}|$. For each Jordan block J_k, we have $J_k = \lambda_k I + N_k$ where N_k is nilpotent ($N_k^k = 0$). Hence,

$$\begin{aligned} e^{-J_k t} = & e^{-\lambda_k I t}e^{-N_k t} = e^{-\lambda_k t} \sum_{m=0}^{k-1} \frac{N_k^m}{m!}(-t)^m \\ = & e^{-(\lambda_k - \epsilon)t} \left[\sum_{m=0}^{k-1} \frac{N_k^m}{m!}(-t)^m e^{-\epsilon t} \right]. \end{aligned}$$

For each $\epsilon > 0$ the norm of the last term is uniformly bounded in t, and hence we obtain the result.

(ii) We denote the similarity transformation $A = PDP^{-1}$ where D is the diagonal matrix of eigenvalues of A. Defining $Q := P^\dagger P$ ($\dagger$ denotes conjugate transpose), we have

$$|e^{-tA}| = \mathrm{Tr}(e^{-tA^T}e^{-tA}) = \mathrm{Tr}(Q^{-1}e^{-tD^\dagger}Qe^{-tD})$$

■

References

Jing An, Jianfeng Lu, and Lexing Ying. Stochastic modified equations for the asynchronous stochastic gradient descent. *arXiv preprint arXiv:1805.08244*, 2018.

Francis Bach and Eric Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate $\mathcal{O}(1/n)$. In *Advances in Neural Information Processing Systems*, pages 773–781, 2013.

Michael Betancourt, Michael I Jordan, and Ashia C Wilson. On symplectic optimization. *arXiv preprint arXiv:1802.03653*, 2018.

Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of political economy*, 81(3):637–654, 1973.

Bart J Daly. The stability properties of a coupled pair of non-linear partial difference equations. *Mathematics of Computation*, 17(84):346–360, 1963.

Alexandre Défossez and Francis Bach. Averaged least-mean-squares: Bias-variance trade-offs and optimal sampling distributions. In *Artificial Intelligence and Statistics*, pages 205–213, 2015.

Rick Durrett. *Probability: theory and examples*. Cambridge university press, 2010.

Lawrence C Evans. Partial differential equations. 2010.

Yuanyuan Feng, Lei Li, and Jian-Guo Liu. A note on semi-groups of stochastic gradient descent and online principal component analysis. *arXiv preprint arXiv:1712.06509*, 2017.

CW Hirt. Heuristic stability theory for finite-difference equations. *Journal of Computational Physics*, 2(4):339–355, 1968.

Wenqing Hu, Chris Junchi Li, Lei Li, and Jian-Guo Liu. On the diffusion approximation of nonconvex stochastic gradient descent. *arXiv preprint arXiv:1705.07562*, 2017.

Peter E. Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, New York, corrected edition, June 2011.

Walid Krichene and Peter L Bartlett. Acceleration and averaging in stochastic descent dynamics. In *Advances in Neural Information Processing Systems*, pages 6796–6806, 2017.

Harold Kushner and G George Yin. *Stochastic approximation and recursive algorithms and applications*, volume 35. Springer Science & Business Media, 2003.

Harold J Kushner. Rates of convergence for sequential monte carlo optimization methods. *SIAM Journal on Control and Optimization*, 16(1):150–168, 1978.

Harold J Kushner and Adam Shwartz. An invariant measure approach to the convergence of stochastic approximations with state dependent noise. *SIAM Journal on Control and Optimization*, 22(1):13–27, 1984.

Harold Joseph Kushner and Dean S Clark. *Stochastic approximation methods for constrained and unconstrained systems*, volume 26. Springer Science & Business Media, 2012.

Qianxiao Li, Cheng Tai, and Weinan E. Dynamics of stochastic gradient algorithms. arxiv preprint. *arXiv preprint arXiv:1511.06251v1*, 2015.

Qianxiao Li, Cheng Tai, and Weinan E. Stochastic modified equations and adaptive stochastic gradient algorithms. In *International Conference on Machine Learning*, pages 2101–2110, 2017.

Lennart Ljung, Georg Ch Pflug, and Harro Walk. *Stochastic approximation and optimization of random systems*, volume 17. Birkhäuser, 2012.

Stephan Mandt, Matthew D Hoffman, and David M Blei. Continuous-time limit of stochastic gradient descent revisited. In *OPT workshop, NIPS*, 2015.

Stephan Mandt, Matthew D Hoffman, and David M Blei. A variational analysis of stochastic gradient algorithms. *arXiv preprint arXiv:1602.02666*, 2016.

Stephan Mandt, Matthew D Hoffman, and David M Blei. Stochastic gradient descent as approximate bayesian inference. *The Journal of Machine Learning Research*, 18(1):4873–4907, 2017.

Panayotis Mertikopoulos and Mathias Staudigl. Convergence to nash equilibrium in continuous games with noisy first-order feedback. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 5609–5614. IEEE, 2017.

Grigori N Milstein. Approximate integration of stochastic differential equations. *Theory of Probability & Its Applications*, 19(3):557–562, 1975.

Grigori N Milstein. Weak approximation of solutions of systems of stochastic differential equations. *Theory of Probability & Its Applications*, 30(4):750–766, 1986.

Eric Moulines and Francis Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems*, pages 451–459, 2011.

Deanna Needell, Rachel Ward, and Nati Srebro. Stochastic gradient descent, weighted sampling, and the randomized algorithm. In *Advances in Neural Information Processing Systems*, pages 1017–1025, 2014.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Yurii E Nesterov. A method for solving the convex programming problem with convergence rate $\mathcal{O}(1/k^2)$. In *Dokl. Akad. Nauk SSSR*, volume 269, pages 543–547, 1983.

WF Noh and MH Protter. Difference methods and the equations of hydrodynamics. Technical report, California. Univ., Livermore. Lawrence Radiation Lab., 1960.

Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

Maxim Raginsky and Jake Bouvrie. Continuous-time stochastic mirror descent on a network: Variance reduction, consensus, convergence. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 6793–6800. IEEE, 2012.

Shai Shalev-Shwartz and Tong Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *Mathematical Programming*, pages 1–41, 2014.

Ohad Shamir and Tong Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International Conference on Machine Learning*, pages 71–79, 2013.

Weijie Su, Stephen Boyd, and Emmanuel Candes. A differential equation for modeling Nesterov's accelerated gradient method: theory and insights. In *Advances in Neural Information Processing Systems*, pages 2510–2518, 2014.

RF Warming and BJ Hyett. The modified equation approach to the stability and accuracy analysis of finite-difference methods. *Journal of computational physics*, 14(2):159–179, 1974.

Andre Wibisono, Ashia C Wilson, and Michael I Jordan. A variational perspective on accelerated methods in optimization. *Proceedings of the National Academy of Sciences*, 113(47):E7351–E7358, 2016.

Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

Journal of Machine Learning Research 20 (2019) 1-57 Submitted 9/17; Revised 11/18; Published 1/19

Decontamination of Mutual Contamination Models

Julian Katz-Samuels JKATZSAM@UMICH.EDU
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109-2122 USA

Gilles Blanchard GILLES.BLANCHARD@MATH.UNI-POTSDAM.DE
Universität Potsdam, Institut für Mathematik
D-14476 Potsdam, Germany

Clayton Scott CLAYSCOT@UMICH.EDU
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109-2122 USA

Editor: Inderjit Dhillon

Abstract

Many machine learning problems can be characterized by *mutual contamination models*. In these problems, one observes several random samples from different convex combinations of a set of unknown base distributions and the goal is to infer these base distributions. This paper considers the general setting where the base distributions are defined on arbitrary probability spaces. We examine three popular machine learning problems that arise in this general setting: multiclass classification with label noise, demixing of mixed membership models, and classification with partial labels. In each case, we give sufficient conditions for identifiability and present algorithms for the infinite and finite sample settings, with associated performance guarantees.

Keywords: multiclass classification with label noise, classification with partial labels, mixed membership models, topic modeling, mutual contamination models

1. Introduction

In many machine learning problems, the learner observes several random samples from different mixtures of unknown base distributions, with unknown mixing weights, and the goal is to infer these base distributions. Examples include binary classification with label noise, multiclass classification with label noise, classification with partial labels, and topic modeling. The goal of this paper is to develop a unified framework and set of tools to study statistical properties of these problems in a very general setting.

To this end, we use the general framework of mutual contamination models (Blanchard and Scott, 2014). In a mutual contamination model, there are L distributions $P_1, \ldots, P_L$ called *base distributions*. The learner observes M random samples

$$X_1^i, \ldots, X_{n_i}^i \overset{i.i.d.}{\sim} \tilde{P}_i = \sum_{j=1}^{L} \pi_{i,j} P_j \tag{1}$$

where $i = 1, \ldots, M$, $\pi_{i,j} \geqslant 0$, and $\sum_j \pi_{i,j} = 1$. Here $\pi_{i,j}$ is the probability that an instance of the *contaminated distribution* $\tilde{P}_i$ is a realization of P_j. The $\pi_{i,j}$s and P_js are unknown and the $\tilde{P}_i$s are observed through data. In this work, we avoid parametric models and assume that the sample space is arbitrary. The model can be stated concisely as

$$\tilde{\boldsymbol{P}} = \boldsymbol{\Pi} \boldsymbol{P} \tag{2}$$

where $\boldsymbol{P} = (P_1, \ldots, P_L)^T$, $\tilde{\boldsymbol{P}} = (\tilde{P}_1, \ldots, \tilde{P}_M)^T$, and $\boldsymbol{\Pi} = \left(\pi_{i,j}\right)$ is an $M \times L$ matrix that we call the *mixing matrix*.

In this paper we study *decontamination* of mutual contamination models, which is the problem of recovering, or estimating, the base distributions $\boldsymbol{P}$ from the contaminated distributions $\tilde{\boldsymbol{P}}$ from which data are observed, without knowledge of the mixing matrix $\boldsymbol{\Pi}$. We focus our attention on three specific types of mutual contamination models, all of which describe modern problems in machine learning: multiclass classification with label noise, demixing of mixed membership models and classification with partial labels. We will demonstrate that these three decontamination problems can be addressed using a common set of concepts and techniques. Before elaborating our contributions in detail, we first offer an overview of the three specific mutual contamination models, and associated decontamination problems, that we study.

Multiclass Classification with Label Noise: In multiclass classification with label noise, $M = L$ and the goal is to recover $\boldsymbol{P}$. Each P_i represents the distribution of a class of examples. The learner observes training examples with noisy labels, that is, realizations from the $\tilde{P}_j$s. This problem arises in nuclear particle classification (Scott et al., 2013). When one draws samples of a specific particle, it is impossible to remove other types of particles from the background. Thus, each example is drawn from a mixture of the different types of particles.

Demixing of Mixed Membership Models: We consider the following decontamination problem in mixed membership models: given a sample from each $\tilde{P}_i$, recover $\boldsymbol{P}$ up to a permutation. We refer to this decontamination problem as *demixing of mixed membership models*. This problem arises in the task of automatically uncovering the thematic topics of a corpus of documents. Under the mixed membership model approach, the words of each document are thought of as being drawn from a document-specific mixture of topics. Specifically, documents correspond to the $\tilde{P}_i$s and the topics to the P_is. This approach is also referred to as topic modeling. As we discuss in the next section, our theory significantly generalizes existing topic modeling theoretical guarantees.

Classification with Partial Labels:[1] In classification with partial labels, each data point is labeled with a *partial label* $Y \subset \{1, \ldots, L\}$; the true label is in Y, but it is not known which label is the true one. In our setup, we view the ith random sample as having partial label $Y_i \coloneqq \{j : \boldsymbol{\pi}_{i,j} > 0\}$ and being distributed according to $\tilde{P}_i = \sum_{j \in Y_i} \pi_{i,j} P_j$. Thus, the learner observes training examples from the contaminated distributions $\tilde{\boldsymbol{P}}$ and the *partial label matrix* $\boldsymbol{\Pi}^+ = (\mathbf{1}_{\{\boldsymbol{\Pi}_{i,j} > 0\}})$, and the goal is to recover $\boldsymbol{P}$.

There are many applications of classification with partial labels because often abundant sources of data are naturally associated with information that can be interpreted as partial

1. Classification with partial labels has also been referred to as the "superset learning problem" or the "multiple label problem" (Liu and Dietterich, 2014).

©2019 Julian Katz-Samuels, Gilles Blanchard and Clayton Scott.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-576.html.

labels. For example, consider the task of face recognition. On the internet, there are many images with captions that indicate who is in the picture but do not indicate which face belongs to which person. A partial label could be formed by associating each face with the names of the individuals appearing in the same image (Cour et al., 2011).

Although our work emphasizes recovery of $\boldsymbol{P}$, it is also possible to think of decontamination of mutual contamination models as concerned with estimation of the mixing matrix $\boldsymbol{\Pi}$. This estimate of $\boldsymbol{\Pi}$ could be used as a plug-in for recently developed debiased losses for multiclass classification with label noise and classification with partial labels, which require knowledge of $\boldsymbol{\Pi}$ (Cid-Sueiro, 2012; Menon et al., 2015b; van Rooyen and Williamson, 2015; Patrini et al., 2017).

In this paper, we make the following contributions: (i) We give sufficient conditions on $\boldsymbol{P}$, $\boldsymbol{\Pi}$, and $\boldsymbol{\Pi}^+$ for identifiability of the three problems. (ii) We establish necessary conditions that in some cases match or are similar to the sufficient conditions. (iii) We introduce novel algorithms for the infinite and finite sample settings. These algorithms are nonparametric in the sense that they do not model P_i as a probability vector or other parametric model. Our algorithmic contributions show that while all three problems can be described in a unified way, the special structure of multiclass classification with label noise allows for a substantially simpler algorithm. (iv) We develop novel estimators for distributions obtained by iteratively applying the κ^* operator (defined below). (v) Finally, our framework gives rise to several novel geometric insights about each of these three problems and leverages concepts from affine geometry, multilinear algebra, and probability.

1.1. Notation

Let $\mathbb{Z}^+$ denote the positive integers. For $n \in \mathbb{Z}^+$, let $[n] = \{1, \ldots, n\}$. If $\boldsymbol{x} \in \mathbb{R}^K$, let x_i denote the ith entry of $\boldsymbol{x}$. If $\boldsymbol{x}_j \in \mathbb{R}^K$, then $x_{j,i}$ denotes the ith entry of $\boldsymbol{x}_j$. Let $\boldsymbol{e}_i$ denote the length L vector with 1 in the ith position and zeros elsewhere. Let $\boldsymbol{\pi}_i \in \Delta_L \subset \mathbb{R}^L$ be the transpose of the ith row of $\boldsymbol{\Pi}$ where Δ_L denotes the $(L-1)$-dimensional simplex, i.e., $\Delta_L = \{\boldsymbol{\mu} = (\mu_1, \ldots, \mu_L)^T \in \mathbb{R}^L \,|\, \sum_{i=1}^L \mu_i = 1 \text{ and } \forall i : \mu_i \geqslant 0\}$. Let Δ_L^M denote the product of M $(L-1)$-dimensional simplices, viewed as the space of $M \times L$ row-stochastic matrices. Let $\mathcal{P}$ denote the space of probability distributions on a measurable space $(\mathcal{X}, \mathcal{C})$. Let $\text{supp}(F)$ denote the support of a distribution F on a Borel space.

2. Related Work

Our work makes various contributions to the statistical understanding of multiclass classification with label noise, demixing of mixed membership models, and classification with partial labels. In the following subsections, we discuss how our results improve upon and relate to previous results in the literature.

2.1. Multiclass Classification with Label Noise

There has not been much work on classification with multiclass label noise. By contrast, label noise in the binary setting has received a fair amount of attention. For a review of work prior to 2013, see Scott et al. (2013). More recently, Natarajan et al. (2013) considered the binary label noise case where the label noise rates are known (in our case, the label noise

rates are unknown). van Rooyen and Williamson (2015) generalized the work of Natarajan et al. (2013) to the multiclass case, but again assumed that the mixing proportions are known. Recent work has proposed various algorithms for the binary setting where the label noise rates are unknown (Scott, 2015; van Rooyen et al., 2015; Menon et al., 2015a), but these algorithms have not been generalized to the multiclass case. Menon et al. (2016) consider the binary setting with instance-dependent corruption, but they assume that the class probability functions take the form of a single-index model, whereas we make no parametric assumptions on the P_is. Ghosh et al. (2017) consider multiclass label noise, but they make two restrictive assumptions: *(i)* in the infinite sample setting, they assume that there exists some function belonging to the chosen hypothesis class that attains 0 risk and *(ii)* in the finite sample setting, they assume that the label noise is symmetric, i.e., there exists a constant $c \in (0, 1)$ such that $\pi_{i,j} = \frac{c}{L-1}$ for all $i \neq j$. Patrini et al. (2017) also study the multiclass setting, but they assume that if their neural network has access to sufficiently many samples, it can perfectly model $\Pr(\tilde{Y} = k \,|\, \boldsymbol{x})$ where $\boldsymbol{x}$ is a given feature vector and $\tilde{Y}$ is a corrupted label. Unlike most previous work that aims to learn a classifier, our focus is on estimating the base distributions. Given these estimates, one could then design a classifier to optimize some performance measure. See, for example, Section 4.3 of our initial work on this subject (Blanchard and Scott, 2014).

Another approach for modeling random label noise, in addition to the mutual contamination model, is the label flipping model. Indeed, several of the above-cited papers adopt this setting. In this model, the label Y of a data point is flipped independently of its features X and

$$\mu_{l,k} \coloneqq \Pr(\tilde{Y} = k \,|\, Y = l)$$

gives the probability that a data point with true label $Y = l$ is corrupted to have an observed label $\tilde{Y} = k$. Under the assumption that Y and X are jointly distributed, the $\mu_{l,k}$s can be related to the $\pi_{i,j}s$ via Bayes' rule. We choose to study the mutual contamination model because we find it more convenient to study the question of identifiability.

In this paper, we extend Scott et al. (2013), which examined binary classification with label noise (the case where $M = L = 2$). The multiclass setting is significantly more challenging and, as such, requires novel sufficient conditions and mathematical notions. In particular, Scott et al. (2013) use the notion of *irreducibility* of distributions as one of their sufficient conditions.

Definition 1 *For distributions G and H, we say that G is* irreducible *with respect to H if it is not possible to write $G = \gamma H + (1-\gamma)F$ where F is a distribution and $0 < \gamma \leqslant 1$.*

Definition 2 *For distributions G and H, we say that G and H are* mutually irreducible *if G is irreducible with respect to H and H is irreducible with respect to G. We denote*

$$IR = \{(G, H) : G \text{ and } H \text{ are mutually irreducible distributions}\}.$$

Scott et al. (2013) require that P_1 and P_2 are mutually irreducible. To treat the multiclass setting, we introduce a generalization of mutual irreducibility, namely *joint irreducibility*.

The work presented below on multiclass label noise originally appeared in a conference paper (Blanchard and Scott, 2014). The purpose of the present paper is to demonstrate that

the framework developed in that paper can be extended to the other two decontamination problems, and to provide a unified presentation of the three settings. In particular, the joint irreducibility assumption plays a pivotal role in all three settings, as does the task of mixture proportion estimation. However, the decontamination procedures for the latter two problems are substantially more complicated than for multiclass classification with label noise.

2.2. Demixing Mixed Membership Models

Mixed membership models have become a powerful modeling tool for data where data points are associated with multiple distributions. Applications have appeared in a wide range of fields including image processing (Li and Perona, 2005), population genetics (Pritchard et al., 2000), document analysis (Blei et al., 2003), and surveys (Berkman et al., 1989). One particularly popular application is topic modeling on a corpus of documents, such as the articles published in the journal Science. Topic modeling is closely related to demixing of mixed membership models and our work may be viewed as studying topic modeling on general domains.

In topic modeling, the base distributions P_i correspond to topics and the contaminated distributions $\tilde{P}_i$ to documents, which are regarded as mixtures of topics. In most cases, the P_is are assumed to have a finite sample space. A variety of approaches have been proposed for topic modeling. The most common approach assumes a generative model for a corpus of documents and determines the maximum likelihood fit of the model given data. However, because maximum likelihood is NP-hard, these approaches must rely on heuristics that can get stuck in local minima (Arora et al., 2012).

Recently, a trend towards algorithms for topic modeling with provable guarantees has emerged. Most of these methods rely on the separability assumption **(SEP)** and its variants (Donoho and Stodden, 2003; Arora et al., 2012, 2013; Ding et al., 2013, 2014; Recht et al., 2012; Huang et al., 2016). According to **(SEP)**, $P_1, \ldots, P_L$ are distributions on a finite sample space and for every $i \in \{1, \ldots, L\}$, there exists a word $x \in \text{supp}(P_i)$ such that $x \notin \cup_{j \neq i} \text{supp}(P_j)$. Our requirement that $P_1, \ldots, P_L$ are jointly irreducible is a natural generalization of separability of $P_1, \ldots, P_L$, as we will argue below. Specifically, if $P_1, \ldots, P_L$ have discrete sample spaces, separability and joint irreducibility coincide; however, if $P_1, \ldots, P_L$ are continuous, under joint irreducibility, $P_1, \ldots, P_L$ can have the same support.

A key ingredient in these algorithms is to use the assumption of a finite sample space to view the distributions as probability vectors in Euclidean space; this leads to approaches based on non-negative matrix factorization (NMF), linear programs, and random projections (Donoho and Stodden, 2003; Arora et al., 2012, 2013; Ding et al., 2013, 2014; Recht et al., 2012; Huang et al., 2016). However, more general distributions cannot be viewed as finite-dimensional vectors. Therefore, topic modeling on general domains requires new techniques. Our work seeks to provide such techniques.

Topic modeling on general domains has several applications, including in high-energy physics (Metodiev and Thaler, 2018a,b). In collider data, quantum chromodynamics causes data samples to be a mixture of different types of particles, where the underlying fraction of the particle type is unknown. In this setting, it is of interest to recover information about

each of the particles. Recently, Metodiev and Thaler (2018b) applied the Demix algorithm, Algorithm 4 in the current paper, to this problem in the case $M = L = 2$.

Topic modeling on general domains is also relevant to recent empirical research on topic modeling with word embeddings, e.g., (Das et al., 2015; Li et al., 2016b,a; Xun et al., 2017; Zhao et al., 2018). Word embeddings map words to vectors in $\mathbb{R}^d$ in a semantically and syntactically meaningful way. Their use has been pivotal to the state-of-art performance of many algorithms in NLP (Luong et al., 2013). Several algorithms for topic modeling with word embeddings model the topics as multivariate Gaussian distributions in order to handle words that do not belong to the vocabulary of the training dataset (Das et al., 2015; Xun et al., 2017). Whereas current topic modeling algorithms with theoretical guarantees do not cover such a modeling approach, the generality of our algorithms does.

2.3. Classification with Partial Labels

Classification with partial labels has had two main formulations in previous work (Liu and Dietterich, 2014). In one formulation **(PL-1)**, instances from each class are drawn independently and the partial label for each instance is drawn independently from a set-valued distribution. In another formulation **(PL-2)**, training data are in the form of bags where each bag is a set of instances and the bag has a set of labels. Each instance belongs to a single class, and the set of labels associated with the bag is given by the union of the labels of the instances in the bag. Our framework is similar to **(PL-2)**, although it does not assume a joint distribution on the features of instances and the partial labels.

Most work takes an empirical risk minimization approach to classification with partial labels (Jin and Ghahramani, 2002; Nyugen and Caruana, 2008; Cour et al., 2011; Liu and Dietterich, 2012). Typically, these algorithms aim to pick a classifier that minimizes the *partial label error*: the probability that a given classifier assigns a label to a training instance that is not contained in the partial label associated with the training instance. By contrast, our approach is to estimate the base distributions. One could then use these estimates to train a classifier under some performance measure.

There has not been much theoretical work on developing a statistical understanding of classification with partial labels. Cid-Sueiro (2012) and van Rooyen and Williamson (2015) develop methods for classification with partial labels that require knowledge of the mixing proportions, e.g., the probability that a label is in a partial label, given the true label. In this work, we make the more realistic assumption that the mixing proportions are unknown.

Liu and Dietterich (2014) consider the question of learnability where the mixing proportions are unknown. They consider two main sufficient conditions for learnability of a partial label problem. First, they require that for every label $l \in [L]$, the probability that l occurs with any particular distinct label l' is less than 1. Our condition on the partial label (described in the next Section) is considerably weaker. For example, it permits the case where there are two labels $l \neq l'$ such that whenever l occurs in a partial label, l' also occurs.

The second sufficient condition of Liu and Dietterich (2014) is based on the class distributions, partial label distributions *and* the hypothesis class of choice. It requires that every hypothesis that attains zero partial label error also attains zero true error. While this condition may be useful for the selection of a suitable hypothesis class for an ERM

approach, it is important to develop interpretable sufficient conditions that only depend on the characteristics of a partial label problem. Our work provides such conditions.

We also note that Liu and Dietterich (2014) consider the realizable case, that is, the case where the supports of $P_1, \ldots, P_L$ do not overlap. By contrast, we make the significantly weaker assumption that $P_1, \ldots, P_L$ are jointly irreducible, which allows $P_1, \ldots, P_L$ to have the same support. Thus, our work addresses the agnostic case in classification with partial labels.

3. Sufficient Conditions for Identifiability

We can think of each problem as requiring a specific factorization of $\tilde{\boldsymbol{P}}$ in terms of $\boldsymbol{P}$ and $\boldsymbol{\Pi}$. We say $\tilde{\boldsymbol{P}}$ is *factorizable* if there exists $(\boldsymbol{\Pi}, \boldsymbol{P}) \in \Delta_L^M \times \mathcal{P}^L$ such that $\tilde{\boldsymbol{P}} = \boldsymbol{\Pi}\boldsymbol{P}$; we call $(\boldsymbol{\Pi}, \boldsymbol{P})$ a *factorization* of $\tilde{\boldsymbol{P}}$. Multiclass classification with label noise requires a specific ordering of the elements of $\boldsymbol{P}$; classification with partial labels requires that $\boldsymbol{\Pi}$ is consistent with $\boldsymbol{\Pi}^+$ and a specific ordering of the elements of $\boldsymbol{P}$.

A factorization is not guaranteed to exist. For example, there is no factorization in the case where $M = 3$, $L = 2$, and $\tilde{P}_1, \tilde{P}_2, \tilde{P}_3$ are linearly independent. When a factorization exists, in general it is not unique. For instance, consider the case where $L = M$, $(\boldsymbol{\Pi}, \boldsymbol{P})$ solves (2), and $\boldsymbol{\Pi}$ is not a permutation matrix. Then, another solution is $\tilde{\boldsymbol{P}} = \boldsymbol{I}\tilde{\boldsymbol{P}}$. Furthermore, there are infinitely many solutions in the following general case.

Proposition 1 *Suppose that $\tilde{\boldsymbol{P}}$ has at least two distinct $\tilde{P}_j$s and has a factorization $(\boldsymbol{\Pi}, \boldsymbol{P})$. If there is some $\tilde{P}_i$ in the interior of* $\mathrm{conv}(P_1, \ldots, P_L)$*, then there are infinitely many distinct non-trivial factorizations of $\tilde{\boldsymbol{P}}$.*

Proof Without loss of generality, suppose that $i = 1$ and $\tilde{P}_1 \neq \tilde{P}_2$. Then, since $\tilde{P}_1$ is in the interior of $\mathrm{conv}(P_1, \ldots, P_L)$, there is some $\delta > 0$ such that for any $\alpha \in (1, 1+\delta)$, $Q_\alpha = \alpha\tilde{P}_1 + (1-\alpha)\tilde{P}_2$ is a distribution. Then, $\mathrm{conv}(\tilde{P}_1, \ldots, \tilde{P}_L) \subseteq \mathrm{conv}(Q_\alpha, \tilde{P}_2, \ldots, \tilde{P}_L)$ and, consequently, there is some $\boldsymbol{\Pi}' \in \Delta_L^L$ such that $(\boldsymbol{\Pi}', (Q_\alpha, \tilde{P}_2, \ldots, \tilde{P}_L)^T)$ solves (2). Clearly, by varying α, there are infinitely many solutions to (2). ■

Identifiability of each problem is equivalent to the existence of a unique factorization for that problem. Therefore, to establish identifiability for the three problems, we must impose conditions on $(\boldsymbol{\Pi}, \boldsymbol{P})$ and $\boldsymbol{\Pi}^+$. To this end, we use the notion of joint irreducibility of distributions.

Definition 3 *The distributions $\{P_i\}_{1 \leqslant i \leqslant L}$ are* jointly irreducible *iff the following equivalent conditions hold*

(a) For all $I \subset [L]$ such that $1 \leqslant |I| < L$, and ϵ_i such that $\epsilon_i \geqslant 0$ and $\sum_{i \in I} \epsilon_i = \sum_{i \notin I} \epsilon_i = 1$,

$$(\sum_{i \in I} \epsilon_i P_i, \sum_{i \notin I} \epsilon_i P_i) \in IR.$$

(b) $\sum_{i=1}^L \gamma_i P_i$ is a distribution implies that $\gamma_i \geqslant 0 \, \forall i$.

Conditions *(a)* and *(b)*, whose equivalence was established by Blanchard and Scott (2014), give two ways to think about joint irreducibility. Condition *(a)* says that every convex combination of a subset of the P_is is irreducible (see Section 2.1) with respect to every convex combination of the other P_is. Condition *(b)* says that if a distribution is in the span of $P_1, \ldots, P_L$, it is in their convex hull. Joint irreducibility holds when each P_i has a region of positive probability that does not belong to the support of any of the other P_is; thus, separability (see Section 2.2) of the P_is entails joint irreducibility of $P_1, \ldots, P_L$. However, the converse is not true: the P_is can have the same support and still be jointly irreducible (e.g., P_is Gaussian with a common variance and distinct means (Scott et al., 2013)).

For all three problems, we assume that

(A) $P_1, \ldots, P_L$ are jointly irreducible.

Henceforth, unless we say otherwise, $P_1, \ldots, P_L$ are assumed to be jointly irreducible. In Appendix G, we provide experiments on real-world datasets that suggest that this assumption is reasonable.

We make different assumptions on $\boldsymbol{\Pi}$ for each of the three problems. For multiclass classification with label noise, we assume that

(B1) $\boldsymbol{\Pi}$ is invertible and $\boldsymbol{\Pi}^{-1}$ is a matrix with strictly positive diagonal entries and non-positive off-diagonal entries.

According to Lemma 1 below, this assumption essentially says that the problem has low noise in the sense that for each i, $\tilde{P}_i$ mostly comes from P_i. In particular, each P_i can be recovered by subtracting small multiples of $\tilde{P}_j$, $j \neq i$ from $\tilde{P}_i$. For example, consider the following case where $\boldsymbol{\Pi}$ satisfies **(B1)**. Suppose that there is a "common background noise" $\boldsymbol{c} \in \Delta_L$ that appears in different proportions in each of the distributions; formally, we have $\boldsymbol{\pi}_i = \gamma_i \boldsymbol{c} + (1-\gamma_i)\boldsymbol{e}_i$ with $\gamma_i \in [0, 1)$. In other words, we shift each of the vertices $\boldsymbol{e}_i$ towards a common point $\boldsymbol{c}$ (see panel (iii) of Figure 1). See Blanchard and Scott (2014) for a proof that this setup satisfies **(B1)**. In the binary case where $M = L = 2$, **(B1)** is equivalent to the simple condition that $\pi_{1,1} + \pi_{2,2} < 1$. This assumption roughly says that in expectation the majority of labels are correct. In Section 4.3, we present Lemma 1, which gives a geometric interpretation of **(B1)**.

For the demixing problem, we assume that

(B2) $\boldsymbol{\Pi}$ has full column rank.

We note that **(B2)** is considerably weaker than **(B1)**, e.g., it allows $M > L$. Of course, it is natural to demand a weaker sufficient condition for demixing the mixed membership problem than muliticlass classification with label noise because the goal of the former problem is to recover any permutation of $\boldsymbol{P}$ while the goal of the latter is to recover $\boldsymbol{P}$ exactly. Nevertheless, the identifiability analysis to establish **(B2)** as a sufficient condition is also significantly more involved than the analysis of **(B1)**.

For classification with partial labels, we assume that

(B3) $\boldsymbol{\Pi}$ has full column rank and the columns of $\boldsymbol{\Pi}^+$ are unique.

The assumption that the columns of $\boldsymbol{\Pi}^+$ are unique says that there are no two classes that always appear together in the partial labels. In Appendix C, we argue that several of the above conditions are also necessary, or are not much stronger than what is necessary.

4. Algorithms for the Population Case

In this section, to establish that the above conditions are indeed sufficient for identifiability, we give a population case analysis of the three problems. The results on multiclass classification with label noise appeared in a conference paper (Blanchard and Scott, 2014); we refer the reader to that paper for the proofs.

4.1. Background

This paper relies on the following quantity from Blanchard et al. (2010).

Definition 4 *Given probability distributions F_0, F_1, define*

$$\kappa^*(F_0 \,|\, F_1) = \max\{\kappa \in [0,1] | \, \exists \text{ a distribution } G \text{ s.t. } F_0 = (1-\kappa)G + \kappa F_1\}.$$

The following Proposition from Blanchard et al. (2010) establishes some useful properties of κ^*.

Proposition 2 *Given probability distributions F_0, F_1 on a measurable space $(\mathcal{X}, \mathcal{C})$, if $F_0 \neq F_1$, then $\kappa^*(F_0 \,|\, F_1) < 1$ and the above maximum is attained for a unique distribution G (which we refer to as the* residue of F_0 wrt. F_1*). Furthermore, the following equivalent characterization holds:*

$$\kappa^*(F_0 \,|\, F_1) = \inf_{C \in \mathcal{C}, F_1(C) > 0} \frac{F_0(C)}{F_1(C)}.$$

Note that $\kappa^*(F_0 \,|\, F_1) = 0$ iff F_0 is irreducible wrt F_1. $\kappa^*(F_0 \,|\, F_1)$ can be thought of as the maximum possible proportion of F_1 in F_0. We can think of $1 - \kappa^*(F_0 \,|\, F_1)$ as a statistical distance since it is non-negative and equal to zero if and only if $F_0 = F_1$. We refer to κ^* as the two-sample κ^* operator. To obtain the residue of F_0 wrt F_1, one computes Residue$(F_0 \,|\, F_1)$ (see Algorithm 1); this is well-defined under Proposition 2 when $F_0 \neq F_1$.

In order to gain intuition about κ^*, we briefly discuss how it can be used to recover $\mathbf{\Pi}^{-1}$ in the case $L = 2$. Under conditions discussed above (Scott et al., 2013), it holds that

$$\tilde{P}_1 = (1-\kappa_1)P_1 + \kappa_1 \tilde{P}_2, \text{ and}$$
$$\tilde{P}_2 = (1-\kappa_2)P_2 + \kappa_2 \tilde{P}_1.$$

and $\kappa_1 = \kappa^*(\tilde{P}_1 \,|\, \tilde{P}_2)$ and $\kappa_2 = \kappa^*(\tilde{P}_2 \,|\, \tilde{P}_1)$. By rearranging this system of equations, we can write

$$\boldsymbol{P} = \mathbf{\Pi}^{-1}\tilde{\boldsymbol{P}} = \begin{pmatrix} \frac{1}{1-\kappa_1} & -\frac{\kappa_1}{1-\kappa_1} \\ -\frac{\kappa_2}{1-\kappa_2} & \frac{1}{1-\kappa_2} \end{pmatrix} \tilde{\boldsymbol{P}}.$$

Next, we turn to the multi-sample generalization of κ^*, which we call the multi-sample κ^* operator.

Definition 5 *Given distributions $F_0, \ldots, F_K$, define*

$$\kappa^*(F_0 \,|\, F_1, \ldots, F_K) = \max_{\boldsymbol{\mu} \in \Delta_K} \kappa^*(F_0 \,|\, \sum_{i=1}^K \mu_i F_i)$$
$$= \max\Big(\sum_{i=1}^K \nu_i : \nu_i \geqslant 0, \sum_{i=1}^K \nu_i \leqslant 1, \exists \text{ distribution } G \text{ s.t. } F_0 = (1 - \sum_{i=1}^K \nu_i)G + \sum_{i=1}^K \nu_i F_i\Big). \quad (3)$$

Algorithm 1 Residue$(F_0 \,|\, F_1)$

1: $\kappa \longleftarrow \kappa^*(F_0 \,|\, F_1)$
2: **return** $\frac{F_0 - \kappa F_1}{1-\kappa}$

Algorithm 2 MultiResidue$(F_0 \,|\, \{F_1, \ldots, F_K\})$

1: $(\nu_1, \ldots, \nu_K)^T \longleftarrow (\nu'_1, \ldots, \nu'_K)^T$ achieving the maximum in $\kappa^*(F_0 \,|\, F_1, \ldots, F_K)$
2: **return** $\frac{F_0 - \sum_{i=1}^K \nu_i F_i}{1 - \sum_{i=1}^K \nu_i}$

Blanchard and Scott (2014) establish the equivalence in line (3), as well as Lemma 15, which shows that the outer maximum is always attained at some $\boldsymbol{\mu} \in \Delta_K$, i.e., κ^* is well-defined. Although there is always a G achieving the max, it is not necessarily unique. Any G attaining the maximum is called a *maximizer* of $\kappa^*(F_0 \,|\, F_1, \ldots, F_K)$. The algorithm MultiResidue$(F_0 \,|\, \{F_1, \ldots, F_K\})$ returns one of these G (see Algorithm 2). If G is unique, we call G the *multi-sample residue of F_0 with respect to $\{F_1, \ldots, F_K\}$*. Under our proposed sufficient conditions, certain residues are shown to exist, and our decontamination methods compute such residues via Algorithm 2. In Section 4.3, we discuss Lemma 1, which establishes useful conditions under which a multi-sample residue exists and is equal to one of the vertices of Δ_L.

In general, one cannot express the multi-sample version of κ^* in terms of the two-sample version. However, it is possible in some special cases. For example, if one had access to feasible $\nu_1, \ldots, \nu_K$ that attain the optimum in (3), then it holds that $\kappa^*(F_0 \,|\, F_1, \ldots, F_K) = \kappa^*(F_0 \,|\, \frac{\sum_{i=1}^K \nu_i F_i}{\sum_{i=1}^K \nu_i})$. Further, it is possible to replace the multi-sample κ^* with several calls of the two-sample κ^* when $K = L-1$, $F_i = P_i$ for all $i \neq 0$ and $F_0 = \sum_{i=1}^L \alpha_i P_i$ where $\sum_i \alpha_i = 1$ and $\forall i \ \alpha_i > 0$ (see Lemmas 6 and 13).

We remark that in previous work that assumes P_i are probability vectors, distributions are compared using l_p distances. By contrast, in our setting of general probability spaces, we use κ^* to compare different distributions.

4.2. Mixture Proportions

Recall that we assume that $P_1, \ldots, P_L$ are jointly irreducible. If $\boldsymbol{\eta} \in \mathbb{R}^L$ and $Q = \boldsymbol{\eta}^T \boldsymbol{P}$, we say that $\boldsymbol{\eta}$ is the *mixture proportion* of Q. Since by Lemma 16, joint irreducibility of $P_1, \ldots, P_L$ implies linear independence of $P_1, \ldots, P_L$, mixture proportions are well-defined, i.e., the mixture proportions are unique.

An important feature of our decontamination strategy is recovering various mixture proportions in the simplex Δ_L. To make this precise, we introduce the following definitions. If $i \in [L]$, we say that conv$(\{\boldsymbol{e}_j : j \neq i\})$ is a *face* of the simplex Δ_L; if $A \subset [L]$ and $|A| = k$, we also say that conv$(\{\boldsymbol{e}_j : j \in A\})$ is a *k-face* of Δ_L. If $\boldsymbol{\eta} \in \mathbb{R}^L$, Q is a distribution, and $Q = \boldsymbol{\eta}^T \boldsymbol{P}$, we say that $\mathcal{N}(Q) = \mathcal{N}(\boldsymbol{\eta}) = \{j : \eta_j > 0\}$ is the *support set* of $\boldsymbol{\eta}$ or the support set of Q. Note that by joint irreducibity, $\mathcal{N}(\boldsymbol{\eta})$ consists of the indices of all the nonzero entries in the mixture proportion $\boldsymbol{\eta}$. Finally, for $\boldsymbol{\eta}_i \in \Delta_L$, and $Q_i = \boldsymbol{\eta}_i^T \boldsymbol{P}$ for $i = 1, 2$, we

Algorithm 3 Multiclass($\tilde{P}_1, \ldots, \tilde{P}_L$)

1: **for** $i = 1, \ldots, L$ **do**
2: $\quad Q_i \longleftarrow \text{MultiResidue}(\tilde{P}_i \,|\, \{\tilde{P}_j : j \neq i\})$
3: **end for**
4: **return** $(Q_1, \ldots, Q_L)^T$

say that the distributions Q_1 and Q_2 (or the mixture proportions $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$) are *on the same face* of the simplex Δ_L if there exists $j \in [L]$ such that $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2 \in \text{conv}(\{\boldsymbol{e}_k : k \neq j\})$.

The heart of our approach is that under joint irreducibility, one can interchange distributions $Q_1, \ldots, Q_K$ and their mixture proportions $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_K$, as indicated by the following Proposition. We note that it is valid to to apply the κ^* operator to $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_K$ since they can be viewed as discrete probability distributions over $[L]$.

Proposition 3 *Let $Q_i = \boldsymbol{\eta}_i^T \boldsymbol{P}$ for $i \in [L]$ and $\boldsymbol{\eta}_i \in \Delta_L$. Suppose $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_L$ are linearly independent and $P_1, \ldots, P_L$ are jointly irreducible. Then,*

1. *for any $i \in [L]$ and $A \subseteq [L] \backslash \{i\}$, $\kappa^*(Q_i \,|\, \{Q_j : j \in A\}) = \kappa^*(\boldsymbol{\eta}_i \,|\, \{\boldsymbol{\eta}_j : j \in A\}) < 1$,*
2. *for any $i \in [L]$ and $A \subseteq [L] \backslash \{i\}$, a maximizer of $\kappa^*(Q_i \,|\, \{Q_j : j \in A\})$ exists, and*
3. *$\boldsymbol{\gamma} \in \Delta_L$ is a maximizer to $\kappa^*(\boldsymbol{\eta}_i \,|\, \{\boldsymbol{\eta}_j : j \in A\})$ if and only if $G = \boldsymbol{\gamma}^T \boldsymbol{P}$ is a maximizer to $\kappa^*(Q_i \,|\, \{Q_j : j \in A\})$.*

In words, this proposition says that the optimization problem given by $\kappa^*(Q_i \,|\, \{Q_j : j \in A\})$ is equivalent to the optimization problem given by $\kappa^*(\boldsymbol{\eta}_i \,|\, \{\boldsymbol{\eta}_j : j \in A\})$. *Thus, joint irreducibility of $P_1, \ldots, P_L$ and linear independence of the mixture proportions enable a reduction of each of the three problems to a geometric problem where the goal is to recover the vertices of a simplex by applying κ^* to points (i.e., the mixture proportions) in the simplex.* This makes the figures below valid for general distributions (see Figures 1, 2, 3, and 4).

4.3. Multiclass Classification with Label Noise

Our algorithm for multiclass classification with label noise is by far the simplest of the three. It simply computes a maximizer of $\kappa^*(\tilde{P}_i \,|\, \{\tilde{P}_j : j \neq i\})$ for every $i \in [L]$.

Theorem 1 *Let $P_1, \ldots, P_L$ be jointly irreducible and $\boldsymbol{\Pi}$ satisfy **(B1)**. Then, Multiclass($\tilde{P}_1, \ldots, \tilde{P}_L$) returns $\boldsymbol{Q} \in \mathcal{P}^L$ such that $\boldsymbol{Q} = \boldsymbol{P}$.*

The proof of this result has two main ideas. First, it applies the one-to-one correspondence established in Proposition 3 between the maximizers of $\kappa^*(\tilde{P}_i \,|\, \{\tilde{P}_j : j \neq i\})$ and the maximizers of $\kappa^*(\boldsymbol{\pi}_i \,|\, \{\boldsymbol{\pi}_j : j \neq i\})$.

Second, the proof shows that $\kappa^*(\boldsymbol{\pi}_i \,|\, \{\boldsymbol{\pi}_j : j \neq i\})$ is well-behaved in the sense that the residue of $\boldsymbol{\pi}_i$ wrt $\{\boldsymbol{\pi}_j : j \neq i\}$ is $\boldsymbol{e}_i$. The key idea is encapsulated in the following Lemma from Blanchard and Scott (2014).

Lemma 1 (Blanchard and Scott, 2014) *The following conditions on $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_L$ are equivalent:*

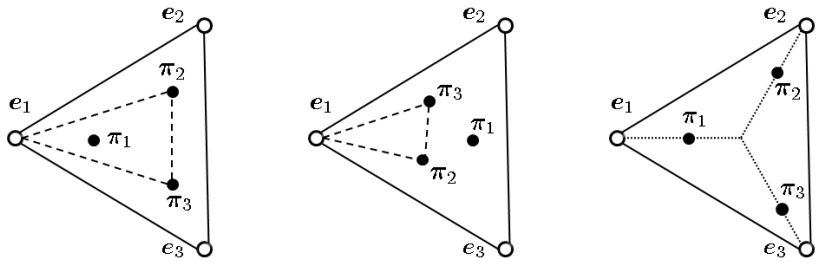


Figure 1: Illustration of the **(B1)** when there are $L = 3$ classes where $\boldsymbol{e}_i$ denotes the ith unit vector. Panel (i): Low noise, $\boldsymbol{\Pi}$ recoverable. Each $\boldsymbol{\pi}_l$ can be written as a convex combination of $\boldsymbol{e}_l$ and the other two $\boldsymbol{\pi}_j$ (with a positive weight on $\boldsymbol{e}_l$), depicted here for $l = 1$. Panel (ii): High noise, $\boldsymbol{\Pi}$ not recoverable. Panel (iii): The setting of "common background noise" described in the text.

1. *For each i, the residue of $\boldsymbol{\pi}_i$ with respect to $\{\boldsymbol{\pi}_j, j \neq i\}$ is $\boldsymbol{e}_i$.*
2. *For every i there exists a decomposition $\boldsymbol{\pi}_i = \kappa_i \boldsymbol{e}_i + (1 - \kappa_i)\boldsymbol{\pi}_i'$ where $\kappa_i > 0$ and $\boldsymbol{\pi}_i'$ is a convex combination of $\boldsymbol{\pi}_j$ for $j \neq i$.*
3. *$\boldsymbol{\Pi}$ is invertible and $\boldsymbol{\Pi}^{-1}$ is a matrix with strictly positive diagonal entries and non-positive off-diagonal entries.*

This lemma establishes that under **(B1)**, for each i, the residue of $\boldsymbol{\pi}_i$ with respect to $\{\boldsymbol{\pi}_j, j \neq i\}$ is $\boldsymbol{e}_i$. The main step in the proof of this Lemma is establishing that *3* implies *1*. The argument identifies the residue of $\boldsymbol{\pi}_i$ with respect to $\{\boldsymbol{\pi}_j\}_{j \neq i}$ by reformulating the linear program in $\kappa^*(\boldsymbol{\pi}_i \,|\, \{\boldsymbol{\pi}_j\}_{j \neq i})$ such that the objective is to maximize $\boldsymbol{e}_i^t \boldsymbol{\Pi}^{-1} \gamma$ subject to some appropriately defined constraint. By the structure of $\boldsymbol{\Pi}^{-1}$ assumed in **(B1)**, it follows that the $\gamma \in \Delta_L$ that maximizes this objective is $\boldsymbol{e}_i$, and it can further be shown that this maximizer satisfies the other constraints.

Thus, combining the above two ideas yields the result. In addition, Lemma 1 provides geometric intuition as to when **(B1)** is satisfied through condition *2*. Figure 1 illustrates the case $L = 3$. See Panel (i) for an example where condition (b) is satisfied and Panel (ii) for an example where (b) is not satisfied.

4.4. Demixing Mixed Membership Models

In this section, we assume that $M = L$; we consider the nonsquare case in the appendix. For certain simple cases of mixture proportions, a straightforward resampling strategy can be used to reduce the problem of demixing mixed membership models to multiclass classi-

fication with label noise. For example, suppose that there are $L = 3$ classes and

$$\mathbf{\Pi} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} \end{pmatrix}. \tag{4}$$

Inspection shows that the inverse of $\mathbf{\Pi}$ does not satisfy the condition in **(B1)** and, therefore, one cannot simply apply Algorithm 3. A simple procedure to circumvent this issue is to resample from the contaminated distributions to obtain the following distributions:

$$\tilde{Q}_1 = \frac{1}{2}\tilde{P}_1 + \frac{1}{2}\tilde{P}_2,\ \tilde{Q}_2 = \frac{1}{2}\tilde{P}_1 + \frac{1}{2}\tilde{P}_3,\ \text{and}\ \tilde{Q}_3 = \frac{1}{2}\tilde{P}_2 + \frac{1}{2}\tilde{P}_3.$$

Then, it can be shown that the resulting mixing matrix

$$\tilde{\mathbf{\Pi}} = \begin{pmatrix} \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \end{pmatrix}$$

associated with the $\tilde{Q}_i$s satisfies the conditions of Lemma 1 so that $\text{Multiclass}(\tilde{Q}_1, \tilde{Q}_2, \tilde{Q}_3)$ gives the desired solution. However, this approach breaks down for most possible mixing matrices. Thus, the challenge is to develop an algorithm that works for a large class of mixture proportions and does not rely on knowledge of the mixture proportions. To meet this challenge, we propose the Demix algorithm.

The Demix algorithm is recursive. Let $S_1, \ldots, S_K$ denote K contaminated distributions. In the base case, the algorithm takes as its input two contaminated distributions S_1 and S_2. It returns $\text{Residue}(S_1 \,|\, S_2)$ and $\text{Residue}(S_2 \,|\, S_1)$, which are a permutation of the two base distributions (see Figure 2). When $K > 2$, Demix uses a subroutine FindFace (see Algorithm 5) to find $K-1$ distributions $R_2, \ldots, R_K$ on the same $(K-1)$-face. FindFace iteratively generates candidates for distributions on the same $(K-1)$-face, which it tests using FaceTest (see Algorithm 6). $\text{FaceTest}(S_1, \ldots, S_{K-1})$ determines whether a set of distributions $S_1, \ldots, S_{K-1}$ are on the interior of the same face by using the two-sample κ^* operator; equivalently, it tests whether there exists a pair of distributions S_i and S_j such that S_i is irreducible with respect to S_j. Once Demix finds $K-1$ distributions $R_2, \ldots, R_K$ on the same $(K-1)$-face, it recursively applies Demix to $R_2, \ldots, R_K$ to obtain distributions $Q_1, \ldots, Q_{K-1}$ that are a permutation of $K-1$ of the base distributions. Subsequently, the algorithm computes a maximizer Q_K of $\kappa^*(\frac{1}{K}\sum_{i=1}^K S_i \,|\, Q_1, \ldots, Q_{K-1})$. Since $Q_1, \ldots, Q_{K-1}$ are a permutation of $K-1$ of the base distributions, the maximizer Q_K is guaranteed to be unique and to be the remaining base distribution (see Figure 3 for an execution of the algorithm).

A number of remarks are in order regarding the Demix algorithm. First, although we compute the residue of $\frac{1}{n}S_i + \frac{n-1}{n}Q$ wrt S_1 for each $i \neq 1$, there is nothing special about the distribution S_1. We could replace S_1 with any S_j where $j \in [K]$, provided that we adjust the rest of the algorithm accordingly. Second, we can replace the sequence $\{\frac{n-1}{n}\}_{n=1}^{\infty}$ with any sequence $\alpha_n \nearrow 1$. Finally, we could replace line 7 with the following sequence of steps: for $i = 1, \ldots, K-1$, compute $Q_K \longleftarrow \text{Residue}(Q_K \,|\, Q_i)$ (see Lemma 13). Then, the algorithm would only use the two-sample κ^* operator. We use such an algorithm in the finite-sample setting.

Algorithm 4 $\text{Demix}(S_1, \ldots, S_K)$

Input: $S_1, \ldots, S_K$ are distributions

1: **if** $K = 2$ **then**
2: **return** $(\text{Residue}(S_1 \,|\, S_2), \text{Residue}(S_2 \,|\, S_1))^T$
3: **else**
4: $(R_2, \ldots, R_K)^T \longleftarrow \text{FindFace}(S_1, \ldots, S_K)$
5: $(Q_1, \ldots, Q_{K-1})^T \longleftarrow \text{Demix}(R_2, \ldots, R_K)$
6: $Q_K \longleftarrow \frac{1}{K}\sum_{i=1}^K S_i$
7: $Q_K \longleftarrow \text{MultiResidue}(Q_K \,|\, Q_1, \ldots, Q_{K-1})$
8: **return** $(Q_1, \ldots, Q_K)^T$
9: **end if**

Algorithm 5 $\text{FindFace}(S_1, \ldots, S_K)$

Input: $S_1, \ldots, S_K$ are distributions

1: $Q \longleftarrow$ uniformly distributed element in $\text{conv}(S_2, \ldots, S_K)$
2: **for** $n = 1, 2, \ldots$ **do**
3: Set $R_i \longleftarrow \text{Residue}(\frac{1}{n}S_i + \frac{n-1}{n}Q \,|\, S_1)$ for all $i \in \{2, \ldots, K\}$
4: **if** $\text{FaceTest}(R_2, \ldots, R_K)$ **then**
5: **return** $(R_2, \ldots, R_K)^T$
6: **end if**
7: **end for**

We also remark that a simplified version of Demix solves the demixing mixed membership models problem if we assume **(B1)** from the multiclass label noise setting. In that case, finding $L-1$ distributions on the same face can be accomplished by simply computing $Q_i \longleftarrow \text{MultiResidue}(\tilde{P}_i \,|\, \{\tilde{P}_j\}_{j \neq i})$ for $i = 2, \ldots, L$. Indeed, then, each Q_i is equal to P_i and P_1 can be obtained by computing $\text{MultiResidue}(\tilde{P}_1 \,|\, \{Q_j\}_{j=2,\ldots,L})$.

We establish the following theorem.

Theorem 2 *Let $P_1, \ldots, P_L$ be jointly irreducible and $\mathbf{\Pi}$ have full column rank. Then, with probability 1, Demix$(\tilde{\boldsymbol{P}})$ returns a permutation of $\boldsymbol{P}$.*

We briefly sketch three key aspects of the proof. First, in the FindFace subroutine, sampling Q uniformly at random from $\text{conv}(S_2, \ldots, S_K)$ ensures that w.p. 1 $\text{Residue}(Q \,|\, S_1)$ is on the interior of a face of the simplex. Then, conditional on this event, we show that by a continuity property of $\text{Residue}(\cdot \,|\, S_1)$ there is a large enough n such that $R_2, \ldots, R_K$ are on the same face of the simplex Δ_{K-1} (see panels (c) and (d) of Figure 3). Second, Proposition 8 in Appendix D establishes that the subroutine $\text{FaceTest}(R_2, \ldots, R_K)$ returns 1 if and only if $R_2, \ldots, R_K$ are on the same face of the simplex. Combining the above two observations implies that eventually $\text{FindFace}(S_1, \ldots, S_K)$ terminates at which point $\{R_k\}_{k \in [K]\setminus\{1\}} \subset \{P_k\}_{k \in [K]\setminus\{l\}}$ for some $l \in [K]$. The final key observation is that $\{R_k\}_{k \in [K]\setminus\{1\}}$ and $\{P_k\}_{k \in [K]\setminus\{l\}}$ form an instance of the demixing problem that satisfies the sufficient conditions **(A)** and **(B2)** (see Figure 2). Therefore, this instance can be solved recursively.

Algorithm 6 FaceTest$(S_1, \ldots, S_K)$

1: Set $\boldsymbol{Z}_{i,j} \coloneqq \mathbf{1}\{\kappa^*(S_i \,|\, S_j) > 0\}$ for all i and j
2: **if** $\boldsymbol{Z}$ has a zero off-diagonal entry **then**
3: **return** 0
4: **else**
5: **return** 1
6: **end if**

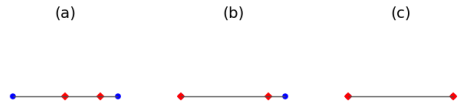


Figure 2: In (a), we consider a demixing problem where there are two classes and $M = L$ (the base case of Algorithm 4). The diamonds represent the mixture proportions of $\tilde{P}_1$ and $\tilde{P}_2$. The circles represent the base distributions. In (b), the residue of a contaminated distribution wrt the other contaminated distribution is computed (line 3), yielding a base distribution. In (c), the residue is computed again switching the roles of the contaminated distributions (line 4); this yields the remaining base distribution.

4.5. Classification with Partial Labels

As in the case of demixing mixed membership models, a simple resampling strategy works in certain nice settings of classification with partial labels. For example, consider an instance of classification with partial labels with the mixing matrix from equation (4). The resampling procedure that yields $\tilde{Q}_1, \tilde{Q}_2, \tilde{Q}_3$ (described in Section 4.4) also works here. Nevertheless, as in demixing mixed membership models, this approach does not meet our goal of an algorithm that solves a broad class of mixing matrices and partial labels.

Indeed, we observe that the partial labels do not provide enough information for choosing the resampling weights. Consider another instance of the problem with the same partial label matrix:

$$\boldsymbol{\Pi} = \begin{pmatrix} \frac{1}{10} & \frac{9}{10} & 0 \\ \frac{9}{10} & 0 & \frac{1}{10} \\ 0 & \frac{1}{10} & \frac{9}{10} \end{pmatrix}. \tag{5}$$

Applying the resampling approach to (5) can be shown to fail by observing that the inverse of the resampled mixing matrix does not satisfy condition *3* of Lemma 1. Thus, although the

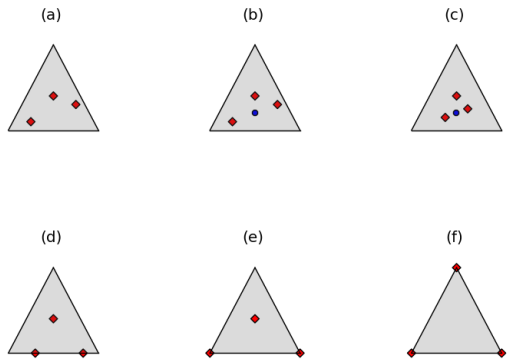


Figure 3: In (a), we consider a demixing problem where there are three classes and $M = L$. The diamonds represent the mixture proportions of $\tilde{P}_1$, $\tilde{P}_2$ and $\tilde{P}_3$. In (b), the blue circle is a random distribution chosen in the convex hull of two of the distributions (line 7). In (c), two of the distributions are resampled so that their residues wrt the other distribution are on the same face of the simplex (lines 12-15). In (d), these particular residues are computed (lines 12-15). In (e), two of the distributions are demixed (lines 3-5). In (f), the residue of the final distribution wrt the final two demixed distributions is computed to obtain the final demixing (line 18-21).

Algorithm 7 PartialLabel$(\boldsymbol{\Pi}^+, (\tilde{P}_1, \ldots, \tilde{P}_M)^T)$

1: **for** $i = 1, \ldots, L$ **do**
2: $Q_i \longleftarrow$ uniformly random distribution in $\text{conv}(\tilde{P}_1, \ldots, \tilde{P}_M)$
3: **end for**
4: **for** $k = 2, 3, \ldots$ **do**
5: $(W_1, \ldots, W_L)^T \longleftarrow$ GenerateCandidates$(k, (Q_1, \ldots, Q_L)^T)$
6: (FoundVertices, $\boldsymbol{C}$) $\longleftarrow$ VertexTest$(\boldsymbol{\Pi}^+, \tilde{P}_1, \ldots, \tilde{P}_M, W_1, \ldots, W_L)$
7: **if** FoundVertices **then**
8: **return** $\boldsymbol{C}(W_1, \ldots, W_L)^T$
9: **end if**
10: **end for**

problem instances (4) and (5) have the same partial label matrix, the resampling procedure only works for one of these.

Next, we turn to presenting an algorithm that solves classification with partial labels for a wide class of mixing matrices and partial labels. We propose the PartialLabel algorithm

Algorithm 8 GenerateCandidates($k, (Q_1, \ldots, Q_L)^T$)

1: Set $W_i \longleftarrow Q_i$ for all $i \in [K]$
2: **for** $i = 1, \ldots, L$ **do**
3: $\quad \bar{Q}_i \longleftarrow \frac{1}{L-1}[\sum_{j>i} Q_j + \sum_{j<i} W_j]$
4: $\quad W_i \longleftarrow \text{MultiResidue}(\frac{1}{k}Q_i + (1-\frac{1}{k})\bar{Q}_i \,|\, \{Q_j\}_{j>i} \cup \{W_j\}_{j<i})$
5: **end for**
6: **return** $(W_1, \ldots, W_L)^T$

Algorithm 9 VertexTest($\boldsymbol{\Pi}^+, (\tilde{P}_1, \ldots, \tilde{P}_M)^T, (Q_1, \ldots, Q_L)^T$)

1: Form the matrix $Y_{i,j} \coloneqq \mathbf{1}\{\kappa^*(Q_i \,|\, Q_j) > 0\}$
2: **if** $\boldsymbol{Y}$ has a non-zero off-diagonal entry **then**
3: $\quad$ **return** $(0, \mathbf{0})$
4: **end if**
5: Form the matrix $Z_{i,j} \coloneqq \mathbf{1}\{\kappa^*(\tilde{P}_i \,|\, Q_j) > 0\}$
6: Use any algorithm that finds a permutation matrix $\boldsymbol{C}$ such that $\boldsymbol{ZC} = \boldsymbol{\Pi}^+$ (if it exists)
7: **if** such a permutation matrix $\boldsymbol{C}$ exists **then**
8: $\quad$ **return** $(1, \boldsymbol{C}^T)$
9: **else**
10: $\quad$ **return** $(0, \mathbf{0})$
11: **end if**

(see Algorithm 7). PartialLabel proceeds by iteratively creating sets of candidate distributions $\boldsymbol{W} \coloneqq (W_1, \ldots, W_L)^T$ via the subroutine CreateCandidates (see Algorithm 8). Given each $\boldsymbol{W}$, it runs an algorithm VertexTest (see Algorithm 9) that uses $\tilde{\boldsymbol{P}}$ and the partial label matrix $\boldsymbol{\Pi}^+$ to determine whether $\boldsymbol{W}$ is a permutation of the base distributions $\boldsymbol{P}$. If $\boldsymbol{W}$ is a permutation of $\boldsymbol{P}$, VertexTest constructs the corresponding permutation matrix for relating these distributions. If not, it reports failure and the PartialLabel algorithm increments k and finds another set of candidate distributions.

The VertexTest algorithm proceeds as follows on a vector of candidate distributions $\boldsymbol{Q} \coloneqq (Q_1, \ldots, Q_L)^T$. First, it determines whether there are two distinct distributions Q_i, Q_j such that Q_i is not irreducible wrt Q_j, in which case $\boldsymbol{Q}$ cannot be a permutation of $\boldsymbol{P}$. If there is such a pair, it reports failure. Otherwise, it forms the matrix $Z_{i,j} \coloneqq \mathbf{1}\{\kappa^*(\tilde{P}_i \,|\, Q_j) > 0\}$ and uses any algorithm that finds a permutation $\boldsymbol{C}$ (if it exists) of the columns of $\boldsymbol{Z}$ to match the columns of $\boldsymbol{\Pi}^+$. If such a permutation $\boldsymbol{C}$ exists, it returns $\boldsymbol{C}^T$ and, as we show in Lemma 7, $\boldsymbol{C}^T\boldsymbol{Q} = \boldsymbol{P}$; otherwise, VertexTest reports failure.

We remark that finding the permutation in line 6 of Algorithm 9 is not NP-hard. One algorithm (but most likely not the most efficient) proceeds as follows: define a total ordering on the columns of binary matrices. Sort the columns of $\boldsymbol{Z}$ and $\boldsymbol{\Pi}^+$ according to this total ordering. Check whether the resulting matrices are equal.

The following theorem gives our identification result for classification with partial labels.

Theorem 3 *Suppose that $P_1, \ldots, P_L$ are jointly irreducible, $\boldsymbol{\Pi}$ has full column rank, and the columns of $\boldsymbol{\Pi}^+$ are unique. Then, PartialLabel($\boldsymbol{\Pi}^+, (\tilde{\boldsymbol{P}})^T$) returns $\boldsymbol{R} \in \mathcal{P}^L$ such that $\boldsymbol{R} = \boldsymbol{P}$.*

There are two key ideas to the proof of Theorem 3. First, the randomization in line 2 of Algorithm 7 ensures through a linear independence argument that with probability 1, the operation MultiResidue($\frac{1}{k}Q_i + (1-\frac{1}{k})\bar{Q}_i \,|\, \{Q_j\}_{j>i} \cup \{W_j\}_{j<i}$) in line 4 of Algorithm 8 is well-defined. Second, in the GenerateCandidates algorithm, let $Q_j = \boldsymbol{\tau}_j^T\boldsymbol{P}$ and $W_j = \boldsymbol{\gamma}_j^T P$. We make the simple observation that the affine hyperplane given by $\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_{i-1}, \boldsymbol{\tau}_{i+1}, \ldots, \boldsymbol{\tau}_L$ bisects Δ_L such that $\boldsymbol{\tau}_i$ and a nonempty subset of $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_L\} \backslash \{\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_{i-1}\}$ are in the same halfspace. Using this observation, we show that for large enough k, W_i is one of the base distributions and is distinct from all W_j with $j < i$.

The VertexTest algorithm connects the demixing problem and classification with partial labels by showing that any algorithm that solves the demixing problem can be used as a subroutine to solve classification with partial labels. For example, consider the following algorithm for classification with partial labels. First, use the Demix algorithm to obtain a permutation $\boldsymbol{Q}$ of the base distributions $\boldsymbol{P}$. Second, use VertexTest to find the permutation matrix relating $\boldsymbol{Q}$ and $\boldsymbol{P}$. This alternate algorithm is the basis of our finite sample algorithm for classification with partial labels (see Section 5.3 for a more thorough discussion).

5. Estimators for the Finite Sample Setting

In this section, we develop the estimation theory to treat the three problems in the finite sample setting. Let $\mathcal{X} = \mathbb{R}^d$ be equipped with the standard Borel σ-algebra $\mathcal{C}$ and $\tilde{P}_1, \ldots, \tilde{P}_L$ be probability distributions on this space. Suppose that we observe for $i = 1, \ldots, L$,

$$X_1^i, \ldots, X_{n_i}^i \overset{i.i.d.}{\sim} \tilde{P}_i.$$

Let $\mathcal{E}$ be any Vapnik-Chervonenkis (VC) class with VC-dimension $V < \infty$, containing the set of all open balls, all open rectangles, or some other collection of sets that generates the Borel σ-algebra $\mathcal{C}$. For example, $\mathcal{E}$ could be the set of all open balls wrt the Euclidean distance, in which case $V = d + 1$. Define $\epsilon_i(\delta_i) \equiv 3\sqrt{\frac{V\log(n_i+1) - \log(\delta_i/2)}{n_i}}$ for $i = 1, \ldots, L$. Our estimators are based on the VC inequality (Devroye et al., 1996). This inequality says that for each $i \in [L]$, and $\delta > 0$, the following holds with probability at least $1 - \delta$:

$$\sup_{E \in \mathcal{E}} |\tilde{P}_i(E) - \tilde{P}_i^\dagger(E)| \leqslant \epsilon_i(\delta)$$

where the *empirical distribution* is given by $\tilde{P}_i^\dagger(E) = \frac{1}{n_i}\sum_{j=1}^{n_i} \mathbf{1}_{\{X_j^i \in E\}}$.

5.1. Multiclass Classification with Label Noise

Let $F_0^\dagger, \ldots, F_M^\dagger$ denote the empirical distributions based on i.i.d. random samples from respective distributions $F_0, \ldots, F_M$. We introduce the following estimator of the multi-sample κ^*:

$$\widehat{\kappa}(F_0^\dagger \,|\, F_1^\dagger, \ldots, F_M^\dagger) = \max_{\boldsymbol{\mu} \in \Delta_M} \inf_{E \in \mathcal{E}} \frac{F_0^\dagger(E) + \epsilon_0(\frac{1}{n_0})}{(\sum_{i=1}^M \mu_i F_i^\dagger(E) - \sum_i \mu_i \epsilon_i(\frac{1}{n_i}))_+} \tag{6}$$

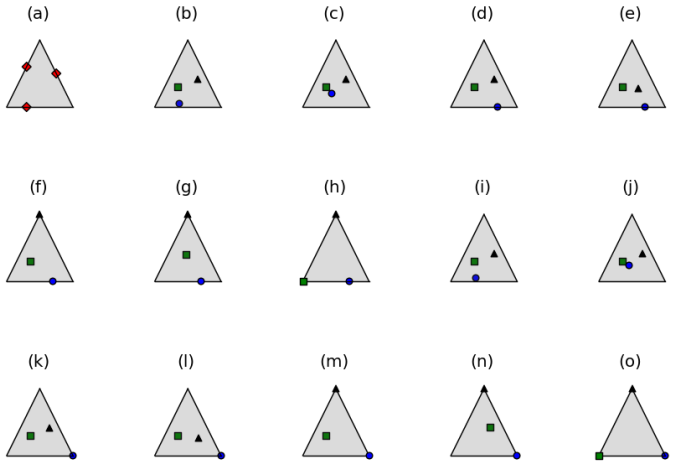


Figure 4: (a) depicts an instance of the partial label problem where there are $L = 3$ classes, $M = 3$ partial labels, and each partial label only contains two of the classes. In (a), the red diamonds represent the mixture proportions of the distributions $\tilde{P}_1, \tilde{P}_2, \tilde{P}_3$. In (b), three distributions Q_1, Q_2, Q_3 are sampled uniformly randomly from the convex hull of $\tilde{P}_1, \tilde{P}_2, \tilde{P}_3$; the blue circle, black triangle, and green square represent their mixture proportions. Figures (c)-(h) show how the algorithm generates a set of candidate distributions $(W_1^{(2)}, W_2^{(2)}, W_3^{(2)})^T$ with $k = 2$. In (h), PartialLabel runs VertexTest on $(W_1^{(2)}, W_2^{(2)}, W_3^{(2)})^T$ and determines that $(W_1^{(2)}, W_2^{(2)}, W_3^{(2)})^T$ is not a permutation of $(P_1, P_2, P_3)^T$. In (i)-(o), PartialLabel begins again with Q_1, Q_2, Q_3 and executes the same series of steps with $k = 3$, generating $(W_1^{(3)}, W_2^{(3)}, W_3^{(3)})^T$. In (o), it runs VertexTest on $(W_1^{(3)}, W_2^{(3)}, W_3^{(3)})^T$ and determines that $(W_1^{(3)}, W_2^{(3)}, W_3^{(3)})^T$ is a permutation of $(P_1, P_2, P_3)^T$.

where the ratio is defined to be ∞ when the denominator is zero. This estimator arises from applying the VC inequality to the following expression:

$$\kappa^*(F_0 \,|\, F_1, \ldots, F_M) = \max_{\boldsymbol{\mu} \in \Delta_M} \kappa^*(F_0 \,|\, \sum_{i=1}^{M} \mu_i F_i) = \max_{\boldsymbol{\mu} \in \Delta_M} \inf_{E \in \mathcal{E}, \sum_{i=1}^{M} \mu_i F_i(E) > 0} \frac{F_0(E)}{\sum_{i=1}^{M} \mu_i F_i(E)},$$

where the last equality uses Proposition 2. Let $\widehat{\boldsymbol{\mu}}$ denote a point where the maximum is achieved in (6). Then, $\widehat{\boldsymbol{\nu}} \coloneqq \widehat{\kappa}\widehat{\boldsymbol{\mu}}$ estimates the vector $(\nu_1, \ldots, \nu_M)$ attaining the maximum

in (3). See Proposition 2 of Blanchard and Scott (2014) to find a proof that the proposed estimator is consistent.

Based on this estimator, we introduce estimators that under the assumptions of Theorem 1 converge to the base distributions uniformly in probability. Let $\boldsymbol{n} \equiv (n_1, \ldots, n_L)$; we write $\boldsymbol{n} \longrightarrow \infty$ to indicate that $\min_i n_i \longrightarrow \infty$.

Theorem 4 *Let $(\widehat{\nu}_{i,j})_{j \neq i}$ be a vector attaining the maximum in the definition of $\widehat{\kappa}_i \coloneqq \widehat{\kappa}(\tilde{P}_i^\dagger \,|\, \{\tilde{P}_j^\dagger : j \neq i\})$ and*

$$\widehat{Q}_i = \frac{\tilde{P}_i^\dagger - \sum_{j \neq i} \widehat{\nu}_{i,j} \tilde{P}_j^\dagger}{1 - \widehat{\kappa}_i}.$$

Then, under the assumptions of Theorem 1, $\forall i = 1, \ldots, L$, $\sup_{E \in \mathcal{E}} |\widehat{Q}_i(E) - P_i(E)| \xrightarrow{i.p.} 0$ as $\boldsymbol{n} \longrightarrow \infty$.

5.2. Demixing Mixed Membership Models

In this section, we develop a novel estimator that can be used to extend the Demix algorithm to the finite sample case. Uniform convergence results typically assume access to i.i.d. samples. The challenge of developing an estimator for Demix is that because of the recursive nature of the Demix algorithm, we cannot assume access to i.i.d. samples to estimate every distribution that arises. Nonetheless, we show that uniform convergence of distributions propagates through the algorithm if we employ an estimator of κ^* with a known rate of convergence.

Let $\widehat{F}$ and $\widehat{H}$ be estimates of distributions F and H, respectively. We introduce the following estimator:

$$\widehat{\kappa}(\widehat{F} \,|\, \widehat{H}) = \inf_{E \in \mathcal{E}} \frac{\widehat{F}(E) + \gamma_{\boldsymbol{n}}}{(\widehat{H}(E) - \gamma_{\boldsymbol{n}})_+}$$

where $\gamma_{\boldsymbol{n}} = \sum_{i=1}^{L} \epsilon_i(\frac{1}{n_i})$. Our estimator is closely related to the estimator from Blanchard et al. (2010): if $\widehat{F}$ and $\widehat{H}$ are empirical distributions, e.g., $\widehat{F} = \tilde{P}_i^\dagger$ and $\widehat{H} = \tilde{P}_j^\dagger$, then their estimator for $\kappa^*(F \,|\, H)$ is $\inf_{E \in \mathcal{E}} \frac{\widehat{F}(E) + \epsilon_i(\frac{1}{n_i})}{(\widehat{H}(E) - \epsilon_j(\frac{1}{n_j}))_+}$. Note that our proofs only require that $\gamma_{\boldsymbol{n}}$ include the terms $\epsilon_i(\frac{1}{n_i})$ corresponding to $\tilde{P}_i$ that the estimators $\widehat{F}$ and $\widehat{H}$ use samples from; to simplify presentation, however, we include all the terms, which leads to bounds that are looser by only a constant factor.

Based on the estimator $\widehat{\kappa}$, we introduce the following estimator of the residue of F wrt H.

Definition 6 *Let $\widehat{F}$ and $\widehat{H}$ be estimators of F and H, respectively, where $F \neq H$ and let $G \longleftarrow$ Residue$(F \,|\, H)$ and $\widehat{G} \longleftarrow$ ResidueHat$(\widehat{F} \,|\, \widehat{H})$. We call $\widehat{G}$ a* ResidueHat estimator *of order $k \geqslant 1$ if (i) $F, H \in \mathrm{conv}(P_1, \ldots, P_L)$, and (ii) at least one of $\widehat{F}$ and $\widehat{H}$ is a ResidueHat estimator of order $k-1$ and the other is either an empirical distribution or a ResidueHat estimator of order less than or equal to $k-1$. We call $\widehat{G}$ a* ResidueHat estimator *of order 0 if (i) holds, and $\widehat{F}$ and $\widehat{H}$ are empirical distributions.*

Algorithm 10 ResidueHat($\widehat{F} \,|\, \widehat{H}$)

Input: $\widehat{F}, \widehat{H}$ are estimates of F, H

$\widehat{\kappa} \longleftarrow \widehat{\kappa}(\widehat{F} \,|\, \widehat{H})$

return $\frac{\widehat{F} - \widehat{\kappa}(1-\widehat{H})}{1-\widehat{\kappa}}$

Algorithm 11 DemixHat($\widehat{S}_1, \ldots, \widehat{S}_K \,|\, \epsilon$)

Input: $\widehat{S}_1, \ldots, \widehat{S}_K$ are ResidueHat estimates

if $K = 2$ **then**
  return $(\text{ResidueHat}(\widehat{S}_1 \,|\, \widehat{S}_2), \text{ResidueHat}(\widehat{S}_2 \,|\, \widehat{S}_1))^T$
else
  $(R_2, \ldots, R_K)^T \longleftarrow \text{FindFaceHat}(\widehat{S}_1, \ldots, \widehat{S}_K \,|\, \epsilon)$
  $(\widehat{Q}_1, \cdots, \widehat{Q}_{K-1})^T \longleftarrow \text{DemixHat}(\widehat{R}_2, \cdots, \widehat{R}_K)$
  $\widehat{Q}_K \longleftarrow \frac{1}{K}\sum_{i=1}^{K} \widehat{S}_i$
  for $i = 1, \ldots, K-1$ **do**
    $\widehat{Q}_K \longleftarrow \text{ResidueHat}(\widehat{Q}_K \,|\, \widehat{Q}_i)$
  end for
  return $(\widehat{Q}_1, \cdots, \widehat{Q}_K)^T$
end if

Note that the above definition is recursive and matches the recursive structure of the Demix algorithm. We suppress the qualifier "of order k" when it is not relevant.

To use ResidueHat estimators to estimate the P_is, we build on the rate of convergence result from Scott (2015). In Scott (2015), a rate of convergence was established for an estimator of κ^* using empirical distributions; we extend these results to our setting of recursive estimators and achieve the same rate of convergence. To ensure that this rate of convergence holds for every estimate in our algorithm, we introduce the following condition.

(A″) $P_1, \ldots, P_L$ are such that $\forall i \ \text{supp}(P_i) \nsubseteq \cup_{j \neq i} \text{supp}(P_j)$.

Note that this assumption implies joint irreducibility and is a natural generalization of the separability assumption.

The following result establishes sufficient conditions under which ResidueHat estimates converge uniformly.

Proposition 4 *If $P_1, \ldots, P_L$ satisfy **(A″)** and $\widehat{G}$ is a ResidueHat estimator of a distribution $G \in \text{conv}(P_1, \ldots, P_L)$, then $\sup_{E \in \mathcal{E}} |\widehat{G}(E) - G(E)| \xrightarrow{i.p.} 0$ as $\boldsymbol{n} \longrightarrow \infty$.*

Based on the ResidueHat estimators, we introduce an empirical version of the Demix algorithm—DemixHat (see Algorithm 11). The main differences are that *(i)* we replace the Residue function with the ResidueHat function, *(ii)* we replace line 7 in the Demix algorithm with a sequence of applications of the two-sample κ^* operator, as mentioned just before Theorem 2, and *(iii)* DemixHat requires specification of a hyperparameter $\epsilon \in (0, 1)$. We replace the multi-sample κ^* with the two-sample κ^* because there is no known estimator with a rate of convergence for the multi-sample κ^*, and the rate of convergence is

Algorithm 12 FindFaceHat($\widehat{S}_1, \ldots, \widehat{S}_K \,|\, \epsilon$)

Input: $\widehat{S}_1, \ldots, \widehat{S}_K$ are ResidueHat estimates

$\widehat{Q} \longleftarrow$ uniformly distributed random element from $\text{conv}(\widehat{S}_2, \ldots, \widehat{S}_K)$
for $n = 2, 3, \ldots$ **do**
  Set $\widehat{R}_i \longleftarrow \text{ResidueHat}(\frac{1}{n}\widehat{S}_i + \frac{n-1}{n}\widehat{Q}) \,|\, \widehat{S}_1)$ for all $i \in \{2, \ldots, K\}$
  if $\text{FaceTestHat}(\widehat{R}_2, \cdots, \widehat{R}_K \,|\, \epsilon)$ **then**
    return $(\widehat{R}_2, \cdots, \widehat{Q}_K)^T$
  end if
end for

Algorithm 13 FaceTestHat($\widehat{Q}_1, \cdots, \widehat{Q}_K \,|\, \epsilon$)

Set $\boldsymbol{Z}_{i,j} \coloneqq \mathbf{1}\{\widehat{\kappa}(\widehat{Q}_i \,|\, \widehat{Q}_j) > \epsilon\}$ for $i \neq j$
if $\boldsymbol{Z}$ has a zero off-diagonal entry **then**
  return 0
else
  return 1
end if

essential to our consistency proof. The hyperparameter ϵ gives a tradeoff between runtime and accuracy. The runtime increases with increasing ϵ, but the amount of uncertainty about whether DemixHat executes successfully decreases with increasing ϵ.

We now state our main estimation result.

Theorem 5 *Let $\delta > 0$ and $\epsilon \in (0, 1)$. Suppose that $P_1, \ldots, P_L$ satisfy **(A″)** and $\boldsymbol{\Pi}$ has full rank. Then, with probability tending to 1 as $\boldsymbol{n} \longrightarrow \infty$, DemixHat($\tilde{P}_1^\dagger, \ldots, \tilde{P}_L^\dagger \,|\, \epsilon$) returns $(\widehat{Q}_1, \ldots, \widehat{Q}_L)$ for which there exists a permutation $\sigma : [L] \longrightarrow [L]$ such that for every $i \in [L]$,*

$$\sup_{E \in \mathcal{E}} |\widehat{Q}_i(E) - P_{\sigma(i)}(E)| < \delta.$$

5.3. Classification with Partial Labels

In this section, we present a finite sample algorithm for the decontamination of a partial label model (see Algorithm 14). This algorithm is based on a different approach from PartialLabel (Algorithm 7): it combines DemixHat with an empirical version of the VertexTest algorithm (see Algorithm 17). The reason for this is that we have an estimator with a rate of convergence for the two-sample κ^*, whereas there is no known estimator with a rate of convergence for the multi-sample κ^*. We leverage this rate of convergence to prove the consistency of our algorithm.

We make an assumption that simplifies our algorithm: $\boldsymbol{\Pi}^+$ satisfies

(D) there does not exist $i, j \in [L]$ such that $\boldsymbol{\Pi}^+_{i,:} = \boldsymbol{e}_j^T$.

In words, this says that there is no contaminated distribution $\tilde{P}_i$ and base distribution P_j such that $\tilde{P}_i = P_j$. We emphasize that we make this assumption only to simplify the

Algorithm 14 PartialLabelHat($\mathbf{\Pi}^+, (\tilde{P}_1^\dagger, \ldots, \tilde{P}_M^\dagger)^T \,|\, \epsilon$)

1: $(\widehat{Q}_1, \ldots, \widehat{Q}_L)^T \longleftarrow$ DemixHat$(\tilde{P}_1^\dagger, \ldots, \tilde{P}_M^\dagger \,|\, \epsilon)$
2: (FoundVertices, $\boldsymbol{C}$) $\longleftarrow$ VertexTestHat$(\mathbf{\Pi}^+, (\tilde{P}_1^\dagger, \ldots, \tilde{P}_M^\dagger)^T, (\widehat{Q}_1, \ldots, \widehat{Q}_L)^T)$
3: **return** $\boldsymbol{C}(\widehat{Q}_1, \ldots, \widehat{Q}_L)^T$

presentation and development of the algorithm; one can reduce any instance of a partial label model satisfying **(B3)** and **(A)** to an instance of a partial label model that also satisfies **(D)**. We defer the sketch of this reduction to Section E.3.

We now state our main estimation result for classification with partial labels.

Theorem 6 *Let $\delta > 0$ and $\epsilon \in (0,1)$. Suppose that $P_1, \ldots, P_L$ satisfy **(A″)**, $\mathbf{\Pi}$ has full rank, the columns of $\mathbf{\Pi}^+$ are unique and $\mathbf{\Pi}^+$ satisfies **(D)**. Then, with probability tending to 1 as $\boldsymbol{n} \longrightarrow \infty$, PartialLabelHat($\mathbf{\Pi}^+, (\tilde{P}_1^\dagger, \ldots, \tilde{P}_M^\dagger)^T \,|\, \epsilon$) returns $(\widehat{Q}_1, \ldots, \widehat{Q}_L)^T$ such that for every $i \in [L]$,*

$$\sup_{E \in \mathcal{E}} |\widehat{Q}_i(E) - P_i(E)| < \delta.$$

5.4. Sieve Estimators

In the preceding, we have assumed a fixed VC class to simplify the presentation. However, these results easily extend to the setting where $\mathcal{E} = \mathcal{E}_k$ and $k \longrightarrow \infty$ at a suitable rate depending on the growth of the VC dimensions V_k. This allows for the P_is to be estimated uniformly on arbitrarily complex events, e.g., $\mathcal{E}_k$ is the set of unions of k open balls.

6. Discussion

In this paper, we have studied the problem of how to recover the base distributions $\boldsymbol{P}$ from the contaminated distributions $\tilde{\boldsymbol{P}}$ without knowledge of the mixing matrix $\mathbf{\Pi}$. We used a common set of concepts and techniques to solve three popular machine learning problems that arise in this setting: multiclass classification with label noise, demixing of mixed membership models, and classification with partial labels. Our technical contributions include: (i) We provide sufficient and sometimes necessary conditions for identifiability for all three problems. (ii) We give nonparametric algorithms for the infinite and finite sample settings. (iii) We provide a new estimator for iterative applications of κ^* that is of independent interest. (iv) Finally, our work provides a novel geometric perspective on each of the three problems.

Our results improve on what was previously known for all three problems. For multiclass classification with label noise and unknown $\mathbf{\Pi}$, previous work had only considered the case $M = L = 2$. Our work achieves a generalization to arbitrarily many distributions. For demixing of mixed membership models, previous algorithms with theoretical guarantees required a finite sample space. Our work allows for a much more general set of distributions. Finally, for classification with partial labels, previous work on learnability assumed the realizable case (non-overlapping $P_1, \ldots, P_L$) and assumed strong conditions on label co-occurence in partial labels. Our analysis covers the agnostic case and a much wider set of partial labels.

Our work has also highlighted the advantages and disadvantages associated with the two-sample κ^* operator and multi-sample κ^* operator, respectively. Algorithms that only use the two-sample κ^* operator have the following two advantages: (i) the geometry of the two-sample κ^* operator is simpler than the geometry of the multi-sample κ^* operator and, as such, can be more tractable. Indeed, in recent years, several practical algorithms for estimating the two-sample κ^* have been developed (see Jain et al. (2016) and references therein). (ii) We have estimators with established rates of convergence for the two-sample κ^* operator, but not for the multi-sample κ^* operator. On the other hand, algorithms that use the multi-sample κ^* operator have fewer steps.

The aims of this work are mainly theoretical, but we believe that our work can inform practical algorithms. First, we note that while we have emphasized recovery of $\boldsymbol{P}$, another interpretation is that our work deals with estimating $\mathbf{\Pi}$. One can then plug our estimate of $\mathbf{\Pi}$ into corrected losses for multiclass classification with label noise and classification with partial labels that require knowledge of $\mathbf{\Pi}$ (Cid-Sueiro, 2012; Menon et al., 2015b; van Rooyen and Williamson, 2015; Patrini et al., 2017). Thus, in general, our work can be applied in this two-stage approach. Second, when L or M are small, the Demix algorithm is practical. For example, Metodiev and Thaler (2018b) apply the Demix algorithm to a high-energy physics application where $L = M = 2$. It is of interest to examine more generally whether variants of Demix work when M or L are small. Third, we conjecture that our analysis suggests novel principles for designing algorithms. For example, an alternative approach to the three problems in question is to embed the contaminated distributions in a reproducing kernel Hilbert Space and to estimate the $\mathbf{\Pi}$ matrix by setting up an optimization problem (e.g., see Ramaswamy et al. (2016) for the setting where there are two distributions). One of our necessary conditions, maximality (see Appendix C), suggests formulating the optimization problem to search for base distributions that *(i)* explain the observed contaminated distributions and *(ii)* whose convex hull is as large as possible. In this way, we believe that our general treatment of these three problems that arise in mutual contamination models provides intuitions that could be useful for designing new algorithms.

Although our experiments in Appendix G suggest that joint irreducibility of the base distributions is a reasonable assumption, it is nevertheless worthwhile to consider what questions arise if the base distributions are not exactly jointly irreducible. We see two possible research directions. First, one could perform a stability analysis: when the base distributions are not jointly irreducible, but are nearly jointly irreducible (in some sense that would need to be defined precisely), does the estimate of $\mathbf{\Pi}$ remain close to the true $\mathbf{\Pi}$? A second research question is to reinterpret the problem of demixing of mixed membership models as a dimensionality reduction problem. That is, given a large set of distributions, one could seek to represent them as convex combinations of a small set of irreducible base distributions. Then, the challenge would be to define an appropriate measure of approximation quality and to determine whether our approach could be useful for designing a consistent algorithm for the best approximation.

Acknowledgements

We thank the anonymous reviewers for their extremely helpful and insightful comments. This work was supported in part by NSF grants 1422157 and 1838179. Gilles Blanchard

acknowledges support from the DFG, under the Research Unit FOR-1735 "Structural Inference in Statistics - Adaptation and Efficiency", and under the Collaborative Research Center SFB-1294 "Data Assimilation".

Appendix A. Outline of Appendices

To begin, we introduce additional notation for the appendices. In Section C, we discuss how strong our sufficient conditions are and present factorization results that suggest that they are reasonable. In Section D, we give our identifiability analysis of demixing mixed membership models and classification with partial labels. In Section E, we prove our results on the ResidueHat estimator, as well as the finite sample algorithms for demixing mixed membership models and classification with partial labels. In Section F, we state some lemmas from related papers that we use in our arguments.

Appendix B. Notation for Appendices

Let $A \subset \mathbb{R}^d$ be a set. Let $\operatorname{aff} A$ denote the affine hull of A, i.e., $\operatorname{aff} A = \{\sum_{i=1}^K \theta_i \boldsymbol{x}_i | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_K \in A, \sum_{i=1}^K \theta_i = 1\}$. A° denotes the relative interior of A, i.e., $A^\circ = \{x \in A | B(x, r) \cap \operatorname{aff} A \subseteq A \text{ for some } r > 0\}$. Then, ∂A denotes the relative boundary of A, i.e., $\partial A = A \backslash A^\circ$. In addition, let $\|\cdot\|$ denote an arbitrary finite-dimensional norm on $\mathbb{R}^L$. For two vectors, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^K$, define

$$\min(\boldsymbol{x}^T, \boldsymbol{y}^T) = (\min(x_1, y_1), \ldots, \min(x_K, y_K)).$$

$\boldsymbol{x} \geqslant \boldsymbol{y}$ means $x_i \geqslant y_i \; \forall i \in [K]$.

For distributions $Q_1, \ldots, Q_k$, we use $\operatorname{conv}(Q_1, \ldots, Q_K)^\circ$ to denote the relative interior of their convex hull and have that

$$\operatorname{conv}(Q_1, \ldots, Q_K)^\circ = \{\sum_{i=1}^K \alpha_i Q_i : \alpha_i > 0, \sum_{i=1}^K \alpha_i = 1\}.$$

Note that when $Q_1, \ldots, Q_K$ are discrete distributions, this definition coincides with the definition of the relative interior of a set of Euclidean vectors.

We use the following affine mapping throughout the paper: $m_\nu(\boldsymbol{x}, \boldsymbol{y}) = (1 - \nu)\boldsymbol{x} + \nu \boldsymbol{y}$ where $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^L$ and $\nu \in [0, 1]$. Overloading notation, when Q_1 and Q_2 are distributions, we define $m_\nu(Q_1, Q_2) = (1 - \nu)Q_1 + \nu Q_2$. Note that if $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2 \in \Delta_L$ and $Q_1 = \boldsymbol{\eta}_1^T \boldsymbol{P}$ and $Q_2 = \boldsymbol{\eta}_2^T \boldsymbol{P}$, then $m_\nu(\boldsymbol{\eta}_1, \boldsymbol{\eta}_2)$ is the mixture proportion for the distribution $m_\nu(Q_1, Q_2)$.

Appendix C. Factorization Results

In this section, we discuss whether our sufficient conditions are necessary. For the problems of demixing mixed membership models and classification with partial labels, we provide factorization results that suggest that our sufficient conditions are not much stronger than what is necessary.

C.1. Multiclass Classification with Label Noise

Our sufficient condition **(B1)** for multiclass classification with label noise is not necessary. Rather, **(B1)** is one of several possible sufficient conditions, and one that reflects a low noise assumption as illustrated in Figure 1. Consider the case $L = M = 2$ where **(B1)** is equivalent to $\pi_{1,2} + \pi_{2,1} < 1$. Recovery is still possible if $\pi_{1,2} + \pi_{2,1} > 1$ since one can simply swap $\tilde{P}_1$ and $\tilde{P}_2$ in a decontamination procedure. $\pi_{1,2} + \pi_{2,1} < 1$ is only necessary if one assumes that most of the training labels are correct, which is what $\pi_{1,2} + \pi_{2,1} < 1$ essentially says. For larger $L = M$, **(B1)** says in a sense that most of the data from $\tilde{P}_i$ come from P_i for every i. Other sufficient conditions are possible (as in the binary case), but these would require at least one $\tilde{P}_i$ to contain a significant portion of some P_j, $j \neq i$. Regarding **(A)**, Blanchard et al. (2016) study the question of necessity for joint irreducibility in the case $L = M = 2$ and show that under mild assumptions on the decontamination procedure, joint irreducibility is necesssary.

C.2. Demixing Mixed Membership Models

Recall the definition of a factorization: $\tilde{\boldsymbol{P}}$ is factorizable if there exists $(\boldsymbol{\Pi}, \boldsymbol{P}) \in \Delta_L^M \times \mathcal{P}^L$ such that $\tilde{\boldsymbol{P}} = \boldsymbol{\Pi}\boldsymbol{P}$; we call $(\boldsymbol{\Pi}, \boldsymbol{P})$ a factorization of $\tilde{\boldsymbol{P}}$.

Our sufficient conditions are not much stronger than what is required by factorizations that satisfy the two forthcoming desirable properties.

Definition 7 *We say a factorization $(\boldsymbol{\Pi}, \boldsymbol{P})$ of $\tilde{\boldsymbol{P}}$ is* maximal ***(M)*** *iff for all factorizations $(\boldsymbol{\Pi}', \boldsymbol{P}')$ of $\tilde{\boldsymbol{P}}$ with $\boldsymbol{P}' = (P_1', \ldots, P_L')^T \in \mathcal{P}^L$, it holds that $\{P_1', \ldots, P_L'\} \subseteq \operatorname{conv}(P_1, \ldots, P_L)$.*

In words, $\boldsymbol{P} = (P_1, \ldots, P_L)^T$ is a maximal collection of base distributions if it is not possible to move any of the P_is outside of $\operatorname{conv}(P_1, \ldots, P_L)$ and represent $\tilde{\boldsymbol{P}}$.

Definition 8 *We say a factorization $(\boldsymbol{\Pi}, \boldsymbol{P})$ of $\tilde{\boldsymbol{P}}$ is* linear ***(L)*** *iff $\{P_1, \ldots, P_L\} \subseteq \operatorname{span}(\tilde{P}_1, \ldots, \tilde{P}_M)$.*

We believe that **(L)** is a reasonable requirement because it holds in the common situation in which there exist $\boldsymbol{\pi}_{i_1}, \ldots, \boldsymbol{\pi}_{i_L}$ that are linearly independent. Then for $I = \{i_1, \ldots, i_L\}$, we can write $\tilde{\boldsymbol{P}}_I = \boldsymbol{\Pi}_I \boldsymbol{P}$ where $\boldsymbol{\Pi}_I$ is the submatrix of $\boldsymbol{\Pi}$ containing only the rows indexed by I and $\tilde{\boldsymbol{P}}_I$ is similarly defined. Then, $\boldsymbol{\Pi}_I$ is invertible and $\boldsymbol{P} = \boldsymbol{\Pi}_I^{-1} \tilde{\boldsymbol{P}}_I$.

Factorizations that satisfy **(A)** and **(B2)** are maximal and linear.

Proposition 5 *Let $(\boldsymbol{\Pi}, \boldsymbol{P})$ be a factorization of $\tilde{\boldsymbol{P}}$. If $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(A)** and **(B2)**, then $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(M)** and **(L)**.*

Proof We first show that $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(L)**. By hypothesis, $P_1, \ldots, P_L$ are jointly irreducible. By Lemma 16, $P_1, \ldots, P_L$ are linearly independent. Since by hypothesis $\boldsymbol{\Pi}$ has full rank, there exist L rows in $\boldsymbol{\Pi}$, $\boldsymbol{\pi}_{i_1}, \ldots, \boldsymbol{\pi}_{i_L}$, that are linearly independent. By Lemma 16, $\tilde{P}_{i_1}, \ldots, \tilde{P}_{i_L}$ are linearly independent. Since $\boldsymbol{\Pi}\boldsymbol{P} = \tilde{\boldsymbol{P}}$, $\operatorname{span}(\tilde{P}_1, \ldots, \tilde{P}_M) \subseteq \operatorname{span}(P_1, \ldots, P_L)$. Since $\dim \operatorname{span}(\tilde{P}_1, \ldots, \tilde{P}_M) \geqslant L$, we have $\operatorname{span}(\tilde{P}_1, \ldots, \tilde{P}_M) = \operatorname{span}(P_1, \ldots, P_L)$. Therefore, $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(L)**.

Now, we show that $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(M)**. Suppose that there is another solution $(\boldsymbol{\Pi}', \boldsymbol{P}')$ with $\boldsymbol{P}' = (P_1', \ldots, P_L')^T$ such that $\boldsymbol{\Pi}'\boldsymbol{P}' = \tilde{\boldsymbol{P}}$ and with some P_i' such that

$P_i' \notin \text{conv}(P_1, \ldots, P_L)$. We claim that $P_i' \notin \text{span}(P_1, \ldots, P_L)$. Towards a contradiction, suppose that $P_i' \in \text{span}(P_1, \ldots, P_L)$ so that we can write $P_i' = \sum_{i=1}^{L} a_i P_i$. Then, at least one of the a_i is negative since, by assumption, $P_i' \notin \text{conv}(P_1, \ldots, P_L)$. But, by joint irreducibility of $P_1, \ldots, P_L$, P_i' is not a distribution, which is a contradiction. So, the claim follows. But, then, since $\text{span}(P_1, \ldots, P_L) = \text{span}(\tilde{P}_1, \ldots, \tilde{P}_M)$, we must have that $\text{span}(\tilde{P}_1, \ldots, \tilde{P}_M) \subseteq \text{span}(P_1', \ldots, P_{i-1}', P_{i+1}', \ldots, P_L)$, which is impossible since $\dim \text{span}(\tilde{P}_1, \ldots, \tilde{P}_M) = L$. ■

Maximal and linear factorizations imply conditions that are not much weaker than our sufficient conditions.

Theorem 7 *Let $(\boldsymbol{\Pi}, \boldsymbol{P})$ be a factorization of $\tilde{\boldsymbol{P}}$. If $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(M)**, then*

(A′) *$\forall i$, P_i is irreducible with respect to every distribution in $\text{conv}(\{P_j : j \neq i\})$.*

*If $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(L)**, then*

(B′) $\text{rank}(\boldsymbol{\Pi}) \geqslant \dim \text{span}(P_1, \ldots, P_L)$.

Proof

(A′) We prove the contrapositive. Suppose that there is some P_i and $Q \in \text{conv}(\{P_j : j \neq i\})$ with $Q = \sum_{j \neq i} \beta_j P_j$ such that P_i is not irreducible wrt Q. Then, there is some distribution G and $\gamma \in (0, 1]$ such that $P_i = \gamma Q + (1 - \gamma) G$.

Suppose $\gamma = 1$. Then, $P_i = Q \in \text{conv}(\{P_k : k \neq i\})$. But, then $\tilde{P}_1, \ldots, \tilde{P}_M \in \text{conv}(\{P_j : j \neq i\} \cup \{R\})$ for any distribution $R \notin \text{conv}(P_1, \ldots, P_L)$. This shows that $(\boldsymbol{\Pi}, \boldsymbol{P})$ does not satisfy **(M)**.

Therefore, assume that $\gamma \in (0, 1)$. Either $G \in \text{conv}(P_1, \ldots, P_L)$ or $G \notin \text{conv}(P_1, \ldots, P_L)$. Suppose that $G \in \text{conv}(P_1, \ldots, P_L)$. Then, there exist $\alpha_1, \ldots, \alpha_L$ all nonnegative and summing to 1 such that

$$P_i = \gamma Q + (1 - \gamma)(\alpha_1 P_1 + \ldots + \alpha_L P_L).$$

Therefore, $P_i \in \text{conv}(\{P_k : k \neq i\})$. Then, by the argument in the previous paragraph, $(\boldsymbol{\Pi}, \boldsymbol{P})$ does not satisfy **(M)**.

Now, suppose that $G \notin \text{conv}(P_1, \ldots, P_L)$. Since $P_i \in \text{conv}(G, Q)$ and $Q \in \text{conv}(\{P_j : j \neq i\})$, we have that $\text{conv}(\{P_j : j \neq i\} \cup \{G\}) \supset \text{conv}(P_1, \ldots, P_L)$. Then, $\tilde{P}_1, \ldots, \tilde{P}_M \in \text{conv}(\{P_j : j \neq i\} \cup \{G\})$. This shows that $(\boldsymbol{\Pi}, \boldsymbol{P})$ does not satisfy **(M)**. The result follows.

(B′) Clearly, $\dim \text{span}(\tilde{P}_1, \ldots, \tilde{P}_M) \leqslant \dim \text{span}(P_1, \ldots, P_L)$ since $\tilde{P}_i = \boldsymbol{\pi}_i^T \boldsymbol{P}$ for all $i \in [M]$. Since $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(L)**, $\text{span}(P_1, \ldots, P_L) \subset \text{span}(\tilde{P}_1, \ldots, \tilde{P}_L)$, which implies that

$$\dim \text{span}(P_1, \ldots, P_L) \leqslant \dim \text{span}(\tilde{P}_1, \ldots, \tilde{P}_M).$$

Therefore, $\dim \text{span}(\tilde{P}_1, \ldots, \tilde{P}_M) = \dim \text{span}(P_1, \ldots, P_L)$. Then, since $\tilde{P}_1, \ldots, \tilde{P}_M \in \text{range}(\boldsymbol{\Pi})$, $\dim \text{span}(P_1, \ldots, P_L) \leqslant \dim \text{range}\, \boldsymbol{\Pi}$. By Result 3.117 of Axler (2015),

$$\text{rank}(\boldsymbol{\Pi}) = \dim \text{range}\, \boldsymbol{\Pi} \geqslant \dim \text{span}(P_1, \ldots, P_L).$$

■

As a corollary, Theorem 7 implies that if there is a linear factorization $(\boldsymbol{\Pi}, \boldsymbol{P})$ of $\tilde{P}$ and $P_1, \ldots, P_L$ are linearly independent, then there must be at least as many contaminated distributions as base distributions, i.e., $M \geqslant L$. Also, note that **(A′)** appears as a sufficient condition in Sanderson and Scott (2014).

By comparing **(A)** with **(A′)** and **(B2)** with **(B′)**, we see that the proposed sufficient conditions are not much stronger than **(M)** and **(L)** require. Since joint irreducibility of $P_1, \ldots, P_L$ entails their linear independence by Lemma 16, under **(A)**, **(B2)** and **(B′)** are the same. **(A)** differs from **(A′)** in that it requires that a slightly larger set of distributions are irreducible with respect to convex combinations of the remaining distributions. Specifically, under **(A)**, *every convex combination* of a subset of the P_is is irreducible with respect to every convex combination of the other P_is whereas **(A′)** only requires that every P_i be irreducible with respect to every convex combination of the other P_is.

C.3. Classification with Partial Labels

Most of our definitions and results for classification with partial labels parallel those of demixing mixed membership models. We say that $\tilde{\boldsymbol{P}}$ is $\boldsymbol{\Pi}^+$-factorizable if there exists a pair $(\boldsymbol{\Pi}, \boldsymbol{P}) \in \Delta_L^M \times \mathcal{P}^L$ that solves (2) such that $\boldsymbol{\Pi}$ is consistent with $\boldsymbol{\Pi}^+$; we call $(\boldsymbol{\Pi}, \boldsymbol{P})$ an $\boldsymbol{\Pi}^+$-factorization of $\tilde{\boldsymbol{P}}$. We say a partial label model is identifiable if given $(\tilde{\boldsymbol{P}}, \boldsymbol{\Pi}^+)$, $\tilde{\boldsymbol{P}}$ has a unique $\boldsymbol{\Pi}^+$-factorization $(\boldsymbol{\Pi}, \boldsymbol{P})$.

Our definitions of maximal and linear $\boldsymbol{\Pi}^+$-factorizations resemble definitions 7 and 8.

Definition 9 *We say a $\boldsymbol{\Pi}^+$-factorization $(\boldsymbol{\Pi}, \boldsymbol{P})$ of $\tilde{\boldsymbol{P}}$ is* maximal **(M)** *iff for all $\boldsymbol{\Pi}^+$-factorizations $(\boldsymbol{\Pi}', \boldsymbol{P}')$ of $\tilde{\boldsymbol{P}}$ with $\boldsymbol{P}' = (P_1', \ldots, P_L')^T \in \mathcal{P}^L$, it holds that $\{P_1', \ldots, P_L'\} \subseteq \text{conv}(P_1, \ldots, P_L)$.*

Definition 10 *We say a $\boldsymbol{\Pi}^+$-factorization $(\boldsymbol{\Pi}, \boldsymbol{P})$ of $\tilde{\boldsymbol{P}}$ is* linear **(L)** *iff $\{P_1, \ldots, P_L\} \subseteq \text{span}(\tilde{P}_1, \ldots, \tilde{P}_M)$.*

Similarly, $\boldsymbol{\Pi}^+$-factorizations that satisfy **(A)** and **(B3)** are maximal and linear. The proof is identical and, accordingly, omitted.

Proposition 6 *Let $(\boldsymbol{\Pi}, \boldsymbol{P})$ be a $\boldsymbol{\Pi}^+$-factorization of $\tilde{\boldsymbol{P}}$. If $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(A)** and **(B3)**, then $(\boldsymbol{\Pi}, \boldsymbol{P})$ satisfies **(M)** and **(L)**.*

Linear $\boldsymbol{\Pi}^+$-factorizations must satisfy **(B′)**; indeed, the proof is identical to the proof for linear factorizations. However, maximal $\boldsymbol{\Pi}^+$-factorizations need not satisfy **(A′)**. Consider the following counterexample. Let $Q_1 \sim \text{unif}(0, 2)$ and $Q_2 \sim \text{unif}(1, 3)$. Let $P_1 = \frac{2}{3}Q_1 + \frac{1}{3}Q_2$, $P_2 = \frac{1}{3}Q_1 + \frac{2}{3}Q_2$, $\tilde{P}_1 = P_1$, and $\tilde{P}_2 = P_2$. Then, $\boldsymbol{\Pi}^+ = \boldsymbol{I}_2$—the identity matrix. Then, any $(\boldsymbol{\Pi}', \boldsymbol{P}')$ that satisfies (2) and is consistent with $\boldsymbol{\Pi}^+$ must be such that $(P_1, P_2)^T = \boldsymbol{P}'$. Therefore, **(M′)** is satisfied. But, clearly, **(A′)** is not satisfied.

In summary, we are unable to offer a necessary condition that is close to **(A)**. On the other hand, **(B3)** is necessary.

Proposition 7 *Let $(\boldsymbol{\Pi}, \boldsymbol{P})$ be an $\boldsymbol{\Pi}^+$-factorization of $\tilde{\boldsymbol{P}}$. If $(\tilde{\boldsymbol{P}}, \boldsymbol{\Pi}^+)$ is identifiable, then the columns of $\boldsymbol{\Pi}^+$ are distinct.*

Proof First, suppose $(\tilde{\boldsymbol{P}}, \boldsymbol{\Pi}^+)$ is identifiable. Then, we can write $\tilde{\boldsymbol{P}} = \boldsymbol{\Pi}\boldsymbol{P}$ where $\boldsymbol{\Pi}$ is consistent with $\boldsymbol{\Pi}^+$. We claim that for all $i \neq j$, $P_i \neq P_j$. To the contrary, suppose that there exists $i \neq j$ such that $P_i = P_j$. Without loss of generality, suppose $i = 1, j = 2$. Then, we can write

$$\tilde{\boldsymbol{P}} = \begin{pmatrix} 2\boldsymbol{\Pi}_{:,1} & \boldsymbol{\Pi}_{:,3} & \dots & \boldsymbol{\Pi}_{:,L} \end{pmatrix} \begin{pmatrix} P_1 \\ P_3 \\ \vdots \\ P_L \end{pmatrix},$$

which contradicts the uniqueness of $\boldsymbol{P}$ and $\boldsymbol{\Pi}$.

Next, we give a proof by contraposition. Suppose that there exists $i \neq j$ such that $\boldsymbol{\Pi}^+_{:,i} = \boldsymbol{\Pi}^+_{:,j}$. Without loss of generality, let $i = 1$ and $j = 2$. Suppose that $(\boldsymbol{\Pi}, \boldsymbol{P})$ is consistent with $\boldsymbol{\Pi}^+$ and solves $\tilde{\boldsymbol{P}} = \boldsymbol{\Pi}\boldsymbol{P}$. Then, the pair $(\boldsymbol{\Pi}', \boldsymbol{P}')$ given by

$$\boldsymbol{\Pi}' = \begin{pmatrix} \boldsymbol{\Pi}_{:,2} & \boldsymbol{\Pi}_{:,1} & \boldsymbol{\Pi}_{:,3} & \dots & \boldsymbol{\Pi}_{:,L} \end{pmatrix}$$

$$\boldsymbol{P}' = \begin{pmatrix} P_2 \\ P_1 \\ P_3 \\ \vdots \\ P_L \end{pmatrix}$$

solves $\tilde{\boldsymbol{P}} = \boldsymbol{\Pi}'\boldsymbol{P}'$ and is consistent with $\boldsymbol{\Pi}^+$. If $P_1 = P_2$, then $(\tilde{\boldsymbol{P}}, \boldsymbol{\Pi}^+)$ is not identifiable, so we may rule out this case. Therefore, $\boldsymbol{P}' \neq \boldsymbol{P}$, yielding the result. ■

Appendix D. Identification

In this section, we establish our identification results, i.e., Theorems 2 and 3. We begin by proving Proposition 3. Second, we prove a set of useful lemmas in Section D.2. Third, we present our results on demixing mixed membership models in Section D.3. Finally, we present our results on classification with partial labels in Section D.4.

D.1. Proof of Proposition 3

Proof We prove the claims in order.

1. Without loss of generality, suppose $i = 1$ and let $A = [L]\backslash\{1\}$. There is at least one point attaining the maximum in the optimization problem $\kappa^*(\boldsymbol{\eta}_1 \,|\, \{\boldsymbol{\eta}_j : j \neq 1\})$ by Lemma 15. Take a G that achieves the maximum in $\kappa^*(Q_1 \,|\, \{Q_j : j \neq 1\})$, which exists also by Lemma 15. Then, we can write:

$$Q_1 = (1 - \sum_{j \geqslant 2} \mu_j)G + \sum_{j \geqslant 2} \mu_j Q_j. \tag{7}$$

Note that since $\boldsymbol{\eta}_1, \dots, \boldsymbol{\eta}_L$ are linearly independent and $P_1, \dots, P_L$ are jointly irreducible, $Q_1, \dots, Q_L$ are linearly independent by Lemma 16. Therefore, $\kappa^*(Q_1 \,|\, \{Q_j : j \neq 1\}) = \sum_{j \geqslant 2} \mu_j < 1$ because, if not, $Q_1 = \sum_{j \geqslant 2} \mu_j Q_j$.

Further, any G that satisfies (7) has the form $\sum_{i=1}^L \gamma_i P_i$ because (7) implies that $G \in \text{span}(Q_1, \dots, Q_L)$ and each $Q_i \in \text{conv}(P_1, \dots, P_L)$ by hypothesis. The γ_i must sum to one, and we have that they are nonnegative by joint irreducibility. That is, $\boldsymbol{\gamma} \equiv (\gamma_1, \dots, \gamma_L)^T$ is a discrete distribution. Then, the above equation is equivalent to

$$\boldsymbol{\eta}_1^T \boldsymbol{P} = (1 - \sum_{j \geqslant 2} \mu_j)\boldsymbol{\gamma}^T \boldsymbol{P} + \sum_{j \geqslant 2} \mu_j \boldsymbol{\eta}_j^T \boldsymbol{P}. \tag{8}$$

Since $P_1, \dots, P_L$ are jointly irreducible, $P_1, \dots, P_L$ are linearly independent by Lemma 16. By linear independence of $P_1, \dots, P_L$, we obtain

$$\boldsymbol{\eta}_1 = (1 - \sum_{j \geqslant 2} \mu_j)\boldsymbol{\gamma} + \sum_{j \geqslant 2} \mu_j \boldsymbol{\eta}_j. \tag{9}$$

Consequently, $\kappa^*(Q_1 \,|\, \{Q_j : j \neq 1\}) = \kappa^*(\boldsymbol{\eta}_1 \,|\, \{\boldsymbol{\eta}_j : j \neq 1\}) < 1$. This completes the proof of statement 1.

2. This result follows immediately from Lemma 15.

3. By equations (8) and (9), there is a one-to-one correspondence between the maximizer G to $\kappa^*(Q_1 \,|\, \{Q_j : j \neq 1\})$ and the maximizer $\boldsymbol{\gamma}$ to $\kappa^*(\boldsymbol{\eta}_1 \,|\, \{\boldsymbol{\eta}_j : j \neq 1\})$. The one-to-one correspondence is given by $G = \boldsymbol{\gamma}^T \boldsymbol{P}$.

■

D.2. Lemmas for Identification

We present some technical results that are used repeatedly for our identification results. Lemma 2 gives us some useful properties of the two-sample κ^* that we exploit in the PartialLabel and Demix algorithms. Statement *1* gives an alternative form of κ^*. Statement *2* gives the intuitive result that the residues lie on the boundary of the simplex. Statement *3* gives a useful relation for determining whether two mixture proportions are on the same face; we use this relation extensively in our algorithms.

Lemma 2 *Let $F_1, \dots, F_K$ be jointly irreducible distributions with $\boldsymbol{F} = (F_1, \dots, F_K)^T$, Q_1, Q_2 be two distributions such that $Q_i = \boldsymbol{\eta}_i^T \boldsymbol{F}$ where $\boldsymbol{\eta}_i \in \Delta_K$ for $i = 1, 2$ and $\boldsymbol{\eta}_1 \neq \boldsymbol{\eta}_2$. Let R be the residue of Q_1 wrt Q_2 and $R = \boldsymbol{\mu}^T \boldsymbol{F}$.*

1. *There is a one-to-one correspondence between the optimization problem in $\kappa^*(Q_1 \,|\, Q_2)$ and the optimization problem*

$$\max(\alpha \geqslant 1 | \exists \text{ a distribution } G \text{ s.t } G = Q_2 + \alpha(Q_1 - Q_2))$$

via $\alpha = (1 - \kappa)^{-1}$.

2. *$\boldsymbol{\mu} \in \partial\Delta_K$.*

3. *$\mathcal{N}(\boldsymbol{\eta}_2) \nsubseteq \mathcal{N}(\boldsymbol{\eta}_1)$ if and only if $R = Q_1$ if and only if $\kappa^*(Q_1 \,|\, Q_2) = 0$.*

Proof We note that we may assume that $R = \boldsymbol{\mu}^T \boldsymbol{F}$ since by definition of the residue, $R \in \text{span}(Q_1, Q_2)$ and $Q_1, Q_2 \in \text{conv}(F_1, \ldots, F_K)$.

1. Consider the linear relation: $Q_1 = (1-\kappa)G + \kappa Q_2$ where $\kappa \in [0,1]$. Since $F_1, \ldots, F_K$ are jointly irreducible and $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$ are linearly independent, Q_1 and Q_2 are linearly independent by Lemma 16. Therefore, $\kappa < 1$. We can rewrite the relation as

$$G = \frac{1}{1-\kappa} Q_1 - \frac{\kappa}{1-\kappa} Q_2 = \alpha Q_1 + (1-\alpha) Q_2$$

where $\alpha = \frac{1}{1-\kappa}$. The equivalence follows.

2. Since R is the residue of Q_1 wrt Q_2, by Proposition 3, $\boldsymbol{\mu}$ is the residue of $\boldsymbol{\eta}_1$ wrt $\boldsymbol{\eta}_2$ and $\boldsymbol{\mu} \in \Delta_K$. Therefore, by statement *1* in Lemma 2, $\boldsymbol{\mu}$ is such that α^* is maximized subject to the following constraints:

$$\begin{aligned} \boldsymbol{\mu} &= (1-\alpha^*)\boldsymbol{\eta}_2 + \alpha^* \boldsymbol{\eta}_1 \\ \alpha^* &\geqslant 1 \\ \boldsymbol{\mu} &\in \Delta_K. \end{aligned}$$

Suppose that $\min_i \mu_i > 0$. Then,

$$\boldsymbol{\mu} = (1-\alpha^*)\boldsymbol{\eta}_2 + \alpha^* \boldsymbol{\eta}_1 = \boldsymbol{\eta}_2 + \alpha(\boldsymbol{\eta}_1 - \boldsymbol{\eta}_2) > 0$$

so that there is some $\epsilon > 0$ such that

$$\begin{aligned} \boldsymbol{\mu}' &= (1-\alpha^*-\epsilon)\boldsymbol{\eta}_2 + (\alpha^*+\epsilon)\boldsymbol{\eta}_1 \\ \alpha^* + \epsilon &\geqslant 1 \\ \boldsymbol{\mu}' &\in \Delta_K. \end{aligned}$$

But, this contradicts the definition of α^* and $\boldsymbol{\mu}$. Therefore, $\min_i \mu_i = 0$. Consequently, $\boldsymbol{\mu} \in \partial \Delta_K$.

3. By definition of κ^*, it is clear that $R = Q_1$ if and only if $\kappa^*(Q_1 \,|\, Q_2) = 0$. Therefore, it suffices to show that $\mathcal{N}(\boldsymbol{\eta}_2) \nsubseteq \mathcal{N}(\boldsymbol{\eta}_1)$ if and only if $\kappa^*(Q_1 \,|\, Q_2) = 0$. Suppose $\mathcal{N}(\boldsymbol{\eta}_2) \nsubseteq \mathcal{N}(\boldsymbol{\eta}_1)$. Then, there must be $i \in [K]$ such that $\eta_{2,i} > 0$ and $\eta_{1,i} = 0$. For any $\alpha > 1$,

$$\min_{i \in [K]} (1-\alpha)\eta_{2,i} + \alpha \eta_{1,i} < 0;$$

but, this violates the constraint of the optimization problem. Therefore, $\alpha = 1$. By statement *1* in Lemma 2, $\kappa^*(\boldsymbol{\eta}_1 \,|\, \boldsymbol{\eta}_2) = 0$. By statement *1* of Proposition 3, $\kappa^*(Q_1 \,|\, Q_2) = \kappa^*(\boldsymbol{\eta}_1 \,|\, \boldsymbol{\eta}_2) = 0$.

Now, suppose $\mathcal{N}(\boldsymbol{\eta}_2) \subseteq \mathcal{N}(\boldsymbol{\eta}_1)$. Then, for any $i \in [K]$, if $\eta_{2,i} > 0$, then $\eta_{1,i} > 0$. Then, there is $\alpha > 1$ sufficiently close to 1 such that

$$\min_{i \in [K]} \eta_{2,i} + \alpha(\eta_{1,i} - \eta_{2,i}) \geqslant 0.$$

By statement *1* in this Lemma, $\kappa^*(\boldsymbol{\eta}_1 \,|\, \boldsymbol{\eta}_2) > 0$. By statement *1* of proposition 3, $\kappa^*(Q_1 \,|\, Q_2) = \kappa^*(\boldsymbol{\eta}_1 \,|\, \boldsymbol{\eta}_2) > 0$.

■

Lemma 3 *Let $0 \leqslant k < L$. If $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k \in \Delta_L$ are linearly independent and $\boldsymbol{w}_{k+1}, \ldots, \boldsymbol{w}_L \in \Delta_L$ are random vectors drawn independently from the uniform distribution on a set $A \subset \Delta_L$ with positive $(L-1)$-dimensional Lebesgue measure, then $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k, \boldsymbol{w}_{k+1}, \ldots, \boldsymbol{w}_L$ are linearly independent with probability* 1.

Proof We prove the result inductively. To begin, we prove the base case, i.e. $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k, \boldsymbol{w}_{k+1}$ are linearly independent w.p. 1. It suffices to show that $\boldsymbol{w}_{k+1} \notin \text{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$ w.p. 1. Thus, it is enough to show that $\text{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k) \cap A$ has $(L-1)$-dimensional Lebesgue measure 0. Since

$$\text{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k) \cap A \subset \text{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k) \cap \Delta_L,$$

it suffices to show that $\text{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k) \cap \Delta_L$ has $(L-1)$-dimensional Lebesgue measure 0.

Next, we claim that $\text{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k) \cap \Delta_L \subseteq \text{aff}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$. Let $\sum_{i=1}^k \alpha_i \boldsymbol{v}_i \in \Delta_L$. Since $\boldsymbol{v}_i \in \Delta_L$ for all $i \in [k]$, we can write $\boldsymbol{v}_i = \sum_{j=1}^L \beta_{i,j} \boldsymbol{e}_i$ where $\beta_{i,j} \geqslant 0$ and $\sum_{j=1}^L \beta_{i,j} = 1$. Then, since $\sum_{i=1}^k \alpha_i \boldsymbol{v}_i = \sum_{i=1}^k \alpha_i \sum_{j=1}^L \beta_{i,j} \boldsymbol{e}_j \in \Delta_L$, it holds that

$$1 = \sum_{i=1}^k \alpha_i \sum_{j=1}^L \beta_{i,j} = \sum_{i=1}^k \alpha_i.$$

Thus, $\sum_{i=1}^k \alpha_i \boldsymbol{v}_i \in \text{aff}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$, establishing the claim.

Thus, it suffices to show that $\text{aff}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$ has $(L-1)$-dimensional Lebesgue measure 0. $\text{aff}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$ has affine dimension at most $k-1$. Since it is not possible to fit a $(L-1)$ dimensional ball in an affine subspace of affine dimension $k-1 < L-1$, $\text{aff}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$ has $(L-1)$-dimensional Lebesgue measure 0. Thus, with probability 1, $\boldsymbol{w}_{k+1} \notin \text{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$. This establishes the base case.

The inductive step follows by a union bound and a similar argument to the base case. Thus, the result follows. ■

D.3. Demixing Mixed Membership Models

In this section, we prove our identification result for demixing mixed membership models, i.e., Theorem 2. First, we present technical lemmas in Section D.3.1. Second, in Section D.3.2, we present the key subroutine FaceTest and prove that it behaves as desired. Third, we prove Theorem 2 in Section D.3.3. Finally, in Section D.3.4, we extend our results to the nonsquare case (where $M > L$).

D.3.1. Lemmas

Lemma 4 establishes an intuitive continuity property of the two-sample version of κ^* and the residue. Recall that $\|\cdot\|$ denotes an arbitrary finite-dimensional norm on $\mathbb{R}^L$.

Lemma 4 *Let $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2 \in \Delta_L$ be distinct vectors and let $\boldsymbol{\mu}$ be the residue of $\boldsymbol{\eta}_2$ wrt $\boldsymbol{\eta}_1$. Let $\boldsymbol{\gamma}_n \in \Delta_L$ be a sequence such that $\|\boldsymbol{\gamma}_n - \boldsymbol{\eta}_2\| \longrightarrow 0$ as $n \longrightarrow \infty$, and let $\boldsymbol{\tau}_n$ be the residue of $\boldsymbol{\gamma}_n$ wrt $\boldsymbol{\eta}_1$. Then,*

1. $\lim_{n\longrightarrow\infty} \kappa^*(\boldsymbol{\gamma}_n \,|\, \boldsymbol{\eta}_1) = \kappa^*(\boldsymbol{\eta}_2 \,|\, \boldsymbol{\eta}_1)$, *and*
2. $\lim_{n\longrightarrow\infty} \|\boldsymbol{\tau}_n - \boldsymbol{\mu}\| = 0$.
3. *If, in addition, $\boldsymbol{\rho}_n \in \Delta_L$ is a sequence such that $\|\boldsymbol{\rho}_n - \boldsymbol{\eta}_2\| \longrightarrow 0$ as $n \longrightarrow \infty$ and $\mathcal{N}(\boldsymbol{\eta}_2) = \mathcal{N}(\boldsymbol{\gamma}_n) = \mathcal{N}(\boldsymbol{\rho}_n)$ for all n. Then, $\lim_{n\longrightarrow\infty} \kappa^*(\boldsymbol{\gamma}_n \,|\, \boldsymbol{\rho}_n) = 1$.*

Proof

1. In order to apply the residue operator κ^* to $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2, \boldsymbol{\gamma}_n$ we think of $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2, \boldsymbol{\gamma}_n$ as discrete probability distributions. By Proposition 2,

$$\kappa^*(\boldsymbol{\eta}_2 \,|\, \boldsymbol{\eta}_1) = \min_{i,\, \eta_{1,i}>0} \frac{\eta_{2,i}}{\eta_{1,i}}.$$

   Clearly, there is a constant $\delta > 0$ such that $\min_{i,\, \boldsymbol{\eta}_{1,i}>0} \boldsymbol{\eta}_{1,i} > \delta$. Let $\epsilon > 0$. By the equivalence of norms on finite-dimensional vector spaces, there exists a constant $C > 0$ such that $\|\cdot\|_\infty \leqslant C\,\|\cdot\|$ where $\|\cdot\|_\infty$ denotes the supremum norm. Thus, since $\|\boldsymbol{\gamma}_n - \boldsymbol{\eta}_2\| \longrightarrow 0$ as $n \longrightarrow \infty$, we can let n large enough such that $|\boldsymbol{\gamma}_{n,i} - \boldsymbol{\eta}_{2,i}| \leqslant \epsilon$ for all $i \in [L]$. Then,

$$\begin{aligned}\kappa^*(\boldsymbol{\gamma}_n \,|\, \boldsymbol{\eta}_1) = \min_{i,\eta_{1,i}>0} \frac{\gamma_{n,i}}{\eta_{1,i}} &\leqslant \min_{i,\eta_{1,i}>0} \frac{\eta_{2,i}+\epsilon}{\eta_{1,i}} \\ &\leqslant \kappa^*(\boldsymbol{\eta}_2 \,|\, \boldsymbol{\eta}_1) + \frac{\epsilon}{\delta}.\end{aligned}$$

   Similarly,

$$\begin{aligned}\kappa^*(\boldsymbol{\gamma}_n \,|\, \boldsymbol{\eta}_1) &= \min_{i,\eta_{1,i}>0} \frac{\gamma_{n,i}}{\eta_{1,i}} \\ &\geqslant \frac{\eta_{2,i}-\epsilon}{\eta_{1,i}} \\ &\geqslant \kappa^*(\boldsymbol{\eta}_2 \,|\, \boldsymbol{\eta}_1) - \frac{\epsilon}{\delta}.\end{aligned}$$

   Since $\epsilon > 0$ was arbitrary, statement *1* follows.

2. Write $\boldsymbol{\mu} = \kappa\boldsymbol{\eta}_2 + (1-\kappa)\boldsymbol{\eta}_1$ and $\boldsymbol{\tau}_n = \kappa_n\boldsymbol{\gamma}_n + (1-\kappa_n)\boldsymbol{\eta}_1$ where $\kappa = \kappa^*(\boldsymbol{\eta}_2 \,|\, \boldsymbol{\eta}_1)$ and $\kappa_n = \kappa^*(\boldsymbol{\gamma}_n \,|\, \boldsymbol{\eta}_1)$. Then, by the triangle inequality,

$$\begin{aligned}\|\boldsymbol{\mu} - \boldsymbol{\tau}_n\| &\leqslant \|\kappa\boldsymbol{\eta}_2 - \kappa_n\boldsymbol{\gamma}_n\| + \|(1-\kappa)\boldsymbol{\eta}_1 - (1-\kappa_n)\boldsymbol{\eta}_1\| \\ &\leqslant |\kappa - \kappa_n|\,\|\boldsymbol{\eta}_2\| + |\kappa_n|\,\|\boldsymbol{\eta}_2 - \boldsymbol{\gamma}_n\| + |\kappa - \kappa_n|\,\|\boldsymbol{\eta}_1\| \longrightarrow 0\end{aligned}$$

   as $n \longrightarrow \infty$.

3. W.l.o.g., suppose that $[K] = \mathcal{N}(\boldsymbol{\eta}_2) = \mathcal{N}(\boldsymbol{\gamma}_n) = \mathcal{N}(\boldsymbol{\rho}_n)$ where $K \leqslant L$. Observe that

$$\kappa^*(\boldsymbol{\gamma}_n \,|\, \boldsymbol{\rho}_n) = \min_{i,\, \rho_{n,i}>0} \frac{\gamma_{n,i}}{\rho_{n,i}} = \min_{i\in[K]} \frac{\gamma_{n,i}}{\rho_{n,i}}$$

   There exists a constant $\delta > 0$ such that $\min_{i\in[K]} \eta_{2,i} \geqslant \delta$. Let $\delta > \epsilon > 0$. By the equivalence of norms on finite-dimensional vector spaces, we can let n large enough such that $|\gamma_{n,i} - \eta_{2,i}| \leqslant \epsilon$ and $|\rho_{n,i} - \eta_{2,i}| \leqslant \epsilon$ for all $i \in [L]$. Then,

$$\begin{aligned}\kappa^*(\boldsymbol{\gamma}_n \,|\, \boldsymbol{\rho}_n) &= \min_{i\in[K]} \frac{\gamma_{n,i}}{\rho_{n,i}} \\ &\leqslant \min_{i\in[K]} \frac{\eta_{2,i}+\epsilon}{\eta_{2,i}-\epsilon} \\ &\leqslant \frac{\eta_{2,i}+\epsilon}{\eta_{2,i}-\epsilon}\end{aligned}$$

   for any $i \in [K]$, which goes to 1 as $\epsilon \longrightarrow 0$. Similarly,

$$\begin{aligned}\kappa^*(\boldsymbol{\gamma}_n \,|\, \boldsymbol{\rho}_n) &= \min_{i\in[K]} \frac{\gamma_{n,i}}{\rho_{n,i}} \\ &\geqslant \min_{i\in[K]} \frac{\eta_{2,i}-\epsilon}{\eta_{2,i}+\epsilon}\end{aligned}$$

   Since for any $i \in [K]$,

$$\frac{\eta_{2,i}-\epsilon}{\eta_{2,i}+\epsilon} \longrightarrow 1$$

   as $\epsilon \longrightarrow 0$, the above lower bound goes to 1 as $\epsilon \longrightarrow 0$. Thus, statement *3* follows.

■

Lemma 5 guarantees that certain operations in the Demix algorithm preserve linear independence of the mixture proportions. The proof uses tools from multilinear algebra.

Lemma 5 *Let $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_K \in \Delta_K$ be linearly independent and $P_1, \ldots, P_K$ be jointly irreducible. Let $Q_i = \boldsymbol{\tau}_i^T \boldsymbol{P}$ for $i \in [K]$. Then for any $i, j \in [K]$ such that $i \neq j$,*

1. *If $\boldsymbol{\eta} = \sum_{k=1}^K a_k \boldsymbol{\tau}_k$ with $a_j \neq 0$, then $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_{j-1}, \boldsymbol{\eta}, \boldsymbol{\tau}_{j+1}, \ldots, \boldsymbol{\tau}_K$ are linearly independent.*
2. *Let R_k be the residue of Q_k with respect to Q_j for all $k \in [K]\backslash\{j\}$. Then, $R_k = \boldsymbol{\eta}_k^T \boldsymbol{P}$ where $\boldsymbol{\eta}_k \in \Delta_L$ and $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_{j-1}, \boldsymbol{\tau}_j, \boldsymbol{\eta}_{j+1}, \ldots, \boldsymbol{\eta}_K$ are linearly independent.*
3. *Let $\boldsymbol{\tau}^* \in \operatorname{conv}(\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_k)^\circ$ and $\boldsymbol{\eta}_i \in \operatorname{conv}(\boldsymbol{\tau}_i, \boldsymbol{\tau}^*)^\circ$ for $i \in [k]$ where $k \leqslant K$. Then,*

$$\boldsymbol{\eta}_1, \boldsymbol{\eta}_2, \ldots, \boldsymbol{\eta}_k, \boldsymbol{\tau}_{k+1}, \ldots, \boldsymbol{\tau}_K$$

   are linearly independent.

Proof We use the multilinear expansion and usual properties of determinants.

1. Viewing each $\boldsymbol{\tau}_i$ as a column vector,
$$\det(\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_{j-1}, \sum_k a_k \boldsymbol{\tau}_k, \boldsymbol{\tau}_{j+1}, \ldots, \boldsymbol{\tau}_K) = a_j \det(\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_K) \neq 0.$$

2. Linear independence of $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_K$ implies that the $Q_1, \ldots, Q_K$ are distinct. Hence, by Proposition 2, we can write $\boldsymbol{\eta}_k = (1-\alpha_k)\boldsymbol{\tau}_j + \alpha_k \boldsymbol{\tau}_k$ where $\alpha_k \neq 0 \, \forall k \neq j$. Then, it holds that
$$\det(\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_{j-1}, \boldsymbol{\tau}_j, \boldsymbol{\eta}_{j+1}, \ldots, \boldsymbol{\eta}_K) = \left(\prod_{i \neq j} \alpha_i\right) \det(\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_K) \neq 0.$$

3. Since $\boldsymbol{\eta}_j = (1-\alpha_j)\boldsymbol{\tau}^* + \alpha_j \boldsymbol{\tau}_j$ where $\alpha_j \in (0,1)$ for all $j \leqslant k$, and $\boldsymbol{\tau}^* = \sum_i \beta_i \boldsymbol{\tau}_i$, it holds
$$\det(\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_{k-1}, \boldsymbol{\tau}_k, \ldots, \boldsymbol{\tau}_K) = \left(1 + \sum_{j=1}^{k} \frac{(1-\alpha_j)}{\alpha_j} \beta_j\right) \left(\prod_{i=1}^{k} \alpha_i\right) \det(\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_K) \neq 0.$$

■

Lemma 6 gives a condition on the mixture proportions under which the multi-sample residue is unique. Lemma 2 in Blanchard and Scott (2014) is very similar and is proved in a very similar way. We give a useful generalization here that reproduces many of the same details.

Lemma 6 *Let $l, k \in [L]$. Let $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_L \in \Delta_L$ be linearly independent. We have that condition 1 implies condition 2 and condition 2 implies condition 3.*

1. *There exists a decomposition*
$$\boldsymbol{\tau}_l = \kappa \boldsymbol{e}_k + (1-\kappa)\boldsymbol{\tau}_l'$$
where $\kappa > 0$ and $\boldsymbol{\tau}_l' \in \mathrm{conv}(\{\boldsymbol{\tau}_j : j \neq l\})$. Further, for every $\boldsymbol{e}_i$ such that $i \neq k$, there exists a decomposition
$$\boldsymbol{e}_i = \sum_{j=1}^{L} a_j \boldsymbol{\tau}_j$$
such that $a_l < \frac{1}{\kappa}$.

2. *Let*
$$\boldsymbol{T} = \begin{pmatrix} \boldsymbol{\tau}_1^T \\ \vdots \\ \boldsymbol{\tau}_L^T \end{pmatrix};$$
the matrix $\boldsymbol{T}$ is invertible and $\boldsymbol{T}^{-1}$ is such that $(\boldsymbol{T}^{-1})_{l,k} > 0$ and $(\boldsymbol{T}^{-1})_{l,i} \leqslant 0$ for $i \neq k$ and $(\boldsymbol{T}^{-1})_{l,k} > (\boldsymbol{T}^{-1})_{j,k}$ for $j \neq l$. In words, the (l,k)th entry in $\boldsymbol{T}^{-1}$ is positive, every other entry in the lth row of $\boldsymbol{T}^{-1}$ is nonpositive and every other entry in the kth column of $\boldsymbol{T}^{-1}$ is strictly less than the (l,k)th entry. [2]

2. $(\boldsymbol{T}^{-1})_{i,j}$ is the $i \times j$ entry in the matrix $\boldsymbol{T}^{-1}$.

3. *The residue of $\boldsymbol{\tau}_l$ with respect to $\{\boldsymbol{\tau}_j, j \neq l\}$ is $\boldsymbol{e}_k$.*

Proof Without loss of generality, let $l = 1$ and $k = 2$. By relabeling the vectors $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_L$, we can assume without loss of generality that $k = 1$. First, we show that condition *1* implies condition *2*. Suppose that condition *1* holds. Then, there exists $\kappa > 0$ such that
$$\boldsymbol{\tau}_1 = \kappa \boldsymbol{e}_1 + (1-\kappa) \sum_{i=2}^{L} \mu_i \boldsymbol{\tau}_i$$
with $\mu_i \geqslant 0$ for $i \in [L] \backslash \{1\}$. Then,
$$\boldsymbol{e}_1 = \frac{1}{\kappa}(\boldsymbol{\tau}_1 - \sum_{i \geqslant 2} (1-\kappa)\mu_i \boldsymbol{\tau}_i).$$
Hence, the first row of $\boldsymbol{T}^{-1}$ is given by $\frac{1}{\kappa}(1, -(1-\kappa)\mu_2, \cdots, -(1-\kappa)\mu_L)$. This shows that the first row is such that $(\boldsymbol{T}^{-1})_{1,1} > 0$ and $(\boldsymbol{T}^{-1})_{1,i} \leqslant 0$ for $i \neq 1$.

Consider $\boldsymbol{e}_i$ such that $i \neq 1$. Then, we have the relation: $\boldsymbol{e}_i = \sum_{j=1}^{L} a_j \boldsymbol{\tau}_j$, which gives the ith row of $\boldsymbol{T}^{-1}$. By assumption, $a_1 < \frac{1}{\kappa}$, so the $(i,1)$th entry is strictly less than the $(1,1)$th entry. Hence, *2* follows.

Now, we prove that condition *2* implies condition *3*. Suppose condition *2* is true. Consider the optimization problem
$$\max_{\boldsymbol{\nu}, \boldsymbol{\gamma}} \sum_{i=2}^{L} \nu_i \ s.t. \ \boldsymbol{\tau}_1 = (1 - \sum_{i \geqslant 2} \nu_i)\boldsymbol{\gamma} + \sum_{i=2}^{L} \nu_i \boldsymbol{\tau}_i$$
over $\boldsymbol{\gamma} \in \Delta_L$ and $\boldsymbol{\nu} = (\nu_2, \cdots, \nu_L) \in C_{L-1} = \{(\nu_2, \cdots, \nu_L) : \nu_i \geqslant 0; \sum_{i=2}^{L} \nu \leqslant 1\}$.

By the same argument given in the proof of Lemma 2 of Blanchard and Scott (2014), this optimization problem is equivalent to the program
$$\max_{\boldsymbol{\gamma} \in \Delta_L} \boldsymbol{e}_1^T (\boldsymbol{T}^T)^{-1} \boldsymbol{\gamma} \ s.t. \ \boldsymbol{\nu}((\boldsymbol{T}^T)^{-1}\boldsymbol{\gamma}) \in C_{L-1}$$
where $\boldsymbol{\nu}(\boldsymbol{\eta}) \coloneqq \eta_1^{-1}(-\eta_2, \cdots, -\eta_L)$. The above objective is of the form $\boldsymbol{a}^T \boldsymbol{\gamma}$ where $\boldsymbol{a}$ is the first column of $\boldsymbol{T}^{-1}$. Since $l = 1$, by assumption, for every $i \neq 1$, $\boldsymbol{T}_{1,1}^{-1} > \boldsymbol{T}_{i,1}^{-1}$. Therefore, the unconstrained maximum over $\boldsymbol{\gamma} \in \Delta_L$ is attained uniquely by $\boldsymbol{\gamma} = \boldsymbol{e}_1$. Notice that $(\boldsymbol{T}^T)^{-1}\boldsymbol{e}_1$ is the first row of $\boldsymbol{T}^{-1}$. Denote this vector $\boldsymbol{b} = (b_1, \cdots, b_L)$. We show that $\boldsymbol{\nu}(\boldsymbol{b}) = b_1^{-1}(-b_2, \cdots, -b_L) \in C_{L-1}$. By assumption, $\boldsymbol{b}$ has its first coordinate positive and the other coordinates are nonpositive. Therefore, all of the components of $\boldsymbol{\nu}(\boldsymbol{b})$ are nonnegative. Furthermore, the sum of the components of $\boldsymbol{\nu}(\boldsymbol{b})$ is
$$\sum_{i=2}^{L} \frac{-b_i}{b_1} = 1 - \frac{\sum_{i=1}^{L} b_i}{b_1} = 1 - \frac{1}{b_1} \leqslant 1.$$
The last equality follows because the rows of $\boldsymbol{T}^{-1}$ sum to 1 since $\boldsymbol{T}$ is a stochastic matrix. Then, we have $\boldsymbol{\nu}((\boldsymbol{T}^T)^{-1}\boldsymbol{e}_1) \in C_{L-1}$. Consequently, the unique maximum of the optimization problem is attained for $\boldsymbol{\gamma} = \boldsymbol{e}_1$. This establishes *3*. ■

D.3.2. The FaceTest Algorithm

Next, we consider the main subroutine in the Demix algorithm: the FaceTest algorithm (see Algorithm 6). Proposition 8 establishes that FaceTest$(Q_1, \ldots, Q_K)$ returns 1 if and only if $Q_1, \ldots, Q_K$ are in the relative interior of the same face of the simplex.

Proposition 8 *Let $Q_j = \boldsymbol{\eta}_j^T \boldsymbol{P}$ for $\boldsymbol{\eta}_j \in \Delta_K$ and all $j \in [K]$. Let $P_1, \ldots, P_K$ be jointly irreducible, $Q_1, \ldots, Q_K \in \mathrm{conv}(P_1, \ldots, P_K)$ be distinct, and for each $i \in [K]$, let $\boldsymbol{\eta}_i$ lie in the relative interior of one of the faces of Δ_K. FaceTest$(Q_1, \ldots, Q_K)$ returns 1 if and only if $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_K$ lie in the relative interior of the same face of Δ_K.*

Proof Suppose that $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_K$ lie on the relative interior of the same face of Δ_K. Then, $\mathcal{N}(Q_1) = \ldots = \mathcal{N}(Q_K)$. By statement 3 of Lemma 2, $\kappa^*(Q_i \,|\, Q_j) > 0$ for all $i \neq j$. Hence, FaceTest$(Q_1, \ldots, Q_K)$ returns 1.

Suppose that $Q_1, \ldots, Q_K$ do not all lie on the relative interior of the same face. Then, there exists Q_i, Q_j $(i \neq j)$ that do not lie on the relative interior of the same face. Without loss of generality, suppose that $\mathcal{N}(Q_j) \nsubseteq \mathcal{N}(Q_i)$. Then, by statement 3 of Lemma 2, $\kappa^*(Q_i \,|\, Q_j) = 0$. Hence, FaceTest$(Q_1, \ldots, Q_K)$ returns 0. ■

D.3.3. The Demix Algorithm

Proof [Proof of Theorem 2]

Let $K \leqslant L$, $\boldsymbol{\gamma}_i \in \Delta_L$ for all $i \in [K]$, $S_i = \boldsymbol{\gamma}_i^T \boldsymbol{P}$ for all $i \in [K]$, and

$$\boldsymbol{\Gamma} = \begin{pmatrix} \boldsymbol{\gamma}_1^T \\ \vdots \\ \boldsymbol{\gamma}_K^T \end{pmatrix}.$$

We claim that for any $\{i_1, \ldots, i_K\} \subset [L]$ and $\{S_1, \ldots, S_K\} \subset \mathrm{conv}(P_{i_1}, \ldots, P_{i_K})$, if $P_1, \ldots, P_L$ are jointly irreducible, and $\boldsymbol{\Gamma}$ has full row rank, then w.p. 1 Demix$(S_1, \ldots, S_K)$ returns a permutation of $(P_{i_1}, \ldots, P_{i_K})$. If the claim holds, then setting $K = L$ and putting $\tilde{P}_i = S_i$ yields the result. We prove the claim by induction on K.

Consider the base case: $K = 2$. Suppose that $\{S_1, S_2\} \subset \mathrm{conv}(P_1, P_2)$ (the other cases are similar). Note that $\boldsymbol{\gamma}_1 \neq \boldsymbol{\gamma}_2$ by linear independence of $\boldsymbol{\gamma}_1$ and $\boldsymbol{\gamma}_2$. Either $\boldsymbol{\gamma}_1 \in \mathrm{conv}(\boldsymbol{e}_1, \boldsymbol{\gamma}_2)$ or $\boldsymbol{\gamma}_1 \in \mathrm{conv}(\boldsymbol{e}_2, \boldsymbol{\gamma}_2)$. Suppose $\boldsymbol{\gamma}_1 \in \mathrm{conv}(\boldsymbol{e}_1, \boldsymbol{\gamma}_2)$. Condition *2* of Lemma 1 is satisfied so that $\boldsymbol{e}_1$ is the residue of $\boldsymbol{\gamma}_1$ with respect to $\boldsymbol{\gamma}_2$ and $\boldsymbol{e}_2$ is the residue of $\boldsymbol{\gamma}_2$ with respect to $\boldsymbol{\gamma}_1$. Thus, by statement *3* of Proposition 3, P_1 is the residue of S_1 with respect to S_2 and P_2 is the residue of S_2 with respect to S_1. If $\boldsymbol{\gamma}_1 \in \mathrm{conv}(\boldsymbol{e}_2, \boldsymbol{\gamma}_2)$, then similar reasoning establishes that P_2 is the residue of S_1 with respect to S_2 and P_1 is the residue of S_2 with respect to S_1. Thus, the base case follows.

Suppose $L \geqslant K > 2$. The inductive hypothesis is:

Inductive Hypothesis: for any $\{i_1, \ldots, i_{K-1}\} \subset [L]$ and $\{S_1, \ldots, S_{K-1}\} \subset \mathrm{conv}(P_{i_1}, \ldots, P_{i_{K-1}})$, if $P_1, \ldots, P_L$ are jointly irreducible and $\boldsymbol{\Gamma}$ has full row rank, then w.p. 1 Demix$(S_1, \ldots, S_{K-1})$ returns a permutation of $(P_{i_1}, \ldots, P_{i_{K-1}})$.

Suppose that $\{S_1, \ldots, S_K\} \subset \mathrm{conv}(P_1, \ldots, P_K)$ (the other cases are similar). Set $\Xi = \mathrm{conv}(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_K)$. With probability 1, $Q \in \mathrm{conv}(S_2, \ldots, S_K)^\circ$. We can write $Q = \boldsymbol{\eta}^T \boldsymbol{P}$ where $\boldsymbol{\eta}$ is a uniformly distributed random vector in $\mathrm{conv}(\boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_K)$. Let R be the residue of Q with respect to S_1. By statement *3* of Proposition 3, we can write $R = \boldsymbol{\lambda}^T \boldsymbol{P}$ where $\boldsymbol{\lambda}$ is the residue of $\boldsymbol{\eta}$ with respect to $\boldsymbol{\gamma}_1$. By statement *2* of Lemma 2, $\boldsymbol{\lambda} \in \partial\Xi$.

Step 1: We claim that with probability 1, there is $l \in [K]$ such that $\boldsymbol{\lambda} \in \mathrm{conv}(\{\boldsymbol{e}_j : j \in [K]\backslash\{l\}\})^\circ$. Let $B_{i,j} = \mathrm{conv}(\{\boldsymbol{\gamma}_1\} \cup \{\boldsymbol{e}_k : k \in [K]\backslash\{i,j\}\})$ where $i, j \in [K]$ and $i \neq j$ and let $C = \mathrm{conv}(\boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_K)$. First, we argue that $C \cap B_{i,j}$ has affine dimension at most $K - 3$.[3] Since $\boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_K$ are linearly independent, C has affine dimension $K - 2$. Since $\{\boldsymbol{e}_k : k \in [K]\backslash\{i,j\}\}$ are linearly independent, $B_{i,j}$ has affine dimension $K - 2$ or $K - 3$. If $B_{i,j}$ has affine dimension $K - 3$, then $C \cap B_{i,j}$ has affine dimension at most $K - 3$. So, suppose that $B_{i,j}$ has affine dimension $K - 2$. If $C \cap B_{i,j}$ has affine dimension $K - 2$, then $\mathrm{aff}\, C = \mathrm{aff}\, B_{i,j}$. Then, in particular, $\boldsymbol{\gamma}_1 \in \mathrm{aff} C$. But, this contradicts the linear independence of $\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_K$. Therefore, $C \cap B_{i,j}$ has affine dimension at most $K - 3$.

Because C has affine dimension $K - 2$ and $\boldsymbol{\eta}$ is a uniformly distributed random vector in C, with probability 1, $\boldsymbol{\eta} \notin \cup_{i,j \in [K], i \neq j} B_{i,j}$. Since $\boldsymbol{\gamma}_1 \in B_{i,j}$ for all $i, j \in [K]$ and $\boldsymbol{\eta} \in \mathrm{conv}(\boldsymbol{\lambda}, \boldsymbol{\gamma}_1)$ by definition, the convexity of $B_{i,j}$ implies that $\boldsymbol{\lambda} \notin \cup_{i,j \in [K], i \neq j} B_{i,j}$. Since $\boldsymbol{\lambda} \in \partial\Xi$, the claim follows.

Step 2: Let $R_i^{(n)}$ be the residue of $m_{\frac{n-1}{n}}(S_i, Q)$ with respect to S_1. We claim that there is some finite integer $N \geqslant 2$ such that for all $n \geqslant N$,

$$\text{FaceTest}(R_2^{(n)}, \ldots, R_K^{(n)})$$

returns 1. By Proposition 8, this is equivalent to the statement that there exists $N \geqslant 2$ such that for all $n \geqslant N$, the mixture proportions of $R_2^{(n)}, \ldots, R_K^{(n)}$ are on the relative interior of the same face. Let $m_{\frac{n-1}{n}}(S_i, Q) = (\boldsymbol{\tau}_i^{(n)})^T \boldsymbol{P}$ for $i \in [K]\backslash\{1\}$; note that $\boldsymbol{\tau}_i^{(n)} = \frac{1}{n}\boldsymbol{\gamma}_i + \frac{n-1}{n}\boldsymbol{\eta}$ and, consequently, $\boldsymbol{\tau}_i^{(n)} \in \Xi$. Since $\boldsymbol{\eta} \in \mathrm{conv}(\boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_K)^\circ$ with probability 1, $\boldsymbol{\tau}_i^{(n)} \in \mathrm{conv}(\boldsymbol{\gamma}_i, \boldsymbol{\eta})^\circ$ for all $i \in [K]\backslash\{1\}$ and $n \in \mathbb{N}$, and $\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_K$ are linearly independent, it follows that for all $n \in \mathbb{N}$ with probability 1, $\boldsymbol{\gamma}_1, \boldsymbol{\tau}_2^{(n)}, \ldots, \boldsymbol{\tau}_K^{(n)}$ are linearly independent by statement *3* in Lemma 5. Fix $i \in [K]\backslash\{1\}$. It suffices to show that there is large enough N such that for $n \geqslant N$, Residue$(m_{\frac{n-1}{n}}(S_i, Q) \,|\, S_1) = R_i^{(n)}$ is on the same face as R. Let $R_i^{(n)} = (\boldsymbol{\mu}_i^{(n)})^T \boldsymbol{P}$; by statement *3* of Proposition 3, $\boldsymbol{\mu}_i^{(n)}$ is the residue of $\boldsymbol{\tau}_i^{(n)}$ with respect to $\boldsymbol{\gamma}_1$ and by statement 2 of Lemma 2 $\boldsymbol{\mu}_i^{(n)} \in \Xi$. It suffices to show that $\mathcal{N}(\boldsymbol{\mu}_i^{(n)}) = \mathcal{N}(\boldsymbol{\lambda})$, i.e., every $\boldsymbol{\mu}_i^{(n)}$ is on the same face as $\boldsymbol{\lambda}$. As $n \longrightarrow \infty$, $\boldsymbol{\tau}_i^{(n)} = (1 - \frac{n-1}{n})\boldsymbol{\gamma}_i + \frac{n-1}{n}\boldsymbol{\eta} \longrightarrow \boldsymbol{\eta}$, hence by statement *2* in Lemma 4, $\left\|\boldsymbol{\mu}_i^{(n)} - \boldsymbol{\lambda}\right\| \longrightarrow 0$. Since with probability 1, $\boldsymbol{\lambda} \in \mathrm{conv}(\{\boldsymbol{e}_j : j \in [K]\backslash\{l\}\})^\circ$ for some l (step 1), it follows that for large enough n, $\boldsymbol{\mu}_i^{(n)} \in \mathrm{conv}(\{\boldsymbol{e}_j : j \in [K]\backslash\{l\}\})^\circ$.

3. Note that if $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \mathbb{R}^L$ are linearly independent and $n \leqslant L$, then $\mathrm{aff}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n)$ has affine dimension $n - 1$.

Algorithm 15 NonSquareDemix($\tilde{P}_1, \ldots, \tilde{P}_M$)

1: $R_1, \ldots, R_L \longleftarrow$ independently uniformly distributed elements in $\mathrm{conv}(\tilde{P}_1, \ldots, \tilde{P}_M)$
2: $(Q_1, \ldots, Q_L)^T \longleftarrow \mathrm{Demix}(R_1, \ldots, R_L)$
3: **return** $(Q_1, \ldots, Q_L)^T$

Step 3: Assume that n is sufficiently large such that $R_2^{(n)}, \ldots, R_K^{(n)}$ are on the same face. The algorithm recurses on $R_2^{(n)}, \ldots, R_K^{(n)}$. Since $\boldsymbol{\gamma}_1, \boldsymbol{\tau}_2^{(n)}, \ldots, \boldsymbol{\tau}_K^{(n)}$ are linearly independent, it follows by statement *2* in Lemma 5 that $\boldsymbol{\mu}_2^{(n)}, \ldots, \boldsymbol{\mu}_K^{(n)}$ are linearly independent. Suppose wlog that $\{R_2^{(n)}, \ldots, R_K^{(n)}\} \subset \{P_1, \ldots, P_{K-1}\}$. Then, by the inductive hypothesis, if $(Q_1, \ldots, Q_{K-1}) \longleftarrow \mathrm{Demix}(R_2^{(n)}, \ldots, R_K^{(n)})$, then $(Q_1, \ldots, Q_{K-1})$ is a permutation of $(P_1, \ldots, P_{K-1})$. Note that $\frac{1}{K}\sum_{i=1}^K S_i \in \mathrm{conv}(P_1, \ldots, P_K)^\circ$ since $\boldsymbol{\Gamma}$ has full rank by assumption.

Write $Q_i = \boldsymbol{\rho}_i^T \boldsymbol{P}$ for $i \in [K]$. Then, there exists of a permutation $\sigma : [K-1] \longrightarrow [K-1]$ such that $\boldsymbol{\rho}_i = \boldsymbol{e}_{\sigma(i)}$. Since $\boldsymbol{\rho}_K \in \Xi^\circ$ and $\boldsymbol{\rho}_i = \boldsymbol{e}_{\sigma(i)}$ for $i \leqslant K-1$, the conditions in statement *1* of Lemma 6 are satisfied. Therefore, by Lemma 6, the residue of $\boldsymbol{\rho}_K$ with respect to $\{\boldsymbol{\rho}_1, \ldots, \boldsymbol{\rho}_{K-1}\}$ is $\boldsymbol{e}_K$. Then, by statement *3* of Proposition 3, the residue of Q_K with respect to $\{Q_1, \ldots, Q_{K-1}\}$ is P_K. This completes the inductive step.

■

D.3.4. The Non-Square Demix Algorithm

Now, we examine the non-square case of the demixing problem ($M > L$). Note that knowledge of L is needed since one must resample exactly L distributions in order to run the square Demix algorithm.

Corollary 1 *Suppose $M > L$. Let $P_1, \ldots, P_L$ be jointly irreducible and $\boldsymbol{\Pi}$ have full rank. Then, with probability 1, NonSquareDemix($\tilde{P}_1, \ldots, \tilde{P}_M$) returns $(Q_1, \ldots, Q_L)$ such that $(Q_1, \ldots, Q_L)$ is a permutation of $(P_1, \ldots, P_L)$.*

Proof We can write $R_i = \boldsymbol{\tau}_i^T \boldsymbol{P}$ where $\boldsymbol{\tau}_i \in \Delta_L$ and $i = 1, \ldots, L$. $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_L$ are drawn uniformly independently from a set with positive $(L-1)$-dimensional Lebesgue measure since $\boldsymbol{\Pi}$ has full rank by hypothesis. By Lemma 3, $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_L$ are linearly independent with probability 1. Then, by Theorem 2, with probability 1, Demix($R_1, \ldots, R_L$) returns a permutation of $(P_1, \ldots, P_L)$. ■

D.4. Classification with Partial Labels

In this section, we present our identification result for classification with partial labels, i.e., Theorem 3. To begin, in Section D.4.1, we prove an important lemma for the main subroutine of the algorithm PartialLabel: VertexTest (algorithm 9). Second, in Section D.4.2, we present the proof of Theorem 3.

D.4.1. VertexTest Algorithm

Lemma 7 establishes that the VertexTest algorithm determines whether one vector of distributions is a permutation of another vector of distributions.

Lemma 7 *Let $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_L \in \Delta_L$ and $Q_i = \boldsymbol{\eta}_i^T \boldsymbol{P}$ for $i \in [L]$ and $\boldsymbol{Q} = (Q_1, \ldots, Q_L)^T$. Suppose that $P_1, \ldots, P_L$ are jointly irreducible, $\boldsymbol{\Pi}$ has full column rank, and the columns of $\boldsymbol{\Pi}^+$ are unique. Then, VertexTest($\boldsymbol{\Pi}^+, \tilde{\boldsymbol{P}}, \boldsymbol{Q}$) returns $(1, \boldsymbol{C}^T)$ with $\boldsymbol{C}$ a permutation matrix if and only if $\boldsymbol{Q}$ is a permutation of $\boldsymbol{P}$. Further, if VertexTest($\boldsymbol{\Pi}^+, \tilde{\boldsymbol{P}}, \boldsymbol{Q}$) returns $(1, \boldsymbol{C}^T)$, then $\boldsymbol{C}^T \boldsymbol{Q} = \boldsymbol{P}$.*

Proof If $\boldsymbol{Q} = (Q_1, \ldots, Q_L)^T$ is such that $\boldsymbol{D}^T \boldsymbol{Q} = \boldsymbol{P}$ where $\boldsymbol{D}$ is a permutation matrix, then it is clear that VertexTest($\boldsymbol{\Pi}^+, (\tilde{P}_1, \ldots, \tilde{P}_M)^T, (Q_1, \ldots, Q_L)^T$) returns $(1, \boldsymbol{C}^T)$ for some permutation matrix $\boldsymbol{C}$ since the entries of $\boldsymbol{\Pi}^+$ are $\boldsymbol{\Pi}^+_{i,j} = \mathbf{1}_{\{\kappa^*(\tilde{P}_i \,|\, P_j) > 0\}}$. Since $\boldsymbol{D}^T \boldsymbol{Q} = \boldsymbol{P}$, clearly, $\boldsymbol{Z}\boldsymbol{D} = \boldsymbol{\Pi}^+$. But since the columns of $\boldsymbol{\Pi}^+$ are unique, there is a unique permutation of the columns of $\boldsymbol{Z}$ to obtain the columns of $\boldsymbol{\Pi}^+$. Therefore, $\boldsymbol{D} = \boldsymbol{C}$.

Consider the "only if" direction. We use the notation from Algorithm 9. Suppose Algorithm 9 has returned $(1, \boldsymbol{C}^T)$ where $\boldsymbol{C}$ is a permutation matrix. W.l.o.g. (reordering the Q_i) we can assume that $\boldsymbol{C}$ is the identity and thus $\boldsymbol{Z} = \boldsymbol{\Pi}^+$.

In the sequel denote $\phi(x) := \mathbf{1}_{\{x>0\}}$ and $\phi(\boldsymbol{M})$ the entry-wise application of ϕ to the matrix or vector $\boldsymbol{M}$. We denote $\boldsymbol{v} \preceq \boldsymbol{w}$ when all entries of $\boldsymbol{v}$ are less than or equal to the corresponding entries of $\boldsymbol{w}$ (where $\boldsymbol{v}$ and $\boldsymbol{w}$ are vectors). This is a partial order, which will be used only for $0-1$ vectors below (essentially to denote support inclusion). W.l.o.g. (reordering the P_i) we can assume that the columns of $\boldsymbol{\Pi}^+$ are reordered in some sequence compatible with $\preceq$ in decreasing order, i.e. such that if $\boldsymbol{\Pi}^+_{:,j} \preceq \boldsymbol{\Pi}^+_{:,i}$, then $i \leqslant j$.

Introduce the following additional notation: let $\boldsymbol{\Lambda}$ be the matrix with rows $\boldsymbol{\Lambda}_{i,:} = \phi(\boldsymbol{\eta}_i^T)$. Observe that by statement 3 of Lemma 2, for any i, j, k, $\kappa^*(\tilde{P}_i | Q_j) > 0$ and $\kappa^*(Q_j | P_k) > 0$ implies $\kappa^*(\tilde{P}_i | P_k) > 0$. Note that we can write $\boldsymbol{\Lambda}_{j,k} = \mathbf{1}_{\{\kappa^*(Q_j \,|\, P_k) > 0\}}$ and $\boldsymbol{\Pi}^+_{j,k} = \mathbf{1}_{\{\kappa^*(\tilde{P}_j \,|\, P_k) > 0\}}$. Thus, we must have $\phi(\boldsymbol{Z}\boldsymbol{\Lambda}) \preceq \boldsymbol{\Pi}^+$.

We now argue that this implies that $\boldsymbol{\Lambda}$ is sub-diagonal, i.e., $\boldsymbol{\Lambda}_{ij} = 0$ for $i < j$. Let $i < j$. If $\boldsymbol{\Lambda}_{ij} > 0$, then $\boldsymbol{Z}_{:,i} \preceq \boldsymbol{\Pi}^+_{:,j}$ by the above relation. Since $\boldsymbol{Z} = \boldsymbol{\Pi}^+$, this implies $\boldsymbol{\Pi}^+_{:,i} \preceq \boldsymbol{\Pi}^+_{:,j}$, which implies $j \leqslant i$ by the assumed ordering of the columns of $\boldsymbol{\Pi}^+$, a contradiction. Hence $\boldsymbol{\Lambda}_{ij} = 0$ for $i < j$.

Now, since the matrix $\boldsymbol{Y}$ (line 1 of Algorithm 9) is diagonal, Statement 3 of Lemma 1 gives that for any $i \neq j$ we have $\boldsymbol{\Lambda}_{i,:} \not\preceq \boldsymbol{\Lambda}_{j,:}$. One can conclude by a straightforward recursion that since $\boldsymbol{\Lambda}$ is sub-diagonal, this implies that $\boldsymbol{\Lambda}$ is in fact diagonal. Start with the first row $\boldsymbol{\Lambda}_{1,:}$ which must be $(1, 0, \ldots, 0)^T$ (by sub-diagonality). Since $\boldsymbol{\Lambda}_{1,:} \not\preceq \boldsymbol{\Lambda}_{j,:}$ for $j > 1$, this implies the first column $\boldsymbol{\Lambda}_{:,1}$ is also $(1, 0, \ldots, 0)$. The subsequent columns/rows are handled in the same way.

Hence $\boldsymbol{\Lambda}$ is the identity, which implies that $\boldsymbol{Q} = \boldsymbol{P}$. ■

D.4.2. Proof of Theorem 3

Proof We adopt the notation from the description of Algorithm 7 with the exception that we make explicit the dependence on k by writing $W_i^{(k)}$ instead of W_i and $\bar{Q}_i^{(k)}$ instead of

$\bar{Q}_i$. We show that there is a K such that for all $k \geqslant K$, $(W_1^{(k)}, \ldots, W_L^{(k)})^T$ is a permutation of $(P_1, \ldots, P_L)^T$. Then, the result will follow from Lemma 7.

Let $Q_i = \boldsymbol{\tau}_i^T \boldsymbol{P}$, $\bar{Q}_i^{(k)} = \bar{\boldsymbol{\tau}}_i^{(k)^T} \boldsymbol{P}$, and $W_i^{(k)} = \boldsymbol{\gamma}_i^{(k)^T} \boldsymbol{P}$. Further, let $0 \leqslant n < L$, $\{i_1, \ldots, i_n\} \subset [L]$, $l \neq j \in [L]$, and define the following events wrt the randomness of $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_L$:

$$
\begin{aligned}
E_{i_1,\ldots,i_n} &= \{\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_n}, \boldsymbol{\tau}_{n+1}, \ldots, \boldsymbol{\tau}_L \text{ are linearly independent}\} \\
E &= \cap_{\{i_1,\ldots,i_n\} \subset [L], 0 \leqslant n < L} E_{i_1,\ldots,i_n} \\
F_{l,j} &= \{\boldsymbol{e}_l - \boldsymbol{e}_j, \boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_L \text{ are linearly independent}\} \\
F &= \cap_{l \neq j \in [L]} F_{l,j} \\
G^{l,j}_{i_1,\ldots,i_{n-1}} &= \{\boldsymbol{e}_l - \boldsymbol{e}_j, \boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}, \boldsymbol{\tau}_{n+1}, \ldots, \boldsymbol{\tau}_L \text{ are linearly independent}\} \\
G &= \cap_{\{i_1,\ldots,i_{n-1}\} \subset [L], 0 \leqslant n < L, l \neq j \in [L] \setminus \{i_1,\ldots,i_{n-1}\}} G^{l,j}_{i_1,\ldots,i_{n-1}}.
\end{aligned}
$$

By Lemma 3, for any $0 \leqslant n < L$, $\{i_1, \ldots, i_n\} \subset [L]$, the event $E_{i_1,\ldots,i_n}$ occurs with probability 1. Similarly, by Lemma 3, for any $l \neq j \in [L]$, the event $F_{l,j}$ occurs with probability 1. Finally, by Lemma 3, for any $0 \leqslant n < L$, $\{i_1, \ldots, i_{n-1}\} \subset [L]$ and any $l \neq j \in [L] \setminus \{i_1, \ldots, i_{n-1}\}$, the event $G^{l,j}_{i_1,\ldots,i_{n-1}}$ occurs with probability 1. Hence, the event $E \cap F \cap G$ occurs with probability 1. For the remainder of the proof, assume event $E \cap F \cap G$ occurs.

We prove the claim inductively. We show that for all $n \leqslant L$ there exists K_n such that if $k \geqslant K_n$, then $W_1^{(k)}, \ldots, W_n^{(k)}$ are distinct base distributions.

Base Case: $n = 1$. We will apply Lemma 6. By event E, $\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_L$ are linearly independent. Therefore, $\text{aff}(\boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_L)$ gives a hyperplane with an associated open halfspace $\boldsymbol{H}$ that contains $\boldsymbol{\tau}_1$ and at least one $\boldsymbol{e}_j$. Inspection of Line 3 of Algorithm 8 shows that $\bar{\boldsymbol{\tau}}_1^{(k)}$ is simply the average of $\boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_L$ and does not depend on k. Thus, there exists K_1 such that for all $k \geqslant K_1$, if $\boldsymbol{e}_j \in \boldsymbol{H}$, then $\boldsymbol{\lambda}_k \coloneqq \frac{1}{k}\boldsymbol{\tau}_1 + \frac{k-1}{k}\bar{\boldsymbol{\tau}}_1^{(k)} \in \text{conv}(\boldsymbol{e}_j, \boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_L)^\circ$. Fix $k \geqslant K_1$. Then, by event E, for all $\boldsymbol{e}_j \in \boldsymbol{H}$, there exists a unique $\kappa_j > 0$ and unique $a_{j,2}, \ldots, a_{j,L}$ such that

$$
\begin{aligned}
\boldsymbol{\lambda}_k &= \kappa_j \boldsymbol{e}_j + \sum_{i=2}^{L} a_{j,i} \boldsymbol{\tau}_i \\
&= \kappa_j \boldsymbol{e}_j + (1 - \kappa_j) \tilde{\boldsymbol{\tau}}_j
\end{aligned}
$$

where $\tilde{\boldsymbol{\tau}}_j \in \text{conv}(\boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_L)$ is unique. We claim that for all $i \neq j$ and $\{\boldsymbol{e}_i, \boldsymbol{e}_j\} \subset \boldsymbol{H}$, $\kappa_i \neq \kappa_j$. Suppose to the contrary that there is $i \neq j$ such that $\{\boldsymbol{e}_i, \boldsymbol{e}_j\} \subset \boldsymbol{H}$ and $\kappa_i = \kappa_j = \kappa$. Then,

$$
\begin{aligned}
\boldsymbol{\lambda}_k &= \kappa \boldsymbol{e}_i + (1 - \kappa) \tilde{\boldsymbol{\tau}}_i \\
\boldsymbol{\lambda}_k &= \kappa \boldsymbol{e}_j + (1 - \kappa) \tilde{\boldsymbol{\tau}}_j.
\end{aligned}
$$

Then, $(1-\kappa)(\tilde{\boldsymbol{\tau}}_j - \tilde{\boldsymbol{\tau}}_i) - \kappa(\boldsymbol{e}_i - \boldsymbol{e}_j) = 0$, from which it follows that $\boldsymbol{e}_i - \boldsymbol{e}_j \in \text{span}(\boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_L)$. But, by event F, $\boldsymbol{e}_i - \boldsymbol{e}_j, \boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_L$ are linearly independent and, hence, we have a contradiction. Thus, the claim follows.

Consequently, there is a unique j that minimizes κ_j. Note that for all $\boldsymbol{e}_i \notin \boldsymbol{H}$, if we write $\boldsymbol{e}_i = \sum_{l \geqslant 2} a_l \boldsymbol{\tau}_l + a_1 \boldsymbol{\lambda}_k$, then $a_1 \leqslant 0$. Then, by Lemma 6, $\boldsymbol{e}_j$ is the residue of $\boldsymbol{\lambda}_k$ with

respect to $\boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_L$. Therefore, by Proposition 3, $\text{MultiResidue}(\frac{1}{k}Q_1 + (1-\frac{1}{k})\bar{Q}_1 \,|\, \{Q_j\}_{j>1})$ is well-defined and if $W_1^{(k)} \longleftarrow \text{MultiResidue}(\frac{1}{k}Q_1 + (1-\frac{1}{k})\bar{Q}_1 \,|\, \{Q_j\}_{j>1})$, $W_1^{(k)}$ is one of the base distributions. This establishes the base case.

The Inductive Step: The proof is similar to the base case. Suppose that there exists K_{n-1} such that for all $k \geqslant K_{n-1}$, $W_1^{(k)}, \ldots, W_{n-1}^{(k)}$ are distinct base distributions. Let $\{i_1, \ldots, i_{n-1}\} \subset [L]$ denote the indices of the base distributions that are equal to $W_1^{(k)}, \ldots, W_{n-1}^{(k)}$ under the inductive hypothesis. By the event E, $\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}, \boldsymbol{\tau}_n, \ldots, \boldsymbol{\tau}_L$ are linearly independent. Hence, $\text{aff}(\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}, \boldsymbol{\tau}_{n+1}, \ldots, \boldsymbol{\tau}_L)$ gives a hyperplane with an associated open halfspace $\boldsymbol{H}_{i_1,\ldots,i_{n-1}}$ such that $\boldsymbol{\tau}_n \in \boldsymbol{H}_{i_1,\ldots,i_{n-1}}$. We claim that there is $\boldsymbol{e}_j \notin \{\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}\}$ such that $\boldsymbol{e}_j \in \boldsymbol{H}_{i_1,\ldots,i_{n-1}}$. Suppose not. Then, $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_L \in \boldsymbol{H}^c_{i_1,\ldots,i_{n-1}}$ and $\boldsymbol{\tau}_n \in \boldsymbol{H}_{i_1,\ldots,i_{n-1}}$, which implies that $\boldsymbol{\tau}_n \notin \Delta_{L-1}$. This is a contradiction, so the claim follows.

Define

$$
\boldsymbol{\lambda}_k^{(i_1,\ldots,i_{n-1})} \coloneqq \frac{1}{k}\boldsymbol{\tau}_n + [\frac{k-1}{k}]\frac{1}{L-1}(\sum_{s>n} \boldsymbol{\tau}_s + \sum_{s<n} \boldsymbol{e}_{i_s}).
$$

There exists an integer $K_n^{(i_1,\ldots,i_{n-1})}$ such that if $k \geqslant K_n^{(i_1,\ldots,i_{n-1})}$, then for all $\boldsymbol{e}_j \in \boldsymbol{H}_{i_1,\ldots,i_{n-1}}$, $\boldsymbol{\lambda}_k^{(i_1,\ldots,i_{n-1})} \in \text{conv}(\boldsymbol{e}_j, \boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}, \boldsymbol{\tau}_{n+1}, \ldots, \boldsymbol{\tau}_L)^\circ$. Set

$$
K_n \coloneqq \max(\max_{\{i_1,\ldots,i_{n-1}\} \subset [L]} (K_n^{(i_1,\ldots,i_{n-1})}), K_{n-1}).
$$

Fix $k \geqslant K_n$. Define

$$
\begin{aligned}
\boldsymbol{\lambda}_k &\coloneqq \frac{1}{k}\boldsymbol{\tau}_n + [\frac{k-1}{k}]\bar{\boldsymbol{\tau}}_n^{(k)} \\
&= \frac{1}{k}\boldsymbol{\tau}_n + [\frac{k-1}{k}]\frac{1}{L-1}(\sum_{s>n} \boldsymbol{\tau}_s + \sum_{s<n} \boldsymbol{\gamma}_s^{(k)}).
\end{aligned}
$$

By the inductive hypothesis, $k \geqslant K_{n-1}$, and Proposition 3, there exists $\{i_1, \ldots, i_{n-1}\} \subset [L]$ such that $\boldsymbol{\gamma}_j^{(k)} = \boldsymbol{e}_{i_j}$ for all $j \in [n-1]$. For the sake of abbreviation, let $\boldsymbol{H} = \boldsymbol{H}_{i_1,\ldots,i_{n-1}}$. Thus, $\boldsymbol{\tau}_n \in \boldsymbol{H}$ and there exists $\boldsymbol{e}_j \in \boldsymbol{H}$ such that $\boldsymbol{e}_j \notin \{\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}\}$. Hence, by our choice of K_n, for every $\boldsymbol{e}_j \in \boldsymbol{H}$

$$
\boldsymbol{\lambda}_k = \boldsymbol{\lambda}_k^{(i_1,\ldots,i_{n-1})} \in \text{conv}(\boldsymbol{e}_j, \boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}, \boldsymbol{\tau}_{n+1}, \ldots, \boldsymbol{\tau}_L)^\circ.
$$

By event E for all $\boldsymbol{e}_j \in \boldsymbol{H}$, there is a unique $\kappa_j > 0$ and unique $a_{j,1}, \ldots, a_{j,n-1}, a_{j,n+1}, \ldots, a_{j,L} > 0$ such that

$$
\begin{aligned}
\boldsymbol{\lambda}_k &= \kappa_j \boldsymbol{e}_j + \sum_{l<n} a_{j,i} \boldsymbol{e}_{i_l} + \sum_{l>n} a_{j,l} \boldsymbol{\tau}_l \\
&= \kappa_j \boldsymbol{e}_j + (1 - \kappa_j) \tilde{\boldsymbol{\tau}}_j
\end{aligned}
$$

where $\tilde{\boldsymbol{\tau}}_j \in \text{conv}(\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}, \boldsymbol{\tau}_{n+1}, \ldots, \boldsymbol{\tau}_L)$ is unique.

We claim that for all $l \neq j$ such that $\{\boldsymbol{e}_l, \boldsymbol{e}_j\} \subset \boldsymbol{H}$, $\kappa_l \neq \kappa_j$. Suppose to the contrary that there exists $l \neq j$ such that $\{\boldsymbol{e}_l, \boldsymbol{e}_j\} \subset \boldsymbol{H}$ and $\kappa_l = \kappa_j = \kappa$. Then,

$$\boldsymbol{\lambda}_k = \kappa \boldsymbol{e}_l + (1-\kappa)\tilde{\boldsymbol{\tau}}_i = \kappa \boldsymbol{e}_j + (1-\kappa)\tilde{\boldsymbol{\tau}}_j.$$

This implies that $\boldsymbol{e}_l - \boldsymbol{e}_j \in \text{span}(\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}, \boldsymbol{\tau}_{n+1}, \ldots, \boldsymbol{\tau}_L)$. Observe that $\{\boldsymbol{e}_l, \boldsymbol{e}_j\} \subset \boldsymbol{H}$ implies that $\boldsymbol{e}_l \notin \{\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}\}$ and $\boldsymbol{e}_j \notin \{\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}\}$. Thus, event G implies that $\boldsymbol{e}_l - \boldsymbol{e}_j, \boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}, \boldsymbol{\tau}_{n+1}, \ldots, \boldsymbol{\tau}_L$ are linearly independent. Therefore, we have a contradiction, establishing the claim.

Consequently, there is a unique j that minimizes κ_j. Note that for all $\boldsymbol{e}_l \notin \boldsymbol{H}$, if we write $\boldsymbol{e}_l = \sum_{m<n} a_m \boldsymbol{e}_{i_m} + \sum_{m>n} a_m \boldsymbol{\tau}_m + a_n \boldsymbol{\lambda}_k$, then $\boldsymbol{a}_n \leqslant 0$. Then, by Lemma 6, $\boldsymbol{e}_j$ is the residue of $\boldsymbol{\lambda}_k$ with respect to $\boldsymbol{\gamma}_1^{(k)}, \ldots, \boldsymbol{\gamma}_{n-1}^{(k)}, \boldsymbol{\tau}_{n+1}, \ldots, \boldsymbol{\tau}_L$. Therefore, by Proposition 3, $\text{MultiResidue}(\frac{1}{k}Q_n + (1-\frac{1}{k})\bar{Q}_n \,|\, \{Q_j\}_{j>n} \cup \{W_j^{(k)}\}_{j<n})$ is well-defined and if $W_n \longleftarrow \text{MultiResidue}(\frac{1}{k}Q_n + (1-\frac{1}{k})\bar{Q}_n \,|\, \{Q_j\}_{j>n} \cup \{W_j^{(k)}\}_{j<n})$, $W_n^{(k)}$ is one of the base distributions. Since $\boldsymbol{e}_j \in \boldsymbol{H}$ implies that $\boldsymbol{e}_j \notin \{\boldsymbol{e}_{i_1}, \ldots, \boldsymbol{e}_{i_{n-1}}\}$, it follows that $W_1^{(k)}, \ldots, W_n^{(k)}$ are distinct base distributions. This establishes the inductive step.

The result follows from applying Lemma 7. ■

Appendix E. Estimation

In this section, we present the estimation results of our paper. To begin, in Section E.1, we present the proof of sufficient conditions under which ResidueHat estimators converge uniformly in probability (Proposition 4). Second, in Section E.2, we prove our main estimation result for demixing mixed membership models (Theorem 5). Finally, in Section E.3, we prove our main estimation result for classification with partial labels (Theorem 6).

E.1. ResidueHat Results

Let $A_1, A_2, \ldots$ denote positive constants whose values may change from line to line. We introduce the following definitions.

Definition 11 *Let $\widehat{F}$ and $\widehat{H}$ be ResidueHat estimators of F and H, respectively, where $F \neq H$ and let $G \longleftarrow$ Residue$(F \,|\, H)$ and $\widehat{G} \longleftarrow$ ResidueHat$(\widehat{F} \,|\, \widehat{H})$. If $\widehat{G}$ is a ResidueHat estimator of order 0, we say its* distributional ancestors *are $\{F, H\}$ and define ancestors$(\widehat{G}) \coloneqq \{F, H\}$. If $\widehat{G}$ is a ResidueHat estimator of the kth order, we define its distributional ancestors to be ancestors$(\widehat{G})$ = ancestors$(\widehat{F}) \cup$ ancestors$(\widehat{H})$.*

The constants in our bounds depend on the distributional ancestors.

Definition 12 *We say that the distribution F satisfies the* support condition ***(SC)*** *with respect to H if there exists a distribution G and $\gamma \in [0, 1)$ such that $\text{supp}(H) \nsubseteq \text{supp}(G)$ and $F = (1-\gamma)G + \gamma H$.*

Definition 13 *If*

$$\sup_{E \in \mathcal{E}} |\widehat{F}(E) - F(E)| \xrightarrow{i.p.} 0$$

as $\boldsymbol{n} \longrightarrow \infty$, we say that $\widehat{F} \longrightarrow F$ uniformly (or $\widehat{F}$ converges uniformly to F) with respect to $\mathcal{E}$.

Definition 14 *Let $\widehat{F}$ be a ResidueHat estimator of a distribution F. We say that $\widehat{F}$ satisfies a Uniform Deviation Inequality* ***(UDI)*** *with respect to $\mathcal{E}$ if for all $\epsilon > 0$, there exist constants $A_{1,\epsilon}, A_{2,\epsilon} > 0$ and $\boldsymbol{N}$ depending on ancestors$(\widehat{F})$ such that if $\boldsymbol{n} \geqslant \boldsymbol{N}$, then for all $E \in \mathcal{E}$*

$$|\widehat{F}(E) - F(E)| < A_{1,\epsilon}\gamma_{\boldsymbol{n}} + \epsilon$$

with probability at least $1 - A_{2,\epsilon}\sum_{i \in [L]} \frac{1}{n_i}$

Henceforth, for the purposes of abbreviation, we will only say that a ResidueHat estimator satisfies a Uniform Deviation Inequality **(UDI)** and omit "with respect to $\mathcal{E}$" because the context makes this clear.

Definition 15 *Let $\widehat{F}$ and $\widehat{H}$ be ResidueHat estimators. We say that $\widehat{\kappa}(\widehat{F} \,|\, \widehat{H})$ satisfies a Rate of Convergence* ***(RC)*** *with respect to $\mathcal{E}$ if for all $\epsilon > 0$, there exists constants $A_{1,\epsilon}, A_{2,\epsilon} > 0$ and $\boldsymbol{N}$ depending on ancestors$(\widehat{F}) \cup$ ancestors$(\widehat{H})$ such that for $\boldsymbol{n} \geqslant \boldsymbol{N}$,*

$$|\widehat{\kappa}(\widehat{F} \,|\, \widehat{H}) - \kappa^*(F \,|\, H)| \leqslant A_{1,\epsilon}\gamma_{\boldsymbol{n}} + \epsilon$$

with probability at least $1 - A_{2,\epsilon}\sum_{i \in [L]} \frac{1}{n_i}$.

Lemma 8 gives sufficient conditions under which F satisfies **(SC)** with respect to H.

Lemma 8 *Let $P_1, \ldots, P_L$ satisfy* ***(A″)*** *and let $F, H \in \text{conv}(P_1, \ldots, P_L)$ such that $F \neq H$. Then, F satisfies* ***(SC)*** *with respect to H.*

Proof Let $A = \arg\min(|B| : B \subseteq \{P_1, \ldots, P_L\}, F, H \in \text{conv}(B))$. Without loss of generality, suppose that $A = \{P_1, \ldots, P_K\}$. F either lies on the boundary of $\text{conv}(P_1, \ldots, P_K)$ or doesn't. If F lies on the boundary of $\text{conv}(P_1, \ldots, P_K)$, then $H \in \text{conv}(P_1, \ldots, P_K)^\circ$ by minimality of A. Then, we pick $G = F$ and $\gamma = 0$ to obtain $F = (1-\gamma)F + \gamma H$. Since $P_1, \ldots, P_L$ satisfy **(A″)**, $\text{supp}(H) \nsubseteq \text{supp}(F)$.

Now, suppose that $F \in \text{conv}(P_1, \ldots, P_K)^\circ$. Let $G \longleftarrow \text{Residue}(F \,|\, H)$; we can write $F = (1-\gamma)G + \gamma H$ for $\gamma \in [0, 1)$ since $F \neq H$. Then, by Statement *2* of Lemma 2 and statement *3* of Proposition 3, G is on the boundary of $\text{conv}(P_1, \ldots, P_K)$. Without loss of generality, suppose that $G \in \text{conv}(P_1, \ldots, P_{K-1})$. Since $F = (1-\gamma)G + \gamma H \in \text{conv}(P_1, \ldots, P_K)^\circ$, and $G \in \text{conv}(P_1, \ldots, P_{K-1})$, $H \notin \text{conv}(P_1, \ldots, P_{K-1})$. Since $P_1, \ldots, P_L$ satisfy **(A″)**, $\text{supp}(H) \nsubseteq \text{supp}(G)$. This completes the proof. ■

Lemma 9 gives sufficient conditions under which an estimator $\widehat{G}$ satisfies a **(UDI)**.

Lemma 9 *Let*

1. *F and H be distributions such that $F \neq H$,*
2. *$G \longleftarrow$ Residue$(F \,|\, H)$, and*
3. *$\widehat{G} \longleftarrow$ ResidueHat$(\widehat{F} \,|\, \widehat{H})$.*

*If $\widehat{\kappa}(\widehat{F} \,|\, \widehat{H})$ satisfies a **(RC)**, $\widehat{H}$ satisfies a **(UDI)**, and $\widehat{F}$ satisfies a **(UDI)**, then $\widehat{G}$ satisfies a **(UDI)**.*

Proof For the sake of abbreviation, let $\widehat{\kappa} = \widehat{\kappa}(\widehat{F} \,|\, \widehat{H})$, $\kappa^* = \kappa^*(F \,|\, H)$, $\widehat{\alpha} = \frac{1}{1-\widehat{\kappa}}$ and $\alpha^* = \frac{1}{1-\kappa^*}$. Let $\epsilon > 0$. We claim that there are constants $A_{1,\epsilon}, A_{2,\epsilon} > 0$ such that for sufficiently large $\boldsymbol{n}$,

$$\Pr(|\widehat{\alpha} - \alpha^*| < A_{1,\epsilon}\gamma_{\boldsymbol{n}} + \epsilon) \geqslant 1 - A_{2,\epsilon} \sum_{i \in [L]} \frac{1}{n_i}. \tag{10}$$

Let $\delta = \frac{\epsilon(1-\kappa^*)^2}{2}$. Since $\widehat{\kappa}$ satisfies a **(RC)**, there exists constants $A_{1,\delta}, A_{2,\delta} > 0$ such that for large enough $\boldsymbol{n}$,

$$|\widehat{\kappa} - \kappa^*| \leqslant A_{1,\delta}\gamma_{\boldsymbol{n}} + \delta$$

with probability at least $1 - A_{2,\delta} \sum_{i \in [L]} \frac{1}{n_i}$. Since $F \neq H$, $\kappa^* < 1$ by Proposition 2, so we can let $\boldsymbol{n}$ large enough so that

$$\frac{1}{(1-\kappa^*)(1-\widehat{\kappa})} \leqslant 2\frac{1}{(1-\kappa^*)^2}$$

with high probability. Then, on this same event, for large enough $\boldsymbol{n}$,

$$\begin{aligned}
|\frac{1}{1-\kappa^*} - \frac{1}{1-\widehat{\kappa}}| &\leqslant \frac{A_{1,\delta}\gamma_{\boldsymbol{n}} + \delta}{(1-\kappa^*)(1-\widehat{\kappa})} \\
&\leqslant 2\frac{A_{1,\delta}\gamma_{\boldsymbol{n}} + \delta}{(1-\kappa^*)^2} \\
&\leqslant 2\frac{A_{1,\delta}\gamma_{\boldsymbol{n}}}{(1-\kappa^*)^2} + \epsilon.
\end{aligned}$$

Thus, we obtain the claim.

We can write $G = \alpha F + (1-\alpha)H$ with $\alpha \geqslant 1$. Then, by the triangle inequality,

$$\begin{aligned}
|\widehat{G} - G| &= |\widehat{\alpha}\widehat{F} + (1-\widehat{\alpha})\widehat{H} - \alpha F - (1-\alpha)H| \\
&\leqslant |\widehat{\alpha}\widehat{F} - \alpha F| + |(1-\widehat{\alpha})\widehat{H} - (1-\alpha)H| \\
&= |\widehat{\alpha}\widehat{F} - \widehat{\alpha}F + \widehat{\alpha}F - \alpha F| + |(1-\widehat{\alpha})\widehat{H} - (1-\widehat{\alpha})H + (1-\widehat{\alpha})H - (1-\alpha)H| \\
&\leqslant |\widehat{\alpha}||\widehat{F} - F| + |\widehat{\alpha} - \alpha| + |1-\widehat{\alpha}||\widehat{H} - H| + |\widehat{\alpha} - \alpha|.
\end{aligned}$$

Since $\widehat{F}$ satisfies a **(UDI)**, $\widehat{H}$ satisfies a **(UDI)**, inequality (10) holds, and $|\widehat{\alpha}|$ and $|1-\widehat{\alpha}|$ are bounded in probability, the result follows by an application of a union bound and picking the ϵs in the uniform deviation inequalities appropriately for each term. ■

Lemma 10 gives sufficient conditions under which $\widehat{\kappa}$ satisfies **(RC)**.

Lemma 10 *Let F and H be distributions such that $F \neq H$. If*

- *F satisfies **(SC)** with respect to H,*

- *$\widehat{F}$ satisfies **(UDI)**, and*
- *$\widehat{H}$ satisfies **(UDI)**,*

*then $\widehat{\kappa}(\widehat{F} \,|\, \widehat{H})$ satisfies **(RC)**.*

Proof For abbreviation, let $\kappa^* = \kappa(F \,|\, H)$ and $\widehat{\kappa} = \widehat{\kappa}(\widehat{F} \,|\, \widehat{H})$.

We first prove the upper bound. F satisfies **(SC)** with respect to H, so there exists a distribution G such that $F = (1-\gamma)G + \gamma H$ for some $\gamma \in [0,1)$ and $\text{supp}(H) \nsubseteq \text{supp}(G)$. Therefore, we have that G is irreducible with respect to H and, by Proposition 2, $\kappa^* = \gamma$.

Let $\delta > 0$ (to be chosen later). Since by hypothesis $\widehat{F}$ and $\widehat{H}$ satisfy **(UDI)**, there exist constants $A_{1,\delta}, A_{2,\delta} > 0$ such that for large enough $\boldsymbol{n}$, with probability at least $1 - A_{1,\delta}[\sum_{i \in [L]} \frac{1}{n_i}]$, for all $E \in \mathcal{E}$,

$$|\widehat{F}(E) - F(E)| < A_{2,\delta}\gamma_{\boldsymbol{n}} + \delta \tag{11}$$

$$|\widehat{H}(E) - H(E)| < A_{2,\delta}\gamma_{\boldsymbol{n}} + \delta. \tag{12}$$

Without loss of generality, let $A_{1,\delta}, A_{2,\delta} > 1$.

Pick $R \in \mathcal{E}$ such that $H(R) > 0$. By inequality (12), there exists $\boldsymbol{N}_1$ such that $\boldsymbol{n} \geqslant \boldsymbol{N}_1$ implies that $\widehat{H}(R) - \gamma_{\boldsymbol{n}} > 0$ with high probability. This implies that for $\boldsymbol{n} \geqslant \boldsymbol{N}_1$, $\widehat{\kappa}$ is finite. Let $\epsilon > 0$. By definition of $\widehat{\kappa}$, there exists $E \in \mathcal{E}$ such that

$$\frac{\epsilon}{2} + \widehat{\kappa} \geqslant \frac{\widehat{F}(E) + \gamma_{\boldsymbol{n}}}{(\widehat{H}(E) - \gamma_{\boldsymbol{n}})_+}.$$

Since $\widehat{\kappa}$ is finite, we have that $\widehat{H}(E) > \gamma_{\boldsymbol{n}}$ and $H(E) > 0$. Then,

$$\begin{aligned}
\frac{\epsilon}{2} + \widehat{\kappa} &\geqslant \frac{\widehat{F}(E) + \gamma_{\boldsymbol{n}}}{\widehat{H}(E) - \gamma_{\boldsymbol{n}}} \\
&\geqslant \frac{F(E) - (A_{2,\delta}-1)\gamma_{\boldsymbol{n}} - \delta}{H(E) + (A_{2,\delta}-1)\gamma_{\boldsymbol{n}} + \delta} \\
&\geqslant \frac{\gamma H(E)}{H(E) + (A_{2,\delta}-1)\gamma_{\boldsymbol{n}} + \delta} - \frac{(A_{2,\delta}-1)\gamma_{\boldsymbol{n}}}{H(E) + (A_{2,\delta}-1)\gamma_{\boldsymbol{n}} + \delta} - \frac{\delta}{H(E) + (A_{2,\delta}-1)\gamma_{\boldsymbol{n}} + \delta} \\
&\geqslant \frac{\gamma H(E)}{H(E)} - \frac{(A_{2,\delta}-1)\gamma_{\boldsymbol{n}} + \delta}{H(E)} - \frac{(A_{2,\delta}-1)\gamma_{\boldsymbol{n}}}{H(E) + (A_{2,\delta}-1)\gamma_{\boldsymbol{n}} + \delta} - \frac{\delta}{H(E) + (A_{2,\delta}-1)\gamma_{\boldsymbol{n}} + \delta} \\
&\geqslant \kappa^* - 2\frac{(A_{2,\delta}-1)\gamma_{\boldsymbol{n}}}{H(E)} - 2\frac{\delta}{H(E)}
\end{aligned}$$

where in the second to last inequality we used the elementary fact that if $a, b, c > 0$ and $a \leqslant b$, then $\frac{a}{b+c} \geqslant \frac{a}{b} - \frac{c}{b}$. Picking $\delta = \frac{H(E)\epsilon}{4}$, we obtain the upper bound.

The proof of the other direction of the inequality is very similar to the proof of Theorem 2 in Scott (2015). By hypothesis, F satisfies **(SC)** with respect to H, so there exists a distribution G such that $F = (1-\gamma)G + \gamma H$ for some $\gamma \in [0,1)$ and $\text{supp}(H) \nsubseteq \text{supp}(G)$. Therefore, we have that G is irreducible with respect to H and, by Proposition 2, $\kappa^*(F \,|\, H) = \gamma$. For

abbreviation, let $\kappa^* = \kappa^*(F \,|\, H)$ and $\widehat{\kappa} = \widehat{\kappa}(\widehat{F} \,|\, \widehat{H})$. Since $\text{supp}(H) \nsubseteq \text{supp}(G)$, there exists an open set O such that

$$\frac{F(O)}{H(O)} = (1-\gamma)\frac{G(O)}{H(O)} + \gamma = \kappa^*.$$

Then, since $\mathcal{E}$ contains a generating set for the standard topology on $\mathbb{R}^d$, there exists $E \in \mathcal{E}$ such that

$$\frac{F(E)}{H(E)} = \kappa^*.$$

Let $\delta > 0$ such that $\delta \leqslant \frac{1}{4}H(E)$. Since by hypothesis $\widehat{F}$ and $\widehat{H}$ satisfy **(UDI)**, there exist constants $A_{3,\delta}, A_{4,\delta} > 0$ such that for large enough $\boldsymbol{n}$, with probability at least $1 - A_{3,\delta}[\sum_{i\in[L]} \frac{1}{n_i}]$,

$$\begin{aligned}\widehat{\kappa} &\leqslant \frac{F(E) + A_{4,\delta}\gamma_{\boldsymbol{n}} + \delta}{(H(E) - A_{4,\delta}\gamma_{\boldsymbol{n}} - \delta)_+} \\ &\leqslant \frac{F(E) + \epsilon}{(H(E) - \epsilon)_+}\end{aligned}$$

where $\epsilon = 2A_{4,\delta}\gamma_{\boldsymbol{n}} + \delta$. The rest of the proof is identical to the proof of Theorem 2 from Scott (2015) and, therefore, we omit it. ■

The following theorem gives sufficient conditions under which a ResidueHat estimator satisfies **(UDI)**. It is the basis of Proposition 4.

Lemma 11 *If $P_1, \ldots, P_L$ satisfy **(A″)** and $\widehat{G}$ is a ResidueHat estimator of order k of a distribution $G \in \text{conv}(P_1, \ldots, P_L)$, then $\widehat{G}$ satisfies **(UDI)**.*

Proof Let $\widehat{G} \longleftarrow \text{ResidueHat}(\widehat{F} \,|\, \widehat{H})$ where $G \longleftarrow \text{ResidueHat}(F \,|\, H)$, $F \neq H$, $F, H \in \text{conv}(P_1, \ldots, P_L)$ and $\widehat{F}, \widehat{H}$ are ResidueHat estimators of F and H respectively. We use induction on k. Suppose $k = 0$. Then, $\widehat{F}$ and $\widehat{H}$ are empirical distributions. Therefore, the VC inequality applies to $\widehat{F}$ and $\widehat{H}$. Consequently, $\widehat{F}$ and $\widehat{H}$ satisfy **(UDI)**. Since $P_1, \ldots, P_L$ satisfy **(A″)**, $F, H \in \text{conv}(P_1, \ldots, P_L)$ and $F \neq H$, by Lemma 8, F satisfies **(SC)** with respect to H. Then, by Lemma 10, $\widehat{\kappa}(\widehat{F} \,|\, \widehat{H})$ satisfies **(RC)**. Then, all of the assumptions of Lemma 9 are satisfied, so $\widehat{G}$ satisfies **(UDI)**. Note that $G \in \text{conv}(P_1, \ldots, P_L)$ by Proposition 3.

The inductive step ($k > 0$) follows by similar reasoning. The difference is that instead of applying the VC inequality to $\widehat{F}$ and $\widehat{H}$, we use the fact that $\widehat{F}$ and $\widehat{H}$ are ResidueHat estimators of order $k-1$ and, therefore, satisfy **(UDI)** by the inductive hypothesis. ■

Proof [Proof of Proposition 4] Let $0 < \delta < \epsilon$. By Lemma 11, $\widehat{G}$ satisfies **(UDI)**. Consequently, there exist constants $A_{1,\delta}, A_{2,\delta} > 0$ such that for large enough $\boldsymbol{n}$ with probability at least $1 - A_{1,\delta}\sum_{i\in[L]} \frac{1}{n_i}$, $\widehat{G}$ satisfies for every $E \in \mathcal{E}$,

$$|\widehat{G}(E) - G(E)| \leqslant A_{2,\delta}\gamma_{\boldsymbol{n}} + \delta = A_{2,\delta}\sum_{i\in[L]} \epsilon_i(\frac{1}{n_i}) + \delta \longrightarrow \delta < \epsilon.$$

■

E.2. Demixing Mixed Membership Models

In this section, we prove our main estimation result for demixing mixed membership models, i.e., Theorem 5. First, in Section E.2.1, we present an important lemma for FaceTestHat. Second, in Section E.2.2, we present an empirical version of Demix and prove Theorem 5.

E.2.1. The FaceTestHat Algorithm

The following establishes that FaceTestHat behaves as desired.

Lemma 12 *Let $\epsilon \in (0,1)$. For all $j \in [K]$, let $Q_j = \boldsymbol{\eta}_j^T \boldsymbol{P}$ and $\boldsymbol{\eta}_j \in \Delta_K$ such that every $\boldsymbol{\eta}_j$ lies in the relative interior of the same face of Δ_K. Let $P_1, \ldots, P_K$ satisfy **(A″)**, and $Q_1, \ldots, Q_K \in \text{conv}(P_1, \ldots, P_K)$ be distinct. Let $\widehat{Q}_i$ be a ResidueHat estimate of Q_i $\forall i \in [K]$.*

1. *With probability tending to 1 as $\boldsymbol{n} \longrightarrow \infty$, if FaceTestHat$(\widehat{Q}_1, \cdots, \widehat{Q}_K \,|\, \epsilon)$ returns 1, then $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_K$ are in the relative interior of the same face.*
2. *Let $\kappa^*_{i,j} = \kappa^*(Q_i \,|\, Q_j)$. If $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_K$ are in the relative interior of the same face and $\min_{i,j} \kappa^*_{i,j} > \epsilon$, then with probability tending to 1 as $\boldsymbol{n} \longrightarrow \infty$, FaceTestHat$(\widehat{Q}_1, \cdots, \widehat{Q}_K \,|\, \epsilon)$ returns 1.*

Proof Let $\epsilon > 0$, $\kappa^*_{i,j} = \kappa^*(Q_i \,|\, Q_j)$ and $\widehat{\kappa}_{i,j} = \widehat{\kappa}(\widehat{Q}_i \,|\, \widehat{Q}_j)$. Since $P_1, \ldots, P_K$ satisfy **(A″)** and $Q_i \neq Q_j$, by Lemma 8, Q_i satisfies **(SC)** wrt Q_j. Since $\widehat{Q}_i$ and $\widehat{Q_j}$ are ResidueHat estimators, $\widehat{Q}_i$ and $\widehat{Q}_j$ satisfy **(UDI)** (Lemma 11). Then, by Lemma 10 $\widehat{\kappa}_{i,j}$ satisfies **(RC)**.

1. We prove the contrapositive. Suppose that $Q_1, \ldots, Q_K$ are not in the relative interior of the same face. Then, by Proposition 8, FaceTest$(Q_1, \ldots, Q_K)$ returns 0, which occurs if and only if there exist $i \neq j$ such that $\kappa^*_{i,j} = 0$. Since $\widehat{\kappa}_{i,j}$ satisfies **(RC)**, as $\boldsymbol{n} \longrightarrow \infty$, with probability tending to 1, $\widehat{\kappa}_{i,j} \longrightarrow 0$. This completes the proof.
2. If $\min_{i,j} \kappa^*_{i,j} > \epsilon$, then as $\boldsymbol{n} \longrightarrow \infty$, with probability tending to 1, $\min_{i,j} \widehat{\kappa}_{i,j} > \epsilon$.

■

E.2.2. The DemixHat Algorithm

The DemixHat algorithm (see Algorithm 11) differs from the Demix algorithm in that *(i)* it requires the specification of a constant $\epsilon \in (0,1)$ and *(ii)* it only uses the two-sample κ^* operator. In the interest of clarity, we state the population version of the algorithm DemixHat, which we call Demix2. The only difference between Demix and Demix2 is that line 7 in Demix has been replaced with lines 6-8 in Demix2.

Lemma 13 establishes that it is possible to replace line 7 of the Algorithm 4 with the sequence of applications of the two-sample κ^* in lines 6-8 of Algorithm 16, without changing the conclusion of Theorem 2.

Lemma 13 *Let $\{i_1, \ldots, i_K\} \subset [L]$ be distinct indices. Let $P_1, \ldots, P_L$ be jointly irreducible, $(Q_1, \ldots, Q_{K-1})$ be a permutation of $(P_{i_1}, \ldots, P_{i_{K-1}})$ and $Q^1_K \in \text{conv}(P_{i_1}, \ldots, P_{i_K})^\circ$. Define*

Algorithm 16 Demix2($S_1, \ldots, S_K$)

Input: $S_1, \ldots, S_K$ are distributions

1: **if** $K = 2$ **then**
2: **return** $(\text{Residue}(S_1 \,|\, S_2), \text{Residue}(S_2 \,|\, S_1))^T$
3: **else**
4: $(R_2, \ldots, R_K)^T \longleftarrow \text{FindFace}(S_1, \ldots, S_K)$
5: $(Q_1, \ldots, Q_{K-1})^T \longleftarrow \text{Demix2}(R_2, \ldots, R_K)$
6: **for** $i = 1, \ldots, K-1$ **do**
7: $Q_K \longleftarrow \text{Residue}(Q_K \,|\, Q_i)$
8: **end for**
9: **return** $(Q_1, \ldots, Q_K)^T$
10: **end if**

the sequence

$$Q_K^i \longleftarrow \mathit{Residue}(Q_K^{i-1} \,|\, Q_{i-1});$$

then, $Q_K^K = P_{i_K}$.

Proof Relabel the distributions so that $Q_j = P_j$. Let $\boldsymbol{\mu}_i$ denote the mixture proportion of Q_K^i and $\boldsymbol{e}_j$ the mixture proportion of P_j. Write $\boldsymbol{\mu}_1 = \sum_{i=1}^K \alpha_i \boldsymbol{e}_i$. We claim that $\boldsymbol{\mu}_k = \frac{\sum_{i \geqslant k} \alpha_i \boldsymbol{e}_i}{\sum_{i \geqslant k} \alpha_i}$ for all $k \leqslant K$. We prove this inductively. The base case $k = 1$ follows since $\sum_{i \geqslant 1} \alpha_i = 1$. Next, we prove the inductive step. Suppose that $\boldsymbol{\mu}_{k-1} = \frac{\sum_{i \geqslant k-1} \alpha_i \boldsymbol{e}_i}{\sum_{i \geqslant k-1} \alpha_i}$. By Proposition 3, the mixture proportion of Q_K^k, $\boldsymbol{\mu}_k$, is the residue of $\boldsymbol{\mu}_{k-1}$ with respect to $\boldsymbol{e}_{k-1}$. By statement 1 of Lemma 2, we can write

$$\begin{aligned}\boldsymbol{\mu}_k &= \boldsymbol{e}_{k-1} + \alpha^*(\boldsymbol{\mu}_{k-1} - \boldsymbol{e}_{k-1}) \\ &= \frac{[\sum_{i \geqslant k-1} \alpha_i (1 - \alpha^*) + \alpha^* \alpha_{k-1}] \boldsymbol{e}_{k-1} + \alpha^* \sum_{i \geqslant k} \alpha_i \boldsymbol{e}_i}{\sum_{i \geqslant k-1} \alpha_i}\end{aligned}$$

where

$$\alpha^* = \frac{1}{1 - \kappa^*(\boldsymbol{\mu}_{k-1} \,|\, \boldsymbol{e}_{k-1})}$$

and we have used the inductive hypothesis $\boldsymbol{\mu}_{k-1} = \frac{\sum_{i \geqslant k-1} \alpha_i \boldsymbol{e}_i}{\sum_{i \geqslant k-1} \alpha_i}$. α^* is the value of the following optimization problem (in statement 1 of Lemma 2):

$$\max(\alpha \geqslant 1 \,|\, \exists G, G = \boldsymbol{\mu}_{k-1} + \alpha(\boldsymbol{e}_{k-1} - \boldsymbol{\mu}_{k-1})).$$

Inspection of the above optimization problem reveals that $\alpha^* = \frac{\sum_{i \geqslant k-1} \alpha_i}{\sum_{i \geqslant k} \alpha_i}$. Plugging this into the above equation gives $\boldsymbol{\mu}_k = \frac{\sum_{i \geqslant k} \alpha_i \boldsymbol{e}_i}{\sum_{i \geqslant k} \alpha_i}$. This establishes the claim.

Setting $k = K$, it follows that $\mu_K = \frac{\alpha_K \boldsymbol{e}_K}{\alpha_K} = \boldsymbol{e}_K$. ■

Corollary 2 *Let* $P_1, \ldots, P_L$ *be jointly irreducible and* $\boldsymbol{\Pi}$ *have full column rank. Then, with probability* 1*, Demix2*($\tilde{\boldsymbol{P}}$) *returns a permutation of* $\boldsymbol{P}$.

Proof [Proof of Theorem 5] Note that every estimator of a distribution in the DemixHat algorithm is a ResidueHat estimator since *(i)* the Demix2 algorithm 16 only considers distributions that are in $\text{conv}(P_1, \ldots, P_L)$ and *(ii)* only computes $\text{Residue}(F \,|\, H)$ if $F \neq H$. To see why *(ii)* is true, consider: the Demix2 algorithm computes $\text{Residue}(\cdot \,|\, \cdot)$ at lines 2 and 7 in Demix2 and line 3 in FindFace. In the proof of Theorem 2, we showed that $S_1, \ldots, S_K$ are always linearly independent and therefore distinct. This implies that in lines 2 and 7 in Demix2 and line 3 in FindFace, the residue function is called on distinct distributions. Thus, every estimator of a distribution of the DemixHat algorithm satisfies the assumptions of Lemma 11.

First, we argue that the order of the ResidueHat estimators is bounded; this implies that the constants in the uniform deviation inequalities associated with the ResidueHat estimators are bounded. We give a very loose bound. DemixHat calls itself at most $L-1$ times and in each call recurses on at most $L-1$ ResidueHat estimators and calculates at most $L-1$ more ResidueHat estimators. Therefore, each ResidueHat estimator has order at most $(L-1)^3$.

Second, let A_i denote the event that DemixHat recurses on i distributions lying in the relative interior of an i-face in the $(L-i)$th recursive call. We show that the event $\cap_{i=2}^{L-1} A_i$ occurs with probability tending to 1 as $\boldsymbol{n} \longrightarrow \infty$. Consider A_{L-1}. Let $\widehat{R}_i^{(n)}$ denote the estimate of the ith distribution in line 3 in the nth iteration of the for loop in Algorithm 12 and let $R_i^{(n)}$ denote the corresponding distribution. Let $\kappa_{i,j,n}^* = \kappa^*(R_i^{(n)} \,|\, R_j^{(n)})$. From the proof of Theorem 2, there exists an integer $N_1 \geqslant 0$ such that for $n \geqslant N_1$, $R_i^{(n)}$ lies in the relative interior of the same face for all $i = 2, \ldots, L$. Further, using the notation from the proof of Theorem 2, we have that the mixture proportions of the $R_i^{(n)}$s, i.e., the $\boldsymbol{\mu}_i^{(n)}$s, converge to a common $\boldsymbol{\lambda}$ on this face, i.e., for all $i = 2, \ldots, L$, $\left\| \boldsymbol{\mu}_i^{(n)} - \boldsymbol{\lambda} \right\| \longrightarrow 0$. Thus, by statement 3 of Lemma 4, for all $i \neq j \in [L] \backslash \{1\}$ $\kappa^*(\boldsymbol{\mu}_i^{(n)} \,|\, \boldsymbol{\mu}_j^{(n)}) \longrightarrow 1$. Hence, there exists $N_2 \geqslant N_1$ such that $\kappa^*(\boldsymbol{\mu}_i^{(N_2)} \,|\, \boldsymbol{\mu}_j^{(N_2)}) > \epsilon$ for all $i \neq j$. By statement 1 of Lemma 12 and a union bound argument, with probability increasing to 1, $\text{FaceTestHat}(\widehat{R}_2^{(n)}, \ldots, \widehat{R}_L^{(n)} \,|\, \epsilon)$ returns 0 for all $n < N_1$ since $R_2^{(n)}, \ldots, R_L^{(n)}$ are not on the relative interior of the same face. Thus, with probability tending to 1, FaceTest does not make the mistake to return 1 before the distributions $R_2^{(n)}, \ldots, R_L^{(n)}$ are on the relative interior of the same face. By statement 2 of Lemma 12, with probability tending to 1 as $\boldsymbol{n} \longrightarrow \infty$, $\text{FaceTestHat}(\widehat{R}_2^{(N_2)}, \ldots, \widehat{R}_L^{(N_2)} \,|\, \epsilon)$ returns 1. Hence, with probability increasing to 1, the event A_{L-1} occurs. Applying the same argument to A_i for $i < L-1$ and taking the union bound shows that $\cap_{i=2}^{L-1} A_i$ occurs with probability tending to 1 as $\boldsymbol{n} \longrightarrow \infty$.

Now, we can complete the proof. Under the assumptions of Theorem 2, there is a permutation σ such that for each distribution Q_i estimated by $\widehat{Q}_i$, $P_{\sigma(i)} = Q_i$. By Proposition 4, as $\boldsymbol{n} \longrightarrow \infty$, $\widehat{Q}_i$ converges uniformly to Q_i. The result follows. ■

Algorithm 17 VertexTestHat($\mathbf{\Pi}^+, (\tilde{P}_1^\dagger, \ldots, \tilde{P}_M^\dagger)^T, (\widehat{Q}_1, \ldots, \widehat{Q}_L)^T$)

Form the matrix $\widehat{\boldsymbol{M}}_{i,j} \coloneqq \widehat{\kappa}(\tilde{P}_i^\dagger \,|\, \widehat{Q}_j)$
Let $|\mathbf{\Pi}^+|$ denote the number of nonzero entries in $\mathbf{\Pi}^+$
Form the matrix $\widehat{\boldsymbol{Z}}$ by making the $|\mathbf{\Pi}^+|$ largest entries of $\widehat{\boldsymbol{M}}$ equal to 1 and the rest of its entries equal to 0
Use any algorithm that finds a permutation matrix $\boldsymbol{C}$ such that $\widehat{\boldsymbol{Z}}\boldsymbol{C} = \mathbf{\Pi}^+$ (if it exists)

if such a permutation matrix $\boldsymbol{C}$ exists **then**
  return $(1, \boldsymbol{C}^T)$
else
  return $(0, \mathbf{0})$
end if

E.3. Classification with Partial Labels

In this section, we prove Theorem 6. To begin, we briefly sketch an argument that one can reduce any instance of a partial label model satisfying **(B3)** and **(A)** to an instance of a partial label model that also satisfies **(D)**. Let $J = \{i : \mathbf{\Pi}^+_{i,:} = \boldsymbol{e}_j^T \text{ for some } j \in [L]\} = \{j_1, \ldots, j_k\}$, the set of indices of contaminated distributions that are equal to some base distribution. Compute Residue($\tilde{P}_i \,|\, \tilde{P}_{j_1}$) for $i \in [L] \backslash J$ if there is l such that $\mathbf{\Pi}^+_{i,l} = \mathbf{\Pi}^+_{j_1,l} = 1$. Replace $\tilde{P}_i$ with Residue($\tilde{P}_i \,|\, \tilde{P}_{j_1}$) (and call it $\tilde{P}_i$ for simplicity of presentation). Update $\mathbf{\Pi}^+$ and remove j_1 from J. Repeat this procedure until J is empty. Then, there will be $(L - |J|)$ $\tilde{P}_i$ lying in a $(L - |J|)$-face of Δ_L that are not equal to any of the base distributions and the other contaminated distributions will be equal to base distributions. Then, it suffices to solve the instance of the partial label model on the $(L - |J|)$-face, which satisfies **(D)**.

Next, we introduce VertexTestHat (Algorithm 17), an empirical version of VertexTest, and prove that it satisfies a useful consistency property.

Lemma 14 *Suppose that $P_1, \ldots, P_L$ satisfy **(A″)**, $\mathbf{\Pi}$ has full column rank, the columns of $\mathbf{\Pi}^+$ are unique and $\mathbf{\Pi}^+$ satisfies **(D)**. Let $\widehat{Q}_1, \ldots, \widehat{Q}_L$ be ResidueHat estimators of $Q_1, \ldots, Q_L$, respectively. Suppose that $(Q_1, \ldots, Q_L)$ is a permutation of $(P_1, \ldots, P_L)$. Then, with probability tending to 1 as $\boldsymbol{n} \longrightarrow \infty$, VertexTestHat($\mathbf{\Pi}^+, (\tilde{P}_1^\dagger, \ldots, \tilde{P}_M^\dagger)^T, (\widehat{Q}_1, \ldots, \widehat{Q}_L)^T$) returns a permutation matrix $\boldsymbol{C}$ such that $\forall i$, $\boldsymbol{C}_{i,:}(\widehat{Q}_1, \ldots, \widehat{Q}_L)^T$ is a ResidueHat estimator of P_i.*

Proof Define $\widehat{\kappa}_{i,j} \coloneqq \widehat{\kappa}(\tilde{P}_i^\dagger \,|\, \widehat{Q}_j)$ and $\kappa^*_{i,j} \coloneqq \kappa^*(\tilde{P}_i \,|Q_j)$. We claim that $\widehat{\kappa}_{i,j}$ satisfies a **(RC)**. Since $P_1, \ldots, P_L$ satisfy **(A″)** and by assumption **(D)** $Q_j \neq \tilde{P}_i$, by Lemma 8, $\tilde{P}_i$ satisfies **(SC)** wrt Q_j. Since $\tilde{P}_i^\dagger$ is an empirical distribution, $\tilde{P}_i^\dagger$ satisfies a **(UDI)**. Since $\widehat{Q}_j$ is a ResidueHat estimator, $\widehat{Q}_j$ satisfies a **(UDI)** by Lemma 11. Therefore, the hypotheses of Lemma 10 are satisfied and $\widehat{\kappa}_{i,j}$ satisfies a **(RC)**.

Form the matrix $Z_{i,j} = \mathbf{1}_{\{\kappa^*(\tilde{P}_i \,|\, Q_j) > 0\}}$ as in Algorithm 9. Since $Q_1, \ldots, Q_L$ are a permutation of $P_1, \ldots, P_L$, $\boldsymbol{Z}$ is formed by permuting the columns of $\mathbf{\Pi}^+$ appropriately. Thus, there are $|\mathbf{\Pi}^+|$ (i, j) pairs such that $\kappa_{i,j} > 0$ and the rest are such that $\kappa_{i,j} = 0$. Then, using Lemma 10 and a union bound, with probability tending to 1 as $\boldsymbol{n} \longrightarrow \infty$, $\widehat{\kappa}_{i,j}$ is among the $|S|$ largest values in the matrix $\widehat{M}$ if and only if $\kappa_{i,j} > 0$. On this event, VertexTestHat($\mathbf{\Pi}^+, (\tilde{P}_1^\dagger, \ldots, \tilde{P}_M^\dagger)^T, (\widehat{Q}_1, \ldots, \widehat{Q}_L)^T$) and VertexTest($\mathbf{\Pi}^+, (\tilde{P}_1, \ldots, \tilde{P}_M)^T, (Q_1, \ldots, Q_L)^T$) return the same output. Since $(Q_1, \ldots, Q_L)$ is a permutation of $(P_1, \ldots, P_L)$ by hypothesis, by Lemma 7 VertexTest($\mathbf{\Pi}^+, (\tilde{P}_1, \ldots, \tilde{P}_M)^T, (Q_1, \ldots, Q_L)^T$) returns $(1, \boldsymbol{C}^T)$ such that $\boldsymbol{C}^T(Q_1, \ldots, Q_L)^T = \boldsymbol{P}$. The result follows. ■

Proof [Proof of Theorem 6] Let $(\widehat{Q}_1, \ldots, \widehat{Q}_L) \longleftarrow$ DemixHat($\tilde{P}_1^\dagger, \ldots, \tilde{P}_M^\dagger \,|\, \epsilon$). By Theorem 5, w.p. tending towards 1 as $\boldsymbol{n} \longrightarrow \infty$, there exists a permutation $\sigma : [L] \longrightarrow [L]$ such that for every $i \in [L]$,

$$\sup_{E \in \mathcal{E}} |\widehat{Q}_i(E) - P_{\sigma(i)}(E)| < \delta.$$

From the proof of Theorem 5, each $\widehat{Q}_i$ is a ResidueHat estimator. The assumptions of Lemma 14 are satisfied. The result follows immediately from Lemma 14. ■

Appendix F. Previous Results

Lemma 15 (Lemma A.1 (Blanchard and Scott, 2014)) *The maximum operation in the definition of κ^* and $\widehat{\kappa}$ (lines (3) and (6), respectively) is well-defined, that is, the outside supremum is attained at at least one point.*

Lemma 16 (Lemma B.1 (Blanchard and Scott, 2014)) *If Π satisfies **(B1)**, then $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_L$ are linearly independent. If $P_1, \ldots, P_L$ are jointly irreducible, then they are linearly independent. If $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_L$ are linearly independent and $P_1, \ldots, P_L$ are linearly independent, then $\tilde{P}_1, \ldots, \tilde{P}_L$ are linearly independent.*

Appendix G. Experiments

In this Section, we perform experiments that suggest that joint irreducibility of $P_1, \ldots, P_L$ is a reasonable assumption. In particular, our experiments suggest that on the datasets in question, **(A″)** holds (which is a strictly stronger condition than joint irreducibility). We consider three datasets: classes 1, 2, and 3 of MNIST (LeCun et al., 1998), the Iris dataset (Fisher, 1936), and the Breast Cancer Wisconsin (Diagnostic) Data Set (Dheeru and K. Taniskidou, 2017). We use the Spectral Support Estimation algorithm (De Vito et al., 2010; Rudi et al., 2014) to estimate the support of each class in each dataset. We split each dataset into training, validation, and test sets, applying the algorithm to the training set, using the validation set to pick the hyperparameters, and evaluating the performance on the test set. We average our results over 60 trials where in each trial we randomly permute the dataset, thus altering the training, validation, and test sets. Let $\widehat{S}_i$ denote an estimate of the support of class i. Tables 1, 3, and 5 display an estimate of the probability that a point sampled from P_i belongs to the estimate of the support $\widehat{S}_i$. They indicate that the Spectral Support Estimation has reasonably good performance in producing $\widehat{S}_i$s containing the support of the associated class. Tables 2, 4, and 6 use the $\widehat{S}_i$ to estimate the quantity

$\Pr_{\boldsymbol{x}\sim P_i}(\boldsymbol{x} \in \cup_{j\neq i} \mathrm{supp}(P_j))$, which must be strictly less than 1 for **(A″)** to hold. We find that our estimates are considerably less than 1, which suggests that joint irreducibility holds on these datasets.

	$i=1$	$i=2$
$\widehat{\Pr}_{\boldsymbol{x}\sim P_i}(\boldsymbol{x} \in \widehat{S}_i)$	0.87	0.89

Table 1: Cancer Support Results.

	$i=1$	$i=2$
$\widehat{\Pr}_{\boldsymbol{x}\sim P_i}(\boldsymbol{x} \in \cup_{j\neq i}\widehat{S}_j)$	0.18	0.38

Table 2: Cancer Separability Results.

	$i=1$	$i=2$	$i=3$
$\widehat{\Pr}_{\boldsymbol{x}\sim P_i}(\boldsymbol{x} \in \widehat{S}_i)$	0.86	0.84	0.84

Table 3: Iris Support Results.

	$i=1$	$i=2$	$i=3$
$\widehat{\Pr}_{\boldsymbol{x}\sim P_i}(\boldsymbol{x} \in \cup_{j\neq i}\widehat{S}_j)$	0.0	0.17	0.19

Table 4: Iris Separability Results.

	$i=1$	$i=2$	$i=3$
$\widehat{\Pr}_{\boldsymbol{x}\sim P_i}(\boldsymbol{x} \in \widehat{S}_i)$	0.98	0.87	0.83

Table 5: MNIST Support Results.

	$i=1$	$i=2$	$i=3$
$\widehat{\Pr}_{\boldsymbol{x}\sim P_i}(\boldsymbol{x} \in \cup_{j\neq i}\widehat{S}_j)$	0.08	0.17	0.14

Table 6: MNIST Separability Results.

References

S. Arora, R. Ge, and A. Moitra. Learning topic models–going beyond svd. *Foundations of Computer Science*, 2012.

S. Arora, R. Ge, Y. Halpern, D. Mimno, A. Moitra, D. Sontag, Y. Wu, and M. Zhu. A practical algorithm for topic modeling with provable guarantees. *International Conference on Machine Learning*, 2013.

S. Axler. *Linear Algebra Done Right*. Springer, 3rd edition, 2015.

L. Berkman, B. H. Singer, and K. Manton. Black/white differences in health status and mortality among the Elderly. *Demography*, 26:661–678, 1989.

G. Blanchard and C. Scott. Decontamination of mutually contaminated models. *International Conference of Artificial Intelligence and Statistics*, 2014.

G. Blanchard, G. Lee, and C. Scott. Semi-supervised novelty detection. *Journal of Machine Learning Research*, 11:2973–3009, 2010.

G. Blanchard, M. Flaska, G. Handy, S. Pozzi, and C. Scott. Classification with asymmetric label noise: Consistency and maximal denoising. *Electronic Journal of Statistics*, 10: 2780–2824, 2016.

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

J. Cid-Sueiro. Proper losses for learning from partial labels. *Advances in Neural Information Processing Systems*, 2012.

T. Cour, B. Sapp, and B. Taskar. Learning from partial labels. *Journal of Machine Learning Research*, 12:1501–1536, 2011.

R. Das, M. Zaheer, and C. Dyer. Gaussian lda for topic models with word embeddings. 2015.

E. De Vito, L. Rosasco, and A. Toigo. Spectral regularization for support estimation. *Advances in neural information processing systems*, 2010.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

D. Dheeru and E. K. Taniskidou. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

W. Ding, M. Rohban, P. Ishwar, and V. Saligrama. Topic discovery through data dependent and random projections. *International Conference on Machine Learning*, 2013.

W. Ding, M. Rohban, P. Ishwar, and V. Saligrama. Efficient distributed topic modeling with provable guarantees. *International Conference on Artificial Intelligence and Statistics*, 2014.

D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts? *Advances in neural information processing systems*, 2003.

R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annual Eugenics*, pages 179–188, 1936.

A. Ghosh, H. Kumar, and P.S. Sastry. Robust loss functions under label noise for deep neural networks. *AAAI*, 2017.

K. Huang, X. Fu, and N. D. Sidiropoulos. Anchor-free correlated topic modeling. *Advances in Neural Information Processing Systems*, 2016.

S. Jain, M. White, M. W. Trosset, and P. Radivojac. Nonparametric semi-supervised learning of class proportions. *arXiv preprint arXiv:1601.01944*, 2016.

R. Jin and Z. Ghahramani. Learning with multiple labels. *Advances in Neural Information Processing Systems*, 2002.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *IEEE*, pages 2278–2324, 1998.

C. Li, H. Wang, Z. Zhang, A. Sun, and Z. Ma. Topic modeling for short texts with auxiliary word embeddings. *ACM SIGIR conference on Research and Development in Information Retrieval*, 2016a.

F. Li and P. Perona. A bayesian hierarchical model for learning natural scene categories. *Computer Vision and Pattern Recognition*, 2005.

S. Li, T. Chua, and C. Miao. Generative topic embedding: a continuous representation of documents. *ACL*, 2016b.

L.-P. Liu and T. G. Dietterich. A conditional multinomial mixture model for superset label learning. *Advances in Neural Information Processing Systems*, 2012.

L.-P. Liu and T. G. Dietterich. Learnability of the superset label learning problem. *International Conference on Machine Learning*, 32, 2014.

M. Luong, R. Socher, and C. Manning. Better word representations with recursive neural networks for morphology. *Conference on Computational Natural Language Learning*, 2013.

A. Menon, B. van Rooyen, C. Ong, and B. Williamson. Learning from corrupted binary labels via class-probability estimation. *International Conference on Machine Learning*, 2015a.

A. Menon, B. van Rooyen, C. S. Ong, and R. Williamson. Learning from corrupted binary labels via class-probability estimation. 2015b.

A. Menon, B. van Rooyen, and N. Natarajan. Learning from binary labels with instance-dependent corruption. *arXiv preprint*, 2016.

E. Metodiev and J. Thaler. On the topic of jets. *arXiv preprint arXiv:1802.00008*, 2018a.

E. Metodiev and J. Thaler. Jet topics: Disentangling quarks and gluons at colliders. *Physical Review Letters*, 120(24):241602, 2018b.

N. Natarajan, I. S. Dhillon, P. Ravikumar, and A. Tewari. Learning with noisy labels. *Advances in Neural Information Processing Systems*, 2013.

N. Nyugen and R. Caruana. Classification with partial labels. *International Conference on Knowledge Discovery and Data Mining*, 2008.

G. Patrini, A. Rozza, A. Menon, R. Nock, and L. Qu. Making deep neural networks robust to label noise: a loss correction approach. *CVPR*, 2017.

J. K. Pritchard, M. Stephens, N. A. Rosenberg, and P. Donnelly. Association mapping in structured populations. *American Journal of Human Genetics*, 67:170–181, 2000.

H. Ramaswamy, C Scott, and A Tewari. Mixture proportion estimation via kernel embeddings of distributions. *International Conference on Machine Learning*, 2016.

B. Recht, C. Re, J. Tropp, and V. Bittorf. Factoring non-negative matrices with linear programs. *Advances in Neural Information Processing Systems*, 2012.

A. Rudi, F. Odone, and E. De Vito. Geometrical and computational aspects of spectral support estimation for novelty detection. *Pattern Recognition Letters*, 2014.

T. Sanderson and C. Scott. Class proportion estimation with application to multiclass anomaly rejection. *International Conference on Artificial Intelligence and Statistics*, 2014.

C. Scott. A rate of convergence for mixture proportion estimation, with application to learning from noisy labels. *International Conference on Artificial Intelligence and Statistics*, 2015.

C. Scott, G. Blanchard, and G. Handy. Classification with asymmetric label noise: Consistency and maximal denoising. *Conference on Learning Theory*, 2013.

B. van Rooyen and R. Williamson. Learning in the presence of corruption. *arXiv preprint*, 2015.

B. van Rooyen, A. Menon, and R. Williamson. Learning with symmetric label noise: The importance of being unhinged. *Advances in Neural Information Processing Systems*, 2015.

G. Xun, Y. Li, W. X. Zhao, J. Gao, and A. Zhang. A correlated topic model using word embeddings. 2017.

H. Zhao, L. Du, W. Buntine, and M. Zhou. Inter and intra topic structure learning with word embeddings. 2018.

Journal of Machine Learning Research 20 (2019) 1-56 Submitted 10/17; Revised 2/19; Published 2/19

Utilizing Second Order Information in Minibatch Stochastic Variance Reduced Proximal Iterations

Jialei Wang JIALEI@CS.UCHICAGO.EDU
Department of Computer Science
University of Chicago
Chicago, IL, 60637

Tong Zhang TONGZHANG@TONGZHANG-ML.ORG
Computer Science & Mathematics
Hong Kong University of Science and Technology
Hong Kong

Editor: Francis Bach

Abstract

We present a novel minibatch stochastic optimization method for empirical risk minimization of linear predictors. The method efficiently leverages both sub-sampled first-order and higher-order information, by incorporating variance-reduction and acceleration techniques. We prove improved iteration complexity over state-of-the-art methods under suitable conditions. In particular, the approach enjoys global fast convergence for quadratic convex objectives and local fast convergence for general convex objectives. Experiments are provided to demonstrate the empirical advantage of the proposed method over existing approaches in the literature.

1. Introduction

We consider the following optimization problem of finite-sums:

$$\min_w f(w) := \frac{1}{n}\sum_{i=1}^n f_i(w), \tag{1}$$

which arises in many machine learning applications. In this paper, we focus on regularized loss minimization of linear predictors, where $f_i(w) = \ell(w^\top x_i; b_i) + \frac{\lambda}{2}\|w\|^2$. Let $x_1, x_2, ..., x_n \in \mathbb{R}^d$ be feature vectors of n data samples, and $b_1, ..., b_n \in \mathbb{R}$ or $\{-1, 1\}$ be the corresponding target variables of interest. We note that (1) covers many popular models used in machine learning: for example when $\ell(w^\top x_i; b_i) = \frac{1}{2}(b_i - w^\top x_i)^2$ we obtain ridge regression; when $\ell(w^\top x_i; b_i) = \log(1 + \exp(-b_i w^\top x_i))$ we obtain ℓ_2 regularized logistic regression. Other examples such as SVMs can also be obtained by setting different loss functions $\ell(w^\top x_i; b_i)$. The ubiquitousness of such finite-sum optimization problems and the massive scale of modern datasets motivate significant research effort on efficient optimization algorithms to solve (1).

We denote $w^* = \arg\min_w f(w)$ to be the optimum of (1). For any approximate solution w, we say that it achieves ϵ-objective suboptimality if $f(w) - f(w^*) \le \epsilon$. For any $\epsilon > 0$, the runtime cost of optimization algorithms to find an ϵ-suboptimal minimizer typically depends on the target accuracy ϵ, as well as the conditions of the problem (1). Throughout the paper, we use the following notion of strong convexity and smoothness to discuss the iteration complexity of various optimization algorithms.

Definition 1 *A function $f_i(w)$ is L-smooth with respect to w if $f_i(w)$ is differentiable and its gradient is L-Lipschitz continuous, i.e. we have*

$$\left\|\nabla f_i(w) - \nabla f_i(w')\right\| \le L\left\|w - w'\right\|, \quad \forall w, w' \in \mathbb{R}^d.$$

A function $f_i(w)$ is λ-strongly convex with respect to w, if we have

$$f_i(w) \ge f_i(w') + \langle \nabla f_i(w'), w - w'\rangle + \frac{\lambda}{2}\left\|w - w'\right\|^2, \quad \forall w, w' \in \mathbb{R}^d.$$

It is well-known that a consequence of L-smoothness is the following quadratic upper bound for $f_i(w)$:

$$f_i(w) \le f_i(w') + \langle \nabla f_i(w'), w - w'\rangle + \frac{L}{2}\left\|w - w'\right\|^2, \quad \forall w, w' \in \mathbb{R}^d.$$

In recent years there have been significant advances in developing fast optimization algorithms for (1), and we refer the readers to (Bottou et al., 2016) for a comprehensive survey of these developments. For large-scale problems in the form of (1), randomized methods are particularly efficient because of their low per iteration computation. Below we briefly review two lines of research: i) randomized variance reduced first-order methods; ii) randomized methods leveraging second-order information. Through out this section we focus on problems where each function $f_i(w), \forall i \in [n]$ is λ-strongly convex and L-smooth. For non-strongly convex or non-smooth objectives, we can use regularization or smoothing technique to reduce it to strongly convex and smooth objectives (Allen-Zhu and Hazan, 2016).

1.1 Variance Reduced First-order Methods

The key technique for developing fast stochastic first-order methods is variance reduction, which makes the variance of the randomized update direction approching zero when the iterate gets closer to the optimum. Representative methods of this category include SAG (Roux et al., 2012), SVRG (Johnson and Zhang, 2013), SDCA (Shalev-Shwartz and Zhang, 2013) and SAGA (Defazio et al., 2014), etc. In order to find a solution that reaches ϵ-suboptimality, these methods requires

$$\mathcal{O}\left(\left(n + \frac{L}{\lambda}\right)\log\left(\frac{1}{\epsilon}\right)\right) \tag{2}$$

calls to the first-order oracle to compute the gradient of an individual function. In the large condition number regime (e.g. $\frac{L}{\lambda} > n$), by using an acceleration technique (such as Catalyst (Shalev-Shwartz and Zhang, 2016; Frostig et al., 2015; Lin et al., 2015a), APCG (Lin et al., 2015b), SPDC (Zhang and Xiao, 2015), Katyusha (Allen-Zhu, 2016), etc), one can further improve the iteration complexity to

$$\mathcal{O}\left(\left(n + \sqrt{n\cdot\frac{L}{\lambda}}\right)\log\left(\frac{1}{\epsilon}\right)\right). \tag{3}$$

©2019 Jialei Wang and Tong Zhang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-594.html.

Recently, Woodworth and Srebro (2016) provided a first-order oracle lower bound for (1) which has the same iteration complexity as (3). This means that (3) is not improvable for general (high dimensional) problems, if the optimization procedure only uses the first order and proximal oracles of individual functions.

1.2 Leveraging Second-order Information

Second-order information is often useful in improving the convergence of optimization algorithms. However for large-scale problems, obtaining and inverting the exact Hessian matrix is often computational expensive, making vanilla Newton methods not well suited in solving (1). Therefore, there have been emerging studies in designing randomized algorithms that effectively utilize the *approximated* Hessian information. One line of such research is the sub-sampled Newton method which approximates the Hessian matrix based on a sub-sampled minibatch of data. Several work, such as (Byrd et al., 2011; Erdogdu and Montanari, 2015; Roosta-Khorasani and Mahoney, 2016a,b; Bollapragada et al., 2016), established local linear convergence rates for different variants of sub-sampled Newton methods, when the size of the sub-sampled data is large enough. (Xu et al., 2016) considered non-uniform sampling in constructing the stochastic Hessian matrix, and showed that weighted sampling according to individual smoothness or leverage score can help constructing better approximation of Hessian, which improves the convergence. (Pilanci and Wainwright, 2017) discussed how to use sketching instead of sub-sampling to approximate the Hessian matrix. All of the aforementioned methods employ the full first-order gradient, and require calling the second-order oracle to compute the Hessian and solving a resulting linear system at every iteration. Therefore the computational complexity (as compared in Table 2 of (Xu et al., 2016)) are often worse than stochastic algorithms such as SVRG.

Another line of research is to consider randomness in both first and second-order information to design lower per-iteration cost algorithms. In particular, (Byrd et al., 2016) considered combining minibatch SGD with Limited-memory BFGS (L-BFGS) (Liu and Nocedal, 1989) type update. Inspired by this, (Moritz et al., 2016; Gower et al., 2016; Wang et al., 2017b) proposed to combine variance reduced stochastic gradient with L-BFGS, and proved linear convergence for strongly convex and smooth objectives. Moreover, experiments in (Moritz et al., 2016; Gower et al., 2016; Wang et al., 2017b) also demonstrated superior empirical performance of this type of methods. However, theoretically it is hard to guarantee the quality of approximated Hessian using L-BFGS update, and thus the iteration complexity obtained by (Moritz et al., 2016; Gower et al., 2016; Wang et al., 2017b) can be pessimistic, and can be much worse than vanilla SVRG. Moreover, (Qu et al., 2015) proposed to incorporate curvature information in minibatch SDCA methods, and showed improved convergence over SDCA, but the method proposed in (Qu et al., 2015) involves solving a much more expensive subproblem than minibatch SDCA. Therefore the overall runtime benefit is still unclear. (Gonen et al., 2016) considered ridge regression problems specifically, and proposed to use sketching to compute the rank-k approximation of the Hessian matrix, based on which SVRG is ran with preconditioning. (Yang et al., 2016) suggested to use preconditioning as a preprocessing step for general stochastic optimization problems. (Agarwal et al., 2016) proposed to approximate the inverse Hessian matrix directly by sampling from its Taylor expansion. In terms of lower bound, (Arjevani and Shamir, 2016) showed under

some mild algorithmic assumptions, second-order oracle generally cannot improve the oracle complexity over first-order methods, for very high-dimensional problems.

Though lower bound have been established showing that the iteration complexity of accelerated SVRG methods cannot be improved in general, the establishments of these lower bounds are based on constructing very high-dimensional hard problems where dimension d can be much larger than sample size n. Such pessimistic situations are not consistent with practice, where second-order information is observed to be very helpful in improving the convergence speed. Therefore it is still interesting to analyze theoretically how second-order information can be helpful under suitable conditions, and more importantly, how to design more efficient methods that make use of possibly noisy second-order information. In this paper, we make effort in solving empirical risk minimization problems with linear predictors more efficiently, and make the following contributions:

- We propose a novel approach that combines the advantages of variance-reduced first-order methods and sub-sampled Newton methods, in a efficient way that does not require expensive Hessian matrix computation and inversion. The method can be naturally extended to solve composite optimization problems with non-smooth regularization.

- We theoretically show under certain conditions the proposed approach can improve state-of-the-art iteration complexity, and empirically demonstrate it can substantially improve the convergence of existing methods. Note that though we only establish theoretical guarantees for empirical risk minimization of linear predictors, the methods it self can be applied to general finite-sum problems.

1.3 Notations

We use $[n]$ to denote the set $1, ..., n$. For a vector $w \in \mathbb{R}^d$, we use $\|w\|$ to denote its ℓ_2 norm and $\|w\|_1$ to denote its ℓ_1 norm. For a matrix X, we use $\|X\|_2$ to denote its spectral norm and $\|X\|_F$ to denote its Frobenius norm, and $\lambda_{\min}(X)$ to denote its minimum singular value, and $\mathrm{tr}(X)$ to denote the trace of X. We use I to denote an identity matrix. For two symmetric matrices A and B, we denote $A \succeq B$ if $A - B$ is positive semi-definite. For two sequences of numbers $\{a_n\}$ and $\{b_n\}$, we say $a_n = \mathcal{O}(b_n)$ if $a_n \leq Cb_n$ for n large enough, with some positive constant C. We use the notation $\widetilde{\mathcal{O}}(\cdot)$ to hide poly-log factors. We also use $a_n \lesssim b_n$ to denote $a_n = \mathcal{O}(b_n)$, $a_n \gtrsim b_n$ to denote $b_n = \mathcal{O}(a_n)$, $a_n \asymp b_n$ to denote $a_n = \mathcal{O}(b_n)$ and $b_n = \mathcal{O}(a_n)$.

1.4 Organization

The rest of the paper is organized as follows: we introduce the proposed method in Section 2, and present the main theoretical results in Section 3. In Section 4 we present the convergence analysis of inexact minibatch accelerated SVRG update as a key step in proving the main results, and this analysis might be of independent interest. We provide some empirical comparisons over existing approaches in Section 5, and conclusions in Section 6. Some detailed proofs are deferred to the Appendices.

2. Minibatch Stochastic Variance Reduced Proximal Iterations

In this section we present the proposed approach for minimizing (1), which naturally integrate both noisy first-order and higher-order information in an efficient way.

2.1 Algorithm Description

The high-level description is given in Algorithm 1, which consists of two main innovations:

- simultaneous incorporation of first-order and higher-order information via sub-sampling;
- allowing larger minibatch sizes through acceleration with inexact minimization.

We explain these features in the remaining of this section.

2.1.1 Sampling both First-order and Higher-order Information

We first discuss the main building block of the algorithm. Suppose at iteration t, given the previous iterate w_{t-1}, we consider the following update rule:

$$w_t = \arg\min_w \frac{1}{b}\sum_{i\in\bar{B}} f_i(w) - \left\langle \frac{1}{b}\sum_{i\in\bar{B}} \nabla f_i(w_{t-1}), w \right\rangle + \left\langle \frac{\eta}{b}\sum_{i\in B_t} \nabla f_i(w_{t-1}) + \eta\nabla f(\widetilde{w}) - \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w}), w \right\rangle + \frac{\bar{\lambda}}{2}\|w - w_{t-1}\|^2, \quad (6)$$

where $\bar{B}, B_t$ are some randomly sampled minibatch from $1, ..., n$, both with minibatch size b; η and $\bar{\lambda}$ are stepsize parameters, and $\widetilde{w}$ is a "reference" predictor used for reducing variance. The main feature of updating rule (6) is it considered both noisy first-order and higher-order information via minibatch sampling. In particular, the term $\left\langle \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \eta\nabla f(\widetilde{w}) - \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w}), w\right\rangle$ and $\frac{1}{b}\sum_{i\in\bar{B}} f_i(w) - \left\langle \frac{1}{b}\sum_{i\in\bar{B}}\nabla f_i(w_{t-1}), w\right\rangle$ in (6) incorporates variance-reduced first-order and noisy higher-order information of the objective function, respectively. To see this, second-order methods (e.g. Newton methods) often use the following second-order Taylor approximation of the function:

$$\frac{1}{b}\sum_{i\in\bar{B}} f_i(w) \approx \frac{1}{b}\sum_{i\in\bar{B}} f_i(w_{t-1}) + \left\langle \frac{1}{b}\sum_{i\in\bar{B}}\nabla f_i(w_{t-1}), w - w_{t-1}\right\rangle + \frac{1}{2}(w - w_{t-1})^\top \left(\frac{1}{b}\sum_{i\in\bar{B}}\nabla^2 f_i(w_{t-1})\right)(w - w_{t-1}). \quad (7)$$

Algorithm 1 MB-SVRP: Minibatch Stochastic Variance Reduced Proximal Iterations.

Parameters η, b, $\bar{\lambda}$, ν, ε.
Initialize $\widetilde{w}_0 = 0$.
Sampling Sampling b items from $[n]$ to form a minibatch $\bar{B}$.
for $s = 1, 2, \ldots, S$ **do**
  Calculate $\widetilde{v} = \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widetilde{w}_{s-1})$.
  Initialize $y_0 = w_0 = \widetilde{w}_s$.
  for $t = 1, 2, \ldots, T$ **do**
    Sampling b items from $[n]$ to form a minibatch B_t.
    Find w_t **that approximately solve** (4) (**Option-I**) or (5) (**Option-II**) such that $\widetilde{f}_t(w_t) - \min_w \widetilde{f}_t(w) \le \varepsilon$.

$$\textbf{Option-I}: \widetilde{f}_t(w) \coloneqq \frac{1}{b}\sum_{i\in\bar{B}} f_i(w) - \left\langle \frac{1}{b}\sum_{i\in\bar{B}}\nabla f_i(y_{t-1}), w\right\rangle + \left\langle \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(y_{t-1}) + \eta\widetilde{v} - \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w}_{s-1}), w\right\rangle + \frac{\bar{\lambda}}{2}\|w - y_{t-1}\|^2, \quad (4)$$

$$\textbf{Option-II}: \widetilde{f}_t(w) \coloneqq \frac{1}{2}(w - y_{t-1})^\top \left(\frac{1}{b}\sum_{i\in\bar{B}}\nabla^2 f_i(y_{t-1})\right)(w - y_{t-1}) + \left\langle \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(y_{t-1}) + \eta\widetilde{v} - \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w}_{s-1}), w\right\rangle + \frac{\bar{\lambda}}{2}\|w - y_{t-1}\|^2. \quad (5)$$

    Update:
$$y_t = w_t + \nu(w_t - w_{t-1}).$$
  end for
  Update $\widetilde{w}_s = w_m$.
end for
Return $\widetilde{w}_s$

If we plug the second-order approximation (7) into (6), we get the update rule as

$$w_t \approx \arg\min_w \frac{1}{2}(w - w_{t-1})^\top \left(\frac{1}{b}\sum_{i\in\bar{B}}\nabla^2 f_i(w_{t-1})\right)(w - w_{t-1}) + \frac{\bar{\lambda}}{2}\|w - w_{t-1}\|^2 + \left\langle \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \eta\nabla f(\widetilde{w}) - \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w}), w\right\rangle \quad (8)$$

$$= w_{t-1} - \eta\left(\frac{1}{b}\sum_{i\in\bar{B}}\nabla^2 f_i(w_{t-1}) + \bar{\lambda}I\right)^{-1}\left(\frac{1}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \nabla f(\widetilde{w}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w})\right), \quad (9)$$

which can be treated as a variant of sub-sampled Newton method, combined with the minibatch stochastic gradient with variance reduction. Moreover, when $f_i(w)$ is a quadratic function of w, the approximation (7) is exact (thus Option-I and Option-II in Algorithm 1 are coincident). Therefore the update rule (6) can be treated as a preconditioned minibatch SVRG update rule, with $\left(\frac{1}{b}\sum_{i\in\bar{B}}\nabla^2 f_i(w_{t-1})+\bar{\lambda}I\right)^{-1}$ as a precondition matrix. This is formalized in the proposition below.

Proposition 2 *When $f_i(w), \forall i$ is a quadratic function of w, then the update rule of* (4) *and* (5) *is equivalent to the following preconditioned minibatch SVRG update rule:*

$$w_t \leftarrow w_{t-1} - \eta\left(\bar{H}+\bar{\lambda}I\right)^{-1}\left(\frac{1}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \nabla f(\widetilde{w}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w})\right),$$

where

$$\bar{H} = \frac{1}{b}\sum_{i\in\bar{B}}\nabla^2 f_i(w)$$

is the sub-sampled Hessian matrix.

2.1.2 Connection to Existing Methods

Depending on the choice of these parameters, we observe that the update rule of (6) can be viewed as a generalization of several update rules proposed recently:

- When $b=1$, $\bar{\lambda}\to\infty$ and $\eta=\frac{\bar{\lambda}}{L}$, the term $\frac{1}{b}\sum_{i\in\bar{B}} f_i(w) - \left\langle\frac{1}{b}\sum_{i\in\bar{B}}\nabla f_i(w_{t-1}), w\right\rangle$ is negligible, thus (6) reduced to standard SVRG update (Johnson and Zhang, 2013):

$$w_t \leftarrow w_{t-1} - \frac{1}{L}\left(\nabla f_i(w_{t-1}) + \frac{1}{n}\sum_{i=1}^n\nabla f_i(\widetilde{w}) - \nabla f_i(\widetilde{w})\right).$$

- When $b>1$, $\bar{\lambda},\eta\to\infty$, then the term $\frac{1}{b}\sum_{i\in\bar{B}} f_i(w) - \left\langle\frac{1}{b}\sum_{i\in\bar{B}}\nabla f_i(w_{t-1}), w\right\rangle$ in (6) is negligible as well, (6) reduces to the following update:

$$w_t \leftarrow w_{t-1} - \frac{\eta}{\bar{\lambda}}\left(\frac{1}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \frac{1}{n}\sum_{i=1}^n\nabla f_i(\widetilde{w}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w})\right),$$

  which recovers the update rule of minibatch semi-stochastic gradient methods (a.k.a minibatch SVRG) (Konečnỳ et al., 2016).

- When $\bar{B}=B_t$, $b=1$, $\eta=1$, (6) reduced to stochastic variance reduced proximal iterations. More specifically, (6) will be reduced to:

$$w_t = \arg\min_w f_i(w) + \left\langle\frac{1}{n}\sum_{i=1}^n\nabla f_i(\widetilde{w}) - \nabla f_i(\widetilde{w}), w\right\rangle + \frac{\bar{\lambda}}{2}\|w-w_{t-1}\|^2, \tag{10}$$

  by checking the first order optimality condition it is easy to verify that (10) is performing the following update:

$$w_t \leftarrow w_{t-1} - \frac{1}{\bar{\lambda}}\left(\nabla f_i(w_t) + \frac{1}{n}\sum_{i=1}^n\nabla f_i(\widetilde{w}) - \nabla f_i(\widetilde{w})\right),$$

compared with standard SVRG update with stepsize $\frac{1}{\bar{\lambda}}$, we see the only difference is the gradient evaluation on f_i is based on "future" iterate w_t rather than the current iterate w_{t-1} in SVRG. Stochastic proximal iterations based on SAGA method has been proposed and analyzed in (Defazio, 2016) recently.

- When $B_t=\bar{B}$, $b>1$, $\eta=1$, and $\widetilde{w}=w_{t-1}$, (6) will be reduced to:

$$w_t = \arg\min_w \frac{1}{b}\sum_{i\in\bar{B}} f_i(w) + \left\langle\frac{1}{n}\sum_{i=1}^n\nabla f_i(w_{t-1}) - \frac{1}{b}\sum_{i\in\bar{B}}\nabla f_i(w_{t-1}), w\right\rangle + \frac{\bar{\lambda}}{2}\|w-w_{t-1}\|^2, \tag{11}$$

  (11) covers the update rule of DANE algorithm (Shamir et al., 2014), which is a communication-efficient distributed optimization algorithm. DANE uses the data on local machine to form the minibatch $\bar{B}$ and every round machines communicate the gradient vector based on their local data. As shown in (Shamir et al., 2014), DANE is provably communication more efficient than the first-order methods in certain scenarios.

- When $\bar{B}=B_t$, and we ignore the linear term $-\left\langle\frac{1}{b}\sum_{i\in\bar{B}}\nabla f_i(w_{t-1}), w\right\rangle + \left\langle\frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \eta\nabla f(\widetilde{w}) - \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w}), w\right\rangle$, then (6) reduces to the minibatch proximal iterations (Li et al., 2014; Wang et al., 2017a):

$$w_t = \arg\min_w \frac{1}{b}\sum_{i\in B_t} f_i(w) + \frac{\bar{\lambda}}{2}\|w-w_{t-1}\|^2.$$

  Such an update allows larger minibatch size than standard minibatch SGD, but without the linear correction term as we considered in (6). For such methods only sublinear convergence can be established for finite-sum problems.

2.1.3 Large Minibatch Size via Acceleration with Inexact Minimization

Using (6) as the building block, we propose the MB-SVRP (minibatch stochastic variance reduced proximal iterations) method, which is detailed in Algorithm 1. At the beginning of the algorithm, we form a minibatch $\bar{B}$ by sampling from $1,...,n$ and fix it for the whole optimization process[1]. Then following the SVRG method (Johnson and Zhang, 2013), the algorithm is divided to multiple stages, indexed by s. At each stage, we iteratively solve a minimization problem of the form (4) (Option I) or (5) (Option II) based on the randomly sampled minibatch B_t.

Compared with the simple update rule in (6), the major difference in Algorithm 1 is that we consider a momentum scheme by maintaining two sequences $\{w_t, y_t\}$, which is inspired by Nesterov's acceleration technique (Nesterov, 2004) and its recent SVRG variant (Nitanda, 2014). The main theoretical advantage over minibatch SVRG without momentum (Konečnỳ et al., 2016) is that such an acceleration allows us to use a much larger minibatch

1. We can also consider the varying $\bar{B}$ option by simply setting $\bar{B}=B_t$, which makes the algorithm simpler to implement. In practice, we observe no significant difference between this option and the pre-fixed $\bar{B}$. Here we consider fixed $\bar{B}$ mainly for the sake of simplicity in our theoretical analysis.

size (up to a size of $\mathcal{O}(\sqrt{n})$) without slowing down the convergence. In comparison the iteration complexity of standard minibatch SVRG will become worse when the minibatch size (Konečnỳ et al., 2016) increases. The advantage of allowing larger minibatch size using acceleration in SVRG type algorithms is in analogy to the situation of stochastic gradient descent (without variance reduction) type algorithms (Dekel et al., 2012; Cotter et al., 2011; Lan, 2012).

Since it is often expensive to find the exact minimizer of (4), we consider an approximate minimizer with objective suboptimality ε. When we choose the appropriate $\bar{\lambda}$ for $\widetilde{f}_t(w)$ to obtain enough strong convexity, the subproblems defined in Option-I and Option-II can be both be solved to high accuracy efficiently. For example, using SVRG-type algorithms (Johnson and Zhang, 2013) to solve (4) or (5) allows us to find an approximate solution with a small suboptimality ε with a constant number of passes over the data in the minibatch $\bar{B} \cup B_t$, if $\bar{\lambda}$ is set appropriately (discussed in details in our theoretical analysis). This approach avoids Hessian matrix construction and inversion operations, which are often computationally expensive for large-scale problems. Allowing error in gradient oracle has been analyzed in several first-order methods (Schmidt et al., 2011; Devolder et al., 2014) in the batch setting, but has been largely unexplored in stochastic gradient methods. The recent work of (Wang et al., 2017a) analyzed inexact minibatch proximal updates, which can be treated as an implicit minibatch stochastic gradient with errors. In Section 4, we establish convergence rate of *inexact*, *minibatched*, *accelerated* SVRG method, which might be of independent interest.

2.2 Extension to Composite Minimization

For many methods that try to incorporate second-order information such as sub-sampled Newton and L-BFGS type algorithms, it is not clear how to extend them to solving non-smooth composite problems. In contrast, the proposed approach can be easily extended to handle non-smooth composite minimization problems. Consider the minimization of:

$$F(w) := \frac{1}{n}\sum_{i=1}^{n} f_i(w) + g(w), \tag{12}$$

where the component functions $f_i(w), \forall i \in [n]$ are smooth and strongly convex, and $g(w)$ is a non-smooth regularizer. For example when $g(w) = \mu \|w\|_1$ and $f_i(w) = \frac{1}{2}(b_i - w^\top x_i)^2$, we get the Lasso objective (Tibshirani, 1996); when $g(w) = \mu \|w\|_1 + \frac{\lambda}{2}\|w\|^2$ and $f_i(w) = \log(1+\exp(-b_i w^\top x_i))$, we get the elastic net regularized logistic regression (Zou and Hastie, 2005).

We can easily modify Algorithm 1 to solve (12). The idea is rather straightforward: at each inner iteration, rather than (4), we simply solve the following minibatch composite minimization problem:

$$\begin{aligned} w_t \approx \arg\min_w \widetilde{F}_t(w) := &\frac{1}{\eta b}\sum_{i\in\bar{B}} f_i(w) - \left\langle \frac{1}{\eta b}\sum_{i\in\bar{B}} \nabla f_i(y_{t-1}), w \right\rangle \\ &+ \left\langle \frac{1}{b}\sum_{i\in B_t} \nabla f_i(y_{t-1}) + \widetilde{v} - \frac{1}{b}\sum_{i\in B_t} \nabla f_i(\widetilde{w}_{s-1}), w \right\rangle + \frac{\bar{\lambda}}{2\eta}\|w - y_{t-1}\|^2 + g(w). \end{aligned} \tag{13}$$

We slightly re-scale the approximated loss term to make sure that the relative weight between the approximated loss and the regularization term is correct. Since (13) is a standard finite-sum composite minimization problem, we could apply prox-SVRG or prox-SAGA to solve (13) efficiently when the term $\frac{\bar{\lambda}}{2\eta}\|w - y_{t-1}\|^2$ has sufficient strong convexity.

3. Convergence Analysis

In this section we present theoretical results for the proposed approach. As declared in the introduction, we focus on establishing convergence analysis on linear model empirical risk minimization problems because the Hessian structure is easier to interpret. However, the algorithms itself can be implemented for solving general finite-sum problems, and the theory can also be extended to handle such case. For non-qudratic convex objectives, our convergence analysis is local, i.e. we assume the initial and subsequent solutions are within a bounded region near the optimum, which is common in the analysis of second-order methods in optimization (Nesterov, 2004; Boyd and Vandenberghe, 2004). As will be discussed in the sequel, the size of the region depends on the condition number of the problem, as well as the Lipschitz parameter of the Hessian. For quadratic objectives, the radius of the region can be infinite, which implies global fast convergence.

In this section, for simplicity, we focus on the analysis of Option-II in Algorithm 1 which uses second-order approximation to construct each sub-problem (5). Similar results hold for Option-I because locally around the optimal solution, the two methods are equivalent. Note that for linear predictors $f_i(w)$ admits the form $f_i(w) = \ell(w^\top x_i; b_i) + \frac{\lambda}{2}\|w\|$, where the Hessian can be computed as $\nabla^2 f_i(y_{t-1}) = \ell''(y_{t-1}^\top x_i; b_i) x_i x_i^\top + \lambda I$. Moreover, we do not need to explicitly maintain the Hessian matrix: when calculating the stochastic gradient of equation (5), especially the gradient of $(1/2)(w - y_{t-1})^\top \nabla^2 f_i(y_{t-1})(w - y_{t-1})$, we get $\nabla^2 f_i(y_{t-1})(w - y_{t-1}) = \ell''(y_{t-1}^\top x_i; b_i)(x_i^\top (w - y_{t-1}))x_i + \lambda(w - y_{t-1})$, which can be computed in time $\mathcal{O}(d)$. Thus approximately solving objective equation (5) would only requires a few stochastic gradients calculation (or equivalent operations in $\mathcal{O}(d)$), we count total number of such gradient computation operations as a measure of Algorithms 1's efficiency in optimization.

Besides the strong convexity and smoothness, we also need the following Hessian Lipschitz condition.

Condition 1 *For each loss function $\ell(w^\top x_i; b_i), \forall i \in [n]$, its second derivative is $\bar{M}$-Lipschitz, i.e.*

$$|\ell''(w_1^\top x_i; b_i) - \ell''(w_2^\top x_i; b_i)| \le \bar{M}\|w_1 - w_2\|, \forall w_1, w_2 \in \mathbb{R}^d, \forall i \in [n].$$

Conditions of Lipschitz Hessians are typically not required for first-order methods such as SVRG and SAGA, but are typically required for second-order methods, such as Newton methods and sub-sampled Newton methods. Moreover, Lipschitz Hessians condition is often satisfied for popular loss functions used in machine learning such as logistic loss and square loss. We consider the component function in form of convex loss function of linear predictor: $f_i(w) = \ell(w^\top x_i; b_i) + \frac{\lambda}{2}\|w\|_2^2$. By Condition 1, combined with the condition that

maximum norm of data points are bounded: $\max_i \|x_i\|^2 \leq D$, we know Hessian matrix of each component loss function $f_i(w)$ is $M = \bar{M}D + \lambda$-Lipschitz, i.e.

$$\left\|\nabla^2 f_i(w_1) - \nabla^2 f_i(w_2)\right\| \leq M \|w_1 - w_2\|, \forall w_1, w_2 \in \mathbb{R}^d, \forall i \in [n].$$

For $f(w) = (1/n)\sum_i f_i(w)$, we have the following formula of Hessian matrix:

$$H_\lambda(w) = \frac{1}{n}\sum_{i=1}^n \nabla^2 f_i(w) = \frac{1}{n}\sum_{i=1}^n \ell''(w^\top x_i; b_i) x_i x_i^\top + \lambda I.$$

Let $H_\lambda(w_{t-1})$ be the Hessian matrix at w_{t-1}, and suppose we have an approximated Hessian

$$\bar{H}_{\widetilde{\lambda}}(w_{t-1}) = \frac{1}{b}\sum_{i\in\bar{B}} \ell''(w_{t-1}^\top x_i; b_i) x_i x_i^\top + \widetilde{\lambda} I,$$

the following proposition states that performing the preconditioned minibatch SVRG update is equivalent to the standard minibatch SVRG in a linear transformed (preconditioned) space:

Proposition 3 *Considering the following minibatch SVRG type update when solving* $\min_w f(w)$*:*

$$w_t \leftarrow w_{t-1} - \eta \bar{H}_{\widetilde{\lambda}}(w_{t-1})^{-1}\left(\frac{1}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widetilde{w}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w})\right), \quad (14)$$

which is equivalent to solving a minimization problem with respect to z*:*

$$\min_z f(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*) z)$$

via the variable transform $z = \bar{H}_{\widetilde{\lambda}}^{1/2}(w^*)w$*. Moreover, the update rule* (14) *is equivalent to the following update on* z*:*

$$\begin{aligned} z_t \leftarrow & z_{t-1} - \eta\left(\frac{1}{b}\sum_{i\in B_t}\nabla_z f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z_{t-1}) + \frac{1}{n}\sum_{i=1}^n \nabla_z f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\widetilde{z}) - \frac{1}{b}\sum_{i\in B_t}\nabla_z f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\widetilde{z})\right) \\ & + \eta \bar{H}_{\widetilde{\lambda}}^{1/2}(w^*)\left(\bar{H}_{\widetilde{\lambda}}(w^*)^{-1} - \bar{H}_{\widetilde{\lambda}}(w_{t-1})^{-1}\right)\left(\frac{1}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widetilde{w}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w})\right). \end{aligned} \quad (15)$$

With the above proposition, and Proposition 2, we can see that the proposed updates (4) and (5) are implicitly performing inexact minibatch SVRG update on the following transformed problem:

$$\min_z f(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z) := \frac{1}{n}\sum_{i=1}^n \ell(z^\top \bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)x_i; b_i) + \lambda\left\|\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z\right\|^2, \quad (16)$$

by choosing $\bar{\lambda} = \widetilde{\lambda} - \lambda$. In the following, we focus on discussing the choice of $\widetilde{\lambda}$, and $\bar{\lambda}$ can be settled accordingly. As a straightforward extension, it is also easy to see that using the acceleration technique as Algorithm 1 is equivalent to performing inexact *accelerated* minibatch SVRG update, similar to the exact update analyzed in (Nitanda, 2014). The analysis of inexact algorithms are presented in Section 4.

The benefit of the transformed problem is that when $\bar{H}_{\widetilde{\lambda}}(w^*) \approx H_\lambda(w^*)$, the condition number of the new problem (16) becomes smaller than that of the original problem. As a consequence, when the computational cost of update (14) is cheap, we expect improved convergence and runtime guarantees for solving the original problem. Before analyzing the condition number, we state the following lemma, which connects the inexactness in w space when performing the update (8) (and (14)) to the inexactness in z space when performing the update (15).

Lemma 4 *Let* $\widetilde{f}_t(w)$ *be the function to be minimized in* (8)*, and* $w_t^* = \arg\min_w \widetilde{f}_t(w)$ *to be its exact minimizer. Let*

$$\bar{z}_t = z_{t-1} - \eta\left(\frac{1}{b}\sum_{i\in B_t}\nabla_z f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z_{t-1}) + \frac{1}{n}\sum_{i=1}^n \nabla_z f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\widetilde{z}) - \frac{1}{b}\sum_{i\in B_t}\nabla_z f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\widetilde{z})\right),$$

be the exact minibatch SVRG update solution in the preconditioned space, and $z_t = \bar{H}_{\widetilde{\lambda}}^{1/2}(w^*)w_t$ *be the actual update in precondition space by approximately solving* (8)*. Let* $\vartheta_t = \frac{1}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widetilde{w}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w})$*, We have the following upper bound on* $\|z_t - \bar{z}_t\|$*:*

$$\|z_t - \bar{z}_t\| \leq \sqrt{\frac{2(\widetilde{f}_t(w_t) - \widetilde{f}_t(w_t^*))(L + \widetilde{\lambda})}{\lambda + \widetilde{\lambda}}} + \frac{\eta M \|w_{t-1} - w^*\| \|\vartheta_t\|}{\widetilde{\lambda}^{3/2}}.$$

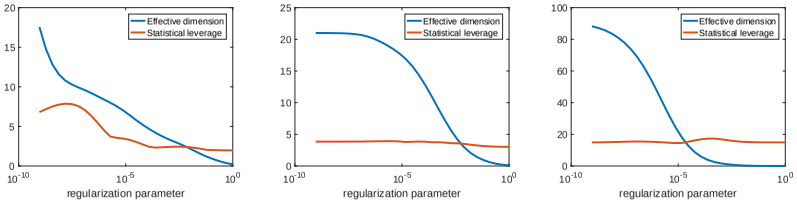
Remark 5 *Lemma 4 states that the inexactness in the minibatch SVRG update in the* z *space can be decomposed to two sources, the first source is the fact that we solve the minimization problem of objective* $\widetilde{f}_t(w)$ *inexactly, and the second source is the fact that the Hessian matrix is changing at different locations. Nevertheless, we can fix the* z *space that use Hessian matrix at optimum* $\bar{H}_{\widetilde{\lambda}}(w^*)$ *as the preconditioner. For quadratic objectives where* $M = 0$*, the second error term becomes zero.*

3.1 Improved Condition Number

In this section we analyze the condition number in the "preconditioned" space (16), where $\bar{B}$ is sampled uniformly from $1, ..., n$. For the analysis, we introduce two notions which describes the global and local properties of data. The following definitions of effective dimension and bounded statistical leverage extend the ones defined in (Hsu et al., 2014) which were used to analyzing the generalization performance of ridge regression.

Definition 6 ***(Effective dimension)*** *Let the* $\lambda_1, ..., \lambda_d$ *be the top-d eigenvalues of* $H_0(w^*) = (1/n)\sum_{i=1}^n \ell''(w^{*\top}x_i; b_i)x_i x_i^\top$*, define the effective dimension* $d_{\widetilde{\lambda}}$ *(for some* $\widetilde{\lambda} \geq 0$*) of* $H_0(w^*)$ *being*

$$d_{\widetilde{\lambda}} = \sum_{j=1}^d \frac{\lambda_j}{\lambda_j + \widetilde{\lambda}}.$$



covtype
$n = 581,012,\ d = 54$

svmguide3
$n = 1,243,\ d = 21$

year
$n = 463,715,\ d = 91$

Figure 1: Effective dimension and maximum statistical leverage with different regularization parameters $\widetilde{\lambda}$ on some empirical datasets for ridge regression problems.

We see $d_{\widetilde{\lambda}}$ is a decreasing function of $\widetilde{\lambda}$, and $d_0 = d$ when $\widetilde{\lambda} = 0$. When the spectrum of $\frac{1}{n}\sum_{i=1}^n \ell''(w^{*\top}x_i; b_i)x_i x_i^\top$ is decaying very fast, the effective dimension $d_{\widetilde{\lambda}}$ can be significantly smaller than d for moderate $\widetilde{\lambda}$. The following lemma demonstrated this for Hessian matrices with fast decaying spectrum, which can be well approximated by low-rank matrices.

Lemma 7 *We have the following upper bound for $d_{\widetilde{\lambda}}$:*

$$d_{\widetilde{\lambda}} \leq \min_k \left\{ k + \frac{\sum_{j>k} \lambda_j}{\widetilde{\lambda}} \right\}.$$

As an example of data with fast decaying spectrum, consider generalized linear predictors in a Hilbert space induced by a Gaussian kernel $K(x_i, x_i') = \exp(-\|x - x_i'\|^2/(2\sigma^2))$ where σ is a bandwidth parameter. The covariance matrix of the induced feature space has exponentially decaying eigenvalues with $\lambda_j \leq c_1 \exp(-c_2 j^2)$, where c_1 and c_2 are constants (Zhang et al., 2015). With exponentially decaying eigenvalues, it is shown in Corollary 5 (Zhang et al., 2015) that $d_{\widetilde{\lambda}} \leq \mathcal{O}(\sqrt{\log(1/\widetilde{\lambda})})$ even though the Gaussian kernel maps the original data to an infinite dimensional space.

The following notion of statistical leverage have been used in regression analysis (Chatterjee and Hadi, 2009; Hsu et al., 2014) and matrix approximation (Mahoney et al., 2011).

Definition 8 ***(Statistical leverage at $\widetilde{\lambda}$)*** *Let $H_{\widetilde{\lambda}}(w^*) = \frac{1}{n}\sum_{i=1}^n \ell''(w^{*\top}x_i; b_i)x_i x_i^\top + \widetilde{\lambda} I$, we say the statistical leverage of data matrix X is bounded by $\rho_{\widetilde{\lambda}}$ at $\widetilde{\lambda}$ if*

$$\max_{i\in[n]} \frac{\left\| H_{\widetilde{\lambda}}^{-1/2}(w^*)\ell'(w^{*\top}x_i; b_i)^{1/2} x_i \right\|}{\sqrt{(1/n)\sum_{j=1}^n \left\| H_{\widetilde{\lambda}}^{-1/2}(w^*)\ell''(w^{*\top}x_j; b_j)^{1/2} x_j \right\|^2}} \leq \rho_{\widetilde{\lambda}}. \tag{17}$$

For ridge regression problems, the above definition is slightly different from the one used in (Hsu et al., 2014) in the sense that the empirical Hessian matrix $H_{\widetilde{\lambda}}$ in (17) is replaced by the population Hessian matrix $\mathbb{E}\left[x_i^\top x_i\right] + \widetilde{\lambda} I$ in (Hsu et al., 2014). When the sample size n is large, the differences in these two definitions are minor. As argued in (Hsu et al., 2014), when x_i is i.i.d. randomly drawn from sub-Gaussian distributions, $\left(\mathbb{E}\left[x_i^\top x_i\right]\right)^{-1/2} x_i$ is isotropic. In this case the statistical leverage only grows logarithmically with dimension. For the empirical statistical leverage considered in (17), if we consider the un-regularized form (when $\widetilde{\lambda} = 0$), then Definition 8 is closely related to the leverage score used in (Mahoney et al., 2011; Drineas and Mahoney, 2017), where the leverage score for x_i is defined as $x_i^\top \left((1/n)\sum_{j=1} x_j x_j^\top\right)^{-1} x_i$, and thus statistical leverage is just the ratio between "maximum leverage score" and "average leverage score". Though the in worse case $\rho_{\widetilde{\lambda}}$ can be as large as $\sqrt{n}$, in many situations it is small. For instance, if the data is generated by random Gaussian distributions, or when we apply certain random projections on the data, the (projected) data matrix is known to have almost uniform leverage scores (i.e. the statistical leverage parameter in (17) can be upper bounded by $\mathcal{O}(\log d)$, see Lemma 70 in (Drineas and Mahoney, 2017) for example).

Our theoretical analysis relies on the condition that the two quantities defined above are not too large. This condition often holds in practice, as shown in Figure 1, which plots the effective dimension and the statistical leverage for several real world datasets with varying regularization parameters.

The convergence of stochastic gradient based algorithms for minimizing the objective (16) depends on two important quantities: strong convexity and smoothness. The strong convexity for problem (16) (with respect to z) is

$$\min_z \lambda_{\min}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*) \left(\frac{1}{n}\sum_{i=1}^n \ell''(z^\top \bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)x_i; b_i)x_i x_i^\top + \lambda I \right) \bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*) \right), \tag{18}$$

which is a property of the overall objective function rather than individual component. However, we must consider the smoothness parameter for individual function $f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z)$ (with respect to z), which in our context, is

$$\max_i \max_z \lambda_{\max}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*) \left(\ell''(z^\top \bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)x_i; b_i)x_i x_i^\top + \lambda I \right) \bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*) \right). \tag{19}$$

By the variable transformation $w = \bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z$, we get $\ell''(z^\top \bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)x_i; b_i) = \ell''(w^\top x_i; b_i)$. Moreover, we consider a local region where w is close to w^* measured by diameter R: $\|w - w^*\| \leq R$, and study the local strong convexity and smoothness quantities((18) and (19)).

Both of these quantities are related to how close is the constructed Hessian approximation $\bar{H}_{\widetilde{\lambda}}(w^*)$ to the true Hessian $H_{\widetilde{\lambda}}(w^*)$ in spectral norm. The following lemma bound this quantity using matrix concentration.

Lemma 9 *If $\bar{B}$ is formed by uniform sampling with replacement from $[n]$, then we have the following concentration bound, with probability at least $1 - \delta$,*

$$\left\| H_{\widetilde{\lambda}}^{-1}(w^*) \left(\frac{1}{b}\sum_{i\in\bar{B}} \ell''(w^{*\top}x_i; b_i)x_i x_i^\top - \frac{1}{n}\sum_{i=1}^n \ell''(w^{*\top}x_i; b_i)x_i x_i^\top \right) \right\|_2 \leq 2\log\left(\frac{d}{\delta}\right) \cdot \sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b}}.$$

3.1.1 Strong Convexity

Based on above lemma, we have the following lower bound of the strong convexity for (16), specified in the following lemma.

Lemma 10 *If we set $\widetilde{\lambda}$ such that $\widetilde{\lambda} \geq \lambda$, then with probability at least $1-\delta$ over the random choice of $\bar{B}$ to form $\bar{H}_{\widetilde{\lambda}}$, we have*

$$\min_{\|w-w^*\|\leq R} \lambda_{\min}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)H_\lambda(w)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\right) \geq \frac{\lambda}{\widetilde{\lambda}} \frac{1}{1+2\log(d/\delta)\sqrt{(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}})/b}(1+MR/\widetilde{\lambda})+MR/\widetilde{\lambda}}.$$

3.1.2 Smoothness

Next we explore the local smoothness parameter of (16). Using the variable transform $w = \bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z$, if we consider the individual smoothness parameter (with respect to z) around the neighborhood of w^* with diameter R, it can be upper bounded by

$$\max_{i\in[n]} \max_{\|w-w^*\|\leq R} \lambda_{\max}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\left(\ell''(w^\top x_i;b_i)x_i x_i^\top + \lambda I\right)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\right).$$

Since

$$\begin{aligned}
&\lambda_{\max}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\left(\ell''(w^\top x_i;b_i)x_i x_i^\top + \lambda I\right)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\right)\\
&\leq \lambda_{\max}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\left(\ell''(w^\top x_i;b_i)x_i x_i^\top\right)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\right) + \lambda_{\max}\left(\lambda\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)\right)\\
&\leq \ell''(w^\top x_i;b_i)x_i^\top \bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i + \frac{\lambda}{\widetilde{\lambda}} \qquad (20)
\end{aligned}$$

The most straightforward way to upper bound $\ell''(w^\top x_i;b_i)x_i^\top \bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i$ is

$$\begin{aligned}
&\max_{i\in[n]} \max_{\|w-w^*\|\leq R} \ell''(w^\top x_i;b_i)x_i^\top \bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i\\
&\leq \max_{i\in[n]} \max_{\|w-w^*\|\leq R} \ell''(w^\top x_i;b_i)\|x_i\|^2 \lambda_{\min}(\bar{H}_{\widetilde{\lambda}}^{-1}(w^*))\\
&= \max_{i\in[n]} \max_{\|w-w^*\|\leq R} \frac{\ell''(w^\top x_i;b_i)\|x_i\|^2}{\widetilde{\lambda}} = \frac{L}{\widetilde{\lambda}}.
\end{aligned}$$

In this way, the condition number after preconditioning becomes

$$\mathcal{O}\left(\frac{4\widetilde{\lambda}}{\lambda}\cdot\frac{L}{\widetilde{\lambda}}\right) = \mathcal{O}\left(\frac{L}{\lambda}\right),$$

which didn't show any advantage of using preconditioning. This is not surprising because it is known that for high dimensional problems, the worse case behavior of second order methods is no better than that of the first order methods.

In the lemma below, we provide an improved analysis of smoothness which is based on the notion of effective dimension (Definition 6) and statistical leverage (Definition 8).

Lemma 11 *If we choose $\widetilde{\lambda} \geq \lambda$ and $b \geq 16\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\log^2(d/\delta)$, then with probability at least $1-\delta$ over the random choice of $\bar{B}$ to form $\bar{H}_{\widetilde{\lambda}}(w^*)$, we have*

$$\max_{i\in[n]} \max_{\|w-w^*\|\leq R} \ell''(w^\top x_i;b_i)x_i^\top \bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i \leq 2\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} + \frac{MR}{\widetilde{\lambda}}.$$

Based on above analysis, we have the following corollary states the local individual smoothness parameter of (16).

Corollary 12 *If we choose $\widetilde{\lambda} \geq \lambda$ and $b \geq 16\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\log^2(d/\delta)$, and $R \leq \frac{\widetilde{\lambda}}{M}$. Then with probability at least $1-\delta$ over the random choice of $\bar{B}$ to form $\bar{H}_{\widetilde{\lambda}}(w^*)$, we have the following upper bound hold for the local smoothness parameter*

$$\max_{i\in[n]} \max_{\|w-w^*\|\leq R} \lambda_{\max}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\left(\ell''(w^\top x_i;b_i)x_i x_i^\top + \lambda I\right)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\right) \leq 2\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} + 2.$$

Proof Combining (20) and Lemma 11, as well as the conditions of $\widetilde{\lambda}$ and R stated in the corollary we get the desired results. ■

Combining Lemma 10 and Corollary 12 we get the following result about the condition number for (16). If we choose $\widetilde{\lambda}$ and b that satisfy

$$\widetilde{\lambda} \geq \max\left\{\lambda, \frac{L}{b}\right\}, \quad b \geq 16\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\log^2\left(\frac{d}{\delta}\right), \quad R \leq \min\left\{\frac{1}{4ML}, \frac{\widetilde{\lambda}}{4M}\right\},$$

then with probability at least $1-\delta$, the condition number for stochastic gradient algorithms after preconditioning scales as $\frac{4\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\widetilde{\lambda}}{\lambda}$, more specifically when $\widetilde{\lambda} = \max\{\lambda, \frac{L}{b}\}$, then the condition number of (16) can be upper bounded by:

$$\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \max\left\{4, \frac{4L}{\lambda b}\right\},$$

which improves the original condition number L/λ by a factor of at least $\widetilde{\mathcal{O}}(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}/b)$.

3.2 Improved Runtime Guarantee

Based on the above analysis, we have the following main results stating the iteration complexity of MB-SVRP algorithms on general convex objectives (1).

Theorem 13 *Consider Algorithm 1 on problems with L-smooth and λ-strongly convex functions that satisfied Condition 1 with Hessian Lipschitz parameter M. Suppose we sample minibatch $\bar{B}$ uniformly from $1,...,n$, and set the tuning parameters as*

$$\widetilde{\lambda} = \max\left\{\lambda, \frac{L}{b}\right\}, \quad b \asymp \min\left\{n, \rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\left(\frac{L}{\lambda}\right)^{1/3}\right\}, \quad \eta \asymp \min\left\{\frac{b^3\lambda}{(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}})^2 L}, \frac{1}{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}\right\},$$

and suppose we start from the initialization point that is sufficiently close to the optimum:

$$\|w_0 - w^*\| \leq R = \mathcal{O}\left(\frac{\lambda^4}{L^2 M}\right). \tag{21}$$

Given $\epsilon > 0$, and let

$$\varepsilon \leq \frac{1}{10^5} \cdot \left(\frac{\lambda}{L}\right)^7 \epsilon. \tag{22}$$

Then for the MB-SVRP algorithm to find the approximate solution $\widetilde{w}_s$ of (1) that reaches the expected ϵ-objective suboptimality

$$\mathbb{E} f(\widetilde{w}_s) - \min_w f(w) \leq \epsilon,$$

we can set

$$T \leq \mathcal{O}\left(\max\left\{\left(\frac{L}{\lambda}\right)^{1/3}, \frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L}{\lambda n^2}\right\}\right),$$

and uses $S \leq \mathcal{O}(\log(1/\epsilon))$ total stages in Algorithm 1. Thus the total number of oracle calls to solve (8) is no more than

$$\mathcal{O}\left(\max\left\{\left(\frac{L}{\lambda}\right)^{1/3}, \frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L}{\lambda n^2}\right\} \cdot \log\left(\frac{1}{\epsilon}\right)\right).$$

Remark 14 *There are two source of randomness in Algorithm 1: i) sampling $\bar{B}$ to form the precondition minibatch before the iterations begins; ii) sampling B_t during the iterations. In particular, sampling $\bar{B}$ affects the new condition number as Section 3.1 analyzed, while sampling B_t affects the precondition SVRG convergence. We state the results about the randomness in $\bar{B}$ in high-probability form as Lemma 10 and Corollary 12; while state the randomness in B_t in expectation results, it is also possible to obtain high-probability results for randomness in B_t by using Markov's inequality, for example as Corollary 1 in (Xiao and Zhang, 2014).*

Note that in Algorithm 1, we can use SVRG to solve each subproblem in (8). This leads to the following result.

Corollary 15 *Assume that the conditions of Theorem 13 hold. Moreover, if we use SVRG to solve each subproblem in (8) up to the suboptimality ε, then the total number of gradient evaluations used in the whole MB-SVRP algorithm can be upper bounded by*

$$\mathcal{O}\left(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \max\left\{\left(\frac{L}{\lambda}\right)^{2/3}, \frac{L}{\lambda n}\right\} \cdot \log\left(\frac{L}{\lambda}\right) \cdot \log^2\left(\frac{1}{\epsilon}\right) + n \cdot \log\left(\frac{1}{\epsilon}\right)\right).$$

We make the following remarks about Theorem 13 and Corollary 15.

Remark 16 *The solution accuracy requirement in (22) (in practice we can monitoring ε via $\|\nabla \widetilde{f}_t(w)\|$ since $\widetilde{f}_t(w)$ is $\widetilde{\lambda}$-strongly convex) is conservative, but does not affect the overall iteration complexity because solving (4) and (5) to high precision only introduces an additional logarithmic factor using SVRG. In practice, we observe that using only one pass SVRG warm started from previous iterate performs well (please refer to the experiments).*

Remark 17 *For quadratic objectives, where $M = 0$, the local fast region defined in (21) is the whole space, implying global fast convergence for quadratic objectives. Moreover, in experiments, we observe that even for the general non-quadratic objectives, the proposed MB-SVRP method also converges fast globally. Therefore the size of the local fast convergence region (21) established here might be too conservative. We leave the investigation on improving the size of the local fast convergence region to future work.*

Remark 18 *Corollary 15 states that the iteration complexity of Algorithm 1 is*

$$\widetilde{\mathcal{O}}\left(n + \rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \max\left\{\left(\frac{L}{\lambda}\right)^{2/3}, \frac{L}{\lambda n}\right\}\right),$$

where $\widetilde{\mathcal{O}}(\cdot)$ hides the minor poly logarithmic factors. When $\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}$ is small so that it can be treated as a constant, then the iteration complexity of MB-SVRP improves over standard SVRG by a factor of $\min\left\{n, \left(\frac{L}{\lambda}\right)^{1/3}\right\}$ when the condition number $\frac{L}{\lambda}$ is much larger than n. Even compared with accelerated methods (such as SVRG equipped with catalyst acceleration (Shalev-Shwartz and Zhang, 2016; Frostig et al., 2015; Lin et al., 2015a)), it can sometimes be better (when $n < L/\lambda < n^2$, see Table 1 for details).

Remark 19 *Though the optimal parameter choice stated in Theorem 13 might depend on the property of the Hessian matrix at w^*, the algorithm actually works for a wide range of parameters without the knowledge of w^* because i) for quadratic objectives the Hessian matrix does not change with w so the optimal parameter choice will not depend on w^*; ii) we only requires some constants $\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}$ to bound certain quantities of the Hessian matrix at w^*, but do not need exactly where w^* is; iii) if we are uncertain about the data we can choose some conservative parameters (as discussed in Section 3.1) to ensure the method converges at least as fast as vanilla SVRG.*

Remark 20 *When $\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}$ is small such that can be treated as constant, then the optimal choice of b according to our theory is often smaller than n, except the case where the condition number is extremely large: $L/\lambda \gtrsim (n/(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}))^3$. Moreover, when combine the proposed approach with Catalyst acceleration as in Section 3.3, the condition number in each proximal point problem can always be controlled to be not too large, in which case the optimal batch size b is smaller than n.*

Remark 21 *MB-SVRP method is able to improve the iteration complexity over SVRG when $\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}$ is small, thus it is also possible to use MB-SVRP itself to solve the minibatch subproblem (4). Such a nested approach allows us to choose a even smaller $\widetilde{\lambda}$, thus can further reduce the dependency on condition number in the iteration complexity. However, this is at the cost of increasing the dependency on $\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}$ and more complicated implementation. We will leave such investigation to future research.*

3.3 With Catalyst Acceleration

Algorithm 2 Acc-MB-SVRP: Accelerated MB-SVRP method.

Initialize $w_0 = z_0 = 0$.
for $r = 1, 2, \ldots$ **do**
Call MB-SVRP algorithm 1 to approximately solve

$$w_r \approx \arg\min_w f(w) + \frac{\gamma}{2}\|w - z_{r-1}\|^2. \tag{23}$$

Update

$$z_r = w_r + \nu_r(w_r - w_{r-1}).$$

end for
Return w_r.

Table 1: Comparison of iteration complexity of various finite-sum quadratic optimization algorithms when effective dimension and statistical leverage are bounded, where we compare the different relative scale of condition number $\kappa = L/\lambda$ and sample size n, ignoring logarithmic factors.

	$\kappa \le n$	$\kappa = n^{4/3}$	$\kappa = n^{3/2}$	$\kappa = n^2$	$\kappa = n^3$
SVRG	n	$n^{4/3}$	$n^{3/2}$	n^2	n^3
MB-SVRP	n	$\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \cdot n$	$\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \cdot n$	$\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \cdot n^{3/2}$	$\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \cdot n^{5/2}$
Acc-SVRG	n	$n^{7/6}$	$n^{5/4}$	$n^{3/2}$	n^2
Acc-MB-SVRP	n	$\left(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\right)^{1/2} \cdot n$	$\left(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\right)^{1/2} \cdot n$	$\left(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\right)^{1/2} \cdot n^{5/4}$	$\left(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\right)^{1/2} \cdot n^{7/4}$

Corollary 15 stated that if the condition number is not too large: $\frac{L}{\lambda} \le n^{5/4}$, then MB-SVRP only requires logarithmic passes over data to find a solution with high accuracy. When the condition number is large, MB-SVRP can still be slow. By using the accelerated proximal point framework proposed in (Shalev-Shwartz and Zhang, 2016; Frostig et al., 2015; Lin et al., 2015a), it is possible to obtain an accelerated convergence rate which has milder dependence on condition number, the algorithm is outlined in Algorithm 2, which iteratively, approximately call the original algorithm MB-SVRP to solve an augmented proximal point problem. The main iteration complexity is stated in the theorem below.

Theorem 22 *For ill-condition problems where $L/\lambda \ge n^{3/2}$, if we set the parameter $\gamma \asymp \frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L}{n^{3/2}} - \lambda$, Algorithm 2 has iteration complexity of*

$$\widetilde{\mathcal{O}}\left(\left(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\right)^{1/2} \cdot n^{1/4} \left(\frac{L}{\lambda}\right)^{1/2}\right).$$

Remark 23 *From theorem 22 we see the iteration complexity of accelerated MB-SVRP grows at a square root rate with respect to the condition number, which is much better than that of the non-accelerated algorithms, and this is similar to the behavior of accelerated SVRG algorithms. Moreover, accelerated MB-SVRP improves the iteration complexity of accelerated SVRG by a factor of $n^{1/4}$ when $\sqrt{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}$ is small.*

4. Convergence Analysis of Inexact Accelerated Minibatch SVRG

Algorithm 3 IMBA-SVRG: Inexact Minibatch Accelerated SVRG Method.

Parameters $\alpha = \sqrt{\eta\lambda/2}$.
Initialize $\widetilde{w}_0 = 0$.
for $s = 1, 2, \ldots, S$ **do**
Calculate $\widetilde{v} = \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widetilde{w}_{s-1})$.
Initialize $y_0 = w_0 = \widetilde{w}_s$.
for $t = 1, 2, \ldots, T$ **do**
Sampling b items from $[n]$ to form a minibatch B_t.
Inexact update w_t such that

$$w_t = \bar{w}_t + \xi_t, \tag{24}$$

where:

$$\bar{w}_t = y_{t-1} - \eta\left(\frac{1}{b}\sum_{i\in B_t}\nabla f_i(y_{t-1}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w}_{s-1}) + \widetilde{v}\right).$$

Update

$$y_t = w_t + \left(\frac{1-\alpha}{1+\alpha}\right)(w_t - w_{t-1}).$$

end for
Update $\widetilde{w}_s = w_m$.
end for
Return $\widetilde{w}_s$

As discussed in Propositions 2 and 3, the proposed MB-SVRP method (Algorithm 1) is essentially perform the inexact minibatch accelerated SVRG updates in the preconditioned space using $\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)$ as the preconditioner. Thus, the main results established in Section 3 rely on the analysis for inexact, minibatch, accelerated SVRG update (IMBA-SVRG Algorithm 3), where in each stochastic step, we allow a small error ξ_t in the updating. In this section, we show that as long as the inexactness at each iteration is small enough, IMBA-SVRG can use a large minibatch size without slowing down the convergence.

Theorem 24 *Consider the IMBA-SVRG algorithm. If we choose the stepsize as*

$$\eta = \min\left\{\frac{b^2\lambda}{6400L^2}, \frac{1}{8L}\right\},$$

and when the deviation ξ_t satisfies $\forall 1 \le t \le T$

$$\|\xi_t\|^2 \le \frac{2\lambda^2\eta^3\sqrt{\lambda\eta}}{15}(f(\widetilde{w}_{s-1}) - f(w^*)), \tag{25}$$

and the number of iterations T satisfies

$$T \ge \frac{10}{9\sqrt{\lambda\eta}}\log(36),$$

then

$$\mathbb{E}\left[f(\widetilde{w}_s) - f(w^*)\right] \le \frac{1}{2}\left[f(\widetilde{w}_{s-1}) - f(w^*)\right].$$

Remark 25 *Depending on the relative magnitude of minibatch size b and condition number $\frac{L}{\lambda}$, the overall iteration complexity can be divided into the following two situations.*

- $b \le 20\sqrt{\frac{2L}{\lambda}}$. *If we choose the stepsize as $\eta = \frac{b^2\lambda}{6400L^2}$, and the deviation ξ_t satisfies $\forall 1 \le t \le T$: $\|\xi_t\|^2 \le \frac{2\lambda^2\eta^3\sqrt{\lambda\eta}}{15}(f(\widetilde{w}_{s-1}) - f(w^*))$, then $\mathbb{E}\left[f(\widetilde{w}_s) - f(w^*)\right] \le \frac{1}{2}\left[f(\widetilde{w}_{s-1}) - f(w^*)\right]$ when $T \ge \frac{90L}{\lambda b}$.*
- $20\sqrt{\frac{2L}{\lambda}} \le b \le n$. *If we choose stepsize as $\eta = \frac{1}{8L}$, and when the deviation ξ_t satisfies $\forall 1 \le t \le T$: $\|\xi_t\|^2 \le \frac{2\lambda^2\eta^3\sqrt{\lambda\eta}}{15}(f(\widetilde{w}_{s-1}) - f(w^*))$, then $\mathbb{E}\left[f(\widetilde{w}_s) - f(w^*)\right] \le \frac{1}{2}\left[f(\widetilde{w}_{s-1}) - f(w^*)\right]$ when $T \ge \sqrt{\frac{8L}{\lambda}}$.*

A direct consequence of Theorem 24 is the following iteration complexity of IMBA-SVRG.

Corollary 26 *For IMBA-SVRG algorithm, If we choose stepsize as $\eta = \min\left\{\frac{b^2\lambda}{6400L^2}, \frac{1}{8L}\right\}$, and when at every state s, the deviation ξ_t at every iteration satisfies $\forall 1 \le t \le T$: $\|\xi_t\|^2 \le \frac{2\lambda^2\eta^3\sqrt{\lambda\eta}}{15}(f(\widetilde{w}_{s-1}) - f(w^*))$, then $\mathcal{O}\left(\log\left(\frac{1}{\epsilon}\right)\right)$ full gradient evaluations and*

$$\mathcal{O}\left(\max\left\{\frac{L}{\lambda b}, \sqrt{\frac{L}{\lambda}}\right\} \cdot \log\left(\frac{1}{\epsilon}\right)\right)$$

inexact update steps of form (24) are sufficient to ensure $\mathbb{E}f(\widetilde{w}_s) - f(w^*) \le \epsilon$.

Our convergence proof of IMBA-SVRG relies on the machinery of stochastic estimation sequence (Lin et al., 2014), which originated from the framework of estimation sequence developed in (Nesterov, 2004). The details are given in Appendix B.

The main difference between our analysis and that of (Lin et al., 2014) is that we no longer require the inequalities in estimation sequences hold almost surely. Instead we only make sure they hold in expectation, and thus the quadratic lower bound used to construct the stochastic estimation sequence is also different. IMBA-SVRG can be viewed as an inexact extension of accelerated minibatch SVRG (Nitanda, 2014). Although (Nitanda, 2014) also considered stochastic estimation sequences, here we allow error in the stochastic gradient update, and thus need to construct a different estimation sequence which takes the inexactness into consideration. On the other hand, batch (accelerated) gradient methods with inexact first order oracle have been studied in (Schmidt et al., 2011; Villa et al., 2013; Devolder et al., 2014), and the analysis of IMBA-SVRG extends these results in the context of stochastic gradient methods with variance reduction.

Table 2: List of datasets used in the experiments.

Name	#Instances	#Features	Task
codrna	59,535	8	Classification
covtype	581,012	54	Classification
svmguide3	1,243	21	Classification
synthetic-c	10,000	1,000	Classification
cadata	20,460	8	Regression
spacega	3,107	6	Regression
synthetic-r	20,000	2,000	Regression
year	463,715	91	Regression

Algorithm 4 Practical implementation of MB-SVRP Algorithm used in Experiments.

Parameters: b.
Initialize $\eta = \frac{1}{L}$, $\nu = \frac{1-\sqrt{\lambda\eta}}{1+\sqrt{\lambda\eta}}$, $m = \lceil\frac{n}{b}\rceil$, $\bar{\lambda} = \frac{1}{\sqrt{b}}$, $\widetilde{w}_0 = 0$.
Sampling Sampling b items from $[n]$ to form a minibatch $\bar{B}$.
for $s = 1, 2, \ldots$ **do**
  Calculate $\widetilde{v} = \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widetilde{w}_{s-1})$.
  Initialize $y_0 = w_0 = \widetilde{w}_s$.
  for $t = 1, 2, \ldots, \lceil 2n/b \rceil$ **do**
    Sampling b items from $[n]$ to form a minibatch B_t.
    Initialize $w_t = y_{t-1}$.
    Calculate $\widetilde{u} = \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(y_{t-1}) + \eta\widetilde{v} - \frac{\eta}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w}_{s-1})$.
    for $k = 1, 2, \ldots, b$ **do**
      Sampling Sampling an index i_k from $\bar{B}$.
      Update (**MB-SVRP-I**): $w_t \leftarrow w_t - \eta\left(\nabla f_{i_k}(w_t) - \nabla f_{i_k}(y_{t-1}) + \bar{\lambda}(w_t - y_{t-1}) + \widetilde{u}\right)$.
      Update (**MB-SVRP-II**): $w_t \leftarrow w_t - \eta\left(\nabla^2 f_{i_k}(y_{t-1})(w_t - y_{t-1}) + \bar{\lambda}(w_t - y_{t-1}) + \widetilde{u}\right)$.
    end for
    Update: $y_t = w_t + \nu(w_t - w_{t-1})$.
  end for
  Update $\widetilde{w}_s = w_m$.
end for
Return $\widetilde{w}_s$

5. Experiments

In this section we compare the proposed MB-SVRP algorithm to several state-of-the-art methods for minimizing (1). The datasets are summarized in Table 2, most of which can be download from the LibSVM website[2]. We also considered a synthetic dataset for each task, where the data $\{x_i, b_i\}_{i=1}^n$ are i.i.d. drawn from the following model:

$$\text{Regression}: \quad b_i = \langle x_i, \bar{w}\rangle + a_i, \quad x_i \sim \mathcal{N}(0,\Sigma), \quad a_i \sim \mathcal{N}(0,1), \quad \forall i \in [n],$$

$$\text{Classification}: \quad P(b_i = \pm 1) = \frac{\exp(b_i\langle x_i, \bar{w}\rangle)}{1+\exp(b_i\langle x_i, \bar{w}\rangle)}, \quad x_i \sim \mathcal{N}(0,\Sigma), \quad \forall i \in [n],$$

where entries of $\bar{w}$ are drawn i.i.d. from $\mathcal{N}(0,1)$. To make the problem ill-conditioned with fast decaying spectrum, we set $\Sigma_{ij} = 2^{-|i-j|/500}, \forall i,j \in [n]$.

5.1 Empirical Results for Smooth Optimization

We consider ridge regression and logistic regression models for regression and classification problems, respectively. We normalize the dataset by $x_i \leftarrow x_i / \left(\max_i \|x_i\|^2\right)$ to ensure that the maximum norm of data points is 1. Note that the normalization will not change statistical property of the problem, the purpose here is to make the smoothness parameter fixed and invariant across different datasets, thus easier for us to set a suitable regularization parameter λ for the target condition number. We tried three settings of λ, as $1/n, 10^{-1}/n$ and $10^{-2}/n$ to represent different levels of regularization. It is expected that when $\lambda = 1/n$, the SVRG/SAGA/SDCA algorithms should converge very fast since the condition number is of the same order as sample size n, while for weak regularization case $\lambda = 10^{-2}/n$ these algorithms are expected to converge slowly.

We compare with SVRG (Johnson and Zhang, 2013), SDCA (Shalev-Shwartz and Zhang, 2013), and SAGA (Defazio et al., 2014), as they represent popular variance reduced optimization algorithms. We also compare with the minibatch version of SVRG and SAGA, in particular, we compare with a related minibatch accelerated SVRG (MB-SVRG) (Nitanda, 2014), which allows large minibatch size without slowing down convergence; while for SAGA, we apply the standard minibatch technique for SAGA and get the MB-SAGA algorithm. For quasi-Newton methods, we compare with L-BFGS with 10 as the limited memory size, and with line search of stepsize to satisfy the Wolfe condition (Wright and Nocedal, 1999). For the proposed MB-SVRP method, we implemented both Option I (MB-SVRP-I) and Option II (MB-SVRP-II), for ridge regression problem, these the two options are equivalent. At every iteration we simply run one pass of SVRG method initialized with y_{t-1}, to approximately solve (4). The implementation is summarized in Algorithm 4, where we fixed many parameters except the minibatch size b as the main tuning parameter. For SVRG and MB-SVRP method, we set $T = \lceil 2n/b \rceil$ as suggested in (Johnson and Zhang, 2013). The minibatch size of MB-SVRP are set to be $\max(\min((L/\lambda)^{1/3}, d), 40)$ inspired by theoretical analysis in Section 3, the minibatch size of MB-SVRG and MB-SAGA is set to be the same as MB-SVRP to demonstrate the direct comparison. In Section 5.3 we also performed a sensitivity study on the minibatch size b in MB-SVRP algorithm. Other parameters, such as stepsize in SVRG and SAGA, are tuned to give the fastest convergence.

We use primal objective suboptimality and the number of gradient evaluations as the evaluation metrics. Since gradient calculations account for the major computing part of

2. https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

first-order optimization methods, it is used as a good estimate (Johnson and Zhang, 2013; Shalev-Shwartz and Zhang, 2013; Defazio et al., 2014) of the computation cost, and it can be evaluated independent of specific implementation and computing hardware. Figure 2 and 3 showed the results for ℓ_2 regularized logistic regression and ridge regression, respectively, where we plot how the objective suboptimality $f(w_t) - f(w^*)$ decreases as the number of gradient evaluations divided by sample size (a.k.a. number of effective passes) increases. We have the following observations:

- When the regularization parameter λ (thus strong convexity) is large enough, all methods (except L-BFGS and MB-SAGA) converges very fast, typically using less than 50 passes over data to converges to numerical precision. The slow convergence of MB-SAGA is due to the usage of minibatch which slow down its computation efficiency.
- The convergence of L-BFGS is typically the slowest. When λ is large, the advantages of variance-reduced stochastic methods over L-BFGS are significant.
- When the regularization parameter is small, the proposed MB-SVRP method starts to show advantages. In particular, when $\lambda = 10^{-2}/n$, MB-SVRP is substantially faster than all other methods compared in our study. We also observed there is only negligible performance difference between MB-SVRP-I and MB-SVRP-II.

5.2 Empirical Results for Composite Optimization

We also consider empirical comparisons in the setting of non-smooth composition optimization problems. We adopted the same datasets used for the smooth optimization problems, but considered the elastic-net regularized logistic regression and linear regression models (Zou and Hastie, 2005). Where the objective is in the form of (12) but with a elastic-net regularization:

$$g(w) = \frac{\lambda}{2}\|w\|^2 + \mu\|w\|_1,$$

where μ is set to be $10^{-1}/n$. We follow the same settings as in the previous section, and compare with prox-SVRG (Xiao and Zhang, 2014), prox-SAGA (Defazio et al., 2014), prox MB-SVRG (Nitanda, 2014), and prox MB-SAGA, but didn't compare with L-BFGS because the method is generally only applicable for smooth problems. The results are shown in Figure 4 and 5, and we have similar observations as those of the smooth optimization case: all variance-reduced methods converges reasonably fast when λ is large, but when λ becomes small, the proposed prox MB-SVRP method converges significantly faster because it effectively leveraged the higher-order information from minibatches.

5.3 Sensitivity Study on the Minibatch Size in MB-SVRP

We also performed a sensitivity study on the minibatch size b, to see how b will affect the convergence behavior of MB-SVRP method. The experimental setting follows Section 5.1 and 5.2, except that we varying b to be $16, 32, 64, 128$ for MB-SVRP, and the regularization parameter λ is fixed to be $10^{-2}/n$. The results were shown in Figure 6, we observed that MB-SVRP method performed well on a wide range settings of minibatch sizes, and converges fast than SVRG in most cases.

6. Conclusion

In this paper, we proposed a novel minibatch stochastic optimization approach for regularized loss minimization in machine learning. This approach efficiently utilizes both variance-reduced first-order gradients and sub-sampled higher order information, and under suitable conditions, it can provably improve the iteration complexity over that of the previous state-of-the-art. Empirical experiments demonstrated that this approach performs well on a variety of smooth and composite optimization tasks in practice. The minibatch nature of the algorithm makes it useful for parallel and distributed computing environment, where additional speed up can be obtained by using many CPU cores for computation.

Acknowledgment

This work was performed when Jialei Wang was a research intern with Tong Zhang at Tencent AI Lab. The authors would like to thank Ji Liu and anonymous reviewers for their helpful feedbacks.

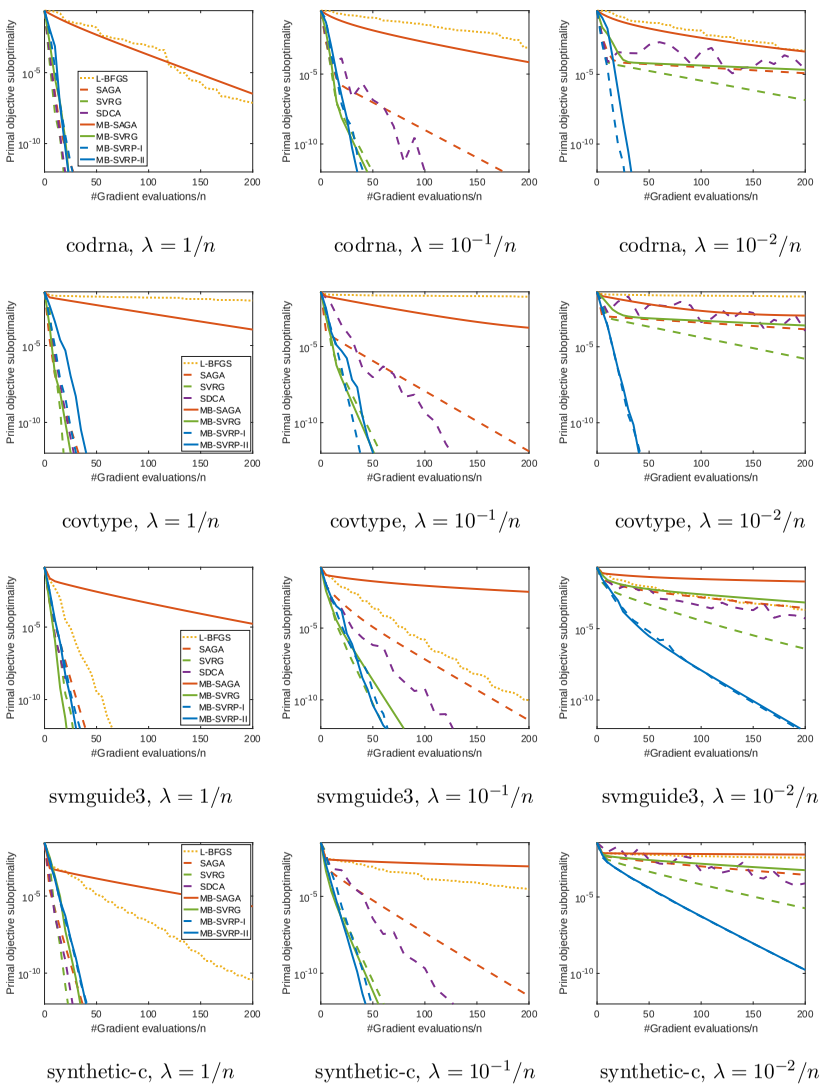
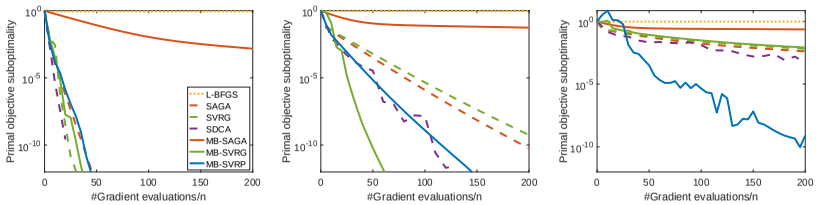
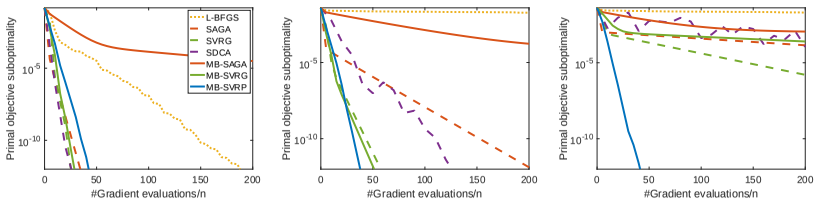


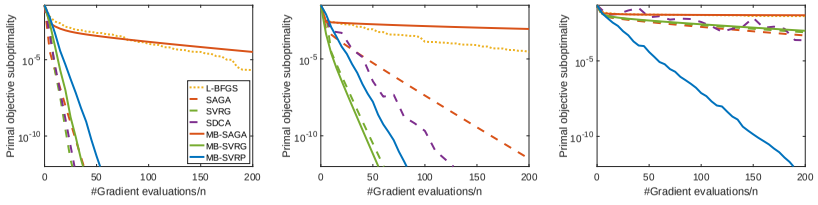
Figure 2: Comparison of various optimization algorithms for solving ℓ_2 regularized logistic regression problems.



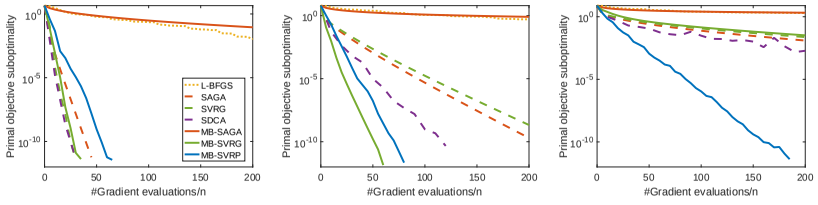
cadata, $\lambda = 1/n$ cadata, $\lambda = 10^{-1}/n$ cadata, $\lambda = 10^{-2}/n$



spacega, $\lambda = 1/n$ spacega, $\lambda = 10^{-1}/n$ spacega, $\lambda = 10^{-2}/n$

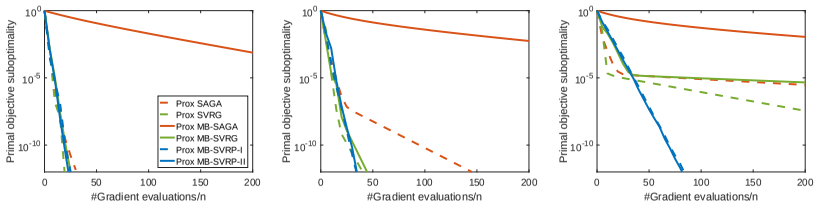


synthetic-r, $\lambda = 1/n$ synthetic-r, $\lambda = 10^{-1}/n$ synthetic-r, $\lambda = 10^{-2}/n$

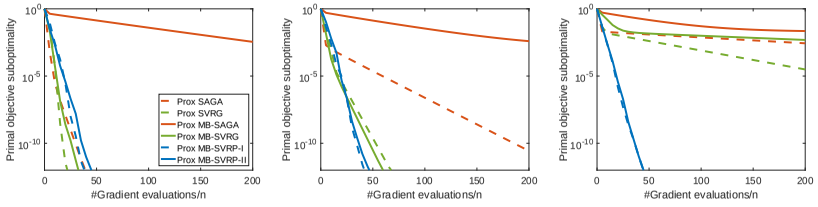


year, $\lambda = 1/n$ year, $\lambda = 10^{-1}/n$ year, $\lambda = 10^{-2}/n$

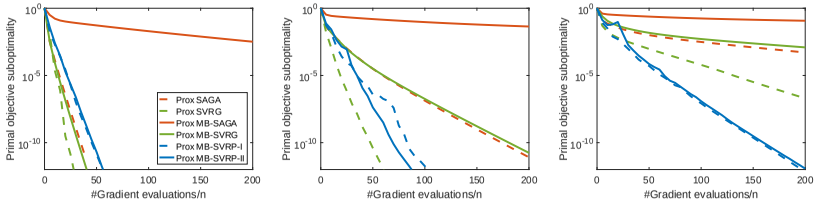
Figure 3: Comparison of various optimization algorithms for solving ridge regression problems.



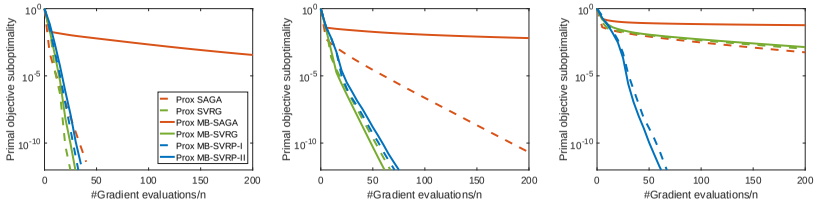
codrna, $\lambda = 1/n$ codrna, $\lambda = 10^{-1}/n$ codrna, $\lambda = 10^{-2}/n$



covtype, $\lambda = 1/n$ covtype, $\lambda = 10^{-1}/n$ covtype, $\lambda = 10^{-2}/n$

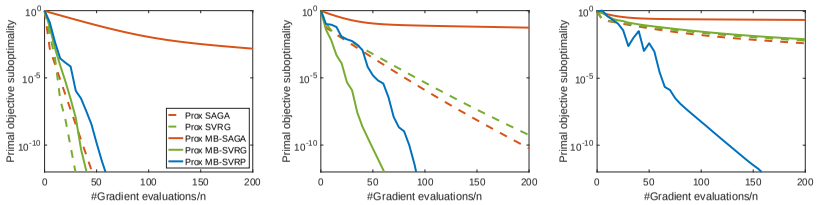


svmguide3, $\lambda = 1/n$ svmguide3, $\lambda = 10^{-1}/n$ svmguide3, $\lambda = 10^{-2}/n$

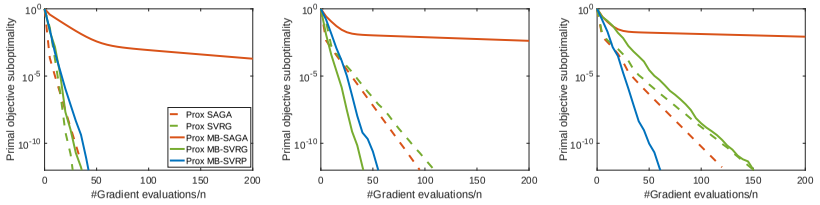


synthetic-c, $\lambda = 1/n$ synthetic-c, $\lambda = 10^{-1}/n$ synthetic-c, $\lambda = 10^{-2}/n$

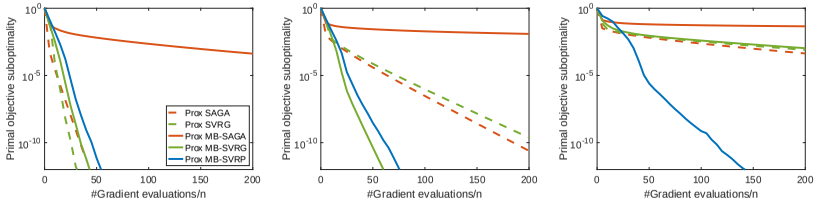
Figure 4: Comparison of various optimization algorithms for solving elastic-net regularized logistic regression problems.



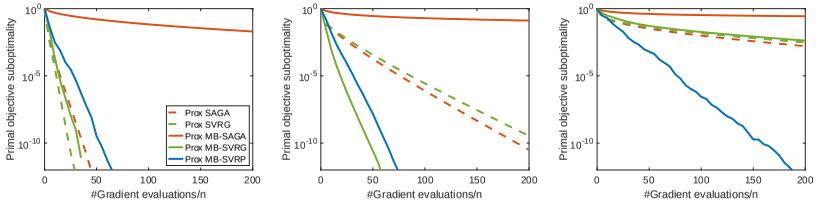
cadata, $\lambda = 1/n$ cadata, $\lambda = 10^{-1}/n$ cadata, $\lambda = 10^{-2}/n$



spacega, $\lambda = 1/n$ spacega, $\lambda = 10^{-1}/n$ spacega, $\lambda = 10^{-2}/n$

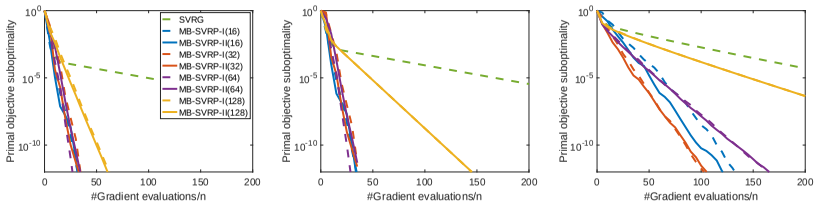


synthetic-r, $\lambda = 1/n$ synthetic-r, $\lambda = 10^{-1}/n$ synthetic-r, $\lambda = 10^{-2}/n$

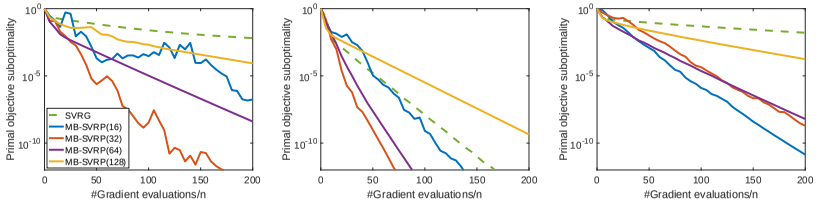


year, $\lambda = 1/n$ year, $\lambda = 10^{-1}/n$ year, $\lambda = 10^{-2}/n$

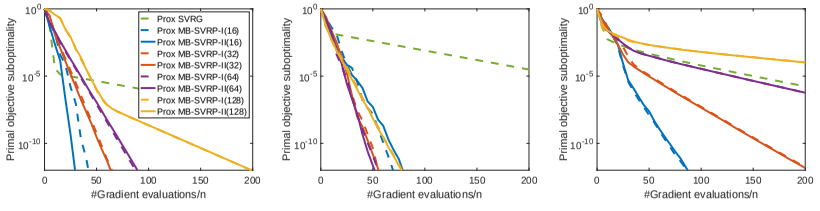
Figure 5: Comparison of various optimization algorithms for solving elastic-net regularized regression problems.



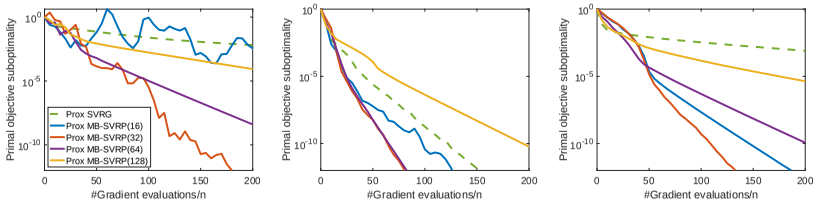
codrna, ℓ_2 + logistic covtype, ℓ_2 + logistic synthetic-c, ℓ_2 + logistic



cadata, ℓ_2 + squared spacega, ℓ_2 + squared synthetic-r, ℓ_2 + squared



codrna, $\ell_1 + \ell_2$ + logistic covtype, $\ell_1 + \ell_2$ + logistic synthetic-c, $\ell_1 + \ell_2$ + logistic



cadata, $\ell_1 + \ell_2$ + squared spacega, $\ell_1 + \ell_2$ + squared synthetic-r, $\ell_1 + \ell_2$ + squared

Figure 6: Convergence of MB-SVRP algorithms with various minibatch size b (in parentheses), compared with SVRG.

Appendix

The appendix contains proofs of some theorems and lemmas stated in the main paper.

Appendix A. Proofs for Section 3

A.1 Proof of Proposition 3

Proof Let $w^* = \arg\min_w f(w)$, and $z^* = \arg\min_z f(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z)$, it is clear that $w^* = \bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z^*$, which is coincident with the exchange of variable rule $z = \bar{H}_{\widetilde{\lambda}}^{1/2}(w^*)w$. Moreover, since

$$\eta\left(\frac{1}{b}\sum_{i\in B_t}\nabla_z f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z_{t-1}) + \frac{1}{n}\sum_{i=1}^n \nabla_z f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\widetilde{z}) - \frac{1}{b}\sum_{i\in B_t}\nabla_z f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\widetilde{z})\right)$$
$$\overset{①}{=} \eta H_{\widetilde{\lambda}}^{-1/2}(w^*)\left(\frac{1}{b}\sum_{i\in B_t}\nabla f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)z_{t-1}) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\widetilde{z}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\widetilde{z})\right)$$
$$\overset{②}{=} \eta H_{\widetilde{\lambda}}^{-1/2}(w^*)\left(\frac{1}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widetilde{w}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w})\right),$$

where at step ① we have used the gradient chain rule, at step ② we used the exchange of variables $z = \bar{H}_{\widetilde{\lambda}}^{1/2}(w^*)w$. Substitute to (15), and multiplying both sides of above equation with $\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)$, we get

$$w_t \leftarrow w_{t-1} - \eta\bar{H}_{\widetilde{\lambda}}(w_{t-1})^{-1}\left(\frac{1}{b}\sum_{i\in B_t}\nabla f_i(w_{t-1}) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widetilde{w}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w})\right),$$

which recovers the exact formulation of (14). ■

A.2 Proof of Lemma 4

Proof We first perform the following decomposition:

$$\|z_t - \bar{z}_t\| \le \|z_t - z_t^*\| + \|z_t^* - \bar{z}_t\|, \tag{26}$$

then we can bound $\|z_t - z_t^*\|$ by

$$\begin{aligned}\|z_t - z_t^*\| =& \left\|\bar{H}_{\widetilde{\lambda}}^{1/2}(w^*)(w_t - w_t^*)\right\| \le \left\|\bar{H}_{\widetilde{\lambda}}(w^*)\right\|^{1/2}\|w_t - w_t^*\| \le \sqrt{L+\widetilde{\lambda}}\,\|w_t - w_t^*\| \\ \le& \sqrt{\frac{2(\widetilde{f}_t(w_t) - \widetilde{f}_t(w_t^*))(L+\widetilde{\lambda})}{\lambda+\widetilde{\lambda}}},\end{aligned} \tag{27}$$

where in the last inequality we use the $\lambda+\widetilde{\lambda}$-strong convexity of $\widetilde{f}_t(w)$. For the term $\|z_t^* - \bar{z}_t\|$, we can bounded it using Hessian Lipschitz property,

$$\begin{aligned}\|z_t^* - \bar{z}_t\| =& \left\|\eta\bar{H}_{\widetilde{\lambda}}^{1/2}(w^*)\left(\bar{H}_{\widetilde{\lambda}}(w^*)^{-1} - \bar{H}_{\widetilde{\lambda}}(w_{t-1})^{-1}\right)\vartheta_t\right\| \\ \le& \eta\left\|\left(\bar{H}_{\widetilde{\lambda}}(w^*)^{-1/2}\bar{H}_{\widetilde{\lambda}}(w_{t-1})^{-1}\right)\left(\bar{H}_{\widetilde{\lambda}}(w^*) - \bar{H}_{\widetilde{\lambda}}(w_{t-1})\right)\right\|\|\vartheta_t\| \\ \le& \frac{\eta}{\widetilde{\lambda}^{3/2}}M\|w_{t-1} - w^*\|\|\vartheta_t\|.\end{aligned} \tag{28}$$

Combining (26), (27) and (28) finishes the proof. ■

A.3 Proof of Lemma 7

Proof We first decompose the terms $\frac{\lambda_j}{\lambda_j+\widetilde{\lambda}}$ into two parts, the first one contains indexes smaller or equal than k and the second part contains indexes larger than k, then we upper bound these two parts get obtain the stated results.

$$\begin{aligned}d_{\widetilde{\lambda}} =& \sum_j \frac{\lambda_j}{\lambda_j+\widetilde{\lambda}} = \min_k\left\{\sum_{j\le k}\frac{\lambda_j}{\lambda_j+\widetilde{\lambda}} + \sum_{j>k}\frac{\lambda_j}{\lambda_j+\widetilde{\lambda}}\right\} \\ \le& \min_k\left\{\sum_{j\le k}\frac{\lambda_j+\widetilde{\lambda}}{\lambda_j+\widetilde{\lambda}} + \sum_{j>k}\frac{\lambda_j}{\widetilde{\lambda}}\right\} = \min_k\left\{k + \frac{\sum_{j>k}\lambda_j}{\widetilde{\lambda}}\right\}\end{aligned}$$

■

A.4 Proof of Lemma 9

Proof For $j = 1, ..., b$, define random matrix

$$v_j = \frac{1}{b}\left(H_{\widetilde{\lambda}}^{-1/2}(w^*)\ell''(w^{*\top}x_k; b_k)x_k x_k^\top H_{\widetilde{\lambda}}^{-1/2}(w^*) - \frac{1}{n}\sum_{i=1}^n H_{\widetilde{\lambda}}^{-1/2}(w^*)\ell''(w^{*\top}x_i; b_i)x_i x_i^\top H_{\widetilde{\lambda}}^{-1/2}(w^*)\right),$$

with probability $1/n, \forall k \in [n]$. It is easy to check that $\mathbb{E}[v_j] = 0$, and

$$\begin{aligned}\|v_j\| \le& \frac{1}{b}\left(\left\|H_{\widetilde{\lambda}}^{-1/2}(w^*)\ell''(w^{*\top}x_k; b_k)x_k x_k^\top H_{\widetilde{\lambda}}^{-1/2}(w^*)\right\|\right) \\ &+ \frac{1}{b}\left(\left\|\frac{1}{n}\sum_{i=1}^n H_{\widetilde{\lambda}}^{-1/2}(w^*)\ell''(w^{*\top}x_i; b_i)x_i x_i^\top H_{\widetilde{\lambda}}^{-1/2}(w^*)\right\|\right) \\ \le& \frac{1}{b}\left(2\max_{i\in[n]}\ell''(w^{*\top}x_i; b_i)x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i\right) \le \frac{2\rho_{\widetilde{\lambda}}^2}{b}\left(\frac{1}{n}\sum_{i=1}^n \ell''(w^{*\top}x_i; b_i)x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i\right) \\ =& \frac{2\rho_{\widetilde{\lambda}}^2}{b}\operatorname{tr}\left(H_{\widetilde{\lambda}}^{-1}(w^*)\left(\frac{1}{n}\sum_{i=1}^n \ell''(w^{*\top}x_i; b_i)x_i x_i^\top\right)\right) = \frac{2\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b}.\end{aligned}$$

for the expected second order moment, denote $\widetilde{x}_k = (\ell''(w^{*\top}x_k;b_k))^{1/2}H_{\widetilde{\lambda}}^{-1/2}(w^*)x_k$ we have

$$\begin{aligned}
\left\|\mathbb{E}\left[v_j^2\right]\right\| &= \left\|\frac{1}{nb^2}\sum_{k=1}^n\left(\widetilde{x}_k\widetilde{x}_k^\top - \frac{1}{n}\sum_{i=1}^n\widetilde{x}_i\widetilde{x}_i^\top\right)^2\right\| \\
&= \left\|\frac{1}{nb^2}\sum_{k=1}^n\left(\left(\widetilde{x}_k\widetilde{x}_k^\top\right)^2 - 2\left(\widetilde{x}_k\widetilde{x}_k^\top\right)\left(\frac{1}{n}\sum_{i=1}^n\widetilde{x}_i\widetilde{x}_i^\top\right) + \left(\frac{1}{n}\sum_{i=1}^n\widetilde{x}_i\widetilde{x}_i^\top\right)^2\right)\right\| \\
&= \left\|\frac{1}{nb^2}\sum_{k=1}^n\left(\widetilde{x}_k\widetilde{x}_k^\top\right)^2 - \frac{2}{b^2}\left(\frac{1}{n}\sum_{i=1}^n\widetilde{x}_i\widetilde{x}_i^\top\right)^2 + \frac{1}{b^2}\left(\frac{1}{n}\sum_{i=1}^n\widetilde{x}_i\widetilde{x}_i^\top\right)^2\right\| \\
&= \left\|\frac{1}{nb^2}\sum_{k=1}^n\left(\widetilde{x}_k\widetilde{x}_k^\top\right)^2 - \frac{1}{b^2}\left(\frac{1}{n}\sum_{i=1}^n\widetilde{x}_i\widetilde{x}_i^\top\right)^2\right\| \\
&\le \left\|\frac{1}{nb^2}\sum_{k=1}^n\left(\widetilde{x}_k\widetilde{x}_k^\top\right)^2\right\| \le \left\|\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b^2}\left(\frac{1}{n}\sum_{k=1}^n\widetilde{x}_k\widetilde{x}_k^\top\right)\right\| \\
&= \frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b^2}\left\|H_{\widetilde{\lambda}}^{-1/2}(w^*)\left(\frac{1}{n}\sum_{i=1}^n\ell''(w^{*\top}x_i;b_i)x_ix_i^\top\right)H_{\widetilde{\lambda}}^{-1/2}(w^*)\right\| \\
&\le \frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b^2}.
\end{aligned}$$

Thus

$$\left\|\sum_{j=1}^b\mathbb{E}\left[v_j^2\right]\right\| \le \frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b},$$

using Lemma 35, we know

$$\begin{aligned}
&P\left(\left\|H_{\widetilde{\lambda}}^{-1/2}(w^*)\left(\frac{1}{b}\sum_{i\in\bar{B}}\ell''(w^{*\top}x_i;b_i)x_ix_i^\top - \frac{1}{n}\sum_{i=1}^n\ell''(w^{*\top}x_i;b_i)x_ix_i^\top\right)H_{\widetilde{\lambda}}^{-1/2}(w^*)\right\|_2 \ge t\right) \\
&= P\left(\left\|\sum_{j=1}^b v_j\right\| \ge t\right) \le d\exp\left(\frac{-t^2/2}{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}/b + 2\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} t/(3b)}\right),
\end{aligned}$$

setting $t = 2\log\left(\frac{d}{\delta}\right)\cdot\sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b}}$ we conclude the proof. ■

A.5 Proof of Lemma 10

Proof When $\widetilde{\lambda} \ge \lambda$, we know

$$\begin{aligned}
&\lambda_{\min}(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)H_\lambda(w)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)) \\
&\ge \lambda_{\min}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\left(\frac{\lambda}{\widetilde{\lambda}}\left(\frac{1}{n}\sum_{i=1}^n\ell''(w^\top x_i;b_i)x_ix_i^\top\right) + \lambda I\right)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\right) \\
&= \frac{\lambda}{\widetilde{\lambda}}\lambda_{\min}(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)H_{\widetilde{\lambda}}(w)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)).
\end{aligned}$$

Further more we can lower bound $\lambda_{\min}(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)H_{\widetilde{\lambda}}(w)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*))$ by

$$\begin{aligned}
&\lambda_{\min}(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)H_{\widetilde{\lambda}}(w)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)) \\
&= \lambda_{\min}(\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)H_{\widetilde{\lambda}}(w)) \\
&= \frac{1}{\lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w)\bar{H}_{\widetilde{\lambda}}(w^*))} \\
&\ge \frac{1}{\lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w)H_{\widetilde{\lambda}}(w)) + \lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w)(H_{\widetilde{\lambda}}(w) - \bar{H}_{\widetilde{\lambda}}(w^*)))} \\
&\ge \frac{1}{1 + \lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w)(H_{\widetilde{\lambda}}(w^*) - \bar{H}_{\widetilde{\lambda}}(w^*))) + \lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w)(H_{\widetilde{\lambda}}(w) - H_{\widetilde{\lambda}}(w^*)))},
\end{aligned} \tag{29}$$

For the term $\lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w)(H_{\widetilde{\lambda}}(w^*) - \bar{H}_{\widetilde{\lambda}}(w^*)))$, we have the following decomposition:

$$\begin{aligned}
\lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w)(H_{\widetilde{\lambda}}(w^*) - \bar{H}_{\widetilde{\lambda}}(w^*))) \le &\lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w^*)(H_{\widetilde{\lambda}}(w^*) - \bar{H}_{\widetilde{\lambda}}(w^*))) \\
&+ \lambda_{\max}((H_{\widetilde{\lambda}}^{-1}(w^*) - H_{\widetilde{\lambda}}^{-1}(w))(H_{\widetilde{\lambda}}(w^*) - \bar{H}_{\widetilde{\lambda}}(w^*)))
\end{aligned} \tag{30}$$

since with probability at least $1-\delta$, we have

$$\lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w^*)(H_{\widetilde{\lambda}}(w^*) - \bar{H}_{\widetilde{\lambda}}(w^*))) \le 2\log\left(\frac{d}{\delta}\right)\sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b}}, \tag{31}$$

and

$$\begin{aligned}
&\lambda_{\max}((H_{\widetilde{\lambda}}^{-1}(w^*) - H_{\widetilde{\lambda}}^{-1}(w))(H_{\widetilde{\lambda}}(w^*) - \bar{H}_{\widetilde{\lambda}}(w^*))) \\
&= \lambda_{\max}((H_{\widetilde{\lambda}}(w^*) - H_{\widetilde{\lambda}}(w))H_{\widetilde{\lambda}}^{-1}(w)H_{\widetilde{\lambda}}^{-1}(w^*)(H_{\widetilde{\lambda}}(w^*) - \bar{H}_{\widetilde{\lambda}}(w^*))) \\
&\le \lambda_{\max}(H_{\widetilde{\lambda}}(w^*) - H_{\widetilde{\lambda}}(w))2\log\left(\frac{d}{\delta}\right)\sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b}}\cdot\frac{1}{\widetilde{\lambda}} \\
&\le \frac{M\left\|w-w^*\right\|}{\widetilde{\lambda}}\cdot 2\log\left(\frac{d}{\delta}\right)\sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b}},
\end{aligned} \tag{32}$$

using the results of Lemma (9), and also Lipschitz Hessian property (Condition 1). Moreover,

$$\lambda_{\max}(H_{\widetilde{\lambda}}^{-1}(w)(H_{\widetilde{\lambda}}(w)-H_{\widetilde{\lambda}}(w^*))) \leq \frac{\lambda_{\max}(H_{\widetilde{\lambda}}(w)-H_{\widetilde{\lambda}}(w^*))}{\widetilde{\lambda}} \leq \frac{M\|w-w^*\|}{\widetilde{\lambda}}. \tag{33}$$

Combining (29), (31), (32), (33) we have with probability at least $1-\delta$

$$\min_{\|w-w^*\|\leq R}\lambda_{\min}(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)H_\lambda(w)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)) \geq \frac{\lambda}{\widetilde{\lambda}}\cdot\frac{1}{1+2\log\left(\frac{d}{\delta}\right)\sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b}}+2\log\left(\frac{d}{\delta}\right)\sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b}}\cdot\frac{MR}{\widetilde{\lambda}}+\frac{MR}{\widetilde{\lambda}}},$$

which finishes the proof. ■

A.6 Proof of Lemma 11

Proof First we perform the following decomposition:

$$\begin{aligned}\max_{i\in[n]}\max_{\|w-w^*\|\leq R}\ell''(w^\top x_i;b_i)x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i \leq& \max_{i\in[n]}\max_{\|w-w^*\|\leq R}|\ell''(w^\top x_i;b_i)-\ell''(w^{*\top}x_i;b_i)|x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i\\ &+\max_{i\in[n]}\ell''(w^{*\top}x_i;b_i)x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i,\end{aligned}$$

where

$$\begin{aligned}\max_{i\in[n]}\max_{\|w-w^*\|\leq R}|\ell''(w^\top x_i;b_i)-\ell''(w^{*\top}x_i;b_i)|x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i \leq& \max_{i\in[n]}\bar{M}Rx_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i\\ \leq& \frac{\bar{M}RD}{\widetilde{\lambda}}=\frac{MR}{\widetilde{\lambda}}.\end{aligned}$$

Thus we have

$$\max_{i\in[n]}\max_{\|w-w^*\|\leq R}\ell''(w^\top x_i;b_i)x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i \leq \max_{i\in[n]}\ell''(w^{*\top}x_i;b_i)x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i+\frac{MR}{\widetilde{\lambda}}.$$

Then we relate $x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i$ to $x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i$, for this purpose, we first upper bound $\lambda_{\max}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)H_{\widetilde{\lambda}}(w^*)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\right)$. By Lemma 9, when $b\geq 16\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\log^2(d/\delta)$ we have

$$\left\|H_{\widetilde{\lambda}}^{-1}(w^*)(H_{\widetilde{\lambda}}(w^*)-\bar{H}_{\widetilde{\lambda}}(w^*))\right\|_2 \leq 2\log\left(\frac{d}{\delta}\right)\sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{b}} \leq \frac{1}{2},$$

thus

$$\begin{aligned}\lambda_{\max}\left(\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)H_{\widetilde{\lambda}}(w^*)\bar{H}_{\widetilde{\lambda}}^{-1/2}(w^*)\right)=&\frac{1}{\lambda_{\min}\left(H_{\widetilde{\lambda}}^{-1}(w^*)\bar{H}_{\widetilde{\lambda}}(w^*)\right)}\\ \leq&\frac{1}{1-\left\|H_{\widetilde{\lambda}}^{-1}(w^*)(H_{\widetilde{\lambda}}(w^*)-\bar{H}_{\widetilde{\lambda}}(w^*))\right\|_2}\\ \leq&\frac{1}{1-1/2}\leq 2.\end{aligned}$$

Then we bound $\max_{i\in[n]}\ell''(w^{*\top}x_i;b_i)x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i$ through $\max_{i\in[n]}\ell''(w^{*\top}x_i;b_i)x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i$, because

$$\begin{aligned}x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i=&x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i+x_i^\top\left(\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)-H_{\widetilde{\lambda}}^{-1}(w^*)\right)x_i\\ =&x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i+x_i^\top H_{\widetilde{\lambda}}^{-1/2}(w^*)\left(H_{\widetilde{\lambda}}^{1/2}(w^*)\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)H_{\widetilde{\lambda}}^{1/2}(w^*)-I\right)H_{\widetilde{\lambda}}^{-1/2}(w^*)x_i\\ \leq&x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i+\lambda_{\max}\left(H_{\widetilde{\lambda}}^{1/2}(w^*)\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)H_{\widetilde{\lambda}}^{1/2}(w^*)-I\right)x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i\\ \overset{①}{\leq}&2x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i,\end{aligned}$$

where in step ① we used the fact that

$$\begin{aligned}&\lambda_{\max}\left(H_{\widetilde{\lambda}}^{1/2}(w^*)\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)H_{\widetilde{\lambda}}^{1/2}(w^*)-I\right)\\ &\leq\max\left\{|\lambda_{\max}(H_{\widetilde{\lambda}}^{1/2}(w^*)\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)H_{\widetilde{\lambda}}^{1/2}(w^*))-1|,|\lambda_{\min}(H_{\widetilde{\lambda}}^{1/2}(w^*)\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)H_{\widetilde{\lambda}}^{1/2}(w^*))-1|\right\}\\ &=\max\left\{|\lambda_{\max}(\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)H_{\widetilde{\lambda}}(w^*))-1|,|\lambda_{\min}(\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)H_{\widetilde{\lambda}}(w^*))-1|\right\}\leq 1.\end{aligned}$$

Based on the definition of effective dimension (Definition 6) and condition of bounded statistical leverage (Assumption 8), we can bound the smoothness as

$$\begin{aligned}\max_{i\in[n]}\max_{\|w-w^*\|\leq R}\ell''(w^\top x_i;b_i)x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i \leq&2\max_{i\in[n]}\ell''(w^{*\top}x_i;b_i)x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i+\frac{MR}{\widetilde{\lambda}}\\ \leq&2\rho_{\widetilde{\lambda}}^2\left(\frac{1}{n}\sum_{i=1}^n\ell''(w^{*\top}x_i;b_i)x_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)x_i\right)+\frac{MR}{\widetilde{\lambda}}\\ \overset{①}{=}&2\rho_{\widetilde{\lambda}}^2\left(\frac{1}{n}\sum_{i=1}^n\mathrm{tr}(\ell''(w^{*\top}x_i;b_i)x_ix_i^\top H_{\widetilde{\lambda}}^{-1}(w^*))\right)+\frac{MR}{\widetilde{\lambda}},\end{aligned}$$

where in step ① we have used the fact that $\mathrm{tr}(ABC)=\mathrm{tr}(CAB)$ for any A,B,C. Furthermore we have

$$\begin{aligned}2\rho_{\widetilde{\lambda}}^2\left(\frac{1}{n}\sum_{i=1}^n\mathrm{tr}(\ell''(w^{*\top}x_i;b_i)x_ix_i^\top H_{\widetilde{\lambda}}^{-1}(w^*))\right)=&2\rho_{\widetilde{\lambda}}^2\left(\mathrm{tr}\left(\frac{1}{n}\sum_i\ell''(w^{*\top}x_i;b_i)x_ix_i^\top H_{\widetilde{\lambda}}^{-1}(w^*)\right)\right)\\ =&2\rho_{\widetilde{\lambda}}^2\left(\mathrm{tr}\left(\left(\frac{1}{n}\sum_i\ell''(w^{*\top}x_i;b_i)x_ix_i^\top\right)H_{\widetilde{\lambda}}^{-1}(w^*)\right)\right)\\ =&2\rho_{\widetilde{\lambda}}^2\sum_{j=1}^d\frac{\lambda_j}{\lambda_j+\widetilde{\lambda}}=2\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}.\end{aligned}$$

Combining above analysis, we get

$$\max_{i\in[n]}\max_{\|w-w^*\|\leq R}\ell''(w^\top x_i;b_i)x_i^\top\bar{H}_{\widetilde{\lambda}}^{-1}(w^*)x_i \leq 2\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}+\frac{MR}{\widetilde{\lambda}},$$

which concludes the proof. ■

A.7 Proof of Theorem 13

Proof Since we choose $\widetilde{\lambda}$ at the scale of $\widetilde{\lambda} = \max\left\{\lambda, \frac{L}{b}\right\}$, then we can apply Lemma 10 and Lemma 11 to verify that the new condition number after "preconditioning" will be

$$\max\left\{4\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}, \frac{4L\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}{\lambda b}\right\},$$

applying Corollary 26 we know as long as the inexactness condition (25) is satisfied, we requires $\mathcal{O}\left(\log\left(\frac{1}{\epsilon}\right)\right)$ full gradient evaluations and

$$\max\left\{C_{\rm sb}, C_{\rm lb}\right\} \cdot \frac{10}{9}\log(36)\log\left(\frac{1}{\epsilon}\right) \tag{34}$$

total calls of approximate minimization of (8) to ensure $\mathbb{E}f(\widetilde{w}_s) - f(w^*) \leq \epsilon$, where the factors $C_{\rm sb}, C_{\rm lb}$ are

$$\begin{aligned} C_{\rm sb} =& \mathcal{O}\left(\max\left\{\frac{240}{b}, \frac{240L}{\lambda b^2}\right\} \cdot \rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}\right), \\ C_{\rm lb} =& \mathcal{O}\left(\max\left\{2\sqrt{6}, 2\sqrt{6}\sqrt{\frac{L}{\lambda b}}\right\} \cdot \sqrt{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}}\right), \end{aligned}$$

which represent the cases of small and large minibatch sizes, respectively. Since $\widetilde{\lambda} \geq \frac{L}{b}$, then the condition number of (8) is never larger than $\frac{L}{\widetilde{\lambda}} \leq L \cdot \frac{b}{L} = b$, we know from Lemma 36, when applying SVRG to solve (8), to reach some point of which objective the suboptimality of (8) satisfies

$$\widetilde{f}_t(w_t) - \min_w \widetilde{f}_t(w) \leq \frac{1}{10^5} \cdot \left(\frac{\lambda}{L}\right)^7 \epsilon, \tag{35}$$

the following number of gradient calls sufficient:

$$C \cdot b \cdot \log\left(\frac{L}{\lambda}\right) \cdot \log\left(\frac{1}{\epsilon}\right), \tag{36}$$

for some universal constant C. On the other hand, for every subproblem (8), if (35) is satisfied, applying Lemma 4 and Lemma 33 we know the gradient error in IMBA-SVRG (Algorithm 3)in the preconditioned space can be upper bounded by

$$\|\xi_t\|^2 \leq \left(\frac{\lambda}{L}\right)^6 \cdot \frac{2\epsilon}{10^5} + \frac{\eta^2 M^2 R^2 L}{\widetilde{\lambda}^4}(f(\widetilde{w}_{s-1}) - f(w^*)),$$

thus condition (25) in Theorem 24 is satisfied. Combining (34) and (36) we know the total number of gradient calls:

$$n \cdot \log\left(\frac{1}{\epsilon}\right) + \frac{10C}{9}\log(36) \cdot \max\left\{C_{\rm sb}, C_{\rm lb}\right\} \cdot b \cdot \log\left(\frac{L}{\lambda}\right) \cdot \log^2\left(\frac{1}{\epsilon}\right) \tag{37}$$

is sufficient to obtain a solution such that $\mathbb{E}f(\widetilde{w}_s) - f(w^*) \leq \epsilon$ is satisfied. Next we choose b such that the total iteration complexity of above expression is minimized. Start from here we will ignore the constants before these factors for simplicity, we know the term $\max\left\{C_{\rm sb}, C_{\rm lb}\right\} \cdot b$ is of order

$$\mathcal{O}\left(\max\left\{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}, \frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L}{\lambda b}, \sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L b}{\lambda}}\right\}\right),$$

when $n \gtrsim \rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \left(\frac{L}{\lambda}\right)^{1/3}$, we can choose $b \asymp \rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \left(\frac{L}{\lambda}\right)^{1/3}$, then $\max\left\{C_{\rm sb}, C_{\rm lb}\right\} \cdot b$ is of order

$$\mathcal{O}\left(\max\left\{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}}, \left(\frac{L}{\lambda}\right)^{2/3}, \rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \left(\frac{L}{\lambda}\right)^{2/3}\right\}\right) = \mathcal{O}\left(\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \left(\frac{L}{\lambda}\right)^{2/3}\right), \tag{38}$$

when $n \lesssim \rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \left(\frac{L}{\lambda}\right)^{1/3}$, we can choose $b \asymp n$, in which case $\max\left\{C_{\rm sb}, C_{\rm lb}\right\} \cdot b$ can be upper bounded by

$$\mathcal{O}\left(\max\left\{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L}{\lambda n}, \sqrt{\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L n}{\lambda}}\right\}\right) = \mathcal{O}\left(\frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L}{\lambda n}\right), \tag{39}$$

Combining (37), (38) and (39), we know the total number of individual function gradient calls to reach ϵ-suboptimality is

$$\mathcal{O}\left(\widetilde{\kappa} \cdot \log\left(\frac{L}{\lambda}\right) \cdot \log^2\left(\frac{1}{\epsilon}\right) + n \cdot \log\left(\frac{1}{\epsilon}\right)\right),$$

where

$$\widetilde{\kappa} = \max\left\{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \left(\frac{L}{\lambda}\right)^{2/3}, \frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L}{\lambda n}\right\},$$

which finishes the proof. ■

A.8 Proof of Theorem 22

Proof Applying the theory of catalyst acceleration (Lemma 37) we know only $\mathcal{O}\left(\left(\sqrt{\frac{\lambda+\gamma}{\lambda}}\right)\log\left(\frac{1}{\epsilon}\right)\right)$ calls of MB-SVRP is sufficient to reach ϵ-objective suboptimality, as long as each iterate w_r satisfies

$$f(w_r) + \frac{\gamma}{2}\|w_r - z_{r-1}\|^2 \leq \min_w f(w) + \frac{\gamma}{2}\|w - z_{r-1}\|^2 + \frac{\lambda\epsilon}{3600(\lambda+\gamma)}\left(1 - \frac{9}{10}\sqrt{\frac{\lambda}{\lambda+\gamma}}\right).$$

Moreover, according to Theorem 13, the iteration complexity of solving a $\lambda+\gamma$ strongly convex problem (23) is

$$\widetilde{\mathcal{O}}\left(n + \max\left\{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} \left(\frac{L}{(\lambda+\gamma)}\right)^{2/3}, \frac{\rho_{\widetilde{\lambda}}^2 d_{\widetilde{\lambda}} L}{(\lambda+\gamma)n}\right\}\right),$$

combining these two results we know the total iteration complexity of Algorithm 2 can be upper bounded by

$$\widetilde{\mathcal{O}}\left(\left(\sqrt{\frac{\lambda+\gamma}{\lambda}}\right)\left(n+\max\left\{\left(\frac{\rho_\lambda^2 d_{\widetilde{\lambda}} L}{(\lambda+\gamma)}\right)^{2/3}, \frac{\rho_\lambda^2 d_{\widetilde{\lambda}} L}{(\lambda+\gamma)n}\right\}\right)\right),$$

When $\frac{\rho_\lambda^2 d_{\widetilde{\lambda}} L}{\lambda} \geq n^{3/2}$, if we choose

$$\gamma = \frac{\rho_\lambda^2 d_{\widetilde{\lambda}} L}{n^{3/2}} - \lambda,$$

we obtain the iteration complexity can be upper bounded by

$$\widetilde{\mathcal{O}}\left(\left(\sqrt{\frac{\rho_\lambda^2 d_{\widetilde{\lambda}} L}{n^{3/2}\lambda}}\right)\cdot n\right) = \widetilde{\mathcal{O}}\left(\sqrt{\rho_\lambda^2 d_{\widetilde{\lambda}}}\cdot n^{1/4}\left(\frac{L}{\lambda}\right)^{1/2}\right),$$

which concludes the proof. ■

Appendix B. Proofs for Section 4

In this appendix, we describe the stochastic estimation sequence approach for inexact accelerated minibatch SVRG in Section 4, and provide the detailed analysis.

B.1 Stochastic Estimation Sequences

We introduce the following definition.

Definition 27 *(**Stochastic estimation sequence**) A sequence of pairs $\{V_t(w), \theta_t\}_{t\geq 0}$ is called an estimation sequence of the function $f(w)$ if $\theta_t > 0$ and for any $w \in \mathbb{R}^d, t \geq 0$ we have*

$$V_t(w) \leq (1-\theta_t)f(w) + \theta_t V_0(w), \tag{40}$$

if $\{V_t(w)\}_{t\geq 0}$ is a sequence of random functions, we call the sequence of pairs $\{V_t(w), \theta_t\}_{t\geq 0}$ a stochastic estimation sequence if (40) holds in expectation, i.e.

$$\mathbb{E}\left[V_t(w)\right] \leq (1-\theta_t)f(w) + \theta_t V_0(w).$$

The following lemma is a generalized version of Lemma 1 in (Lin et al., 2014), which shows if we can construct upper bound of $\mathbb{E}\left[f(w_t)\right]$ using $\mathbb{E}\left[V_t(w)\right]$, then we get convergence of $\mathbb{E}\left[f(w_t)\right]$.

Lemma 28 *Suppose $\{V_t(w), \theta_t\}_{t\geq 0}$ is a stochastic estimation sequence of the function $f(w)$. Let w^* be the minimizer of $f(w)$. If there are sequences of random variables $\{w_t\}_{t\geq 0}$ and $\{\epsilon_t\}_{t\geq 0}$ in $\mathbb{R}^d$, $\{\delta_t\}_{t\geq 0}$ in $\mathbb{R}$ such that*

$$\mathbb{E}\left[f(w_t)\right] \leq \min_w \left\{\mathbb{E}\left[V_t(w)\right]\right\} + \mathbb{E}\left[\delta_t\right] \tag{41}$$

holds for all $t \geq 0$, then

$$\mathbb{E}\left[f(w_t)\right] - f(w^*) \leq \theta_t(V_0(w^*) - f(w^*)) + \mathbb{E}\left[\delta_t\right].$$

The following lemma constructs a concrete stochastic estimation sequence.

Lemma 29 *Assume $f(w) = \frac{1}{n}\sum_{i=1}^n f_i(w)$, where each $f_i(w)$ is λ-strongly convex and L-smooth. Suppose that*

- *$V_0(w)$ is an arbitrary deterministic function on $\mathbb{R}^d$;*
- *$\{\alpha_t\}_{t\geq 0}$ is a sequence that satisfies $\alpha_t \in (0,1), \forall t \geq 0$, and $\sum_{t=0}^\infty \alpha_t = \infty$.*
- *$\{y_t\}_{t=0}^\infty$ is an arbitrary sequence in $\mathbb{R}^d$;*
- *Define $\{v_t\}_{t\geq 1}$ as:*

$$v_t = \frac{1}{b}\sum_{i\in B_t}\nabla f_i(y_{t-1}) - \frac{1}{b}\sum_{i\in B_t}\nabla f_i(\widetilde{w}_{s-1}) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\widetilde{w}_{s-1}).$$

Define the sequence $\{w_t\}_{t\geq 0}$ and $\{V_t(w)\}_{t\geq 0}$ as follows. Let $y_0 = w_0$ and w_t be arbitrary vector in $\mathbb{R}^d$ such that

$$w_t = y_{t-1} - \eta v_t + \xi_t,$$

where the stepsize η satisfies $0 < \eta \leq \frac{1}{L}$. Let

$$\begin{aligned} V_t(w) =& (1-\alpha_{t-1})V_{t-1}(w) \\ &+ \alpha_{t-1}\left(\frac{1}{b}\sum_{i\in B_t} f_i(y_{t-1}) + \left\langle v_t - \frac{\xi_t}{\eta}, w - y_{t-1}\right\rangle + \frac{\lambda}{4}\left\|w - y_{t-1}\right\|^2 - \frac{\left\|\xi_t\right\|^2}{\lambda\eta^2}\right), \end{aligned} \tag{42}$$

let $\theta_0 = 1$, and $\theta_t = (1-\alpha_{t-1})\theta_{t-1}, \forall t \geq 1$. Then the sequence $\{V_t(w), \theta_t\}_{t\geq 0}$ is a stochastic estimation sequence of $f(w)$.

By above construction, we know $V_t(w)$ is a quadratic function, its minimizer and minimum value has the following iterative form:

Lemma 30 *Define $\forall t \geq 0$:*

$$V_t^* = \min_w V_t(w),$$

suppose we choose the function $V_0(w)$ in Lemma 29 as

$$V_0(w) = V_0^* + \frac{\lambda}{4}\left\|w - z_0\right\|^2,$$

with $V_0^* = f(z_0), \epsilon_0 = 0$ and $\delta_0 = 0$ for $z_0 = w_0$. Then the sequence $\{V_t(w)\}_{t\geq 0}$ defined in 42 can be written as

$$V_t(w) = V_t^* + \frac{\lambda}{4}\left\|w - z_t\right\|^2, \tag{43}$$

where the sequences $\{V_t^*\}, \{z_t\}$ *are defined recursively as:*

$$z_t = (1-\alpha_{t-1})z_{t-1} + \alpha_{t-1}y_{t-1} - \frac{2\alpha_{t-1}}{\lambda}\left(v_t - \frac{\xi_t}{\eta}\right), \tag{44}$$

$$V_t^* = (1-\alpha_{t-1})V_{t-1}^* + \frac{\alpha_{t-1}(1-\alpha_{t-1})\lambda}{4}\|z_{t-1}-y_{t-1}\|^2 + \alpha_{t-1}(1-\alpha_{t-1})\left\langle v_t - \frac{\xi_t}{\eta}, z_{t-1}-y_{t-1}\right\rangle - \frac{\alpha_{t-1}^2}{\lambda}\|v_t\|^2 + \frac{\alpha_{t-1}}{b}\sum_{i\in B_t} f_i(y_{t-1}) - \frac{(\alpha_{t-1}+\alpha_{t-1}^2)\|\xi_t\|^2}{\lambda\eta^2} + \frac{2\alpha_{t-1}^2}{\lambda\eta}\langle \xi_t, v_t\rangle. \tag{45}$$

The following lemma establishes a connection between z_t, y_t, w_t when z_t is updated as (44), and y_t, w_t is updated as Algorithm 3.

Lemma 31 *Based on the updating rule of* z_t *in* (44)*, if we choose* $\alpha_t = \alpha = \sqrt{\frac{\lambda\eta}{2}}, \forall t \geq 0$*, we have the following inequalities hold for all* $t \geq 0$*:*

$$z_t - y_t = \frac{1}{\alpha}(y_t - w_t).$$

The following lemma gives concrete expression of δ_t such that for the IMBA-SVRG algorithm, the condition (41) in Lemma 28 is satisfied.

Lemma 32 *Suppose we choose* $\forall t \geq 0$

$$\alpha_t = \alpha = \sqrt{\frac{\lambda\eta}{2}},$$

and the stepsize η *satisfying* $\eta \leq \frac{1}{8L}$*, we have* $\forall t \geq 0$

$$\mathbb{E}\left[f(w_t)\right] \leq \mathbb{E}\left[V_t^*\right] + \mathbb{E}\left[\delta_t\right],$$

where $\delta_0 = 0$ *and* $\forall t \geq 1$*:*

$$\delta_t = \left(1 - \sqrt{\frac{\lambda\eta}{2}}\right)\delta_{t-1} + \left(\eta\|v_t - \nabla f(y_{t-1})\|^2 - \frac{(1-\sqrt{\lambda\eta/2})\lambda}{\sqrt{2\lambda\eta}}\|y_{t-1}-w_{t-1}\|^2 + \frac{1}{4\lambda^2\eta^3}\|\xi_t\|^2\right).$$

The lemma below bounds the term $\mathbb{E}\left[\eta\|v_t - \nabla f(y_{t-1})\|^2 - \frac{(1-\sqrt{\lambda\eta/2})\lambda}{\sqrt{2\lambda\eta}}\|y_{t-1}-w_{t-1}\|^2\right]$.

Lemma 33 *Suppose* B_t *is constructed by uniform sampling with or without replacement, we have the following inequality holds:*

$$\mathbb{E}\left[\eta\|v_t - \nabla f(y_{t-1})\|^2 - \frac{(1-\sqrt{\lambda\eta/2})\lambda}{\sqrt{2\lambda\eta}}\|y_{t-1}-w_{t-1}\|^2\right] \leq \frac{8\eta L}{b}\left(f(w_{t-1}) - f(w^*) + f(\widetilde{w}_{s-1}) - f(w^*)\right) + \left(\frac{2\eta L^2}{b} - \frac{(1-\sqrt{\lambda\eta/2})\lambda}{\sqrt{2\lambda\eta}}\right)\|y_{t-1}-w_{t-1}\|^2.$$

Based on above lemma, by carefully choosing the stepsize η, we get the following lemma which is important to obtain iteration complexity for IMBA-SVRG.

Lemma 34 *If we choose the stepsize satisfying*

$$\eta \leq \min\left\{\frac{b^2\lambda}{6400L^2}, \frac{1}{8L}\right\}$$

then the following inequality holds $\forall t \geq 0$*:*

$$\mathbb{E}f(w_t) - f(w^*) \leq \left(1 - \sqrt{\frac{\lambda\eta}{2}}\right)^t (V_0(w^*) - f(w^*)) + \mathbb{E}\left[\sum_{k=1}^{t}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-k}\left(\frac{1}{4\lambda^2\eta^3}\|\xi_k\|^2 + \frac{8\eta L}{b}(f(w_{k-1}) + f(\widetilde{w}_{s-1}) - 2f(w^*))\right)\right],$$

B.2 Proof of Lemma 28

Proof Since

$$\begin{aligned}\mathbb{E}\left[f(w_t)\right] \leq& \min_w\left\{\mathbb{E}\left[V_t(w)\right]\right\} + \mathbb{E}\left[\delta_t\right]\\ \leq& \mathbb{E}\left[V_t(w^*)\right] + \mathbb{E}\left[\delta_t\right]\\ \leq& (1-\theta_t)f(w^*) + \theta_t V_0(w^*) + \mathbb{E}\left[\delta_t\right],\end{aligned}$$

subtracting both sides by $f(w^*)$ finishes the proof. ■

B.3 Proof of Lemma 29

Proof We proceed the proof by induction. When $t = 0$, we have

$$V_0(w) = (1-\theta_0)f(w) + \theta_0 V_0(w) = V_0(w),$$

suppose

$$\mathbb{E}\left[V_{t-1}(w)\right] \leq (1-\theta_{t-1})f(w) + \theta_{t-1}V_0(w)$$

is true for some $t \geq 1$, then for $\mathbb{E}[V_t(w)]$, since $\mathbb{E}[v_t] = \nabla f(y_{t-1})$,we have

$$\begin{aligned}
\mathbb{E}[V_t(w)] =&(1-\alpha_{t-1})\mathbb{E}[V_{t-1}(w)] \\
&+\alpha_{t-1}\mathbb{E}\left[\left(\frac{1}{b}\sum_{i\in B_t} f_i(y_{t-1}) + \left\langle v_t - \frac{\xi_t}{\eta}, w - y_{t-1}\right\rangle + \frac{\lambda}{4}\|w-y_{t-1}\|^2 - \frac{\|\xi_t\|^2}{\lambda\eta^2}\right)\right] \\
=&(1-\alpha_{t-1})\mathbb{E}[V_{t-1}(w)] \\
&+\alpha_{t-1}\left(f(y_{t-1}) + \langle \nabla f(y_{t-1}), w - y_{t-1}\rangle + \frac{\lambda}{4}\|w-y_{t-1}\|^2 - \left\langle \frac{\xi_t}{\eta}, w - y_{t-1}\right\rangle - \frac{\|\xi_t\|^2}{\lambda\eta^2}\right) \\
\overset{①}{\leq}&(1-\alpha_{t-1})\mathbb{E}[V_{t-1}(w)] + \alpha_{t-1}\left(f(y_{t-1}) + \langle \nabla f(y_{t-1}), w - y_{t-1}\rangle + \frac{\lambda}{2}\|w-y_{t-1}\|^2\right) \\
\overset{②}{\leq}&(1-\alpha_{t-1})(1-\theta_{t-1})f(w) + (1-\alpha_{t-1})\theta_{t-1}V_0(w) + \alpha_{t-1}f(w) \\
=&(1-(1-\alpha_{t-1})\theta_{t-1})f(w) + (1-\alpha_{t-1})\theta_{t-1}V_0(w) \\
=&(1-\theta_t)f(w) + \theta_t V_0(w),
\end{aligned}$$

which concludes the proof, where at step ① we used the inequality $-\langle v_1, v_2\rangle - \|v_2\|^2 \leq \frac{\|v_1\|^2}{4}$,at step ② we used the inductive hypothesis and the fact that $f(w)$ is λ-strongly convex. ■

B.4 Proof of Lemma 30

Proof We proceed the proof by induction, when $t = 0$ this is true by construction. Suppose for some $t \geq 1$ the following holds:

$$V_{t-1}(w) = V_{t-1}^* + \frac{\lambda}{4}\|w - z_{t-1}\|^2,$$

then at time t, we have

$$\begin{aligned}
V_t(w) =&(1-\alpha_{t-1})V_{t-1}(w) \\
&+\alpha_{t-1}\left(\frac{1}{b}\sum_{i\in B_t} f_i(y_{t-1}) + \left\langle v_t - \frac{\xi_t}{\eta}, w - y_{t-1}\right\rangle + \frac{\lambda}{4}\|w-y_{t-1}\|^2 - \frac{\|\xi_t\|^2}{\lambda\eta^2}\right) \\
=&(1-\alpha_{t-1})V_{t-1}^* + \frac{(1-\alpha_{t-1})\lambda}{4}\|w - z_{t-1}\|^2 + \frac{\alpha_{t-1}\lambda}{4}\|w-y_{t-1}\|^2 + \alpha_{t-1}\left\langle v_t - \frac{\xi_t}{\eta}, w\right\rangle \\
&+\frac{\alpha_{t-1}}{b}\sum_{i\in B_t} f_i(y_{t-1}) - \alpha_{t-1}\left\langle v_t - \frac{\xi_t}{\eta}, y_{t-1}\right\rangle - \frac{\alpha_{t-1}\|\xi_t\|^2}{\lambda\eta^2},
\end{aligned}$$

by first order optimality condition, it is clear that the minimizer of $V_t(w)$: z_t satisfies the following:

$$(1-\alpha_{t-1})\lambda(z_t - z_{t-1}) + \alpha_{t-1}\lambda(z_t - y_{t-1}) + 2\alpha_{t-1}\left(v_t - \frac{\xi_t}{\eta}\right) = 0,$$

from above we obtain the recursive form of z_t as (44) defines. Plug in (44) to (42) we get

$$\begin{aligned}
V_t^* =&V_t(z_t) = (1-\alpha_{t-1})V_{t-1}^* + \frac{(1-\alpha_{t-1})\lambda}{4}\|z_t - z_{t-1}\|^2 + \frac{\alpha_{t-1}\lambda}{4}\|z_t - y_{t-1}\|^2 + \alpha_{t-1}\left\langle v_t - \frac{\xi_t}{\eta}, z_t\right\rangle \\
&+\frac{\alpha_{t-1}}{b}\sum_{i\in B_t} f_i(y_{t-1}) - \alpha_{t-1}\left\langle v_t - \frac{\xi_t}{\eta}, y_{t-1}\right\rangle - \frac{\alpha_{t-1}\|\xi_t\|^2}{\lambda\eta^2} \\
=&(1-\alpha_{t-1})V_{t-1}^* + \frac{(1-\alpha_{t-1})\lambda}{4}\left\|\alpha_{t-1}(y_{t-1} - z_{t-1}) - \frac{2\alpha_{t-1}}{\lambda}\left(v_t - \frac{\xi_t}{\eta}\right)\right\|^2 + \frac{\alpha_{t-1}}{b}\sum_{i\in B_t} f_i(y_{t-1}) \\
&+\frac{\alpha_{t-1}\lambda}{4}\left\|(1-\alpha_{t-1})(z_{t-1} - y_{t-1}) - \frac{2\alpha_{t-1}}{\lambda}\left(v_t - \frac{\xi_t}{\eta}\right)\right\|^2 - \frac{\alpha_{t-1}\|\xi_t\|^2}{\lambda\eta^2} \\
&+\alpha_{t-1}\left\langle v_t - \frac{\xi_t}{\eta}, (1-\alpha_{t-1})(z_{t-1} - y_{t-1}) - \frac{2\alpha_{t-1}}{\lambda}\left(v_t - \frac{\xi_t}{\eta}\right)\right\rangle \\
=&(1-\alpha_{t-1})V_{t-1}^* + \frac{\alpha_{t-1}(1-\alpha_{t-1})\lambda}{4}\|z_{t-1} - y_{t-1}\|^2 + \alpha_{t-1}(1-\alpha_{t-1})\left\langle v_t - \frac{\xi_t}{\eta}, z_{t-1} - y_{t-1}\right\rangle \\
&-\frac{\alpha_{t-1}^2}{\lambda}\|v_t\|^2 + \frac{\alpha_{t-1}}{b}\sum_{i\in B_t} f_i(y_{t-1}) - \frac{(\alpha_{t-1} + \alpha_{t-1}^2)\|\xi_t\|^2}{\lambda\eta^2} + \frac{2\alpha_{t-1}^2}{\lambda\eta}\langle \xi_t, v_t\rangle,
\end{aligned}$$

which verified (45). ■

B.5 Proof of Lemma 31

Proof We prove by induction, when $t = 0$ it is obviously true, suppose it is true for $t-1, \forall t \geq 1$, i.e.

$$z_{t-1} - y_{t-1} = \frac{1}{\alpha}(y_{t-1} - w_{t-1}),$$

for iteration t, based on (44) we have

$$\begin{aligned}
z_t - y_t =&(1-\alpha)z_{t-1} + \alpha y_{t-1} - \frac{2\alpha}{\lambda}\left(v_t - \frac{\xi_t}{\eta}\right) - y_t \\
=&(1-\alpha)(z_{t-1} - y_{t-1}) + y_{t-1} - \frac{2\alpha}{\lambda}\left(v_t - \frac{\xi_t}{\eta}\right) - y_t \\
\overset{①}{=}&\frac{1-\alpha}{\alpha}(y_{t-1} - w_{t-1}) + y_{t-1} - \frac{2\alpha}{\lambda}\left(v_t - \frac{\xi_t}{\eta}\right) - y_t \\
=&\frac{1}{\alpha}\left(y_{t-1} - \frac{2\alpha^2}{\lambda}\left(v_t - \frac{\xi_t}{\eta}\right)\right) - \frac{1-\alpha}{\alpha}w_{t-1} - y_t \\
\overset{②}{=}&\frac{1}{\alpha}w_t - \frac{1-\alpha}{\alpha}w_{t-1} - y_t \\
\overset{③}{=}&\frac{1}{\alpha}\left(w_t + \frac{1-\alpha}{1+\alpha}(w_t - w_{t-1}) - w_t\right) \\
\overset{④}{=}&\frac{1}{\alpha}(y_t - w_t),
\end{aligned}$$

which concludes the proof, where step ① used the inductive hypothesis, step ② used the update rule of w_t in Algorithm 3, step ③ and ④ used the update rule of y_t in Algorithm 3. ■

B.6 Proof of Lemma 32

Proof We prove by induction, when $t = 0$, it is true that

$$f(w_0) \leq V_0^* = f(z_0) = f(w_0),$$

suppose

$$\mathbb{E}\left[f(w_{t-1})\right] \leq \mathbb{E}\left[V_{t-1}^*\right] + \mathbb{E}\left[\delta_{t-1}\right], \tag{46}$$

for some $t \geq 1$, then based on smoothness, we know

$$f(w_t) \leq f(y_{t-1}) + \langle \nabla f(y_{t-1}), w_t - y_{t-1}\rangle + \frac{L}{2}\left\|w_t - y_{t-1}\right\|^2,$$

thus

$$\begin{aligned}
V_t^* =& (1-\alpha_{t-1})V_{t-1}^* + \frac{\alpha_{t-1}(1-\alpha_{t-1})\lambda}{4}\left\|z_{t-1} - y_{t-1}\right\|^2 \\
&+ \alpha_{t-1}(1-\alpha_{t-1})\left\langle v_t - \frac{\xi_t}{\eta}, z_{t-1} - y_{t-1}\right\rangle \\
&- \frac{\alpha_{t-1}^2}{\lambda}\left\|v_t\right\|^2 + \frac{\alpha_{t-1}}{b}\sum_{i\in B_t} f_i(y_{t-1}) - \frac{(\alpha_{t-1} + \alpha_{t-1}^2)\left\|\xi_t\right\|^2}{\lambda\eta^2} + \frac{2\alpha_{t-1}^2}{\lambda\eta}\langle \xi_t, v_t\rangle.
\end{aligned}$$

Hence we have

$$\begin{aligned}
&\mathbb{E}\left[f(w_t) - V_t^*\right] \leq \mathbb{E}\left[f(y_{t-1}) + \langle \nabla f(y_{t-1}), w_t - y_{t-1}\rangle + \frac{L}{2}\left\|w_t - y_{t-1}\right\|^2\right] \\
&- \mathbb{E}\left[(1-\alpha)V_{t-1}^* - \frac{\alpha(1-\alpha)\lambda}{4}\left\|z_{t-1} - y_{t-1}\right\|^2 - \alpha(1-\alpha)\left\langle v_t - \frac{\xi_t}{\eta}, z_{t-1} - y_{t-1}\right\rangle\right] \\
&+ \mathbb{E}\left[\frac{\alpha^2}{\lambda}\left\|v_t\right\|^2 - \frac{\alpha}{b}\sum_{i\in B_t} f_i(y_{t-1}) + \frac{(\alpha+\alpha^2)\left\|\xi_t\right\|^2}{\lambda\eta^2} - \frac{2\alpha^2}{\lambda\eta}\langle \xi_t, v_t\rangle\right] \\
&\overset{①}{=} \mathbb{E}\left[(1-\alpha)(f(y_{t-1}) - V_{t-1}^* + \langle v_t, w_{t-1} - y_{t-1}\rangle) - \langle \nabla f(y_{t-1}), \eta v_t - \xi_t\rangle + \frac{\eta}{2}\left\|v_t\right\|^2\right] \\
&+ \mathbb{E}\left[\frac{L}{2}\left\|\eta v_t - \xi_t\right\|^2 - \frac{(1-\alpha)\lambda}{4\alpha}\left\|y_{t-1} - w_{t-1}\right\|^2 + \frac{(\alpha+\alpha^2)\left\|\xi_t\right\|^2}{\lambda\eta^2}\right] \\
&+ \mathbb{E}\left[-\frac{2\alpha^2}{\lambda\eta}\langle \xi_t, v_t\rangle - (1-\alpha)\left\langle \frac{\xi_t}{\eta}, w_{t-1} - y_{t-1}\right\rangle\right] \\
&\overset{②}{\leq} \mathbb{E}\left[(1-\alpha)(f(y_{t-1}) - V_{t-1}^* + \langle v_t, w_{t-1} - y_{t-1}\rangle) - \langle \nabla f(y_{t-1}), \eta v_t - \xi_t\rangle + \frac{9\eta}{16}\left\|v_t\right\|^2\right] \\
&+ \mathbb{E}\left[\frac{L}{2}\left\|\eta v_t - \xi_t\right\|^2 + \left(\frac{2\alpha}{\lambda\eta^2} + \frac{16\alpha^4}{\lambda^2\eta^3} + \frac{(1-\alpha)}{2\lambda\eta^2}\right)\left\|\xi_t\right\|^2\right] \\
&+ \mathbb{E}\left[\left(\frac{(1-\alpha)\lambda}{2} - \frac{(1-\alpha)\lambda}{4\alpha}\right)\left\|w_{t-1} - y_{t-1}\right\|^2\right]
\end{aligned} \tag{47}$$

where at step ① we used the fact of updating rule

$$w_t = y_{t-1} - \eta v_t + \xi_t,$$

as well as Lemma 31, and $\mathbb{E}\left[\frac{1}{b}\sum_{i\in B_t} f_i(y_{t-1})\right] = f(y_{t-1})$; and at step ② we used the following two inequalities:

$$\begin{aligned}
\left|(1-\alpha)\left\langle \frac{\xi_t}{\eta}, w_{t-1} - y_{t-1}\right\rangle\right| \leq& \frac{(1-\alpha)\lambda}{2}\left\|y_{t-1} - w_{t-1}\right\|^2 + \frac{(1-\alpha)}{2\lambda\eta^2}\left\|\xi_t\right\|^2, \\
\left|\frac{2\alpha^2}{\lambda\eta}\langle \xi_t, v_t\rangle\right| \leq& \frac{\eta}{16}\left\|v_t\right\|^2 + \frac{16\alpha^4}{\lambda^2\eta^3}\left\|\xi_t\right\|^2
\end{aligned}$$

Since

$$\begin{aligned}
\mathbb{E}\left[f(y_{t-1}) - V_{t-1}^* + \langle v_t, w_{t-1} - y_{t-1}\rangle\right] =& \mathbb{E}\left[f(y_{t-1}) - V_{t-1}^* + \langle \nabla f(y_{t-1}), w_{t-1} - y_{t-1}\rangle\right] \\
\overset{①}{\leq}& \mathbb{E}\left[f(w_{t-1}) - \frac{\lambda}{2}\left\|w_{t-1} - y_{t-1}\right\|^2 - V_{t-1}^*\right] \\
\overset{②}{\leq}& \mathbb{E}\left[\delta_{t-1} - \frac{\lambda}{2}\left\|w_{t-1} - y_{t-1}\right\|^2\right],
\end{aligned} \tag{48}$$

where at step ① we used the λ-strong convexity of $f(w)$, at step ② we used the inductive hypothesis (46). Combining (47) and (48) we obtain

$$\begin{aligned}
&\mathbb{E}\left[f(w_t) - V_t^*\right] \leq (1-\alpha)\delta_{t-1} - \mathbb{E}\left[\frac{(1-\alpha)\lambda}{4\alpha}\left\|y_{t-1} - w_{t-1}\right\|^2\right] \\
&+ \mathbb{E}\left[\frac{L}{2}\left\|\eta v_t - \xi_t\right\|^2 + \frac{9\eta}{16}\left\|v_t\right\|^2 - \langle \nabla f(y_{t-1}), \eta v_t - \xi_t\rangle + \left(\frac{2\alpha}{\lambda\eta^2} + \frac{16\alpha^4}{\lambda^2\eta^3} + \frac{(1-\alpha)}{2\lambda\eta^2}\right)\left\|\xi_t\right\|^2\right].
\end{aligned} \tag{49}$$

Next we bound the third term on the right hand side, since

$$\begin{aligned}
&\frac{L}{2}\left\|\eta v_t - \xi_t\right\|^2 + \frac{9\eta}{16}\left\|v_t\right\|^2 - \langle \nabla f(y_{t-1}), \eta v_t + \xi_t\rangle \\
&\overset{①}{\leq} L\eta^2\left\|v_t\right\|^2 + L\left\|\xi_t\right\|^2 + \frac{9\eta}{16}\left\|v_t\right\|^2 - \langle \nabla f(y_{t-1}), \eta v_t - \xi_t\rangle \\
&= \frac{\eta}{2}\left\|v_t - \nabla f(y_{t-1})\right\|^2 - \frac{\eta}{2}\left\|\nabla f(y_{t-1})\right\|^2 \\
&+ \left(L\eta^2 + \frac{\eta}{16}\right)\left\|v_t\right\|^2 + \langle \nabla f(y_{t-1}), \xi_t\rangle + L\left\|\xi_t\right\|^2 \\
&\overset{②}{\leq} \frac{\eta}{2}\left\|v_t - \nabla f(y_{t-1})\right\|^2 - \frac{\eta}{2}\left\|\nabla f(y_{t-1})\right\|^2 \\
&+ \left(2L\eta^2 + \frac{\eta}{8}\right)\left(\left\|v_t - \nabla f(y_{t-1})\right\|^2 + \left\|\nabla f(y_{t-1})\right\|^2\right) \\
&+ \frac{\eta}{8}\left\|\nabla f(y_{t-1})\right\|^2 + \frac{2}{\eta}\left\|\xi_t\right\|^2 + L\left\|\xi_t\right\|^2,
\end{aligned}$$

where at step ① we used $\|v_1+v_2\|^2 \le 2(\|v_1\|^2+\|v_2\|^2)$, and at step ② we used it again and also the inequality $2|\langle v_1, v_2\rangle| \le \|v_1\|^2 + \|v_2\|^2$. Thus we know when $\eta \le \frac{1}{8L}$, we have

$$\begin{aligned}
&\frac{L}{2}\|\eta v_t - \xi_t\|^2 + \frac{9\eta}{16}\|v_t\|^2 - \langle \nabla f(y_{t-1}), \eta v_t + \xi_t\rangle \qquad (50)\\
&\le \left(\frac{\eta}{2} + 2L\eta^2 + \frac{\eta}{8}\right)\|v_t - \nabla f(y_{t-1})\|^2 + \left(L + \frac{2}{\eta}\right)\|\xi_t\|^2\\
&- \left(\frac{\eta}{2} - \frac{\eta}{8} - \frac{\eta}{8} - 2L\eta^2\right)\|\nabla f(y_{t-1})\|^2\\
&\le \eta\|v_t - \nabla f(y_{t-1})\|^2 + \frac{3}{\eta}\|\xi_t\|^2. \qquad (51)
\end{aligned}$$

Combining (49) and (51) we obtain

$$\begin{aligned}
\mathbb{E}\left[f(w_t) - V_t^*\right] \le& (1-\alpha)\delta_{t-1} + \mathbb{E}\left[\eta\|v_t - \nabla f(y_{t-1})\|^2 - \frac{(1-\alpha)\lambda}{4\alpha}\|y_{t-1} - w_{t-1}\|^2\right]\\
&+ \mathbb{E}\left[\left(\frac{2\alpha}{\lambda\eta^2} + \frac{16\alpha^4}{\lambda^2\eta^3} + \frac{(1-\alpha)}{2\lambda\eta^2} + \frac{3}{\eta}\right)\|\xi_t\|^2\right], \qquad (52)
\end{aligned}$$

also since

$$\alpha = \sqrt{\frac{\lambda\eta}{2}} \le \frac{1}{4}\sqrt{\frac{\lambda}{L}} \le \frac{1}{4},$$

we know

$$\frac{2\alpha}{\lambda\eta^2} + \frac{16\alpha^4}{\lambda^2\eta^3} + \frac{(1-\alpha)}{2\lambda\eta^2} + \frac{3}{\eta} \le \frac{1}{16\lambda^2\eta^3} + \frac{1}{16\lambda^2\eta^3} + \frac{1}{16\lambda^2\eta^3} + \frac{3}{64\lambda^2\eta^3} \le \frac{1}{4\lambda^2\eta^3},$$

combining the inequality above and (52), then substituting $\alpha = \sqrt{\frac{\lambda\eta}{2}}$ finishes the proof. ■

B.7 Proof of Lemma 33

Proof The proof follows the strategy in (Johnson and Zhang, 2013; Nitanda, 2014). First, based on the minibatch sampling (here for simplicity we only consider sampling with replacement, for sampling without replacement, the bound below can be tightened by a factor

of $\frac{n-b}{n-1}$, see, e.g. Section 2.8 of (Lohr, 2009)) we know

$$\begin{aligned}
\mathbb{E}\left[\|v_t - \nabla f(y_{t-1})\|^2\right] =& \frac{1}{b^2}\mathbb{E}\left[\sum_{j\in B_t}\|\nabla f_j(y_{t-1}) - \nabla f(y_{t-1}) - (\nabla f_j(\widetilde{w}_{s-1}) - \nabla f(\widetilde{w}_{s-1}))\|^2\right]\\
\overset{①}{\le}& \frac{1}{b^2}\mathbb{E}\left[\sum_{j\in B_t}\|\nabla f_j(y_{t-1}) - \nabla f_j(\widetilde{w}_{s-1})\|^2\right]\\
\overset{②}{\le}& \frac{2}{b^2}\mathbb{E}\left[\sum_{j\in B_t}(\|\nabla f_j(y_{t-1}) - \nabla f_j(w_{t-1})\|^2 + \|\nabla f_j(\widetilde{w}_{s-1}) - \nabla f_j(w_{t-1})\|^2)\right]\\
\overset{③}{\le}& \frac{4}{b^2}\mathbb{E}\left[\sum_{j\in B_t}(\|\nabla f_j(\widetilde{w}_{s-1}) - \nabla f_j(w^*)\|^2) + \|\nabla f_j(w^*) - \nabla f_j(w_{t-1})\|^2)\right]\\
&+ \frac{2L^2}{b}\|y_{t-1} - w_{t-1}\|^2\\
\overset{④}{\le}& \frac{8L}{b}\left(f(w_{t-1}) - f(w^*) + f(\widetilde{w}_{s-1}) - f(w^*)\right) + \frac{2L^2}{b}\|y_{t-1} - w_{t-1}\|^2,
\end{aligned}$$

where at step ① we used $\mathbb{E}\|v - \mathbb{E}v\|^2 \le \mathbb{E}\|v\|^2$, at step ② we used $\|v_1+v_2\|^2 \le 2(\|v_1\|^2 + \|v_2\|^2)$, at step ③ we used it again along with the L-smoothness of $f_j(w)$, at step ④ we used standard results in SVRG analysis, e.g. Lemma 1 of (Xiao and Zhang, 2014). Substituting above into the term $\mathbb{E}\left[\eta\|v_t - \nabla f(y_{t-1})\|^2 - \frac{(1-\sqrt{\lambda\eta/2})\lambda}{\sqrt{2\lambda\eta}}\|y_{t-1} - w_{t-1}\|^2\right]$ concludes the proof. ■

B.8 Proof of Lemma 34

Proof Combining Lemma 32 and 33, we have

$$\begin{aligned}
\delta_t =& \left(1 - \sqrt{\frac{\lambda\eta}{2}}\right)\delta_{t-1} \qquad (53)\\
&+ \left(\eta\|v_t - \nabla f(y_{t-1})\|^2 - \frac{(1-\sqrt{\lambda\eta/2})\lambda}{\sqrt{2\lambda\eta}}\|y_{t-1} - w_{t-1}\|^2 + \frac{1}{4\lambda^2\eta^3}\|\xi_t\|^2\right)\\
\le& \left(1 - \sqrt{\frac{\lambda\eta}{2}}\right)\delta_{t-1} + \left(\frac{1}{4\lambda^2\eta^3}\|\xi_t\|^2 + \frac{8\eta L}{b}(f(w_{t-1}) + f(\widetilde{w}_{s-1}) - 2f(w^*))\right)\\
&+ \left(\frac{2\eta L^2}{b} - \frac{(1-\sqrt{\lambda\eta/2})\lambda}{\sqrt{2\lambda\eta}}\right)\|y_{t-1} - w_{t-1}\|^2. \qquad (54)
\end{aligned}$$

First we verify the factor before $\|y_{t-1} - w_{t-1}\|^2$ is non positive, since $1 - \eta\lambda \ge 1 - \frac{\lambda}{8L} \ge \frac{1}{2}$, then

$$\frac{2\eta L^2}{b}\cdot\frac{\sqrt{2\lambda\eta}}{(1-\sqrt{\eta\lambda/2})\lambda} \le \frac{4\eta^{3/2}L^2}{b\sqrt{\lambda}} \le \min\left\{\frac{b^2\lambda}{128000L}, \frac{L^{1/2}}{4b\sqrt{\lambda}}\right\} \le 1,$$

where the last inequality is true because

$$\frac{b^2\lambda}{128000L}\left(\frac{L^{1/2}}{4b\sqrt{\lambda}}\right)^2 \leq 1,$$

thus we must have $\min\left\{\frac{b^2\lambda}{128000L}, \frac{L^{1/2}}{4b\sqrt{\lambda}}\right\} \leq 1$ otherwise it leads to a contradiction. Thus $\frac{2\eta L^2}{b} - \frac{(1-\sqrt{\eta\lambda/2})\lambda}{\sqrt{2\eta\lambda}} \leq 0$, combining with (54), we have

$$\delta_t \leq \left(1-\sqrt{\frac{\lambda\eta}{2}}\right)\delta_{t-1} + \left(\frac{1}{4\lambda^2\eta^3}\left\|\xi_t\right\|^2 + \frac{8\eta L}{b}(f(w_{t-1}) + f(\widetilde{w}_{s-1}) - 2f(w^*))\right),$$

applying above inequality recursively, we get

$$\delta_t \leq \sum_{k=1}^{t}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-k}\left(\frac{1}{4\lambda^2\eta^3}\left\|\xi_k\right\|^2 + \frac{8\eta L}{b}(f(w_{k-1}) + f(\widetilde{w}_{s-1}) - 2f(w^*))\right),$$

then applying Lemma 28 and Lemma 32 concludes the proof. ■

B.9 Proof of Theorem 24

Proof We prove this theorem by induction, note that when $b \leq 20\sqrt{\frac{2L}{\lambda}}$, it is clear that

$$\frac{8\eta L}{b} \leq \frac{b^2\lambda}{6400L^2}\cdot\frac{8L}{b} = \frac{\lambda b}{800L} = \frac{\sqrt{\eta\lambda}}{10},$$

and when $b \geq 20\sqrt{\frac{2L}{\lambda}}$,

$$\frac{8\eta L}{b} = \frac{1}{b} \leq \frac{\sqrt{\lambda}}{20\sqrt{2L}} = \frac{\sqrt{\eta\lambda}}{10},$$

thus in both cases we have $\frac{8\eta L}{b} \leq \frac{\sqrt{\eta\lambda}}{10}$, thus

$$\begin{aligned}\sum_{k=1}^{t}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-k}\left(\frac{8\eta L}{b}\right) \leq& \frac{1}{10}\sum_{k=1}^{t}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-k}\sqrt{\lambda\eta}\\ \leq& \frac{1}{10}\sum_{k=1}^{\infty}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{k}\sqrt{\lambda\eta} \leq \frac{\sqrt{2}}{10}.\end{aligned}$$

By Lemma 34 we know

$$\begin{aligned}\mathbb{E}f(w_t) - f(w^*) \leq& \left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t}(V_0(w^*) - f(w^*)) + \frac{\sqrt{2}(f(\widetilde{w}_{s-1}) - f(w^*))}{10} \qquad (55)\\ &+ \mathbb{E}\left[\sum_{k=1}^{t}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-k}\left(\frac{1}{4\lambda^2\eta^3}\left\|\xi_k\right\|^2 + \frac{\sqrt{\lambda\eta}}{10}(f(w_{k-1}) - f(w^*))\right)\right], \qquad (56)\end{aligned}$$

also when $\forall k \in [t]$,

$$\left\|\xi_k\right\|^2 \leq \frac{2\lambda^2\eta^3\sqrt{\lambda\eta}}{15}(f(\widetilde{w}_{s-1}) - f(w^*)),$$

we know

$$\begin{aligned}\sum_{k=1}^{t}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-k}\left(\frac{1}{4\lambda^2\eta^3}\left\|\xi_k\right\|^2\right) \leq& \frac{1}{30}\sum_{k=1}^{t}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-k}\sqrt{\lambda\eta}(f(\widetilde{w}_{s-1}) - f(w^*))\\ \leq& \frac{1}{30}\sum_{k=1}^{\infty}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{k}\sqrt{\lambda\eta}(f(\widetilde{w}_{s-1}) - f(w^*))\\ \leq& \frac{(f(\widetilde{w}_{s-1}) - f(w^*))}{20}. \qquad (57)\end{aligned}$$

Combining (56) and (57), we have

$$\begin{aligned}\mathbb{E}f(w_t) - f(w^*) \leq& \left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t}(V_0(w^*) - f(w^*)) + \frac{(f(\widetilde{w}_{s-1}) - f(w^*))}{5}\\ &+ \mathbb{E}\left[\sum_{k=1}^{t}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-k}\left(\frac{\sqrt{\lambda\eta}}{10}(f(w_{k-1}) - f(w^*))\right)\right],\end{aligned}$$

denote A_t be the right hand side on above inequality, i.e.

$$\begin{aligned}A_t =& \left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t}(V_0(w^*) - f(w^*)) + \frac{(f(\widetilde{w}_{s-1}) - f(w^*))}{5}\\ &+ \mathbb{E}\left[\sum_{k=1}^{t}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-k}\left(\frac{\sqrt{\lambda\eta}}{10}(f(w_{k-1}) - f(w^*))\right)\right]\\ =& \left(1-\sqrt{\frac{\lambda\eta}{2}}\right)\left(\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-1}(V_0(w^*) - f(w^*)) + \frac{(f(\widetilde{w}_{s-1}) - f(w^*))}{5}\right)\\ &+ \left(1-\sqrt{\frac{\lambda\eta}{2}}\right)\mathbb{E}\left[\sum_{k=1}^{t-1}\left(1-\sqrt{\frac{\lambda\eta}{2}}\right)^{t-1-k}\left(\frac{\sqrt{\lambda\eta}}{10}(f(w_{k-1}) - f(w^*))\right)\right]\\ &+ \frac{\sqrt{\lambda\eta}}{5\sqrt{2}}(f(\widetilde{w}_{s-1}) - f(w^*)) + \frac{\sqrt{\lambda\eta}}{10}(f(w_{t-1}) - f(w^*))\\ \overset{①}{\leq}& \left(1-\sqrt{\frac{\lambda\eta}{2}}\right)A_{t-1} + \frac{\sqrt{\lambda\eta}}{5\sqrt{2}}A_0 + \frac{\sqrt{\lambda\eta}}{10}A_{t-1}\\ =& \left(1-\frac{9\sqrt{\lambda\eta}}{10}\right)A_{t-1} + \frac{\sqrt{\lambda\eta}}{5\sqrt{2}}A_0,\end{aligned}$$

where at step ① we used the fact that $f(\widetilde{w}_{s-1}) - f(w^*) \leq A_0$, and $f(w_{t-1}) - f(w^*) \leq A_{t-1}$, applying above inequality recursively we get

$$\begin{aligned} A_t &\leq \left(1 - \frac{9\sqrt{\lambda\eta}}{10}\right)^t A_0 + \frac{\sqrt{\lambda\eta}}{5\sqrt{2}} \sum_{k=0}^{t-1} \left(1 - \frac{9\sqrt{\lambda\eta}}{10}\right)^k A_0 \\ &\leq \left(1 - \frac{9\sqrt{\lambda\eta}}{10}\right)^t A_0 + \frac{\sqrt{\lambda\eta}}{5\sqrt{2}} \sum_{k=0}^{\infty} \left(1 - \frac{9\sqrt{\lambda\eta}}{10}\right)^k A_0 \leq \left(\left(1 - \frac{9\sqrt{\lambda\eta}}{10}\right)^t + \frac{2}{9}\right) A_0, \end{aligned}$$

thus we know when $t \geq \frac{10}{9\sqrt{\lambda\eta}} \log(36)$, we have

$$A_t \leq \left(\frac{1}{36} + \frac{2}{9}\right) A_0 = \frac{1}{4} A_0,$$

also since

$$A_0 = V_0(w^*) - f(w^*) = f(\widetilde{w}_{s-1}) - f(w^*) + \frac{\lambda}{2} \|\widetilde{w}_{s-1} - w^*\| \overset{①}{\leq} 2(f(\widetilde{w}_{s-1}) - f(w^*)),$$

where at step ① we used the λ-strong convexity of $f(w)$. Combine above two inequality we get when $t \geq \frac{10}{9\sqrt{\lambda\eta}} \log(36)$, the expected objective suboptimality is halved:

$$\mathbb{E} f(w_t) - f(w^*) \leq A_t \leq \frac{1}{4} A_0 \leq \frac{1}{2}(f(\widetilde{w}_{s-1}) - f(w^*)),$$

which concludes the proof. ■

Appendix C. Collections of Tools in the Analysis

Lemma 35 ***(Matrix Bernstein**, Theorem 1.4 of (Tropp, 2012) rephrased) Let $X_1, ..., X_k$ be some independent, self-adjoint random matrices with dimension d, and assume each random matrix satisfies:*

$$\mathbb{E} X_k = 0 \quad \text{and} \quad \|X_k\| \leq R \quad \textit{almost surely.}$$

Then, for all $t \geq 0$,

$$P\left(\left\|\sum_k X_k\right\| \geq t\right) \leq d \exp\left(\frac{-t^2/2}{\sigma^2 + Rt/3}\right)$$

where

$$\sigma^2 := \left\|\sum_k \mathbb{E}(X_k^2)\right\|.$$

Lemma 36 ***(Iteration complexity of SVRG**, Corollary 1 of (Xiao and Zhang, 2014) rephrased) If we apply SVRG to any finite-sum optimization objective $f(w)$ where each individual function is λ-strongly convex and L-smooth, then there are universal constant C such that for any target accuracy ε, and success probability $1 - \delta$, SVRG is able to find a solution that satisfies ε objective suboptimality using*

$$C \cdot \left(n + \frac{L}{\lambda}\right) \log\left(\frac{\varepsilon_0}{\delta\varepsilon}\right)$$

first order oracle calls of individual functions, where n is the total number of individual functions in $f(w)$, ε_0 is the initial objective suboptimality.

Lemma 37 ***(Iteration complexity of catalyst acceleration**, Theorem 3.1 of (Lin et al., 2015a) rephrased) For any λ-strongly convex and L-smooth function $f(w)$, If the minimization step of (23) satisfies*

$$f(w_r) - \frac{\gamma}{2}\|w_r - z_{r-1}\| - \min_w \left(f(w) + \frac{\gamma}{2}\|w - z_{r-1}\|\right) \leq \frac{2}{9}(f(w_0) - f(w^*)) \left(1 - \frac{9}{10}\sqrt{\frac{\lambda}{\lambda+\gamma}}\right)^r,$$

then if we initialize $\nu_0 = \sqrt{\frac{\lambda}{\lambda+\gamma}}$ and set ν_r such that $\nu_r^2 = (1-\nu_r)\nu_{r-1}^2 + (\lambda\nu_r)/(\lambda+\gamma)$, then the sequences $\{w_r\}$ in Algorithm 2 satisfies

$$f(w_r) - f(w^*) \leq \frac{800(\lambda+\gamma)}{\lambda} \left(1 - \frac{9}{10}\sqrt{\frac{\lambda}{\lambda+\gamma}}\right)^{r+1} (f(w_0) - f(w^*)).$$

References

Naman Agarwal, Brian Bullins, and Elad Hazan. Second order stochastic optimization in linear time. *arXiv preprint arXiv:1602.03943*, 2016.

Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. *arXiv preprint arXiv:1603.05953*, 2016.

Zeyuan Allen-Zhu and Elad Hazan. Optimal black-box reductions between optimization objectives. In *Advances in Neural Information Processing Systems*, pages 1614–1622, 2016.

Yossi Arjevani and Ohad Shamir. Oracle complexity of second-order methods for finite-sum problems. *arXiv preprint arXiv:1611.04982*, 2016.

Raghu Bollapragada, Richard Byrd, and Jorge Nocedal. Exact and inexact subsampled newton methods for optimization. *arXiv preprint arXiv:1609.08502*, 2016.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *arXiv preprint arXiv:1606.04838*, 2016.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Richard H Byrd, Gillian M Chin, Will Neveitt, and Jorge Nocedal. On the use of stochastic hessian information in optimization methods for machine learning. *SIAM Journal on Optimization*, 21(3):977–995, 2011.

Richard H Byrd, Samantha L Hansen, Jorge Nocedal, and Yoram Singer. A stochastic quasi-newton method for large-scale optimization. *SIAM Journal on Optimization*, 26 (2):1008–1031, 2016.

Samprit Chatterjee and Ali S Hadi. *Sensitivity analysis in linear regression*, volume 327. John Wiley & Sons, 2009.

Andrew Cotter, Ohad Shamir, Nati Srebro, and Karthik Sridharan. Better mini-batch algorithms via accelerated gradient methods. In *Advances in neural information processing systems*, pages 1647–1655, 2011.

Aaron Defazio. A simple practical accelerated method for finite sums. In *Advances In Neural Information Processing Systems*, pages 676–684, 2016.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pages 1646–1654, 2014.

Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13(Jan):165–202, 2012.

Olivier Devolder, François Glineur, and Yurii Nesterov. First-order methods of smooth convex optimization with inexact oracle. *Mathematical Programming*, 146(1-2):37–75, 2014.

Petros Drineas and Michael W Mahoney. Lectures on randomized numerical linear algebra. *arXiv preprint arXiv:1712.08880*, 2017.

Murat A. Erdogdu and Andrea Montanari. Convergence rates of sub-sampled newton methods. In *Advances in Neural Information Processing Systems*, pages 3052–3060, 2015.

Roy Frostig, Rong Ge, Sham Kakade, and Aaron Sidford. Un-regularizing: approximate proximal point and faster stochastic algorithms for empirical risk minimization. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 2540–2548, 2015.

Alon Gonen, Francesco Orabona, and Shai Shalev-Shwartz. Solving ridge regression using sketched preconditioned svrg. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1397–1405, 2016.

Robert Gower, Donald Goldfarb, and Peter Richtárik. Stochastic block bfgs: squeezing more curvature out of data. In *International Conference on Machine Learning*, pages 1869–1878, 2016.

Daniel Hsu, Sham M Kakade, and Tong Zhang. Random design analysis of ridge regression. *Foundations of Computational Mathematics*, 3(14):569–600, 2014.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

Jakub Konečnỳ, Jie Liu, Peter Richtárik, and Martin Takáč. Mini-batch semi-stochastic gradient descent in the proximal setting. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):242–255, 2016.

Guanghui Lan. An optimal method for stochastic composite optimization. *Mathematical Programming*, 133(1):365–397, 2012.

Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J. Smola. Efficient mini-batch training for stochastic optimization. In *Proc. of the 20th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (SIGKDD 2014)*, pages 661–670, 2014.

Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems*, pages 3384–3392, 2015a.

Qihang Lin, Xi Chen, and Javier Peña. A sparsity preserving stochastic gradient methods for sparse regression. *Computational Optimization and Applications*, 58(2):455–482, 2014.

Qihang Lin, Zhaosong Lu, and Lin Xiao. An accelerated randomized proximal coordinate gradient method and its application to regularized empirical risk minimization. *SIAM Journal on Optimization*, 25(4):2244–2273, 2015b.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Sharon Lohr. *Sampling: design and analysis*. Nelson Education, 2009.

Michael W Mahoney et al. Randomized algorithms for matrices and data. *Foundations and Trends® in Machine Learning*, 3(2):123–224, 2011.

Philipp Moritz, Robert Nishihara, and Michael Jordan. A linearly-convergent stochastic l-bfgs algorithm. In *Artificial Intelligence and Statistics*, pages 249–258, 2016.

Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*, volume 87. Springer Science & Business Media, 2004.

Atsushi Nitanda. Stochastic proximal gradient descent with acceleration techniques. In *Advances in Neural Information Processing Systems*, pages 1574–1582, 2014.

Mert Pilanci and Martin J Wainwright. Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence. *SIAM Journal on Optimization*, 27(1):205–245, 2017.

Zheng Qu, Peter Richtárik, Martin Takáč, and Olivier Fercoq. Sdna: Stochastic dual newton ascent for empirical risk minimization. *arXiv preprint arXiv:1502.02268*, 2015.

Farbod Roosta-Khorasani and Michael W Mahoney. Sub-sampled newton methods i: globally convergent algorithms. *arXiv preprint arXiv:1601.04737*, 2016a.

Farbod Roosta-Khorasani and Michael W Mahoney. Sub-sampled newton methods ii: Local convergence rates. *arXiv preprint arXiv:1601.04738*, 2016b.

Nicolas L Roux, Mark Schmidt, and Francis R Bach. A stochastic gradient method with an exponential convergence _rate for finite training sets. In *Advances in Neural Information Processing Systems*, pages 2663–2671, 2012.

Mark Schmidt, Nicolas L Roux, and Francis R Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in neural information processing systems*, pages 1458–1466, 2011.

Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599, 2013.

Shai Shalev-Shwartz and Tong Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *Mathematical Programming*, 155(1-2):105–145, 2016.

Ohad Shamir, Nathan Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate Newton-type method. In *Proc. of the 31st Int. Conf. Machine Learning (ICML 2014)*, pages 1000–1008, 2014.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Joel A Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434, 2012.

Silvia Villa, Saverio Salzo, Luca Baldassarre, and Alessandro Verri. Accelerated and inexact forward-backward algorithms. *SIAM Journal on Optimization*, 23(3):1607–1633, 2013.

Jialei Wang, Weiran Wang, and Nathan Srebro. Memory and communication efficient distributed stochastic optimization with minibatch prox. In *Proceedings of The 30th Conference on Learning Theory*, 2017a.

Xiao Wang, Shiqian Ma, Donald Goldfarb, and Wei Liu. Stochastic quasi-newton methods for nonconvex stochastic optimization. *SIAM Journal on Optimization*, 27(2):927–956, 2017b.

Blake E Woodworth and Nati Srebro. Tight complexity bounds for optimizing composite objectives. In *Advances in Neural Information Processing Systems*, pages 3639–3647, 2016.

Stephen Wright and Jorge Nocedal. Numerical optimization. *Springer Science*, 35:67–68, 1999.

Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

Peng Xu, Jiyan Yang, Farbod Roosta-Khorasani, Christopher Ré, and Michael W Mahoney. Sub-sampled newton methods with non-uniform sampling. In *Advances in Neural Information Processing Systems*, pages 3000–3008, 2016.

Tianbao Yang, Rong Jin, Shenghuo Zhu, and Qihang Lin. On data preconditioning for regularized loss minimization. *Machine Learning*, 103(1):57–79, 2016.

Yuchen Zhang and Lin Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 353–361, 2015.

Yuchen Zhang, John Duchi, and Martin Wainwright. Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates. *The Journal of Machine Learning Research*, 16(1):3299–3340, 2015.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

Journal of Machine Learning Research 20 (2019) 1-58 Submitted 10/17; Published 2/19

DSCOVR: Randomized Primal-Dual Block Coordinate Algorithms for Asynchronous Distributed Optimization

Lin Xiao LIN.XIAO@MICROSOFT.COM
Microsoft Research AI
Redmond, WA 98052, USA

Adams Wei Yu WEIYU@CS.CMU.EDU
Machine Learning Department, Carnegie Mellon University
Pittsburgh, PA 15213, USA

Qihang Lin QIHANG-LIN@UIOWA.EDU
Tippie College of Business, The University of Iowa
Iowa City, IA 52245, USA

Weizhu Chen WZCHEN@MICROSOFT.COM
Microsoft AI and Research
Redmond, WA 98052, USA

Editor: Francis Bach

Abstract

Machine learning with big data often involves large optimization models. For distributed optimization over a cluster of machines, frequent communication and synchronization of all model parameters (optimization variables) can be very costly. A promising solution is to use parameter servers to store different subsets of the model parameters, and update them asynchronously at different machines using local datasets. In this paper, we focus on distributed optimization of large linear models with convex loss functions, and propose a family of randomized primal-dual block coordinate algorithms that are especially suitable for asynchronous distributed implementation with parameter servers. In particular, we work with the saddle-point formulation of such problems which allows simultaneous data and model partitioning, and exploit its structure by doubly stochastic coordinate optimization with variance reduction (DSCOVR). Compared with other first-order distributed algorithms, we show that DSCOVR may require less amount of overall computation and communication, and less or no synchronization. We discuss the implementation details of the DSCOVR algorithms, and present numerical experiments on an industrial distributed computing system.

1. Introduction

Algorithms and systems for distributed optimization are critical for solving large-scale machine learning problems, especially when the dataset cannot fit into the memory or storage of a single machine. In this paper, we consider distributed optimization problems of the form

$$\underset{w\in\mathbf{R}^d}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m} f_i(X_i w) + g(w), \tag{1}$$

where $X_i \in \mathbf{R}^{N_i\times d}$ is the local data stored at the ith machine, $f_i : \mathbf{R}^{N_i} \to \mathbf{R}$ is a convex cost function associated with the linear mapping $X_i w$, and $g(w)$ is a convex regularization function. In addition, we assume that g is separable, i.e., for some integer $n > 0$, we can write

$$g(w) = \sum_{k=1}^{n} g_k(w_k), \tag{2}$$

where $g_k : \mathbf{R}^{d_k} \to \mathbf{R}$, and $w_k \in \mathbf{R}^{d_k}$ for $k = 1, \ldots, n$ are non-overlapping subvectors of $w \in \mathbf{R}^d$ with $\sum_{k=1}^n d_k = d$ (they form a partition of w). Many popular regularization functions in machine learning are separable, for example, $g(w) = (\lambda/2)\|w\|_2^2$ or $g(w) = \lambda\|w\|_1$ for some $\lambda > 0$.

An important special case of (1) is distributed empirical risk minimization (ERM) of linear predictors. Let $(x_1, y_1), \ldots, (x_N, y_N)$ be N training examples, where each $x_j \in \mathbf{R}^d$ is a feature vector and $y_j \in \mathbf{R}$ is its label. The ERM problem is formulated as

$$\underset{w\in\mathbf{R}^d}{\text{minimize}} \quad \frac{1}{N}\sum_{j=1}^{N} \phi_j(x_j^T w) + g(w), \tag{3}$$

where each $\phi_j : \mathbf{R} \to \mathbf{R}$ is a loss function measuring the mismatch between the linear prediction $x_j^T w$ and the label y_j. Popular loss functions in machine learning include, e.g., for regression, the squared loss $\phi_j(t) = (1/2)(t - y_j)^2$, and for classification, the logistic loss $\phi_j(t) = \log(1 + \exp(-y_j t))$ where $y_j \in \{\pm 1\}$. In the distributed optimization setting, the N examples are divided into m subsets, each stored on a different machine. For $i = 1, \ldots, m$, let $\mathcal{I}_i$ denote the subset of $\{1, \ldots, N\}$ stored at machine i and let $N_i = |\mathcal{I}_i|$ (they satisfy $\sum_{i=1}^m N_i = N$). Then the ERM problem (3) can be written in the form of (1) by letting X_i consist of x_j^T with $j \in \mathcal{I}_i$ as its rows and defining $f_i : \mathbf{R}^{N_i} \to \mathbf{R}$ as

$$f_i(u_{\mathcal{I}_i}) = \frac{m}{N}\sum_{j\in\mathcal{I}_i} \phi_j(u_j), \tag{4}$$

where $u_{\mathcal{I}_i} \in \mathbf{R}^{N_i}$ is a subvector of $u \in \mathbf{R}^N$, consisting of u_j with $j \in \mathcal{I}_i$.

The nature of distributed algorithms and their convergence properties largely depend on the model of the communication network that connects the m computing machines. A popular setting in the literature is to model the communication network as a graph, and each node can only communicate (in one step) with their neighbors connected by an edge, either synchronously or asynchronously (e.g., Bertsekas and Tsitsiklis, 1989; Nedić and Ozdaglar, 2009). The convergence rates of distributed algorithms in this setting often depend on characteristics of the graph, such as its diameter and the eigenvalues of the graph Laplacian (e.g. Xiao and Boyd, 2006; Duchi et al., 2012; Nedić et al., 2017; Scaman et al., 2017). This is often called the *decentralized* setting.

Another model for the communication network is *centralized*, where all the machines participate synchronous, collective communication, e.g., broadcasting a vector to all m machines, or computing the sum of m vectors, each from a different machine (AllReduce). These collective communication protocols hide the underlying implementation details, which often involve operations on graphs. They are adopted by many popular distributed computing standards and packages, such as MPI (MPI Forum, 2012), MapReduce (Dean and Ghemawat, 2008) and Aparche Spark (Zaharia et al., 2016), and are widely used in machine learning practice (e.g., Lin et al., 2014; Meng et al., 2016). In particular, collective communications are very useful for addressing *data parallelism*, i.e., by allowing different machines to work in parallel to improve the same model $w \in \mathbf{R}^d$ using their local dataset. A disadvantage of collective communications is their synchronization cost: faster machines or machines with less computing tasks have to become idle while waiting for other machines to finish their tasks in order to participate a collective communication.

©2019 Lin Xiao, Adams Wei Yu, Qihang Lin and Weizhu Chen.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-608.html.

One effective approach for reducing synchronization cost is to exploit *model parallelism* (here "model" refers to $w \in \mathbf{R}^d$, including all optimization variables). The idea is to allow different machines work in parallel with different versions of the full model or different parts of a common model, with little or no synchronization. The model partitioning approach can be very effective for solving problems with large models (large dimension d). Dedicated *parameter servers* can be set up to store and maintain different subsets of the model parameters, such as the w_k's in (2), and be responsible for coordinating their updates at different workers (Li et al., 2014; Xing et al., 2015). This requires flexible point-to-point communication.

In this paper, we develop a family of randomized algorithms that exploit *simultaneous* data and model parallelism. Correspondingly, we adopt a centralized communication model that support both synchronous collective communication and asynchronous point-to-point communication. In particular, it allows any pair of machines to send/receive a message in a single step, and multiple point-to-point communications may happen in parallel in an event-driven, asynchronous manner. Such a communication model is well supported by the MPI standard. To evaluate the performance of distributed algorithms in this setting, we consider the following three measures.

- *Computation complexity:* total amount of computation, measured by the number of passes over all datasets X_i for $i = 1, \ldots, m$, which can happen in parallel on different machines.
- *Communication complexity:* the total amount of communication required, measured by the equivalent number of vectors in $\mathbf{R}^d$ sent or received across all machines.
- *Synchronous communication:* measured by the total number of vectors in $\mathbf{R}^d$ that requires synchronous collective communication involving all m machines. We single it out from the overall communication complexity as a (partial) measure of the synchronization cost.

In Section 2, we introduce the framework of our randomized algorithms, **D**oubly **S**tochastic **C**oordinate **O**ptimization with **V**ariance **R**eduction (DSCOVR), and summarize our theoretical results on the three measures achieved by DSCOVR. Compared with other first-order methods for distributed optimization, we show that DSCOVR may require less amount of overall computation and communication, and less or no synchronization. Then we present the details of several DSCOVR variants and their convergence analysis in Sections 3-6. We discuss the implementation of different DSCOVR algorithms in Section 7, and present results of our numerical experiments in Section 8.

2. The DSCOVR Framework and Main Results

First, we derive a saddle-point formulation of the convex optimization problem (1). Let f_i^* be the convex conjugate of f_i, i.e., $f_i^*(\alpha_i) = \sup_{u_i \in \mathbf{R}^{N_i}} \left\{\alpha_i^T u_i - f_i(u_i)\right\}$, and define

$$L(w, \alpha) \equiv \frac{1}{m}\sum_{i=1}^m \alpha_i^T X_i w - \frac{1}{m}\sum_{i=1}^m f_i^*(\alpha_i) + g(w), \tag{5}$$

where $\alpha = [\alpha_1; \ldots; \alpha_m] \in \mathbf{R}^N$. Since both the f_i's and g are convex, $L(w, \alpha)$ is convex in w and concave in α. We also define a pair of *primal* and *dual* functions:

$$P(w) = \max_{\alpha \in \mathbf{R}^N} L(w, \alpha) = \frac{1}{m}\sum_{i=1}^m f_i(X_i w) + g(w), \tag{6}$$

$$D(\alpha) = \min_{w \in \mathbf{R}^d} L(w, \alpha) = -\frac{1}{m}\sum_{i=1}^m f_i^*(\alpha_i) - g^*\Big(-\frac{1}{m}\sum_{i=1}^m (X_i)^T \alpha_i\Big), \tag{7}$$

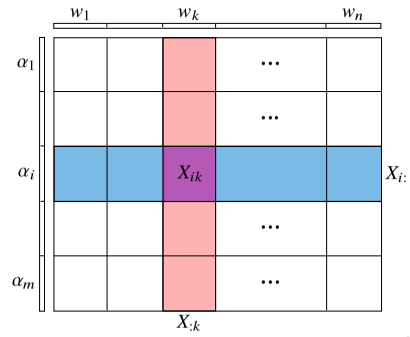


Figure 1: Partition of primal variable w, dual variable α, and the data matrix X.

where $P(w)$ is exactly the objective function in (1)[1] and g^* is the convex conjugate of g. We assume that L has a saddle point $(w^\star, \alpha^\star)$, that is,

$$L(w^\star, \alpha) \le L(w^\star, \alpha^\star) \le L(w, \alpha^\star), \qquad \forall (w, \alpha) \in \mathbf{R}^d \times \mathbf{R}^N.$$

In this case, we have $w^\star = \arg\min\ P(w)$ and $\alpha^\star = \arg\min\ D(\alpha)$, and $P(w^\star) = D(\alpha^\star)$.

The DSCOVR framework is based on solving the convex-concave saddle-point problem

$$\min_{w \in \mathbf{R}^d}\ \max_{\alpha \in \mathbf{R}^N}\ L(w, \alpha). \tag{8}$$

Since we assume that g has a separable structure as in (2), we rewrite the saddle-point problem as

$$\min_{w \in \mathbf{R}^d}\ \max_{\alpha \in \mathbf{R}^N}\ \left\{\frac{1}{m}\sum_{i=1}^m\sum_{k=1}^n \alpha_i^T X_{ik} w_k - \frac{1}{m}\sum_{i=1}^m f_i^*(\alpha_i) + \sum_{k=1}^n g_k(w_k)\right\}, \tag{9}$$

where $X_{ik} \in \mathbf{R}^{N_i \times d_k}$ for $k = 1, \ldots, n$ are column partitions of X_i. For convenience, we define the following notations. First, let $X = [X_1; \ldots; X_m] \in \mathbf{R}^{N \times d}$ be the overall data matrix, by stacking the X_i's vertically. Conforming to the separation of g, we also partition X into block columns $X_{:k} \in \mathbf{R}^{N \times d_k}$ for $k = 1, \ldots, n$, where each $X_{:k} = [X_{1k}; \ldots; X_{mk}]$ (stacked vertically). For consistency, we also use $X_{i:}$ to denote X_i from now on. See Figure 1 for an illustration.

We exploit the doubly separable structure in (9) by a doubly stochastic coordinate update algorithm outlined in Algorithm 1. Let $p = \{p_1, \ldots, p_m\}$ and $q = \{q_1, \ldots, q_n\}$ be two probability distributions. During each iteration t, we randomly pick an index $j \in \{1, \ldots, m\}$ with probability p_j, and independently pick an index $l \in \{1, \ldots, n\}$ with probability q_l. Then we compute two vectors $u_j^{(t+1)} \in \mathbf{R}^{N_j}$ and $v_l^{(t+1)} \in \mathbf{R}^{d_l}$ (details to be discussed later), and use them to update the block coordinates α_j and w_l while leaving other block coordinates unchanged. The update formulas in (10) and (11) use the proximal mappings of the (scaled) functions f_j^* and g_l respectively. We

1. More technically, we need to assume that each f_i is convex and lower semi-continuous so that $f_i^{**} = f_i$ (see, e.g., Rockafellar, 1970, Section 12). It automatically holds if f_i is convex and differentiable, which we will assume later.

Algorithm 1 DSCOVR framework

input: initial points $w^{(0)}, \alpha^{(0)}$, and step sizes σ_i for $i = 1, \ldots, m$ and τ_k for $k = 1, \ldots, n$.

1: **for** $t = 0, 1, 2, \ldots,$ **do**

2: pick $j \in \{1, \ldots, m\}$ and $l \in \{1, \ldots, n\}$ randomly with distributions p and q respectively.

3: compute *variance-reduced* stochastic gradients $u_j^{(t+1)}$ and $v_l^{(t+1)}$.

4: update primal and dual block coordinates:

$$\alpha_i^{(t+1)} = \begin{cases} \mathbf{prox}_{\sigma_j f_j^*}(\alpha_j^{(t)} + \sigma_j u_j^{(t+1)}) & \text{if } i = j, \\ \alpha_i^{(t)}, & \text{if } i \neq j, \end{cases} \tag{10}$$

$$w_k^{(t+1)} = \begin{cases} \mathbf{prox}_{\tau_l g_l}(w_l^{(t)} - \tau_l v_l^{(t+1)}) & \text{if } k = l, \\ w_k^{(t)}, & \text{if } k \neq l. \end{cases} \tag{11}$$

5: **end for**

recall that the proximal mapping for any convex function $\phi : \mathbf{R}^d \to \mathbf{R} \cup \{\infty\}$ is defined as

$$\mathbf{prox}_\phi(v) \triangleq \arg\min_{u \in \mathbf{R}^d} \left\{ \phi(u) + \frac{1}{2}\|u - v\|^2 \right\}.$$

There are several different ways to compute the vectors $u_j^{(t+1)}$ and $v_l^{(t+1)}$ in Step 3 of Algorithm 1. They should be the partial gradients or stochastic gradients of the bilinear coupling term in $L(w, \alpha)$ with respect to α_j and w_l respectively. Let

$$K(w, \alpha) = \alpha^T X w = \sum_{i=1}^m \sum_{k=1}^n \alpha_i^T X_{ik} w_k,$$

which is the bilinear term in $L(w, \alpha)$ without the factor $1/m$. We can use the following partial gradients in Step 3:

$$\begin{aligned} \tilde{u}_j^{(t+1)} &= \frac{\partial K(w^{(t)}, \alpha^{(t)})}{\partial \alpha_j} = \sum_{k=1}^n X_{jk} w_k^{(t)}, \\ \tilde{v}_l^{(t+1)} &= \frac{1}{m} \frac{\partial K(w^{(t)}, \alpha^{(t)})}{\partial w_l} = \frac{1}{m} \sum_{i=1}^m (X_{il})^T \alpha_i^{(t)}. \end{aligned} \tag{12}$$

We note that the factor $1/m$ does not appear in the first equation because it multiplies both $K(w, \alpha)$ and $f_j^*(\alpha_j)$ in (9) and hence does not appear in updating α_j. Another choice is to use

$$\begin{aligned} u_j^{(t+1)} &= \frac{1}{q_l} X_{jl} w_l^{(t)}, \\ v_l^{(t+1)} &= \frac{1}{p_j} \frac{1}{m} (X_{jl})^T \alpha_j^{(t)}, \end{aligned} \tag{13}$$

which are unbiased stochastic partial gradients, because

$$\begin{aligned} \mathbf{E}_l[u_j^{(t+1)}] &= \sum_{k=1}^n q_k \frac{1}{q_k} X_{jk} w_k^{(t)} = \sum_{k=1}^n X_{jk} w_k^{(t)} = \tilde{u}_j^{(t+1)}, \\ \mathbf{E}_j[v_l^{(t+1)}] &= \sum_{i=1}^m p_i \frac{1}{p_i} \frac{1}{m} (X_{il})^T \alpha_i^{(t)} = \frac{1}{m} \sum_{i=1}^m (X_{il})^T \alpha_i^{(t)} = \tilde{v}_l^{(t+1)}, \end{aligned}$$

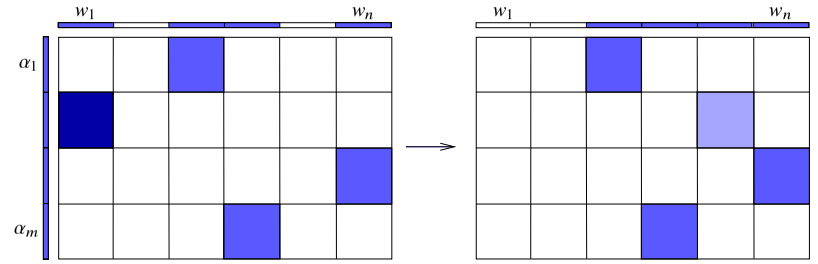


Figure 2: Simultaneous data and model parallelism. At any given time, each machine is busy updating one parameter block and its own dual variable. Whenever some machine is done, it is assigned to work on a random block that is not being updated.

where $\mathbf{E}_j$ and $\mathbf{E}_l$ are expectations with respect to the random indices j and l respectively.

It can be shown that, Algorithm 1 converges to a saddle point of $L(w, \alpha)$ with either choice (12) or (13) in Step 3, and with suitable *step sizes* σ_i and τ_k. It is expected that using the stochastic gradients in (13) leads to a slower convergence rate than applying (12). However, using (13) has the advantage of much less computation during each iteration. Specifically, it employs only one block matrix-vector multiplication for both updates, instead of n and m block multiplications done in (12).

More importantly, the choice in (13) is suitable for parallel and distributed computing. To see this, let $(j^{(t)}, l^{(t)})$ denote the pair of random indices drawn at iteration t (we omit the superscript (t) to simplify notation whenever there is no confusion from the context). Suppose for a sequence of consecutive iterations $t, \ldots, t+s$, there is no common index among $j^{(t)}, \ldots, j^{(t+s)}$, nor among $l^{(t)}, \ldots, l^{(t+s)}$, then these $s+1$ iterations can be done in parallel and they produce the same updates as being done sequentially. Suppose there are $s+1$ processors or machines, then each can carry out one iteration, which includes the updates in (13) as well as (10) and (11). These $s+1$ iterations are independent of each other, and in fact can be done in any order, because each only involve one primal block $w_{l^{(t)}}$ and one dual block $\alpha_{j^{(t)}}$, for both input and output (variables on the right and left sides of the assignments respectively). In contrast, the input for the updates in (12) depend on all primal and dual blocks at the previous iteration, thus cannot be done in parallel.

In practice, suppose we have m machines for solving problem (9), and each holds the data matrix $X_{i:}$ in memory and maintains the dual block α_i, for $i = 1, \ldots, m$. We assume that the number of model partitions n is larger than m, and the n model blocks $\{w_1, \ldots, w_n\}$ are stored at one or more parameter servers. In the beginning, we can randomly pick m model blocks (sampling without replacement) from $\{w_1, \ldots, w_n\}$, and assign each machine to update one of them. If machine i is assigned to update block k, then both α_i and w_k are updated, using only the matrix X_{ik}; moreover, it needs to communicate only the block w_k with the parameter server that are responsible to maintain it. Whenever one machine finishes its update, a scheduler can randomly pick another parameter block that is not currently updated by other machines, and assign it to the free machine. Therefore all machines can work in parallel, in an asynchronous, event-driven manner. Here an event is the completion of a block update at any machine, as illustrated in Figure 2. We will discuss the implementation details in Section 7.

The idea of using doubly stochastic updates for distributed optimization in not new. It has been studied by Yun et al. (2014) for solving the matrix completion problem, and by Matsushima et al. (2014) for solving the saddle-point formulation of the ERM problem. Despite their nice features for parallelization, these algorithms inherit the $O(1/\sqrt{t})$ (or $O(1/t)$ with strong convexity) sublinear convergence rate of the classical stochastic gradient method. They translate into high communication and computation cost for distributed optimization. In this paper, we propose new variants of doubly stochastic update algorithms by using *variance-reduced* stochastic gradients (Step 3 of Algorithm 1). More specifically, we borrow the variance-reduction techniques from SVRG (Johnson and Zhang, 2013) and SAGA (Defazio et al., 2014) to develop the DSCOVR algorithms, which enjoy fast linear rates of convergence. In the rest of this section, we summarize our theoretical results characterizing the three measures for DSCOVR: computation complexity, communication complexity, and synchronization cost. We compare them with distributed implementation of batch first-order algorithms.

2.1. Summary of Main Results

Throughout this paper, we use $\|\cdot\|$ to denote the standard Euclidean norm for vectors. For matrices, $\|\cdot\|$ denotes the operator (spectral) norm and $\|\cdot\|_F$ denotes the Frobenius norm. We make the following assumption regarding the optimization problem (1).

Assumption 1 *Each f_i is convex and differentiable, and its gradient is $(1/\gamma_i)$-Lipschitz continuous, i.e.,*

$$\|\nabla f_i(u) - \nabla f_i(v)\| \le \frac{1}{\gamma_i}\|u - v\|, \quad \forall\, u, v \in \mathbf{R}^{N_i}, \quad i = 1, \ldots, m. \tag{14}$$

In addition, the regularization function g is λ-strongly convex, i.e.,

$$g(w') \ge g(w) + \xi^T(w' - w) + \frac{\lambda}{2}\|w' - w\|^2, \quad \forall\, \xi \in \partial g(w), \quad w', w \in \mathbf{R}^d.$$

Under Assumption 1, each f_i^* is γ_i-strongly convex (see, e.g., Hiriart-Urruty and Lemaréchal, 2001, Theorem 4.2.2), and $L(w, \alpha)$ defined in (5) has a unique saddle point $(w^\star, \alpha^\star)$.

The condition (14) is often referred to as f_i being $1/\gamma_i$-smooth. To simplify discussion, here we assume $\gamma_i = \gamma$ for $i = 1, \ldots, m$. Under these assumptions, each composite function $f_i(X_i w)$ has a smoothness parameter $\|X_i\|^2/\gamma$ (upper bound on the largest eigenvalue of its Hessian). Their average $(1/m)\sum_{i=1}^m f_i(X_i w)$ has a smooth parameter $\|X\|^2/(m\gamma)$, which no larger than the average of the individual smooth parameters $(1/m)\sum_{i=1}^m \|X_i\|^2/\gamma$. We define a condition number for problem (1) as the ratio between this smooth parameter and the convexity parameter λ of g:

$$\kappa_{\rm bat} = \frac{\|X\|^2}{m\lambda\gamma} \le \frac{1}{m}\sum_{i=1}^m \frac{\|X_{i:}\|^2}{\lambda\gamma} \le \frac{\|X\|_{\max}^2}{\lambda\gamma}, \tag{15}$$

where $\|X\|_{\max} = \max_i\{\|X_{i:}\|\}$. This condition number is a key factor to characterize the iteration complexity of *batch* first-order methods for solving problem (1), i.e., minimizing $P(w)$. Specifically, to find a w such that $P(w) - P(w^\star) \le \epsilon$, the proximal gradient method requires $O\left((1 + \kappa_{\rm bat})\log(1/\epsilon)\right)$ iterations, and their accelerated variants require $O\left((1 + \sqrt{\kappa_{\rm bat}})\log(1/\epsilon)\right)$ iterations (e.g., Nesterov, 2004; Beck and Teboulle, 2009; Nesterov, 2013). Primal-dual first order methods for solving the saddle-point problem (8) share the same complexity (Chambolle and Pock, 2011, 2015).

Algorithms	Computation complexity (number of passes over data)	Communication complexity (number of vectors in $\mathbf{R}^d$)
batch first-order methods	$(1 + \kappa_{\rm bat})\log(1/\epsilon)$	$m(1 + \kappa_{\rm bat})\log(1/\epsilon)$
DSCOVR	$(1 + \kappa_{\rm rand}/m)\log(1/\epsilon)$	$(m + \kappa_{\rm rand})\log(1/\epsilon)$
accelerated batch first-order methods	$\left(1 + \sqrt{\kappa_{\rm bat}}\right)\log(1/\epsilon)$	$m\left(1 + \sqrt{\kappa_{\rm bat}}\right)\log(1/\epsilon)$
accelerated DSCOVR	$(1 + \sqrt{\kappa_{\rm rand}/m})\log(1/\epsilon)$	$\left(m + \sqrt{m\cdot\kappa_{\rm rand}}\right)\log(1/\epsilon)$

Table 1: Computation and communication complexities of batch first-order methods and DSCOVR (for both SVRG and SAGA variants). We omit the $O(\cdot)$ notation in all entries and an extra $\log(1 + \kappa_{\rm rand}/m)$ factor for accelerated DSCOVR algorithms.

A fundamental baseline for evaluating any distributed optimization algorithms is the distributed implementation of batch first-order methods. Let's consider solving problem (1) using the proximal gradient method. During every iteration t, each machine receives a copy of $w^{(t)} \in \mathbf{R}^d$ from a master machine (through Broadcast), and computes the local gradient $z_i^{(t)} = X_i^T \nabla f_i(X_i w^{(t)}) \in \mathbf{R}^d$. Then a collective communication is invoked to compute the batch gradient $z^{(t)} = (1/m)\sum_{i=1}^m z_i^{(t)}$ at the master (Reduce). The master then takes a proximal gradient step, using $z^{(t)}$ and the proximal mapping of g, to compute the next iterate $w^{(t+1)}$ and broadcast it to every machine for the next iteration. We can also use the AllReduce operation in MPI to obtain $z^{(t)}$ at each machine without a master. In either case, the total number of passes over the data is twice the number of iterations (due to matrix-vector multiplications using both X_i and X_i^T), and the number of vectors in $\mathbf{R}^d$ sent/received across all machines is $2m$ times the number of iterations (see Table 1). Moreover, all communications are collective and synchronous.

Since DSCOVR is a family of randomized algorithms for solving the saddle-point problem (8), we would like to find (w, α) such that $\|w^{(t)} - w^\star\|^2 + (1/m)\|\alpha^{(t)} - \alpha^\star\|^2 \le \epsilon$ holds in expectation and with high probability. We list the communication and computation complexities of DSCOVR in Table 1, comparing them with batch first-order methods. Similar guarantees also hold for reducing the duality gap $P(w^{(t)}) - D(\alpha^{(t)})$, where P and D are defined in (6) and (7) respectively.

The key quantity characterizing the complexities of DSCOVR is the condition number $\kappa_{\rm rand}$, which can be defined in several different ways. If we pick the data block i and model block k with uniform distribution, i.e., $p_i = 1/m$ for $i = 1, \ldots, m$ and $q_k = 1/n$ for $k = 1, \ldots, n$, then

$$\kappa_{\rm rand} = \frac{n\|X\|_{m\times n}^2}{\lambda\gamma}, \qquad \text{where} \qquad \|X\|_{m\times n} = \max_{i,k}\|X_{ik}\|. \tag{16}$$

Comparing the definition of $\kappa_{\rm bat}$ in (15), we have $\kappa_{\rm bat} \le \kappa_{\rm rand}$ because

$$\frac{1}{m}\|X\|^2 \le \frac{1}{m}\sum_{i=1}^m \|X_i\|^2 \le \frac{1}{m}\sum_{i=1}^m\sum_{k=1}^n \|X_{ik}\|^2 \le n\|X\|_{m\times n}^2.$$

With $X_{i:} = [X_{i1}\cdots X_{im}] \in \mathbf{R}^{N_i\times d}$ and $X_{:k} = [X_{1k}; \ldots; X_{mk}] \in \mathbf{R}^{N\times d_k}$, we can also define

$$\kappa'_{\rm rand} = \frac{\|X\|_{\max,F}^2}{\lambda\gamma}, \qquad \text{where} \qquad \|X\|_{\max,F} = \max_{i,k}\{\|X_{i:}\|_F,\ \|X_{:k}\|_F\}. \tag{17}$$

Algorithms	Synchronous Communication (number of vectors in $\mathbf{R}^d$)	Asynchronous Communication (equiv. number of vectors in $\mathbf{R}^d$)
DSCOVR-SVRG	$m \log(1/\epsilon)$	$\kappa_{\text{rand}} \log(1/\epsilon)$
DSCOVR-SAGA	m	$(m + \kappa_{\text{rand}}) \log(1/\epsilon)$
accelerated DSCOVR-SVRG	$m \log(1/\epsilon)$	$\left(1 + \sqrt{m \cdot \kappa_{\text{rand}}}\right) \log(1/\epsilon)$
accelerated DSCOVR-SAGA	m	$\left(1 + \sqrt{m \cdot \kappa_{\text{rand}}}\right) \log(1/\epsilon)$

Table 2: Breakdown of communication complexities into synchronous and asynchronous communications for two different types of DSCOVR algorithms. We omit the $O(\cdot)$ notation and an extra $\log(1 + \kappa_{\text{rand}}/m)$ factor for accelerated DSCOVR algorithms.

In this case, we also have $\kappa_{\text{bat}} \leq \kappa'_{\text{rand}}$ because $\|X\|_{\max} \leq \|X\|_{\max,F}$. Finally, if we pick the pair (i, k) with non-uniform distribution $p_i = \|X_{i:}\|_F^2 / \|X\|_F^2$ and $q_k = \|X_{:k}\|_F^2 / \|X\|_F^2$, then we can define

$$\kappa''_{\text{rand}} = \frac{\|X\|_F^2}{m\lambda\gamma}. \tag{18}$$

Again we have $\kappa_{\text{bat}} \leq \kappa''_{\text{rand}}$ because $\|X\| \leq \|X\|_F$. We may replace κ_{rand} in Tables 1 and 2 by either κ'_{rand} or κ''_{rand}, depending on the probability distributions p and q and different proof techniques.

From Table 1, we observe similar type of speed-ups in computation complexity, as obtained by variance reduction techniques over the batch first-order algorithms for convex optimization (e.g., Le Roux et al., 2012; Johnson and Zhang, 2013; Defazio et al., 2014; Xiao and Zhang, 2014; Lan and Zhou, 2018; Allen-Zhu, 2017), as well as for convex-concave saddle-point problems (Zhang and Xiao, 2017; Balamurugan and Bach, 2016). Basically, DSCOVR algorithms have potential improvement over batch first-order methods by a factor of m (for non-accelerated algorithms) or $\sqrt{m}$ (for accelerated algorithms), but with a worse condition number. In the worst case, the ratio between κ_{rand} and κ_{bat} may be of order m or larger, thus canceling the potential improvements.

More interestingly, DSCOVR also has similar improvements in terms of communication complexity over batch first-order methods. In Table 2, we decompose the communication complexity of DSCOVR into synchronous and asynchronous communication. The decomposition turns out to be different depending on the variance reduction techniques employed: SVRG (Johnson and Zhang, 2013) versus SAGA (Defazio et al., 2014). We note that DSCOVR-SAGA essentially requires only asynchronous communication, because the synchronous communication of m vectors are only necessary for initialization with non-zero starting point.

The comparisons in Table 1 and 2 give us good understanding of the complexities of different algorithms. However, these complexities are not accurate measures of their performance in practice. For example, collective communication of m vectors in $\mathbf{R}^d$ can often be done in parallel over a spanning tree of the underlying communication network, thus only cost $\log(m)$ times (instead of m times) compared with sending only one vector. Also, for point-to-point communication, sending one vector in $\mathbf{R}^d$ altogether can be much faster than sending n smaller vectors of total length d separately. A fair comparison in term of wall-clock time on a real-world distributed computing system requires customized, efficient implementation of different algorithms. We will shed some light on timing comparisons with numerical experiments in Section 8.

2.2. Related Work

There is an extensive literature on distributed optimization. Many algorithms developed for machine learning adopt the centralized communication setting, due to the wide availability of supporting standards and platforms such as MPI, MapReduce and Spark (as discussed in the introduction). They include parallel implementations of the batch first-order and second-order methods (e.g., Lin et al., 2014; Chen et al., 2014; Lee et al., 2017a), ADMM (Boyd et al., 2011), and distributed dual coordinate ascent (Yang, 2013; Jaggi et al., 2014; Ma et al., 2015).

For minimizing the average function $(1/m) \sum_{i=1}^m f_i(w)$, in the centralized setting and with only first-order oracles (i.e., gradients of f_i's or their conjugates), it has been shown that distributed implementation of accelerated gradient methods achieves the optimal convergence rate and communication complexity (Arjevani and Shamir, 2015; Scaman et al., 2017). The problem (1) we consider has the extra structure of composition with a linear transformation by the local data, which allows us to exploit simultaneous data and model parallelism using randomized algorithms and obtain improved communication and computation complexity.

Most work on asynchronous distributed algorithms exploit model parallelism in order to reduce the synchronization cost, especially in the setting with parameter servers (e.g., Li et al., 2014; Xing et al., 2015; Aytekin et al., 2016). Besides, delay caused by the asynchrony can be incorporated to the step size to gain practical improvement on convergence (e.g., Agarwal and Duchi, 2011; McMahan and Streeter, 2014; Sra et al., 2016), though the theoretical sublinear rates remain. There are also many recent work on asynchronous parallel stochastic gradient and coordinate-descent algorithms for convex optimization (e.g., Recht et al., 2011; Liu et al., 2014; Shi et al., 2015; Reddi et al., 2015; Richtárik and Takáč, 2016; Peng et al., 2016). When the workloads or computing power of different machines or processors are nonuniform, they may significantly increase *iteration efficiency* (number of iterations done in unit time), but often at the cost of requiring more iterations than their synchronous counterparts (due to delays and stale updates). So there is a subtle balance between iteration efficiency and iteration complexity (e.g., Hannah and Yin, 2017). Our discussions in Section 2.1 show that DSCOVR is capable of improving both aspects.

For solving bilinear saddle-point problems with a finite-sum structure, Zhang and Xiao (2017) proposed a randomized algorithm that works with dual coordinate update but full primal update. Yu et al. (2015) proposed a doubly stochastic algorithm that works with both primal and dual coordinate updates based on equation (12). Both of them achieved accelerated linear convergence rates, but neither can be readily applied to distributed computing. In addition, Balamurugan and Bach (2016) proposed stochastic variance-reduction methods (also based on SVRG and SAGA) for solving more general convex-concave saddle point problems. For the special case with bilinear coupling, they obtained similar computation complexity as DSCOVR. However, their methods require full model updates at each iteration (even though working with only one sub-block of data), thus are not suitable for distributed computing.

With additional assumptions and structure, such as similarity between the local cost functions at different machines or using second-order information, it is possible to obtain better communication complexity for distributed optimization; see, e.g., Shamir et al. (2014); Zhang and Xiao (2015); Reddi et al. (2016). However, these algorithms rely on much more computation at each machine for solving a local sub-problem at each iteration. With additional memory and preprocessing at each machine, Lee et al. (2017b) showed that SVRG can be adapted for distributed optimization to obtain low communication complexity.

Algorithm 2 DSCOVR-SVRG

input: initial points $\bar{w}^{(0)}$, $\bar{\alpha}^{(0)}$, number of stages S and number of iterations per stage M.

for $s = 0, 1, 2, \ldots, S-1$ **do**

  $\bar{u}^{(s)} = X\bar{w}^{(s)}$ and $\bar{v}^{(s)} = \frac{1}{m}X^T\bar{\alpha}^{(s)}$

  $w^{(0)} = \bar{w}^{(s)}$ and $\alpha^{(0)} = \bar{\alpha}^{(s)}$

  for $t = 0, 1, 2, \ldots, M-1$ **do**

    pick $j \in \{1, \ldots, m\}$ and $l \in \{1, \ldots, n\}$ randomly with distributions p and q respectively.

    compute variance-reduced stochastic gradients:

$$u_j^{(t+1)} = \bar{u}_j^{(s)} + \frac{1}{q_l}X_{jl}(w_l^{(t)} - \bar{w}_l^{(s)}), \tag{19}$$

$$v_l^{(t+1)} = \bar{v}_l^{(s)} + \frac{1}{p_j}\frac{1}{m}(X_{jl})^T(\alpha_j^{(t)} - \bar{\alpha}_j^{(s)}). \tag{20}$$

    update primal and dual block coordinates:

$$\alpha_i^{(t+1)} = \begin{cases} \mathbf{prox}_{\sigma_j f_j^*}(\alpha_j^{(t)} + \sigma_j u_j^{(t+1)}) & \text{if } i = j, \\ \alpha_i^{(t)}, & \text{if } i \neq j, \end{cases}$$

$$w_k^{(t+1)} = \begin{cases} \mathbf{prox}_{\tau_l g_l}(w_l^{(t)} - \tau_l v_l^{(t+1)}) & \text{if } k = l, \\ w_k^{(t)}, & \text{if } k \neq l. \end{cases}$$

  end for

  $\bar{w}^{(s+1)} = w^{(M)}$ and $\bar{\alpha}^{(s+1)} = \alpha^{(M)}$.

end for

output: $\bar{w}^{(S)}$ and $\bar{\alpha}^{(S)}$.

3. The DSCOVR-SVRG Algorithm

From this section to Section 6, we present several realizations of DSCOVR using different variance reduction techniques and acceleration schemes, and analyze their convergence properties. These algorithms are presented and analyzed as sequential randomized algorithms. We will discuss how to implement them for asynchronous distributed computing in Section 7.

Algorithm 2 is a DSCOVR algorithm that uses the technique of SVRG (Johnson and Zhang, 2013) for variance reduction. The iterations are divided into stages and each stage has a inner loop. Each stage is initialized by a pair of vectors $\bar{w}^{(s)} \in \mathbf{R}^d$ and $\bar{\alpha}^{(s)} \in \mathbf{R}^N$, which come from either initialization (if $s = 0$) or the last iterate of the previous stage (if $s > 0$). At the beginning of each stage, we compute the batch gradients

$$\bar{u}^{(s)} = \frac{\partial}{\partial\bar{\alpha}^{(s)}}\left((\bar{\alpha}^{(s)})^T X\bar{w}^{(s)}\right) = X\bar{w}^{(s)}, \qquad \bar{v}^{(s)} = \frac{\partial}{\partial\bar{w}^{(s)}}\left(\frac{1}{m}(\bar{\alpha}^{(s)})^T X\bar{w}^{(s)}\right) = \frac{1}{m}X^T\bar{\alpha}^{(s)}.$$

The vectors $\bar{u}^{(s)}$ and $\bar{v}^{(s)}$ share the same partitions as $\alpha^{(t)}$ and $w^{(t)}$, respectively. Inside each stage s, the variance-reduced stochastic gradients are computed in (19) and (20). It is easy to check that they

are unbiased. More specifically, taking expectation of $u_j^{(t+1)}$ with respect to the random index l gives

$$\mathbf{E}_l[u_j^{(t+1)}] = \bar{u}_j^{(s)} + \sum_{k=1}^n q_k\frac{1}{q_k}X_{jk}(w_k^{(t)} - \bar{w}_k^{(s)}) = \bar{u}_j^{(s)} + X_{j:}w^{(t)} - X_{j:}\bar{w}^{(s)} = X_{j:}w^{(t)},$$

and taking expectation of $v_l^{(t+1)}$ with respect to the random index j gives

$$\mathbf{E}_j[v_l^{(t+1)}] = \bar{v}_l^{(s)} + \sum_{i=1}^m p_i\frac{1}{p_i}\frac{1}{m}(X_{il})^T(\alpha_i^{(t)} - \bar{\alpha}_i^{(s)}) = \bar{v}_l^{(s)} + \frac{1}{m}(X_{:l})^T\left(\alpha^{(t)} - \bar{\alpha}^{(s)}\right) = \frac{1}{m}(X_{:l})^T\alpha^{(t)}.$$

In order to measure the distance of any pair of primal and dual variables to the saddle point, we define a weighted squared Euclidean norm on $\mathbf{R}^{d+N}$. Specifically, for any pair (w, α) where $w \in \mathbf{R}^d$ and $\alpha = [\alpha_1, \ldots, \alpha_m] \in \mathbf{R}^N$ with $\alpha_i \in \mathbf{R}^{N_i}$, we define

$$\Omega(w, \alpha) = \lambda\|w\|^2 + \frac{1}{m}\sum_{i=1}^m \gamma_i\|\alpha_i\|^2. \tag{21}$$

If $\gamma_i = \gamma$ for all $i = 1, \ldots, m$, then $\Omega(w, \alpha) = \lambda\|w\|^2 + \frac{\gamma}{m}\|\alpha\|^2$. We have the following theorem concerning the convergence rate of Algorithm 2.

Theorem 1 *Suppose Assumption 1 holds, and let $(w^\star, \alpha^\star)$ be the unique saddle point of $L(w, \alpha)$. Let Γ be a constant that satisfies*

$$\Gamma \geq \max_{i,k}\left\{\frac{1}{p_i}\left(1 + \frac{9\|X_{ik}\|^2}{2q_k\lambda\gamma_i}\right), \frac{1}{q_k}\left(1 + \frac{9n\|X_{ik}\|^2}{2mp_i\lambda\gamma_i}\right)\right\}. \tag{22}$$

In Algorithm 2, if we choose the step sizes as

$$\sigma_i = \frac{1}{2\gamma_i(p_i\Gamma - 1)}, \qquad i = 1, \ldots, m, \tag{23}$$

$$\tau_k = \frac{1}{2\lambda(q_k\Gamma - 1)}, \qquad k = 1, \ldots, n, \tag{24}$$

and the number of iterations during each stage satisfies $M \geq \log(3)\Gamma$, then for any $s > 0$,

$$\mathbf{E}\left[\Omega(\bar{w}^{(s)} - w^\star, \bar{\alpha}^{(s)} - \alpha^\star)\right] \leq \left(\frac{2}{3}\right)^s \Omega(\bar{w}^{(0)} - w^\star, \bar{\alpha}^{(0)} - \alpha^\star). \tag{25}$$

The proof of Theorem 1 is given in Appendix A. Here we discuss how to choose the parameter Γ to satisfy (22). For simplicity, we assume $\gamma_i = \gamma$ for all $i = 1, \ldots, m$.

- If we let $\|X\|_{m\times n} = \max_{i,k}\{\|X_{ik}\|\}$ and sample with the uniform distribution across both rows and columns, i.e., $p_i = 1/m$ for $i = 1, \ldots, m$ and $q_k = 1/n$ for $k = 1, \ldots, n$, then we can set

$$\Gamma = \max\{m, n\}\left(1 + \frac{9n\|X\|_{m\times n}^2}{2\lambda\gamma}\right) = \max\{m, n\}\left(1 + \frac{9}{2}\kappa_{\text{rand}}\right),$$

  where $\kappa_{\text{rand}} = n\|X\|_{m\times n}^2/(\lambda\gamma)$ as defined in (16).

- An alternative condition for Γ to satisfy is (shown in Section A.1 in the Appendix)

$$\Gamma \geq \max_{i,k} \left\{ \frac{1}{p_i} \left(1 + \frac{9\|X_{:k}\|_F^2}{2q_k m\lambda\gamma_i} \right), \ \frac{1}{q_k} \left(1 + \frac{9\|X_{i:}\|_F^2}{2p_i m\lambda\gamma_i} \right) \right\}. \tag{26}$$

  Again using uniform sampling, we can set

$$\Gamma = \max\{m,n\} \left(1 + \frac{9\|X\|_{\max,F}^2}{2\lambda\gamma} \right) = \max\{m,n\} \left(1 + \frac{9}{2}\kappa'_{\text{rand}} \right),$$

  where $\|X\|_{\max,F} = \max_{i,k}\{\|X_{i:}\|_F, \|X_{:k}\|_F\}$ and $\kappa'_{\text{rand}} = \|X\|_{\max,F}^2/(\lambda\gamma)$ as defined in (17).

- Using the condition (26), if we choose the probabilities to be proportional to the squared Frobenius norms of the data partitions, i.e.,

$$p_i = \frac{\|X_{i:}\|_F^2}{\|X\|_F^2}, \qquad q_k = \frac{\|X_{:k}\|_F^2}{\|X\|_F^2}, \tag{27}$$

  then we can choose

$$\Gamma = \frac{1}{\min_{i,k}\{p_i, q_k\}} \left(1 + \frac{9\|X\|_F^2}{2m\lambda\gamma} \right) = \frac{1}{\min_{i,k}\{p_i, q_k\}} \left(1 + \frac{9}{2}\kappa''_{\text{rand}} \right),$$

  where $\kappa''_{\text{rand}} = \|X\|_F^2/(m\lambda\gamma)$. Moreover, we can set the step sizes as (see Appendix A.1)

$$\sigma_i = \frac{m\lambda}{9\|X\|_F^2}, \qquad \tau_k = \frac{m\gamma_i}{9\|X\|_F^2}.$$

- For the ERM problem (3), we assume that each loss function ϕ_j, for $j = 1, \ldots, N$, is $1/\nu$-smooth. According to (4), the smooth parameter for each f_i is $\gamma_i = \gamma = (N/m)\nu$. Let R be the largest Euclidean norm among all rows of X (or we can normalize each row to have the same norm R), then we have $\|X\|_F^2 \leq NR^2$ and

$$\kappa''_{\text{rand}} = \frac{\|X\|_F^2}{m\lambda\gamma} \leq \frac{NR^2}{m\lambda\gamma} = \frac{R^2}{\lambda\nu}. \tag{28}$$

  The upper bound $R^2/(\lambda\nu)$ is a condition number used for characterizing the iteration complexity of many randomized algorithms for ERM (e.g., Shalev-Shwartz and Zhang, 2013; Le Roux et al., 2012; Johnson and Zhang, 2013; Defazio et al., 2014; Zhang and Xiao, 2017). In this case, using the non-uniform sampling in (27), we can set the step sizes to be

$$\sigma_i = \frac{\lambda}{9R^2}\frac{m}{N}, \qquad \tau_k = \frac{\gamma}{9R^2}\frac{m}{N} = \frac{\nu}{9R^2}. \tag{29}$$

Next we estimate the overall computation complexity of DSCOVR-SVRG in order to achieve $\mathbf{E}[\Omega(\bar{w}^{(s)} - w^\star, \bar{\alpha}^{(s)} - \alpha^\star)] \leq \epsilon$. From (25), the number of stages required is $\log(\Omega^{(0)}/\epsilon)/\log(3/2)$, where $\Omega^{(0)} = \Omega(\bar{w}^{(0)} - w^\star, \bar{\alpha}^{(0)} - \alpha^\star)$. The number of inner iterations within each stage is $M = \log(3)\Gamma$. At the beginning of of each stage, computing the batch gradients $\bar{u}^{(s)}$ and $\bar{v}^{(s)}$

requires going through the whole data set X, whose computational cost is equivalent to $m \times n$ inner iterations. Therefore, the overall complexity of Algorithm 2, measured by total number of inner iterations, is

$$O\left((mn + \Gamma) \log\left(\frac{\Omega^{(0)}}{\epsilon} \right) \right).$$

To simplify discussion, we further assume $m \leq n$, which is always the case for distributed implementation (see Figure 2 and Section 7). In this case, we can let $\Gamma = n(1 + (9/2)\kappa_{\text{rand}})$. Thus the above iteration complexity becomes

$$O\big(n(1 + m + \kappa_{\text{rand}}) \log(1/\epsilon)\big). \tag{30}$$

Since the iteration complexity in (30) counts the number of blocks X_{ik} being processed, the number of passes over the whole dataset X can be obtained by dividing it by mn, i.e.,

$$O\left(\left(1 + \frac{\kappa_{\text{rand}}}{m} \right) \log(1/\epsilon) \right). \tag{31}$$

This is the computation complexity of DSCOVR listed in Table 1. We can replace κ_{rand} by κ'_{rand} or κ''_{rand} depending on different proof techniques and sampling probabilities as discussed above. We will address the communication complexity for DSCOVR-SVRG, including its decomposition into synchronous and asynchronous ones, after describing its implementation details in Section 7.

In addition to convergence to the saddle point, our next result shows that the primal-dual optimality gap also enjoys the same convergence rate, under slightly different conditions.

Theorem 2 *Suppose Assumption 1 holds, and let $P(w)$ and $D(\alpha)$ be the primal and dual functions defined in* (6) *and* (7)*, respectively. Let Λ and Γ be two constants that satisfy*

$$\Lambda \ \geq \ \|X_{ik}\|_F^2, \qquad i = 1, \ldots, m, \quad k = 1, \ldots, n,$$

and

$$\Gamma \ \geq \ \max_{i,k} \left\{ \frac{1}{p_i} \left(1 + \frac{18\Lambda}{q_k\lambda\gamma_i} \right), \ \frac{1}{q_k} \left(1 + \frac{18n\Lambda}{p_i m\lambda\gamma_i} \right) \right\}.$$

In Algorithm 2, if we choose the step sizes as

$$\sigma_i \ = \ \frac{1}{\gamma_i(p_i\Gamma - 1)}, \qquad i = 1, \ldots, m, \tag{32}$$

$$\tau_k \ = \ \frac{1}{\lambda(q_k\Gamma - 1)}, \qquad k = 1, \ldots, n, \tag{33}$$

and the number of iterations during each stage satisfies $M \geq \log(3)\Gamma$, then

$$\mathbf{E}\left[P(\bar{w}^{(s)}) - D(\bar{\alpha}^{(s)}) \right] \ \leq \ \left(\frac{2}{3} \right)^s 2\Gamma \left(P(\bar{w}^{(0)}) - D(\bar{\alpha}^{(0)}) \right). \tag{34}$$

The proof of Theorem 2 is given in Appendix B. In terms of iteration complexity or total number of passes to reach $\mathbf{E}[P(\bar{w}^{(s)}) - D(\bar{\alpha}^{(s)})] \leq \epsilon$, we need to add an extra factor of $\log(1 + \kappa_{\text{rand}})$ to (30) or (31), due to the factor Γ on the right-hand side of (34).

Algorithm 3 DSCOVR-SAGA

input: initial points $w^{(0)}, \alpha^{(0)}$, and number of iterations M.

1: $\bar{u}^{(0)} = Xw^{(0)}$ and $\bar{v}^{(0)} = \frac{1}{m}X^T\alpha^{(0)}$

2: $U_{ik}^{(0)} = X_{ik}w_k^{(0)}$, $V_{ik}^{(0)} = \frac{1}{m}(\alpha_i^{(0)})^T X_{ik}$, for all $i = 1, \ldots, m$ and $k = 1, \ldots, K$.

3: **for** $t = 0, 1, 2, \ldots, M-1$ **do**

4: pick $j \in \{1, \ldots, m\}$ and $l \in \{1, \ldots, n\}$ randomly with distributions p and q respectively.

5: compute variance-reduced stochastic gradients:

$$u_j^{(t+1)} = \bar{u}_j^{(t)} - \frac{1}{q_l}U_{jl}^{(t)} + \frac{1}{q_l}X_{jl}w_l^{(t)}, \tag{35}$$

$$v_l^{(t+1)} = \bar{v}_l^{(t)} - \frac{1}{p_j}(V_{jl}^{(t)})^T + \frac{1}{p_j}\frac{1}{m}(X_{jl})^T\alpha_j^{(t)}. \tag{36}$$

6: update primal and dual block coordinates:

$$\alpha_i^{(t+1)} = \begin{cases} \mathbf{prox}_{\sigma_j\Phi_j^*}(\alpha_j^{(t)} + \sigma_j u_j^{(t+1)}) & \text{if } i = j. \\ \alpha_i^{(t)}, & \text{if } i \neq j, \end{cases}$$

$$w_k^{(t+1)} = \begin{cases} \mathbf{prox}_{\tau_l g_l}(w_l^{(t)} - \tau_l v_l^{(t+1)}) & \text{if } k = l, \\ w_k^{(t)}, & \text{if } i \neq j. \end{cases}$$

7: update averaged stochastic gradients:

$$\bar{u}_i^{(t+1)} = \begin{cases} \bar{u}_j^{(t)} - U_{jl}^{(t)} + X_{jl}w_l^{(t)} & \text{if } i = j, \\ \bar{u}_i^{(t)} & \text{if } i \neq j, \end{cases}$$

$$\bar{v}_k^{(t+1)} = \begin{cases} \bar{v}_l^{(t)} - (V_{jl}^{(t)})^T + \frac{1}{m}(X_{jl})^T\alpha_j^{(t)} & \text{if } k = l, \\ \bar{v}_k^{(t)} & \text{if } k \neq l, \end{cases}$$

8: update the table of historical stochastic gradients:

$$U_{ik}^{(t+1)} = \begin{cases} X_{jl}w_l^{(t)} & \text{if } i = j \text{ and } k = l, \\ U_{ik}^{(t)} & \text{otherwise.} \end{cases}$$

$$V_{ik}^{(t+1)} = \begin{cases} \frac{1}{m}((X_{jl})^T\alpha_j^{(t)})^T & \text{if } i = j \text{ and } k = l, \\ V_{ik}^{(t)} & \text{otherwise.} \end{cases}$$

9: **end for**

output: $w^{(M)}$ and $\alpha^{(M)}$.

4. The DSCOVR-SAGA Algorithm

Algorithm 3 is a DSCOVR algorithm that uses the techniques of SAGA (Defazio et al., 2014) for variance reduction. This is a single stage algorithm with iterations indexed by t. In order to compute the variance-reduced stochastic gradients $u_j^{(t+1)}$ and $v_l^{(t+1)}$ at each iteration, we also need to maintain and update two vectors $\bar{u}^{(t)} \in \mathbf{R}^N$ and $\bar{v}^{(t)} \in \mathbf{R}^d$, and two matrices $U^{(t)} \in \mathbf{R}^{N\times n}$ and $V^{(t)} \in \mathbf{R}^{m\times d}$.

The vector $\bar{u}^{(t)}$ shares the same partition as $\alpha^{(t)}$ into m blocks, and $\bar{v}^{(t)}$ share the same partitions as $w^{(t)}$ into n blocks. The matrix $U^{(t)}$ is partitioned into $m \times n$ blocks, with each block $U_{ik}^{(t)} \in \mathbf{R}^{N_i\times 1}$. The matrix $V^{(t)}$ is also partitioned into $m \times n$ blocks, with each block $V_{ik}^{(t)} \in \mathbf{R}^{1\times d_k}$. According to the updates in Steps 7 and 8 of Algorithm 3, we have

$$\bar{u}_i^{(t)} = \sum_{k=1}^n U_{ik}^{(t)}, \qquad i = 1, \ldots, m, \tag{37}$$

$$\bar{v}_k^{(t)} = \sum_{i=1}^m (V_{ik}^{(t)})^T, \qquad k = 1, \ldots, n. \tag{38}$$

Based on the above constructions, we can show that $u_j^{(t+1)}$ is an unbiased stochastic gradient of $(\alpha^{(t)})^T Xw^{(t)}$ with respect to α_j, and $v_l^{(t+1)}$ is an unbiased stochastic gradient of $(1/m)((\alpha^{(t)})^T Xw^{(t)})$ with respect to w_l. More specifically, according to (35), we have

$$\begin{aligned} \mathbf{E}_l[u_j^{(t+1)}] &= \bar{u}_j^{(t)} - \sum_{k=1}^n q_k\left(\frac{1}{q_k}U_{jk}^{(t)}\right) + \sum_{k=1}^n q_k\left(\frac{1}{q_k}X_{jk}w_k^{(t)}\right) \\ &= \bar{u}_j^{(t)} - \sum_{k=1}^n U_{jk}^{(t)} + \sum_{k=1}^n X_{jk}w_k^{(t)} \\ &= \bar{u}_j^{(t)} - \bar{u}_j^{(t)} + X_{j:}w^{(t)} \\ &= X_{j:}w^{(t)} = \frac{\partial}{\partial\alpha_j}\left((\alpha^{(t)})^T Xw^{(t)}\right), \end{aligned} \tag{39}$$

where the third equality is due to (37). Similarly, according to (36), we have

$$\begin{aligned} \mathbf{E}_j[v_l^{(t+1)}] &= \bar{v}_l^{(t)} - \sum_{i=1}^m p_i\left(\frac{1}{p_i}(V_{il}^{(t)})^T\right) + \sum_{i=1}^m p_i\left(\frac{1}{p_i m}(X_{il})^T\alpha_i^{(t)}\right) \\ &= \bar{v}_l^{(t)} - \sum_{i=1}^m V_{il}^{(t)} + \frac{1}{m}\sum_{i=1}^m (X_{il})^T\alpha_i^{(t)} \\ &= \bar{v}_l^{(t)} - \bar{v}_l^{(t)} + \frac{1}{m}(X_{:l})^T\alpha^{(t)} \\ &= \frac{1}{m}(X_{:l})^T\alpha^{(t)} = \frac{\partial}{\partial w_l}\left(\frac{1}{m}(\alpha^{(t)})^T Xw^{(t)}\right), \end{aligned} \tag{40}$$

where the third equality is due to (38). The following theorem is proved in Appendix C.

Theorem 3 *Suppose Assumption 1 holds, and let $(w^\star, \alpha^\star)$ be the unique saddle point of $L(w, \alpha)$. Let Γ be a constant that satisfies*

$$\Gamma \geq \max_{i,k}\left\{\frac{1}{p_i}\left(1 + \frac{9\|X_{ik}\|^2}{2q_k\lambda\gamma_i}\right), \frac{1}{q_k}\left(1 + \frac{9n\|X_{ik}\|^2}{2p_i m\lambda\gamma_i}\right), \frac{1}{p_i q_k}\right\}. \tag{41}$$

If we choose the step sizes as

$$\sigma_i = \frac{1}{2\gamma_i(p_i\Gamma - 1)}, \qquad i = 1, \ldots, m, \tag{42}$$

$$\tau_k = \frac{1}{2\lambda(q_k\Gamma - 1)}, \qquad k = 1, \ldots, n, \tag{43}$$

Algorithm 4 Accelerated DSCOVR

input: initial points $\tilde{w}^{(0)}$, $\tilde{\alpha}^{(0)}$, and parameter $\delta > 0$.

1: **for** $r = 0, 1, 2, \ldots,$ **do**

2: find an approximate saddle point of (46) using one of the following two options:

- *option 1:* run Algorithm 2 with $S = \frac{2\log(2(1+\delta))}{\log(3/2)}$ and $M = \log(3)\Gamma_\delta$ to obtain

$$(\tilde{w}^{(r+1)}, \tilde{\alpha}^{(r+1)}) = \text{DSCOVR-SVRG}(\tilde{w}^{(r)}, \tilde{\alpha}^{(r)}, S, M).$$

- *option 2:* run Algorithm 3 with $M = 6\log\left(\frac{8(1+\delta)}{3}\right)\Gamma_\delta$ to obtain

$$(\tilde{w}^{(r+1)}, \tilde{\alpha}^{(r+1)}) = \text{DSCOVR-SAGA}(\tilde{w}^{(r)}, \tilde{\alpha}^{(r)}, M).$$

3: **end for**

Then the iterations of Algorithm 3 satisfy, for $t = 1, 2, \ldots$,

$$\mathbf{E}\left[\Omega(w^{(t)} - w^\star, \alpha^{(t)} - \alpha^\star)\right] \;\le\; \left(1 - \frac{1}{3\Gamma}\right)^t \frac{4}{3}\Omega(w^{(0)} - w^\star, \alpha^{(0)} - \alpha^\star). \tag{44}$$

The condition on Γ in (41) is very similar to the one in (22), except that here we have an additional term $1/(p_i q_k)$ when taking the maximum over i and k. This results in an extra mn term in estimating Γ under uniform sampling. Assuming $m \le n$ (true for distributed implementation), we can let

$$\Gamma = n\left(1 + \frac{9}{2}\kappa_{\text{rand}}\right) + mn.$$

According to (44), in order to achieve $\mathbf{E}[\Omega(w^{(t)} - w^\star, \alpha^{(t)} - \alpha^\star)] \le \epsilon$, DSCOVR-SAGA needs $O\left(\Gamma\log(1/\epsilon)\right)$ iterations. Using the above expression for Γ, the iteration complexity is

$$O\big(n(1 + m + \kappa_{\text{rand}})\log(1/\epsilon)\big), \tag{45}$$

which is the same as (30) for DSCOVR-SVRG. This also leads to the same computational complexity measured by the number of passes over the whole dataset, which is given in (31). Again we can replace κ_{rand} by κ'_{rand} or κ''_{rand} as discussed in Section 3. We will discuss the communication complexity of DSCOVR-SAGA in Section 7, after describing its implementation details.

5. Accelerated DSCOVR Algorithms

In this section, we develop an accelerated DSCOVR algorithm by following the "catalyst" framework (Lin et al., 2015; Frostig et al., 2015). More specifically, we adopt the same procedure by Balamurugan and Bach (2016) for solving convex-concave saddle-point problems.

Algorithm 4 proceeds in rounds indexed by $r = 0, 1, 2, \ldots$. Given the initial points $\tilde{w}^{(0)} \in \mathbf{R}^d$ and $\tilde{\alpha}^{(0)} \in \mathbf{R}^N$, each round r computes two new vectors $\tilde{w}^{(r+1)}$ and $\tilde{\alpha}^{(r+1)}$ using either the DSCOVR-SVRG or DSCOVR-SAGA algorithm for solving a regulated saddle-point problem, similar to the classical proximal point algorithm (Rockafellar, 1976).

Let $\delta > 0$ be a parameter which we will determine later. Consider the following perturbed saddle-point function for round r:

$$L_\delta^{(r)}(w, a) = L(w, \alpha) + \frac{\delta\lambda}{2}\|w - \tilde{w}^{(r)}\|^2 - \frac{\delta}{2m}\sum_{i=1}^m \gamma_i\|\alpha_i - \tilde{\alpha}_i^{(r)}\|^2. \tag{46}$$

Under Assumption 1, the function $L_\delta^{(r)}(w, a)$ is $(1+\delta)\lambda$-strongly convex in w and $(1+\delta)\gamma_i/m$-strongly concave in α_i. Let Γ_δ be a constant that satisfies

$$\Gamma_\delta \;\ge\; \max_{i,k}\left\{\frac{1}{p_i}\left(1 + \frac{9\|X_{ik}\|^2}{2q_k\lambda\gamma_i(1+\delta)^2}\right),\; \frac{1}{q_k}\left(1 + \frac{9n\|X_{ik}\|^2}{2p_i m\lambda\gamma_i(1+\delta)^2}\right),\; \frac{1}{p_i q_k}\right\},$$

where the right-hand side is obtained from (41) by replacing λ and γ_i with $(1+\delta)\lambda$ and $(1+\delta)\gamma_i$ respectively. The constant Γ_δ is used in Algorithm 4 to determine the number of inner iterations to run with each round, as well as for setting the step sizes. The following theorem is proved in Appendix D.

Theorem 4 *Suppose Assumption 1 holds, and let $(w^\star, \alpha^\star)$ be the saddle-point of $L(w, \alpha)$. With either options in Algorithm 4, if we choose the step sizes (inside Algorithm 2 or Algorithm 3) as*

$$\sigma_i \;=\; \frac{1}{2(1+\delta)\gamma_i(p_i\Gamma_\delta - 1)}, \qquad i = 1, \ldots, m, \tag{47}$$

$$\tau_k \;=\; \frac{1}{2(1+\delta)\lambda(q_k\Gamma_\delta - 1)}, \qquad k = 1, \ldots, n. \tag{48}$$

Then for all $r \ge 1$,

$$\mathbf{E}\left[\Omega(\tilde{w}^{(r)} - w^\star, \tilde{\alpha}^{(r)} - \alpha^\star)\right] \;\le\; \left(1 - \frac{1}{2(1+\delta)}\right)^{2r}\Omega(\tilde{w}^{(0)} - w^\star, \tilde{\alpha}^{(0)} - \alpha^\star).$$

According to Theorem 4, in order to have $\mathbf{E}[\Omega(\tilde{w}^{(r)} - w^\star, \tilde{\alpha}^{(r)} - \alpha^\star)] \le \epsilon$, we need the number of rounds r to satisfy

$$r \ge (1+\delta)\log\Big(\frac{\Omega(\tilde{w}^{(0)} - w^\star, \tilde{\alpha}^{(0)} - \alpha^\star)}{\epsilon}\Big).$$

Following the discussions in Sections 3 and 4, when using uniform sampling and assuming $m \le n$, we can have

$$\Gamma_\delta = n\left(1 + \frac{9\kappa_{\text{rand}}}{2(1+\delta)^2}\right) + mn. \tag{49}$$

Then the total number of block coordinate updates in Algorithm 4 is

$$O((1+\delta)\Gamma_\delta\log(1+\delta)\log(1/\epsilon)),$$

where the $\log(1+\delta)$ factor comes from the number of stages S in option 1 and number of steps M in option 2. We hide the $\log(1+\delta)$ factor with the $\widetilde{O}$ notation and plug (49) into the expression above to obtain

$$\widetilde{O}\left(n\left((1+\delta)(1+m) + \frac{\kappa_{\text{rand}}}{(1+\delta)}\right)\log\left(\frac{1}{\epsilon}\right)\right).$$

Now we can choose δ depending on the relative size of κ_{rand} and m:

- If $\kappa_{\text{rand}} > 1 + m$, we can minimizing the above expression by choosing $\delta = \sqrt{\frac{\kappa_{\text{rand}}}{1+m}} - 1$, so that the overall iteration complexity becomes $\widetilde{O}\left(n\sqrt{m\kappa_{\text{rand}}}\log(1/\epsilon)\right)$.
- If $\kappa_{\text{rand}} \le m + 1$, then no acceleration is necessary and we can choose $\delta = 0$ to proceed with a single round. In this case, the iteration complexity is $O(mn)$ as seen from (49).

Therefore, in either case, the total number of block iterations by Algorithm 4 can be written as

$$\widetilde{O}\left(mn + n\sqrt{m\kappa_{\text{rand}}}\log(1/\epsilon)\right). \tag{50}$$

As discussed before, the total number of passes over the whole dataset is obtained by dividing by mn:

$$\widetilde{O}\left(1 + \sqrt{\kappa_{\text{rand}}/m}\log(1/\epsilon)\right).$$

This is the computational complexity of accelerated DSCOVR listed in Table 1.

5.1. Proximal Mapping for Accelerated DSCOVR

When applying Algorithm 2 or 3 to approximate the saddle-point of (46), we need to replace the proximal mappings of $g_k(\cdot)$ and $f_i^*(\cdot)$ by those of $g_k(\cdot) + (\delta\lambda/2)\|\cdot - \tilde{w}_k^{(r)}\|^2$ and $f_i^*(\cdot) + (\delta\gamma_i/2)\|\cdot - \tilde{\alpha}_i^{(r)}\|^2$, respectively. More precisely, we replace $w_k^{(t+1)} = \mathbf{prox}_{\tau_k g_k}\left(w_k^{(t)} - \tau_k v_k^{(t+1)}\right)$ by

$$\begin{aligned} w_k^{(t+1)} &= \operatorname*{arg\,min}_{w_k \in \mathbf{R}^{d_k}} \left\{ g_k(w_k) + \frac{\delta\lambda}{2}\|w_k - \tilde{w}_k^{(r)}\|^2 + \frac{1}{2\tau_k}\left\|w_k - \left(w_k^{(t)} - \tau_k v_k^{(t+1)}\right)\right\|^2 \right\} \\ &= \mathbf{prox}_{\frac{\tau_k}{1+\tau_k\delta\lambda} g_k}\left(\frac{1}{1+\tau_k\delta\lambda}\left(w_k^{(t)} - \tau_k v_k^{(t+1)}\right) + \frac{\tau_k\delta\lambda}{1+\tau_k\delta\lambda}\tilde{w}_k^{(r)}\right), \end{aligned} \tag{51}$$

and replace $\alpha_i^{(t+1)} = \mathbf{prox}_{\sigma_i f_i^*}\left(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}\right)$ by

$$\begin{aligned} \alpha_i^{(t+1)} &= \operatorname*{arg\,min}_{\alpha_i \in \mathbf{R}^{N_i}} \left\{ f_i^*(\alpha_i) + \frac{\delta\gamma_i}{2}\|\alpha_i - \tilde{\alpha}_i^{(r)}\|^2 + \frac{1}{2\sigma_i}\left\|\alpha_i - \left(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}\right)\right\|^2 \right\} \\ &= \mathbf{prox}_{\frac{\sigma_i}{1+\sigma_i\delta\gamma_i} f_i^*}\left(\frac{1}{1+\sigma_i\delta\gamma_i}\left(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}\right) + \frac{\sigma_i\delta\gamma_i}{1+\sigma_i\delta\gamma_i}\tilde{\alpha}_i^{(r)}\right). \end{aligned} \tag{52}$$

We also examine the number of inner iterations determined by Γ_δ and how to set the step sizes. If we choose $\delta = \sqrt{\frac{\kappa_{\text{rand}}}{1+m}} - 1$, then Γ_δ in (49) becomes

$$\Gamma_\delta = n\left(1 + \frac{9\kappa_{\text{rand}}}{2(1+\delta)^2}\right) + mn = n\left(1 + \frac{9\kappa_{\text{rand}}}{2\kappa_{\text{rand}}/(m+1)}\right) + mn = 5.5(m+1)n.$$

Therefore a small constant number of passes is sufficient within each round. Using the uniform sampling, the step sizes can be estimated as follows:

$$\sigma_i = \frac{1}{2(1+\delta)\gamma_i(p_i\Gamma_\delta - 1)} \approx \frac{1}{2\sqrt{\kappa_{\text{rand}}/m}\gamma_i(5.5n-1)} \approx \frac{1}{11\gamma_i n\sqrt{\kappa_{\text{rand}}/m}}, \tag{53}$$

$$\tau_k = \frac{1}{2(1+\delta)\lambda(q_k\Gamma_\delta - 1)} \approx \frac{1}{2\sqrt{\kappa_{\text{rand}}/m}\lambda(5.5m-1)} \approx \frac{1}{11\lambda\sqrt{m\cdot\kappa_{\text{rand}}}}. \tag{54}$$

As shown by our numerical experiments in Section 8, the step sizes can be set much larger in practice.

6. Conjugate-Free DSCOVR Algorithms

A major disadvantage of primal-dual algorithms for solving problem (1) is the requirement of computing the proximal mapping of the conjugate function f_i^*, which may not admit closed-formed solution or efficient computation. This is especially the case for logistic regression, one of the most popular loss functions used in classification.

Lan and Zhou (2018) developed "conjugate-free" variants of primal-dual algorithms that avoid computing the proximal mapping of the conjugate functions. The main idea is to replace the Euclidean distance in the dual proximal mapping with a Bregman divergence defined over the conjugate function itself. This technique has been used by Wang and Xiao (2017) to solve structured ERM problems with primal-dual first order methods. Here we use this approach to derive conjugate-free DSCOVR algorithms. In particular, we replace the proximal mapping for the dual update

$$\alpha_i^{(t+1)} = \mathbf{prox}_{\sigma_i f_i^*}\left(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}\right) = \operatorname*{arg\,min}_{\alpha_i \in \mathbf{R}^{n_i}}\left\{ f_i^*(\alpha_i) - \langle \alpha_i, u_i^{(t+1)}\rangle + \frac{1}{2\sigma_i}\|\alpha_i - \alpha_i^{(t)}\|^2 \right\},$$

by

$$\alpha_i^{(t+1)} = \operatorname*{arg\,min}_{\alpha_i \in \mathbf{R}^{n_i}}\left\{ f_i^*(\alpha_i) - \langle \alpha_i, u_i^{(t+1)}\rangle + \frac{1}{\sigma_i}\mathcal{B}_i(\alpha_i, \alpha_i^{(t)}) \right\}, \tag{55}$$

where $\mathcal{B}_i(\alpha_i, \alpha_i^{(t)}) = f_i^*(\alpha_i) - \langle \nabla f_i^*(\alpha_i^{(t)}),\, \alpha_i - \alpha_i^{(t)}\rangle$. The solution to (55) is given by

$$\alpha_i^{(t+1)} = \nabla f_i(\beta_i^{(t+1)}),$$

where $\beta_i^{(t+1)}$ can be computed recursively by

$$\beta_i^{(t+1)} = \frac{\beta_i^{(t)} + \sigma_i u_i^{(t+1)}}{1+\sigma_i}, \qquad t \ge 0,$$

with initial condition $\beta_i^{(0)} = \nabla f_i^*(\alpha_i^{(0)})$ (see Lan and Zhou, 2018, Lemma 1). Therefore, in order to update the dual variables α_i, we do not need to compute the proximal mapping for the conjugate function f_i^*; instead, taking the gradient of f_i at some easy-to-compute points is sufficient. This conjugate-free update can be applied in Algorithms 1, 2 and 3.

For the accelerated DSCOVR algorithms, we replace (52) by

$$\alpha_i^{(t+1)} = \operatorname*{arg\,min}_{\alpha_i \in \mathbf{R}^{n_i}}\left\{ f_i^*(\alpha_i) - \langle \alpha_i, u_i^{(t+1)}\rangle + \frac{1}{\sigma_i}\mathcal{B}_i(\alpha_i, \alpha_i^{(t)}) + \delta\gamma\mathcal{B}_i(\alpha_i, \tilde{\alpha}_i^{(t+1)}) \right\}.$$

The solution to the above minimization problem can also be written as

$$\alpha_i^{(t+1)} = \nabla f_i(\beta_i^{(t+1)}),$$

where $\beta_i^{(t+1)}$ can be computed recursively as

$$\beta_i^{(t+1)} = \frac{\beta_t^{(t)} + \sigma_i u_i^{(t+1)} + \sigma_i\delta\gamma\tilde{\beta}_i}{1+\sigma_i+\sigma_i\delta\gamma}, \qquad t \ge 0,$$

with the initialization $\beta_i^{(0)} = \nabla f_i^*(\alpha_i^{(0)})$ and $\tilde{\beta}_i = \nabla f_i^*(\tilde{\alpha}_i^{(r)})$.

The convergence rates and computational complexities of the conjugate-free DSCOVR algorithms are very similar to the ones given in Sections 3–5. We omit details here, but refer the readers to Lan and Zhou (2018) and Wang and Xiao (2017) for related results.

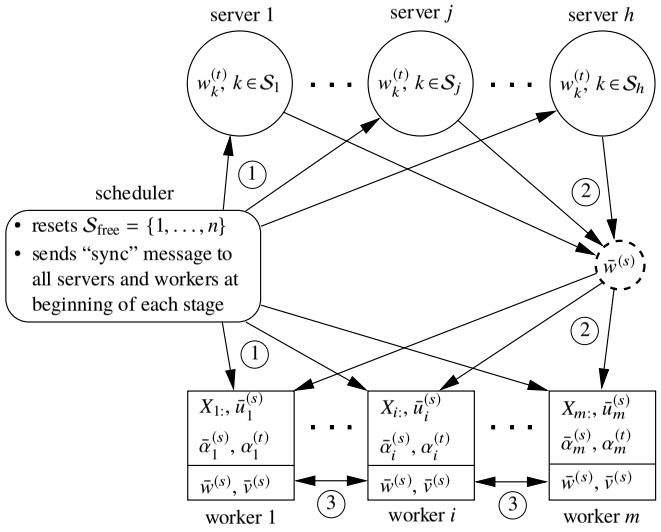


Figure 3: A distributed system for implementing DSCOVR consists of m workers, h parameter servers, and one scheduler. The arrows labeled with the numbers 1, 2 and 3 represent three collective communications at the beginning of each stage in DSCOVR-SVRG.

7. Asynchronous Distributed Implementation

In this section, we show how to implement the DSCOVR algorithms presented in Sections 3–6 in a distributed computing system. We assume that the system provide both synchronous collective communication and asynchronous point-to-point communication, which are all supported by the MPI standard (MPI Forum, 2012). Throughout this section, we assume $m < n$ (see Figure 2).

7.1. Implementation of DSCOVR-SVRG

In order to implement Algorithm 2, the distributed system need to have the following components (see Figure 3):

- *m workers*. Each worker i, for $i = 1, \ldots, m$, stores the following local data and variables :
  - data matrix $X_{i:} \in \mathbf{R}^{N_i \times d}$.
  - vectors in $\mathbf{R}^{N_i}$: $\bar{u}_i^{(s)}, \alpha_i^{(t)}, \bar{\alpha}_i^{(s)}$.
  - vectors in $\mathbf{R}^d$: $\bar{w}^{(s)}, \bar{v}^{(s)}$.
  - extra buffers for computation and communication: $u_j^{(t+1)}, v_l^{(t+1)}, w_l^{(t)}$ and $w_l^{(t+1)}$.

- *h parameter servers*. Each server j stores a subset of the blocks $\{w_k^{(t)} \in \mathbf{R}^{d_k} : k \in \mathcal{S}_j\}$, where $\mathcal{S}_1, \ldots, \mathcal{S}_h$ form a partition of the set $\{1, \ldots, n\}$.
- *one scheduler*. It maintains a set of block indices $\mathcal{S}_{\text{free}} \subseteq \{1, \ldots, n\}$. At any given time, $\mathcal{S}_{\text{free}}$ contains indices of parameter blocks that are not currently updated by any worker.

The reason for having $h > 1$ servers is not about insufficient storage for parameters, but rather to avoid the communication overload between only one server and all m workers (m can be in hundreds).

At the beginning of each stage s, the following three collective communications take place across the system (illustrated in Figure 3 by arrows with circled labels 1, 2 and 3):

(1) The scheduler sends a "sync" message to all servers and workers, and resets $\mathcal{S}_{\text{free}} = \{1, \ldots, n\}$.

(2) Upon receiving the "sync" message, the servers aggregate their blocks of parameters together to form $\bar{w}^{(s)}$ and send it to all workers (e.g., through the AllReduce operation in MPI).

(3) Upon receiving $\bar{w}^{(s)}$, each worker compute $\bar{u}_i^{(s)} = X_{i:}\bar{w}^{(s)}$ and $(X_{i:})^T\bar{\alpha}_i^{(s)}$, then invoke a collective communication (AllReduce) to compute $\bar{v}^{(s)} = (1/m)\sum_{i=1}^m (X_{i:})^T\bar{\alpha}_i^{(s)}$.

The number of vectors in $\mathbf{R}^d$ sent and received during the above process is $2m$, counting the communications to form $\bar{w}^{(s)}$ and $\bar{v}^{(s)}$ at m workers (ignoring the short "sync" messages).

After the collective communications at the beginning of each stage, all workers start working on the inner iterations of Algorithm 2 in parallel in an asynchronous, event-driven manner. Each worker interacts with the scheduler and the servers in a four-step loop shown in Figure 4. There are always m iterations taking place concurrently (see also Figure 2), each may at a different phase of the four-step loop:

(1) Whenever worker i finishes updating a block k', it sends the pair (i, k') to the scheduler to request for another block to update. At the beginning of each stage, k' is not needed.

(2) When the scheduler receives the pair (i, k'), it randomly choose a block k from the list of free blocks $\mathcal{S}_{\text{free}}$ (which are not currently updated by any worker), looks up for the server j which stores the parameter block $w_k^{(t)}$ (i.e., $\mathcal{S}_j \ni k$), and then send the pair (i, k) to server j. In addition, the scheduler updates the list $\mathcal{S}_{\text{free}}$ by adding k' and deleting k.

(3) When server j receives the pair (i, k), it sends the vector $w_k^{(t)}$ to worker i, and waits for receiving the updated version $w_k^{(t+1)}$ from worker i.

(4) After worker i receives $w_k^{(t)}$, it computes the updates $\alpha_i^{(t)}$ and $w_k^{(t)}$ following steps 6-7 in Algorithm 2, and then send $w_k^{(t+1)}$ back to server j. At last, it assigns the value of k to k' and send the pair (i, k') to the scheduler, requesting the next block to work on.

The amount of point-to-point communication required during the above process is $2d_k$ float numbers, for sending and receiving $w_k^{(t)}$ and $w_k^{(t+1)}$ (we ignore the small messages for sending and receiving (i, k') and (i, k)). Since the blocks are picked randomly, the average amount of communication per iteration is $2d/n$, or equivalent to $2/n$ vectors in $\mathbf{R}^d$. According to Theorem 1, each stage of Algorithm 2 requires $\log(3)\Gamma$ inner iterations; In addition, the discussions above (30) show that we can take $\Gamma = n(1 + (9/2)\kappa_{\text{rand}})$. Therefore, the average amount of point-to-point communication within each stage is $O(\kappa_{\text{rand}})$ vectors in $\mathbf{R}^d$.

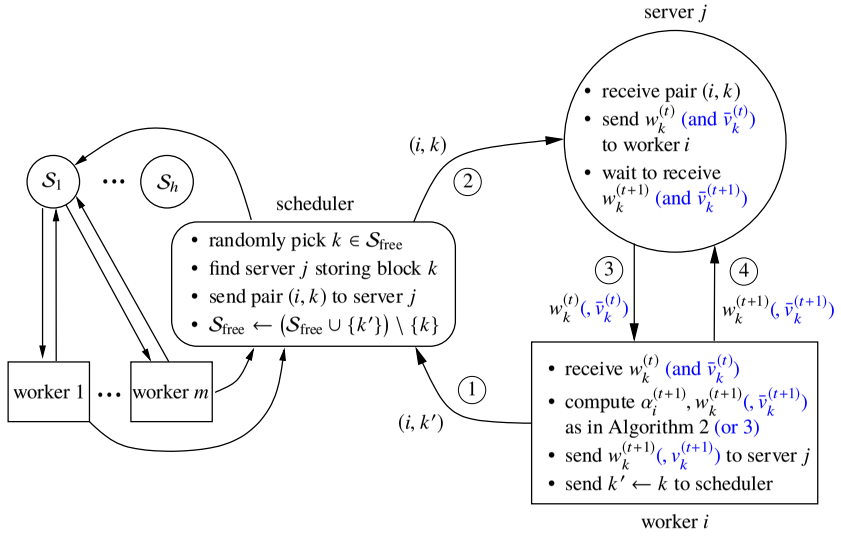


Figure 4: Communication and computation processes for one inner iteration of DSCOVR-SVRG (Algorithm 2). The blue texts in the parentheses are the additional vectors required by DSCOVR-SAGA (Algorithm 3). There are always m iterations taking place in parallel asynchronously, each evolving around one worker. A server may support multiple (or zero) iterations if more than one (or none) of its stored parameter blocks are being updated.

Now we are ready to quantify the communication complexity of DSCOVR-SVRG to find an ϵ-optimal solution. Our discussions above show that each stage requires collective communication of $2m$ vectors in $\mathbf{R}^d$ and asynchronous point-to-point communication of equivalently κ_{rand} such vectors. Since there are total $O(\log(1/\epsilon))$ stages, the total communication complexity is

$$O\left((m + \kappa_{\text{rand}}) \log(1/\epsilon)\right).$$

This gives the communication complexity shown in Table 1, as well as its decomposition in Table 2.

7.2. Implementation of DSCOVR-SAGA

We can implement Algorithm 3 using the same distributed system shown in Figure 3, but with some modifications described below. First, the storage at different components are different:

- *m workers.* Each worker i, for $i = 1, \ldots, m$, stores the following data and variables:
  - data matrix $X_{i:} \in \mathbf{R}^{N_i \times d}$
  - vectors in $\mathbf{R}^{N_i}$: $\alpha_i^{(t)}$, $u_i^{(t)}$, $\bar{u}_i^{(t)}$, and $U_{ik}^{(t)}$ for $k = 1, \ldots, n$.

  - vector in $\mathbf{R}^d$: $V_{i:}^{(t)} = [V_{i1}^{(t)} \cdots V_{in}^{(t)}]^T$ (which is the ith row of $V^{(t)}$, with $V_{ik}^{(t)} \in \mathbf{R}^{1 \times d_k}$).
  - buffers for communication and update of $w_k^{(t)}$ and $\bar{v}_k^{(t)}$ (both stored at some server).
- *h servers.* Each server j stores a subset of blocks $\{w_k^{(t)}, \bar{v}_k^{(t)} \in \mathbf{R}^{d_k} : k \in \mathcal{S}_j\}$, for $j = 1, \ldots, n$.
- *one scheduler.* It maintains the set of indices $\mathcal{S}_{\text{free}} \subseteq \{1, \ldots, n\}$, same as in DSCOVR-SVRG.

Unlike DSCOVR-SVRG, there is no stage-wise "sync" messages. All workers and servers work in parallel asynchronously all the time, following the four-step loops illustrated in Figure 4 (including blue colored texts in the parentheses). Within each iteration, the main difference from DSCOVR-SVRG is that, the server and worker need to exchange two vectors of length d_k: $w_k^{(t)}$ and $v_k^{(t)}$ and their updates. This doubles the amount of point-to-point communication, and the average amount of communication per iteration is $4/n$ vectors of length d. Using the iteration complexity in (45), the total amount of communication required (measured by number of vectors of length d) is

$$O\left((m + \kappa_{\text{rand}}) \log(1/\epsilon)\right),$$

which is the same as for DSCOVR-SVRG. However, its decomposition into synchronous and asynchronous communication is different, as shown in Table 2. If the initial vectors $w^{(0)} \neq 0$ or $\alpha^{(0)} \neq 0$, then one round of collective communication is required to propagate the initial conditions to all servers and workers, which reflect the $O(m)$ synchronous communication in Table 2.

7.3. Implementation of Accelerated DSCOVR

Implementation of the accelerated DSCOVR algorithm is very similar to the non-accelerated ones. The main differences lie in the two proximal mappings presented in Section 5.1. In particular, the primal update in (51) needs the extra variable $\tilde{w}_k^{(r)}$, which should be stored at a parameter server together with $w_k^{(t)}$. We modify the four-step loops shown in Figures 4 as follows:

- Each parameter server j stores the extra block parameters $\{\tilde{w}_k^{(r)}, k \in \mathcal{S}_j\}$. During step (3), $\tilde{w}_k^{(r)}$ is send together with $w_k^{(t)}$ (for SVRG) or $(w_k^{(t)}, v_k^{(t)})$ (for SAGA) to a worker.
- In step (4), no update of $\tilde{w}_k^{(r)}$ is sent back to the server. Instead, whenever switching rounds, the scheduler will inform each server to update their $\tilde{w}_k^{(r)}$ to the most recent $w_k^{(t)}$.

For the dual proximal mapping in (52), each worker i needs to store an extra vector $\tilde{\alpha}_i^{(r)}$, and reset it to the most recent $\alpha_i^{(t)}$ when moving to the next round. There is no need for additional synchronization or collective communication when switching rounds in Algorithm 4. The communication complexity (measured by the number of vectors of length d sent or received) can be obtained by dividing the iteration complexity in (50) by n, i.e., $O((m + \sqrt{m\kappa_{\text{rand}}}) \log(1/\epsilon))$, as shown in Table 1.

Finally, in order to implement the conjugate-free DSCOVR algorithms described in Section 6, each worker i simply need to maintain and update an extra vector $\beta_i^{(t)}$ locally.

8. Experiments

In this section, we present numerical experiments on an industrial distributed computing system. This system has hundreds of computers connected by high speed Ethernet in a data center. The hardware and software configurations for each machine are listed in Table 3. We implemented all DSCOVR

CPU	#cores	RAM	network	operating system
dual Intel® Xeon® processors E5-2650 (v2), 2.6 GHz	16	128 GB 1.8 GHz	10 Gbps Ethernet adapter	Windows® Server (version 2012)

Table 3: Configuration of each machine in the distributed computing system.

algorithms presented in this paper, including the SVRG and SAGA versions, their accelerated variants, as well as the conjugate-free algorithms. All implementations are written in C++, using MPI for both collective and point-to-point communications (see Figures 3 and 4 respectively). On each worker machine, we also use OpenMP (OpenMP Architecture Review Board, 2011) to exploit the multi-core architecture for parallel computing, including sparse matrix-vector multiplications and vectorized function evaluations.

Implementing the DSCOVR algorithms requires $m+h+1$ machines, among them m are workers with local datasets, h are parameter servers, and one is a scheduler (see Figure 3). We focus on solving the ERM problem (3), where the total of N training examples are evenly partitioned and stored at m workers. We partition the d-dimensional parameters into n subsets of roughly the same size (differ at most by one), where each subset consists of randomly chosen coordinates (without replacement). Then we store the n subsets of parameters on h servers, each getting either $\lfloor n/h \rfloor$ or $\lceil n/h \rceil$ subsets. As described in Section 7, we make the configurations to satisfy $n > m > h \geq 1$.

For DSCOVR-SVRG and DSCOVR-SAGA, the step sizes in (29) are very conservative. In the experiments, we replace the coefficient $1/9$ by two tuning parameter $\eta_{\rm d}$ and $\eta_{\rm p}$ for the dual and primal step sizes respectively, i.e.,

$$\sigma_i = \eta_{\rm d} \frac{\lambda}{R^2} \cdot \frac{m}{N}, \qquad \tau_k = \eta_{\rm p} \frac{\nu}{R^2}. \tag{56}$$

For the accelerated DSCOVR algorithms, we use $\kappa_{\rm rand} = R^2/(\lambda\nu)$ as shown in (28) for ERM. Then the step sizes in (53) and (54), with $\gamma_i = (m/N)\nu$ and a generic constant coefficient η, become

$$\sigma_i = \frac{\eta_{\rm d}}{nR} \sqrt{\frac{m\lambda}{\nu}} \cdot \frac{m}{N}, \qquad \tau_k = \frac{\eta_{\rm p}}{R} \sqrt{\frac{\nu}{m\lambda}}. \tag{57}$$

For comparison, we also implemented the following first-order methods for solving problem 1:

- PGD: parallel implementation of the Proximal Gradient Descent method (using synchronous collective communication over m machines). We use the adaptive line search procedure proposed in Nesterov (2013), and the exact form used is Algorithm 2 in Lin and Xiao (2015).
- APG: parallel implementation of the Accelerated Proximal Gradient method (Nesterov, 2004, 2013). We use a similar adaptive line search scheme to the one for PGD, and the exact form used (with strong convexity) is Algorithm 4 in Lin and Xiao (2015).
- ADMM: the Alternating Direction Method of Multipliers. We use the regularized consensus version in Boyd et al. (2011, Section 7.1.1). For solving the local optimization problems at each node, we use the SDCA method (Shalev-Shwartz and Zhang, 2013).
- CoCoA+: the adding version of CoCoA in Ma et al. (2015). Following the suggestion in Ma et al. (2017), we use a randomized coordinate descent algorithm (Nesterov, 2012; Richtárik and Takáč, 2014) for solving the local optimization problems.

Dataset	#instances (N)	#features (d)	#nonzeros
`rcv1-train`	677,399	47,236	49,556,258
`webspam`	350,000	16,609,143	1,304,697,446
`splice-site`	50,000,000	11,725,480	166,167,381,622

Table 4: Statistics of three datasets. Each feature vector is normalized to have unit norm.

These four algorithms all require m workers only. Specifically, we use the AllReduce call in MPI for the collective communications so that a separate master machine is not necessary.

We conducted experiments on three binary classification datasets obtained from the collection maintained by Fan and Lin (2011). Table 4 lists their sizes and dimensions. In our experiments, we used two configurations: one with $m = 20$ and $h = 10$ for two relatively small datasets, `rcv1-train` and `webspam`, and the other with $m = 100$ and $h = 20$ for the large dataset `splice-site`.

For `rcv1-train`, we solve the ERM problem (3) with a smoothed hinge loss defined as

$$\phi_j(t) = \begin{cases} 0 & \text{if } y_j t \geq 1, \\ \frac{1}{2} - y_j t & \text{if } y_j t \leq 0, \\ \frac{1}{2}(1 - y_j t)^2 & \text{otherwise,} \end{cases} \qquad \text{and} \qquad \phi_j^*(\beta) = \begin{cases} y_j\beta + \frac{1}{2}\beta^2 & \text{if } -1 \leq y_j\beta \leq 0, \\ +\infty & \text{otherwise.} \end{cases}$$

for $j = 1, \ldots, N$. This loss function is 1-smooth, therefore $\nu = 1$; see discussion above (28). We use the ℓ_2 regularization $g(w) = (\lambda/2)\|w\|^2$. Figures 5 and 6 show the reduction of the primal objective gap $P(w^{(t)}) - P(w^\star)$ by different algorithms, with regularization parameter $\lambda = 10^{-4}$ and $\lambda = 10^{-6}$ respectively. All started from the zero initial point. Here the N examples are randomly shuffled and then divided into m subsets. The labels SVRG and SAGA mean DSCOVR-SVRG and DSCOVR-SAGA, respectively, and A-SVRG and A-SAGA are their accelerated versions.

Since PGD and APG both use adaptive line search, there is no parameter to tune. For ADMM, we manually tuned the penalty parameter ρ (see Boyd et al., 2011, Section 7.1.1) to obtain good performance: $\rho = 10^{-5}$ in Figure 5 and $\rho = 10^{-6}$ in Figure 6. For CoCoA+, two passes over the local datasets using a randomized coordinate descent method are sufficient for solving the local optimization problem (more passes do not give meaningful improvement). For DSCOVR-SVRG and SAGA, we used $\eta_{\rm p} = \eta_{\rm d} = 20$ to set the step sizes in (56). For DSCOVR-SVRG, each stage goes through the whole dataset 10 times, i.e., the number of inner iterations in Algorithm 2 is $M = 10mn$. For the accelerated DSCOVR algorithms, better performance are obtained with small periods to update the proximal points and we set it to be every 0.2 passes over the dataset, i.e., $0.2mn$ inner iterations. For accelerated DSCOVR-SVRG, we set the stage period (for variance reduction) to be $M = mn$, which is actually longer than the period for updating the proximal points.

From Figures 5 and 6, we observe that the two distributed algorithms based on model averaging, ADMM and CoCoA+, converges relatively fast in the beginning but becomes very slow in the later stage. Other algorithms demonstrate more consistent linear convergence rates. For $\lambda = 10^{-4}$, the DSCOVR algorithms are very competitive compared with other algorithms. For $\lambda = 10^{-6}$, the non-accelerated DSCOVR algorithms become very slow, even after tuning the step sizes. But the accelerated DSCOVR algorithms are superior in terms of both number of passes over data and wall-clock time (with adjusted step size coefficient $\eta_{\rm p} = 10$ and $\eta_{\rm d} = 40$).

For ADMM and CoCoA+, each marker represents the finishing of one iteration. It can be seen that they are mostly evenly spaced in terms of number of passes over data, but have large variations in terms of wall-clock time. The variations in time per iteration are due to resource sharing with

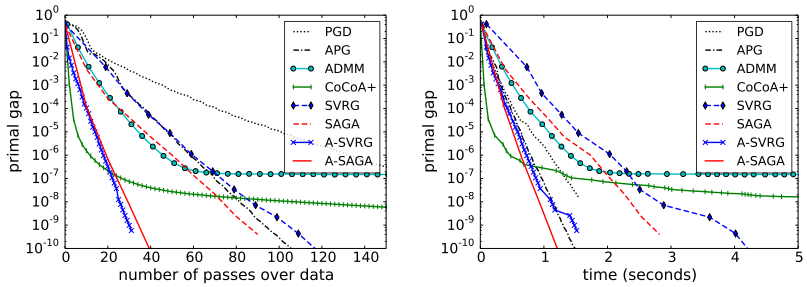


Figure 5: `rcv1-train`: smoothed-hinge loss, $\lambda = 10^{-4}$, randomly shuffled, $m = 20$, $n = 37$, $h = 10$.

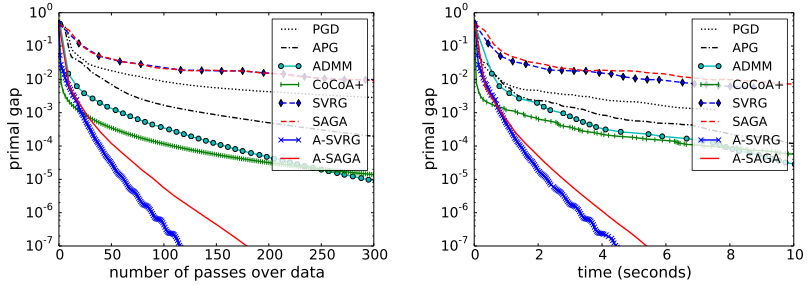


Figure 6: `rcv1-train`: smoothed-hinge loss, $\lambda = 10^{-6}$, randomly shuffled, $m = 20$, $n = 37$, $h = 10$.

other jobs running simultaneously on the distributed computing cluster. Even if we have exclusive use of each machine, sharing communications with other jobs over the Ethernet is unavoidable. This reflects the more realistic environment in cloud computing.

For the `webspam` dataset, we solve the ERM problem with logistic loss $\phi_j(t) = \log(1+\exp(-y_j t))$ where $y_j \in \{\pm 1\}$. The logistic loss is 1/4-smooth, so we have $\nu = 4$. Since the proximal mapping of its conjugate ϕ_j^* does not have a closed-form solution, we used the conjugate-free DSCOVR algorithms described in Section 6. Figures 7 and 8 shows the reduction of primal objective gap by different algorithms, for $\lambda = 10^{-4}$ and $\lambda = 10^{-6}$ respectively. Here the starting point is no longer the all-zero vectors. Instead, each machine i first computes a local solution by minimizing $f_i(X_i w) + g(w)$, and then compute their average using an AllReduce operation. Each algorithm starts from this average point. This averaging scheme has been proven to be very effective to warm start distributed algorithms for ERM (Zhang et al., 2013). In addition, it can be shown that when starting from the zero initial point, the first step of CoCoA+ computes exactly such an averaged point.

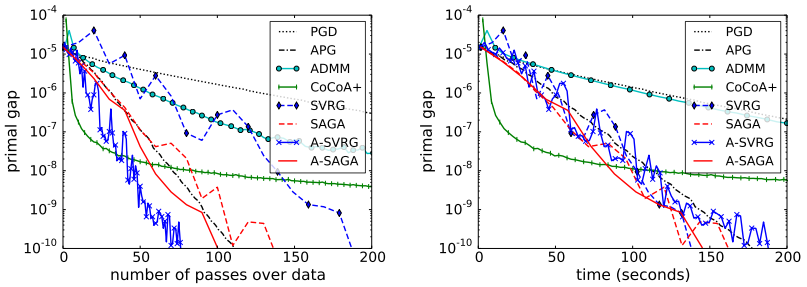


Figure 7: `webspam`: logistic regression, $\lambda = 10^{-4}$, randomly shuffled, $m = 20$, $n = 50$, $h = 10$.

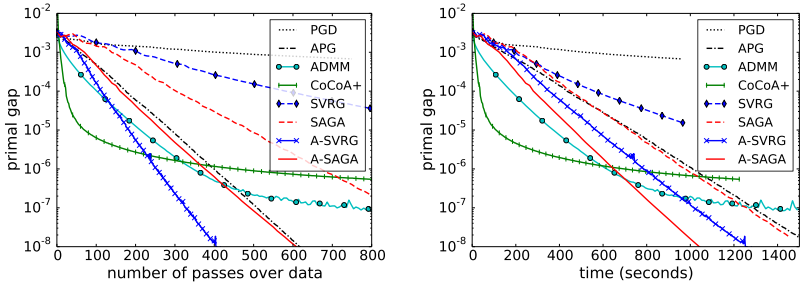


Figure 8: `webspam`: logistic regression, $\lambda = 10^{-6}$, randomly shuffled, $m = 20$, $n = 50$, $h = 10$.

From Figures 7 and 8, we again observe that CoCoA+ has very fast convergence in the beginning but converges very slowly towards higher precision. The DSCOVR algorithms, especially the accelerated variants, are very competitive in terms of both number of iterations and wall-clock time.

In order to investigate the fast initial convergence of CoCoA+ and ADMM, we repeated the experiments on `webspam` without random shuffling. More specifically, we sorted the N examples by their labels, and then partitioned them into m subsets sequentially. That is, most of the machines have data with only $+1$ or -1 labels, and only one machine has mixed ± 1 examples. The results are shown in Figures 9 and 10. Now the fast initial convergence of CoCoA+ and ADMM disappeared. In particular, CoCoA+ converges with very slow linear rate. This shows that statistical properties of random shuffling of the dataset is the main reason for the fast initial convergence of model-averaging based algorithms such as CoCoA+ and ADMM (see, e.g., Zhang et al., 2013).

On the other hand, this should not have any impact on PGD and APG, because their iterations are computed over the whole dataset, which is the same regardless of random shuffling or sorting. The differences between the plots for PGD and APG in Figures 7 and 9 (also for Figures 8 and 10) are due to different initial points computed through averaging local solutions, which does depends on the distribution of data at different machines.

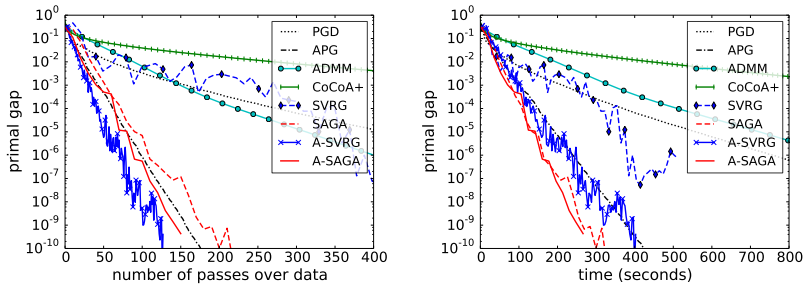


Figure 9: `webspam`: logistic regression, $\lambda = 10^{-4}$, sorted labels, $m = 20$, $n = 50$, $h = 10$.

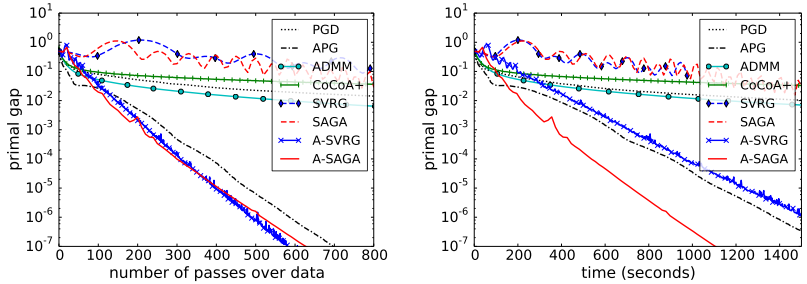


Figure 10: `webspam`: logistic regression, $\lambda = 10^{-6}$, sorted labels, $m = 20$, $n = 50$, $h = 10$.

Different ways for splitting the data over the m workers also affect the DSCOVR algorithms. In particular, the non-accelerated DSCOVR algorithms become very slow, as shown in Figures 9 and 10. However, the accelerated DSCOVR algorithms are still very competitive against the adaptive APG. The accelerated DSCOVR-SAGA algorithm performs best. In fact, the time spent by accelerated DSCOVR-SAGA should be even less than shown in Figures 9 and 10. Recall that other than the initialization with non-zero starting point, DSCOVR-SAGA is completely asynchronous and does not need any collective communication (see Section 7.2). However, in order to record the objective function for the purpose of plotting its progress, we added collective communication and computation to evaluate the objective value for every 10 passes over the data. For example, in Figure 10, such extra collective communications take about 160 seconds (about 15% of total time) for accelerated DSCOVR-SAGA, which can be further deducted from the horizontal time axis.

Finally, we conducted experiments on the `splice-site` dataset with 100 workers and 20 parameter servers. The results are shown in Figure 11. Here the dataset is again randomly shuffled and evenly distributed to the workers. The relative performance of different algorithms are similar to those for the other datasets.

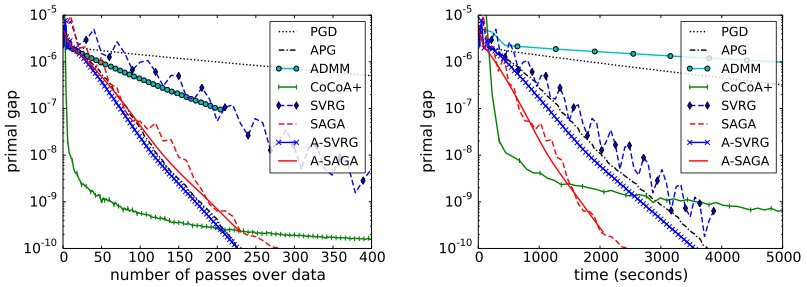


Figure 11: `splice-site`: logistic loss, $\lambda = 10^{-6}$, randomly shuffled, $m = 100$, $n = 150$, $h = 20$.

9. Conclusions and Discussions

We proposed a class of DSCOVR algorithms for asynchronous distributed optimization of large linear models with convex loss functions. They avoid dealing with delays and stale updates in an asynchronous, event-driven environment by exploiting simultaneous data and model parallelism. Compared with other first-order distributed algorithms, DSCOVR may require less amount of overall communication and computation, and especially much less or no synchronization. These conclusions are well supported by our computational experiments on a distributed computing cluster.

We note that there is still some gap between theory and practice. In our theoretical analysis, we assume that the primal and dual block indices in different iterations of DSCOVR are i.i.d. random variables, sampled sequentially with replacement. But the parallel implementations described in Section 7 impose some constraints on how they are generated. In particular, the parameter block to be updated next is randomly chosen from the set of blocks that are not being updated by any worker simultaneously, and the next worker available is event-driven, depending on the loads and processing power of different workers as well as random communication latency. These constraints violate the i.i.d. assumption, but our experiments show that they still lead to very competitive performance. Intuitively some of them can be potentially beneficial, reminiscent of the practical advantage of sampling without replacement over sampling with replacement in randomized coordinate descent methods (e.g., Shalev-Shwartz and Zhang, 2013). This is an interesting topic worth future study.

In our experiments, the parallel implementation of Nesterov's accelerated gradient method (APG) is very competitive on all the datasets we tried and for different regularization parameters used. In addition to the theoretical justifications in Arjevani and Shamir (2015) and Scaman et al. (2017), the adaptive line-search scheme turns out to be critical for its good performance in practice. The accelerated DSCOVR algorithms demonstrated comparable or better performance than APG in the experiments, but need careful tuning of the constants in their step size formula. On one hand, it supports our theoretical results that DSCOVR is capable of outperforming other first-order algorithms including APG, in terms of both communication and computation complexity (see Section 2.1). On the other hand, there are more to be done in order to realize the full potential of DSCOVR in practice. In particular, we plan to follow the ideas in Wang and Xiao (2017) to develop adaptive schemes that can automatically tune the step size parameters, as well as exploit strong convexity from data.

Acknowledgment

Adams Wei Yu is supported by the NVIDIA PhD Fellowship. The authors would like to thank Chiyuan Zhang for helpful discussion of the system implementation.

Appendix A. Proof of Theorem 1

We first prove two lemmas concerning the primal and dual proximal updates in Algorithm 2. Throughout this appendix, $\mathbf{E}_t[\cdot]$ denotes the conditional expectation taken with respect to the random indices j and l generated during the tth inner iteration in Algorithm 2, conditioned on all quantities available at the beginning of the tth iteration, including $w^{(t)}$ and $\alpha^{(t)}$. Whenever necessary, we also use the notation $j^{(t)}$ and $l^{(t)}$ to denote the random indices generated in the tth iteration.

Lemma 5 *For each $i = 1, \ldots, m$, let $u_i^{(t+1)} \in \mathbf{R}^{N_i}$ be a random variable and define*

$$\tilde{\alpha}_i^{(t+1)} = \mathbf{prox}_{\sigma_i f_i^*}(\alpha_i^{(t)} + \sigma_i u_i^{(t+1)}). \tag{58}$$

We choose an index j randomly from $\{1, \ldots, m\}$ with probability distribution $\{p_j\}_{j=1}^m$ and let

$$\alpha_i^{(t+1)} = \begin{cases} \tilde{\alpha}_i^{(t+1)} & \text{if } i = j, \\ \alpha_i^{(t)} & \text{otherwise.} \end{cases}$$

If each $u_i^{(t+1)}$ is independent of j and satisfies $\mathbf{E}_t[u_i^{(t+1)}] = X_{i:}w^{(t)}$ for $i = 1, \ldots, m$, then we have

$$\begin{aligned} & \sum_{i=1}^m \left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right) - \gamma_i\right)\|\alpha_i^{(t)} - \alpha_i^\star\|^2 \\ \geq\ & \sum_{i=1}^m \frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] + \sum_{i=1}^m \frac{1}{p_i}\left(\frac{1}{2\sigma_i} - \frac{1}{a_i}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] \\ & - \sum_{i=1}^m \frac{a_i}{4}\mathbf{E}_t\left[\|u_i^{(t+1)} - X_{i:}w^{(t)}\|^2\right] + \left\langle w^{(t)} - w^\star,\, X^T(\alpha^\star - \alpha^{(t)})\right\rangle \\ & - \sum_{i=1}^m \frac{1}{p_i}\mathbf{E}_t\left[\left\langle \alpha_i^{(t+1)} - \alpha_i^{(t)},\, X_{i:}(w^{(t)} - w^\star)\right\rangle\right], \end{aligned} \tag{59}$$

where $(w^\star, \alpha^\star)$ is the saddle point of $L(w, \alpha)$ in (5), and the a_i's are arbitrary positive numbers.

Proof First, consider a fixed index $i \in \{1, \ldots, m\}$. The definition of $\tilde{\alpha}_i^{(t+1)}$ in (58) is equivalent to

$$\tilde{\alpha}_i^{(t+1)} = \operatorname*{arg\,min}_{\beta \in \mathbf{R}^{N_i}}\left\{ f_i^*(\beta) - \langle \beta, u_i^{(t+1)}\rangle + \frac{\|\beta - \alpha_i^{(t)}\|^2}{2\sigma_i}\right\}. \tag{60}$$

By assumption, $f_i^*(\beta)$ and $\frac{1}{2\sigma_i}\|\beta - \alpha_i^{(t)}\|^2$ are strongly convex with convexity parameters γ_i and $\frac{1}{\sigma_i}$ respectively. Therefore, the objective function in (60) is $(\frac{1}{\sigma_i} + \gamma_i)$-strongly convex, which implies

$$\begin{aligned} & \frac{\|\alpha_i^\star - \alpha_i^{(t)}\|^2}{2\sigma_i} - \langle \alpha_i^\star, u_i^{(t+1)}\rangle + f_i^*(\alpha_i^\star) \\ \geq\ & \frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2}{2\sigma_i} - \langle \tilde{\alpha}_i^{(t+1)}, u_i^{(t+1)}\rangle + f_i^*(\tilde{\alpha}_i^{(t+1)}) + \left(\frac{1}{\sigma_i} + \gamma_i\right)\frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2}{2}. \end{aligned} \tag{61}$$

In addition, since $(w^\star, \alpha^\star)$ is the saddle-point of $L(w, \alpha)$, the function $f_i^*(\alpha_i) - \langle \alpha_i, X_{i:}w^\star\rangle$ is γ_i-strongly convex in α_i and attains its minimum at $\alpha_i^\star$. Thus we have

$$f_i^*(\tilde{\alpha}_i^{(t+1)}) - \langle \tilde{\alpha}_i^{(t+1)},\, X_{i:}w^\star\rangle \geq f_i^*(\alpha_i^\star) - \langle \alpha_i^\star,\, X_{i:}w^\star\rangle + \frac{\gamma_i}{2}\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2.$$

Summing up the above two inequalities gives

$$\begin{aligned} \frac{\|\alpha_i^\star - \alpha_i^{(t)}\|^2}{2\sigma_i} &\geq \frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2}{2\sigma_i} + \left(\frac{1}{2\sigma_i} + \gamma_i\right)\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2 + \left\langle \alpha_i^\star - \tilde{\alpha}_i^{(t+1)},\, u_i^{(t+1)} - X_{i:}w^\star\right\rangle \\ &= \frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2}{2\sigma_i} + \left(\frac{1}{2\sigma_i} + \gamma_i\right)\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2 + \left\langle \alpha_i^\star - \tilde{\alpha}_i^{(t+1)},\, X_{i:}(w^{(t)} - w^\star)\right\rangle \\ &\quad + \left\langle \alpha_i^\star - \alpha_i^{(t)},\, u_i^{(t+1)} - X_{i:}w^{(t)}\right\rangle + \left\langle \alpha_i^{(t)} - \tilde{\alpha}_i^{(t+1)},\, u_i^{(t+1)} - X_{i:}w^{(t)}\right\rangle \\ &\geq \frac{\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2}{2\sigma_i} + \left(\frac{1}{2\sigma_i} + \gamma_i\right)\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2 + \left\langle \alpha_i^\star - \tilde{\alpha}_i^{(t+1)},\, X_{i:}(w^{(t)} - w^\star)\right\rangle \\ &\quad + \left\langle \alpha_i^\star - \alpha_i^{(t)},\, u_i^{(t+1)} - X_{i:}w^{(t)}\right\rangle - \frac{\|\alpha_i^{(t)} - \tilde{\alpha}_i^{(t+1)}\|^2}{a_i} - \frac{a_i\|u_i^{(t+1)} - X_{i:}w^{(t)}\|^2}{4}, \end{aligned}$$

where in the last step we used Young's inequality with a_i being an arbitrary positive number. Taking conditional expectation $\mathbf{E}_t$ on both sides of the above inequality, and using the assumption $\mathbf{E}_t[u_i^{(t+1)}] = X_{i:}w^{(t)}$, we have

$$\begin{aligned} \frac{\|\alpha_i^\star - \alpha_i^{(t)}\|^2}{2\sigma_i} \geq\ & \frac{\mathbf{E}_t[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2]}{2\sigma_i} + \left(\frac{1}{2\sigma_i} + \gamma_i\right)\mathbf{E}_t[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2] + \mathbf{E}_t\left[\left\langle \alpha_i^\star - \tilde{\alpha}_i^{(t+1)}, X_{i:}(w^{(t)} - w^\star)\right\rangle\right] \\ & - \frac{\mathbf{E}_t[\|\alpha_i^{(t)} - \tilde{\alpha}_i^{(t+1)}\|^2]}{a_i} - \frac{a_i\mathbf{E}_t[\|u_i^{(t+1)} - X_{i:}w^{(t)}\|^2]}{4}. \end{aligned} \tag{62}$$

Notice that each $\tilde{\alpha}_i^{(t+1)}$ depends on the random variable $u_i^{(t+1)}$ and is independent of the random index j. But $\alpha_i^{(t+1)}$ depends on both $u_i^{(t+1)}$ and j. Using the law of total expectation,

$$\mathbf{E}_t[\,\cdot\,] \;=\; \mathbf{P}(j = i)\mathbf{E}_t[\,\cdot\,|\,j = i] + \mathbf{P}(j \neq i)\mathbf{E}_t[\,\cdot\,|\,j \neq i],$$

we obtain

$$\mathbf{E}_t[\alpha_i^{(t+1)}] = p_i\mathbf{E}_t[\tilde{\alpha}_i^{(t+1)}] + (1 - p_i)\alpha_i^{(t)}, \tag{63}$$

$$\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] = p_i\mathbf{E}_t[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2], \tag{64}$$

$$\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] = p_i\mathbf{E}_t[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^\star\|^2] + (1 - p_i)\mathbf{E}_t[\|\alpha_i^{(t)} - \alpha_i^\star\|^2]. \tag{65}$$

Next, using the equalities (63), (64) and (65), we can replace each term in (62) containing $\tilde{\alpha}_i^{(t+1)}$ with terms that contain only $\alpha_i^{(t)}$ and $\alpha_i^{(t+1)}$. By doing so and rearranging terms afterwards, we obtain

$$\begin{aligned}
&\left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)-\gamma_i\right)\|\alpha_i^{(t)}-\alpha_i^\star\|^2 \\
\geq\ & \frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2+\frac{1}{p_i}\left(\frac{1}{2\sigma_i}-\frac{1}{a_i}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{(t)}\|^2 \\
& -\frac{a_i\mathbf{E}_t[\|u_i^{(t+1)}-X_{i:}w^{(t)}\|^2]}{4}+\left\langle\alpha_i^\star-\alpha_i^{(t)},\, X_{i:}(w^{(t)}-w^\star)\right\rangle \\
& -\mathbf{E}_t\left[\left\langle\frac{1}{p_i}(\alpha_i^{(t+1)}-\alpha_i^{(t)}),\, X_{i:}(w^{(t)}-w^\star)\right\rangle\right].
\end{aligned}$$

Summing up the above inequality for $i=1,\ldots,m$ gives the desired result in (59). ■

Lemma 6 *For each $k=1,\ldots,n$, let $v_k^{(t+1)}\in\mathbf{R}^{d_i}$ be a random variable and define*

$$\tilde{w}_k^{(t+1)}=\mathbf{prox}_{\tau_k g_k}(w_k^{(t)}-\tau_k v_k^{(t+1)}).$$

We choose an index l randomly from $\{1,\ldots,n\}$ with probability distribution $\{q_l\}_{l=1}^n$ and let

$$w_k^{(t+1)}=\begin{cases}\tilde{w}_k^{(t+1)} & \text{if } k=l,\\ w_k^{(t)} & \text{otherwise.}\end{cases}$$

If each $v_k^{(t+1)}$ is independent of l and satisfies $\mathbf{E}_t[v_k^{(t+1)}]=\frac{1}{m}(X_{:k})^T\alpha^{(t)}$, then we have

$$\begin{aligned}
&\sum_{k=1}^n\left(\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)-\lambda\right)\|w_k^{(t)}-w_k^\star\|^2 \\
\geq\ & \sum_{k=1}^n\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]+\sum_{k=1}^n\frac{1}{q_k}\left(\frac{1}{2\tau_k}-\frac{1}{b_k}\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^{(t)}\|^2] \\
& -\sum_{k=1}^n\frac{b_k}{4}\mathbf{E}_t\left[\left\|v_k^{(t+1)}-\frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right]+\frac{1}{m}\left\langle X(w^{(t)}-w^\star),\,\alpha^{(t)}-\alpha^\star\right\rangle \\
& +\sum_{k=1}^n\frac{1}{q_k}\mathbf{E}_t\left[\left\langle w_k^{(t+1)}-w_k^{(t)},\,\frac{1}{m}(X_{:k})^T(\alpha^{(t)}-\alpha^\star)\right\rangle\right],
\end{aligned}\tag{66}$$

where $(w^\star,\alpha^\star)$ is the saddle point of $L(w,\alpha)$ defined in (5)*, and the b_i's are arbitrary positive numbers.*

Lemma 6 is similar to Lemma 5 and can be proved using the same techniques. Based on these two lemmas, we can prove the following proposition.

Proposition 7 *The t-th iteration within the s-th stage of Algorithm 2 guarantees*

$$\begin{aligned}
&\sum_{i=1}^m\frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)-\gamma_i+\sum_{k=1}^n\frac{3\tau_k\|X_{ik}\|^2}{mp_i}\right]\|\alpha_i^{(t)}-\alpha_i^\star\|^2+\sum_{i=1}^m\sum_{k=1}^n\frac{2\tau_k\|X_{ik}\|^2}{m^2p_i}\|\tilde{\alpha}_i^{(s)}-\alpha_i^\star\|^2 \\
&+\sum_{k=1}^n\left[\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)-\lambda+\sum_{i=1}^m\frac{3\sigma_i\|X_{ik}\|^2}{mq_k}\right]\|w_k^{(t)}-w_k^\star\|^2+\sum_{i=1}^m\sum_{k=1}^n\frac{2\sigma_i\|X_{ik}\|^2}{mq_k}\|\tilde{w}_k^{(s)}-w_k^\star\|^2 \\
\geq\ & \sum_{i=1}^m\frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]+\sum_{k=1}^n\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2].
\end{aligned}\tag{67}$$

Proof Multiplying both sides of the inequality (59) by $\frac{1}{m}$ and adding to the inequality (66) gives

$$\begin{aligned}
&\sum_{i=1}^m\frac{1}{m}\left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)-\gamma_i\right)\|\alpha_i^{(t)}-\alpha_i^\star\|^2+\sum_{k=1}^n\left(\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)-\lambda\right)\|w_k^{(t)}-w_k^\star\|^2 \\
\geq\ & \sum_{i=1}^m\frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]+\sum_{k=1}^n\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2] \\
& +\sum_{i=1}^m\frac{1}{mp_i}\left(\frac{1}{2\sigma_i}-\frac{1}{a_i}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^{(t)}\|^2]+\sum_{k=1}^n\frac{1}{q_k}\left(\frac{1}{2\tau_k}-\frac{1}{b_k}\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^{(t)}\|^2] \\
& -\sum_{k=1}^n\frac{b_k}{4}\mathbf{E}_t\left[\left\|v_k^{(t+1)}-\frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right]+\sum_{k=1}^n\frac{1}{q_k}\mathbf{E}_t\left[\left\langle w_k^{(t+1)}-w_k^{(t)},\,\frac{1}{m}(X_{:k})^T(\alpha^{(t)}-\alpha^\star)\right\rangle\right] \\
& -\sum_{i=1}^m\frac{a_i}{4m}\mathbf{E}_t[\|u_i^{(t+1)}-X_{i:}w^{(t)}\|^2]-\sum_{i=1}^m\frac{1}{mp_i}\mathbf{E}_t\left[\left\langle\alpha_i^{(t+1)}-\alpha_i^{(t)},\,X_{i:}(w^{(t)}-w^\star)\right\rangle\right].
\end{aligned}\tag{68}$$

We notice that the terms containing $\frac{1}{m}\left\langle X(w^{(t)}-w^\star),\,\alpha^{(t)}-\alpha^\star\right\rangle$ from (59) and (66) canceled each other. Next we bound the last four terms on the right-hand side of (68).

As in Algorithm 2, for each $i=1,\ldots,m$, we define a random variable

$$u_i^{(t+1)}=\bar{u}_i^{(s)}-\frac{1}{q_l}X_{il}\bar{w}_l^{(s)}+\frac{1}{q_l}X_{il}w_l^{(t)},$$

which depends on the random index $l\in\{1,\ldots,n\}$. Taking expectation with respect to l yields

$$\mathbf{E}_t[u_i^{(t+1)}]=\sum_{k=1}^n q_k\left(\bar{u}_i^{(s)}-\frac{1}{q_k}X_{ik}\bar{w}_k^{(s)}+\frac{1}{q_k}X_{ik}w_k^{(t)}\right)=X_{i:}w^{(t)},\qquad i=1,2,\ldots,m.$$

Therefore $u_i^{(t+1)}$ satisfies the assumption in Lemma 5. In order to bound its variance, we notice that

$$\sum_{k=1}^n q_k\left(\frac{1}{q_k}X_{ik}\bar{w}_k^{(s)}-\frac{1}{q_k}X_{ik}w_k^{(t)}\right)=X_{i:}\bar{w}^{(s)}-X_{i:}w^{(t)}=\bar{u}_i^{(s)}-X_{i:}w^{(t)}.$$

Using the relation between variance and the second moment, we have

$$\begin{aligned}\mathbf{E}_t\left[\|u_i^{(t+1)} - X_{i:}w^{(t)}\|^2\right] &= \sum_{k=1}^n q_k\left\|\bar{u}_i^{(s)} - \frac{1}{q_k}X_{ik}\bar{w}_k^{(s)} + \frac{1}{q_k}X_{ik}w_k^{(t)} - X_{i:}w^{(t)}\right\|^2 \\ &= \sum_{k=1}^n \frac{1}{q_k}\left\|X_{ik}\bar{w}_k^{(s)} - X_{ik}w_k^{(t)}\right\|^2 - \|\bar{u}_i^{(s)} - X_{i:}w^{(t)}\|^2 \\ &\le \sum_{k=1}^n \frac{1}{q_k}\left\|X_{ik}(\bar{w}_k^{(s)} - w_k^{(t)})\right\|^2 \\ &\le \sum_{k=1}^n \frac{2\|X_{ik}\|^2}{q_k}\left(\|\bar{w}_k^{(s)} - w_k^\star\|^2 + \|w_k^{(t)} - w_k^\star\|^2\right). \end{aligned} \tag{69}$$

Similarly, for $k = 1, \ldots, n$, we have

$$\mathbf{E}_t[v_k^{(t+1)}] = \sum_{i=1}^m p_i\left(\bar{v}_k^{(s)} - \frac{1}{p_i}\frac{1}{m}(X_{ik})^T\bar{\alpha}_i^{(s)} + \frac{1}{p_i}\frac{1}{m}(X_{ik})^T\alpha_i^{(t)}\right) = \frac{1}{m}(X_{:k})^T\alpha^{(t)}.$$

Therefore $v_k^{(t+1)}$ satisfies the assumption in Lemma 6. Furthermore, we have

$$\begin{aligned}\mathbf{E}_t\left[\left\|v_k^{(t+1)} - \frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right] &= \sum_{i=1}^m p_i\left\|\bar{v}_k^{(s)} - \frac{1}{p_i}\frac{1}{m}(X_{ik})^T\bar{\alpha}_i^{(s)} + \frac{1}{p_i}\frac{1}{m}(X_{ik})^T\alpha_i^{(t)} - \frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2 \\ &= \sum_{i=1}^m \frac{1}{p_i}\left\|\frac{1}{m}(X_{ik})^T\bar{\alpha}_i^{(s)} - \frac{1}{m}(X_{ik})^T\alpha_i^{(t)}\right\|^2 - \left\|\bar{v}_k^{(s)} - \frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2 \\ &\le \sum_{i=1}^m \frac{1}{p_i}\left\|\frac{1}{m}(X_{ik})^T\left(\bar{\alpha}_i^{(s)} - \alpha_i^{(t)}\right)\right\|^2 \\ &\le \sum_{i=1}^m \frac{2\|X_{ik}\|^2}{p_i m^2}\left(\|\bar{\alpha}_i^{(s)} - \alpha_i^\star\|^2 + \|\alpha_i^{(t)} - \alpha_i^\star\|^2\right). \end{aligned} \tag{70}$$

Now we consider the two terms containing inner products in (68). Using the conditional expectation relation (63), we have

$$\begin{aligned}\mathbf{E}_t\left[-\left\langle\alpha_i^{(t+1)} - \alpha_i^{(t)},\, X_{i:}(w^{(t)} - w^\star)\right\rangle\right] &= p_i\mathbf{E}_t\left[-\left\langle\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)},\, X_{i:}(w^{(t)} - w^\star)\right\rangle\right] \\ &\ge p_i\mathbf{E}_t\left[-\frac{1}{c_i}\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2 - \frac{c_i}{4}\|X_{i:}(w^{(t)} - w^\star)\|^2\right] \\ &= -\frac{p_i}{c_i}\mathbf{E}_t[\|\tilde{\alpha}_i^{(t+1)} - \alpha_i^{(t)}\|^2] - \frac{c_i p_i}{4}\|X_{i:}(w^{(t)} - w^\star)\|^2 \\ &= -\frac{1}{c_i}\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] - \frac{c_i p_i}{4}\|X_{i:}(w^{(t)} - w^\star)\|^2, \end{aligned} \tag{71}$$

where we used Young's inequality with c_i being an arbitrary positive number, and the last equality used (64). We note that for any n vectors $z_1, \ldots, z_n \in \mathbf{R}^{N_i}$, it holds that

$$\left\|\sum_{k=1}^n z_k\right\|^2 \le \sum_{k=1}^n \frac{1}{q_k}\|z_k\|^2.$$

To see this, we let $z_{k,j}$ denote the jth component of z_k and use the Cauchy-Schwarz inequality:

$$\begin{aligned}\left\|\sum_{k=1}^n z_k\right\|^2 &= \sum_{j=1}^{N_i}\left(\sum_{k=1}^n z_{k,j}\right)^2 = \sum_{j=1}^{N_i}\left(\sum_{k=1}^n \frac{z_{k,j}}{\sqrt{q_k}}\sqrt{q_k}\right)^2 \\ &\le \sum_{j=1}^{N_i}\left(\sum_{k=1}^n\left(\frac{z_{k,j}}{\sqrt{q_k}}\right)^2\right)\left(\sum_{k=1}^n(\sqrt{q_k})^2\right) \\ &= \sum_{j=1}^{N_i}\left(\sum_{k=1}^n \frac{z_{k,j}^2}{q_k}\right) = \sum_{k=1}^n \frac{1}{q_k}\sum_{j=1}^{N_i} z_{k,j}^2 = \sum_{k=1}^n \frac{1}{q_k}\|z_k\|^2. \end{aligned}$$

Applying this inequality to the vector $X_{i:}(w^{(t)} - w^\star) = \sum_{k=1}^n X_{ik}(w_k^{(t)} - w_k^\star)$, we get

$$\|X_{i:}(w^{(t)} - w^\star)\|^2 \le \sum_{k=1}^n \frac{1}{q_k}\|X_{ik}(w_k^{(t)} - w_k^\star)\|^2.$$

Therefore we can continue the inequality (71), for each $i = 1, \ldots, m$, as

$$\begin{aligned}&\mathbf{E}_t\left[-\left\langle\alpha_i^{(t+1)} - \alpha_i^{(t)},\, X_{i:}(w^{(t)} - w^\star)\right\rangle\right] \\ \ge\ & -\frac{1}{c_i}\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] - \frac{c_i p_i}{4}\sum_{k=1}^n \frac{1}{q_k}\|X_{ik}(w_k^{(t)} - w_k^\star)\|^2 \\ \ge\ & -\frac{1}{c_i}\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] - \frac{c_i p_i}{4}\sum_{k=1}^n \frac{1}{q_k}\|X_{ik}\|^2\|w_k^{(t)} - w_k^\star\|^2. \end{aligned} \tag{72}$$

Using similarly arguments, we can obtain, for each $k = 1, \ldots, n$ and arbitrary $h_k > 0$,

$$\begin{aligned}&\mathbf{E}_t\left[\left\langle w_k^{(t+1)} - w_k^{(t)},\, \frac{1}{m}(X_{:k})^T(\alpha^{(t)} - \alpha^\star)\right\rangle\right] \\ \ge\ & -\frac{1}{h_k}\mathbf{E}_t[\|w_k^{(t+1)} - w_k^{(t)}\|^2] - \frac{h_k q_k}{4m^2}\sum_{i=1}^m \frac{1}{p_i}\|X_{ik}\|^2\|\alpha_i^{(t)} - \alpha_i^\star\|^2. \end{aligned} \tag{73}$$

Applying the bounds in (69), (70), (72) and (73) to (68) and rearranging terms, we have

$$\begin{aligned}&\sum_{i=1}^m \frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right) - \gamma_i + \sum_{k=1}^n \frac{b_k\|X_{ik}\|^2}{2mp_i} + \sum_{k=1}^n \frac{h_k\|X_{ik}\|^2}{4mp_i}\right]\|\alpha_i^{(t)} - \alpha_i^\star\|^2 \\ &+ \sum_{k=1}^n\left[\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right) - \lambda + \sum_{i=1}^m \frac{a_i\|X_{ik}\|^2}{2mq_k} + \sum_{i=1}^m \frac{c_i\|X_{ik}\|^2}{4mq_k}\right]\|w_k^{(t)} - w_k^\star\|^2 \\ &+ \sum_{i=1}^m\sum_{k=1}^n \frac{b_k\|X_{ik}\|^2}{2m^2p_i}\|\bar{\alpha}_i^{(s)} - \alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{a_i\|X_{ik}\|^2}{2mq_k}\|\bar{w}_k^{(s)} - w_k^\star\|^2 \\ \ge\ & \sum_{i=1}^m \frac{1}{mp_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] + \sum_{k=1}^n \frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)\mathbf{E}_t[\|w_k^{(t+1)} - w_k^\star\|^2] \\ &+ \sum_{i=1}^m \frac{1}{mp_i}\left(\frac{1}{2\sigma_i} - \frac{1}{a_i} - \frac{1}{c_i}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] \\ &+ \sum_{k=1}^n \frac{1}{q_k}\left(\frac{1}{2\tau_k} - \frac{1}{b_k} - \frac{1}{h_k}\right)\mathbf{E}_t[\|w_k^{(t+1)} - w_k^{(t)}\|^2]. \end{aligned}$$

The desired result (67) is obtained by choosing $a_i = c_i = 4\sigma_i$ and $b_k = h_k = 4\tau_k$. ■

Finally, we are ready to prove Theorem 1. Let $\theta \in (0,1)$ be a parameter to be determined later, and let Γ and η be two constants such that

$$\Gamma \geq \max_{i,k}\left\{\frac{1}{p_i}\left(1+\frac{3\|X_{ik}\|^2}{2\theta q_k\lambda\gamma_i}\right),\ \frac{1}{q_k}\left(1+\frac{3n\|X_{ik}\|^2}{2\theta m p_i\lambda\gamma_i}\right)\right\}, \tag{74}$$

$$\eta = 1-\frac{1-\theta}{\Gamma}. \tag{75}$$

It is easy to check that $\Gamma > 1$ and $\eta \in (0,1)$. By the choices of σ_i and τ_k in (23) and (24) respectively, we have

$$\frac{1}{p_i}\left(1+\frac{1}{2\sigma_i\gamma_i}\right) = \frac{1}{q_k}\left(1+\frac{1}{2\tau_k\lambda}\right) = \Gamma, \tag{76}$$

for all $i = 1,\ldots,m$ and $k = 1,\ldots,n$. Comparing the above equality with the definition of Γ in (74), we have

$$\frac{3\|X_{ik}\|^2}{2\theta q_k\lambda\gamma_i} \leq \frac{1}{2\sigma_i\gamma_i} \qquad \text{and} \qquad \frac{3n\|X_{ik}\|^2}{2\theta m p_i\lambda\gamma_i} \leq \frac{1}{2\tau_k\lambda},$$

which implies

$$\frac{3\sigma_i\|X_{ik}\|^2}{q_k} \leq \theta\lambda \qquad \text{and} \qquad \frac{3n\tau_k\|X_{ik}\|^2}{mp_i} \leq \theta\gamma_i,$$

for all $i = 1,\ldots,m$ and $k = 1,\ldots,n$. Therefore, we have

$$\sum_{k=1}^n \frac{3\tau_k\|X_{ik}\|^2}{mp_i} = \frac{1}{n}\sum_{k=1}^n \frac{3n\tau_k\|X_{ik}\|^2}{mp_i} \leq \frac{1}{n}\sum_{k=1}^n \theta\gamma_i = \theta\gamma_i, \qquad i = 1,\ldots,m, \tag{77}$$

$$\sum_{i=1}^m \frac{3\sigma_i\|X_{ik}\|^2}{mq_k} = \frac{1}{m}\sum_{i=1}^m \frac{3\sigma_i\|X_{ik}\|^2}{q_k} \leq \frac{1}{m}\sum_{i=1}^m \theta\lambda = \theta\lambda, \qquad k = 1,\ldots,n. \tag{78}$$

Now we consider the inequality (67), and examine the ratio between the coefficients of $\|\alpha_i^{(t)}-\alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]$. Using (77) and (76), we have

$$\frac{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)-\gamma_i+\sum_{k=1}^n\frac{3\tau_k\|X_{ik}\|^2}{mp_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)} \leq 1-\frac{(1-\theta)\gamma_i}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)} = 1-\frac{1-\theta}{\Gamma} = \eta. \tag{79}$$

Similarly, the ratio between the coefficients of $\|w_k^{(t)}-w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]$ can be bounded using (78) and (76):

$$\frac{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)-\lambda+\sum_{i=1}^m\frac{3\sigma_i\|X_{ik}\|^2}{mq_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)} \leq 1-\frac{(1-\theta)\lambda}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)} = 1-\frac{1-\theta}{\Gamma} = \eta. \tag{80}$$

In addition, the ratio between the coefficients of $\|\bar\alpha_i^{(s)}-\alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]$ and that of $\|\bar w_k^{(s)}-w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]$ can be bounded as

$$\frac{\sum_{k=1}^k\frac{2\tau_k\|X_{ik}\|^2}{mp_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)} \leq \frac{(2/3)\theta\gamma_i}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)} = \frac{(2/3)\theta}{\Gamma} = \frac{2\theta(1-\eta)}{3(1-\theta)}, \tag{81}$$

$$\frac{\sum_{i=1}^m\frac{3\sigma_i\|X_{ik}\|^2}{mq_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)} \leq \frac{(2/3)\theta\lambda}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)} = \frac{(2/3)\theta}{\Gamma} = \frac{2\theta(1-\eta)}{3(1-\theta)}. \tag{82}$$

Using (76) and the four inequalities (79), (80), (81) and (82), we conclude that the inequality (67) implies

$$\begin{aligned}
&\eta\sum_{i=1}^m\frac{\Gamma\gamma_i}{m}\|\alpha_i^{(t)}-\alpha_i^\star\|^2+\frac{2\theta(1-\eta)}{3(1-\theta)}\sum_{i=1}^m\frac{\Gamma\gamma_i}{m}\|\bar\alpha_i^{(s)}-\alpha_i^\star\|^2\\
&+\eta\sum_{k=1}^n\Gamma\lambda\|w_k^{(t)}-w_k^\star\|^2+\frac{2\theta(1-\eta)}{3(1-\theta)}\sum_{k=1}^n\Gamma\lambda\|\bar w_k^{(s)}-w_k^\star\|^2\\
\geq\ &\sum_{i=1}^m\frac{\Gamma\gamma_i}{m}\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]+\sum_{k=1}^n\Gamma\lambda\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2].
\end{aligned}$$

Using the definite of $\Omega(\cdot)$ in (21), the inequality above is equivalent to

$$\begin{aligned}
&\eta\,\Omega(w^{(t)}-w^\star,\alpha^{(t)}-\alpha^\star)+\frac{2\theta(1-\eta)}{3(1-\theta)}\Omega(\bar w^{(s)}-w^\star,\bar\alpha^{(s)}-\alpha^\star)\\
\geq\ &\mathbf{E}_t\left[\Omega(w^{(t+1)}-w^\star,\alpha^{(t+1)}-\alpha^\star)\right].
\end{aligned} \tag{83}$$

To simplify further derivation, we define

$$\begin{aligned}
\Delta^{(t)} &= \mathbf{E}\left[\Omega(w^{(t)}-w^\star,\alpha^{(t)}-\alpha^\star)\right],\\
\bar\Delta^{(s)} &= \mathbf{E}\left[\Omega(\bar w^{(s)}-w^\star,\bar\alpha^{(s)}-\alpha^\star)\right],
\end{aligned}$$

where the expectation is taken with respect to all randomness in the sth stage, that is, the random variables $\{(j^{(0)},l^{(0)}),(j^{(1)},l^{(1)}),\ldots,(j^{(M-1)},l^{(M-1)})\}$. Then the inequality (83) implies

$$\frac{2\theta(1-\eta)}{3(1-\theta)}\bar\Delta^{(s)}+\eta\Delta^{(t)} \geq \Delta^{(t+1)}.$$

Dividing both sides of the above inequality by η^{t+1} gives

$$\frac{2\theta(1-\eta)}{3(1-\theta)}\frac{\bar\Delta^{(s)}}{\eta^{t+1}}+\frac{\Delta^{(t)}}{\eta^t} \geq \frac{\Delta^{(t+1)}}{\eta^{t+1}}.$$

Summing for $t = 0,1,,\ldots,M-1$ gives

$$\left(\frac{1}{\eta}+\frac{1}{\eta^2}+\cdots+\frac{1}{\eta^M}\right)\frac{2\theta(1-\eta)}{3(1-\theta)}\bar\Delta^{(s)}+\Delta^{(0)} \geq \frac{\Delta^{(T)}}{\eta^M},$$

which further leads to

$$(1-\eta^M)\frac{2\theta}{3(1-\theta)}\bar{\Delta}^{(s)} + \eta^M \Delta^{(0)} \geq \Delta^{(M)}.$$

Now choosing $\theta = 1/3$ and using the relation $\bar{\Delta}^{(s)} = \Delta^{(0)}$ for each stage, we obtain

$$\left(\frac{1}{3} + \frac{2}{3}\eta^M\right)\Delta^{(0)} \geq \Delta^{(M)}.$$

Therefore if we choose M large enough such that $\eta^M \leq \frac{1}{2}$, then

$$\Delta^{(M)} \leq \frac{2}{3}\Delta^{(0)}, \quad \text{or equivalently,} \quad \bar{\Delta}^{(s+1)} \leq \frac{2}{3}\bar{\Delta}^{(s)}.$$

The condition $\eta^M \leq \frac{1}{2}$ is equivalent to $M \geq \frac{\log(2)}{\log(1/\eta)}$, which can be guaranteed by

$$M \geq \frac{\log(2)}{1-\eta} = \frac{\log(2)}{1-\theta}\Gamma = \frac{3\log(2)}{2}\Gamma = \log(\sqrt{8})\Gamma.$$

To further simplify, it suffices to have $M \geq \log(3)\Gamma$. Finally, we notice that $\bar{\Delta}^{(s+1)} \leq (2/3)\bar{\Delta}^{(s)}$ implies $\bar{\Delta}^{(s)} \leq (2/3)^s\, \bar{\Delta}^{(0)}$, which is the desired result in Theorem 1.

A.1. Alternative bounds and step sizes

Alternatively, we can let Γ to satisfy

$$\Gamma \geq \max_{i,k}\left\{\frac{1}{p_i}\left(1 + \frac{3\|X_{:k}\|_F^2}{2\theta m q_k \lambda \gamma_i}\right),\ \frac{1}{q_k}\left(1 + \frac{3\|X_{i:}\|_F^2}{2\theta m p_i \lambda \gamma_i}\right)\right\}, \tag{84}$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. Then by choosing σ_i and τ_k that satisfy (76), we have

$$\frac{3\sigma_i\|X_{:k}\|_F^2}{mq_k} \leq \theta\lambda \qquad \text{and} \qquad \frac{3\tau_k\|X_{i:}\|_F^2}{mp_i} \leq \theta\gamma_i,$$

We can bound the left-hand sides in (77) and (78) using Hölder's inequality, which results in

$$\sum_{k=1}^n \frac{3\tau_k\|X_{ik}\|^2}{mp_i} \leq \sum_{k=1}^n \frac{3\tau_k\|X_{ik}\|_F^2}{mp_i} \leq \frac{3\max_k\{\tau_k\}\|X_{i:}\|_F^2}{mp_i} \leq \theta\gamma_i, \qquad i = 1,\ldots,m, \tag{85}$$

$$\sum_{i=1}^m \frac{3\sigma_i\|X_{ik}\|^2}{mq_k} \leq \sum_{i=1}^m \frac{3\sigma_i\|X_{ik}\|_F^2}{mq_k} \leq \frac{3\max_i\{\sigma_i\}\|X_{:k}\|_F^2}{mq_k} \leq \theta\lambda, \qquad k = 1,\ldots,n. \tag{86}$$

The rest of the proof hold without any change. Setting $\theta = 1/3$ gives the condition on Γ in (26).

In Theorem 1 and the proof above, we choose Γ as a uniform bound over all combinations of (i,k) in order to obtain a uniform convergence rates on all blocks of the primal and dual variables $w_k^{(t)}$ and $\alpha_i^{(t)}$, so we have a simple conclusion as in (25). In practice, we can use different bounds on different blocks and choose step sizes to allow them to converge at different rates.

For example, we can choose the step sizes σ_i and τ_k such that

$$\begin{aligned}
\frac{1}{p_i}\left(1 + \frac{1}{2\sigma_i\gamma_i}\right) &= \max_k\left\{\frac{1}{p_i}\left(1 + \frac{3\|X_{:k}\|_F^2}{2\theta m q_k \lambda\gamma_i}\right)\right\}, \qquad i = 1,\ldots,m,\\
\frac{1}{q_k}\left(1 + \frac{1}{2\tau_k\lambda}\right) &= \max_i\left\{\frac{1}{q_k}\left(1 + \frac{3\|X_{i:}\|_F^2}{2\theta m p_i \lambda\gamma_i}\right)\right\}, \qquad k = 1,\ldots,n.
\end{aligned}$$

Then the inequalities (85) and (86) still hold, and we can still show linear convergence with a similar rate. In this case, the step sizes are chosen as

$$\begin{aligned}
\sigma_i &= \min_k\left\{\frac{\theta m q_k \lambda}{3\|X_{:k}\|_F^2}\right\}, \qquad i = 1,\ldots,m,\\
\tau_k &= \min_i\left\{\frac{\theta m p_i \gamma_i}{3\|X_{i:}\|_F^2}\right\}, \qquad k = 1,\ldots,n.
\end{aligned}$$

If we choose the probabilities to be proportional to the norms of the data blocks, i.e.,

$$p_i = \frac{\|X_{i:}\|_F^2}{\|X\|_F^2}, \qquad q_k = \frac{\|X_{:k}\|_F^2}{\|X\|_F^2},$$

then we have

$$\sigma_i = \frac{\theta m\lambda}{3\|X\|_F^2}, \qquad \tau_k = \frac{\theta m\gamma_i}{3\|X\|_F^2}.$$

If we further normalize the rows of X, and let R be the norm of each row, then (with $\theta = 1/3$)

$$\sigma_i = \frac{\theta\lambda}{3R^2}\frac{m}{N} = \frac{\lambda}{9R^2}\frac{m}{N}, \qquad \tau_k = \frac{\theta\gamma_i}{3R^2}\frac{m}{N} = \frac{\gamma_i}{9R^2}\frac{m}{N}.$$

For distributed ERM, we have $\gamma_i = \frac{N}{m}\gamma$, thus $\tau_k = \frac{\gamma}{9R^2}$ as in (29).

Appendix B. Proof of Theorem 2

Consider the following saddle-point problem with doubly separable structure:

$$\min_{w\in\mathbf{R}^D}\max_{\alpha\in\mathbf{R}^N}\left\{L(w,\alpha) \equiv \frac{1}{m}\sum_{i=1}^m\sum_{k=1}^n \alpha_i^T X_{ik} w_k - \frac{1}{m}\sum_{i=1}^m f_i^*(\alpha_i) + \sum_{k=1}^n g_k(w_k)\right\}. \tag{87}$$

Under Assumption 1, L has a unique saddle point $(w^\star, \alpha^\star)$. We define

$$\widetilde{P}_k(w_k) \equiv \frac{1}{m}(\alpha^\star)^T X_{:k} w_k + g_k(w_k) - \frac{1}{m}(\alpha^\star)^T X_{:k} w_k^\star - g_k(w_k^\star), \qquad k = 1,\ldots,n, \tag{88}$$

$$\widetilde{D}_i(\alpha_i) \equiv \frac{1}{m}\left(\alpha_i^T X_{i:} w^\star - f_i^*(\alpha_i) - (\alpha_i^\star)^T X_{i:} w^\star + f_i^*(\alpha_i^\star)\right), \qquad i = 1,\ldots,m. \tag{89}$$

We note that $w_k^\star$ is the minimizer of $\widetilde{P}_k$ with $\widetilde{P}_k(w_k^\star) = 0$ and $\alpha_i^\star$ is the maximizer of $\widetilde{D}_i$ with $\widetilde{D}_i(\alpha_i^\star) = 0$. Moreover, by the assumed strong convexity,

$$\widetilde{P}_k(w_k) \geq \frac{\lambda}{2}\|w_k - w_k^\star\|^2, \qquad \widetilde{D}_i(\alpha_i) \leq -\frac{\gamma_i}{2m}\|\alpha_i - \alpha_i^\star\|^2. \tag{90}$$

Moreover, we have the following lower bound for the duality gap $P(w) - D(\alpha)$:

$$\sum_{k=1}^{n} \widetilde{P}_k(w_k) - \sum_{i=1}^{m} \widetilde{D}_i(\alpha_i) \;=\; L(w, \alpha^\star) - L(w^\star, \alpha) \;\le\; P(w) - D(\alpha). \tag{91}$$

We can also use them to derive an upper bound for the duality gap, as in the following lemma.

Lemma 8 *Suppose Assumption 1 holds. Let $(w^\star, \alpha^\star)$ be the saddle-point of $L(w, \alpha)$ and define*

$$P(w) = \sup_{\alpha} L(w, \alpha), \qquad D(\alpha) = \inf_{w} L(w, \alpha).$$

Then we have

$$P(w) - D(\alpha) \;\le\; L(w, \alpha^\star) - L(w^\star, \alpha) + \Big(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\Big)\|w - w^\star\|^2 + \frac{\|X\|^2}{2m^2\lambda}\|\alpha - \alpha^\star\|^2.$$

Proof By definition, the primal function can be written as $P(w) = F(w) + g(w)$, where

$$F(w) = \frac{1}{m}\sum_{i=1}^{m} f_i(X_{i:}w) = \frac{1}{m}\max_{\alpha}\Big\{\alpha^T X w - \sum_{i=1}^{m} f_i^*(\alpha_i)\Big\}.$$

From the optimality conditions satisfied by the saddle point $(w^\star, \alpha^\star)$, we have

$$\nabla F(w^\star) = \frac{1}{m} X^T \alpha^\star.$$

By assumption, $\nabla F(w)$ is Lipschitz continuous with smooth constant $\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{\gamma_i}$, which implies

$$\begin{aligned}
F(w) \;&\le\; F(w^\star) + \langle \nabla F(w^\star), w - w^\star\rangle + \Big(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\Big)\|w - w^\star\|^2 \\
&=\; \frac{1}{m}\Big((\alpha^\star)^T X w^\star - \sum_{i=1}^{m} f_i^*(\alpha_i^\star)\Big) + \frac{1}{m}(\alpha^\star)^T X (w - w^\star) + \Big(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\Big)\|w - w^\star\|^2 \\
&=\; \frac{1}{m}\Big((\alpha^\star)^T X w - \sum_{i=1}^{m} f_i^*(\alpha_i^\star)\Big) + \Big(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\Big)\|w - w^\star\|^2.
\end{aligned}$$

Therefore,

$$\begin{aligned}
P(w) \;&=\; F(w) + g(w) \\
&\le\; \frac{1}{m}(\alpha^\star)^T X w - \frac{1}{m}\sum_{i=1}^{m} f_i^*(\alpha_i^\star) + g(w) + \Big(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\Big)\|w - w^\star\|^2 \\
&=\; L(w, \alpha^\star) + \Big(\frac{1}{m}\sum_{i=1}^{m} \frac{\|X_{i:}\|^2}{2\gamma_i}\Big)\|w - w^\star\|^2.
\end{aligned}$$

Using similar arguments, especially that $\nabla g^*(\alpha)$ has Lipschitz constant $\frac{\|X\|}{m^2\lambda}$, we can show that

$$D(\alpha) \;\ge\; L(w^\star, \alpha) - \frac{\|X\|^2}{2m^2\lambda}\|\alpha - \alpha^\star\|^2.$$

Combining the last two inequalities gives the desired result. ■

The rest of the proof follow similar steps as in the proof of Theorem 1. The next two lemmas are variants of Lemmas 5 and 6.

Lemma 9 *Under the same assumptions and setup in Lemma 5, we have*

$$\begin{aligned}
&\sum_{i=1}^{m}\left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \frac{\gamma_i}{2}\right) - \frac{\gamma_i}{2}\right)\|\alpha_i^{(t)} - \alpha_i^\star\|^2 - \sum_{i=1}^{m}\left(\frac{1}{p_i} - 1\right) m\widetilde{D}_i(\alpha_i^{(t)}) \\
\ge\;& \sum_{i=1}^{m}\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \frac{\gamma_i}{2}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] + \sum_{i=1}^{m}\frac{1}{2p_i\sigma_i}\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] - \sum_{i=1}^{m}\frac{m}{p_i}\mathbf{E}_t\,[\widetilde{D}_i(\alpha_i^{(t+1)})] \\
&+ \left\langle w^{(t)} - w^\star,\, X^T(\alpha^\star - \alpha^{(t)})\right\rangle - \sum_{i=1}^{m}\frac{1}{p_i}\mathbf{E}_t\left[\left\langle \alpha_i^{(t+1)} - \alpha_i^{(t)},\, u_i^{(t+1)} - X_{i:}w^\star\right\rangle\right].
\end{aligned} \tag{92}$$

Proof We start by taking conditional expectation $\mathbf{E}_t$ on both sides of the inequality (61), and would like to replace every term containing $\tilde{\alpha}_i^{(t+1)}$ with terms that contain only $\alpha_i^{(t)}$ and $\alpha_i^{(t+1)}$. In addition to the relations in (63), (64) and (65), we also need

$$\mathbf{E}_t\,[f_i^*(\alpha_i^{(t+1)})] \;=\; p_i f_i^*(\tilde{\alpha}_i^{(t+1)}) + (1 - p_i) f_i^*(\alpha_i^{(t)}).$$

After the substitutions and rearranging terms, we have

$$\begin{aligned}
&\left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \frac{\gamma_i}{2}\right) - \frac{\gamma_i}{2}\right)\|\alpha_i^{(t)} - \alpha_i^\star\|^2 + \left(\frac{1}{p_i} - 1\right)(f_i^*(\alpha_i^{(t)}) - f_i^*(\alpha_i^\star)) \\
\ge\;& \frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \frac{\gamma_i}{2}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] + \frac{1}{2p_i\sigma_i}\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] + \frac{1}{p_i}\mathbf{E}_t[(f_i^*(\alpha_i^{(t+1)}) - f_i^*(\alpha_i^\star))] \\
&\mathbf{E}_t\,[\langle \alpha_i^\star - \alpha_i^{(t)},\, u_i^{(t+1)}\rangle] - \frac{1}{p_i}\mathbf{E}_t\left[\left\langle \alpha_i^{(t+1)} - \alpha_i^{(t)},\, u_i^{(t+1)}\right\rangle\right].
\end{aligned}$$

Next, we use the assumption $\mathbf{E}_t\,[u_i^{(t+1)}] = X_{i:}w^{(t)}$ and the definition of $\widetilde{D}_i(\cdot)$ in (89) to obtain

$$\begin{aligned}
&\left(\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \frac{\gamma_i}{2}\right) - \frac{\gamma_i}{2}\right)\|\alpha_i^{(t)} - \alpha_i^\star\|^2 - \left(\frac{1}{p_i} - 1\right) m\widetilde{D}_i(\alpha_i^{(t)}) \\
\ge\;& \frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \frac{\gamma_i}{2}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] + \frac{1}{2p_i\sigma_i}\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^{(t)}\|^2] - \frac{m}{p_i}\mathbf{E}_t\,[\widetilde{D}_i(\alpha_i^{(t+1)})] \\
&+ \left\langle \alpha_i^\star - \alpha_i^{(t)},\, X_{i:}(w^{(t)} - w^\star)\right\rangle - \frac{1}{p_i}\mathbf{E}_t\left[\left\langle \alpha_i^{(t+1)} - \alpha_i^{(t)},\, u_i^{(t+1)} - X_{i:}w^\star\right\rangle\right].
\end{aligned}$$

Summing up the above inequality for $i = 1, \ldots, m$ gives the desired result (92). ■

Lemma 10 *Under the same assumptions and setup in Lemma 6, we have*

$$\begin{aligned}
&\sum_{k=1}^{K}\left(\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \frac{\lambda}{2}\right) - \frac{\lambda}{2}\right)\|w_k^{(t)} - w_k^\star\|^2 + \sum_{k=1}^{n}\left(\frac{1}{q_k} - 1\right)\widetilde{P}_k(w_k^{(t)}) \\
\ge\;& \sum_{k=1}^{n}\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \frac{\lambda}{2}\right)\mathbf{E}_t[\|w_k^{(t+1)} - w_k^\star\|^2] + \sum_{k=1}^{n}\frac{1}{2q_k\tau_k}\mathbf{E}_t[\|w_k^{(t+1)} - w_k^{(t)}\|^2] + \sum_{k=1}^{n}\frac{1}{q_k}\mathbf{E}_t[\widetilde{P}_k(w_k^{(t+1)})] \\
&+ \frac{1}{m}\left\langle X(w^{(t)} - w^\star),\, \alpha^{(t)} - \alpha^\star\right\rangle + \sum_{k=1}^{n}\frac{1}{q_k}\mathbf{E}_t\left[\left\langle w_k^{(t+1)} - w_k^{(t)},\, v_k^{(t+1)} - \frac{1}{m}(X_{:k})^T\alpha^\star\right\rangle\right].
\end{aligned}$$

Based on Lemma 9 and Lemma 10, we can prove the following proposition. The proof is very similar to that of Proposition 7, thus we omit the details here.

Proposition 11 *The t-th iteration within the s-th stage of Algorithm 2 guarantees*

$$\begin{aligned}
&\sum_{k=1}^n \left(\frac{1}{q_k}-1\right)\widetilde{P}_k(w_k^{(t)}) - \sum_{i=1}^m \left(\frac{1}{p_i}-1\right)\widetilde{D}_i(\alpha_i^{(t)}) \\
&+\sum_{i=1}^m \frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)-\frac{\gamma_i}{2}+\sum_{k=1}^n \frac{3\tau_k\|X_{ik}\|^2}{mp_i}\right]\|\alpha_i^{(t)}-\alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{2\tau_k\|X_{ik}\|^2}{m^2p_i}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2 \\
&+\sum_{k=1}^n \left[\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)-\frac{\lambda}{2}+\sum_{i=1}^m \frac{3\sigma_i\|X_{ik}\|^2}{mq_k}\right]\|w_k^{(t)}-w_k^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{2\sigma_i\|X_{ik}\|^2}{mq_k}\|\bar{w}_k^{(s)}-w_k^\star\|^2 \\
\geq\ &\sum_{k=1}^n \frac{1}{q_k}\mathbf{E}_t[\widetilde{P}_k(w_k^{(t+1)})] - \sum_{i=1}^m \frac{1}{p_i}\mathbf{E}_t[\widetilde{D}_i(\alpha_i^{(t+1)})] \\
&+\sum_{i=1}^m \frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2] + \sum_{k=1}^n \frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]. \qquad (93)
\end{aligned}$$

Now we proceed to prove Theorem 2. Let $\theta \in (0,1)$ be a parameter to be determined later, and let Γ and η be two constants such that

$$\Gamma \geq \max_{i,k}\left\{\frac{1}{p_i}\left(1+\frac{6\Lambda}{\theta q_k\lambda\gamma_i}\right),\ \frac{1}{q_k}\left(1+\frac{6n\Lambda}{\theta p_i m\lambda\gamma_i}\right)\right\}, \qquad (94)$$

$$\eta = 1-\frac{1-\theta}{\Gamma}. \qquad (95)$$

It is easy to check that $\Gamma > 1$ and $\eta \in (0,1)$. The choices of σ_i and τ_k in (32) and (33) satisfy

$$\frac{1}{p_i}\left(\frac{1}{2}+\frac{1}{2\sigma_i\gamma_i}\right) = \frac{\Gamma}{2}, \qquad i=1,\ldots,m, \qquad (96)$$

$$\frac{1}{q_k}\left(\frac{1}{2}+\frac{1}{2\tau_k\lambda}\right) = \frac{\Gamma}{2}, \qquad k=1,\ldots,n. \qquad (97)$$

Comparing them with the definition of Γ in (94), and using the assumption $\Lambda \geq \|X_{ik}\|_F^2 \geq \|X_{ik}\|^2$, we get

$$\frac{6\|X_{ik}\|^2}{\theta q_k\lambda\gamma_i} \leq \frac{6\Lambda}{\theta q_k\lambda\gamma_i} \leq \frac{1}{\sigma_i\gamma_i} \qquad \text{and} \qquad \frac{6n\|X_{ik}\|^2}{\theta p_i m\lambda\gamma_i} \leq \frac{6n\Lambda}{\theta p_i m\lambda\gamma_i} \leq \frac{1}{\tau_k\lambda},$$

which implies

$$\frac{3\sigma_i\|X_{ik}\|^2}{q_k} \leq \theta\frac{\lambda}{2} \qquad \text{and} \qquad \frac{3n\tau_k\|X_{ik}\|^2}{mp_i} \leq \theta\frac{\gamma_i}{2},$$

for all $i=1,\ldots,m$ and $k=1,\ldots,n$. Therefore, we have

$$\sum_{k=1}^n \frac{3\tau_k\|X_{ik}\|^2}{mp_i} = \frac{1}{n}\sum_{k=1}^n \frac{3n\tau_k\|X_{ik}\|^2}{mp_i} \leq \theta\frac{\gamma_i}{2}, \qquad i=1,\ldots,m, \qquad (98)$$

$$\sum_{i=1}^m \frac{3\sigma_i\|X_{ik}\|^2}{mq_k} = \frac{1}{m}\sum_{i=1}^m \frac{3\sigma_i\|X_{ik}\|^2}{q_k} \leq \theta\frac{\lambda}{2}, \qquad k=1,\ldots,n. \qquad (99)$$

Now we consider the inequality (93), and examine the ratio between the coefficients of $\|\alpha_i^{(t)}-\alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]$. Using (98) and (96), we have

$$\frac{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)-\frac{\gamma_i}{2}+\sum_{k=1}^n \frac{3\tau_k\|X_{ik}\|^2}{mp_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)} \leq 1-\frac{(1-\theta)\frac{\gamma_i}{2}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)} = 1-\frac{1-\theta}{\Gamma} = \eta. \qquad (100)$$

Similarly, the ratio between the coefficients of $\|w_k^{(t)}-w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]$ can be bounded using (99) and (97):

$$\frac{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)-\frac{\lambda}{2}+\sum_{i=1}^m \frac{3\sigma_i\|X_{ik}\|^2}{mq_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)} \leq 1-\frac{(1-\theta)\frac{\lambda}{2}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)} = 1-\frac{1-\theta}{\Gamma} = \eta. \qquad (101)$$

In addition, the ratio between the coefficients of $\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2]$ and that of $\|\bar{w}_k^{(s)}-w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2]$ can be bounded as

$$\frac{\sum_{k=1}^n \frac{2\tau_k\|X_{ik}\|^2}{mp_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)} \leq \frac{(2/3)\theta\frac{\gamma_i}{2}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\frac{\gamma_i}{2}\right)} = \frac{(2/3)\theta}{\Gamma} = \frac{2\theta(1-\eta)}{3(1-\theta)}, \qquad (102)$$

$$\frac{\sum_{i=1}^m \frac{2\sigma_i\|X_{ik}\|^2}{mq_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)} \leq \frac{(2/3)\theta\frac{\lambda}{2}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\frac{\lambda}{2}\right)} = \frac{(2/3)\theta}{\Gamma} = \frac{2\theta(1-\eta)}{3(1-\theta)}. \qquad (103)$$

Also, the ratios between the coefficients of $\widetilde{P}_k(w_k^{(t)})$ and $\mathbf{E}_t[\widetilde{P}_k(w_k^{(t+1)})]$ is $1-q_k$, and that of $\widetilde{D}_k(\alpha_i^{(t)})$ and $\mathbf{E}_t[\widetilde{D}_i(\alpha_i^{(t+1)})]$ is $1-p_i$. From the definition of Γ and η in (94) and (95), we have

$$1-p_i \leq \eta \quad \text{for} \quad i=1,\ldots,m, \quad \text{and} \quad 1-q_k \leq \eta \quad \text{for} \quad k=1,\ldots,n. \qquad (104)$$

Using the relations in (96) and (97) and the inequalities (100), (101), (102), (103) and (104), we conclude that the inequality (93) implies

$$\begin{aligned}
&\eta\left(\sum_{k=1}^n \frac{1}{q_k}\widetilde{P}_k(w_k^{(t)}) - \sum_{i=1}^m \frac{1}{p_i}\widetilde{D}_i(\alpha_i^{(t)})\right) + \eta\left(\sum_{i=1}^m \frac{\Gamma\gamma_i}{2m}\|\alpha_i^{(t)}-\alpha_i^\star\|^2 + \sum_{k=1}^n \frac{\Gamma\lambda}{2}\|w_k^{(t)}-w_k^\star\|^2\right) \\
&+\frac{2\theta(1-\eta)}{3(1-\theta)}\left(\sum_{i=1}^m \frac{\Gamma\gamma_i}{2m}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2 \sum_{k=1}^n \frac{\Gamma\lambda}{2}\|\bar{w}_k^{(s)}-w_k^\star\|^2\right) \\
\geq\ &\sum_{k=1}^n \frac{1}{q_k}\mathbf{E}_t[\widetilde{P}_k(w_k^{(t+1)})] - \sum_{i=1}^m \frac{1}{p_i}\mathbf{E}_t[\widetilde{D}_i(\alpha_i^{(t+1)})] \\
&+\sum_{i=1}^m \frac{\Gamma\gamma_i}{2m}\mathbf{E}_t[\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2] + \sum_{k=1}^n \frac{\Gamma\lambda}{2}\mathbf{E}_t[\|w_k^{(t+1)}-w_k^\star\|^2],
\end{aligned}$$

which is equivalent to

$$\eta\left(\sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(w_k^{(t)})-\sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\alpha_i^{(t)})+\frac{\Gamma\lambda}{2}\|w^{(t)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\alpha_i^{(t)}-\alpha_i^\star\|^2\right)$$
$$+\frac{2\theta(1-\eta)}{3(1-\theta)}\left(\frac{\Gamma\lambda}{2}\|\bar{w}^{(s)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2\right) \tag{105}$$
$$\geq \mathbf{E}_t\left[\sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(w_k^{(t+1)})-\sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\alpha_i^{(t+1)})+\frac{\Gamma\lambda}{2}\|w^{(t+1)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\alpha_i^{(t+1)}-\alpha_i^\star\|^2\right].$$

To simplify further derivation, we define

$$\Delta^{(t)} = \sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(w_k^{(t)})-\sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\alpha_i^{(t)})+\frac{\Gamma\lambda}{2}\|w^{(t)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\alpha_i^{(t)}-\alpha_i^\star\|^2,$$
$$\bar{\Delta}^{(s)} = \sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(\bar{w}_k^{(s)})-\sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\bar{\alpha}_i^{(s)})+\frac{\Gamma\lambda}{2}\|\bar{w}^{(s)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2.$$

Using the facts that $\widetilde{P}_k(\bar{w}_k^{(s)})\geq 0$ and $-\widetilde{D}_i(\bar{\alpha}_i^{(s)})\geq 0$, the inequality (105) implies

$$\frac{2\theta(1-\eta)}{3(1-\theta)}\bar{\Delta}^{(s)}+\eta\mathbf{E}[\Delta^{(t)}] \;\geq\; \mathbf{E}[\Delta^{(t+1)}],$$

where the expectation is taken with respect to all randomness in the s-th stage, that is, the random variables $\{(j^{(0)},l^{(0)}),(j^{(1)},l^{(1)}),\ldots,(j^{(M-1)},l^{(M-1)})\}$. Next we choose $\theta=1/3$ and follow the same arguments as in the proof for Theorem 1 to obtain $\mathbf{E}[\Delta^{(M)}]\leq\frac{2}{3}\Delta^{(0)}$, provided $M\geq\log(3)\Gamma$. This further implies

$$\mathbf{E}[\bar{\Delta}^{(s)}] \;\leq\; \left(\frac{2}{3}\right)^s\bar{\Delta}^{(0)}. \tag{106}$$

By definition of Γ in (94), we have $\frac{1}{q_k}<\Gamma$ for $k=1,\ldots,n$ and $\frac{1}{p_i}<\Gamma$ for $i=1,\ldots,m$. Therefore,

$$\bar{\Delta}^{(0)} \leq \Gamma\left(\sum_{k=1}^{n}\widetilde{P}_k(\bar{w}_k^{(0)})-\sum_{i=1}^{m}\widetilde{D}_i(\bar{\alpha}_i^{(0)})+\frac{\lambda}{2}\|\bar{w}^{(0)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\gamma_i}{2}\|\bar{\alpha}_i^{(0)}-\alpha_i^\star\|^2\right)$$
$$\leq 2\Gamma\left(\sum_{k=1}^{n}\widetilde{P}_k(\bar{w}_k^{(0)})-\sum_{i=1}^{m}\widetilde{D}_i(\bar{\alpha}_i^{(0)})\right)$$
$$\leq 2\Gamma\left(P(\bar{w}^{(0)})-D(\bar{\alpha}^{(0)})\right), \tag{107}$$

where the second inequality used (90) and the last inequality used (91). On the other hand, we can also lower bound $\bar{\Delta}^{(s)}$ using $P(\bar{w}^{(s)})-D(\bar{\alpha}^{(s)})$. To this end, we notice that with $\theta=1/3$,

$$\Gamma \;\geq\; \max_{i,k}\left\{\frac{1}{p_i}\left(1+\frac{18\Lambda}{q_k\lambda\gamma_i}\right),\;\frac{1}{q_k}\left(1+\frac{18n\Lambda}{p_i m\lambda\gamma_i}\right)\right\} \;\geq\; \max_{i,k}\left\{\frac{18\Lambda}{p_iq_k\lambda\gamma_i},\;\frac{18n\Lambda}{p_iq_m\lambda\gamma_i}\right\}.$$

Noticing that $\max_k\{1/q_k\}\geq n$ and $n\Lambda\geq\|X_{i:}\|_F^2$ for all $i=1,\ldots,m$, we have

$$\Gamma\geq\max_{i,k}\left\{\frac{18\Lambda}{q_k\lambda\gamma_i}\right\}\geq\max_i\left\{\frac{18n\Lambda}{\lambda\gamma_i}\right\}\geq\frac{18}{m\lambda}\sum_{i=1}^{m}\frac{n\Lambda}{\gamma_i}\geq\frac{18}{m\lambda}\sum_{i=1}^{m}\frac{\|X_{i:}\|_F^2}{\gamma_i}\geq\frac{18}{m\lambda}\sum_{i=1}^{m}\frac{\|X_{i:}\|^2}{\gamma_i}.$$

Moreover, since $\Gamma\geq\max_k\left\{\frac{18\Lambda}{p_iq_k\lambda\gamma_i}\right\}\geq\frac{18n\Lambda}{p_i\lambda\gamma_i}$ for all i and $mn\Lambda\geq\|X\|_F^2$, we have

$$\frac{1}{m}\sum_{i=1}^{m}\Gamma\gamma_i\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2 \geq \frac{1}{m}\sum_{i=1}^{m}\frac{18n\Lambda}{p_i\lambda\gamma_i}\gamma_i\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2=\frac{18mn\Lambda}{m^2\lambda}\sum_{i=1}^{m}\frac{\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2}{p_i}$$
$$\geq \frac{18\|X\|_F^2}{m^2\lambda}\Big(\sum_{i=1}^{m}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|\Big)^2$$
$$\geq \frac{18\|X\|^2}{m^2\lambda}\sum_{i=1}^{m}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2=\frac{18\|X\|^2}{m^2\lambda}\|\bar{\alpha}^{(s)}-\alpha^\star\|^2.$$

Therefore, from the definition of $\bar{\Delta}^{(s)}$,

$$\bar{\Delta}^{(s)} = \sum_{k=1}^{n}\frac{1}{q_k}\widetilde{P}_k(\bar{w}_k^{(s)})-\sum_{i=1}^{m}\frac{1}{p_i}\widetilde{D}_i(\bar{\alpha}_i^{(s)})+\frac{\Gamma\lambda}{2}\|\bar{w}^{(s)}-w^\star\|^2+\frac{1}{m}\sum_{i=1}^{m}\frac{\Gamma\gamma_i}{2}\|\bar{\alpha}_i^{(s)}-\alpha_i^\star\|^2$$
$$\geq \sum_{k=1}^{n}\widetilde{P}_k(\bar{w}_k^{(s)})-\sum_{i=1}^{m}\widetilde{D}_i(\bar{\alpha}_i^{(s)})+\Big(\frac{18}{m}\sum_{i=1}^{m}\frac{\|X_{i:}\|^2}{\gamma_i}\Big)\|\bar{w}^{(s)}-w^\star\|^2+\frac{18\|X\|^2}{m^2\lambda}\|\bar{\alpha}^{(s)}-\alpha^\star\|^2$$
$$= L(\bar{w}^{(s)},\alpha^\star)-L(w^\star,\bar{\alpha}^{(s)})+\Big(\frac{18}{m}\sum_{i=1}^{m}\frac{\|X_{i:}\|^2}{\gamma_i}\Big)\|\bar{w}^{(s)}-w^\star\|^2+\frac{18\|X\|^2}{m^2\lambda}\|\bar{\alpha}^{(s)}-\alpha^\star\|^2$$
$$\geq P(\bar{w}^{(s)})-D(\bar{\alpha}^{(s)}), \tag{108}$$

where the last inequality is due to Lemma 8. Combining (106), (107) and (108) gives the result:

$$\mathbf{E}\left[P(\bar{w}^{(s)})-D(\bar{\alpha}^{(s)})\right] \;\leq\; \left(\frac{2}{3}\right)^s 2\Gamma\left(P(\bar{w}^{(0)})-D(\bar{\alpha}^{(0)})\right).$$

Appendix C. Proof of Theorem 3

To facilitate the analysis of DSCOVR-SAGA in Algorithm 3, we define two sequences of matrices recursively. The first is $\{W^{(t)}\}_{t\geq 0}$, where each $W^{(t)}\in\mathbf{R}^{m\times d}$. They are partitioned into $m\times n$ blocks, and we denote each block as $W_{ik}^{(t)}\in\mathbf{R}^{1\times d_k}$. The recursive updates for $W^{(t)}$ are as follows:

$$W^{(0)} = \mathbf{1}_m\otimes(w^{(0)})^T,$$
$$W_{ik}^{(t+1)} = \begin{cases}(w_l^{(t)})^T & \text{if } i=j \text{ and } k=l,\\ W_{ik}^{(t)} & \text{otherwise,}\end{cases} \qquad t=0,1,2,\ldots, \tag{109}$$

where $\mathbf{1}_m$ denotes the vector of all ones in $\mathbf{R}^m$, and $\otimes$ denotes the Kronecker product of two matrices. The second sequence is $\{A^{(t)}\}_{t\geq 0}$, where each $A^{(t)}\in\mathbf{R}^{N\times n}$. They are partitioned into $m\times n$ blocks, and we denote each block as $A_{ik}^{(t)}\in\mathbf{R}^{N_i\times 1}$. The recursive updates for $A^{(t)}$ are as follows:

$$A^{(0)} = \alpha^{(0)}\otimes\mathbf{1}_n^T,$$
$$A_{ik}^{(t+1)} = \begin{cases}\alpha_j^{(t)} & \text{if } i=j \text{ and } k=l,\\ A_{ik}^{(t)} & \text{otherwise,}\end{cases} \qquad t=0,1,2,\ldots. \tag{110}$$

The matrices $W^{(t)}$ and $A^{(t)}$ consist of most recent values of the primal and dual block coordinates, updated at different times, up to time t.

Notice that in Algorithm 3, the matrices $U^{(t)} \in \mathbf{R}^{N\times n}$ follow the same partitioning as the matrices $A^{(t)}$, and the matrices $V^{(t)} \in \mathbf{R}^{m\times d}$ follow the same partitioning as the matrices $W^{(t)}$. According to the updates of $U^{(t)}$, $V^{(t)}$, $\bar{u}^{(t)}$ and $\bar{v}^{(t)}$ in Algorithm 3, we have for each $t \geq 0$,

$$\begin{aligned} U_{ik}^{(t)} &= X_{ik}(W_{ik}^{(t)})^T, \quad i = 1,\ldots,m, \quad k = 1,\ldots,n, \\ V_{ik}^{(t)} &= \frac{1}{m}(A_{ik}^{(t)})^T X_{ik}, \quad i = 1,\ldots,m, \quad k = 1,\ldots,n. \end{aligned}$$

Proposition 12 *Suppose Assumption 1 holds. The t-th iteration of Algorithm 3 guarantees*

$$\begin{aligned} &\sum_{i=1}^m \frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right) - \gamma_i + \sum_{k=1}^n \frac{3\tau_k\|X_{ik}\|^2}{mp_i}\right]\|\alpha_i^{(t)} - \alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{2\tau_k\|X_{ik}\|^2}{m^2p_i}\|A_{ik}^{(t)} - \alpha_i^\star\|^2 \\ &+ \sum_{k=1}^n\left[\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right) - \lambda + \sum_{i=1}^m \frac{3\sigma_i\|X_{ik}\|^2}{mq_k}\right]\|w_k^{(t)} - w_k^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{2\sigma_i\|X_{ik}\|^2}{mq_k}\|(W_{ik}^{(t)})^T - w_k^\star\|^2 \\ \geq\ & \sum_{i=1}^m \frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t\left[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2\right] + \sum_{k=1}^n \frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t\left[\|w_k^{(t+1)} - w_k^\star\|^2\right] \end{aligned} \tag{111}$$

Proof The main differences between Algorithm 2 and Algorithm 3 are the definitions of $u_j^{(t+1)}$ and $v_l^{(t+1)}$. We start with the inequality (68) and revisit the bounds for the following two quantities:

$$\mathbf{E}_t\left[\|u_i^{(t+1)} - X_{i:}w^{(t)}\|^2\right] \qquad \text{and} \qquad \mathbf{E}_t\left[\left\|v_k^{(t+1)} - \frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right].$$

For Algorithm 3, we have

$$\begin{aligned} u_i^{(t+1)} &= \bar{u}_i^{(t)} - \frac{1}{q_l}U_{il}^{(t)} + \frac{1}{q_l}X_{il}w_l^{(t)}, \qquad i = 1,\ldots,m, \\ v_k^{(t+1)} &= \bar{v}_k^{(t)} - \frac{1}{p_j}(V_{jk}^{(t)})^T + \frac{1}{p_j}\frac{1}{m}(X_{jk})^T\alpha_j^{(t)}, \qquad k = 1,\ldots,n. \end{aligned}$$

We can apply the reasoning in (39) and (40) to every block coordinate and obtain

$$\begin{aligned} \mathbf{E}_t\left[u_i^{(t+1)}\right] &= X_{i:}w^{(t)}, \qquad i = 1,\ldots,m, \\ \mathbf{E}_t\left[v_k^{(t+1)}\right] &= \frac{1}{m}(X_{:k})^T\alpha^{(t)}, \qquad k = 1,\ldots,n. \end{aligned}$$

Therefore they satisfy the assumptions in Lemma 5 and Lemma 6, respectively. Moreover, following similar arguments as in (69) and (70), we have

$$\begin{aligned} \mathbf{E}_t\left[\|u_i^{(t+1)} - X_{i:}w^{(t)}\|^2\right] &\leq \sum_{k=1}^n \frac{2\|X_{ik}\|^2}{q_k}\left(\left\|(W_{ik}^{(t)})^T - w_k^\star\right\|^2 + \|w_k^{(t)} - w_k^\star\|^2\right), \\ \mathbf{E}_t\left[\left\|v_k^{(t+1)} - \frac{1}{m}(X_{:k})^T\alpha^{(t)}\right\|^2\right] &\leq \sum_{i=1}^m \frac{2\|X_{ik}\|^2}{m^2p_i}\left(\left\|(A_{ik}^{(t)})^T - \alpha_i^\star\right\|^2 + \|\alpha_i^{(t)} - \alpha_i^\star\|^2\right). \end{aligned}$$

The rest of the proof are the same as in the proof of Proposition 7. ■

Now we are ready to prove Theorem 3. By definition of $W^{(t)}$ in (109) and $A^{(t)}$ in (110), we have

$$\begin{aligned} \mathbf{E}_t\left[\|(W_{ik}^{(t+1)})^T - w_k^\star\|^2\right] &= p_iq_k\|w_k^{(t)} - w_k^\star\|^2 + (1-p_iq_k)\|(W_{ik}^{(t)})^T - w_k^\star\|^2, \qquad (112) \\ \mathbf{E}_t\left[\|A_{ik}^{(t+1)} - \alpha_i^\star\|^2\right] &= p_iq_k\|\alpha_i^{(t)} - \alpha_i^\star\|^2 + (1-p_iq_k)\|A_{ik}^{(t)} - \alpha_i^\star\|^2. \qquad (113) \end{aligned}$$

For all $i = 1,\ldots,m$ and $k = 1,\ldots,n$, let

$$\xi_{ik} = \frac{3\sigma_i\|X_{ik}\|^2}{mp_iq_k^2} \qquad \text{and} \qquad \zeta_{ik} = \frac{3\tau_k\|X_{ik}\|^2}{m^2p_i^2q_k}. \tag{114}$$

We multiply (112) by ξ_{ik} and (113) by ζ_{ik} and add them to (111) to obtain

$$\begin{aligned} &\sum_{i=1}^m \frac{1}{m}\left[\frac{1}{p_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right) - \gamma_i + \sum_{k=1}^n \frac{6\tau_k\|X_{ik}\|^2}{mp_i}\right]\|\alpha_i^{(t)} - \alpha_i^\star\|^2 \\ &+ \sum_{k=1}^n\left[\frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right) - \lambda + \sum_{i=1}^m \frac{6\sigma_i\|X_{ik}\|^2}{mq_k}\right]\|w_k^{(t)} - w_k^\star\|^2 \\ &+ \sum_{i=1}^m\sum_{k=1}^n\left(1-\frac{1}{3}p_iq_k\right)\zeta_{ik}\|A_{ik}^{(t)} - \alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n\left(1-\frac{1}{3}p_iq_k\right)\xi_{ik}\|(W_{ik}^{(t)})^T - w_k^\star\|^2 \\ \geq\ & \sum_{i=1}^m \frac{1}{mp_i}\left(\frac{1}{2\sigma_i}+\gamma_i\right)\mathbf{E}_t\left[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2\right] + \sum_{k=1}^n \frac{1}{q_k}\left(\frac{1}{2\tau_k}+\lambda\right)\mathbf{E}_t\left[\|w_k^{(t+1)} - w_k^\star\|^2\right] \\ &+ \sum_{i=1}^m\sum_{k=1}^n \zeta_{ik}\mathbf{E}_t\left[\|A_{ik}^{(t+1)} - \alpha_i^\star\|^2\right] + \sum_{i=1}^m\sum_{k=1}^n \xi_{ik}\mathbf{E}_t\left[\|(W_{ik}^{(t+1)})^T - w_k^\star\|^2\right]. \end{aligned} \tag{115}$$

Let $\theta \in (0,1)$ be a parameter to be determined later, and Γ be a constant such that

$$\Gamma \geq \max_{i,k}\left\{\frac{1}{p_i}\left(1+\frac{3\|X_{ik}\|^2}{2\theta q_k\lambda\gamma_i}\right), \frac{1}{q_k}\left(1+\frac{3n\|X_{ik}\|^2}{2\theta p_im\lambda\gamma_i}\right), \frac{1}{p_iq_k}\right\}. \tag{116}$$

The choices of σ_i in (42) and τ_k in (43) satisfy

$$\frac{1}{p_i}\left(1+\frac{1}{2\sigma_i\gamma_i}\right) = \frac{1}{q_k}\left(1+\frac{1}{2\tau_k\lambda}\right) = \Gamma. \tag{117}$$

Comparing the above equality with the definition of Γ in (116), we have

$$\frac{3\|X_{ik}\|^2}{2\theta q_k\lambda\gamma_i} \leq \frac{1}{2\sigma_i\gamma_i} \qquad \text{and} \qquad \frac{3n\|X_{ik}\|^2}{2\theta p_im\lambda\gamma_i} \leq \frac{1}{2\tau_k\lambda},$$

which implies that

$$\frac{6\sigma_i\|X_{ik}\|^2}{q_k} \leq 2\theta\lambda \qquad \text{and} \qquad \frac{6n\tau_k\|X_{ik}\|^2}{mp_i} \leq 2\theta\gamma_i \tag{118}$$

hold for all $i = 1,\ldots,m$ and $k = 1,\ldots,n$. Therefore, we have

$$\sum_{k=1}^n \frac{6\tau_k\|X_{ik}\|^2}{mp_i} = \frac{1}{n}\sum_{k=1}^n \frac{6n\tau_k\|X_{ik}\|^2}{mp_i} \leq 2\theta\gamma_i, \qquad i = 1,\ldots,m, \tag{119}$$

$$\sum_{i=1}^m \frac{6\sigma_i\|X_{ik}\|^2}{mq_k} = \frac{1}{m}\sum_{i=1}^m \frac{6\sigma_i\|X_{ik}\|^2}{q_k} \leq 2\theta\lambda, \qquad k = 1,\ldots,n. \tag{120}$$

Now we consider the inequality (115), and examine the ratio between the coefficients of $\|\alpha_i^{(t)} - \alpha_i^\star\|^2$ and $\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2]$. Using (119) and (117), we have

$$\frac{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right) - \gamma_i + \sum_{k=1}^n \frac{6\tau_k \|X_{ik}\|^2}{mp_i}}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)} \le 1 - \frac{(1-2\theta)\gamma_i}{\frac{1}{p_i}\left(\frac{1}{2\sigma_i} + \gamma_i\right)} = 1 - \frac{1-2\theta}{\Gamma}. \tag{121}$$

Similarly, the ratio between the coefficients of $\|w_k^{(t)} - w_k^\star\|^2$ and $\mathbf{E}_t[\|w_k^{(t+1)} - w_k^\star\|^2]$ can be bounded using (120) and (117):

$$\frac{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right) - \lambda + \sum_{i=1}^m \frac{6\sigma_i \|X_{ik}\|^2}{mq_k}}{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)} \le 1 - \frac{(1-2\theta)\lambda}{\frac{1}{q_k}\left(\frac{1}{2\tau_k} + \lambda\right)} = 1 - \frac{1-2\theta}{\Gamma}. \tag{122}$$

We notice that in (115), the ratios between the coefficients of $\zeta_{ik}\|A_{ik}^{(t)} - \alpha_i^\star\|^2$ and $\zeta_{ik}\mathbf{E}_t\left[\|A_{ik}^{(t+1)} - \alpha_i^\star\|^2\right]$, as well as and that of $\xi_{ik}\|(W_{ik}^{(t)})^T - w_k^\star\|^2$ and $\xi_{ik}\mathbf{E}_t[\|(W_{ik}^{(t+1)})^T - w_k^\star\|^2]$, are all $1 - \frac{1}{3}p_i q_k$. By definition of Γ in (116), we have

$$1 - \frac{1}{3}p_i q_k \;\le\; 1 - \frac{1}{3\Gamma}, \qquad i = 1, \ldots, m, \quad k = 1, \ldots, n. \tag{123}$$

We choose $\theta = 1/3$ so that the ratios in (121) and (122) have the same bound $1 - \frac{1}{3\Gamma}$. Therefore, it follows from inequality (115) that

$$\begin{aligned}
&\sum_{i=1}^m \frac{\Gamma\gamma_i}{m}\mathbf{E}_t[\|\alpha_i^{(t+1)} - \alpha_i^\star\|^2] + \sum_{k=1}^n \Gamma\lambda\mathbf{E}_t[\|w_k^{(t+1)} - w_k^\star\|^2] \\
&+ \sum_{i=1}^m\sum_{k=1}^n \zeta_{ik}\mathbf{E}_t[\|A_{ik}^{(t+1)} - \alpha_i^\star\|^2] + \sum_{i=1}^m\sum_{k=1}^n \xi_{ik}\mathbf{E}_t[\|(W_{ik}^{(t+1)})^T - w_k^\star\|^2]. \\
\le\;& \left(1 - \frac{1}{3\Gamma}\right)\left(\sum_{i=1}^m \frac{\Gamma\gamma_i}{m}\|\alpha_i^{(t)} - \alpha_i^\star\|^2 + \sum_{k=1}^n \Gamma\lambda\|w_k^{(t)} - w_k^\star\|^2\right. \\
&\left. + \sum_{i=1}^m\sum_{k=1}^n \zeta_{ik}\|A_{ik}^{(t)} - \alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \xi_{ik}\|(W_{ik}^{(t)})^T - w_k^\star\|^2\right).
\end{aligned} \tag{124}$$

Let's define

$$\Delta^{(t)} = \lambda\|w^{(t)} - w^\star\|^2 + \frac{1}{m}\sum_{i=1}^m \gamma_i\|\alpha_i^{(t)} - \alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{\zeta_{ik}}{\Gamma}\|A_{ik}^{(t)} - \alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{\xi_{ik}}{\Gamma}\|(W_{ik}^{(t)})^T - w_k^\star\|^2.$$

Then (124) implies

$$\mathbf{E}\left[\Delta^{(t)}\right] \le \left(1 - \frac{1}{3\Gamma}\right)^t \Delta^{(0)}, \tag{125}$$

where the expectation is taken with respect to all random variables generated by Algorithm 3 up to iteration t.

By the definition of ξ_{ik} in (114), we have

$$\frac{\xi_{ik}}{\Gamma} = \frac{3\sigma_i\|X_{ik}\|^2}{mp_i q_k^2}\frac{1}{\Gamma} \le \frac{\theta\lambda}{mp_i q_k}\frac{1}{\Gamma} \le \frac{\theta\lambda}{m} = \frac{\lambda}{3m},$$

where the first inequality is due to (118) and the second inequality is due to the relation $\Gamma \ge \frac{1}{p_i q_k}$ from the definition of Γ in (116). Similarly, we have

$$\frac{\zeta_{ik}}{\Gamma} = \frac{3\tau_k\|X_{ik}\|^2}{m^2 p_i^2 q_k}\frac{1}{\Gamma} \le \frac{\theta\gamma_i}{mnp_i q_k}\frac{1}{\Gamma} \le \frac{\theta\gamma_i}{3mn} = \frac{\gamma_i}{3mn}.$$

Moreover, by the construction in (109) and (110), we have for $t = 0$,

$$\begin{aligned}
A_{ik}^{(0)} &= \alpha_i^{(0)}, \quad \text{for} \quad k = 1, \ldots, n \quad \text{and} \quad i = 1, \ldots, m, \\
(W_{ik}^{(0)})^T &= w_k^{(0)}, \quad \text{for} \quad i = 1, \ldots, m \quad \text{and} \quad k = 1, \ldots, n.
\end{aligned}$$

Therefore, the last two terms in the definition of $\Delta^{(0)}$ can be bounded as

$$\begin{aligned}
&\sum_{i=1}^m\sum_{k=1}^n \frac{\zeta_{ik}}{\Gamma}\|A_{ik}^{(0)} - \alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{\xi_{ik}}{\Gamma}\|(W_{ik}^{(0)})^T - w_k^\star\|^2 \\
\le\;& \sum_{i=1}^m\sum_{k=1}^n \frac{\gamma_i}{3mn}\|\alpha_i^{(0)} - \alpha_i^\star\|^2 + \sum_{i=1}^m\sum_{k=1}^n \frac{\lambda}{3m}\|w_k^{(0)} - w_k^\star\|^2 \\
=\;& \frac{1}{3m}\sum_{i=1}^m \gamma_i\|\alpha_i^{(0)} - \alpha_i^\star\|^2 + \frac{\lambda}{3}\|w^{(0)} - w^\star\|^2,
\end{aligned}$$

which implies

$$\Delta^{(0)} \le \frac{4}{3}\left(\lambda\|w^{(0)} - w^\star\|^2 + \frac{1}{m}\sum_{i=1}^m \gamma_i\|\alpha_i^{(0)} - \alpha_i^\star\|^2\right).$$

Finally, combining with (125), we have

$$\mathbf{E}\left[\Delta^{(t)}\right] \le \left(1 - \frac{1}{3\Gamma}\right)^t \frac{4}{3}\left(\lambda\|w^{(0)} - w^\star\|^2 + \frac{1}{m}\sum_{i=1}^m \gamma_i\|\alpha_i^{(0)} - \alpha_i^\star\|^2\right),$$

which further implies the desired result.

Appendix D. Proof of Theorem 4

To simplify the presentation, we present the proof for the case $\gamma_i = \gamma$ for all $i = 1, \ldots, m$. It is straightforward to generalize to the case where the γ_i's are different.

Lemma 13 *Let $g : \mathbf{R}^D \to \mathbf{R}$ be λ-strongly convex, and $f_i^* : \mathbf{R}^{N_i} \to \mathbf{R} \cup \{\infty\}$ be γ-strongly convex over its domain. Given any $\tilde{w} \in \mathbf{R}^d$ and $\tilde{\alpha} \in \mathbf{R}^N$, we define the following two functions:*

$$L(w, \alpha) = g(w) + \frac{1}{m}\alpha^T X w - \frac{1}{m}\sum_{i=1}^m f_i*(\alpha_i), \tag{126}$$

$$L_\delta(w, \alpha) = L(w, \alpha) + \frac{\delta\lambda}{2}\|w - \tilde{w}\|^2 - \frac{\delta\gamma}{2m}\|\alpha - \tilde{\alpha}\|^2. \tag{127}$$

Let $(w^\star, \alpha^\star)$ and $(\tilde{w}^\star, \tilde{\alpha}^\star)$ be the (unique) saddle points of $L(w, \alpha)$ and $L_\delta(w, \alpha)$, respectively. Then we have

$$\lambda\|\tilde{w} - \tilde{w}^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \tilde{\alpha}^\star\|^2 \le \lambda\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \alpha^\star\|^2, \tag{128}$$

$$\left(\lambda\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \alpha^\star\|^2\right)^{1/2} \le \frac{\delta}{1+\delta}\left(\lambda\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \alpha^\star\|^2\right)^{1/2}. \tag{129}$$

Proof This lemma can be proved using the theory of monotone operators (e.g., Rockafellar, 1976; Ryu and Boyd, 2016), as done by Balamurugan and Bach (2016). Here we give an elementary proof based on first-order optimality conditions.

By assumption, we have

$$\begin{aligned}(w^\star, \alpha^\star) &= \arg\min_w \max_\alpha L(w, \alpha),\\ (\tilde{w}^\star, \tilde{\alpha}^\star) &= \arg\min_w \max_\alpha L_\delta(w, \alpha).\end{aligned}$$

Optimality conditions for $(\tilde{w}^\star, \tilde{\alpha}^\star)$ as a saddle point of L_δ:

$$-\frac{1}{m}X^T\tilde{\alpha}^\star - \delta\lambda\left(\tilde{w}^\star - \tilde{w}\right) \in \partial g(\tilde{w}^\star), \tag{130}$$

$$X\tilde{w}^\star - \delta\gamma\left(\tilde{\alpha}^\star - \tilde{\alpha}\right) \in \partial\sum_{i=1}^m f_i^*(\tilde{\alpha}^\star). \tag{131}$$

For any $\xi \in \partial g(\tilde{w}^\star)$, it holds that $\xi + \frac{1}{m}X^T\alpha^\star \in \partial_w L(\tilde{w}^\star, \alpha^\star)$. Therefore using (130) we have

$$\frac{1}{m}X^T(\alpha^\star - \tilde{\alpha}^\star) - \delta\lambda\left(\tilde{w}^\star - \tilde{w}\right) \in \partial_w L(\tilde{w}^\star, \alpha^\star).$$

Since $L(w, \alpha^\star)$ is strongly convex in w with convexity parameter λ, we have

$$L(\tilde{w}^\star, \alpha^\star) + \left(\frac{1}{m}X^T(\alpha^\star - \tilde{\alpha}^\star) - \delta\lambda\left(\tilde{w}^\star - \tilde{w}\right)\right)^T(w^\star - \tilde{w}^\star) + \frac{\lambda}{2}\|\tilde{w}^\star - w^\star\|^2 \le L(w^\star, \alpha^\star). \tag{132}$$

Similarly, we have

$$\frac{1}{m}X^T(\tilde{w}^\star - w^\star) - \frac{\delta\gamma}{m}\left(\tilde{\alpha}^\star - \tilde{\alpha}\right) \in \partial_\alpha\left(-L(w^\star, \tilde{\alpha}^\star)\right),$$

and since $-L(w^\star, \alpha)$ is strongly convex in α with convexity parameter $\frac{\gamma}{m}$, we have

$$-L(w^\star, \tilde{\alpha}^\star) + \left(\frac{1}{m}X^T(\tilde{w}^\star - w^\star) - \frac{\delta\gamma}{m}\left(\tilde{\alpha}^\star - \tilde{\alpha}\right)\right)^T(\alpha^\star - \tilde{\alpha}^\star) + \frac{\gamma}{2m}\|\tilde{\alpha}^\star - \alpha^\star\|^2 \le -L(w^\star, \alpha^\star). \tag{133}$$

Adding inequalities (132) and (133) together gives

$$\begin{aligned}&L(\tilde{w}^\star, \alpha^\star) - L(w^\star, \tilde{\alpha}^\star)\\ &+ \delta\lambda(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) + \frac{\delta\gamma}{m}(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) + \frac{\lambda}{2}\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{2m}\|\tilde{\alpha}^\star - \alpha^\star\|^2 \le 0.\end{aligned}$$

Combining with the inequality

$$L(\tilde{w}^\star, \alpha^\star) - L(w^\star, \tilde{\alpha}^\star) \ge \frac{\lambda}{2}\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{2m}\|\tilde{\alpha}^\star - \alpha^\star\|^2,$$

we obtain

$$\lambda\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \alpha^\star\|^2 + \delta\lambda(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) + \frac{\delta\gamma}{m}(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) \le 0. \tag{134}$$

Proof of the first claim. We can drop the nonnegative terms on the left-hand side of (134) to obtain

$$\lambda(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) + \frac{\gamma}{m}(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) \le 0.$$

The two inner product terms on the left-hand side of the inequality above can be expanded as follows:

$$\begin{aligned}(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) &= (\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - \tilde{w} + \tilde{w} - w^\star) = \|\tilde{w}^\star - \tilde{w}\|^2 + (\tilde{w}^\star - \tilde{w})^T(\tilde{w} - w^\star),\\ (\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) &= (\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \tilde{\alpha} + \tilde{\alpha} - \alpha^\star) = \|\tilde{\alpha}^\star - \tilde{\alpha}\|^2 + (\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha} - \alpha^\star).\end{aligned}$$

Combining them with the last inequality, we have

$$\begin{aligned}\lambda\|\tilde{w}^\star - \tilde{w}\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \tilde{\alpha}\|^2 &\le -\lambda(\tilde{w}^\star - \tilde{w})^T(\tilde{w} - w^\star) - \frac{\gamma}{m}(\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha} - \alpha^\star)\\ &\le \frac{\lambda}{2}\left(\|\tilde{w}^\star - \tilde{w}\|^2 + \|\tilde{w} - w^\star\|^2\right) + \frac{\gamma}{2m}\left(\|\tilde{\alpha}^\star - \tilde{\alpha}\|^2 + \|\tilde{\alpha} - \alpha^\star\|^2\right),\end{aligned}$$

which implies

$$\frac{\lambda}{2}\|\tilde{w}^\star - \tilde{w}\|^2 + \frac{\gamma}{2m}\|\tilde{\alpha}^\star - \tilde{\alpha}\|^2 \le \frac{\lambda}{2}\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{2m}\|\tilde{\alpha} - \alpha^\star\|^2.$$

Proof of the second claim. We expand the two inner product terms in (134) as follows:

$$\begin{aligned}(\tilde{w}^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) &= (\tilde{w}^\star - w^\star + w^\star - \tilde{w})^T(\tilde{w}^\star - w^\star) = \|\tilde{w}^\star - w^\star\|^2 + (w^\star - \tilde{w})^T(\tilde{w}^\star - w^\star),\\ (\tilde{\alpha}^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) &= (\tilde{\alpha}^\star - \alpha^\star + \alpha^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star) = \|\tilde{\alpha}^\star - \alpha^\star\|^2 + (\alpha^\star - \tilde{\alpha})^T(\tilde{\alpha}^\star - \alpha^\star).\end{aligned}$$

Then (134) becomes

$$\begin{aligned}&(1+\delta)\lambda\|\tilde{w}^\star - w^\star\|^2 + (1+\delta)\frac{\Gamma}{m}\|\tilde{w}^\star - w^\star\|^2\\ \le\ &\delta\lambda(\tilde{w} - w^\star)^T(\tilde{w}^\star - w^\star) + \frac{\delta\gamma}{m}(\tilde{\alpha} - \alpha^\star)^T(\tilde{\alpha}^\star - \alpha^\star)\\ \le\ &\delta\left(\lambda\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \alpha^\star\|^2\right)^{1/2}\left(\lambda\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \alpha^\star\|^2\right)^{1/2},\end{aligned}$$

where in the second inequality we used the Cauchy-Schwarz inequality. Therefore we have

$$\left(\lambda\|\tilde{w}^\star - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha}^\star - \alpha^\star\|^2\right)^{1/2} \le \frac{\delta}{1+\delta}\left(\lambda\|\tilde{w} - w^\star\|^2 + \frac{\gamma}{m}\|\tilde{\alpha} - \alpha^\star\|^2\right)^{1/2},$$

which is the desired result. ■

To simplify notations in the rest of the proof, we let $z = (w, \alpha)$ and define

$$\|z\| = \left(\lambda\|w\|^2 + \frac{\gamma}{m}\|\alpha\|^2\right)^{1/2}.$$

The results of Lemma 13 can be written as

$$\|\tilde{z} - \tilde{z}^\star\| \le \|\tilde{z} - z^\star\|, \tag{135}$$

$$\|\tilde{z}^\star - z^\star\| \le \frac{\delta}{1+\delta}\|\tilde{z} - z^\star\|. \tag{136}$$

Next consider the convergence of Algorithm 4, and follow the proof ideas in Balamurugan and Bach (2016, Section D.3).

If we use DSCOVR-SVRG (option 1) in each round of Algorithm 4, then Algorithm 2 is called with initial point $\tilde{z}^{(r)} = (\tilde{w}^{(r)}, \tilde{\alpha}^{(r)})$ and after S stages, it outputs $\tilde{z}^{(r+1)}$ as an approximate saddle point of $L_\delta^{(r)}(w, \alpha)$, which is defined in (46). Then Theorem 1 implies

$$\mathbf{E}[\|\tilde{z}^{(r+1)} - \tilde{z}^{\star(r)}\|^2] \le \left(\frac{2}{3}\right)^S \mathbf{E}[\|\tilde{z}^{(r)} - \tilde{z}^{\star(r)}\|^2], \tag{137}$$

where $\tilde{z}^{\star(r)}$ denotes the unique saddle point of $L_\delta^{(r)}(w, \alpha)$. By Minkowski's inequality, we have

$$\left(\mathbf{E}[\|\tilde{z}^{(r+1)} - z^\star\|^2]\right)^{1/2} \le \left(\mathbf{E}[\|\tilde{z}^{(r+1)} - \tilde{z}^{\star(r)}\|^2]\right)^{1/2} + \left(\mathbf{E}[\|\tilde{z}^{\star(r)} - z^\star\|^2]\right)^{1/2},$$

where $z^\star$ is the unique saddle point of $L(w, \alpha)$. Using (137), (135) and (136), we obtain

$$\begin{aligned}
\left(\mathbf{E}[\|\tilde{z}^{(r+1)} - z^\star\|^2]\right)^{1/2} &\le \left(\frac{2}{3}\right)^{S/2} \left(\mathbf{E}[\|\tilde{z}^{(r)} - \tilde{z}^{\star(r)}\|^2]\right)^{1/2} + \left(\mathbf{E}[\|\tilde{z}^{\star(r)} - z^\star\|^2]\right)^{1/2} \\
&\le \left(\frac{2}{3}\right)^{S/2} \left(\mathbf{E}[\|\tilde{z}^{(r)} - z^\star\|^2]\right)^{1/2} + \frac{\delta}{1+\delta} \left(\mathbf{E}[\|\tilde{z}^{(r)} - z^\star\|^2]\right)^{1/2} \\
&= \left[\left(\frac{2}{3}\right)^{S/2} + \frac{\delta}{1+\delta}\right] \left(\mathbf{E}[\|\tilde{z}^{(r)} - z^\star\|^2]\right)^{1/2},
\end{aligned} \tag{138}$$

Therefore, if $S \ge \frac{2\log(2(1+\delta))}{\log(3/2)}$, we have

$$\left(\frac{2}{3}\right)^{S/2} + \frac{\delta}{1+\delta} \le \frac{1}{2(1+\delta)} + \frac{\delta}{1+\delta} = \frac{1+2\delta}{2(1+\delta)} = 1 - \frac{1}{2(1+\delta)},$$

which implies

$$\mathbf{E}[\|\tilde{z}^{(r+1)} - z^\star\|^2] \le \left(1 - \frac{1}{2(1+\delta)}\right)^2 \mathbf{E}[\|\tilde{z}^{(r)} - z^\star\|^2]. \tag{139}$$

If we use DISCOVR-SAGA (option 2) in Algorithm 4, then Algorithm 3 is called with initial point $\tilde{z}^{(r)} = (\tilde{w}^{(r)}, \tilde{\alpha}^{(r)})$ and after M steps, it outputs $\tilde{z}^{(r+1)}$ as an approximate saddle point of $L_\delta^{(r)}(w, \alpha)$. Then Theorem 3 implies

$$\mathbf{E}[\|\tilde{z}^{(r+1)} - \tilde{z}^{\star(r)}\|^2] \le \frac{4}{3}\left(1 - \frac{1}{3\Gamma_\delta}\right)^M \mathbf{E}[\|\tilde{z}^{(r)} - \tilde{z}^{\star(r)}\|^2].$$

Using similar arguments as in (138), we have

$$\left(\mathbf{E}[\|\tilde{z}^{(r+1)} - z^\star\|]^2\right)^{1/2} \le \left[\frac{4}{3}\left(1 - \frac{1}{3\Gamma_\delta}\right)^{M/2} + \frac{\delta}{1+\delta}\right] \left(\mathbf{E}[\|\tilde{z}^{(r)} - z^\star\|^2]\right)^{1/2}.$$

Therefore, if $M \ge 6\log\left(\frac{8(1+\delta)}{3}\right)\Gamma_\delta$, we have

$$\frac{4}{3}\left(1 - \frac{1}{3\Gamma_\delta}\right)^{M/2} + \frac{\delta}{1+\delta} \le \frac{1}{2(1+\delta)} + \frac{\delta}{1+\delta} = \frac{1+2\delta}{2(1+\delta)} = 1 - \frac{1}{2(1+\delta)},$$

which implies the same inequality in (139).

In summary, using either option 1 or option 2 in Algorithm 4, we have

$$\mathbf{E}[\|\tilde{z}^{(r)} - z^\star\|^2] \le \left(1 - \frac{1}{2(1+\delta)}\right)^{2r} \|\tilde{z}^{(0)} - z^\star\|^2.$$

In order to have $\mathbf{E}[\|\tilde{z}^{(r)} - z^\star\|^2] \le \epsilon$, it suffices to have $r \ge (1+\delta)\log\left(\|\tilde{z}^{(0)} - z^\star\|^2/\epsilon\right)$.

References

Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems 24*, pages 873–881, 2011.

Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. In *Proceedings of 49th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 1200–1205, 2017.

Yossi Arjevani and Ohad Shamir. Communication complexity of distributed convex learning and optimization. In *Advances in Neural Information Processing Systems 28*, pages 1756–1764. 2015.

Arda Aytekin, Hamid Reza Feyzmahdavian, and Mikael Johansson. Analysis and implementatino of an asynchronous optimization algorithm for the parameter server. arXiv:1610.05507, 2016.

Palaniappan Balamurugan and Francis Bach. Stochastic variance reduction methods for saddle-point problems. In *Advances in Neural Information Processing Systems 29*, pages 1416–1424, 2016.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-threshold algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, 1989.

Stephen P. Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

Antonin Chambolle and Thomas Pock. On the ergodic convergence rates of a first-order primal–dual algorithm. *Mathematical Programming*, pages 1–35, 2015.

Weizhu Chen, Zhenghao Wang, and Jingren Zhou. Large-scale L-BFGS using MapReduce. In *Advances in Neural Information Processing Systems 27*, pages 1332–1340. 2014.

Jeff Dean and Sanjay Ghemawat. MapReduce: Simplfied data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems 27*, pages 1646–1654. 2014.

John C Duchi, Alekh Agarwal, and Martin J Wainwright. Dual averaging for distributed optimization: convergence analysis and network scaling. *IEEE Transactions on Automatic Control*, 57(3):592–606, 2012.

Rong-En Fan and Chih-Jen Lin. LIBSVM data: Classification, regression and multi-label. URL: http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets, 2011.

Roy Frostig, Rong Ge, Sham Kakade, and Aaron Sidford. Un-regularizing: approximate proximal point and faster stochastic algorithms for empirical risk minimization. In *Proceedings of The 32nd International Conference on Machine Learning (ICML)*, pages 2540–2548. 2015.

Robert Hannah and Wotao Yin. More iterations per second, same quality — why asynchronous algorithms may drastically outperform traditional ones. CAM Report 17-50, University of California at Los Angeles, 2017.

Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Fundamentals of Convex Analysis*. Springer, 2001.

Martin Jaggi, Virginia Smith, Martin Takac, Jonathan Terhorst, Sanjay Krishnan, Thomas Hofmann, and Michael I Jordan. Communication-efficient distributed dual coordinate ascent. In *Advances in Neural Information Processing Systems 27*, pages 3068–3076. 2014.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems 26*, pages 315–323. 2013.

Guanghui Lan and Yi Zhou. An optimal randomized incremental gradient method. *Mathematical Programming*, 171:167–215, 2018.

Nicolas Le Roux, Mark Schmidt, and Francis Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems 25*, pages 2672–2680. 2012.

Ching-Pei Lee, Po-Wei Wang, Weizhu Chen, and Chih-Jen Lin. Limited-memory common-directions method for distributed optimization and its application on empirical risk minimization. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 732–740, 2017a.

Jason D. Lee, Qihang Lin, Tengyu Ma, and Tianbao Yang. Distributed stochastic variance reduced gradient methods by sampling extra data with replacement. *Journal of Machine Learning Research*, 18(122):1–43, 2017b.

Mu Li, David G. Andersen, Jun Woo Park, Alexander J. Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J. Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 583–598, 2014.

Chieh-Yen Lin, Cheng-Hao Tsai, Ching-Pei Lee, and Chih-Jen Lin. Large-scale logistic regression and linear support vector machines using Spark. In *Proceedings of the IEEE Conference on Big Data*, Washington DC, USA, 2014.

Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems 28*, pages 3384–3392. 2015.

Qihang Lin and Lin Xiao. An adaptive accelerated proximal gradient method and its homotopy continuation for sparse optimization. *Computatoinal Optimization and Applications*, 60(3):633–674, 2015.

Ji Liu, Stephen J. Wright, Christopher Ré, Victor Bittorf, and Srikrishna Sridhar. An asynchronous parallel stochastic coordinate descent algorithm. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 469–477, 2014.

Chenxin Ma, Virginia Smith, Martin Jaggi, Michael I. Jordan, Peter Richtárik, and Martin Takáč. Adding vs. averaging in distributed primal-dual optimization. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning (ICML)*, pages 1973–1982, 2015.

Chenxin Ma, Virginia Smith, Martin Jaggi, Michael I. Jordan, Peter Richtárik, and Martin Takáč. Distributed optimization with arbitrary local solvers. *Optimization Methods and Software*, 32(4): 813–848, 2017.

Shin Matsushima, Hyokun Yun, Xinhua Zhang, and S. V. N. Vishwanathan. Distributed stochastic optimization of the regularized risk. arXiv:1406.4363, 2014.

H. Brendan McMahan and Matthew J. Streeter. Delay-tolerant algorithms for asynchronous distributed online learning. In *Advances in Neural Information Processing Systems 27*, pages 2915–2923, 2014.

Xiangrui Meng, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman, DB Tsai, Manish Amde, Sean Owen, Doris Xin, Reynold Xin, Michael J. Franklin, Reza Zadeh, Matei Zaharia, and Ameet Talwalkar. MLlib: Machine learning in Apache Spark. *Journal of Machine Learning Research*, 17(34):1–7, 2016.

MPI Forum. MPI: a message-passing interface standard, Version 3.0. Document available at `http://www.mpi-forum.org`, 2012.

Angelia Nedić and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, January 2009.

Angelia Nedić, Alex Olshevsky, and Wei Shi. Achieving geometric convergence for distributed optimizaiton over time-varying graphs. *SIAM Jorunal on Optimization*, 27(4):2597–2633, 2017.

Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer, Boston, 2004.

Yurii Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Yurii Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming, Ser. B*, 140:125–161, 2013.

OpenMP Architecture Review Board. OpenMP Application Program Interface, Version 3.1. Available at `http://www.openmp.org`, July 2011.

Zhimin Peng, Yangyang Xu, Ming Yan, and Wotao Yin. ARock: An algorithmic framework for asynchronous parallel coordinate updates. *SIAM Journal on Scientific Computing*, 38(5):2851–2879, 2016.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems 24*, pages 693–701, 2011.

Sashank J. Reddi, Ahmed Hefny, Suvrit Sra, Barnabás Póczós, and Alexander J Smola. On variance reduction in stochastic gradient descent and its asynchronous variants. In *Advances in Neural Information Processing Systems 28*, pages 2647–2655. 2015.

Sashank J. Reddi, Jakub Konečný, Peter Richtárik, Barnabás Póczós, and Alex Smola. AIDE: Fast and communication efficient distributed optimization. arXiv:1608.06879, 2016.

Peter Richtárik and Martin Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1):1–38, 2014.

Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1):433–484, 2016.

R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

R. Tyrrell Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14(5), 1976.

Ernest K. Ryu and Stephen P. Boyd. A primer on monotone operator methods. *Applied and Computational Mathematics: an International Journal*, 15(1):3–43, 2016.

Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal algorithms for smooth and strongly convex distributed optimization in networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3027–3036, Sydney, Australia, 2017.

Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14:567–599, 2013.

Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1000–1008, Bejing, China, 2014.

Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. EXTRA: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

Suvrit Sra, Adams Wei Yu, Mu Li, and Alexander J. Smola. Adadelay: Delay adaptive distributed stochastic optimization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 957–965, 2016.

Jialei Wang and Lin Xiao. Exploiting strong convexity from data with primal-dual first-order algorithms. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3694–3702, Sydney, Australia, 2017.

Lin Xiao and Stephen P. Boyd. Optimal scaling of a gradient method for distributed resource allocation. *Journal of Optimization Theory and Applications*, 129(3):469–488, June 2006.

Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

Eric P. Xing, Qirong Ho, Wei Dai, Jin Kyu Kim, Jinliang Wei, Seunghak Lee, Xun Zheng, Pengtao Xie, Abhimanu Kumar, and Yaoliang Yu. Petuum: A new platform for distributed machine learning on big data. *IEEE Transactions on Big Data*, 1(2):49–67, 2015.

Tianbao Yang. Trading computation for communication: Distributed stochastic dual coordinate ascent. In *Advances in Neural Information Processing Systems 26*, pages 629–637. 2013.

Adams Wei Yu, Qihang Lin, and Tianbao Yang. Doubly stochastic primal-dual coordinate method for bilinear saddle-point problem. arXiv:1508.03390, 2015.

Hyokyun Yun, Hsiang-Fu Yu, Cho-Jui Hsieh, S V N Vishwanathan, and Inderjit Dhillon. NOMAD: Non-locking, stochastic multi-machine algorithm for asynchronous and decentralized matrix completion. In *Proceedings of the VLDB Endowment*, volume 7, pages 975–986, 2014.

Matei Zaharia, Reynold Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J. Franklin, Ali Ghodsi, Joseph Gonzalez, Scott Shenker, and Ion Stoica. Apache Spark: A unified engine for big data processing. *Communications of the ACM*, 59(11):56–65, 2016.

Yuchen Zhang and Lin Xiao. DiSCO: Distributed optimization for self-concordant empirical loss. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 362–370, Lille, France, 2015.

Yuchen Zhang and Lin Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. *Journal of Machine Learning Research*, 18(84):1–42, 2017.

Yuchen Zhang, John C Duchi, and Martin J Wainwright. Communication-efficient algorithms for statistical optimization. *Journal of Machine Learning Research*, 14:3321–3363, 2013.

Journal of Machine Learning Research 20 (2019) 1-5 Submitted 12/17; Revised 10/18; Published 3/19

Picasso: A Sparse Learning Library for High Dimensional Data Analysis in R and Python*

Jason Ge† JIANGE@PRINCETON.EDU
Xingguo Li§ LIXX1661@UMN.EDU
Haoming Jiang◇ JIANGHM@GATECH.EDU
Han Liu‡ HANLIU@NORTHWESTERN.EDU
Tong Zhang‡ TONGZHANG@TONGZHANG-ML.ORG
Mengdi Wang† MENGDIW@PRINCETON.EDU
Tuo Zhao◇ TUO.ZHAO@ISYE.GATECH.EDU
†*Department of Operations Research and Financial Engineering, Princeton University*
§*Computer Science Department, Princeton University*
◇*School of Industrial and Systems Engineering, Georgia Tech*
‡*Tencent Artificial Intelligence Lab*

Editor: Cheng Soon Ong

Abstract

We describe a new library named `picasso` [1], which implements a unified framework of pathwise coordinate optimization for a variety of sparse learning problems (e.g., sparse linear regression, sparse logistic regression, sparse Poisson regression and scaled sparse linear regression) combined with efficient active set selection strategies. Besides, the library allows users to choose different sparsity-inducing regularizers, including the convex ℓ_1, nonvoncex MCP and SCAD regularizers. The library is coded in `C++` and has user-friendly `R` and `Python` wrappers. Numerical experiments demonstrate that `picasso` can scale up to large problems efficiently.

1. Overview

Sparse Learning arises due to the demand of analyzing high-dimensional data such as high-throughput genomic data (Neale et al., 2012) and functional Magnetic Resonance Imaging (Liu et al., 2015). The pathwise coordinate optimization is undoubtedly one the of the most popular solvers for a large variety of sparse learning problems. By leveraging the solution sparsity through a simple but elegant algorithmic structure, it significantly boosts the computational performance in practice (Friedman et al., 2007). Some recent progresses in (Zhao et al., 2017; Li et al., 2017) establish theoretical guarantees to further justify its computational and statistical superiority for both convex and nonvoncex sparse learning, which makes it even more attractive to practitioners.

We recently developed a new library named `picasso`, which implements a unified toolkit of pathwise coordinate optimization for solving a large class of convex and nonconvex regularized sparse learning problems. Efficient active set selection strategies are provided to guarantee superior statistical and computational preference. Specifically, we implement sparse linear regression, sparse logistic regression, sparse Poisson regression and scaled sparse linear regression (Tibshirani, 1996; Belloni et al., 2011; Sun and Zhang, 2012). The options of regularizers include the ℓ_1, MCP, and SCAD regularizers (Fan and Li, 2001; Zhang, 2010). Unlike existing libraries implementing heuristic optimization algorithms such as `ncvreg` or `glmnet` (Breheny, 2013; Friedman et al., 2010), our implemented algorithm `picasso` have strong theoretical guarantees that it attains a global linear convergence to a unique sparse local optimum with optimal statistical properties (e.g. minimax optimality and oracle properties). See more details in Zhao et al. (2017); Li et al. (2017).

2. Algorithm Design and Implementation

The algorithm implemented in `picasso` is mostly based on the generic pathwise coordinate optimization framework proposed by Zhao et al. (2017); Li et al. (2017), which integrates the warm start initialization, active set selection strategy, and strong rule for coordinate preselection into the classical coordinate optimization. The algorithm contains three structurally nested loops as shown in Figure 1:

(1) Outer loop: The warm start initialization, also referred to as the pathwise optimization scheme, is applied to minimize the objective function in a multistage manner using a sequence of decreasing regularization parameters, which yields a sequence of solutions from sparse to dense. At each stage, the algorithm uses the solution from the previous stage as initialization.

(2) Middle loop: The algorithm first divides all coordinates into active ones (active set) and inactive ones (inactive set) by a so-called strong rule based on coordinate gradient thresholding (Tibshirani et al., 2012). Then the algorithm calls an inner loop to optimize the objective, and update the active set based on efficient active set selection strategies. Such a routine is repeated until the active set no longer changes

(3) Inner loop: The algorithm conducts coordinate optimization (for sparse linear regression) or proximal Newton optimization combined with coordinate optimization (for sparse logistic regression, Possion regression, scaled sparse linear regression, sparse undirected graph estimation) only over active coordinates until convergence, with all inactive coordinates staying zero values. The active coordinates are updated efficiently using an efficient "naive update" rule that only operates on the non-zero coefficients. Better efficiency is achieved by the "covariance update" rule. See more details in (Friedman et al., 2010). The inner loop terminates when the successive descent is within a predefined numerical precision.

The warm start initialization, active set selection strategies, and strong rule for coordinate preselection significantly boost the computational performance, making pathwise coordinate optimization one of the most important computational frameworks for sparse learning. The numerical evaluations show that `picasso` is highly scalable and efficient.

The library is implemented in `C++` with the memory optimized using sparse matrix output, and called from `R` and `Python` by user-friendly interfaces. Linear algebra is supported by the Eigen3 library (Guennebaud et al., 2010) for portable high performance computation.

*. Jason Ge and Xingguo Li contributed equally. Tuo Zhao is the corresponding author.
1. More details can be found in our Github page: `https://github.com/jasonge27/picasso`

©2019 Jason Ge, Xingguo Li, Haoming Jiang, Han Liu, Tong Zhang, Mengdi Wang and Tuo Zhao.
License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v20/17-772.html`.

The implementation is modularized so that the algorithm in `src/solver/actnewton.cpp` works with popular sparsity-inducing regularizer functions and any convex objective function that exhibits restricted strong convexity property (Zhao et al., 2017). Users can easily extend the package by writing customized objective function subclass and regularizer function subclass following the virtual function interfaces of `class ObjFunction` and `class RegFunction` in `include/picasso/objective.hpp`.

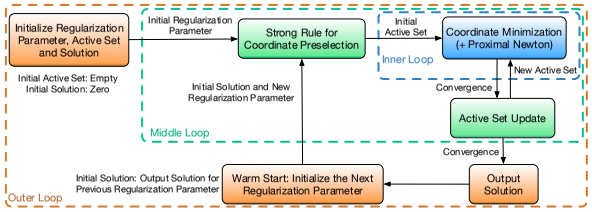


Figure 1: The pathwise coordinate optimization framework with 3 nested loops: (1) Warm start initialization; (2) Active set selection, and strong rule for coordinate preselection; (3) Active coordinate minimization.

3. Example of R User Interface

We illustrate the user interface by analyzing the eye disease data set in `picasso`.

```
> library(picasso); data(eyedata) # Load the data set
> out1 = picasso(x,y,method="l1",type.gaussian="naive",nlambda=20,
+                lambda.min.ratio=0.2) # Lasso
> out2 = picasso(x,y,method="mcp", gamma = 1.25, prec=1e-4)  # MCP regularizer
> plot(out1); plot(out2) # Plot solution paths
```

The program automatically generates a sequence of regularization parameters and estimate the corresponding solution paths based on the ℓ_1 and MCP regularizers respectively. For the ℓ_1 regularizer, we set the number of regularization parameters as 20, and the minimum regularization parameter as `0.2*lambda.max`. For the MCP regularizer, we set the concavity parameter as $\gamma = 1.25$, and the pre-defined accuracy as 10^{-4}. Here `nlambda` and `lambda.min.ratio` are omitted, and therefore set by the default values (`nlambda=100` and `lambda.min.ratio=0.05`). We further plot two solution paths in Figure 2.

4. Numerical Simulation

To demonstrate the superior efficiency of our library, we compare `picasso` with a popular `R` library `ncvreg` (version 3.9.1) for nonconvex regularized sparse regression, the most popular `R` library `glmnet` (version 2.0-13) for convex regularized sparse regression, and two `R` libraries `scalreg-v1.0` and `flare-v1.5.0` for scaled sparse linear regression. All experiments are evaluated on an Intel Core CPU i7-7700k 4.20GHz and under R version 3.4.3. Timings of the CPU execution are recored in seconds and averaged over 10 replications on a sequence of 100 regularization parameters. All algorithms are compared on the same regularization path and the convergence threshold are adjusted so that similar objective gaps are achieved.

We compare the timing performance and the optimization performance in Table 1. We choose the problem size to be $(n = 3000, d = 30000)$, where n is the number of observation and d is the dimension of the parameter vector. We tests the algorithms for both well-conditioned cases and ill-conditioned cases. The details of data generation can be found in the `R` library vignette. Here is our summary:

(1) For sparse linear regression using any regularizer and sparse logistic regression using the ℓ_1 regularizer, all libraries achieve almost identical optimization objective values, and `picasso` slightly outperforms `glmnet` and `ncvreg` in the timing performance.

(2) For sparse logistic regression using nonconvex regularizers, `picasso` achieves comparable objective value with `ncvreg`, and significantly outperforms `ncvreg` in timing performance. We also remark that `picasso` performs stably for various settings and tuning parameters. However, `ncvreg` may converge very slow or fail to converge for sparse logistic regression using nonconvex regularizers, especially when the tuning parameters are relatively small (corresponding to denser estimators), as the ill-conditioned SCAD case shows.

(3) For scaled Lasso, in order to make other competitors (`flare` and `scalreg`) converges in resonable time, we swtich to a smaller problem size $(n = 1000, d = 10000)$. We see that `picasso` much more time saving than `flare` and `scalreg`.

Table 1: Average timing performance (in seconds) with standard errors in the parentheses and achieved objective values.

		well-conditioned		ill-conditioned	
Sparse Linear Regression					
ℓ_1	picasso	1.61(0.03)s	27.691	3.62(0.02)s	32.543
	glmnet	4.15(0.03)s	27.692	9.43(0.01)s	32.537
	ncvreg	5.92(0.01)s	27.690	6.66(0.01)s	32.536
SCAD	picasso	1.56(0.01)s	27.668	3.74(0.01)s	33.133
	ncvreg	5.61(0.01)s	27.673	7.05(0.01)s	33.156
MCP	picasso	1.47(0.02)s	27.161	1.89(0.02)s	32.468
	ncvreg	4.07(0.03)s	27.161	2.56(0.01)s	32.468
Sparse Logistic Regression					
ℓ_1	picasso	2.03(0.01)s	0.363	2.10(0.03)s	0.327
	glmnet	16.32(0.12)s	0.363	20.31(0.02)s	0.327
	ncvreg	4.04(0.01)s	0.363	62.89(0.04)s	0.327
SCAD	picasso	4.25(0.01)s	0.227	4.35(0.02)s	0.172
	ncvreg	11.47(0.04)s	0.278	error (>300s)	
MCP	picasso	4.32(0.01)s	0.221	4.38(0.02)s	0.165
	ncvreg	9.37(0.08)s	0.248	6.99(0.01)s	0.242
Scaled Lasso					
ℓ_1	picasso	0.36(0.01)s	4.454	0.15(0.01)s	5.495
	flare	5.23(0.06)s	5.188	297.36(2.77)s	5.959
	scalreg	40.20(0.63)s	4.492	49.12(10.98)s	5.507

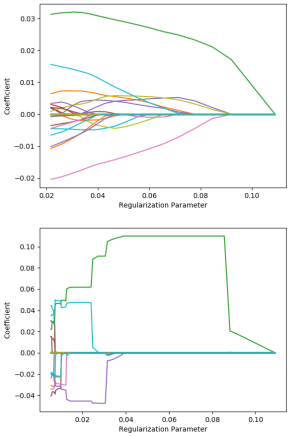


Figure 2: The solution paths of ℓ_1 (up) and MCP (down) regularizers.

5. Conclusion

The `picasso` library demonstrates significantly improved computational and statistical performance over existing libraries for nonconvex regularized sparse learning such as `ncvreg`. Besides, `picasso` also shows improvement over the popular libraries for convex regularized sparse learning such as `glmnet`. Overall, the `picasso` library has the potential to serve as a powerful toolbox for high dimensional sparse learning. We will continue to maintain and support this library.

References

A. Belloni, V. Chernozhukov, and L Wang. Square-root lasso: pivotal recovery of sparse signals via conic programming. *Biometrika*, 98(4):791–806, 2011.

P Breheny. ncvreg: Regularization paths for scad-and mcp-penalized regression models. *R package version*, pages 2–6, 2013.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1–13, 2010.

Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

Xingguo Li, Jian Ge, Haoming Jiang, Mengdi Wang, Mingyi Hong, and Tuo Zhao. Boosting pathwise coordinate optimization in high dimensions: Sequential screening and proximal sub-sampled newton algorithm. Technical report, Georgia Tech, 2017.

Han Liu, Lie Wang, and Tuo Zhao. Calibrated multivariate regression with application to neural semantic basis discovery. *Journal of Machine Learning Research*, 16:1579–1606, 2015.

Benjamin M Neale, Yan Kou, Li Liu, Avi Ma'Ayan, Kaitlin E Samocha, Aniko Sabo, Chiao-Feng Lin, Christine Stevens, Li-San Wang, Vladimir Makarov, et al. Patterns and rates of exonic de novo mutations in autism spectrum disorders. *Nature*, 485(7397):242, 2012.

T. Sun and C.H Zhang. Scaled sparse linear regression. *Biometrika*, 2012. to appear.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

R. Tibshirani, J. Bien, J. Friedman, T. Hastie, N. Simon, J. Taylor, and R. Tibshirani. Strong rules for discarding predictors in lasso-type problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(2):245–266, 2012.

C. Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 38(2):894–942, 2010.

T. Zhao, H. Liu, and T. Zhang. Pathwise coordinate optimization for nonconvex sparse learning: Algorithm and theory. *Annals of Statistics*, 2017.

Journal of Machine Learning Research 20 (2019) 1-41 Submitted 12/17; Revised 8/18; Published 2/19

Robust Frequent Directions with Application in Online Learning

Luo Luo RICKY@SJTU.EDU.CN
Cheng Chen JACK_CHEN1990@SJTU.EDU.CN
Department of Computer Science and Engineering
Shanghai Jiao Tong University
800 Dongchuan Road, Shanghai, China 200240

Zhihua Zhang* ZHZHANG@MATH.PKU.EDU.CN
National Engineering Lab for Big Data Analysis and Applications
School of Mathematical Sciences
Peking University
5 Yiheyuan Road, Beijing, China 100871

Wu-Jun Li LIWUJUN@NJU.EDU.CN
National Key Laboratory for Novel Software Technology
Collaborative Innovation Center of Novel Software Technology and Industrialization
Department of Computer Science and Technology
Nanjing University
163 Xianlin Avenue, Nanjing, China 210023

Tong Zhang TONGZHANG@TONGZHANG-ML.ORG
Computer Science & Mathematics
Hong Kong University of Science and Technology
Hong Kong

Editor: Qiang Liu

Abstract

The frequent directions (FD) technique is a deterministic approach for online sketching that has many applications in machine learning. The conventional FD is a heuristic procedure that often outputs rank deficient matrices. To overcome the rank deficiency problem, we propose a new sketching strategy called robust frequent directions (RFD) by introducing a regularization term. RFD can be derived from an optimization problem. It updates the sketch matrix and the regularization term adaptively and jointly. RFD reduces the approximation error of FD without increasing the computational cost. We also apply RFD to online learning and propose an effective hyperparameter-free online Newton algorithm. We derive a regret bound for our online Newton algorithm based on RFD, which guarantees the robustness of the algorithm. The experimental studies demonstrate that the proposed method outperforms state-of-the-art second order online learning algorithms.

Keywords: Matrix approximation, sketching, frequent directions, online convex optimization, online Newton algorithm

1. Introduction

The sketching technique is a powerful tool to deal with large scale matrices (Ghashami et al., 2016; Halko et al., 2011; Woodruff, 2014), and it has been widely used to speed up machine learning algorithms such as second order optimization algorithms (Erdogdu and Montanari, 2015; Luo et al., 2016; Pilanci and Wainwright, 2017; Roosta-Khorasani and Mahoney, 2016a,b; Xu et al., 2016; Ye et al., 2017). There exist several families of matrix sketching strategies, including sparsification, column sampling, random projection (Achlioptas, 2003; Indyk and Motwani, 1998; Kane and Nelson, 2014; Wang et al., 2016), and frequent directions (FD) (Desai et al., 2016; Ghashami et al., 2016; Huang, 2018; Liberty, 2013; Mroueh et al., 2017; Ye et al., 2016).

Sparsification techniques (Achlioptas and McSherry, 2007; Achlioptas et al., 2013; Arora et al., 2006; Drineas and Zouzias, 2011) generate a sparse version of the matrix by element-wise sampling, which allows the matrix multiplication to be more efficient with lower space. Column (row) sampling algorithms (Mahoney, 2011) include the importance sampling (Drineas et al., 2006a,b; Frieze et al., 2004) and leverage score sampling (Drineas et al., 2012, 2008; Papailiopoulos et al., 2014). They define a probability for each row (column) and select a subset by the probability to construct the estimation. Random projection maps the rows (columns) of the matrix into lower dimensional space by a projection matrix. The projection matrix can be constructed in various ways (Woodruff, 2014) such as Gaussian random projections (Johnson and Lindenstrauss, 1984; Sarlos, 2006), fast Johnson-Lindenstrauss transforms (Ailon and Chazelle, 2006; Ailon and Liberty, 2009, 2013; Kane and Nelson, 2014) and sparse random projections (Clarkson and Woodruff, 2013; Nelson and Nguyên, 2013). The frequent directions (Ghashami et al., 2016; Liberty, 2013) is a deterministic sketching algorithm and achieves optimal tradeoff between approximation error and space.

In this paper we are especially concerned with the FD sketching (Desai et al., 2016; Ghashami et al., 2016; Liberty, 2013), because it is a stable online sketching approach. FD considers the matrix approximation in the streaming setting. In this case, the data is available in a sequential order and should be processed immediately. Typically, streaming algorithms can only use limited memory at any time. The FD algorithm extends the method of frequent items (Misra and Gries, 1982) to matrix approximation and has tight approximation error bound. However, FD usually leads to a rank deficient approximation, which in turn makes its applications less robust. For example, Newton-type algorithms require a non-singular and well-conditioned approximated Hessian matrix but FD sketching usually generates low-rank matrices. An intuitive and simple way to conquer this gap is to introduce a regularization term to enforce the matrix to be invertible (Luo et al., 2016; Roosta-Khorasani and Mahoney, 2016a,b). Typically, the regularization parameter is regarded as a hyperparameter and its choice is separable from the sketching procedure. Since the regularization parameter affects the the performance heavily in practice, it should be chosen carefully.

To overcome the weakness of the FD algorithm, we propose a new sketching approach that we call *robust frequent directions* (RFD). Unlike conventional sketching methods which only approximate the matrix with a low-rank structure, RFD constructs the low-rank part and updates the regularization term simultaneously. In particular, the update procedure of RFD can be regarded as solving an optimization problem (see Theorem 2). This method is different from the standard FD, giving rise to a tighter error bound.

Note that Zhang (2014) proposed matrix ridge approximation (MRA) to approximate a positive semi-definite matrix using an idea similar to RFD. There are two main differences between RFD and MRA. First, RFD is designed for the case that data samples come sequentially and memory is limited, while MRA has to access the whole data set. Second, MRA aims to minimize the approximation error with respect to the Frobenius norm while RFD tries to minimize the spectral-norm approximation error. In general, the spectral norm error bound is more meaningful than the Frobenius norm error bound (Tropp, 2015).

∗. Corresponding author.

©2019 Luo Luo, Cheng Chen, Zhihua Zhang, Wu-Jun Li and Tong Zhang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-773.html.

In a recent study, Luo et al. (2016) proposed a FD-based sketched online Newton (SON) algorithm (FD-SON) to accelerate the standard online Newton algorithms. Owing to the shortcoming of FD, the performance of FD-SON is significantly affected by the choice of the hyperparameter. Naturally, we can leverage RFD to improve online Newton algorithms. Accordingly, we propose a sketched online Newton step based on RFD (RFD-SON). Different from conventional sketched Newton algorithms, RFD-SON is hyperparameter-free. Setting the regularization parameter to be zero initially, RFD-SON will adaptively increase the regularization term. The approximation Hessian will be well-conditioned after a few iterations. Moreover, we prove that RFD-SON has a more robust regret bound than FD-SON, and the experimental results also validate better performance of RFD-SON.

The remainder of the paper is organized as follows. In Section 2 we present notation and preliminaries. In Section 3 we review the background of second order online learning and its sketched variants. In Sections 4 and 5 we propose our robust frequent directions (RFD) method and the applications in online learning, with some related theoretical analysis. In Section 6 we demonstrate empirical comparisons with baselines on serval real-world data sets to show the superiority of our algorithms. Finally, we conclude our work in Section 7.

2. Notation and Preliminaries

We let $\mathbf{I}_d$ denote the $d\times d$ identity matrix. For a matrix $\mathbf{A} = [A_{ij}] \in \mathbb{R}^{n\times d}$ of rank r where $r \leq \min(n,d)$, we let the condensed singular value decomposition (SVD) of $\mathbf{A}$ be $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ where $\mathbf{U} \in \mathbb{R}^{n\times r}$ and $\mathbf{V} \in \mathbb{R}^{d\times r}$ are column orthogonal and $\mathbf{\Sigma} = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r)$ with $\sigma_1 \geq \sigma_2 \geq \ldots, \geq \sigma_r > 0$ places the nonzero singular values on its diagonal entries.

We use $\sigma_{\max}(\mathbf{A})$ to denote the largest singular value and $\sigma_{\min}(\mathbf{A})$ to denote the smallest non-zero singular value. Thus, the condition number of $\mathbf{A}$ is $\kappa(\mathbf{A}) = \frac{\sigma_{\max}(\mathbf{A})}{\sigma_{\min}(\mathbf{A})}$. The matrix pseudoinverse of $\mathbf{A}$ is defined by $\mathbf{A}^\dagger = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^\top \in \mathbb{R}^{d\times n}$.

Additionally, we let $\|\mathbf{A}\|_F = \sqrt{\sum_{i,j} A_{ij}^2} = \sqrt{\sum_{i=1}^r \sigma_i^2}$ be the Frobenius norm and $\|\mathbf{A}\|_2 = \sigma_{\max}(\mathbf{A})$ be the spectral norm. A matrix norm $\|\cdot\|$ is said to be unitarily invariant if $\|\mathbf{PAQ}\| = \|\mathbf{A}\|$ for any unitary matrices $\mathbf{P} \in \mathbb{R}^{n\times n}$ and $\mathbf{Q} \in \mathbb{R}^{d\times d}$. It is easy to verify that both the Frobenius norm and spectral norm are unitarily invariant. We define $[\mathbf{A}]_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^\top$ for $k \leq r$, where $\mathbf{U}_k \in \mathbb{R}^{n\times k}$ and $\mathbf{V}_k \in \mathbb{R}^{d\times k}$ are the first k columns of $\mathbf{U}$ and $\mathbf{V}$, and $\mathbf{\Sigma}_k = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k) \in \mathbb{R}^{k\times k}$. Then $[\mathbf{A}]_k$ is the best rank-k approximation to $\mathbf{A}$ in both the Frobenius and spectral norms, that is,

$$[\mathbf{A}]_k = \operatorname*{argmin}_{\operatorname{rank}(\widehat{\mathbf{A}})\leq k} \|\mathbf{A} - \widehat{\mathbf{A}}\|_F = \operatorname*{argmin}_{\operatorname{rank}(\widehat{\mathbf{A}})\leq k} \|\mathbf{A} - \widehat{\mathbf{A}}\|_2.$$

Given a positive semidefinite matrix $\mathbf{H} \in \mathbb{R}^{d\times d}$, the notation $\|\mathbf{x}\|_\mathbf{H}$ is called $\mathbf{H}$-norm of vector $\mathbf{x} \in \mathbb{R}^d$, that is, $\|\mathbf{x}\|_\mathbf{H} = \sqrt{\mathbf{x}^\top\mathbf{H}\mathbf{x}}$. If matrices $\mathbf{A}$ and $\mathbf{B}$ have the same size, we let $\langle \mathbf{A}, \mathbf{B}\rangle$ denote $\sum_{i,j} A_{ij}B_{ij}$.

2.1. Frequent Directions

We give a brief review of frequent directions (FD) (Ghashami et al., 2016; Liberty, 2013), because it is closely related to our proposed method. FD is a deterministic matrix sketching in the row-updates model. For any input matrix $\mathbf{A} \in \mathbb{R}^{T\times d}$ whose rows come sequentially, it maintains a sketch matrix $\mathbf{B} \in \mathbb{R}^{(m-1)\times d}$ with $m \ll T$ to approximate $\mathbf{A}^\top\mathbf{A}$ by $\mathbf{B}^\top\mathbf{B}$.

We present the detailed implementation of FD in Algorithm 1. The intuition behind FD is similar to that of frequent items. FD periodically shrinks orthogonal vectors by roughly the same amount (Line 5 of Algorithm 1). The shrinking step reduces the square Frobenius norm of the sketch reasonable and guarantees that no direction is reduced too much.

Algorithm 1 Frequent Directions

1: **Input:** $\mathbf{A} = [\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(T)}]^\top \in \mathbb{R}^{T\times d}$, $\mathbf{B}^{(m-1)} = [\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(m-1)}]^\top$
2: **for** $t = m, \ldots, T$ **do**
3: $\quad\widehat{\mathbf{B}}^{(t-1)} = \begin{bmatrix}\mathbf{B}^{(t-1)} \\ (\mathbf{a}^{(t)})^\top\end{bmatrix}$
4: $\quad$Compute SVD: $\widehat{\mathbf{B}}^{(t-1)} = \mathbf{U}^{(t-1)}\mathbf{\Sigma}^{(t-1)}(\mathbf{V}^{(t-1)})^\top$
5: $\quad\mathbf{B}^{(t)} = \sqrt{\left(\mathbf{\Sigma}_{m-1}^{(t-1)}\right)^2 - \left(\sigma_m^{(t-1)}\right)^2\mathbf{I}_{m-1}} \cdot \left(\mathbf{V}_{m-1}^{(t-1)}\right)^\top$
6: **end for**
7: **Output:** $\mathbf{B} = \mathbf{B}^{(T)}$

FD has the following error bound for any $k < m$,

$$\|\mathbf{A}^\top\mathbf{A} - \mathbf{B}^\top\mathbf{B}\|_2 \leq \frac{1}{m-k}\|\mathbf{A} - [\mathbf{A}]_k\|_F^2. \tag{1}$$

The above result means that the space complexity of FD is optimal regardless of streaming issues because any algorithm satisfying $\|\mathbf{A}^\top\mathbf{A} - \mathbf{B}^\top\mathbf{B}\|_2 \leq \|\mathbf{A} - [\mathbf{A}]_k\|_F^2/(m-k)$ requires $\mathcal{O}(md)$ space to represent matrix $\mathbf{B}$ (Ghashami et al., 2016). The dominated computation of the algorithm is computating the SVD of $\widehat{\mathbf{B}}^{(t-1)}$, which costs $\mathcal{O}(m^2d)$ by the standard SVD implementation. However, the total cost can be reduced from $\mathcal{O}(Tm^2d)$ to $\mathcal{O}(Tmd)$ by doubling the space (Algorithm 4 in Appendix A) or using the Gu-Eisenstat procedure (Gu and Eisenstat, 1993).

Desai et al. (2016) proposed some extensions of FD. More specifically, Parameterized FD (PFD) uses an extra hyperparameter to describe the proportion of singular values shrunk in each iteration. PFD improves the performance empirically, but has worse error bound than FD by a constant. Compensative FD (CFD) modifies the output of FD by increasing the singular values and keeps the same error guarantees as FD.

3. Online Newton Methods

For ease of demonstrating our work, we would like to introduce sketching techniques in online learning scenarios. First of all, we introduce the background of convex online learning including online Newton step algorithms. Then we discuss the connection between online learning and sketched second order methods, which motivates us to propose a more robust sketching algorithm.

3.1. Convex Online Learning

Online learning is performed in a sequence of consecutive rounds (Shalev-Shwartz, 2011). We consider the problem of convex online optimization as follows. For a sequence of examples $\{\mathbf{x}^{(t)} \in \mathbb{R}^d\}$, and convex smooth loss functions $\{f_t : \mathcal{K}_t \to \mathbb{R}\}$ where $f_t(\mathbf{w}) \triangleq \ell_t(\mathbf{w}^\top\mathbf{x}^{(t)})$ and $\mathcal{K}_t \subset \mathbb{R}^d$

are convex compact sets, the learner outputs a predictor $\mathbf{w}^{(t)}$ and suffers the loss $f_t(\mathbf{w}^{(t)})$ at the t-th round. The cumulative regret at round T is defined as:

$$R_T(\mathbf{w}^*) = \sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) - \sum_{t=1}^{T} f_t(\mathbf{w}^*),$$

where $\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}\in\mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{w})$ and $\mathcal{K} = \bigcap_{t=1}^{T} \mathcal{K}_t$.

We make the following assumptions on the loss functions.

Assumption 1 *The loss functions ℓ_t satisfy $|\ell_t'(z)| \leq L$ whenever $|z| \leq C$, where L and C are positive constants.*

Assumption 2 *There exists a $\mu_t \leq 0$ such that for all $\mathbf{u}, \mathbf{w} \in \mathcal{K}$, we have*

$$f_t(\mathbf{w}) \geq f_t(\mathbf{u}) + \nabla f_t(\mathbf{u})^\top(\mathbf{w}-\mathbf{u}) + \frac{\mu_t}{2}\big(\nabla f_t(\mathbf{u})^\top(\mathbf{w}-\mathbf{u})\big)^2.$$

Note that for a loss function f_t whose domain and gradient have bounded diameter, holding Assumption 2 only requires the exp-concave property, which is more general than strong convexity (Hazan, 2016). For example, the square loss function $f_t(\mathbf{w}) = (\mathbf{w}^\top\mathbf{x}_t - y_t)^2$ satisfies Assumption 2 with $\mu_t = \frac{1}{8C^2}$ if the function is subject to constraints $|\mathbf{w}^\top\mathbf{x}_t| \leq C$ and $y_t \leq C$ (Luo et al., 2016), but it is not strongly convex.

One typical online learning algorithm is online gradient descent (OGD) (Hazan et al., 2007; Zinkevich, 2003). At the $(t+1)$-th round, OGD exploits the following update rules:

$$\begin{aligned}\mathbf{u}^{(t+1)} &= \mathbf{w}^{(t)} - \beta_t \mathbf{g}^{(t)},\\ \mathbf{w}^{(t+1)} &= \operatorname*{argmin}_{\mathbf{w}\in\mathcal{K}_{t+1}} \|\mathbf{w} - \mathbf{u}^{(t+1)}\|,\end{aligned}$$

where $\mathbf{g}^{(t)} = \nabla f_t(\mathbf{w}^{(t)})$ and β_t is the learning rate. The algorithm has linear computation cost and achieves $\mathcal{O}(\frac{L^2}{H}\log T)$ regret bound for the H-strongly convex loss.

In this paper, we are more interested in online Newton step algorithms (Hazan et al., 2007; Luo et al., 2016). The standard online Newton step keeps the curvature information in the matrix $\mathbf{H}^{(t)} \in \mathbb{R}^{d\times d}$ sequentially and iterates as follows:

$$\begin{aligned}\mathbf{u}^{(t+1)} &= \mathbf{w}^{(t)} - \beta_t (\mathbf{H}^{(t)})^{-1}\mathbf{g}^{(t)},\\ \mathbf{w}^{(t+1)} &= \operatorname*{argmin}_{\mathbf{w}\in\mathcal{K}_{t+1}} \|\mathbf{w} - \mathbf{u}^{(t+1)}\|_{\mathbf{H}^{(t)}}.\end{aligned} \tag{2}$$

The matrix $\mathbf{H}^{(t)}$ is constructed by the outer product of historical gradients (Duchi et al., 2011; Luo et al., 2016), such as

$$\mathbf{H}^{(t)} = \sum_{i=1}^{t} \mathbf{g}^{(i)}(\mathbf{g}^{(i)})^\top + \alpha_0\mathbf{I}_d, \tag{3}$$

$$\text{or}\quad \mathbf{H}^{(t)} = \sum_{i=1}^{t} (\mu_t + \eta_t)\mathbf{g}^{(i)}(\mathbf{g}^{(i)})^\top + \alpha_0\mathbf{I}_d, \tag{4}$$

where $\alpha_0 \geq 0$ is a fixed regularization parameter, μ_t is the constant in Assumption 2, and η_t is typically chosen as $\mathcal{O}(1/t)$. The second order algorithms enjoy logarithmical regret bound without the strongly convex assumption but require quadratical space and computation cost. Some variants of online Newton algorithms have been applied to optimize neural networks (Martens and Grosse, 2015; Grosse and Martens, 2016; Ba et al., 2017), but they do not provide theoretical guarantee on nonconvex cases.

3.2. Efficient Algorithms by Sketching

To make the online Newton step scalable, it is natural to use sketching techniques (Woodruff, 2014). The matrix $\mathbf{H}^{(t)}$ in online learning has the form $\mathbf{H}^{(t)} = (\mathbf{A}^{(t)})^\top\mathbf{A}^{(t)} + \alpha_0\mathbf{I}_d$, where $\mathbf{A}^{(t)} \in \mathbb{R}^{t\times d}$ is the corresponding term of (3) or (4) such as

$$\mathbf{A}^{(t)} = [\mathbf{g}^{(1)}, \ldots, \mathbf{g}^{(t)}]^\top, \quad \text{or} \quad \mathbf{A}^{(t)} = [\sqrt{\mu_1+\eta_1}\mathbf{g}^{(1)}, \ldots, \sqrt{\mu_t+\eta_t}\mathbf{g}^{(t)}]^\top.$$

The sketching algorithm employs an approximation of $(\mathbf{A}^{(t)})^\top\mathbf{A}^{(t)}$ by $(\mathbf{B}^{(t)})^\top\mathbf{B}^{(t)}$, where the sketch matrix $\mathbf{B}^{(t)} \in \mathbb{R}^{m\times d}$ is much smaller than $\mathbf{A}^{(t)}$ and $m \ll d$. Then we can use $(\mathbf{B}^{(t)})^\top\mathbf{B}^{(t)} + \alpha_0\mathbf{I}_d$ to replace $\mathbf{H}^{(t)}$ in update (2) (Luo et al., 2016). By the Woodbury identity formula, we can reduce the computation of the update from $\mathcal{O}(d^2)$ to $\mathcal{O}(m^2d)$ or $\mathcal{O}(md)$. There are several choices of sketching techniques, such as random projection (Achlioptas, 2003; Indyk and Motwani, 1998; Kane and Nelson, 2014), frequent directions (Ghashami et al., 2016; Liberty, 2013) and Oja's algorithm (Oja, 1982; Oja and Karhunen, 1985). However, all above methods treat α_0 as a given hyperparameter which is independent of the sketch matrix $\mathbf{B}^{(t)}$. In practice, the performance of sketched online Newton methods is sensitive to the choice of the hyperparamter α_0.

4. Robust Frequent Directions

In many machine learning applications such as online learning (Hazan and Arora, 2006; Hazan et al., 2007; Hazan, 2016; Luo et al., 2016), Gaussian process regression (Rasmussen and Williams, 2006) and kernel ridge regression (Drineas and Mahoney, 2005), we usually require an additional regularization term to make the matrix invertible and well-conditioned, while conventional sketching methods only focus on the low-rank approximation. On the other hand, the update of frequent directions is not optimal in the view of minimizing the approximation error in each iteration. Both of them motivate us to propose robust frequent directions (RFD) that incorporates the update of sketch matrix and the regularization term into one framework.

4.1. The Algorithm

The RFD approximates $\mathbf{A}^\top\mathbf{A}$ by $\mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d$ with $\alpha > 0$. We demonstrate the detailed implementation of RFD in Algorithm 2.

The main difference between RFD and conventional sketching algorithms is the additional term $\alpha\mathbf{I}_d$. We can directly use Algorithm 2 to approximate $\mathbf{A}^\top\mathbf{A}$ with $\alpha^{(m-1)} = \alpha_0 > 0$ if the target matrix is $\mathbf{A}^\top\mathbf{A} + \alpha_0\mathbf{I}_d$. Compared with the standard FD, RFD only needs to maintain one extra variable $\alpha^{(t)}$ by scalar operations in each iteration, hence the cost of RFD is almost the same as FD. Because the value of $\alpha^{(t)}$ is typically increasing from the $(m+1)$-th round in practice, the resulting $\mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d$ is positive definite even the initial $\alpha^{(0)}$ is zero. Also, we can further accelerate the algorithm by doubling the space.

Algorithm 2 Robust Frequent Directions

Input: $\mathbf{A} = [\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(T)}]^\top \in \mathbb{R}^{T\times d}$, $\mathbf{B}^{(m-1)} = [\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(m-1)}]^\top$, $\alpha^{(m-1)} = 0$
for $t = m, \ldots, T$ **do**
  $\widehat{\mathbf{B}}^{(t-1)} = \begin{bmatrix} \mathbf{B}^{(t-1)} \\ (\mathbf{a}^{(t)})^\top \end{bmatrix}$
  Compute SVD: $\widehat{\mathbf{B}}^{(t-1)} = \mathbf{U}^{(t-1)}\mathbf{\Sigma}^{(t-1)}(\mathbf{V}^{(t-1)})^\top$
  $\mathbf{B}^{(t)} = \sqrt{(\mathbf{\Sigma}_{m-1}^{(t-1)})^2 - (\sigma_m^{(t-1)})^2\mathbf{I}_{m-1}} \cdot (\mathbf{V}_{m-1}^{(t-1)})^\top$
  $\alpha^{(t)} = \alpha^{(t-1)} + (\sigma_m^{(t-1)})^2/2$
end for
Output: $\mathbf{B} = \mathbf{B}^{(T)}$ and $\alpha = \alpha^{(T)}$.

4.2. Theoretical Analysis

Before demonstrating the theoretical results of RFD, we review FD from the aspect of low-rank approximation which provides a motivation to the design of our algorithm. At the t-th round iteration of FD (Algorithm 1), we have the matrix $\mathbf{B}^{(t-1)}$ which is used to approximate $(\mathbf{A}^{(t-1)})^\top\mathbf{A}^{(t-1)}$ by $(\mathbf{B}^{(t-1)})^\top\mathbf{B}^{(t-1)}$ and we aim to construct a new approximation which includes the new data $\mathbf{a}^{(t)}$, that is,

$$(\mathbf{B}^{(t)})^\top\mathbf{B}^{(t)} \approx (\mathbf{B}^{(t-1)})^\top\mathbf{B}^{(t-1)} + \mathbf{a}^{(t)}(\mathbf{a}^{(t)})^\top = (\widehat{\mathbf{B}}^{(t-1)})^\top\widehat{\mathbf{B}}^{(t-1)}. \tag{5}$$

The straightforward way to find $\mathbf{B}^{(t)}$ is to minimize the approximation error of (5) based on the spectral norm with low-rank constraint:

$$\mathbf{B}'^{(t)} = \operatorname*{argmin}_{\mathrm{rank}(\mathbf{C})=m-1} \left\|(\widehat{\mathbf{B}}^{(t-1)})^\top\widehat{\mathbf{B}}^{(t-1)} - \mathbf{C}^\top\mathbf{C}\right\|_2. \tag{6}$$

By the SVD of $\widehat{\mathbf{B}}^{(t-1)}$, we have the solution $\mathbf{B}'^{(t)} = \mathbf{\Sigma}_{m-1}^{(t-1)}(\mathbf{V}_{m-1}^{(t-1)})^\top$. In this view, the update of FD

$$\mathbf{B}^{(t)} = \sqrt{(\mathbf{\Sigma}_{m-1}^{(t-1)})^2 - (\sigma_m^{(t-1)})^2\mathbf{I}_{m-1}} \cdot (\mathbf{V}_{m-1}^{(t-1)})^\top \tag{7}$$

looks imperfect, because it is not an optimal low-rank approximation. However, the shrinkage operation in (7) is necessary. If we take a greedy strategy (Brand, 2002; Hall et al., 1998; Levey and Lindenbaum, 2000; Ross et al., 2008) which directly replaces $\mathbf{B}^{(t)}$ with $\mathbf{B}'^{(t)}$ in FD, it will perform worse in some specific cases[1] and also has no valid global error bound like (1).

Hence, the question is: can we devise a method which enjoys the optimality in each step and maintains global tighter error bound in the same time? Fortunately, RFD is just such an algorithm holding both the properties. We now explain the update rule of RFD formally, and provide the approximation error bound. We first give the following theorem which plays an important role in our analysis.

1. We provide an example in Appendix F.

Theorem 1 *Given a positive semi-definite matrix $\mathbf{M} \in \mathbb{R}^{d\times d}$ and a positive integer $k < d$, let $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^\top$ be the SVD of $\mathbf{M}$. Let $\mathbf{U}_k$ denote the matrix of the first k columns of $\mathbf{U}$ and σ_k be the k-th singular value of $\mathbf{M}$. Then the pair $(\widehat{\mathbf{C}}, \widehat{\delta})$, defined as*

$$\widehat{\mathbf{C}} = \mathbf{U}_k(\mathbf{\Sigma}_k - \xi\mathbf{I}_k)^{1/2}\mathbf{Q} \quad \textit{and} \quad \widehat{\delta} = (\sigma_{k+1} + \sigma_d)/2$$

where $\xi \in [\sigma_d, \sigma_{k+1}]$ and $\mathbf{Q}$ is an arbitrary $k \times k$ orthonormal matrix, is the global minimizer of

$$\min_{\mathbf{C}\in\mathbb{R}^{d\times k}, \delta\in\mathbb{R}} \|\mathbf{M} - (\mathbf{C}\mathbf{C}^\top + \delta\mathbf{I}_d)\|_2. \tag{8}$$

Additionally, we have

$$\|\mathbf{M} - (\widehat{\mathbf{C}}\widehat{\mathbf{C}}^\top + \widehat{\delta}\mathbf{I}_d)\|_2 \le \|\mathbf{M} - \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{U}_k^\top\|_2,$$

and the equality holds if and only if $\mathrm{rank}(\mathbf{M}) \le k$.

Theorem 1 provides the optimal solution with the closed form for matrix approximation with a regularization term. In the case of $\mathrm{rank}(\mathbf{M}) > k$, the approximation $\widehat{\mathbf{C}}\widehat{\mathbf{C}}^\top + \widehat{\delta}\mathbf{I}_d$ is full rank and has strictly lower spectral norm error than the rank-k truncated SVD. Note that Zhang (2014) has established the Frobenius norm based result about the optimal analysis[2].

Recall that in the streaming case, our goal is to approximate the concentration of historical approximation and current data at the t-th round. The following theorem shows that the update of RFD is optimal with respect to the spectral norm for each step.

Theorem 2 *Based on the updates in Algorithm 2, we have*

$$(\mathbf{B}^{(t)}, \alpha^{(t)}) = \operatorname*{argmin}_{\mathbf{B}\in\mathbb{R}^{d\times(m-1)}, \alpha\in\mathbb{R}} \left\|(\widehat{\mathbf{B}}^{(t-1)})^\top\widehat{\mathbf{B}}^{(t-1)} + \alpha^{(t-1)}\mathbf{I}_d - (\mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d)\right\|_2. \tag{9}$$

Theorem 2 explains RFD from an optimization viewpoint. It shows that each step of RFD is optimal for current information. Based on this theorem, the update of the standard FD corresponds $(\mathbf{B}, \alpha) = (\mathbf{B}^{(t)}, 0)$, which is not the optimal solution of (9). Intuitively, the regularization term of RFD compensates each direction for the over reduction from the shrinkage operation of FD. Theorem 2 also implies RFD is an online extension to the approximation of Theorem 1. We can prove Theorem 2 by using Theorem 1 with $\mathbf{M} = (\widehat{\mathbf{B}}^{(t-1)})^\top\widehat{\mathbf{B}}^{(t-1)} + \alpha^{(t-1)}\mathbf{I}_d$. We defer the details to Appendix C.

RFD also enjoys a tighter approximation error than FD as the following theorem shows.

Theorem 3 *For any $k < m$ and using the notation of Algorithm 2, we have*

$$\left\|\mathbf{A}^\top\mathbf{A} - (\mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d)\right\|_2 \le \frac{1}{2(m-k)}\|\mathbf{A} - [\mathbf{A}]_k\|_F^2, \tag{10}$$

where $[\mathbf{A}]_k$ is the best rank-k approximation to $\mathbf{A}$ in both the Frobenius and spectral norms.

The right-hand side of inequality (10) is the half of the one in (1), which means RFD reduces the approximation error significantly with only one extra scalar.

The real applications usually consider the matrix with a regularization term. Hence we also consider approximating the matrix $\mathbf{M} = \mathbf{A}^\top\mathbf{A} + \alpha_0\mathbf{I}_d$ where $\alpha_0 > 0$ and the rows of $\mathbf{A}$ are available

2. We also give a concise proof for the result of Zhang (2014) in Appendix B.

in sequentially order. Suppose that the standard FD approximates $\mathbf{A}^\top\mathbf{A}$ by $\mathbf{B}^\top\mathbf{B}$. Then it estimates $\mathbf{M}$ as $\mathbf{M}_{\text{FD}} = \mathbf{B}^\top\mathbf{B} + \alpha_0\mathbf{I}_d$. Meanwhile, RFD generates the approximation $\mathbf{M}_{\text{RFD}} = \mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d$ by setting $\alpha^{(m-1)} = \alpha_0$. Theorem 4 shows that the condition number of $\mathbf{M}_{\text{RFD}}$ is better than $\mathbf{M}_{\text{FD}}$ and $\mathbf{M}$. In general, the equality in Theorem 4 usually can not be achieved for $t > m$ unless $(\mathbf{a}^{(t)})^\top$ lies in the row space of $\mathbf{B}^{(t-1)}$ exactly or the first t rows of $\mathbf{A}$ have perfect low rank structure. Hence RFD is more likely to generate a well-conditioned approximation than others.

Theorem 4 *With the notation of Algorithms 1 and 2, let* $\mathbf{M} = \mathbf{A}^\top\mathbf{A} + \alpha_0\mathbf{I}_d$, $\mathbf{M}_{\text{FD}} = \mathbf{B}^\top\mathbf{B} + \alpha_0\mathbf{I}_d$, $\mathbf{M}_{\text{RFD}} = \mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d$ *and* $\alpha^{(m-1)} = \alpha_0$, *where* $\alpha_0 > 0$ *is a fixed scalar. Then we have* $\kappa(\mathbf{M}_{\text{RFD}}) \leq \kappa(\mathbf{M}_{\text{FD}})$ *and* $\kappa(\mathbf{M}_{\text{RFD}}) \leq \kappa(\mathbf{M})$.

5. The Online Newton Step by RFD

We now present the sketched online Newton step by robust frequent directions (RFD-SON). The procedure is shown in Algorithm 3, which is similar to sketched online Newton step (SON) algorithms (Luo et al., 2016) but uses the new sketching method RFD. The matrix $\mathbf{H}^{(t)}$ in Line 10 will not be constructed explicitly in practice, which is only used to the ease of analysis. The updates of $\mathbf{u}^{(t)}$ and $\mathbf{w}^{(t)}$ can be finished in $\mathcal{O}(md)$ time and space complexity by the Woodbury identity. We demonstrate the details in Appendix . When d is large, RFD-SON is much efficient than the standard online Newton step with the full Hessian that requires $\mathcal{O}(d^2)$ both in time and space.

Note that we do not require the hyperparameter α_0 to be strictly positive in RFD-SON. In practice, RFD-SON always archives good performance by setting $\alpha_0 = 0$, which leads to a hyperparameter-free algorithm, while the existing SON algorithm needs to select α_0 carefully. We consider the general case that $\alpha_0 \geq 0$ in this section for the ease of analysis.

Theorem 5 *Let* $\mu = \min_{t=1}^T\{\mu_t\}$ *and* $\mathcal{K} = \bigcap_{t=1}^T\mathcal{K}_t$. *Then under Assumptions 1 and 2 for any* $\mathbf{w} \in \mathcal{K}$, *Algorithm 3 has the following regret for* $\alpha_0 > 0$

$$R_T(\mathbf{w}) \leq \frac{\alpha_0}{2}\|\mathbf{w}\|^2 + 2(CL)^2\sum_{t=1}^T\eta_t + \frac{m}{2(\mu+\eta_T)}\ln\Big(\frac{\text{tr}\big((\mathbf{B}^{(T)})^\top\mathbf{B}^{(T)}\big)}{m\alpha_0} + \frac{\alpha^{(T)}}{\alpha_0}\Big) + \Omega_{\text{RFD}} \tag{11}$$

where

$$\Omega_{\text{RFD}} = \frac{d-m}{2(\mu+\eta_T)}\ln\frac{\alpha^{(T)}}{\alpha_0} + \frac{m}{4(\mu+\eta_T)}\sum_{t=1}^T\frac{(\sigma_m^{(t)})^2}{\alpha^{(t)}} + C^2\sum_{t=1}^T(\sigma_m^{(t-1)})^2.$$

We present the regret bound of RFD-SON for positive α_0 in Theorem 5. The term Ω_{RFD} in (11) is the main gap between RFD-SON and the standard online Newton step without sketching. Ω_{RFD} is dominated by the last term which can be bounded as (1). If we exploit the standard FD to sketched online Newton step (Luo et al., 2016) (FD-SON), the regret bound is similar to (11) but the gap will be

$$\Omega_{\text{FD}} = \frac{m\Omega_k}{2(m-k)(\mu+\eta_T)\alpha_0},$$

Algorithm 3 RFD for Online Newton Step

1: **Input:** $\alpha^{(0)} = \alpha_0 \geq 0$, $m < d$, $\eta_t = \mathcal{O}(1/t)$, $\mathbf{w}^{(0)} = \mathbf{0}^d$ and $\mathbf{B}^{(0)}$ be empty.
2: **for** $t = 0, \ldots, T-1$ **do**
3: Receive example $\mathbf{x}^{(t)}$, and loss function $f_t(\mathbf{w})$
4: Predict the output of $\mathbf{x}^{(t)}$ by $\mathbf{w}^{(t)}$ and suffer loss $f_t(\mathbf{w}^{(t)})$
5: $\mathbf{g}^{(t)} = \nabla f_t(\mathbf{w}^{(t)})$
6: $\widehat{\mathbf{B}}^{(t-1)} = \begin{bmatrix}\mathbf{B}^{(t-1)} \\ (\sqrt{\mu_t+\eta_t}\mathbf{g}^{(t)})^\top\end{bmatrix}$
7: Compute SVD: $\widehat{\mathbf{B}}^{(t-1)} = \mathbf{U}^{(t-1)}\mathbf{\Sigma}^{(t-1)}(\mathbf{V}^{(t-1)})^\top$
8: $\mathbf{B}^{(t)} = \sqrt{\big(\mathbf{\Sigma}_{m-1}^{(t-1)}\big)^2 - \big(\sigma_m^{(t-1)}\big)^2\mathbf{I}_{m-1}}\cdot(\mathbf{V}_{m-1}^{(t-1)})^\top$
9: $\alpha^{(t)} = \alpha^{(t-1)} + \big(\sigma_m^{(t-1)}\big)^2/2$
10: $\mathbf{H}^{(t)} = (\mathbf{B}^{(t)})^\top\mathbf{B}^{(t)} + \alpha^{(t)}\mathbf{I}_d$
11: $\mathbf{u}^{(t+1)} = \mathbf{w}^{(t)} - (\mathbf{H}^{(t)})^\dagger\mathbf{g}^{(t)}$
12: $\mathbf{w}^{(t+1)} = \text{argmin}_{\mathbf{w}\in\mathcal{K}_{t+1}}\|\mathbf{w} - \mathbf{u}^{(t+1)}\|_{\mathbf{H}^{(t)}}$
13: **end for**

where Ω_k plays the similar role to term $\sum_{t=1}^T(\sigma_m^{(t-1)})^2$ in RFD-SON and the detailed definition can be found in Luo et al. (2016). This result is heavily dependent on the hyperparameter α_0. If we increase the value of α_0, the gap Ω_{FD} can be reduced but the term $\frac{\alpha_0}{2}\|\mathbf{w}\|^2$ in the bound will increase, and vice versa. In other words, we need to trade off $\frac{\alpha_0}{2}\|\mathbf{w}\|^2$ and Ω_{FD} by tuning α_0 carefully. For RFD-SON, Theorem 5 implies that we can set α_0 be sufficiently small to reduce $\frac{\alpha_0}{2}\|\mathbf{w}\|^2$ and it has limited effect on the term Ω_{RFD}. The reason is that the first term of Ω_{RFD} contains $\frac{1}{\alpha_0}$ in the logarithmic function and the second term contains $\alpha^{(t)} = \alpha_0 + \frac{1}{2}\sum_{i=1}^{t-1}(\sigma_m^{(i)})^2$ in the denominator. For large t, $\alpha^{(t)}$ is mainly dependent on $\sum_{i=1}^{t-1}(\sigma_m^{(i)})^2$, rather than α_0. Hence the regret bound of RFD-SON is much less sensitive to the hyperparameter α_0 than FD-SON. We have $\sum_{t=1}^T(\sigma_m^{(t-1)})^2 \leq \|\mathbf{A} - [\mathbf{A}]_k\|_F^2/(m-k)$ for $k < m$ by using (17) with $\mathbf{A} = \sum_{t=1}^T(\mu_t+\eta_t)\mathbf{g}^{(t)}(\mathbf{g}^{(t)})^\top$.

Consider RFD-SON with $\alpha_0 = 0$. Typically, the parameter $\alpha^{(t)}$ is zero at very few first iterations and increase to be strictly positive later. Hence the learning algorithm can be divided into two phases based on whether $\alpha^{(t)}$ is zero. Suppose that T' satisfies

$$\alpha^{(t)}\begin{cases}= 0, & t < T' \\ > 0, & t \geq T'.\end{cases} \tag{12}$$

Then the regret can be written as

$$R_T(\mathbf{w}) = R_{1:T'}(\mathbf{w}) + R_{(T'+1):T}(\mathbf{w}),$$

where

$$R_{1:T'}(\mathbf{w}) = \sum_{t=1}^{T'} f_t(\mathbf{w}^{(t)}) - \sum_{t=1}^{T'} f_t(\mathbf{w})$$

and

$$R_{(T'+1):T}(\mathbf{w}) = \sum_{t=T'+1}^{T} f_t(\mathbf{w}^{(t)}) - \sum_{t=T'+1}^{T'} f_t(\mathbf{w}).$$

The regret from the first T' iterations can be bounded by Theorem 6 and the bound of $R_{(T'+1):T}(\mathbf{w})$ can be derived by the similar proof of Theorem 5.

Theorem 6 *By the condition of Theorem 5, $\mu' = \max_{t=1}^{T'}\{\mu_t\}$ and letting $\alpha_0 = 0$, σ^* be the smallest nonzero singular values of $\sum_{t=1}^{T'} \mathbf{g}^{(t)}(\mathbf{g}^{(t)})^\top$ and T' satisfy (12), we have that the first T' iterations of Algorithm 3 has the following regret*

$$R_{1:T'}(\mathbf{w}) \leq 2(CL)^2 \sum_{t=1}^{T'} \eta_t + \frac{m-1}{2(\eta_1+\mu')} + \frac{m(m-1)}{2(\eta_1+\mu')} \ln\Big(1 + \frac{2\sum_{t=1}^{T'} \|\mathbf{g}^{(t)}\|_2^2}{(1+r)r\sigma^*}\Big). \tag{13}$$

Combining above results, we can conclude the regret bound for the hyperparameter-free algorithm in Theorem 7. In practice, we often set m to be much smaller than d and T, which implies T' is much smaller than T. Hence, the regret bound of Theorem 7 is similar to the one of Theorem 5 when α_0 is close to 0. We can use RFD-SON with $\alpha_0 = 0$ and $\eta_t = 1/t$ to obtain a hyperparameter-free sketched online Newton algorithm. Luo et al. (2016) have proposed a hyperparameter-free online Newton algorithm without sketching and their regret contains a term with coefficient d.

Theorem 7 *Consider Algorithm 3 with $\alpha_0 = 0$, let T' satisfy (12), $\mu = \min_{t=1}^{T}\{\mu_t\}$, $\mu' = \max_{t=1}^{T'}\{\mu_t\}$, $\mathcal{K} = \bigcap_{t=1}^{T}\mathcal{K}_t$, σ^* has the same definition of Theorem 6 and $\alpha_0' = \det(\mathbf{H}^{(T')})$. Then under Assumptions 1 and 2 for any $\mathbf{w} \in \mathcal{K}$, we have that*

$$\begin{aligned} R_T(\mathbf{w}) \leq\ & 2(CL)^2 \sum_{t=1}^{T} \eta_t + \frac{m-1}{2(\eta_1+\mu')} + \frac{m(m-1)}{2(\eta_1+\mu')} \ln\Big(1 + \frac{2\sum_{t=1}^{T'} \|\mathbf{g}^{(t)}\|_2^2}{(1+r)r\sigma^*}\Big) \\ & + \frac{1}{2}\|\mathbf{w}^{(T')}\|^2_{\mathbf{H}^{(T')}} + \frac{m}{2(\mu+\eta_T)} \ln\Big(\mathrm{tr}\frac{\big((\mathbf{B}^{(T)})^\top \mathbf{B}^{(T)}\big)}{m\alpha^{(T')}} + \frac{\alpha^{(T)}}{\alpha_0'}\Big) + \Omega'_{\mathrm{RFD}} \end{aligned} \tag{14}$$

where

$$\Omega'_{\mathrm{RFD}} = \frac{d-m}{2(\mu+\eta_T)} \ln\frac{\alpha^{(T)}}{\alpha_0'} + \frac{m}{4(\mu+\eta_T)} \sum_{t=T'+1}^{T} \frac{(\sigma_m^{(t)})^2}{\alpha^{(t)}} + C^2 \sum_{t=T'+1}^{T} (\sigma_m^{(t-1)})^2.$$

6. Experiments

In this section, we evaluate the performance of robust frequent directions (RFD) and online Newton step by RFD (RFD-SON) on six real-world data sets "a9a," "gisette," "sido0," "farm-ads," "rcv1" and "real-sim," whose details are listed in Table 1. The data sets "sido0" and "farm-ads"can be found

on Causality Workbench[3], and UCI Machine Learning Repository[4]. The others can be downloaded from LIBSVM repository[5]. The experiments are conducted in Matlab and run on a server with Intel (R) Core (TM) i7-3770 CPU 3.40GHz×2, 8GB RAM and 64-bit Windows Server 2012 system.

data sets	n	d	source
a9a	32,561	123	(Platt, 1999)
gisette	6,000	5,000	(Guyon et al., 2004)
sido0	12,678	4,932	(Guyon et al., 2008)
farm-ads	4,143	54,877	(Mesterharm and Pazzani, 2011)
rcv1	20,242	47,236	(Lewis et al., 2004)
real-sim	72,309	20,958	(McCallum)

Table 1: Summary of data sets used in our experiments

6.1. Matrix Approximation

We evaluate the approximation errors of the deterministic sketching algorithms including frequent directions (FD) (Liberty, 2013; Ghashami et al., 2016), parameterized frequent directions (PFD), compensative frequent directions (CFD) (Desai et al., 2016) and robust frequent directions (RFD). For a given data set $\mathbf{A} \in \mathbb{R}^{n\times d}$ of n samples with d features, we use the accelerated algorithms (see details in Appendix A) to approximate the covariance matrix $\mathbf{A}^\top\mathbf{A}$ by $\mathbf{B}^\top\mathbf{B}$ for FD, PFD, CFD and by $\mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d$ for RFD, respectively. We measure the performance according to the relative spectral norm error. We report the relative spectral norm error by varying the sketch size m.

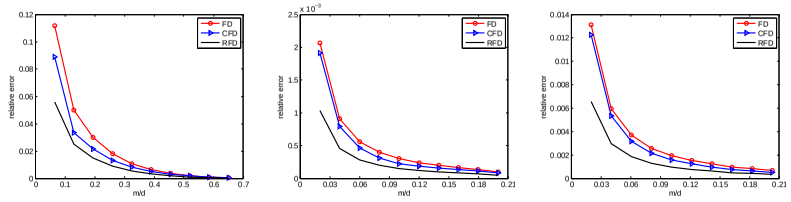
Figure 1 shows the performance of FD, CFD and RFD. These three methods have no extra hyperparameter and their outputs only rely on the sketch size. The relative error of RFD is always smaller than that of FD and CFD. The error of RFD is nearly half of the error of FD in most cases, which matches our theoretical analysis in Theorem 3 very well.

Figure 2 compares the performance of RFD and PFD with different choices of the hyperparameter. We use PFD-β to refer the PFD algorithm where $\lfloor \beta m \rfloor$ singular values will get affected by the shrinkage steps. The extra hyperparameter β is tuned from $\{0.2, 0.4, 0.6, 0.8\}$. The result shows that RFD is better than PFD in most cases. PFD sometimes can achieve lower approximation error with a good choice of β. However, selecting the hyperparameter requires additional computation.

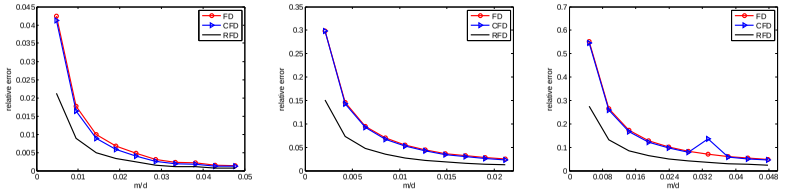
6.2. Online Learning

We now evaluate the performance of RFD-SON. We use the least squares loss $f_t(\mathbf{w}) = (\mathbf{w}^\top\mathbf{x}^{(t)} - y^{(t)})^2$, and set $\mathcal{K}_t = \{\mathbf{w} : |\mathbf{w}^\top\mathbf{x}^{(t)}| \leq 1\}$. In the experiments, we use the doubling space strategy (Algorithm 5 in Appendix A). We use 70% of the data set for training and the rest for test. The algorithms in the experiments include ADAGRAD, the standard online Newton step with the full Hessian (Duchi et al., 2011) (FULL-ON), the sketched online Newton step with frequent directions (FD-SON), the parameterized frequent directions (PFD-SON), the random projections (RP-SON), Oja's algorithms (Oja-SON) (Luo et al., 2016; Desai et al., 2016), and our proposed sketched online Newton step with RFD (RFD-SON).

3. https://www.causality.inf.ethz.ch/data/SIDO.html
4. https://archive.ics.uci.edu/ml/datasets/Farm+Ads
5. https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/

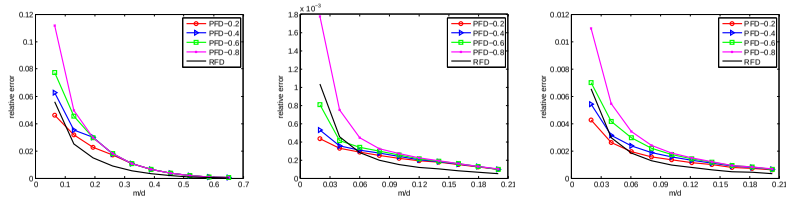


(a) a9a (b) gisette (c) sido0

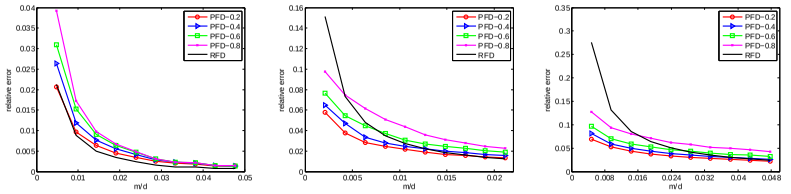


(d) farm-ads (e) rcv1 (f) real-sim

Figure 1: Comparison of relative spectral error of FD, CFD and RFD with proportion of sketching



(a) a9a (b) gisette (c) sido0



(d) farm-ads (e) rcv1 (f) real-sim

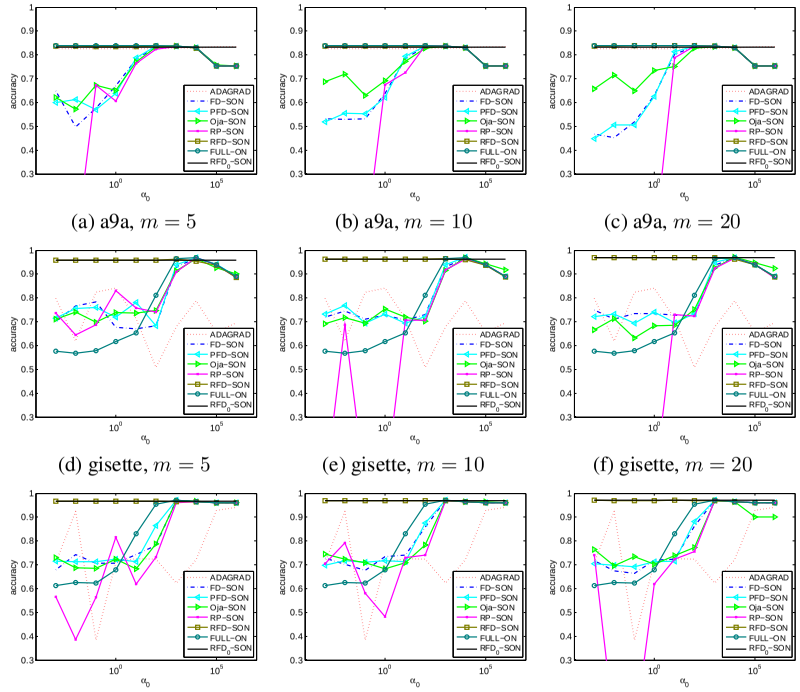
Figure 2: Comparison of relative spectral error of PFD and RFD with proportion of sketching

The hyperparameter α_0 is tuned from $\{10^{-3}, 10^{-2} \dots 10^5, 10^6\}$ for all methods and let $\eta = 1/t$ for SON algorithms. FULL-ON is too expensive and impractical for large d, so we exclude it from experiments on "farm-ads," "rcv1" and "real-sim." For PFD-SON, we let $\beta = 0.2$ heuristically because it usually achieves good performance on approximating the covariance matrix. Additionally, RFD-SON includes the result with $\alpha_0 = 0$ (RFD$_0$-SON). The sketch size of sketched online Newton methods is chosen from $\{5, 10, 20\}$ for "a9a," "gisette," "sido0," and $\{20, 30, 50\}$ for "farm-ads," "rcv1" and "real-sim." We measure performance according to two metrics (Duchi et al., 2011): the

online error rate and the test set performance of the predictor at the end of one pass through the training data.

We are interested in how the hyperparameter α_0 affects the performance of the algorithms. We display the test set performance in Figures 3 and 4. We compare the online error rate of RFD$_0$-SON with the one of FULL-ON in Figure 5 and show the comparison between RFD$_0$-SON and other SON methods with different choices of α_0 in Figures 6 - 11.

We also report the accuracy on the test sets for all algorithms at one pass with the best α_0 in Table 2 and the corresponding running times in Table 3. All SON algorithms can perform well with the best choice of α_0. However, only RFD$_0$-SON can perform well without tuning the hyperparameter while all baseline methods ADAGRAD, FD-SON, PFD-SON, RP-SON and Oja-SON are very sensitive to the value of α_0. The sub-figure (j)-(l) in Figures 5-11 shows RFD-SON usually has good performance with small α_0, which validates our theoretical analysis in Theorem 5. The choice of the hyperparameter almost has no effect of RFD-SON on data set "a9a'," "gisette," "sido0" and "farm-ads." These results verify that RFD-SON is a very stable algorithm in practice.

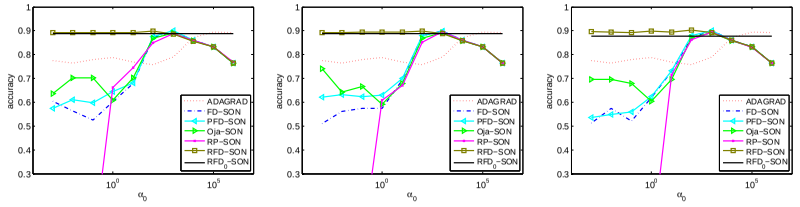


(a) a9a, $m = 5$ (b) a9a, $m = 10$ (c) a9a, $m = 20$

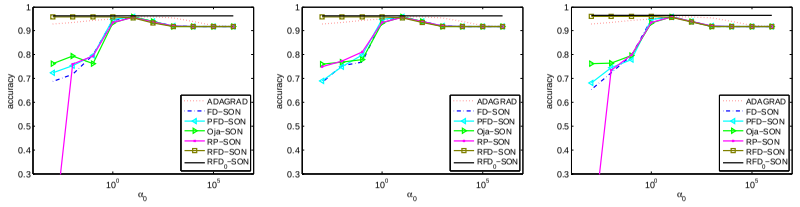
(d) gisette, $m = 5$ (e) gisette, $m = 10$ (f) gisette, $m = 20$

(g) sido0, $m = 5$ (h) sido0, $m = 10$ (i) sido0, $m = 20$

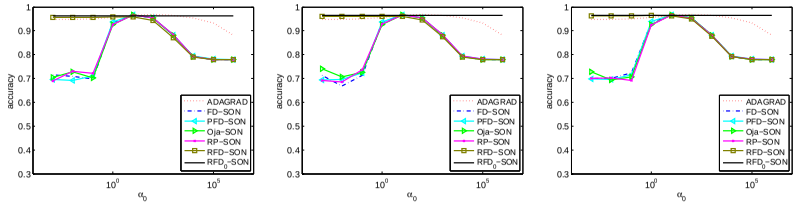
Figure 3: Comparison of the test error at the end of one pass on "a9a", "gisette", "sido0"



(a) farm-ads, $m = 20$ (b) farm-ads, $m = 30$ (c) farm-ads, $m = 50$

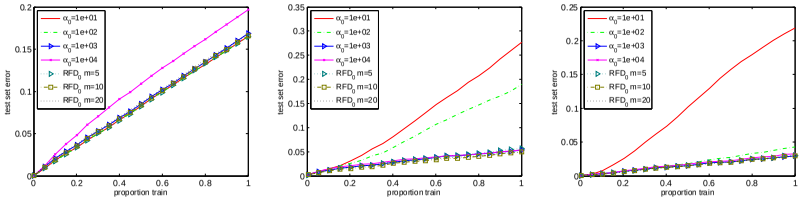


(d) rcv1, $m = 20$ (e) rcv1, $m = 30$ (f) rcv1, $m = 50$



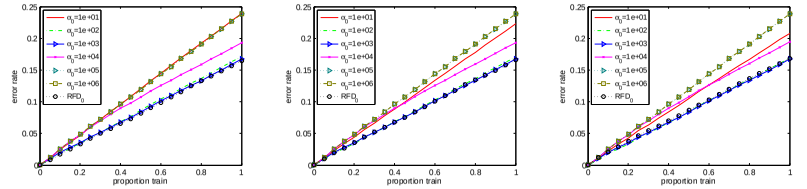
(g) real-sim, $m = 20$ (h) real-sim, $m = 30$ (i) real-sim, $m = 50$

Figure 4: Comparison of the test error at the end of one pass on "farm-ads", "rcv1", "real-sim"

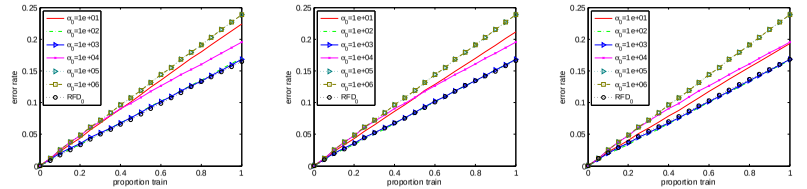


(a) a9a (a) gisette (c) sido0

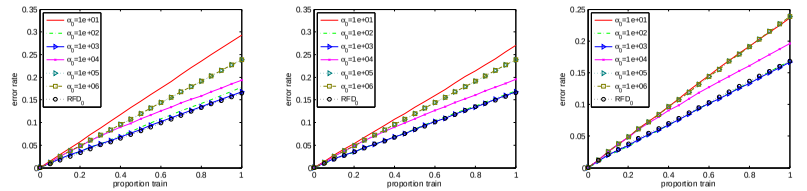
Figure 5: Comparison of the online error rate between algorithm FULL-ON and RFD$_0$



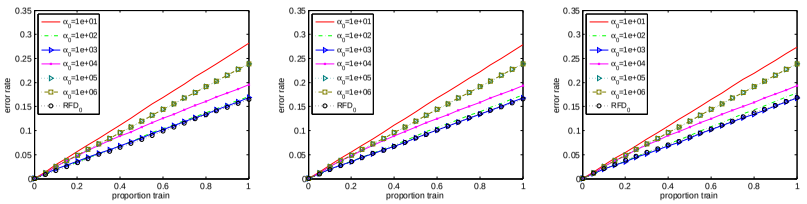
(a) FD vs RFD$_0$, $m = 5$ (b) FD vs RFD$_0$, $m = 10$ (c) FD vs RFD$_0$, $m = 20$



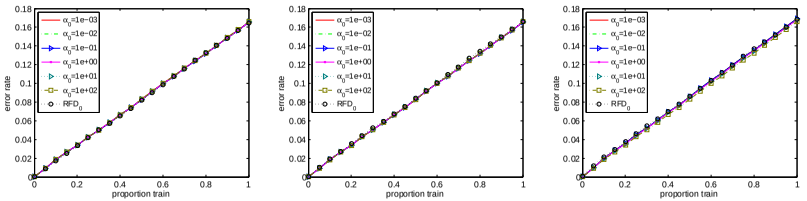
(a) PFD vs RFD$_0$, $m = 5$ (b) PFD vs RFD$_0$, $m = 10$ (c) PFD vs RFD$_0$, $m = 20$



(d) RP vs RFD$_0$, $m = 5$ (e) RP vs RFD$_0$, $m = 10$ (f) RP vs RFD$_0$, $m = 20$

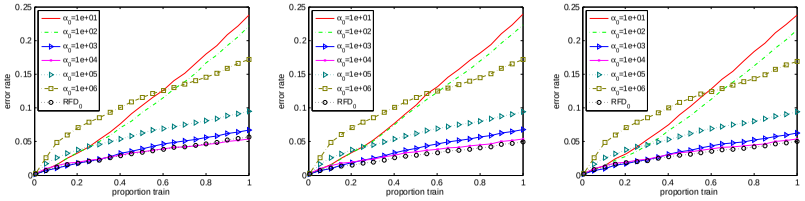


(g) Oja vs RFD$_0$, $m = 5$ (h) Oja vs RFD$_0$, $m = 10$ (i) Oja vs RFD$_0$, $m = 20$

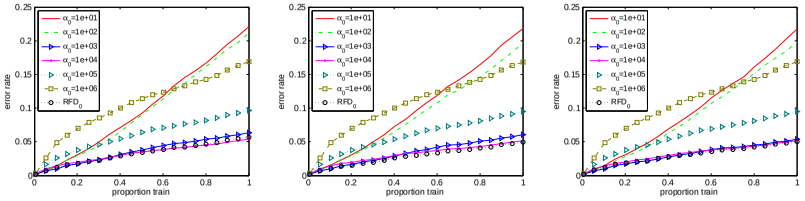


(j) RFD vs RFD$_0$, $m = 5$ (k) RFD vs RFD$_0$, $m = 10$ (l) RFD vs RFD$_0$, $m = 20$

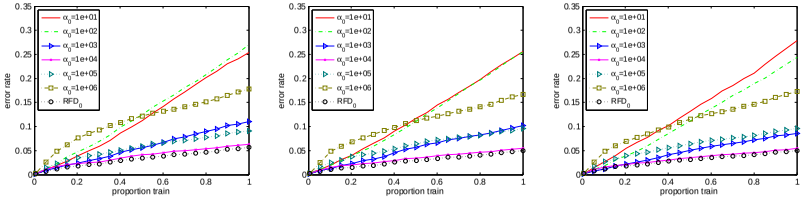
Figure 6: Comparison of the online error rate on "a9a"



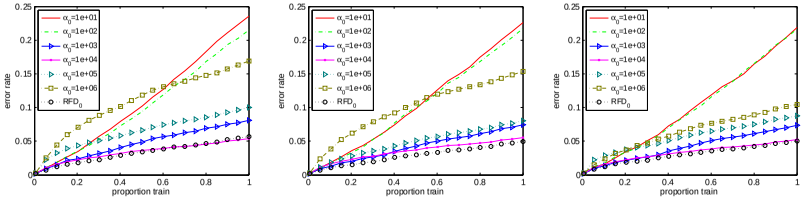
(a) FD vs RFD$_0$, $m = 5$ (b) FD vs RFD$_0$, $m = 10$ (c) FD vs RFD$_0$, $m = 20$



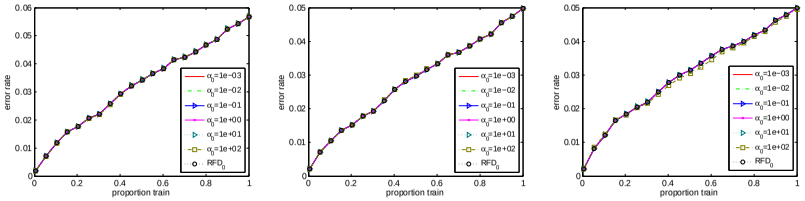
(a) PFD vs RFD$_0$, $m = 5$ (b) PFD vs RFD$_0$, $m = 10$ (c) PFD vs RFD$_0$, $m = 20$



(d) RP vs RFD$_0$, $m = 5$ (e) RP vs RFD$_0$, $m = 10$ (f) RP vs RFD$_0$, $m = 20$

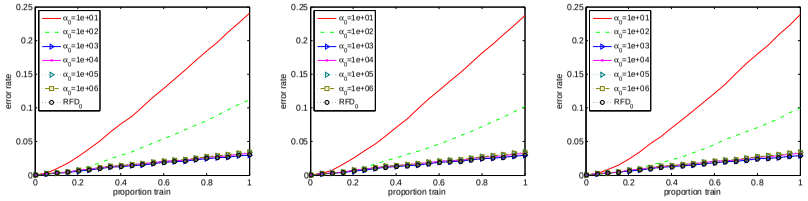


(g) Oja vs RFD$_0$, $m = 5$ (h) Oja vs RFD$_0$, $m = 10$ (i) Oja vs RFD$_0$, $m = 20$

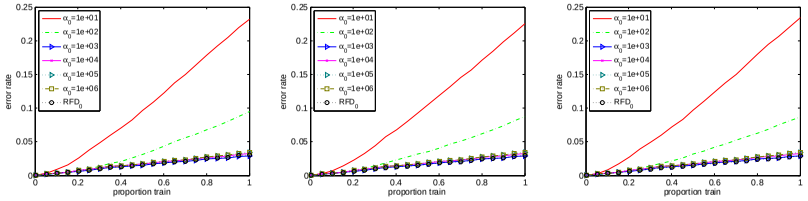


(j) RFD vs RFD$_0$, $m = 5$ (k) RFD vs RFD$_0$, $m = 10$ (l) RFD vs RFD$_0$, $m = 20$

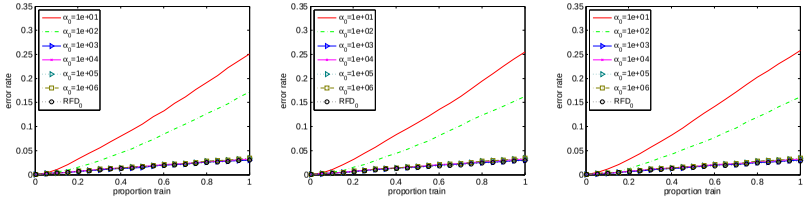
Figure 7: Comparison of the online error rate on "gisette"



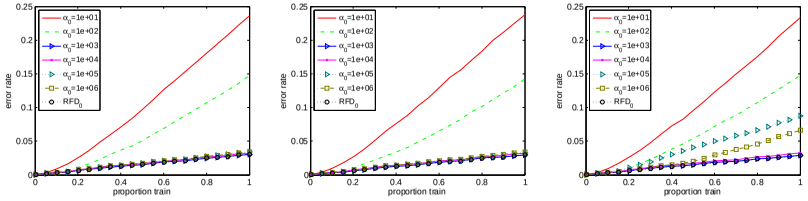
(a) FD vs RFD$_0$, $m = 5$ (b) FD vs RFD$_0$, $m = 10$ (c) FD vs RFD$_0$, $m = 20$



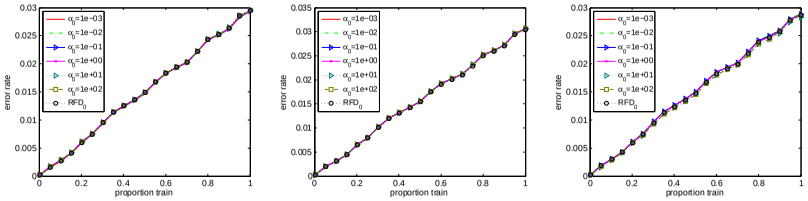
(a) PFD vs RFD$_0$, $m = 5$ (b) PFD vs RFD$_0$, $m = 10$ (c) PFD vs RFD$_0$, $m = 20$



(d) RP vs RFD$_0$, $m = 5$ (e) RP vs RFD$_0$, $m = 10$ (f) RP vs RFD$_0$, $m = 20$

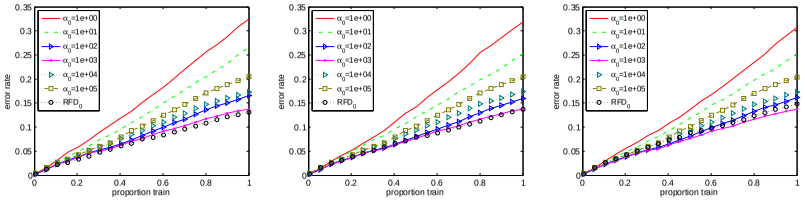


(g) Oja vs RFD$_0$, $m = 5$ (h) Oja vs RFD$_0$, $m = 10$ (i) Oja vs RFD$_0$, $m = 20$

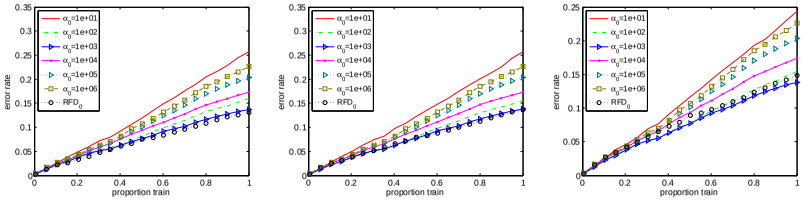


(j) RFD vs RFD$_0$, $m = 5$ (k) RFD vs RFD$_0$, $m = 10$ (l) RFD vs RFD$_0$, $m = 20$

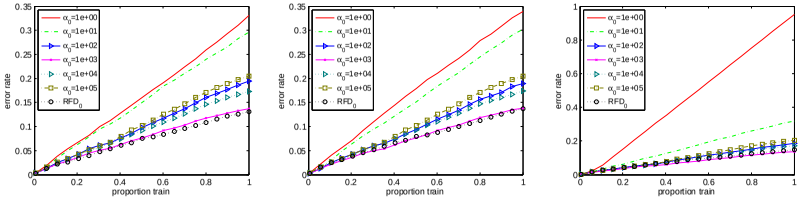
Figure 8: Comparison of the online error rate on "sido0"



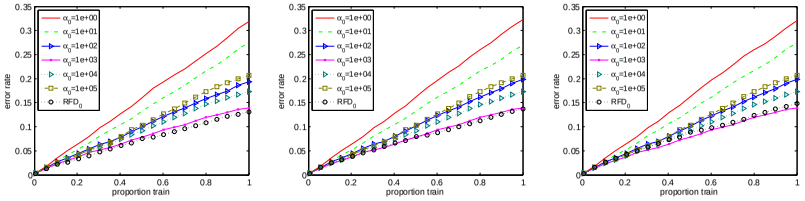
(a) FD vs RFD$_0$, $m = 20$ (b) FD vs RFD$_0$, $m = 30$ (c) FD vs RFD$_0$, $m = 50$



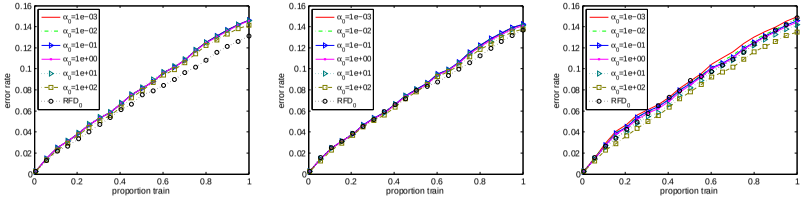
(a) PFD vs RFD$_0$, $m = 20$ (b) PFD vs RFD$_0$, $m = 30$ (c) PFD vs RFD$_0$, $m = 50$



(d) RP vs RFD$_0$, $m = 20$ (e) RP vs RFD$_0$, $m = 30$ (f) RP vs RFD$_0$, $m = 50$

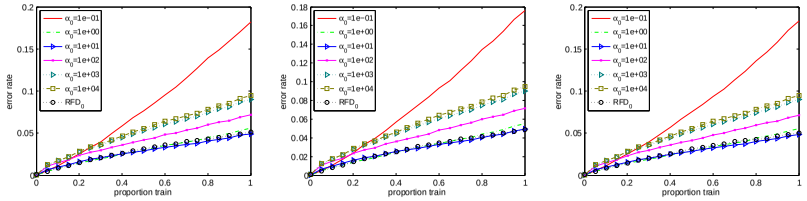


(g) Oja vs RFD$_0$, $m = 20$ (h) Oja vs RFD$_0$, $m = 30$ (i) Oja vs RFD$_0$, $m = 50$

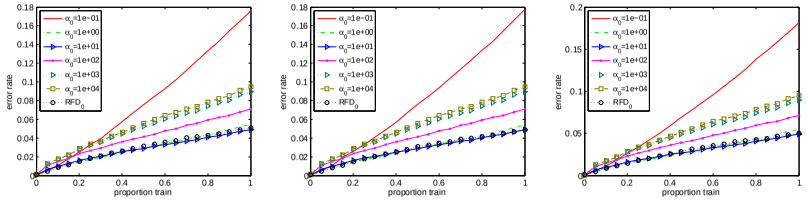


(j) RFD vs RFD$_0$, $m = 20$ (k) RFD vs RFD$_0$, $m = 30$ (l) RFD vs RFD$_0$, $m = 50$

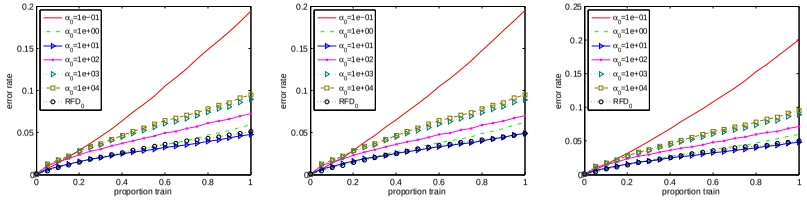
Figure 9: Comparison of the online error rate on "farm-ads"



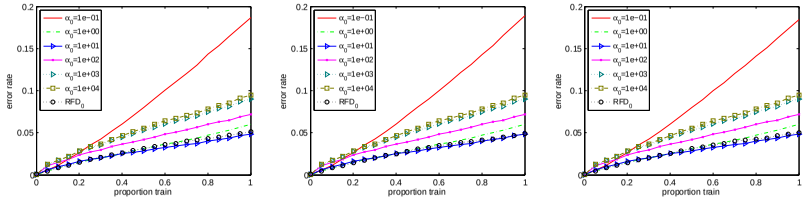
(a) FD vs RFD$_0$, $m = 20$ (b) FD vs RFD$_0$, $m = 30$ (c) FD vs RFD$_0$, $m = 50$



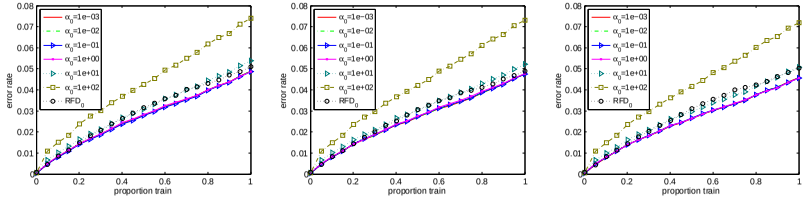
(a) PFD vs RFD$_0$, $m = 20$ (b) FD vs PFD$_0$, $m = 30$ (c) PFD vs RFD$_0$, $m = 50$



(d) RP vs RFD$_0$, $m = 20$ (e) RP vs RFD$_0$, $m = 30$ (f) RP vs RFD$_0$, $m = 50$

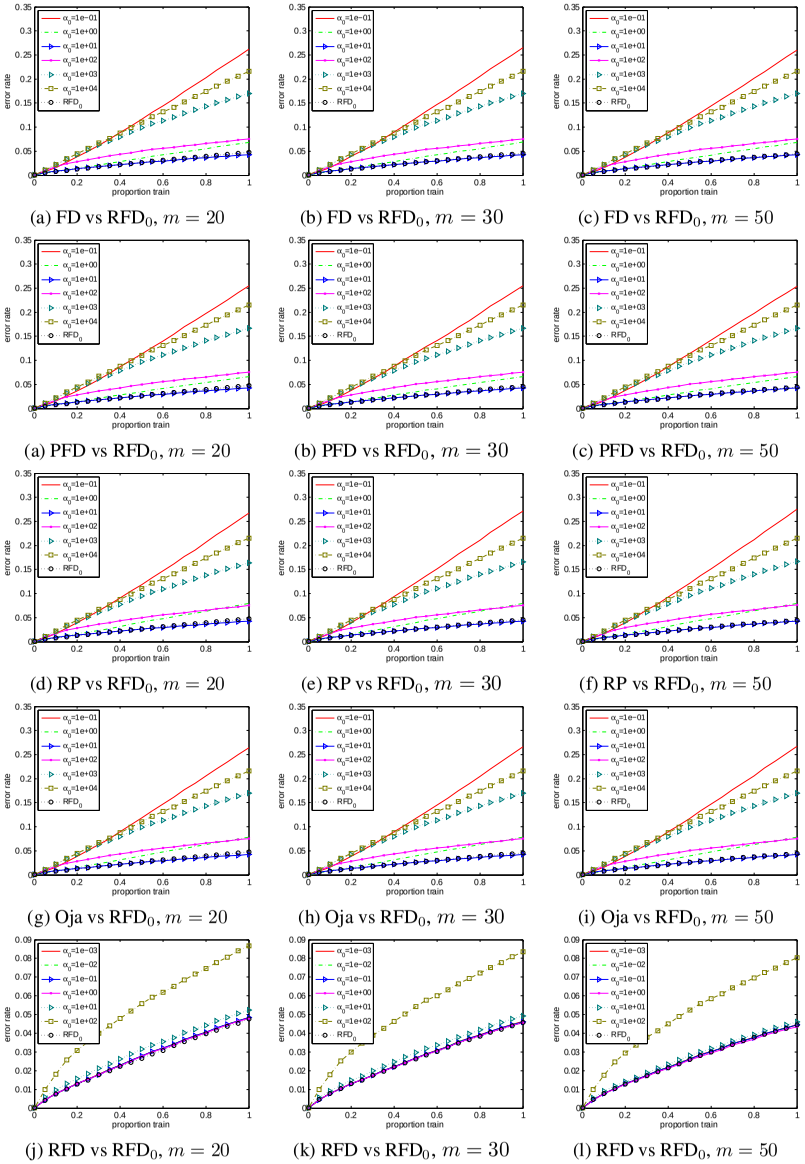


(g) Oja vs RFD$_0$, $m = 20$ (h) Oja vs RFD$_0$, $m = 30$ (i) Oja vs RFD$_0$, $m = 50$



(j) RFD vs RFD$_0$, $m = 20$ (k) RFD vs RFD$_0$, $m = 30$ (l) RFD vs RFD$_0$, $m = 50$

Figure 10: Comparison of the online error rate on "rcv1"



(a) FD vs RFD_0, $m = 20$ (b) FD vs RFD_0, $m = 30$ (c) FD vs RFD_0, $m = 50$

(a) PFD vs RFD_0, $m = 20$ (b) PFD vs RFD_0, $m = 30$ (c) PFD vs RFD_0, $m = 50$

(d) RP vs RFD_0, $m = 20$ (e) RP vs RFD_0, $m = 30$ (f) RP vs RFD_0, $m = 50$

(g) Oja vs RFD_0, $m = 20$ (h) Oja vs RFD_0, $m = 30$ (i) Oja vs RFD_0, $m = 50$

(j) RFD vs RFD_0, $m = 20$ (k) RFD vs RFD_0, $m = 30$ (l) RFD vs RFD_0, $m = 50$

Figure 11: Comparison of the online error rate on "real-sim"

Algorithms	a9a	gisette	sido0	farm-ads	rcv1	real-sim
ADAGRAD	83.4783	84.1111	94.0326	89.3805	95.8340	96.7315
FULL-ON	83.8264	96.9444	97.0557	/	/	/
FD, $m = m_1$	83.6524	96.7222	97.0557	89.7023	95.6200	96.7315
FD, $m = m_2$	83.6728	96.7222	97.0557	89.9437	95.6529	96.7361
FD, $m = m_3$	83.6728	96.7222	97.0557	89.9437	95.6529	96.7361
PFD, $m = m_1$	83.6728	97.0000	97.0294	90.0241	95.8340	96.7407
PFD, $m = m_2$	83.7138	97.0556	97.0820	89.8632	95.8340	96.7407
PFD, $m = m_3$	83.6626	97.0000	97.0557	90.0241	95.8340	96.7176
RP, $m = m_1$	83.4374	96.9444	96.5037	88.8174	95.7188	96.7499
RP, $m = m_2$	**83.9492**	96.2778	96.8980	89.7023	95.6694	96.7499
RP, $m = m_3$	83.7650	96.7222	97.0557	89.4610	95.7846	96.7315
Oja, $m = m_1$	83.6831	96.3889	96.6351	89.0587	95.7846	**96.7776**
Oja, $m = m_2$	83.6319	96.9444	96.8980	89.1392	95.7846	**96.7776**
Oja, $m = m_3$	83.5091	**97.1111**	96.8980	89.1392	95.7846	**96.7776**
RFD, $m = m_1$	83.6319	96.2222	96.6877	89.7828	95.8340	95.7666
RFD, $m = m_2$	83.6831	96.4444	96.9243	89.8632	95.8834	96.1267
RFD, $m = m_3$	83.9390	96.9444	97.0820	**90.3459**	96.1139	96.4037
RFD_0, $m = m_1$	83.2429	95.9444	96.6877	87.6911	96.3280	96.1636
RFD_0, $m = m_2$	83.2634	96.2778	96.9243	88.8174	96.3115	96.3806
RFD_0, $m = m_3$	83.2736	96.8889	**97.1083**	88.8978	**96.4762**	96.5560

Table 2: We list the accuracy (%) on test set at the end of one pass with best choice of α_0. The sketch size is set $(m_1, m_2, m_3) = (5, 10, 20)$ for "a9a", "gisette", "sido0" and $(m_1, m_2, m_3) = (20, 30, 50)$ for "farm-ads", "rcv1" and "real-sim".

7. Conclusions

In this paper we have proposed a novel sketching method robust frequent directions (RFD), and our theoretical analysis shows that RFD is superior to FD. We have also studied the use of RFD in the second order online learning algorithms. The online learning algorithm with RFD achieves better performance than baselines. It is worth pointing out that the application of RFD is not limited to convex online optimization. In future work, we would like to explore the use of RFD in stochastic optimization and non-convex problems.

Acknowledgments

We thank the anonymous reviewers for their helpful suggestions. Luo Luo, Cheng Chen and Zhihua Zhang have been supported by the National Natural Science Foundation of China (No. 61572017 and 11771002) and by Beijing Municipal Commission of Science and Technology under Grant No. 181100008918005. Wu-Jun Li has been supported by the NSFC-NRF Joint Research Project (No. 61861146001) and by the NSFC (No. 61472182).

Algorithms	a9a	gisette	sido0	farm-ads	rcv1	real-sim
ADAGRAD	9.8279e-05	1.9677e-04	1.9504e-04	3.1976e-04	7.0888e-04	3.2917e-04
FULL-ON	2.3141e-04	2.6296e-01	1.6299e-01	/	/	/
FD, $m = m_1$	2.8552e-04	5.2073e-04	6.0705e-04	1.6387e-03	2.0727e-02	6.6130e-03
FD, $m = m_2$	2.7276e-04	7.2830e-04	7.8723e-04	3.2000e-03	4.1148e-02	1.3530e-02
FD, $m = m_3$	3.4899e-04	3.4821e-03	1.4404e-03	5.1365e-03	7.2211e-02	3.0713e-02
PFD, $m = m_1$	2.6090e-04	5.8330e-04	5.7862e-04	2.4524e-02	2.0121e-02	6.6757e-03
PFD, $m = m_2$	2.8206e-04	2.0234e-03	7.7824e-04	3.2372e-02	4.1385e-02	1.2965e-02
PFD, $m = m_3$	3.2096e-04	3.3901e-03	1.4680e-03	5.4829e-02	7.1159e-02	3.0677e-02
RP, $m = m_1$	1.3597e-04	3.2097e-04	3.6333e-04	7.3933e-04	1.8736e-03	1.5551e-03
RP, $m = m_2$	2.7308e-04	7.2830e-04	7.8723e-04	3.2000e-03	2.8015e-03	1.8377e-03
RP, $m = m_3$	3.3307e-04	3.4821e-03	1.4404e-03	5.1365e-03	4.1585e-03	2.1537e-03
Oja, $m = m_1$	1.5500e-04	6.0334e-04	2.9098e-04	1.3061e-03	6.7158e-03	5.2530e-03
Oja, $m = m_2$	1.6719e-04	7.3481e-04	2.4472e-04	2.8887e-03	4.1148e-02	1.3530e-02
Oja, $m = m_3$	1.6631e-04	3.3918e-03	1.3920e-03	4.4386e-03	7.2211e-02	3.0713e-02
RFD, $m = m_1$	2.8549e-04	5.1545e-04	6.0296e-04	2.0484e-03	2.0557e-02	9.7858e-03
RFD, $m = m_2$	3.1813e-04	7.5013e-04	7.5699e-04	3.8129e-03	4.0695e-02	1.6527e-02
RFD, $m = m_3$	3.3405e-04	3.3495e-03	1.4472e-03	5.2458e-03	7.1764e-02	3.0175e-02
RFD_0, $m = m_1$	2.7466e-04	6.8607e-04	5.9339e-04	2.0843e-03	2.0033e-02	9.8373e-03
RFD_0, $m = m_2$	1.9542e-04	8.0961e-04	7.6749e-04	3.7857e-03	4.0779e-02	1.6892e-02
RFD_0, $m = m_3$	2.3561e-04	1.4328e-03	7.6749e-04	5.5628e-03	7.3725e-02	3.0490e-02

Table 3: We list the average iteration cost corresponding to Table 2.

Appendix A. Accelerating by Doubling Space

The cost of FD (Algorithm 1) is dominated by the steps of SVD. It takes $\mathcal{O}(Tm^2d)$ time by standard SVD in total. We can accelerate FD by doubling the sketch size (Liberty, 2013). The details are shown in Algorithm 4. Then the SVD is called only every m rows of $\mathbf{A}$ come and the time complexity is reduced to $\mathcal{O}(Tmd)$. Similarly, RFD can also be speeded up in this way. We demonstrate it in Algorithm 5.

We can apply similar strategy on RFD-SON, just as Algorithm 6 shows. For $\alpha^{(t)} > 0$, the parameter $\mathbf{w}^{(t)}$ can be updated in $\mathcal{O}(md)$ cost by Woodbury identity (Luo et al., 2016). Suppose $\mathbf{B}^{(t)} \in \mathbb{R}^{m' \times d}$, where $m' \leq 2m$. We have

$$\mathbf{u}^{(t+1)} = \mathbf{w}^{(t)} - \frac{1}{\alpha^{(t)}}\big(\mathbf{g}^{(t)} - (\mathbf{B}^{(t)})^\top \mathbf{M}^{(t)} \mathbf{B}^{(t)} \mathbf{g}^{(t)}\big),$$
$$\mathbf{w}^{(t+1)} = \mathbf{u}^{(t+1)} - \gamma^{(t)}\big(\mathbf{x}^{(t)} - (\mathbf{B}^{(t)})^\top \mathbf{M}^{(t)} \mathbf{B}^{(t)} \mathbf{x}^{(t)}\big),$$

where

$$\gamma^{(t)} = \frac{\tau\big((\mathbf{u}^{(t)})^\top \mathbf{x}^{(t)}\big)}{(\mathbf{x}^{(t)})^\top \mathbf{x}^{(t)} - (\mathbf{x}^{(t)})^\top (\mathbf{B}^{(t)})^\top \mathbf{M}^{(t)} \mathbf{B}^{(t)} \mathbf{x}^{(t)}},$$
$$\mathbf{M}^{(t)} = \big(\mathbf{B}^{(t)}(\mathbf{B}^{(t)})^\top + \alpha^{(t)} \mathbf{I}_m\big)^{-1},$$
$$\tau(z) = \operatorname{sgn}(z) \max\{|z| - 1, 0\}.$$

Let's check the cost of the above steps in detail. The matrix $\mathbf{M}^{(t)}$ costs $\mathcal{O}(m^2)$ space and the computation of $\mathbf{M}^{(t)}(\mathbf{B}^{(t)}\mathbf{g}^{(t)})$ or $\mathbf{M}^{(t)}(\mathbf{B}^{(t)}\mathbf{x}^{(t)})$ takes $\mathcal{O}(md + m^3)$ time given $\mathbf{B}^{(t)}(\mathbf{B}^{(t)})^\top + \alpha^{(t)}\mathbf{I}_{m'}$. The result of $\mathbf{B}^{(t)}(\mathbf{B}^{(t)})^\top$ also can be obtained in $\mathcal{O}(md)$ time. If we have executed SVD at current iteration (when $\widehat{\mathbf{B}}^{(t-1)}$ has $2m$ rows), $\mathbf{B}^{(t)}(\mathbf{B}^{(t)})$ is diagonal and we can directly obtain

Algorithm 4 Fast Frequent Directions

Input: $\mathbf{A} = [\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(T)}]^\top \in \mathbb{R}^{T \times d}$, $\mathbf{B}^{(0)} = [\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(m-1)}]^\top$
for $t = m, \ldots, T$ **do**
  $\widehat{\mathbf{B}}^{(t-1)} = \begin{bmatrix} \mathbf{B}^{(t-1)} \\ (\mathbf{a}^{(t)})^\top \end{bmatrix}$
  if $\widehat{\mathbf{B}}^{(t-1)}$ has $2m$ rows
    Compute SVD: $\widehat{\mathbf{B}}^{(t-1)} = \mathbf{U}^{(t-1)}\mathbf{\Sigma}^{(t-1)}(\mathbf{V}^{(t-1)})^\top$
    $\mathbf{B}^{(t)} = \sqrt{\big(\mathbf{\Sigma}_{m-1}^{(t-1)}\big)^2 - \big(\sigma_m^{(t-1)}\big)^2 \mathbf{I}_{m-1}} \cdot \big(\mathbf{V}_{m-1}^{(t-1)}\big)^\top$
  else
    $\mathbf{B}^{(t)} = \widehat{\mathbf{B}}^{(t-1)}$
  end if
end for
Output: $\mathbf{B} = \mathbf{B}^{(T)}$

Algorithm 5 Fast Robust Frequent Directions

Input: $\mathbf{A} = [\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(T)}]^\top \in \mathbb{R}^{T \times d}$, $\mathbf{B}^{(m-1)} = [\mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(m-1)}]^\top$, $\alpha^{(m-1)} = 0$
for $t = m, \ldots, T$ **do**
  $\widehat{\mathbf{B}}^{(t-1)} = \begin{bmatrix} \mathbf{B}^{(t-1)} \\ (\mathbf{a}^{(t)})^\top \end{bmatrix}$
  if $\widehat{\mathbf{B}}^{(t-1)}$ has $2m$ rows
    Compute SVD: $\widehat{\mathbf{B}}^{(t-1)} = \mathbf{U}^{(t-1)}\mathbf{\Sigma}^{(t-1)}(\mathbf{V}^{(t-1)})^\top$
    $\mathbf{B}^{(t)} = \sqrt{\big(\mathbf{\Sigma}_{m-1}^{(t-1)}\big)^2 - \big(\sigma_m^{(t-1)}\big)^2 \mathbf{I}_{m-1}} \cdot \big(\mathbf{V}_{m-1}^{(t-1)}\big)^\top$
    $\alpha^{(t)} = \alpha^{(t-1)} + \big(\sigma_m^{(t-1)}\big)^2/2$
  else
    $\mathbf{B}^{(t)} = \widehat{\mathbf{B}}^{(t-1)}$
    $\alpha^{(t)} = \alpha^{(t-1)}$
  end if
end for
Output: $\mathbf{B} = \mathbf{B}^{(T)}$ and $\alpha = \alpha^{(T)}$

the SVD of

$$\mathbf{B}^{(t)}(\mathbf{B}^{(t)})^\top = \left(\mathbf{\Sigma}_{m-1}^{(t-1)}\right)^2 - \left(\sigma_m^{(t-1)}\right)^2\mathbf{I}_{m-1},$$

otherwise it can be updated incrementally in $\mathcal{O}(md)$ as follows

$$\mathbf{B}^{(t)}(\mathbf{B}^{(t)})^\top = \begin{bmatrix} \mathbf{B}^{(t-1)}(\mathbf{B}^{(t-1)})^\top & \sqrt{\mu_t+\eta_t}\mathbf{B}^{(t-1)}\mathbf{g}^{(t)} \\ \sqrt{\mu_t+\eta_t}(\mathbf{B}^{(t-1)}\mathbf{g}^{(t)})^\top & (\mu_t+\eta_t)(\mathbf{g}^{(t)})^\top\mathbf{g}^{(t)} \end{bmatrix}.$$

Since we have $m \ll d$, all above operations only require $\mathcal{O}(md)$ time and space complexity in total for each iteration.

In the case of $\alpha^{(t)} = 0$, we can iterate $\mathbf{u}^{(t)}$ and $\mathbf{w}^{(t)}$ by using SVD on $\mathbf{B}^{(t)}$. Let the condensed SVD of $\mathbf{B}^{(t)}$ be $\mathbf{B}^{(t)} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where $\mathbf{U} \in \mathbb{R}^{m'\times r}$, $\mathbf{\Sigma} \in \mathbb{R}^{r\times r}$, $\mathbf{V} \in \mathbb{R}^{r\times d}$ and $r = \operatorname{rank}(\mathbf{B}^{(t)}) < m'$. Then we have

$$\mathbf{H}^{(t)} = (\mathbf{B}^{(t)})^\top\mathbf{B}^{(t)} = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^\top,$$
$$(\mathbf{H}^{(t)})^\dagger = \mathbf{V}\mathbf{\Sigma}^{-2}\mathbf{V}^\top.$$

We can update $\mathbf{u}^{(t+1)}$ and $\mathbf{w}^{(t+1)}$ as follows

$$\mathbf{u}^{(t+1)} = \mathbf{w}^{(t)} - \mathbf{V}^\top(\mathbf{\Sigma}^{-2}(\mathbf{V}^\top\mathbf{g}^{(t)})),$$
$$\mathbf{w}^{(t+1)} = \operatorname*{argmin}_{\mathbf{w}\in\{\mathbf{w}_1^{(t)},\mathbf{w}_2^{(t)}\}} \|\mathbf{w} - \mathbf{u}^{(t+1)}\|_{\mathbf{H}^{(t)}},$$

where

$$\mathbf{w}_1^{(t)} = \mathbf{u}^{(t+1)} - \frac{\tau\left((\mathbf{u}^{(t+1)})^\top\mathbf{x}^{(t)}\right)}{(\mathbf{x}^{(t)})^\top\mathbf{V}\mathbf{\Sigma}^{-2}\mathbf{V}^\top\mathbf{x}^{(t)}}\mathbf{V}\mathbf{\Sigma}^{-2}\mathbf{V}^\top\mathbf{x}^{(t)},$$
$$\mathbf{w}_2^{(t)} = \mathbf{u}^{(t+1)} - \frac{\tau\left((\mathbf{u}^{(t+1)})^\top\mathbf{x}^{(t)}\right)}{(\mathbf{x}^{(t)})^\top\mathbf{x}^{(t)} - (\mathbf{x}^{(t)})^\top\mathbf{V}\mathbf{\Sigma}^{-2}\mathbf{V}^\top\mathbf{x}^{(t)}}(\mathbf{x}^{(t)} - \mathbf{V}\mathbf{\Sigma}^{-2}\mathbf{V}^\top\mathbf{x}^{(t)}).$$

The iteration costs $\mathcal{O}(m^2d)$ in total (dominated by the SVD of $\mathbf{B}^\top$), but $\alpha^{(t)} = 0$ only appears at a few early iterations. Hence the average iteration complexity of RFD-SON is dominated by the case $\alpha^{(t)} > 0$ which takes $\mathcal{O}(md)$. Note that the algorithm is also valid without the smoothness of f_t. We can replace the gradient ∇f_t with the corresponding subgradient.

Appendix B. The Proof of Theorem 1

In this section, we firstly provide several lemmas from the book "Topics in matrix analysis" (Horn and Johnson, 1991), then we prove Theorem 1. The proof of Lemma 1 and 2 can be found in the book and we give the proof of Lemma 3 here.

Lemma 1 (Theorem 3.4.5 of (Horn and Johnson, 1991)) *Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$ be given, and suppose $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{A}-\mathbf{B}$ have decreasingly ordered singular values, $\sigma_1(\mathbf{A}) \geq \cdots \geq \sigma_q(\mathbf{A})$, $\sigma_1(\mathbf{B}) \geq \cdots \geq \sigma_q(\mathbf{B})$, and $\sigma_1(\mathbf{A}-\mathbf{B}) \geq \cdots \geq \sigma_q(\mathbf{A}-\mathbf{B})$, where $q = \min\{m,n\}$. Define $s_i(\mathbf{A},\mathbf{B}) \equiv |\sigma_i(\mathbf{A}) - \sigma_i(\mathbf{B})|$, $i = 1,\ldots,q$ and let $s_{[1]}(\mathbf{A},\mathbf{B}) \geq \cdots \geq s_{[i]}(\mathbf{A},\mathbf{B})$ denote a decreasingly ordered rearrangement of the values $s_i(\mathbf{A},\mathbf{B})$. Then*

$$\sum_{i=1}^k s_{[i]}(\mathbf{A},\mathbf{B}) \leq \sum_{i=1}^k \sigma_i(\mathbf{A}-\mathbf{B}) \text{ for } k = 1,\ldots,q.$$

Algorithm 6 Fast RFD for Online Newton Step

Input: $\alpha^{(0)} = \alpha_0$, $m < d$, $\eta_t = \mathcal{O}(1/t)$ and $\mathbf{B}^{(0)} = \mathbf{0}^{m\times d}$.
for $t = 1,\ldots,T$ **do**
  Receive example $\mathbf{x}^{(t)}$, and loss function $f_t(\mathbf{w})$
  Predict the output of $\mathbf{x}^{(t)}$ by $\mathbf{w}^{(t)}$ and suffer loss $f_t(\mathbf{w}^{(t)})$
  $\mathbf{g}^{(t)} = \nabla f_t(\mathbf{w}^{(t)})$
  $\widehat{\mathbf{B}}^{(t-1)} = \begin{bmatrix} \mathbf{B}^{(t-1)} \\ (\sqrt{\mu_t+\eta_t}\mathbf{g}^{(t)})^\top \end{bmatrix}$
  if $\widehat{\mathbf{B}}^{(t-1)}$ has $2m$ rows
    Compute SVD: $\widehat{\mathbf{B}}^{(t-1)} = \mathbf{U}^{(t-1)}\mathbf{\Sigma}^{(t-1)}(\mathbf{V}^{(t-1)})^\top$
    $\mathbf{B}^{(t)} = \sqrt{\left(\mathbf{\Sigma}_{m-1}^{(t-1)}\right)^2 - \left(\sigma_m^{(t-1)}\right)^2\mathbf{I}_{m-1}}\cdot(\mathbf{V}_{m-1}^{(t-1)})^\top$
    $\alpha^{(t)} = \alpha^{(t-1)} + \left(\sigma_m^{(t-1)}\right)^2/2$
  else
    $\mathbf{B}^{(t)} = \widehat{\mathbf{B}}^{(t-1)}$
    $\alpha^{(t)} = \alpha^{(t-1)}$
  end if
  $\mathbf{H}^{(t)} = (\mathbf{B}^{(t)})^\top\mathbf{B}^{(t)} + \alpha^{(t)}\mathbf{I}_d$
  $\mathbf{u}^{(t+1)} = \mathbf{w}^{(t)} - (\mathbf{H}^{(t)})^\dagger\mathbf{g}^{(t)}$
  $\mathbf{w}^{(t+1)} = \operatorname{argmin}_{\mathbf{w}\in\mathcal{K}_t}\|\mathbf{w} - \mathbf{u}^{(t+1)}\|_{\mathbf{H}^{(t)}}$
end for

Lemma 2 (Corollary 3.5.9 of (Horn and Johnson, 1991)) *Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$ be given, and let $q = \min\{m,n\}$. The following are equivalent*

1. *$\|\mathbf{A}\| \leq \|\mathbf{B}\|$ for every unitarily invariant norm $\|\cdot\|$ on $\mathbb{R}^{m\times n}$.*
2. *$N_k(\mathbf{A}) \leq N_k(\mathbf{B})$ for $k = 1,\ldots,q$ where $N_k(\mathbf{X}) \equiv \sum_{i=1}^k \sigma_k(\mathbf{X})$ denotes Ky Fan k-norm.*

Lemma 3 (Page 215 of (Horn and Johnson, 1991)) *Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$ be given, and let $q = \min\{m,n\}$. Define the diagonal matrix $\mathbf{\Sigma}(\mathbf{A}) = [\sigma_{ij}] \in \mathbb{R}^{m\times n}$ by $\sigma_{ii} = \sigma_i(\mathbf{A})$, all other $\sigma_{ij} = 0$, where $\sigma_1(\mathbf{A}) \geq, \ldots, \geq \sigma_q(\mathbf{A})$ are the decreasingly ordered singular values of $\mathbf{A}$. We define $\mathbf{\Sigma}(\mathbf{B})$ similarly. Then we have $\|\mathbf{A}-\mathbf{B}\| \geq \|\mathbf{\Sigma}(\mathbf{A}) - \mathbf{\Sigma}(\mathbf{B})\|$ for every unitarily invariant norm $\|\cdot\|$.*

Proof Using the notation of Lemma 1 and 2, matrices $\mathbf{A}-\mathbf{B}$ and $\mathbf{\Sigma}(\mathbf{A})-\mathbf{\Sigma}(\mathbf{B})$ have the decreasingly ordered singular values $\sigma_1(\mathbf{A}-\mathbf{B}) \geq \cdots \geq \sigma_q(\mathbf{A}-\mathbf{B})$ and $s_{[1]}(\mathbf{A},\mathbf{B}) \geq \cdots \geq s_{[q]}(\mathbf{A},\mathbf{B})$. Then we have

$$N_k(\mathbf{A}-\mathbf{B}) = \sum_{i=1}^k \sigma_i(\mathbf{A}-\mathbf{B}) \geq \sum_{i=1}^k s_{[i]}(\mathbf{A},\mathbf{B}) = N_k(\mathbf{\Sigma}(\mathbf{A}) - \mathbf{\Sigma}(\mathbf{B})), \tag{15}$$

where the inequality is obtained by Lemma 1. The Lemma 2 implies (15) is equivalent to $\|\mathbf{A}-\mathbf{B}\| \geq \|\mathbf{\Sigma}(\mathbf{A}) - \mathbf{\Sigma}(\mathbf{B})\|$ for every unitarily invariant norm $\|\cdot\|$. ■

Then we give the proof of Theorem 1 as follows:

Proof Using the notation in above lemmas, we can bound the objective function as follows

$$\begin{aligned}
\left\|\mathbf{M}-\mathbf{C}\mathbf{C}^\top-\delta\mathbf{I}_d\right\|_2 &\geq \left\|\mathbf{\Sigma}(\mathbf{M})-\mathbf{\Sigma}(\mathbf{C}\mathbf{C}^\top+\delta\mathbf{I}_d)\right\|_2 \\
&= \max_{i\in\{1,\ldots,d\}} \left|\sigma_i(\mathbf{M})-\sigma_i(\mathbf{C}\mathbf{C}^\top)-\delta\right| \\
&\geq \max_{i\in\{k+1,\ldots,d\}} \left|\sigma_i(\mathbf{M})-\sigma_i(\mathbf{C}\mathbf{C}^\top)-\delta\right| \\
&= \max_{i\in\{k+1,\ldots,d\}} \left|\sigma_i(\mathbf{M})-\delta\right| \\
&\geq \max_{i\in\{k+1,\ldots,d\}} \left|\sigma_i(\mathbf{M})-\widehat{\delta}\right|.
\end{aligned}$$

The first inequality is obtained by Lemma 3 since the spectral norm is unitarily invariant, and the second inequality is the property of maximization operator. The last inequality can be checked easily by the property of max operation and the equivalence of SVD and eigenvector decomposition for positive semi-definite matrix. The first equality is based on the definition of spectral norm. The second equality holds due to the fact $\mathrm{rank}(\mathbf{C}\mathbf{C}^\top) \leq k$ which leads $\sigma_i(\mathbf{C}\mathbf{C}^\top) = 0$ for any $i > k$. Note that all above equalities occur for $\mathbf{C} = \widehat{\mathbf{C}} = \mathbf{U}_k(\mathbf{\Sigma}_k - \xi\mathbf{I}_k)^{1/2}\mathbf{Q}$, $\delta = \widehat{\delta}$ and $\xi \in [\sigma_d, \sigma_{k+1}]$. Hence we prove the optimality of $(\widehat{\mathbf{C}}, \widehat{\delta})$.

The approximation error of rank-k SVD corresponds to the objective of (8) by taking $\mathbf{C} = \mathbf{U}_k(\mathbf{\Sigma}_k)^{1/2}$ and $\delta = 0$, which is impossible to be smaller than the minimum. It is easy to verify we have $\widehat{\mathbf{C}} = \mathbf{U}_k(\mathbf{\Sigma}_k)^{1/2}$ and $\widehat{\delta} = 0$ if and only of $\mathrm{rank}(\mathbf{M}) \leq k$. ■

Theorem 1 means the choice of ξ in the solution of problem (9) is not unique, but taking $\xi = \sigma_{k+1}$ minimizes the condition number of $\widehat{\mathbf{C}}\widehat{\mathbf{C}}^\top + \widehat{\delta}\mathbf{I}_d$. Hence, we use $\xi = \sigma_{k+1}$ in the derivation of RFD. We also demonstrate similar result with respect to Frobenius norm in Corollary 1. This analysis includes the global optimality of the problem, while Zhang (2014)'s analysis only prove the solution is locally optimal.

Corollary 1 *Using the same notation in Theorem 1, the pair* $(\tilde{\mathbf{C}}, \tilde{\delta})$ *defined as*

$$\tilde{\mathbf{C}} = \mathbf{U}_k(\mathbf{\Sigma}_k - \tilde{\delta}\mathbf{I}_k)^{1/2}\mathbf{V} \quad \textit{and} \quad \tilde{\delta} = \frac{1}{d-k}\sum_{i=j+1}^{d}\sigma_i$$

is the global minimizer of

$$\min_{\mathbf{C}\in\mathbb{R}^{d\times k},\delta\in\mathbb{R}} \|\mathbf{M}-\mathbf{C}\mathbf{C}^\top-\delta\mathbf{I}_d\|_F^2,$$

where $\mathbf{V}$ *is an arbitrary* $k \times k$ *orthogonal matrix.*

Proof We have the result similar to Theorem 1.

$$\begin{aligned}
\|\mathbf{M}-\mathbf{C}\mathbf{C}^\top-\delta\mathbf{I}_d\|_F^2 &\geq \|\mathbf{\Sigma}(\mathbf{M})-\mathbf{\Sigma}(\mathbf{C}\mathbf{C}^\top+\delta\mathbf{I}_d)\|_F^2 \\
&= \sum_{i=1}^{d}(\sigma_i(\mathbf{M})-\sigma_i(\mathbf{C}\mathbf{C}^\top)-\delta)^2 \\
&\geq \sum_{i=k+1}^{d}(\sigma_i(\mathbf{M})-\sigma_i(\mathbf{C}\mathbf{C}^\top)-\delta)^2 \\
&= \sum_{i=k+1}^{d}(\sigma_i(\mathbf{M})-\delta)^2 \\
&\geq \sum_{i=k+1}^{d}(\sigma_i(\mathbf{M})-\tilde{\delta})^2
\end{aligned}$$

The first four steps are similar to the ones of Theorem 1, but replace the spectral norm and absolute operator with Frobenius norm and square function. The last step comes from the property of the mean value.

We can check that all above equalities occur for $\mathbf{C} = \tilde{\mathbf{C}}$ and $\delta = \tilde{\delta}$, which completes the proof. ■

Appendix C. The Proof of Theorem 2

Proof The Algorithm 2 implies the singular values of $(\widehat{\mathbf{B}}^{(t-1)})^\top\widehat{\mathbf{B}}^{(t-1)} + \alpha^{(t-1)}\mathbf{I}$ are

$$(\sigma_1^{(t-1)})^2 + \alpha^{(t-1)} \geq \cdots \geq (\sigma_m^{(t-1)})^2 + \alpha^{(t-1)} \geq \alpha^{(t-1)} = \cdots = \alpha^{(t-1)}.$$

Then we can use Theorem 1 by taking

$$\begin{aligned}
\mathbf{M} &= (\widehat{\mathbf{B}}^{(t-1)})^\top\widehat{\mathbf{B}}^{(t-1)} + \alpha^{(t-1)}\mathbf{I}, \\
k &= m-1, \\
\xi &= \sigma_{k+1} = (\sigma_m^{(t-1)})^2 + \alpha^{(t-1)} \\
\widehat{\mathbf{C}} &= \mathbf{V}_{m-1}^{(t-1)}\sqrt{\left(\mathbf{\Sigma}_{m-1}^{(t-1)}\right)^2 - \left(\sigma_m^{(t-1)}\right)^2\mathbf{I}_{m-1}} = (\mathbf{B}^{(t)})^\top, \\
\widehat{\delta} &= [(\sigma_m^{(t-1)})^2 + \alpha^{(t-1)} + \alpha^{(t-1)}]/2 = \alpha^{(t)},
\end{aligned}$$

which just means that $(\mathbf{B}^{(t)}, \alpha^{(t)})$ is the minimizer of the problem in this theorem. ■

Appendix D. The Proof of Theorem 3

The algorithms of FD and RFD share the same $\mathbf{B}^{(t)}$ and we have Lemma 4 (Ghashami et al., 2016) as follows.

Lemma 4 *For any $k < m$ and using the notation of Algorithm 1 or Algorithm 2, we have*

$$\mathbf{A}^\top\mathbf{A} - \mathbf{B}^\top\mathbf{B} \succeq \mathbf{0}, \tag{16}$$

$$\sum_{t=1}^{T-1}(\sigma_m^{(t)})^2 \leq \frac{1}{m-k}\|\mathbf{A} - [\mathbf{A}]_k\|_F^2. \tag{17}$$

Then we prove the Theorem 3 based on Lemma 4.

Proof Define $(\mathbf{B}^{(0)})^\top\mathbf{B}^{(0)} = \mathbf{0}^{d\times d}$, then we can derive the error bound as follows

$$\begin{aligned}
&\left\|\mathbf{A}^\top\mathbf{A} - (\mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d)\right\|_2\\
=&\left\|\sum_{t=1}^{T}\left[(\mathbf{a}^{(t)})^\top\mathbf{a}^{(t)} - (\mathbf{B}^{(t)})^\top\mathbf{B}^{(t)} + (\mathbf{B}^{(t-1)})^\top\mathbf{B}^{(t-1)} - \frac{1}{2}(\sigma_m^{(t-1)})^2\mathbf{I}_d\right]\right\|_2\\
\leq&\sum_{t=1}^{T}\left\|(\mathbf{a}^{(t)})^\top\mathbf{a}^{(t)} - (\mathbf{B}^{(t)})^\top\mathbf{B}^{(t)} + (\mathbf{B}^{(t-1)})^\top\mathbf{B}^{(t-1)} - \frac{1}{2}(\sigma_m^{(t-1)})^2\mathbf{I}_d\right\|_2\\
=&\sum_{t=1}^{T}\left\|\mathbf{V}_{m-1}^{(t-1)}(\mathbf{\Sigma}_{m-1}^{(t-1)})^2(\mathbf{V}_{m-1}^{(t-1)})^\top - \mathbf{V}_{m-1}^{(t-1)}[(\mathbf{\Sigma}_{m-1}^{(t-1)})^2 - (\sigma_m^{(t-1)})^2\mathbf{I}_d](\mathbf{V}_{m-1}^{(t-1)})^\top - \frac{1}{2}(\sigma_m^{(t-1)})^2\mathbf{I}_d\right\|_2\\
=&\sum_{t=1}^{T}\left\|(\sigma_m^{(t-1)})^2\mathbf{V}_m^{(t-1)}(\mathbf{V}_m^{(t-1)})^\top - \frac{1}{2}(\sigma_m^{(t-1)})^2\mathbf{I}_d\right\|_2\\
=&\frac{1}{2}\sum_{t=1}^{T-1}(\sigma_m^{(t)})^2\\
\leq&\frac{1}{2(m-k)}\|\mathbf{A} - [\mathbf{A}]_k\|_F^2.
\end{aligned}$$

The first three equalities are direct from the procedure of the algorithm, and the last one is based on the fact that $\mathbf{V}^{(t-1)}$ is column orthonormal. The first inequality comes from the triangle inequality of spectral norm. The last one can be obtained by the result (17) of Lemma 4. ■

We also have similar error bound for fast RFD with doubling space. We first rewrite Algorithm 5 as the block formulation. Consider the procedure of Algorithm 5, we suppose that matrix $\widehat{\mathbf{B}}^{(t-1)}$ has $2m$ rows at round $t = p_1, p_2, \ldots, p_{T'-1}$, where $1 < p_1 < p_2 < \cdots < p_{T'-1} \leq T$. Letting $p_0 = 0$ and $p_{T'} = T$, we can partition matrix $\mathbf{A}$ into T' blocks

$$\mathbf{A} = \left[\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \cdots, \mathbf{A}^{(T')}\right]^\top, \tag{18}$$

where

$$\mathbf{A}^{(t')} = \left[\mathbf{a}^{(p_{t'-1}+1)}, \mathbf{a}^{(p_{t'-1}+2)}, \cdots, \mathbf{a}^{(p_{t'})}\right]^\top \text{ for } t' = 1, 2, \ldots, T'. \tag{19}$$

Based on the notation of (18) and (19), we can rewrite Algorithm 5 as Algorithm 7. It is obvious that two algorithms have the same output results. We present Lemma 5 which extends Lemma 4 to block version and establishes the error bound for Algorithm 7 in Corollary 2.

Algorithm 7 Fast Robust Frequent Directions (Block Formulation)

Input: $\mathbf{A} = [\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(T')}]^\top \in \mathbb{R}^{T\times d}$, $\mathbf{B}^{(1)} = (\mathbf{A}^{(1)})^\top$, $\alpha^{(1)} = 0$

for $t' = 2, \ldots, T'$ **do**

$\quad\widehat{\mathbf{B}}^{(t-1)} = \begin{bmatrix}\mathbf{B}^{(t'-1)}\\ (\mathbf{A}^{(t')})^\top\end{bmatrix}$

$\quad$Compute SVD: $\widehat{\mathbf{B}}^{(t'-1)} = \mathbf{U}^{(t'-1)}\mathbf{\Sigma}^{(t'-1)}(\mathbf{V}^{(t'-1)})^\top$

$\quad\mathbf{B}^{(t')} = \sqrt{\left(\mathbf{\Sigma}_{m-1}^{(t'-1)}\right)^2 - \left(\sigma_m^{(t'-1)}\right)^2\mathbf{I}_{m-1}}\cdot\left(\mathbf{V}_{m-1}^{(t'-1)}\right)^\top$

$\quad\alpha^{(t')} = \alpha^{(t'-1)} + \left(\sigma_m^{(t'-1)}\right)^2/2$

end for

Output: $\mathbf{B} = \mathbf{B}^{(T')}$ and $\alpha = \alpha^{(T')}$

Lemma 5 *For any $k < m$ and using the notation of Algorithm 7, we have*

$$\sum_{t'=1}^{T'-1}(\sigma_m^{(t')})^2 \leq \frac{1}{m-k}\|\mathbf{A} - [\mathbf{A}]_k\|_F^2.$$

Proof We let $\sigma_m^{(T')} = 0$. For any unit vector $\mathbf{x} \in \mathbb{R}^d$, the procedure of Algorithm 7 implies

$$\begin{aligned}
&\|\mathbf{A}\mathbf{x}\|^2 - \|\mathbf{B}\mathbf{x}\|^2\\
=&\sum_{t'=1}^{T'}\left(\|\mathbf{A}^{(t')}\mathbf{x}\|^2 + \|\mathbf{B}^{(t'-1)}\mathbf{x}\|^2 - \|\mathbf{B}^{(t')}\mathbf{x}\|^2\right)\\
=&\sum_{t'=1}^{T'}\mathbf{x}^\top\left((\mathbf{A}^{(t')})^\top\mathbf{A}^{(t')} + (\mathbf{B}^{(t'-1)})^\top(\mathbf{B}^{(t'-1)}) - (\mathbf{B}^{(t')})^\top\mathbf{B}^{(t')}\right)\mathbf{x}\\
=&\sum_{t'=1}^{T'}\mathbf{x}^\top\left((\widehat{\mathbf{B}}^{(t')})^\top\widehat{\mathbf{B}}^{(t')} - (\mathbf{B}^{(t')})^\top\mathbf{B}^{(t')}\right)\mathbf{x}\\
\leq&\sum_{t'=1}^{T'}\left(\sigma_m^{(t')}\right)^2.
\end{aligned} \tag{20}$$

Using the property of Frobenius norm, we have

$$\begin{aligned}
\|\widehat{\mathbf{B}}^{(t'-1)}\|_F^2 =&\|\mathbf{\Sigma}^{(t'-1)}\|_F^2\\
\geq&\left\|\sqrt{\left(\mathbf{\Sigma}_m^{(t'-1)}\right)^2 - \left(\sigma_m^{(t'-1)}\right)^2\mathbf{I}_m}\right\|_F^2 + m(\sigma_m^{(t'-1)})^2\\
=&\|\mathbf{B}^{(t')}\|_F^2 + m(\sigma_m^{(t'-1)})^2.
\end{aligned} \tag{21}$$

The term $\|\mathbf{A}\|_F^2$ satisfies

$$\|\mathbf{A}\|_F^2 = \sum_{t'=1}^{T'}\|\mathbf{A}^{(t')}\|_F^2$$

$$
\begin{aligned}
&= \sum_{t'=1}^{T'} \left(\|\widehat{\mathbf{B}}^{(t'-1)}\|_F^2 - \|\mathbf{B}^{(t'-1)}\|_F^2 \right) \\
&\geq \sum_{t'=1}^{T'} \left(\|\mathbf{B}^{(t')}\|_F^2 + m(\sigma_m^{(t'-1)})^2 - \|\mathbf{B}^{(t'-1)}\|_F^2 \right) \\
&= \|\mathbf{B}\|_F^2 + m \sum_{t'=1}^{T'} \left(\sigma_m^{(t')}\right)^2, \qquad (22)
\end{aligned}
$$

where the inequality is due to (21).

Let $\mathbf{y}_i$ be the singular vectors of $\mathbf{A}$ with respect to $\sigma_i(\mathbf{A})$. Then we have

$$
\begin{aligned}
m \sum_{t'=1}^{T'} (\sigma_m^{(t'-1)})^2 \leq & \|\mathbf{A}\|_F^2 - \|\mathbf{B}\|_F^2 \\
= & \sum_{i=1}^{k} \|\mathbf{A}\mathbf{y}_i\|_F^2 + \sum_{i=k+1}^{d} \|\mathbf{A}\mathbf{y}_i\|_F^2 - \|\mathbf{B}\|_F^2 \\
= & \sum_{i=1}^{k} \|\mathbf{A}\mathbf{y}_i\|_F^2 + \|\mathbf{A} - [\mathbf{A}]_k\|_F^2 - \|\mathbf{B}\|_F^2 \\
\leq & \|\mathbf{A} - [\mathbf{A}]_k\|_F^2 + \sum_{i=1}^{k} \left(\|\mathbf{A}\mathbf{y}_i\|_F^2 - \|\mathbf{B}\mathbf{y}_i\|_F^2 \right) \\
\leq & \|\mathbf{A} - [\mathbf{A}]_k\|_F^2 + k \sum_{t'=1}^{T'} \left(\sigma_m^{(t')}\right)^2, \qquad (23)
\end{aligned}
$$

where the first inequality comes from (22), the second inequality is based on the fact $\sum_{i=1}^{k} \|\mathbf{B}\mathbf{y}_i\|^2 \leq \|\mathbf{B}\|_F^2$, and the last one comes from (20). We can obtain the result of this lemma by (23) directly. ■

Corollary 2 *For any $k < m$ and using the notation of Algorithm 7, we have*

$$
\left\|\mathbf{A}^\top\mathbf{A} - (\mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d)\right\|_2 \leq \frac{1}{2(m-k)} \|\mathbf{A} - [\mathbf{A}]_k\|_F^2,
$$

where $[\mathbf{A}]_k$ is the best rank-k approximation to $\mathbf{A}$ in both the Frobenius and spectral norms.

Proof Define $(\mathbf{B}^{(0)})^\top \mathbf{B}^{(0)} = \mathbf{0}^{d\times d}$, then we can derive the error bound as follows

$$
\begin{aligned}
&\left\|\mathbf{A}^\top\mathbf{A} - (\mathbf{B}^\top\mathbf{B} + \alpha\mathbf{I}_d)\right\|_2 \\
=& \left\| \sum_{t'=1}^{T'} \left[(\mathbf{A}^{(t')})^\top \mathbf{A}^{(t')} - (\mathbf{B}^{(t')})^\top \mathbf{B}^{(t')} + (\mathbf{B}^{(t'-1)})^\top \mathbf{B}^{(t'-1)} - \frac{1}{2}(\sigma_m^{(t'-1)})^2 \mathbf{I}_d \right] \right\|_2 \\
\leq & \sum_{t'=1}^{T'} \left\| (\mathbf{A}^{(t')})^\top \mathbf{A}^{(t')} - (\mathbf{B}^{(t')})^\top \mathbf{B}^{(t')} + (\mathbf{B}^{(t'-1)})^\top \mathbf{B}^{(t'-1)} - \frac{1}{2}(\sigma_m^{(t'-1)})^2 \mathbf{I}_d \right\|_2
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{t'=1}^{T'} \left\| \mathbf{V}_{m-1}^{(t'-1)} (\mathbf{\Sigma}_{m-1}^{(t'-1)})^2 (\mathbf{V}_{m-1}^{(t'-1)})^\top - \mathbf{V}_{m-1}^{(t'-1)} [(\mathbf{\Sigma}_{m-1}^{(t'-1)})^2 - (\sigma_m^{(t'-1)})^2 \mathbf{I}_d] (\mathbf{V}_{m-1}^{(t'-1)})^\top - \frac{1}{2}(\sigma_m^{(t'-1)})^2 \mathbf{I}_d \right\|_2 \\
&= \sum_{t'=1}^{T'} \left\| (\sigma_m^{(t'-1)})^2 \mathbf{V}_m^{(t'-1)} (\mathbf{V}_m^{(t'-1)})^\top - \frac{1}{2}(\sigma_m^{(t'-1)})^2 \mathbf{I}_d \right\|_2 \\
&= \frac{1}{2} \sum_{t'=1}^{T'-1} (\sigma_m^{(t')})^2 \\
&\leq \frac{1}{2(m-k)} \|\mathbf{A} - [\mathbf{A}]_k\|_F^2.
\end{aligned}
$$

The last inequality is based on Lemma 5 and other steps are similar to the proof of Theorem 3. ■

Appendix E. The Proof of Theorem 4

Proof We can compare $\kappa(\mathbf{M}_{\text{RFD}})$ and $\kappa(\mathbf{M}_{\text{FD}})$ by the fact $\alpha \geq \alpha^{(0)}$ as follows

$$
\begin{aligned}
\kappa(\mathbf{M}_{\text{RFD}}) &= \frac{\sigma_{\max}(\mathbf{B}^\top\mathbf{B}) + \alpha}{\alpha} \\
&\leq \frac{\sigma_{\max}(\mathbf{B}^\top\mathbf{B}) + \alpha_0}{\alpha_0} \\
&= \kappa(\mathbf{M}_{\text{FD}}).
\end{aligned}
$$

The other inequality can be derived as

$$
\begin{aligned}
\kappa(\mathbf{M}_{\text{RFD}}) &= \frac{\sigma_{\max}(\mathbf{B}^\top\mathbf{B}) + \alpha}{\alpha} \\
&\leq \frac{\sigma_{\max}(\mathbf{A}^\top\mathbf{A}) + \alpha}{\alpha} \\
&\leq \frac{\sigma_{\max}(\mathbf{A}^\top\mathbf{A}) + \alpha_0}{\alpha_0} \\
&= \kappa(\mathbf{M}),
\end{aligned}
$$

where the first inequality comes from (16) of Lemma 4 and the others are easy to obtain. ■

Appendix F. The Greedy Low-rank Approximation

We present the greedy low-rank approximation (Brand, 2002; Hall et al., 1998; Levey and Lindenbaum, 2000; Ross et al., 2008) as Algorithm 8. The algorithm does not work in general although $(\mathbf{B}'^{(t)})^\top \mathbf{B}'^{(t)}$ is the best low-rank approximation to $(\widehat{\mathbf{B}}^{(t)})^\top \widehat{\mathbf{B}}^{(t)}$.

We provide an example to show the failure of this method. We define $\tilde{\mathbf{A}} = [\tilde{\mathbf{A}}_1^\top, \tilde{\mathbf{A}}_2^\top]^\top \in \mathbb{R}^{(m-s+1)\times d}$, where $\tilde{\mathbf{A}}_1 \in \mathbb{R}^{(m-1)\times d}$, $\tilde{\mathbf{A}}_2 \in \mathbb{R}^{s\times d}$ and $m \ll s$. Suppose that the smallest singular value of $\tilde{\mathbf{A}}_1$ is λ, and each row of $\tilde{\mathbf{A}}_2$ is $\mathbf{a} \in \mathbb{R}^d$ that satisfies $\|\mathbf{a}\| = \lambda - \epsilon$ and $\tilde{\mathbf{A}}_1^\top \mathbf{a} = \mathbf{0}$, where ϵ is a very small positive number. Since s is much larger than m, a good approximation to $\tilde{\mathbf{A}}^\top\tilde{\mathbf{A}}$ is

Algorithm 8 Greedy Low-rank Approximation

Input: $\mathbf{A} = [\mathbf{a}^{(1)}, \dots, \mathbf{a}^{(T)}]^\top \in \mathbb{R}^{T\times d}$, $\mathbf{B}'^{(m-1)} = [\mathbf{a}^{(1)}, \dots, \mathbf{a}^{(m-1)}]^\top$
for $t = m, \dots, T$ **do**
  $\widehat{\mathbf{B}}^{(t-1)} = \begin{bmatrix} \mathbf{B}'^{(t-1)} \\ (\mathbf{a}^{(t)})^\top \end{bmatrix}$
  Compute SVD: $\widehat{\mathbf{B}}^{(t-1)} = \mathbf{U}^{(t-1)}\mathbf{\Sigma}^{(t-1)}(\mathbf{V}^{(t-1)})^\top$
  $\mathbf{B}'^{(t)} = \mathbf{\Sigma}_{m-1}^{(t-1)}(\mathbf{V}_{m-1}^{(t-1)})^\top$
end for
Output: $\mathbf{B}' = \mathbf{B}'^{(T)}$

dominated by $\tilde{\mathbf{A}}_2^\top\tilde{\mathbf{A}}_2$. If we use Algorithm 8 with $\mathbf{A} = \tilde{\mathbf{A}}$, any row of $\tilde{\mathbf{A}}_2$ will be neglected because the m-th singular value of $\widehat{\mathbf{B}}^{(t-1)}$ is $\|\mathbf{a}\| = \lambda - \epsilon < \lambda$, which leads to the fact that output is $\mathbf{B}' = \tilde{\mathbf{A}}_1$. Apparently, $\tilde{\mathbf{A}}_1^\top\tilde{\mathbf{A}}_1$ is not a good approximation to $\mathbf{A}^\top\mathbf{A}$. Hence the shrinking of FD or RFD is necessary. In this example, it reduces the impact of $\tilde{\mathbf{A}}_1$ and let $\tilde{\mathbf{A}}_2$ be involved in final result.

Besides above discussion, Desai et al. (2016) has shown that the greedy algorithm is much worse than FD based methods on data sets "Adversarial" and "ConnectUS".

Appendix G. The Proof of Theorem 5

Lemma 6 shows the general regret bound for any choice of $\mathbf{H}^{(t)} \succ \mathbf{0}$ in update (2)

$$\mathbf{u}^{(t+1)} = \mathbf{w}^{(t)} - \beta_t(\mathbf{H}^{(t)})^{-1}\mathbf{g}^{(t)},$$
$$\mathbf{w}^{(t+1)} = \operatorname*{argmin}_{\mathbf{w}\in\mathcal{K}_{t+1}} \|\mathbf{w} - \mathbf{u}^{(t+1)}\|_{\mathbf{H}^{(t)}}.$$

Lemma 6 (Proposition 1 of Luo et al. (2016)) *For any sequence of positive definite matrices $\mathbf{H}^{(t)}$ and sequences of losses satisfying Assumption 1 and 2, regret of updates (2) satisfies*

$$2R_T(\mathbf{w}) \le \|\mathbf{w}\|^2_{\mathbf{H}^{(0)}} + R_G + R_D,$$

where

$$R_G = \sum_{t=1}^T (\mathbf{g}^{(t)})^\top(\mathbf{H}^{(t)})^{-1}\mathbf{g}^{(t)},$$
$$R_D = \sum_{t=1}^T (\mathbf{w}^{(t)} - \mathbf{w})^\top(\mathbf{H}^{(t)} - \mathbf{H}^{(t-1)} - \mu^{(t)}\mathbf{g}^{(t)}(\mathbf{g}^{(t)})^\top)(\mathbf{w}^{(t)} - \mathbf{w}).$$

Then we prove the regret bound for RFD-SON based on Lemma 6 and property of RFD.

Proof Let $\mathbf{V}_\perp^{(t)}$ be the orthogonal complement of $\mathbf{V}_{m-1}^{(t)}$'s column space, that is $\mathbf{V}_{m-1}^{(t)}(\mathbf{V}_{m-1}^{(t)})^\top + \mathbf{V}_\perp^{(t)}(\mathbf{V}_\perp^{(t)})^\top = \mathbf{I}_d$, then we have

$$\begin{aligned}
&\mathbf{H}^{(t)} - \mathbf{H}^{(t-1)} \\
=&\alpha^{(t)}\mathbf{I} + (\mathbf{B}^{(t)})^\top\mathbf{B}^{(t)} - \alpha^{(t-1)}\mathbf{I} - (\mathbf{B}^{(t-1)})^\top\mathbf{B}^{(t-1)} \\
=&\frac{1}{2}(\sigma_m^{(t-1)})^2\mathbf{I} - (\sigma_m^{(t-1)})^2\mathbf{V}_{m-1}^{(t-1)}(\mathbf{V}_{m-1}^{(t-1)})^\top + (\mu_t + \eta_t)\mathbf{g}^{(t)}(\mathbf{g}^{(t)})^\top \\
=&\frac{1}{2}(\sigma_m^{(t-1)})^2\big[\mathbf{V}_\perp^{(t-1)}(\mathbf{V}_\perp^{(t-1)})^\top - \mathbf{V}_{m-1}^{(t-1)}(\mathbf{V}_{m-1}^{(t-1)})^\top\big] + (\mu_t + \eta_t)\mathbf{g}^{(t)}(\mathbf{g}^{(t)})^\top. \qquad (24)
\end{aligned}$$

Since $\mathbf{H}^{(t)}$ is positive semidefinite for any t, we have $(\mathbf{H}^{(t)})^\dagger = (\mathbf{H}^{(t)})^{-1}$. Combining with Lemma 6, we have

$$2R_T(\mathbf{w}) \le \alpha_0\|\mathbf{w}\|^2 + R_G + R_D,$$

where

$$R_G = \sum_{t=1}^T (\mathbf{g}^{(t)})^\top(\mathbf{H}^{(t)})^{-1}\mathbf{g}^{(t)},$$

and

$$R_D = \sum_{t=1}^T (\mathbf{w}^{(t)} - \mathbf{w})^\top[\mathbf{H}^{(t)} - \mathbf{H}^{(t-1)} - \mu_t\mathbf{g}^{(t)}(\mathbf{g}^{(t)})^\top](\mathbf{w}^{(t)} - \mathbf{w}).$$

We can bound R_G as follows

$$\begin{aligned}
&\sum_{t=1}^T (\mathbf{g}^{(t)})^\top(\mathbf{H}^{(t)})^{-1}\mathbf{g}^{(t)} \\
=&\sum_{t=1}^T \big\langle(\mathbf{H}^{(t)})^{-1}, \mathbf{g}^{(t)}(\mathbf{g}^{(t)})^\top\big\rangle \\
=&\sum_{t=1}^T \frac{1}{\mu_t + \eta_t}\big\langle(\mathbf{H}^{(t)})^{-1}, \mathbf{H}^{(t)} - \mathbf{H}^{(t-1)} + \frac{1}{2}(\sigma_m^{(t-1)})^2[\mathbf{V}_\perp^{(t-1)}(\mathbf{V}_\perp^{(t-1)})^\top - \mathbf{V}^{(t-1)}(\mathbf{V}^{(t-1)})^\top]\big\rangle \\
\le&\frac{1}{\mu + \eta_T}\sum_{t=1}^T \big\langle(\mathbf{H}^{(t)})^{-1}, \mathbf{H}^{(t)} - \mathbf{H}^{(t-1)} + \frac{1}{2}(\sigma_m^{(t-1)})^2\mathbf{V}^{(t-1)}(\mathbf{V}^{(t-1)})^\top\big\rangle \\
=&\frac{1}{\mu + \eta_T}\sum_{t=1}^T \Big[\big\langle(\mathbf{H}^{(t)})^{-1}, \mathbf{H}^{(t)} - \mathbf{H}^{(t-1)}\big\rangle + \frac{1}{2}(\sigma_m^{(t-1)})^2\mathrm{tr}\big(\mathbf{V}^{(t-1)}(\mathbf{H}^{(t)})^{-1}(\mathbf{V}^{(t-1)})^\top\big)\Big].
\end{aligned}$$

The above equalities come from the properties of trace operator and (24) and the inequality is due to the fact that η_t is non-increasing.

The term $\sum_{t=1}^T\langle(\mathbf{H}^{(t)})^{-1}, \mathbf{H}^{(t)} - \mathbf{H}^{(t-1)}\rangle$ can be bounded as

$$\begin{aligned}
\sum_{t=1}^T\big\langle(\mathbf{H}^{(t)})^{-1}, \mathbf{H}^{(t)} - \mathbf{H}^{(t-1)}\big\rangle &\le \sum_{t=1}^T \ln\frac{\det(\mathbf{H}^{(t)})}{\det(\mathbf{H}^{(t-1)})} \\
&= \ln\frac{\det(\mathbf{H}^{(T)})}{\det(\mathbf{H}^{(0)})}
\end{aligned}$$

$$
\begin{aligned}
&= \ln \frac{\prod_{i=1}^{d} \sigma_i(\mathbf{H}^{(T)})}{\alpha_0} \\
&= \sum_{i=1}^{d} \ln \frac{\sigma_i\big((\mathbf{B}^{(T)})^\top \mathbf{B}^{(T)} + \alpha^{(T)} \mathbf{I}_d\big)}{\alpha_0} \\
&= \sum_{i=1}^{m} \ln \frac{\sigma_i\big((\mathbf{B}^{(T)})^\top \mathbf{B}^{(T)}\big) + \alpha^{(T)}}{\alpha_0} + (d-m) \ln \frac{\alpha^{(T)}}{\alpha_0} \\
&\leq m \ln \frac{\sum_{i=1}^{m} [\sigma_i\big((\mathbf{B}^{(T)})^\top \mathbf{B}^{(T)}\big) + \alpha^{(T)}]}{m\alpha_0} + (d-m) \ln \frac{\alpha^{(T)}}{\alpha_0} \\
&= m \ln \Big(\frac{\mathrm{tr}\big((\mathbf{B}^{(T)})^\top \mathbf{B}^{(T)}\big)}{m\alpha_0} + \frac{\alpha^{(T)}}{\alpha_0} \Big) + (d-m) \ln \frac{\alpha^{(T)}}{\alpha_0}.
\end{aligned}
$$

The first inequality is obtained by the concavity of the log determinant function (Boyd and Vandenberghe, 2004), the second inequality comes from the Jensen's inequality and the other steps are based on the procedure of the algorithm.

The other one $\frac{1}{2}\sum_{t=1}^{T} (\sigma_m^{(t)})^2 \mathrm{tr}\big(\mathbf{V}^{(t)} (\mathbf{H}^{(t)})^{-1} (\mathbf{V}^{(t)})^\top\big)$ can be bounded as

$$
\begin{aligned}
\frac{1}{2}\sum_{t=1}^{T} (\sigma_m^{(t)})^2 \mathrm{tr}\big(\mathbf{V}^{(t)} (\mathbf{H}^{(t)})^{-1} (\mathbf{V}^{(t)})^\top\big) &\leq \frac{1}{2}\sum_{t=1}^{T} \frac{(\sigma_m^{(t)})^2}{\alpha^{(t)}} \mathrm{tr}\big(\mathbf{V}^{(t)} (\mathbf{V}^{(t)})^\top\big) \\
&= \frac{m}{2}\sum_{t=1}^{T} \frac{(\sigma_m^{(t)})^2}{\alpha^{(t)}}. \qquad (25)
\end{aligned}
$$

Hence, we have

$$
R_G \leq \frac{1}{\mu + \eta_T} \left[m \ln \Big(\mathrm{tr}\big((\mathbf{B}^{(T)})^\top \mathbf{B}^{(T)}\big) + \frac{\alpha^{(T)}}{\alpha_0} \Big) + (d-m) \ln \frac{\alpha^{(T)}}{\alpha_0} + \frac{m}{2}\sum_{t=1}^{T} \frac{(\sigma_m^{(t)})^2}{\alpha^{(t)}} \right]. \qquad (26)
$$

Then we bound the term R_D by using equation (24), Assumption 1 and Assumption 2.

$$
\begin{aligned}
R_D =& \sum_{t=1}^{T} (\mathbf{w}^{(t)} - \mathbf{w})^\top \big[\eta_t \mathbf{g}^{(t)} (\mathbf{g}^{(t)})^\top + \frac{1}{2}(\sigma_m^{(t-1)})^2 \mathbf{I} - \mathbf{V}^{(t-1)} (\mathbf{V}^{(t-1)})^\top\big] (\mathbf{w}^{(t)} - \mathbf{w}) \\
\leq& \sum_{t=1}^{T} \eta_t (\mathbf{w}^{(t)} - \mathbf{w})^\top \mathbf{g}^{(t)} (\mathbf{g}^{(t)})^\top (\mathbf{w}^{(t)} - \mathbf{w}) + \frac{1}{2}\sum_{t=1}^{T} (\sigma_m^{(t-1)})^2 (\mathbf{w}^{(t)} - \mathbf{w})^\top (\mathbf{w}^{(t)} - \mathbf{w}) \\
\leq& 4(CL)^2 \sum_{t=1}^{T} \eta_t + 2C^2 \sum_{t=1}^{T} (\sigma_m^{(t-1)})^2. \qquad (27)
\end{aligned}
$$

Finally, we obtain the result by combining (26) and (27). ■

Additionally, the term $\sum_{t=1}^{T} \frac{(\sigma_m^{(t)})^2}{\alpha^{(t)}}$ in Ω_{RFD} can be bounded by $\mathcal{O}(\ln T)$ if we further assume all $\sigma_m^{(t)}$ are bounded by positive constants. Exactly, suppose that $0 < C_1 \leq \big(\sigma_m^{(t)}\big)^2 \leq C_2$ for any

$t = 0, \ldots, T-1$, then we have

$$
\begin{aligned}
\sum_{t=1}^{T} \frac{(\sigma_m^{(t)})^2}{\alpha^{(t)}} &= \sum_{t=1}^{T} \frac{(\sigma_m^{(t)})^2}{\alpha_0 + \frac{1}{2}\sum_{i=0}^{t-1} \big(\sigma_m^{(i)}\big)^2} \\
&\leq \sum_{t=1}^{T} \frac{C_2}{\alpha_0 + \frac{1}{2} C_1 t} \\
&\leq \frac{2C_2}{C_1} \sum_{t=1}^{T} \frac{1}{t} \\
&= \mathcal{O}(\ln T).
\end{aligned}
$$

The last inequality is due to the property of harmonic series.

Appendix H. The Proof of Theorem 6

Considering the update without positive semidefinite assumption on $\mathbf{H}^{(t)}$

$$
\begin{aligned}
\mathbf{u}^{(t+1)} &= \mathbf{w}^{(t)} - \beta_t (\mathbf{H}^{(t)})^\dagger \mathbf{g}^{(t)}, \\
\mathbf{w}^{(t+1)} &= \operatorname*{argmin}_{\mathbf{w} \in \mathcal{K}_{t+1}} \|\mathbf{w} - \mathbf{u}^{(t+1)}\|_{\mathbf{H}^{(t)}}, \qquad (28)
\end{aligned}
$$

we have the results as follows.

Lemma 7 (Appendix D of Luo et al. (2016)) *Let $\widehat{\mathbf{H}}^{(t)} = \sum_{s=1}^{t} \mathbf{g}^{(t)} (\mathbf{g}^{(t)})^\top$ with $\mathrm{rank}(\widehat{\mathbf{H}}^{(T)}) = r$ and σ^* be the minimum among the smallest non-zeros singular values of $\widehat{\mathbf{H}}^{(T)}$. Then the regret of update (28) satisfies*

$$
R_T(\mathbf{w}) \leq 2(CL)^2 \sum_{t=1}^{T} \eta^{(t)} + \frac{1}{2}\sum_{t=1}^{T} (\mathbf{g}^{(t)})^\top (\mathbf{H}^{(t)})^\dagger \mathbf{g}^{(t)}. \qquad (29)
$$

and

$$
\sum_{t=1}^{T} \mathbf{g}_t^\top (\widehat{\mathbf{H}}^{(t)})^\dagger \mathbf{g}_t \leq m - 1 + \frac{m(m-1)}{2} \ln \Big(1 + \frac{2\sum_{t=1}^{T} \|\mathbf{g}^{(t)}\|_2^2}{m(m-1)\sigma^*}\Big) \qquad (30)
$$

Then we can derive Theorem 6.

Proof For any $t \leq T'$, RFD-SON with $\alpha^{(0)} = 0$ satisfies $\mathrm{rank}(\widehat{\mathbf{B}}^{(t)}) \leq m - 1$. Hence we have

$$
\begin{aligned}
\mathbf{H}^{(t)} &= (\mathbf{B}^{(t)})^\top \mathbf{B}^{(t)} \\
&= \sum_{s=1}^{t} \frac{1}{\eta_t + \mu_t} (\mathbf{g}^{(t)})(\mathbf{g}^{(t)})^\top \\
&\succeq \frac{1}{\eta_1 + \mu'} \sum_{s=1}^{t} (\mathbf{g}^{(t)})(\mathbf{g}^{(t)})^\top \\
&= \frac{1}{\eta_1 + \mu'} \widehat{\mathbf{H}}^{(t)}. \qquad (31)
\end{aligned}
$$

Then we can bound the regret for the first T' iterations

$$\begin{aligned}
R_{1:T'}(\mathbf{w}) &\leq 2(CL)^2 \sum_{t=1}^{T'} \eta^{(t)} + \frac{1}{2} \sum_{t=1}^{T'} (\mathbf{g}^{(t)})^\top (\mathbf{H}^{(t)})^\dagger \mathbf{g}^{(t)} \\
&\leq 2(CL)^2 \sum_{t=1}^{T'} \eta^{(t)} + \frac{1}{2(\eta_1 + \mu')} \sum_{t=1}^{T'} (\mathbf{g}^{(t)})^\top (\widehat{\mathbf{H}}^{(t)})^\dagger \mathbf{g}^{(t)} \\
&\leq 2(CL)^2 \sum_{t=1}^{T'} \eta^{(t)} + \frac{m-1}{2(\eta_1 + \mu')} + \frac{m(m-1)}{2(\eta_1 + \mu')} \ln \Big(1 + \frac{2\sum_{t=1}^{T'} \|\mathbf{g}^{(t)}\|_2^2}{m(m-1)\sigma^*}\Big).
\end{aligned}$$

The first inequality is based on the inequality (29). The second inequality comes from (31) that is $\mathbf{H}^{(t)} \succeq \frac{1}{\eta_1 + \mu'} \widehat{\mathbf{H}}^{(t)}$. And the last one is due to the result (30). ■

Appendix I. The Proof of Theorem 7

Proof Since $\mathbf{H}^{(t)} \succ \mathbf{0}$ for $t \geq T'$, by similar proof of Theorem 5, we have

$$2R_{T'+1:T}(\mathbf{w}) \leq \alpha_0 \|\mathbf{w}^{(T')}\|^2_{\mathbf{H}^{(T')}} + R'_G + R'_D, \tag{32}$$

where

$$\begin{aligned}
R'_G &= \sum_{t=T'+1}^{T} (\mathbf{g}^{(t)})^\top (\mathbf{H}^{(t)})^{-1} \mathbf{g}^{(t)} \\
&\leq \frac{1}{\mu + \eta_T} \left[m \ln \Big(\frac{\mathrm{tr}\big((\mathbf{B}^{(T)})^\top \mathbf{B}^{(T)}\big)}{m\alpha^{(T')}} + \frac{\alpha^{(T)}}{\alpha'_0} \Big) + (d-m) \ln \frac{\alpha^{(T)}}{\alpha'_0} + \frac{m}{2} \sum_{t=T'+1}^{T} \frac{(\sigma_m^{(t)})^2}{\alpha^{(t)}} \right],
\end{aligned} \tag{33}$$

$$\begin{aligned}
R'_D &= \sum_{t=T'+1}^{T} (\mathbf{w}^{(t)} - \mathbf{w})^\top [\mathbf{H}^{(t)} - \mathbf{H}^{(t-1)} - \mu_t \mathbf{g}^{(t)} (\mathbf{g}^{(t)})^\top] (\mathbf{w}^{(t)} - \mathbf{w}) \\
&\leq 4(CL)^2 \sum_{t=T'+1}^{T} \eta_t + 2C^2 \sum_{t=T'+1}^{T} (\sigma_m^{(t-1)})^2.
\end{aligned} \tag{34}$$

Combining Theorem 6, (32), (33) and (34), we have the final result of (14). ■

References

Dimitris Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671–687, 2003.

Dimitris Achlioptas and Frank McSherry. Fast computation of low-rank matrix approximations. *Journal of the ACM*, 54(2):article 9, 2007.

Dimitris Achlioptas, Zohar S. Karnin, and Edo Liberty. Near-optimal entrywise sampling for data matrices. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. In *ACM Symposium on Theory of Computing (STOC)*, 2006.

Nir Ailon and Edo Liberty. Fast dimension reduction using Rademacher series on dual BCH codes. *Discrete & Computational Geometry*, 42(4):615–630, 2009.

Nir Ailon and Edo Liberty. An almost optimal unrestricted fast Johnson-Lindenstrauss transform. *ACM Transactions on Algorithms*, 9(3):article 21, 2013.

Sanjeev Arora, Elad Hazan, and Satyen Kale. A fast random sampling algorithm for sparsifying matrices. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. 2006.

Jimmy Ba, Roger Grosse, and James Martens. Distributed second-order optimization using Kronecker-factored approximations. In *International Conference on Learning Representations (ICLR)*, 2017.

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

Matthew Brand. Incremental singular value decomposition of uncertain data with missing values. In *European Conference on Computer Vision (ECCV)*, 2002.

Kenneth L. Clarkson and David P. Woodruff. Low rank approximation and regression in input sparsity time. In *ACM Symposium on Theory of Computing (STOC)*, 2013.

Amey Desai, Mina Ghashami, and Jeff M. Phillips. Improved practical matrix sketching with guarantees. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1678–1690, 2016.

Petros Drineas and Michael W. Mahoney. On the Nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

Petros Drineas and Anastasios Zouzias. A note on element-wise matrix sparsification via a matrix-valued Bernstein inequality. *Information Processing Letters*, 111(8):385–389, 2011.

Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006a.

Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, 36(1):158–183, 2006b.

Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, 2008.

Petros Drineas, Malik Magdon-Ismail, Michael W. Mahoney, and David P. Woodruff. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13: 3475–3506, 2012.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

Murat A. Erdogdu and Andrea Montanari. Convergence rates of sub-sampled Newton methods. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

Alan Frieze, Ravi Kannan, and Santosh Vempala. Fast Monte-Carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51(6):1025–1041, 2004.

Mina Ghashami, Edo Liberty, Jeff M. Phillips, and David P. Woodruff. Frequent directions: Simple and deterministic matrix sketching. *SIAM Journal on Computing*, 45(5):1762–1792, 2016.

Roger Grosse and James Martens. A Kronecker-factored approximate fisher matrix for convolution layers. In *International Conference on Machine Learning (ICML)*, 2016.

Ming Gu and Stanley C. Eisenstat. A stable and fast algorithm for updating the singular value decomposition. *Technical Report YALEU/DCS/RR-966*, 1993.

Isabelle Guyon, Steve R. Gunn, Asa Ben-Hur, and Gideon Dror. Result analysis of the NIPS 2003 feature selection challenge. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.

Isabelle Guyon, Constantin F. Aliferis, Gregory F. Cooper, André Elisseeff, Jean-Philippe Pellet, Peter Spirtes, and Alexander R. Statnikov. Design and analysis of the causation and prediction challenge. In *Causation and Prediction Challenge at WCCI*, 2008.

Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53 (2):217–288, 2011.

Peter M. Hall, David Marshall, and Ralph R. Martin. Incremental eigenanalysis for classification. In *British Machine Vision Conference (BMVC)*, 1998.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Elad Hazan and Sanjeev Arora. *Efficient algorithms for online convex optimization and their applications*. Princeton University, 2006.

Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis, Vol. 2*. Cambridge University Press, 1991.

Zengfeng Huang. Near optimal frequent directions for sketching dense and sparse matrices. In *International Conference on Machine Learning (ICML)*, 2018.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *ACM Symposium on Theory of Computing (STOC)*, 1998.

William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

Daniel M. Kane and Jelani Nelson. Sparser Johnson-Lindenstrauss transforms. *Journal of the ACM*, 61(1):article 4, 2014.

Avraham Levey and Michael Lindenbaum. Sequential Karhunen-Loeve basis extraction and its application to images. *IEEE Transactions on Image Processing*, 9(8):1371–1374, 2000.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

Edo Liberty. Simple and deterministic matrix sketching. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2013.

Haipeng Luo, Alekh Agarwal, Nicolò Cesa-Bianchi, and John Langford. Efficient second order online learning by sketching. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

Michael W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011.

James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *International Conference on Machine Learning (ICML)*, 2015.

Andrew McCallum. Sraa: Simulated/real/aviation/auto usenet data. URL `https://people.cs.umass.edu/~mccallum/data.html`.

Chris Mesterharm and Michael J Pazzani. Active learning using on-line algorithms. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2011.

Jayadev Misra and David Gries. Finding repeated elements. *Science of Computer Programming*, 2 (2):143–152, 1982.

Youssef Mroueh, Etienne Marcheret, and Vaibhava Goel. Co-occurring directions sketching for approximate matrix multiply. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

Jelani Nelson and Huy L. Nguyên. Osnap: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *Symposium on Foundations of Computer Science (FOCS)*, 2013.

Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15(3):267–273, 1982.

Erkki Oja and Juha Karhunen. On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of Mathematical Analysis and Applications*, 106(1): 69–84, 1985.

Dimitris Papailiopoulos, Anastasios Kyrillidis, and Christos Boutsidis. Provable deterministic leverage score sampling. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2014.

Mert Pilanci and Martin J. Wainwright. Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence. *SIAM Journal on Optimization*, 27(1):205–245, 2017.

John C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods: Support Vector Learning*, pages 185–208. Cambridge, MA: MIT Press, 1999.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Farbod Roosta-Khorasani and Michael W. Mahoney. Sub-sampled Newton methods I: globally convergent algorithms. *CoRR*, abs/1601.04737, 2016a.

Farbod Roosta-Khorasani and Michael W. Mahoney. Sub-sampled Newton methods II: local convergence rates. *CoRR*, abs/1601.04738, 2016b.

David A. Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1-3):125–141, 2008.

Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In *Symposium on Foundations of Computer Science (FOCS)*, 2006.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

Joel A. Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8(1-2):1–230, 2015.

Shusen Wang, Luo Luo, and Zhihua Zhang. SPSD matrix approximation vis column selection: Theories, algorithms, and extensions. *Journal of Machine Learning Research*, 17:(49)1–49, 2016.

David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.

Peng Xu, Jiyan Yang, Farbod Roosta-Khorasani, Christopher Ré, and Michael W. Mahoney. Sub-sampled Newton methods with non-uniform sampling. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

Haishan Ye, Luo Luo, and Zhihua Zhang. Approximate Newton methods and their local convergence. In *International Conference on Machine Learning (ICML)*, 2017.

Qiaomin Ye, Luo Luo, and Zhihua Zhang. Frequent direction algorithms for approximate matrix multiplication with applications in CCA. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

Zhihua Zhang. The matrix ridge approximation: algorithms and applications. *Machine Learning*, 97 (3):227–258, 2014.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning (ICML)*, 2003.

Journal of Machine Learning Research 20 (2019) 1-36 Submitted 2/18; Revised 12/18; Published 2/19

Boosted Kernel Ridge Regression: Optimal Learning Rates and Early Stopping

Shao-Bo Lin sblin1983@gmail.com
Department of Mathematics
Wenzhou University
Wenzhou, China

Yunwen Lei leiyw@sustc.edu.cn
Department of Computer Science and Engineering
Southern University of Science and Technology
Shenzhen, China

Ding-Xuan Zhou mazhou@cityu.edu.hk
School of Data Science and Department of Mathematics
City University of Hong Kong
Kowloon, Hong Kong, China

Editor: Arthur Gretton

Abstract

In this paper, we introduce a learning algorithm, boosted kernel ridge regression (BKRR), that combines L_2-Boosting with the kernel ridge regression (KRR). We analyze the learning performance of this algorithm in the framework of learning theory. We show that BKRR provides a new bias-variance trade-off via tuning the number of boosting iterations, which is different from KRR via adjusting the regularization parameter. A (semi-)exponential bias-variance trade-off is derived for BKRR, exhibiting a stable relationship between the generalization error and the number of iterations. Furthermore, an adaptive stopping rule is proposed, with which BKRR achieves the optimal learning rate without saturation.

Keywords: learning theory, kernel ridge regression, boosting, integral operator

1. Introduction

Supervised learning aims at learning function relationships between input and output variables, based on input-output pair samples. Kernel ridge regression (KRR) is a classical and standard approach for supervised learning due to its easy implementation and theoretical optimality (Evgeniou et al., 2000; Caponnetto and De Vito, 2007; Steinwart et al., 2009; Lin et al., 2017) and thus has triggered enormous research activities in the statistical and machine learning communities (Bauer et al., 2007; Caponnetto and De Vito, 2007; Cucker and Zhou, 2007; Smale and Zhou, 2007; Steinwart et al., 2009; Lin et al., 2017). However, KRR suffers from a so-called saturation phenomenon (Gerfo et al., 2008) meaning that its learning rate cannot be improved once the target (regression) function goes beyond a certain level of regularity. Furthermore, there lacks efficient parameter-selection strategies for KRR to realize its theoretically optimal learning performance. Many users' spirit is dampened by these drawbacks and then turns to other learning algorithms such as the kernel-based gradient descent (Yao et al., 2007), kernel-based conjugate gradient descent (Blanchard and Krämer, 2016) and kernel-based partial least squares (Lin and Zhou, 2018b).

Boosting, originally proposed by Schapire (1990); Freund (1995), is devoted to producing a strong composite learner from a given class of weak learners. Some boosting algorithms can be interpreted from a viewpoint of statistical gradient descent to solve optimization problems with different loss functions (Friedman et al., 2000; Friedman, 2001). In this way, a special boosting algorithm, L_2-Boosting, was interpreted as an iterative least squares fitting of residuals (Friedman, 2001; Bühlmann and Yu, 2003). L_2-Boosting was utilized by Park et al. (2009) to improve the learning performance of Nadaya-Watson kernel estimators by overcoming saturation and was also proved in Bühlmann and Yu (2003) to be almost over-fitting resistant by exhibiting its exponential bias-variance trade-off for linear regression, reducing the difficulty of model selection.

The aim of this paper is to combine L_2-Boosting with KRR to overcome the saturation and reduce the difficulty of model selection of KRR. Let $(\mathcal{H}_K, \|\cdot\|_K)$ be the reproducing kernel Hilbert space (RKHS) induced by a Mercer kernel K on a metric (input) space $\mathcal{X}$ and $D = \{(x_i, y_i)\}_{i=1}^N \subset \mathcal{X} \times \mathcal{Y}$ be a sample with $\mathcal{Y} \subseteq \mathbb{R}$ the output space. KRR is defined by

$$f_{D,\lambda}^{(1)} := \arg\min_{f\in\mathcal{H}_K} \left\{ \frac{1}{|D|} \sum_{(x,y)\in D} (f(x)-y)^2 + \lambda \|f\|_K^2 \right\}, \tag{1}$$

where $\lambda > 0$ is a regularization parameter and $|D| = N$ is the cardinality of D. Implementing KRR needs the inverse of the matrix $\mathbb{K}+\lambda|D|\mathbb{I}$ and thus requires $\mathcal{O}(|D|^3)$ computational complexity in time for a fixed λ, where $\mathbb{K}$ is the kernel matrix $(K(x_i,x_j))_{i,j=1}^{|D|}$ and $\mathbb{I}$ is the $|D| \times |D|$ identity matrix. The performance of KRR is sensitive to regularization parameters, which need be carefully tuned to achieve satisfactory learning rates close to the optimal one.

Boosted KRR (BKRR) studied in this paper iteratively defines an estimator $f_{D,\lambda}^{(k)}$ by running KRR on the data set $\{(x_i, y_i - f_{D,\lambda}^{(k-1)}(x_i))\}_{(x_i,y_i)\in D}$, whose outputs are residuals of the previous iteration $f_{D,\lambda}^{(k-1)}$. To be detailed, the estimator at the k-th boosting iteration is given by

$$f_{D,\lambda}^{(k)} := f_{D,\lambda}^{(k-1)} + f_{D,\lambda}^{(k)\diamond}, \qquad k > 1, \tag{2}$$

where

$$f_{D,\lambda}^{(k)\diamond} := \arg\min_{f\in\mathcal{H}_K} \left\{ \frac{1}{|D|} \sum_{(x,y)\in D} \left\{ f(x) - [y - f_{D,\lambda}^{(k-1)}(x)] \right\}^2 + \lambda \|f\|_K^2 \right\}, \qquad k > 1. \tag{3}$$

An advantage of BKRR over KRR is the flexibility of choosing some relatively large regularization parameter λ. Satisfactory learning rates are achieved by the boosting iterations. As the inputs $\{x_i\}_{i=1}^N$ of KRR and $f_{D,\lambda}^{(k)\diamond}$ are the same and the inverse of $\mathbb{K}+\lambda|D|\mathbb{I}$ has already been derived in the first step, it only requires $\mathcal{O}(|D|^2)$ computational complexity to solve (3). Then, BKRR with k iterations only needs $\mathcal{O}(|D|^3 + k|D|^2)$ computational complexity, which does not bring additional computational burden over KRR.

As pointed out by Friedman (2001), boosting leads to over-fitting if the weak learners are already over-fitting. So the regularization parameter of (1) must be relatively large,

©2019 Shao-Bo Lin, Yunwen Lei and Ding-Xuan Zhou.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-063.html.

implying under-fitting of the original KRR. BKRR then tunes k to reduce the bias, which enlarges the variance, reflecting the bias-variance trade-off. Our first main result is to deduce a (semi-)exponential bias-variance trade-off of BKRR: the bias of BKRR decreases exponentially with respect to k, while the variance increases by an exponentially diminishing amount as k gets large for the in-sample estimate and by an algebraic diminishing amount with respect to k for the out-sample estimate. The (semi-)exponential bias-variance trade-off shows that BKRR can reach its optimal learning performance with a relatively small number of iterations. It also exhibits that moderately large k does not degrade the learning performance of BKRR very much, making the model selection much easier than that of other iteration-based learning algorithms such as kernel-based gradient descent, kernel-based conjugate gradient descent and kernel-based partial least squares for which only polynomial bias-variance trade-off is obtained (Yao et al., 2007; Blanchard and Krämer, 2016; Lin and Zhou, 2018b).

The exponential bias-variance trade-off does not mean that over-fitting never happens for BKRR, and it requires a stopping rule of high quality. Our second main result is to propose an adaptive stopping rule based on an empirical effective dimension (Lu et al., 2018; Mücke, 2018), with which we prove that BKRR achieves the optimal learning rate without saturation. In a nutshell, our analysis shows that BKRR reduces the difficulty of model selection of KRR in terms of providing a stable relationship between the learning performance and model selection. Furthermore, BKRR with an adaptive stopping rule can improve the learning performance of KRR via overcoming the saturation. The main tools of our analysis are detailed spectral analysis of BKRR, the recently developed integral operator approach (Lin et al., 2017; Guo et al., 2017) and a tight bound for the number of iterations of the stopping rule.

2. Main Results

Our analysis is conducted in a standard learning theory framework for regression (Cucker and Zhou, 2007), in which the samples in D are independently drawn according to ρ, a Borel probability measure on $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$. The purpose of regression is to derive an estimator based on D to approximate the regression function $f_\rho(x) := \int_{\mathcal{Y}} y d\rho(y|x)$ with $\rho(\cdot|x)$ being the conditional distribution of ρ induced at $x \in \mathcal{X}$. Let ρ_X be the marginal distribution of ρ on $\mathcal{X}$ and $L^2_{\rho_X}$ be the space of ρ_X square integrable functions endowed with norm $\|\cdot\|_\rho$. Throughout this paper, we assume $\mathcal{X}$ is compact, which implies $\kappa := \sqrt{\sup_{x\in\mathcal{X}} K(x,x)} < \infty$.

2.1. Optimal Learning Rates

Before presenting the exponential bias-variance trade-off and stopping rule, we derive optimal learning rates for BKRR with a priori knowledge involving λ and k to show the necessity of our studies. For this purpose, some assumptions on the decay of the outputs, regularity of the regression function and capacity of $\mathcal{H}_K$ are needed. To be detailed, we assume $\int_{\mathcal{Y}} y^2 d\rho < \infty$ and the following output decay condition

$$\int_{\mathcal{Y}} \left(e^{\frac{|y-f_\rho(x)|}{M}} - \frac{|y-f_\rho(x)|}{M} - 1 \right) d\rho(y|x) \le \frac{\gamma^2}{2M^2}, \qquad \forall x \in \mathcal{X}, \tag{4}$$

where M and γ are positive constants. Condition (4) is satisfied if the noise is uniformly bounded, Gaussian or sub-Gaussian (Caponnetto and De Vito, 2007). In particular, if $|y| \le \mathcal{B}$ almost surely for some $\mathcal{B} > 0$, then (4) holds with $\gamma/2 = M = \mathcal{B}$.

Let $L_K : \mathcal{H}_K \to \mathcal{H}_K$ (or $L^2_{\rho_X} \to L^2_{\rho_X}$) be the integral operator defined by

$$L_K f = \int_{\mathcal{X}} K_x f(x) d\rho_X(x)$$

with $K_x := K(\cdot, x)$. We assume that f_ρ satisfies the standard regularity condition

$$f_\rho = L_K^r h_\rho, \quad \text{for some } r > 0 \text{ and } h_\rho \in L^2_{\rho_X}, \tag{5}$$

where L_K^r is the r-th power of $L_K : L^2_{\rho_X} \to L^2_{\rho_X}$. The regularity condition (5) describes the regularity of f_ρ and has been adopted in a large literature to quantify learning rates for some algorithms (Smale and Zhou, 2007; Bauer et al., 2007; Caponnetto and De Vito, 2007; Caponnetto and Yao, 2010; Shi et al., 2011; Blanchard and Krämer, 2016; Guo et al., 2017; Lin et al., 2017; Lin and Zhou, 2018b; Ying and Zhou, 2017) .

We also introduce the effective dimension $\mathcal{N}(\lambda) := \mathrm{Tr}[(L_K + \lambda I)^{-1} L_K]$ to measure the capacity of $\mathcal{H}_K$. Here $\mathrm{Tr}(A)$ denotes the trace of an operator A with a detailed definition to be given in Appendix B. In particular, we assume with a parameter $0 < s \le 1$ and a constant $C_0 > 0$ that

$$\mathcal{N}(\lambda) \le C_0 \lambda^{-s}, \qquad \forall \lambda > 0. \tag{6}$$

Condition (6) with $s = 1$ is always satisfied by taking $C_0 = \mathrm{Tr}(L_K) \le \kappa^2$. For $0 < s < 1$, it was shown in Guo et al. (2017, Page 7) that (6) is slightly more general than the eigenvalue decaying assumption in the literature (Caponnetto and De Vito, 2007) and has been extensively employed to derive fast learning rates for some algorithms (Caponnetto and De Vito, 2007; Blanchard and Krämer, 2016; Guo et al., 2017; Lin et al., 2017; Lin and Zhou, 2018a,b). With these assumptions, we present in the following theorem the optimal learning rates for BKRR.

Theorem 1 *Let $0 < \delta < 1$, $k \le \sqrt{|D|}$ and $\lambda = \left(k^2/|D|\right)^{1/(2\min\{k,r\}+s)}$ with $r > 3/2$ and $0 < s \le 1$. Under assumptions (4), (5) and (6), with confidence $1-\delta$ there holds*

$$\left\| f_{D,\lambda}^{(k)} - f_\rho \right\|_\rho \le \tilde{C} (k^2/|D|)^{\frac{\min\{k,r\}}{2\min\{k,r\}+s}} \log^3 \frac{8}{\delta}, \tag{7}$$

where $\tilde{C}$ is a constant independent of $|D|$, k or δ.

For an iteration number $k \ge r$ independent of $|D|$, Theorem 1 presents an optimal learning rate for BKRR since (7) achieves the minimax lower bound established by Caponnetto and De Vito (2007). However, as shown in (7), too large k may worsen the learning rate. An extreme case is $k = \sqrt{|D|}$ which leads to a constant upper bound. In a nutshell, small k (smaller than r) suffers from the saturation but too large k leads to slow learning rates. Thus, it is important to derive an adaptive stopping rule on selecting k. Furthermore, Theorem 1 implies that as long as $k \ge r$, the boosting iteration with large k does not have any benefits in the learning process, which contradicts the boosting theory developed in Friedman (2001); Bühlmann and Yu (2003) at first glance. Thus, it requires a more

delicate analysis to explore the power of iterations, especially when $k \geq r$. Based on this observation, we conduct a detailed bias-variance analysis in the following subsection and find a so-called exponential bias-variance trade-off of BKRR.

Remark 2 *In Theorem 1 as well as Theorem 4 below, we require $r > \frac{3}{2}$. We believe that similar optimal learning rates can be derived for $r \geq 1/2$ by using the technique in Caponnetto and Yao (2010); Guo et al. (2017). Since one of the main advantages of BKRR is to conquer the saturation, we focus on relatively large r in this paper. Throughout this paper, we assume $r \geq 1/2$, implying $f_\rho \in \mathcal{H}_K$. For $0 < r < \frac{1}{2}$, i.e. $f_\rho \notin \mathcal{H}_K$, like KRR (Caponnetto and Yao, 2010; Chang et al., 2017), BKRR usually requires additional unlabeled data to achieve the optimal learning rates.*

2.2. Exponential Bias-variance Trade-off

One of the most important advantages of L_2 boosting in linear regression is its almost over-fitting resistance (Bühlmann and Yu, 2003) meaning that for the in-sample error estimate, the bias decreases exponentially fast and the variance increases with exponentially diminishing terms as k increases. In this subsection, we will show that BKRR also possesses this property and show the power of boosting iteration for $k \geq r$.

Let $S_D : \mathcal{H}_K \to \mathbb{R}^{|D|}$ be the sampling operator (Smale and Zhou, 2004) defined by

$$S_D f := (f(x))_{(x,y)\in D}.$$

Its scaled adjoint $S_D^T : \mathbb{R}^{|D|} \to \mathcal{H}_K$ (or $\mathbb{R}^{|D|} \to L^2_{\rho_X}$) is given by

$$S_D^T \mathbf{c} := \frac{1}{|D|}\sum_{i=1}^{|D|} c_i K_{x_i}, \qquad \mathbf{c} := (c_1, c_2, \ldots, c_{|D|})^T \in \mathbb{R}^{|D|}.$$

Define a discretization of the integral operator L_K by

$$L_{K,D} f := S_D^T S_D f = \frac{1}{|D|}\sum_{(x,y)\in D} f(x) K_x.$$

Our bias-variance trade-off will be stated in terms of some quantities involving the difference between the compact and positive operators L_K and $L_{K,D}$ given by

$$\mathcal{Q}_{D,\lambda} := \|(L_{K,D} + \lambda I)^{-1/2}(L_K + \lambda I)^{1/2}\|, \qquad \mathcal{R}_D := \|L_{K,D} - L_K\|_{HS} \tag{8}$$

and the difference between $S_D^T y_D$ and $L_{K,D} f_\rho$ given by

$$\mathcal{P}_{D,\lambda} := \left\|(L_K + \lambda I)^{-1/2}(L_{K,D} f_\rho - S_D^T y_D)\right\|_K, \tag{9}$$

where $\|A\|_{HS}$ denotes the Hilbert-Schmidt norm of a Hilbert-Schmidt operator A, $\|\cdot\|$ denotes the operator norm, $y_D := (y_1, \ldots, y_{|D|})^T$ and I is the identity operator. We refer the readers to Appendix B for some basic definitions of linear operators. Let $\{(\sigma_i^{\mathbf{x}}, \phi_i^{\mathbf{x}})\}$ be a set of normalized eigenpairs of $L_{K,D}$ with the eigenfunctions $\{\phi_i^{\mathbf{x}}\}_i$ forming an orthonormal basis of $\mathcal{H}_K$ and the eigenvalue sequence $\{\sigma_i^{\mathbf{x}}\}$ non-increasing. Since $L_{K,D}$ is a positive

operator of rank at most $|D|$, we have $\sigma_k^{\mathbf{x}} = 0$ for $k \geq |D|+1$. Denote by $\sigma_{min}^{\mathbf{x}}$ the minimum positive eigenvalue of $L_{K,D}$. Denote by $\|f\|_D^2 := \frac{1}{|D|}\sum_{i=1}^{|D|} f^2(x_i)$. The following theorem shows a trade-off between the bias and variance of BKRR under the $\|\cdot\|_D$ semi-norm.

Theorem 3 *Let $k \geq 1$. Then, under condition (5) with $r \geq 1/2$, there holds*

$$\begin{aligned}\left\|f_{D,\lambda}^{(k)} - f_\rho\right\|_D &\leq \frac{(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho}{\sqrt{k}}\lambda^{1/2}\mathcal{R}_D + \mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda} \\ + \quad \mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}&\sum_{j=1}^{k-1}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{j-1/2} + 4(\mathcal{Q}_{D,\lambda}^2+1)\lambda^k\|h_\rho\|_\rho \begin{cases} (\sigma_{min}^{\mathbf{x}}+\lambda)^{r-k}, & \text{if } k > r, \\ \kappa^{2r-2k}+\lambda^{r-k}, & \text{if } k \leq r. \end{cases}\end{aligned} \tag{10}$$

The dominant terms on the right-hand side of (10) are the third and fourth terms and we call them the variance and bias for BKRR, respectively. For $0 < \lambda \leq 1$, it follows from Theorem 3 that the bias of BKRR decreases exponentially fast while the variance increases with exponentially diminishing terms as k increases, showing the exponential bias-variance trade-off. Bound (10) also exhibits a sudden change of the rate of bias decay when k is around the regularity level r of f_ρ. Its rate drops from λ^k to $\lambda^k(\sigma_{min}^{\mathbf{x}}+\lambda)^{r-k} = \lambda^r\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r}$. Since Proposition 17 below shows that the variance of KRR can be bounded by $\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}$, the additional term in the variance of BKRR, $\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}\sum_{j=1}^{k-1}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{j-1/2}$, implies that BKRR degrades the learning performance of KRR if their regularization parameters are identical. This coincides with the consensus that boosting is not worthwhile if the learner is already complex. Hence, in BKRR, a large λ should be chosen to guarantee under-fitting of the original KRR, i.e., large bias and small variance. The trade-off can be achieved via an appropriately tuned k such that the bias and variance are close.

Theorem 3 presents an error estimate for BKRR in terms of the empirical semi-norm $\|\cdot\|_D$. In the following theorem, we present error analysis for BKRR in terms of the $\|\cdot\|_\rho$ norm.

Theorem 4 *Let $k \geq 1$. Assume condition (5) with $r > 3/2$. Then there holds*

$$\begin{aligned}\|f_{D,\lambda}^{(k)} - f_\rho\|_\rho &\leq 2\mathcal{Q}_{D,\lambda}(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\lambda^{1/2}\mathcal{R}_D + 2k\mathcal{Q}_{D,\lambda}^2\mathcal{P}_{D,\lambda} \\ + \quad \mathcal{Q}_{D,\lambda}\lambda^k\|h_\rho\|_\rho &\begin{cases} \lambda^{r-k} + \kappa^{2r-2k-1}(\kappa+\lambda^{\frac{1}{2}}), & \text{if } k \leq r, \\ 2(\sigma_{min}^{\mathbf{x}}+\lambda)^{r-k}, & \text{if } k > r. \end{cases}\end{aligned} \tag{11}$$

Different from the exponential bias-variance trade-off of the error estimate in terms of the $\|\cdot\|_D$ semi-norm shown in Theorem 3, there exhibits a semi-exponential bias-variance trade-off for the error estimate in terms of the $\|\cdot\|_\rho$ norm. To be detailed, the bias decreases exponentially, while the variance increases polynomially as k increases. Based on Theorem 3 and Theorem 4, we can derive the following corollary.

Corollary 5 *Under condition (5) with $r \geq 1/2$, $\|L_{K,D} f_{D,\lambda}^{(k)} - S_D^T y_D\|_K$ decreases with respect to k. Moreover,*

$$\lim_{k\to\infty}\|L_{K,D} f_{D,\lambda}^{(k)} - S_D^T y_D\|_K \leq \lambda^{\frac{1}{2}}\mathcal{P}_{D,\lambda}\mathcal{Q}_{D,\lambda} \tag{12}$$

and

$$\lim_{k\to\infty}\left\|f_{D,\lambda}^{(k)}-f_\rho\right\|_D\le\left(1+\frac{(\sigma_{min}^{\mathbf{x}}+\lambda)^{1/2}\lambda^{1/2}}{\sigma_{min}^{\mathbf{x}}}\right)\mathcal{P}_{D,\lambda}\mathcal{Q}_{D,\lambda}. \tag{13}$$

Corollary 5 exhibits an almost over-fitting resistance phenomenon of BKRR (neglecting the constant) for some kernels, since the sample error of KRR is bounded by $\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}$ (see Proposition 17 below). The behavior of the boosting iteration in (13) is different from that of the kernel-based (conjugate) gradient descent (Blanchard and Krämer, 2016; Lin and Zhou, 2018a), where the generalization error becomes ∞ for an arbitrary kernel, as the iteration number tends to infinity.

2.3. Adaptive Stopping Rule

We present in this subsection an adaptive stopping rule for BKRR to guarantee its optimal learning rates. To introduce the stopping rule, a user-friendly measurement of the capacity, empirical effective dimension (Lu et al., 2018; Mücke, 2018), defined by

$$\mathcal{N}_D(\lambda)=\mathrm{Tr}[(L_{K,D}+\lambda I)^{-1}L_{K,D}]=\mathrm{Tr}[(\lambda|D|I+\mathbb{K})^{-1}\mathbb{K}] \tag{14}$$

is needed. Denote

$$\mathcal{W}_{D,\lambda}=\frac{16\sqrt{2}(\kappa^2+\kappa+1)(\kappa M+\gamma)}{\sqrt{|D|}}\left(\frac{(\sqrt{|D|\lambda}+9)\sqrt{\max\{\mathcal{N}_D(\lambda),1\}}}{|D|\lambda}+1\right)\frac{(\sqrt{|D|\lambda}+9)\sqrt{\max\{\mathcal{N}_D(\lambda),1\}}}{\sqrt{|D|\lambda}}. \tag{15}$$

If $\delta\in(0,1)$ is the parameter corresponding to the confidence level, the boosting iteration will stop at the first positive integer $\hat{k}:=\hat{k}_{D,\lambda,\delta,K}$ satisfying

$$\|L_{K,D}f_{D,\lambda}^{(\hat{k})}-S_D^Ty_D\|_K\le\lambda^{\frac{1}{2}}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta}. \tag{16}$$

Since

$$L_{K,D}f_{D,\lambda}^{(\hat{k})}-S_D^Ty_D=\frac{1}{|D|}\sum_{i=1}^{|D|}(f_{D,\lambda}^{(\hat{k})}(x_i)-y_i)K_{x_i},$$

we have

$$\|L_{K,D}f_{D,\lambda}^{(\hat{k})}-S_D^Ty_D\|_K^2=\frac{1}{|D|^2}(f_{D,\lambda}^{(\hat{k})}(\mathbf{x})-\mathbf{y})^T\mathbb{K}(f_{D,\lambda}^{(\hat{k})}(\mathbf{x})-\mathbf{y}),$$

where $f_{D,\lambda}^{(\hat{k})}(\mathbf{x})-\mathbf{y}$ is the vector $\left(f_{D,\lambda}^{(\hat{k})}(x_i)-y_i\right)_{i=1}^{|D|}$. This together with (14) shows that the stopping rule in (16) is implementable. Moreover, Lemma 23 in Appendix A shows that

$$\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}\le\frac{1}{2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \tag{17}$$

holds with confidence $1-\delta$. Then Corollary 5 verifies the existence of $\hat{k}$ with high probability since (16) is satisfied for sufficiently large $\hat{k}$ with high probability.

We are now in a position to present our second main result in the following theorem.

Theorem 6 *Let $\delta\in(0,1)$. Under conditions (4), (6) with $0<s\le1$ and condition (5) with $r\ge1/2$, if $\lambda=(c/|D|)^{1/(2r+s)}$ for some $c\ge1$, and $\hat{k}$ is the smallest positive integer satisfying (16), then with confidence at least $1-\delta$, there holds*

$$\|f_{D,\lambda}^{(\hat{k})}-f_\rho\|_\rho\le C|D|^{-\frac{r}{2r+s}}\log^{10}\frac{16}{\delta}, \tag{18}$$

where C is a constant independent of δ or $|D|$.

Theorem 6 shows that BKRR equipped with the stopping rule (16) achieves the same optimal learning rate without saturation, that is, the optimal learning rate holds for an arbitrary $r\ge1/2$ rather than $\frac{1}{2}\le r\le1$ shown by Caponnetto and De Vito (2007) and Lin et al. (2017) for KRR. Theorem 4 and Theorem 6 state that BKRR provides a novel semi-exponential bias-variance trade-off achieved by the boosting iteration, and the stopping rule (16) can realize its good performance. It follows from Corollary 5 and Theorem 6 that combining L_2 boosting with KRR reduces the difficulty of model selection (almost over-fitting resistance) and overcomes the saturation of KRR.

Remark 7 *Theoretically, a more delicate stopping rule for BKRR should be the first positive integer satisfying*

$$\|L_{K,D}f_{D,\lambda}^{(\hat{k})}-S_D^Ty_D\|_K\le2\lambda^{\frac{1}{2}}\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}.$$

Since the quantities $\mathcal{Q}_{D,\lambda}$ and $\mathcal{P}_{D,\lambda}$ cannot be implemented, we have to present a bound for them and thus get a confidence-dependent stopping rule (16). It should be noted that the constant in the definition of $\mathcal{W}_{D,\lambda}$ is not tight, which makes the algorithm stop much earlier than the optimal one. Due to the (semi-)exponential bias-variance trade-off presented in the previous subsection, a relatively large number of iterations does not degrade the generalization ability of BKRR very much. We thus multiply by a small factor to make the algorithm stop later. In a word, we implement the stopping rule (16) as

$$\begin{aligned}&\|L_{K,D}f_{D,\lambda}^{(\hat{k})}-S_D^Ty_D\|_K\\ \le\ &\frac{\theta\sqrt{\lambda}}{\sqrt{|D|}}\left(\frac{(\sqrt{|D|\lambda}+1)\sqrt{\max\{\mathcal{N}_D(\lambda),1\}}}{|D|\lambda}+1\right)\frac{(\sqrt{|D|\lambda}+1)\sqrt{\max\{\mathcal{N}_D(\lambda),1\}}}{\sqrt{|D|\lambda}}\end{aligned} \tag{19}$$

for some small θ such as $\theta=0.05$ (or other values).

Remark 8 *In Theorem 6, although k can be adaptively determined by (16), λ depends on r and s. It should be noted in Theorem 1 that $\lambda\sim|D|^{-1/(2r+s)}$ is the optimal regularization parameter for BKRR to achieve the optimal learning rate. The reason for this phenomenon is that we do not impose additional restrictions to the kernel K and the marginal distribution ρ_X other than (6). In particular, we use $\frac{\lambda}{\lambda+\sigma_{min}^{\mathbf{x}}}\le1$ directly in the proof. For some particular kernel and ρ_X, the minimum positive eigenvalue $\sigma_{min}^{\mathbf{x}}$ for the matrix $\mathbb{K}/|D|$ can be computed. Then, we can derive similar error estimates by Theorem 4. In this way, optimal learning rates of BKRR hold for large values of λ. It would be interesting to determine the kernel and marginal distribution ρ_X, with which BKRR achieves the similar optimal*

learning rates as Theorem 6 for large λ. The other reason that we do not focus on the selection of λ is that boosting theory usually requires large λ to keep the algorithm underfitting and using iteration to reduce the bias. Thus, we can select a relatively large λ in advance numerically. Our experimental results in Section 6 show that the generalization ability of BKRR is not very sensitive to λ provided it is larger than some value.

Remark 9 *The constant exhibited in (18) is a bit pessimistic, compared with the classical results in the literature (Caponnetto and De Vito, 2007; Caponnetto and Yao, 2010; Blanchard and Krämer, 2016; Lin et al., 2017; Guo et al., 2017). One of the reasons for this pessimistic estimate is that we do not impose any restriction on the relation between $|D|$ and δ. In particular, as shown in our proof, if we assume $2\log(16/\delta) \le \sqrt{|D|\lambda}$, i.e. $\delta \ge 16\exp\left\{-\frac{1}{2}c^{-1/(4r+2s)}|D|^{\frac{2r+s-1}{4r+2s}}\right\}$, then the exponent should be reduced from 10 to 6. Since the optimal constant is difficult to obtain, we only pursue the optimal learning rate in Theorem 6.*

3. Related Work

In this section, we discuss some related work in the literature and show the novelty of our results.

3.1. Boosting

A functional gradient descent viewpoint in statistics (Friedman et al., 2000; Friedman, 2001) reformulates boosting as a family of stage-wise optimization problems with different loss functions. Gradient boosting requires computing the negative gradient vector and line search in each boosting iteration. For L_2-Boosting, the gradient computation and line search can be unified in solving least squares fitting of residuals (Bühlmann and Yu, 2003). Thus, L_2-Boosting is essentially iterative least squares of residuals. An important advantage of boosting is its almost resistance to over-fitting (e.g. Friedman (2001) and its discussion papers), showing an easy way for model selection.

In Bühlmann and Yu (2003), an exponential bias-variance trade-off for linear regression was derived to illustrate the almost resistance to over-fitting for L_2-Boosting in a fixed design setting. In particular, Theorem 1 in Bühlmann and Yu (2003) shows that as the boosting iteration goes on, the bias decreases exponentially with a quantity depending on the minimum eigenvalue of the data matrix, while the variance increases with exponentially diminishing terms. Our Theorem 3 presents a similar result as Theorem 1 of Bühlmann and Yu (2003) for taking KRR as weak learners in L_2-Boosting, but highlights the importance of the regularity of the regression function by showing a sudden change of bias decaying. In Theorem 4, we also analyze the changes of bias and variance in a random design setting and show a semi-exponential bias-variance trade-off.

In Park et al. (2009), the learning performance of L_2-Boosting whose weak learners are Nadaraya-Watson kernel estimates was analyzed in the same framework as ours. It was shown in Theorem 2 and Theorem 3 of Park et al. (2009) that L_2-Boosting overcomes the saturation of Nadaraya-Watson kernel estimates (Györfi et al., 2002, Chapter 5). Differently, we utilize KRR as the weak learners instead of the kernel estimates, requiring totally different analysis. Furthermore, we present an adaptive stopping rule to select the number of boosting iterations, while Park et al. (2009) requires an a priori knowledge-dependent number of boosting iterations. In this paper, we are concerned with combining L_2-Boosting with KRR. It would be interesting to consider boosted versions of other algorithms such as the kernel-based gradient descent (Yao et al., 2007) and more generally the kernel-based spectral algorithms (Gerfo et al., 2008).

3.2. Iterated Tikhonov Regularization

From Lemma 12 below, we find that for a fixed k, BKRR can be regarded as a special spectral algorithm, the iterated Tikhonov regularization (Gerfo et al., 2008). In this framework, the learning rate of BKRR with a fixed k may be derived directly from general results for spectral algorithms (Bauer et al., 2007; Caponnetto and Yao, 2010; Guo et al., 2017,b).

Different from the iterated Tikhonov regularization, BKRR focuses on fixed but relatively large λ and parameterizes the number of iterations, though they possess the same spectral representation (see (27) below). It follows from Theorem 3 that BKRR has an eventually stable relationship between the generalization error and boosting iteration in the sense that the generalization error does not increase much with the boosting iteration after some k. Theorem 6 shows that BKRR with adaptive stopping rule (16) can overcome the saturation of KRR, just as iterated Tikhonov regularization does but with an a priori knowledge-dependent selected and fixed k.

In a recent paper (Wu, 2017), a bias correction algorithm was proposed for ridge regression and detailed analysis was provided for the changes of bias and variance. It was found that one-step iteration can reduce the bias without increasing the variance much. The analysis in Wu (2017) is carried out in a more general framework than that in this paper. It should be pointed out that with the same setting in this paper, the algorithm in (Wu, 2017) possesses the spectral representation (27) below with $k = 1$.

Iterated Tikhonov regularization is closely related to BKRR and widely used in the community of inverse problems. Analysis of iterated Tikhonov regularization in solving ill-posed inverse problems can be dated back to the 1970's (e.g. King and Chillingworth (1979)). The optimal convergence rates and parameter selection of iterated Tikhonov regularization are important topics in inverse problems (Engl, 1987; Jin and Hou, 1997; Hanke and Groetsch, 1998; Jin and Stals, 2012). In particular, our stopping rule (16) is motivated by the discrepancy principle provided in Hanke and Groetsch (1998).

3.3. Iteration-based Learning Schemes and Stopping Rules

Saturation is a well known design-flaw of KRR (Gerfo et al., 2008) and limits its usage. Due to this phenomenon, researchers turn to other iteration-based learning schemes such as the kernel-based gradient descent (Yao et al., 2007), kernel-based conjugate gradient descent (Blanchard and Krämer, 2016) and kernel-based partial least squares (Lin and Zhou, 2018b). The theoretical results in Blanchard and Krämer (2016); Lin and Zhou (2018a,b) showed that these strategies can reach the optimal learning rates without saturation.

As an iteration-based algorithm, the bias and variance of the kernel-based gradient descent algorithm were analyzed in Lin and Zhou (2018a) and a polynomial bias-variance trade-off was exhibited. In particular, as the iteration goes on, its bias decreases as $\mathcal{O}(k^{-r})$ and its variance increases as a polynomial of k. Similar results on polynomial

bias-variance trade-off of the kernel-based conjugate gradient descent and kernel-based partial least squares were derived in Blanchard and Krämer (2016) and Lin and Zhou (2018b), respectively. Different from these iteration-based algorithms, BKRR shows an exponential bias-variance trade-off, reducing the difficulty of model selection.

Stopping rules play an important role in iteration-based learning schemes. Learning rates of iteration-based algorithms were built in Bauer et al. (2007); Yao et al. (2007); Guo et al. (2017); Blanchard and Krämer (2016) upon prior knowledge-based stopping rules. In Caponnetto and Yao (2010); Lin and Zhou (2018b), cross-validation based stopping rules were presented for general spectral algorithms and kernel-based partial least squares. In Raskutti et al. (2014), an adaptive stopping rule was deduced for the kernel-based gradient descent algorithm under the regularity condition (5) with $r = 1/2$. More recently, another adaptive stopping rule based on a balancing principle for general spectral algorithms was presented in Lu et al. (2018). Different from these results, our stopping rule presented in (16) requires neither dividing the sample set (compared with the cross-validation), nor computing estimators with various λ (compared with the balancing principle). Compared with Raskutti et al. (2014), our results hold under condition (5) with all $r \geq 1/2$, i.e., we adaptively select r rather than fixing it to be $1/2$. At first glance, the dependence of the confidence level in (16) may make the stopping rule not so stable. However, the (semi-) exponential bias-variance trade-off of BKRR compensates this instability by showing that the learning performance remains stable for a large range of k. It would be interesting to derive a confidence-independent stopping rule for BKRR.

4. Operator Representations and Error Estimates

We analyze the learning performance of BKRR by using the integral operator approach (Smale and Zhou, 2007; Lin et al., 2017; Guo et al., 2017). The novelties of our proof are special operator representations of BKRR, special spectral properties of BKRR and a tight bound for $\hat{k}$ defined by (16). In Subsections 4.1 and 4.2, we provide detailed spectral analysis for BKRR, which is crucial for deriving the bias and variance estimates in Subsections 4.3 and 4.4. In Subsection 4.5, we provide a tight bound on the number of boosting iteration defined by (16) by utilizing the special spectral properties of BKRR.

4.1. Special Operator Representations of BKRR

Define the noise-free version of $f_{D,\lambda}^{(k)}$ by

$$f_{D,\lambda}^{(1,*)} := \arg\min_{f\in\mathcal{H}_K} \left\{ \frac{1}{|D|} \sum_{(x,y)\in D} (f(x) - f_\rho(x))^2 + \lambda \|f\|_K^2 \right\} \tag{20}$$

and

$$f_{D,\lambda}^{(k,*)} := f_{D,\lambda}^{(k-1,*)} + f_{D,\lambda}^{(k,*)\diamond}, \qquad k > 1, \tag{21}$$

where

$$f_{D,\lambda}^{(k,*)\diamond} := \arg\min_{f\in\mathcal{H}_K} \left\{ \frac{1}{|D|} \sum_{(x,y)\in D} [f(x) - (f_\rho(x) - f_{D,\lambda}^{(k-1,*)}(x))]^2 + \lambda \|f\|_K^2 \right\}, \qquad k > 1. \tag{22}$$

For KRR, the classical result in Smale and Zhou (2007) shows

$$f_{D,\lambda}^{(1)} = (L_{K,D} + \lambda I)^{-1} S_D^T y_D, \quad \text{and} \quad f_{D,\lambda}^{(1,*)} = (L_{K,D} + \lambda I)^{-1} L_{K,D} f_\rho. \tag{23}$$

Similar to (23), the following Lemma 10 presents operator representations for $f_{D,\lambda}^{(k)}$ and $f_{D,\lambda}^{(k,*)}$.

Lemma 10 *Let $k \geq 2$. We have*

$$f_{D,\lambda}^{(k)} = [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}] f_{D,\lambda}^{(k-1)} + f_{D,\lambda}^{(1)}, \tag{24}$$

$$L_{K,D} f_{D,\lambda}^{(k)} = S_D^T y_D - [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}]^k S_D^T y_D \tag{25}$$

and

$$f_{D,\lambda}^{(k,*)} = [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}] f_{D,\lambda}^{(k-1,*)} + f_{D,\lambda}^{(1,*)} = f_\rho - [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}]^k f_\rho. \tag{26}$$

Proof. Since $f_{D,\lambda}^{(k)\diamond}$ is the solution to KRR (1) with data $\{x_i, y_i - f_{D,\lambda}^{(k-1)}(x_i)\}_{(x_i,y_i)\in D}$, it follows from (2), (23) and the definition $L_{K,D} = S_D^T S_D$ that

$$\begin{aligned} f_{D,\lambda}^{(k)} &= f_{D,\lambda}^{(k-1)} + (L_{K,D} + \lambda I)^{-1} S_D^T (y_D - S_D f_{D,\lambda}^{(k-1)}) \\ &= [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}] f_{D,\lambda}^{(k-1)} + f_{D,\lambda}^{(1)}. \end{aligned}$$

This verifies (24). Combining this with (23) yields

$$\begin{aligned} & L_{K,D} f_{D,\lambda}^{(k)} - S_D^T y_D = L_{K,D}[I - (L_{K,D} + \lambda I)^{-1} L_{K,D}] f_{D,\lambda}^{(k-1)} + L_{K,D} f_{D,\lambda}^{(1)} - S_D^T y_D \\ = \; & [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}] L_{K,D} f_{D,\lambda}^{(k-1)} + [(L_{K,D} + \lambda I)^{-1} L_{K,D} - I] S_D^T y_D \\ = \; & [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}][L_{K,D} f_{D,\lambda}^{(k-1)} - S_D^T y_D]. \end{aligned}$$

Applying this relation iteratively and using (23) give

$$\begin{aligned} L_{K,D} f_{D,\lambda}^{(k)} - S_D^T y_D &= [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}]^{k-1} [L_{K,D} f_{D,\lambda}^{(1)} - S_D^T y_D] \\ &= -[I - (L_{K,D} + \lambda I)^{-1} L_{K,D}]^k S_D^T y_D. \end{aligned}$$

This proves (25). As for deriving (26), we have

$$f_{D,\lambda}^{(k,*)} = [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}] f_{D,\lambda}^{(k-1,*)} + (L_{K,D} + \lambda I)^{-1} L_{K,D} f_\rho.$$

It follows that

$$f_{D,\lambda}^{(k,*)} - f_\rho = [I - (L_{K,D} + \lambda I)^{-1} L_{K,D}][f_{D,\lambda}^{(k-1,*)} - f_\rho]$$

and by iterations,

$$\begin{aligned} f_{D,\lambda}^{(k,*)} - f_\rho &= [I - (L_{K,D}+\lambda I)^{-1}L_{K,D}]^{k-1}[f_{D,\lambda}^{(1,*)} - f_\rho] \\ &= -[I - (L_{K,D}+\lambda I)^{-1}L_{K,D}]^k f_\rho. \end{aligned}$$

This completes the proof of Lemma 10. ∎

From Lemma 10, we have

$$f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)} = [I - (L_{K,D}+\lambda I)^{-1}L_{K,D}](f_{D,\lambda}^{(k-1)} - f_{D,\lambda}^{(k-1,*)}) + f_{D,\lambda}^{(1)} - f_{D,\lambda}^{(1,*)},$$

from which the following expression is obtained by iterations.

Lemma 11 *For $k \in \mathbb{N}$, we have*

$$f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)} = \sum_{j=0}^{k-1}[I - (L_{K,D}+\lambda I)^{-1}L_{K,D}]^j[f_{D,\lambda}^{(1)} - f_{D,\lambda}^{(1,*)}].$$

4.2. Special Spectral Properties of BKRR

Our analysis depends on some spectral analysis of BKRR, viewed as a special class of spectral algorithms. It follows iteratively from the identity

$$f_{D,\lambda}^{(k)} = \lambda(L_{K,D}+\lambda I)^{-1}f_{D,\lambda}^{(k-1)} + (L_{K,D}+\lambda I)^{-1}S_D^T y_D$$

obtained from (24) by writing $L_{K,D} = L_{K,D} + \lambda I - \lambda I$.

Lemma 12 *For $k \in \mathbb{N}$, we have*

$$f_{D,\lambda}^{(k)} = g_\lambda^{(k)}(L_{K,D})S_D^T y_D, \tag{27}$$

where $g_\lambda^{(k)}(L_{K,D})$ is an operator on $\mathcal{H}_K$ defined by spectral calculus and

$$g_\lambda^{(k)}(\sigma) = \sum_{j=0}^{k-1}(\lambda(\sigma+\lambda)^{-1})^{k-j-1}(\sigma+\lambda)^{-1} = \sum_{j=0}^{k-1}\frac{\lambda^{k-1-j}}{(\sigma+\lambda)^{k-j}}. \tag{28}$$

Based on Lemma 12, we derive the following two lemmas, showing some special spectral properties of BKRR.

Lemma 13 *Let $g_\lambda^{(k)}$ be defined by (28), then we have*

$$I - L_{K,D}g_\lambda^{(k)}(L_{K,D}) = \lambda^k(L_{K,D}+\lambda I)^{-k}, \tag{29}$$

$$\|L_{K,D}g_\lambda^{(k)}(L_{K,D})]\| \le 1, \qquad \lambda\|g_\lambda^{(k)}(L_{K,D})\| \le k, \tag{30}$$

and for all $u > v > 0$, there holds

$$\|L_{K,D}^v\lambda^u(L_{K,D}+\lambda I)^{-u}\| \le v^v\left(\frac{\lambda}{u}\right)^v. \tag{31}$$

Proof. Observe from (28) that

$$\begin{aligned} \sigma g_\lambda^{(k)}(\sigma) &= (\sigma+\lambda-\lambda)\sum_{j=0}^{k-1}((\sigma+\lambda)^{-1})^{-j}\lambda^{k-j-1}(\sigma+\lambda)^{-k} \\ &= \left\{\sum_{j=0}^{k-1}(\sigma+\lambda)^{j+1}\lambda^{k-j-1} - \sum_{j=0}^{k-1}(\sigma+\lambda)^j\lambda^{k-j}\right\}(\sigma+\lambda)^{-k} \\ &= \{(\sigma+\lambda)^k - \lambda^k\}(\sigma+\lambda)^{-k}. \end{aligned}$$

Hence

$$L_{K,D}g_\lambda^{(k)}(L_{K,D}) = \left[(L_{K,D}+\lambda I)^k - \lambda^k I\right](L_{K,D}+\lambda I)^{-k} \tag{32}$$

and (29) follows. Then spectral analysis with the eigenpairs $\{(\sigma_i^{\mathbf{x}}, \phi_i^{\mathbf{x}})\}$ of $L_{K,D}$ verifies the first inequality of (30). The second inequality of (30) follows directly from (28). Set a function $h_{v,u}$ on $[0,\infty)$ by

$$h_{v,u}(\sigma) = \frac{\sigma^v\lambda^u}{(\sigma+\lambda)^u}.$$

Since $u > v$, we have $h_{v,u}(0) = h_{v,u}(\infty) = 0$. It is easy to check that $\sigma = \frac{v\lambda}{u-v}$ is the unique maximum point of $h_{v,u}$ on $(0,\infty)$. Thus, $\|L_{K,D}\| \le \kappa^2$ yields

$$\begin{aligned} \|L_{K,D}^v\lambda^u(L_{K,D}+\lambda I)^{-u}\| &\le \max_{0\le\sigma\le\kappa^2} h_{v,u}(\sigma) \le \max_{0\le\sigma<\infty} h_{v,u}(\sigma) \le h_{v,u}\left(\frac{v\lambda}{u-v}\right) \\ = \lambda^v v^v\frac{(u-v)^u}{(u-v)^v u^u} &= v^v\left(\frac{\lambda}{u}\right)^v\left(\frac{u-v}{u}\right)^{u-v} \le v^v\left(\frac{\lambda}{u}\right)^v. \end{aligned}$$

This completes the proof of Lemma 13. ∎

Lemma 14 *Let $u > 0$ and $\ell \in \mathbb{N}_0 := \{0\} \cup \mathbb{N}$. Then for $f \in \mathcal{H}_K$, we have*

$$\|L_{K,D}^u[I - (L_{K,D}+\lambda I)^{-1}L_{K,D}]^\ell f\|_K \le \begin{cases} \kappa^{2(u-\ell)}\lambda^\ell\|f\|_K, & \text{if } \ell \le u, \\ \lambda^u\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{\ell-u}\|f\|_K, & \text{if } \ell > u. \end{cases} \tag{33}$$

Proof. Due to spectral calculus with $f = \sum_i\langle f, \phi_i^{\mathbf{x}}\rangle_K\phi_i^{\mathbf{x}}$, we have

$$\begin{aligned} \|L_{K,D}^u[I - (L_{K,D}+\lambda I)^{-1}L_{K,D}]^\ell f\|_K^2 &= \sum_i\frac{(\sigma_i^{\mathbf{x}})^{2u}\lambda^{2\ell}}{(\sigma_i^{\mathbf{x}}+\lambda)^{2\ell}}|\langle f, \phi_i^{\mathbf{x}}\rangle_K|^2 \\ = \sum_{\sigma_i^{\mathbf{x}}>0}\frac{(\sigma_i^{\mathbf{x}})^{2u}\lambda^{2\ell}}{(\sigma_i^{\mathbf{x}}+\lambda)^{2\ell}}|\langle f, \phi_i^{\mathbf{x}}\rangle_K|^2. \end{aligned} \tag{34}$$

If $\ell \le u$, we have from $\max_i\sigma_i^{\mathbf{x}} \le \kappa^2$ and (34) that

$$\|L_{K,D}^u[I - (L_{K,D}+\lambda I)^{-1}L_{K,D}]^\ell f\|_K^2 \le \lambda^{2\ell}\kappa^{4(u-\ell)}\sum_{\sigma_i^{\mathbf{x}}>0}|\langle f, \phi_i^{\mathbf{x}}\rangle_K|^2.$$

Then

$$\|L_{K,D}^u[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^\ell f\|_K \le \kappa^{2(u-\ell)}\lambda^\ell \|f\|_K.$$

If $\ell > u$, we get from (34) that

$$\begin{aligned}\|L_{K,D}^u[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^\ell f\|_K^2 &\le \lambda^{2u}\sum_{\sigma_i^{\mathbf{x}}>0}\frac{\lambda^{2\ell-2u}}{(\sigma_i^{\mathbf{x}}+\lambda)^{2\ell-2u}}|\langle f,\phi_i^{\mathbf{x}}\rangle_K|^2\\ \le\ \lambda^{2u}\frac{\lambda^{2\ell-2u}}{(\sigma_{min}^{\mathbf{x}}+\lambda)^{2\ell-2u}}\sum_{\sigma_i^{\mathbf{x}}>0}|\langle f,\phi_i^{\mathbf{x}}\rangle_K|^2 &\le \lambda^{2u}\frac{\lambda^{2\ell-2u}}{(\sigma_{min}^{\mathbf{x}}+\lambda)^{2\ell-2u}}\sum_i|\langle f,\phi_i^{\mathbf{x}}\rangle_K|^2.\end{aligned}$$

Hence,

$$\|L_{K,D}^u[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^\ell f\|_K \le \lambda^u\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{\ell-u}\|f\|_K.$$

This completes the proof of Lemma 14. ■

4.3. Bounding the Bias

Our error decomposition will be carried out by bounding the two terms, bias and variance, as follows

$$\|f_{D,\lambda}^{(k)}-f_\rho\| \le \|f_{D,\lambda}^{(k,*)}-f_\rho\| + \|f_{D,\lambda}^{(k)}-f_{D,\lambda}^{(k,*)}\|,$$

where $\|\cdot\|$ denotes either the $\|\cdot\|_D$ semi-norm or $\|\cdot\|_\rho$ norm. For $f\in\mathcal{H}_K$, it is easy to check that

$$\|f-f_\rho\|_D = \|L_{K,D}^{1/2}(f-f_\rho)\|_K. \tag{35}$$

In this subsection, we present two bounds for the bias term $f_{D,\lambda}^{(k,*)}-f_\rho$ in terms of the $\|\cdot\|_D$ semi-norm and $\|\cdot\|_\rho$ norm.

Proposition 15 *Let $0\le\nu\le 1/2$. Under condition (5) with $r>3/2$, we have*

$$\begin{aligned}\|L_{K,D}^\nu(f_{D,\lambda}^{(k,*)}-f_\rho)\|_K &\le \frac{(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho}{k^\nu}\lambda^\nu\mathcal{R}_D\\ +\ &\begin{cases}\kappa^{2(r+\nu-1/2-k)}\lambda^k\|h_\rho\|_\rho, & \text{if } k\le r+\nu-1/2,\\ \lambda^{r+\nu-1/2}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r-\nu+1/2}\|h_\rho\|_\rho, & \text{if } k>r+\nu-1/2.\end{cases}\end{aligned} \tag{36}$$

Proof. Since $r>3/2$, from (5) and (26) we find

$$\begin{aligned}&\|L_{K,D}^\nu(f_{D,\lambda}^{(k,*)}-f_\rho)\|_K = \|L_{K,D}^\nu[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k L_K^{r-1/2}L_K^{1/2}h_\rho\|_K\\ =\ &\left\|L_{K,D}^\nu[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k\left(L_K^{r-1/2}-L_{K,D}^{r-1/2}+L_{K,D}^{r-1/2}\right)L_K^{1/2}h_\rho\right\|_K\\ \le\ &\left\|L_{K,D}^{r+\nu-1/2}[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k L_K^{1/2}h_\rho\right\|_K\\ +\ &\left\|L_{K,D}^\nu[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k\left(L_K^{r-1/2}-L_{K,D}^{r-1/2}\right)L_K^{1/2}h_\rho\right\|_K\\ =:\ &A_1+A_2.\end{aligned} \tag{37}$$

We first estimate A_2. Since $r>3/2$, the bounds $\|L_{K,D}\|\le\kappa^2$, $\|L_K\|\le\kappa^2$, we get from (68) in Appendix A that

$$\|L_{K,D}^{r-1/2}-L_K^{r-1/2}\|_{HS} \le (r-1/2)\kappa^{2r-3}\|L_{K,D}-L_K\|_{HS}. \tag{38}$$

When $\nu>0$, we apply (31) with $v=\nu$ to obtain

$$\begin{aligned}A_2 &\le \|L_{K,D}^\nu[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k\|\left\|L_K^{r-1/2}-L_{K,D}^{r-1/2}\right\|\|L_K^{1/2}h_\rho\|_K\\ &\le \frac{(r-1/2)\nu^\nu\kappa^{2r-3}\|h_\rho\|_\rho}{k^\nu}\lambda^\nu\mathcal{R}_D.\end{aligned}$$

If $\nu=0$, we can also obtain from $\|[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k\|\le 1$ that

$$A_2 \le (r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\mathcal{R}_D.$$

Then we estimate A_1 by applying (33) with $u=r+\nu-1/2$ and $f=L_K^{1/2}h_\rho$ to get

$$A_1 \le \begin{cases}\kappa^{2(r+\nu-1/2-k)}\lambda^k\|h_\rho\|_\rho, & \text{if } k\le r+\nu-1/2,\\ \lambda^{r+\nu-1/2}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r-\nu+1/2}\|h_\rho\|_\rho, & \text{if } k>r+\nu-1/2.\end{cases}$$

Plugging the estimates of A_1 and A_2 into (37), we obtain (36), which completes the proof of Proposition 15. ■

Proposition 16 *Under condition (5) with $r>3/2$, we have*

$$\begin{aligned}\|f_{D,\lambda}^{(k,*)}-f_\rho\|_\rho &\le 2\mathcal{Q}_{D,\lambda}(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\lambda^{\frac12}\mathcal{R}_D\\ +\ \mathcal{Q}_{D,\lambda}\|h_\rho\|_\rho&\begin{cases}\kappa^{2r-2k-1}(\lambda^{\frac12}+\kappa)\lambda^k, & \text{if } k\le r-1/2,\\ \lambda^r\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r+1/2}+\kappa^{2r-2k}\lambda^k, & \text{if } r-1/2<k\le r,\\ 2\lambda^r\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r}, & \text{if } k>r.\end{cases}\end{aligned}$$

Proof. Since $f_{D,\lambda}^{(k,*)}-f_\rho\in\mathcal{H}_K$, we have from Lemma 24 in Appendix A that

$$\|f_{D,\lambda}^{(k,*)}-f_\rho\|_\rho \le \mathcal{Q}_{D,\lambda}\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k,*)}-f_\rho)\|_K + \mathcal{Q}_{D,\lambda}\lambda^{1/2}\|f_{D,\lambda}^{(k,*)}-f_\rho\|_K.$$

For $k\le r-1/2$, it follows from (36) with $\nu=1/2$ and $\nu=0$ that

$$\begin{aligned}\|f_{D,\lambda}^{(k,*)}-f_\rho\|_\rho &\le \mathcal{Q}_{D,\lambda}\left(\frac{(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho}{\sqrt{k}}\lambda^{1/2}\mathcal{R}_D+\kappa^{2(r-k)}\lambda^k\|h_\rho\|_\rho\right)\\ +\ \lambda^{1/2}\mathcal{Q}_{D,\lambda}&\left((r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\mathcal{R}_D+\kappa^{2(r-1/2-k)}\lambda^k\|h_\rho\|_\rho\right).\end{aligned}$$

For $r-1/2<k\le r$,

$$\begin{aligned}\|f_{D,\lambda}^{(k,*)}-f_\rho\|_\rho &\le \mathcal{Q}_{D,\lambda}\left(\frac{(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho}{\sqrt{k}}\lambda^{1/2}\mathcal{R}_D+\kappa^{2(r-k)}\lambda^k\|h_\rho\|_\rho\right)\\ +\ \lambda^{1/2}\mathcal{Q}_{D,\lambda}&\left((r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\mathcal{R}_D+\lambda^{r-1/2}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r+1/2}\|h_\rho\|_\rho\right).\end{aligned}$$

For $k > r$,

$$\|f_{D,\lambda}^{(k,*)} - f_\rho\|_\rho \le \mathcal{Q}_{D,\lambda}\left(\frac{(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\lambda^{1/2}}{\sqrt{k}}\mathcal{R}_D + \lambda^r\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r}\|h_\rho\|_\rho\right)$$
$$+ \quad \lambda^{1/2}\mathcal{Q}_{D,\lambda}\left((r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\mathcal{R}_D + \lambda^{r-1/2}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r+1/2}\|h_\rho\|_\rho\right).$$

This completes the proof of Proposition 16. ∎

4.4. Bounding the Variance

In this subsection, we present the bounds for the variance term $f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)}$.

Proposition 17 *Let $0 < \nu \le 1/2$. We have*

$$\left\|L_{K,D}^\nu(f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)})\right\|_K \le \lambda^{\nu-1/2}\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}\left(1 + \sum_{j=1}^{k-1}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{j-\nu}\right). \tag{39}$$

Proof. Due to Lemma 11, we get

$$L_{K,D}^\nu(f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)}) = \sum_{j=0}^{k-1} L_{K,D}^\nu[I - (L_{K,D}+\lambda I)^{-1}L_{K,D}]^j[f_{D,\lambda}^{(1)} - f_{D,\lambda}^{(1,*)}].$$

It then follows from (33) with $u = \nu$, $\ell = 1, 2, \ldots, k-1$ and $f = f_{D,\lambda}^{(1)} - f_{D,\lambda}^{(1,*)}$ that

$$\left\|L_{K,D}^\nu(f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)})\right\|_K \le \sum_{j=0}^{k-1}\left\|L_{K,D}^\nu[I - (L_{K,D}+\lambda I)^{-1}L_{K,D}]^j[f_{D,\lambda}^{(1)} - f_{D,\lambda}^{(1,*)}]\right\|_K$$
$$\le \quad \|L_{K,D}^\nu(f_{D,\lambda}^{(1)} - f_{D,\lambda}^{(1,*)})\|_K + \sum_{j=1}^{k-1}\lambda^\nu\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{j-\nu}\|f_{D,\lambda}^{(1)} - f_{D,\lambda}^{(1,*)}\|_K.$$

But (23) implies that for $0 \le u \le 1/2$, there holds

$$\|L_{K,D}^u(f_{D,\lambda}^{(1)} - f_{D,\lambda}^{(1,*)})\|_K = \|L_{K,D}^u(L_{K,D}+\lambda I)^{-1}(S_D^T y_D - L_{K,D}f_\rho)\|_K$$
$$\le \quad \lambda^{-1/2+u}\|(L_{K,D}+\lambda I)^{-1/2}(S_D^T y_D - L_{K,D}f_\rho)\|_K \le \lambda^{-1/2+u}\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}. \tag{40}$$

Applying this inequality with $u = \nu$ and $u = 0$ yields

$$\left\|L_{K,D}^\nu(f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)})\right\|_K \quad \le \quad \lambda^{\nu-1/2}\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda} + \sum_{j=1}^{k-1}\lambda^{\nu-1/2}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{j-\nu}\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}.$$

This completes the proof of Proposition 17. ∎

Different from Proposition 15, Proposition 17 does not hold for $\nu = 0$, which makes the bound of variance in the out-sample case totally different from that in the in-sample case.

Proposition 18 *For $k \in \mathbb{N}$, we have*

$$\|f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)}\|_\rho \le 2k\mathcal{Q}_{D,\lambda}^2\mathcal{P}_{D,\lambda}.$$

Proof. We obtain from Lemma 11 and (40) with $u = 0$ that

$$\|f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)}\|_K \le \sum_{j=0}^{k-1}\|f_{D,\lambda}^{(1)} - f_{D,\lambda}^{(1,*)}\|_K \le k\lambda^{-\frac{1}{2}}\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}.$$

Then it follows from Lemma 24 in Appendix A and (39) with $\nu = 1/2$ that

$$\|f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)}\|_\rho \le \mathcal{Q}_{D,\lambda}\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)})\|_K + \mathcal{Q}_{D,\lambda}\lambda^{1/2}\|f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)}\|_K$$
$$\le \quad \mathcal{Q}_{D,\lambda}^2\mathcal{P}_{D,\lambda}\left(1 + \sum_{j=1}^{k-1}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{j-1/2}\right) + k\mathcal{Q}_{D,\lambda}^2\mathcal{P}_{D,\lambda} \le 2k\mathcal{Q}_{D,\lambda}^2\mathcal{P}_{D,\lambda}.$$

This completes the proof of Proposition 18. ∎

4.5. Bounding the Number of Boosting Iterations

We first show the important role of the stopping rule (16) in controlling the bias.

Lemma 19 *Let $\delta \in (0,1)$ and $\lambda > 0$. If $\hat{k}$ is the smallest positive integer satisfying (16), then with confidence $1-\delta$, there holds*

$$\|L_{K,D}f_\rho - L_{K,D}f_{D,\lambda}^{(\hat{k},*)}\|_K \le \frac{3}{2}\lambda^{\frac{1}{2}}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta}, \tag{41}$$

and

$$\|L_{K,D}f_\rho - L_{K,D}f_{D,\lambda}^{(\hat{k}-1,*)}\|_K \ge \frac{1}{2}\lambda^{\frac{1}{2}}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta}, \qquad \text{if } \hat{k} \ge 2. \tag{42}$$

Proof. For $k \in \mathbb{N}$, we have

$$L_{K,D}f_\rho - L_{K,D}f_{D,\lambda}^{(k,*)} = S_D^T y_D - L_{K,D}f_{D,\lambda}^{(k)} + L_{K,D}f_{D,\lambda}^{(k)} - L_{K,D}f_{D,\lambda}^{(k,*)} + L_{K,D}f_\rho - S_D^T y_D.$$

But (27) and

$$f_{D,\lambda}^{(k,*)} = g_\lambda^{(k)}(L_{K,D})L_{K,D}f_\rho \tag{43}$$

yield

$$f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)} = g_\lambda^{(k)}(L_{K,D})(S_D^T y_D - L_{K,D}f_\rho). \tag{44}$$

It then follows from (32) that

$$L_{K,D}f_\rho - L_{K,D}f_{D,\lambda}^{(k,*)} \quad = \quad S_D^T y_D - L_{K,D}f_{D,\lambda}^{(k)} + [L_{K,D}g_\lambda^{(k)}(L_{K,D}) - I](S_D^T y_D - L_{K,D}f_\rho)$$
$$= \quad S_D^T y_D - L_{K,D}f_{D,\lambda}^{(k)} - \lambda^k(L_{K,D}+\lambda I)^{-k}(S_D^T y_D - L_{K,D}f_\rho). \tag{45}$$

Moreover, (17) implies

$$\begin{aligned}
&\|\lambda^k(L_{K,D}+\lambda I)^{-k}(S_D^T y_D - L_{K,D}f_\rho)\|_K \le \|\lambda^k(L_{K,D}+\lambda I)^{-k}(L_{K,D}+\lambda I)^{1/2}\| \\
\times\ &\|(L_{K,D}+\lambda I)^{-1/2}(L_K+\lambda I)^{1/2}\|\|(L_K+\lambda I)^{-1/2}(S_D^T y_D - L_{K,D}f_\rho)\|_K \\
\le\ &\lambda^{1/2}\mathcal{P}_{D,\lambda}\mathcal{Q}_{D,\lambda} \le \frac{1}{2}\lambda^{1/2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \qquad (46)
\end{aligned}$$

with confidence $1-\delta$. Combining this with (16) and (45), we have that with confidence $1-\delta$, there holds

$$\|L_{K,D}f_\rho - L_{K,D}f_{D,\lambda}^{(\hat{k},*)}\|_K \le \frac{3}{2}\lambda^{1/2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta},$$

which proves (41). To prove (42), the definition of $\hat{k}$ implies

$$\|S_D^T y_D - L_{K,D}f_{D,\lambda}^{(\hat{k}-1)}\|_K > \lambda^{1/2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta}, \qquad \text{if } \hat{k}\ge 2.$$

It follows from (45) with $k=\hat{k}-1$ when $\hat{k}\ge 2$ that

$$\|L_{K,D}f_\rho - L_{K,D}f_{D,\lambda}^{(\hat{k}-1,*)}\|_K \ge \lambda^{1/2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} - \frac{1}{2}\lambda^{1/2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} = \frac{1}{2}\lambda^{1/2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta}$$

holds with confidence at least $1-\delta$. This completes the proof of Lemma 19. ■

Based on the above important lemma, we derive the following bound for $\hat{k}$.

Proposition 20 *Let $\delta\in(0,1)$ and $\hat{k}$ be the smallest positive integer satisfying (16). We have with confidence $1-\delta$ that*

$$\begin{aligned}
\hat{k} \ &\le\ (4r+2) + 4\mathcal{W}_{D,\lambda}^{-1}\left(\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}}{\sqrt{\lambda}}\right)^2+1\right)\lambda^r\|h_\rho\|_\rho\log^{-2}\frac{16}{\delta} \\
&+\ (4r-2)\mathcal{W}_{D,\lambda}^{-1}\frac{4\kappa^{2r-1}}{\sqrt{|D|}}\lambda^{1/2}\|h_\rho\|_\rho\log^{-3}\frac{16}{\delta}, \qquad (47)
\end{aligned}$$

where

$$\mathcal{A}_{D,\lambda} := \frac{1}{\sqrt{|D|}}\left(\frac{1}{\sqrt{|D|\lambda}}+\sqrt{\mathcal{N}(\lambda)}\right). \qquad (48)$$

Proof. If $\hat{k}\le r+3/2$, (47) obviously holds. Now we prove (47) for $\hat{k}>r+3/2$. Due to Lemma 12, Lemma 19 and (43), we have with confidence at least $1-\delta$ that

$$\begin{aligned}
&\lambda^{\frac{1}{2}}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \le 2\|L_{K,D}(f_\rho - f_{D,\lambda}^{(\hat{k}-1,*)})\|_K \\
=\ &2\left\|L_{K,D}\left(g_\lambda^{(\hat{k}-1)}(L_{K,D})L_{K,D}-I\right)f_\rho\right\|_K \le 2\|L_{K,D}\lambda^{\hat{k}-1}(L_{K,D}+\lambda I)^{-\hat{k}+1}L_K^{r-1/2}\|\|h_\rho\|_\rho,
\end{aligned}$$

where the last inequality is due to (32).

If $\frac{1}{2}\le r\le\frac{3}{2}$, we have from Lemma 13 and $\hat{k}-r-\frac{1}{2}>1$ that

$$\begin{aligned}
&\lambda^{\frac{1}{2}}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \le 2\|L_{K,D}\lambda^{\hat{k}-1}(L_{K,D}+\lambda I)^{-\hat{k}+1}(L_{K,D}+\lambda I)^{r-1/2}\|\mathcal{Q}_{D,\lambda}^{2r-1}\|h_\rho\|_\rho \\
=\ &2\mathcal{Q}_{D,\lambda}^{2r-1}\|h_\rho\|_\rho\lambda^{r-1/2}\|L_{K,D}\lambda^{\hat{k}-1-r+1/2}(L_{K,D}+\lambda I)^{-\hat{k}+1+r-1/2}\| \\
\le\ &2\max(\mathcal{Q}_{D,\lambda}^2,1)\|h_\rho\|_\rho\lambda^{r+1/2}(\hat{k}-r-1/2)^{-1}.
\end{aligned}$$

Thus, it follows from Lemma 22 in Appendix A that with confidence at least $1-\delta$

$$\hat{k}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \le (r+1/2)\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} + 4\log^2\frac{16}{\delta}\left(\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}}{\sqrt{\lambda}}\right)^2+1\right)\lambda^r\|h_\rho\|_\rho,$$

which implies (47).

If $r>3/2$, it follows from (38), Lemma 13 and $\hat{k}>r+3/2$ that

$$\begin{aligned}
&\lambda^{1/2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \le 2\|L_{K,D}\lambda^{\hat{k}-1}(L_{K,D}+\lambda I)^{-\hat{k}+1}L_{K,D}^{r-1/2}\|\|h_\rho\|_\rho \\
+\ &2\|L_{K,D}\lambda^{\hat{k}-1}(L_{K,D}+\lambda I)^{-\hat{k}+1}(L_K^{r-1/2}-L_{K,D}^{r-1/2})\|\|h_\rho\|_\rho \\
\le\ &2\|h_\rho\|_\rho\lambda^{r-1/2}\|L_{K,D}\lambda^{\hat{k}-1-r+1/2}(L_{K,D}+\lambda I)^{-\hat{k}+1+r-1/2}\| \\
+\ &(2r-1)\kappa^{2r-3}\|h_\rho\|_\rho\|L_{K,D}\lambda^{\hat{k}-1}(L_{K,D}+\lambda I)^{-\hat{k}+1}\|\|L_K-L_{K,D}\| \\
\le\ &2\|h_\rho\|_\rho\lambda^{r+1/2}(\hat{k}-r-1/2)^{-1} + (2r-1)\kappa^{2r-3}\|h_\rho\|_\rho\frac{\lambda}{\hat{k}-1}\mathcal{R}_D.
\end{aligned}$$

If

$$\lambda^{r+1/2}(\hat{k}-r-1/2)^{-1} \le (r-1/2)\kappa^{2r-3}\frac{\lambda}{\hat{k}-1}\mathcal{R}_D,$$

we have

$$\lambda^{\frac{1}{2}}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \le (4r-2)\kappa^{2r-3}\frac{\lambda}{\hat{k}-1}\mathcal{R}_D\|h_\rho\|_\rho,$$

which together with Lemma 22 in Appendix A yields with confidence $1-\delta$

$$\hat{k}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \le \mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} + (4r-2)\frac{4\kappa^{2r-1}}{\sqrt{|D|}}\lambda^{1/2}\|h_\rho\|_\rho\log\frac{16}{\delta}.$$

Thus, (47) holds. If

$$\lambda^{r+1/2}(\hat{k}-r-1/2)^{-1} > (r-1/2)\kappa^{2r-3}\frac{\lambda}{\hat{k}-1}\mathcal{R}_D,$$

we get

$$\lambda^{\frac{1}{2}}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \le 4\lambda^{r+1/2}(\hat{k}-r-1/2)^{-1}\|h_\rho\|_\rho.$$

Hence,

$$\hat{k}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} \le (r+\frac{1}{2})\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta} + 4\lambda^r\|h_\rho\|_\rho,$$

which together with $\log\frac{16}{\delta}>1$ yields (47). The proof of Proposition 20 is completed. ■

5. Proofs of Main Results

Based on the previous bounds we can now prove our main results.

Proof of Theorem 3. If $1/2 \le r \le 3/2$, we get from (5) and (26) that

$$\begin{aligned}
&\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k,*)} - f_\rho)\|_K = \|L_{K,D}^{1/2}[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k L_K^r h_\rho\|_K \\
= \;& \|L_{K,D}^{1/2}[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k (L_{K,D}+\lambda I)^{r-1/2}(L_{K,D}+\lambda I)^{1/2-r}L_K^r h_\rho\|_K \\
\le \;& 2^{r-1/2}\|L_{K,D}^{r}[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k (L_{K,D}+\lambda I)^{1/2-r}L_K^r h_\rho\|_K \\
+ \;& 2^{r-1/2}\lambda^{r-1/2}\|L_{K,D}^{1/2}[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k (L_{K,D}+\lambda I)^{1/2-r}L_K^r h_\rho\|_K.
\end{aligned}$$

Here we have used the inequality

$$\|(L_{K,D}+\lambda I)^r f\|_K \le 2^{r-1/2}\big[\|L_{K,D}^r f\|_K + \lambda^r \|f\|_K\big], \quad \forall f \in \mathcal{H}_K$$

which follows by means of the normalized eigenpairs $\{(\sigma_i^{\mathbf{x}}, \phi_i^{\mathbf{x}})\}_i$ of $L_{K,D}: \mathcal{H}_K \to \mathcal{H}_K$ as

$$\begin{aligned}
\|(L_{K,D}+\lambda I)^r f\|_K^2 &= \sum_i \big(\sigma_i^{\mathbf{x}}+\lambda\big)^{2r} |\langle \phi_i^{\mathbf{x}}, f\rangle_K|^2 \\
&\le 2^{2r-1}\sum_i \big[(\sigma_i^{\mathbf{x}})^{2r}|\langle \phi_i^{\mathbf{x}}, f\rangle_K|^2 + \lambda^{2r}|\langle \phi_i^{\mathbf{x}}, f\rangle_K|^2\big].
\end{aligned}$$

Since $0 \le r - \frac{1}{2} \le 1$, (67) in Appendix B shows

$$\|(L_{K,D}+\lambda I)^{1/2-r}L_K^r h_\rho\|_K \le \mathcal{Q}_{D,\lambda}^{2r-1}\|h_\rho\|_\rho.$$

It then follows from (33) with $u = r$, $f = (L_{K,D}+\lambda I)^{1/2-r}L_K^r h_\rho$ and $u = 1/2$, $f = (L_{K,D}+\lambda I)^{1/2-r}L_K^r h_\rho$ that

$$\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k,*)} - f_\rho)\|_K \le 2\sqrt{2}\mathcal{Q}_{D,\lambda}^{2r-1}\|h_\rho\|_\rho \lambda^r \left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r}, \qquad \text{if } 1/2 \le r < 1,$$

$$\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k,*)} - f_\rho)\|_K \le 2\mathcal{Q}_{D,\lambda}^{2r-1}\lambda\|h_\rho\|_\rho(\kappa^{2r-2}+\lambda^{r-1}), \qquad \text{if } 1 \le r \le 3/2, k = 1,$$

and

$$\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k,*)} - f_\rho)\|_K \le 4\mathcal{Q}_{D,\lambda}^{2r-1}\lambda^r\|h_\rho\|_\rho \left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-r}, \qquad \text{if } 1 \le r \le 3/2, k \ge 2.$$

The above estimates together with (36) with $\nu = 1/2$ yield for $r \ge 1/2$,

$$\begin{aligned}
\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k,*)} - f_\rho)\|_K &\le \frac{(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho}{\sqrt{k}}\lambda^{1/2}\mathcal{R}_D \\
&+ 4(\mathcal{Q}_{D,\lambda}^2+1)\lambda^k\|h\|_\rho \begin{cases} (\sigma_{min}^{\mathbf{x}}+\lambda)^{r-k}, & \text{if } k > r, \\ \kappa^{2r-2k}+\lambda^{r-k}, & \text{if } k \le r. \end{cases}
\end{aligned}$$

Furthermore, (39) with $\nu = 1/2$ implies

$$\Big\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)})\Big\|_K \le \mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}\left(1 + \sum_{j=1}^{k-1}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{j-1/2}\right).$$

Hence

$$\begin{aligned}
&\Big\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k)} - f_\rho)\Big\|_K \le \|L_{K,D}^{1/2}(f_{D,\lambda}^{(k,*)} - f_\rho)\|_K + \Big\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)})\Big\|_K \\
\le \;& \mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}\sum_{j=1}^{k-1}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{j-1/2} + \frac{(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho}{\sqrt{k}}\lambda^{1/2}\mathcal{R}_D + \mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda} \\
+ \;& 4(\mathcal{Q}_{D,\lambda}^2+1)\lambda^k\|h\|_\rho \begin{cases} (\sigma_{min}^{\mathbf{x}}+\lambda)^{r-k}, & \text{if } k > r, \\ \kappa^{2r-2k}+\lambda^{r-k}, & \text{if } k \le r. \end{cases}
\end{aligned}$$

This completes the proof of Theorem 3. ∎

Proof of Theorem 4. Since $r > 3/2$, we get from Proposition 16 that

$$\begin{aligned}
\|f_{D,\lambda}^{(k,*)} - f_\rho\|_\rho &\le 2\mathcal{Q}_{D,\lambda}(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\lambda^{1/2}\mathcal{R}_D \\
+ \;& \mathcal{Q}_{D,\lambda}\lambda^k\|h_\rho\|_\rho \begin{cases} \lambda^{r-k}+\kappa^{2r-2k-1}(\kappa+\lambda^{\frac{1}{2}}), & \text{if } k \le r, \\ 2(\sigma_{min}^{\mathbf{x}}+\lambda)^{r-k}, & \text{if } k > r. \end{cases}
\end{aligned}$$

Furthermore, it follows from Proposition 18 that

$$\|f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)}\|_\rho \le 2k\mathcal{Q}_{D,\lambda}^2\mathcal{P}_{D,\lambda}.$$

Then

$$\begin{aligned}
&\|f_{D,\lambda}^{(k)} - f_\rho\|_\rho \le \|f_{D,\lambda}^{(k,*)} - f_\rho\|_\rho + \|f_{D,\lambda}^{(k)} - f_{D,\lambda}^{(k,*)}\|_\rho \\
\le \;& 2\mathcal{Q}_{D,\lambda}(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\lambda^{1/2}\mathcal{R}_D \\
+ \;& 2k\mathcal{Q}_{D,\lambda}^2\mathcal{P}_{D,\lambda} + \mathcal{Q}_{D,\lambda}\lambda^k\|h_\rho\|_\rho \begin{cases} \lambda^{r-k}+\kappa^{2r-2k-1}(\kappa+\lambda^{\frac{1}{2}}), & \text{if } k \le r, \\ 2(\sigma_{min}^{\mathbf{x}}+\lambda)^{r-k}, & \text{if } k > r. \end{cases}
\end{aligned}$$

This completes the proof of Theorem 4. ∎

Proof of Theorem 1. We get from Lemma 22 in Appendix A, (48), $r > 3/2$, (6) and $\lambda = \left(\frac{k^2}{|D|}\right)^{\frac{1}{2\min\{k,r\}+s}}$ with $k \le \sqrt{|D|}$ that with confidence $1-\delta$,

$$\mathcal{R}_D \le \frac{4\kappa^2}{\sqrt{|D|}}\log\frac{8}{\delta} \le \frac{4\kappa^2}{\sqrt{|D|\lambda^s}}\log\frac{8}{\delta},$$

$$\mathcal{Q}_{D,\lambda}^2 \le 2\left(2(\kappa^2+\kappa)\left(\frac{1}{|D|\lambda}+\sqrt{\frac{\mathcal{N}(\lambda)}{|D|\lambda}}\right)\log\frac{8}{\delta}\right)^2 + 2 \le 8(\kappa^2+\kappa+1)^2(1+\sqrt{C_0})^2\log^2\frac{8}{\delta},$$

and

$$\begin{aligned}
&\mathcal{P}_{D,\lambda}\mathcal{Q}_{D\lambda} \le \frac{4\sqrt{2}(\kappa^2+\kappa+1)(\kappa M+\gamma)}{\sqrt{|D|}}\left(\frac{1}{\lambda|D|}+\frac{\sqrt{\mathcal{N}(\lambda)}}{\sqrt{|D|\lambda}}+1\right)\left(\frac{1}{\sqrt{\lambda|D|}}+\sqrt{\mathcal{N}(\lambda)}\right)\log^2\frac{8}{\delta} \\
\le \;& \frac{4\sqrt{2}(\kappa^2+\kappa+1)(\kappa M+\gamma)(2+\sqrt{C_0})(1+\sqrt{C_0})}{\sqrt{|D|\lambda^s}}\log^2\frac{8}{\delta}.
\end{aligned}$$

Here we have used $\frac{1}{\sqrt{\lambda|D|}} + \sqrt{\mathcal{N}(\lambda)} \le \frac{\sqrt{C_0}+1}{\sqrt{\lambda^s}}$. Applying these three estimates to Theorem 4, we get

$$\begin{aligned}
&\|f_{D,\lambda}^{(k)} - f_\rho\|_\rho \le 4\sqrt{2}(\kappa^2+\kappa+1)(1+\sqrt{C_0})(r-1/2)\kappa^{2r-3}\|h_\rho\|_\rho\lambda^{1/2}\frac{4\kappa^2}{\sqrt{|D|\lambda^s}}\log^2\frac{8}{\delta}\\
+\;& 32(2+\sqrt{C_0})^3(\kappa^2+\kappa+1)^2(\kappa M+\gamma)\frac{k}{\sqrt{|D|\lambda^s}}\log^3\frac{8}{\delta}\\
+\;& 2\sqrt{2}(1+\sqrt{C_0})(\kappa^2+\kappa+1)\log\frac{8}{\delta}\lambda^k\|h_\rho\|_\rho\begin{cases}\lambda^{r-k}+\kappa^{2r-2k-1}(\kappa+\lambda^{\frac{1}{2}}), & \text{if } k\le r,\\ 2(\sigma_{min}^{\mathbf{x}}+\lambda)^{r-k}, & \text{if } k>r.\end{cases}\\
\le\;& \frac{\tilde{C}k}{2\sqrt{|D|\lambda^s}}\log^3\frac{8}{\delta}+\frac{\tilde{C}\log\frac{8}{\delta}}{2}\lambda^k\begin{cases}1, & \text{if } k\le r,\\ (\sigma_{min}^{\mathbf{x}}+\lambda)^{r-k}, & \text{if } k>r,\end{cases}
\end{aligned}$$

where $\tilde{C}$ is a constant independent of $|D|, k$ or δ given by

$$\begin{aligned}\tilde{C} = 2(\kappa^2+\kappa+1)\max\{&16\sqrt{2}(1+\sqrt{C_0})(r-1/2)\kappa^{2r-1}\|h_\rho\|_\rho+32(2+\sqrt{C_0})^3(\kappa^2+\kappa+1)(\kappa M+\gamma),\\ &2\sqrt{2}(1+\sqrt{C_0})\|h_\rho\|_\rho(2+\kappa^{2r-2k}+\kappa^{2r-2k-1})\}.\end{aligned}$$

Plugging $\lambda = (k^2/|D|)^{\frac{1}{2\min\{k,r\}+s}}$ into the above estimate and noting $\sigma_{min}^{\mathbf{x}} > 0$, we get that

$$\left\|f_{D,\lambda}^{(k)} - f_\rho\right\|_\rho \le \tilde{C}(k^2/|D|)^{\frac{\min\{k,r\}}{2\min\{k,r\}+s}}\log^3\frac{8}{\delta},$$

holds with confidence $1-\delta$. This completes the proof of Theorem 1. ∎

Proof of Corollary 5. It follows from (25) that

$$\begin{aligned}
&\|L_{K,D}f_{D,\lambda}^{(k+1)} - S_D^T y_D\|_K = \left\|[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^{k+1}S_D^T y_D\right\|_K\\
\le\;& \|I-(L_{K,D}+\lambda I)^{-1}L_{K,D}\|\|[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k S_D^T y_D\|_K\\
\le\;& \|L_{K,D}f_{D,\lambda}^{(k)} - S_D^T y_D\|_K.
\end{aligned}$$

Thus $\|L_{K,D}f_{D,\lambda}^{(k)} - S_D^T y_D\|_K$ decreases with respect to k. Furthermore, we have from (25) again and Lemma 14 that for $k>1$

$$\begin{aligned}
&\|L_{K,D}f_{D,\lambda}^{(k)} - S_D^T y_D\|_K \le \|[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k(S_D^T y_D - L_{K,D}f_\rho)\|_K\\
+\;& \|[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k L_{K,D}f_\rho\|_K\\
\le\;& \|[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k(L_{K,D}+\lambda I)^{1/2}\|\|(L_{K,D}+\lambda I)^{-1/2}(S_D^T y_D - L_{K,D}f_\rho)\|_K\\
+\;& \|[I-(L_{K,D}+\lambda I)^{-1}L_{K,D}]^k L_{K,D}f_\rho\|_K\\
\le\;& \lambda^{1/2}\|(L_{K,D}+\lambda I)^{-1/2}(L_{K,D}f_\rho - S_D^T y_D)\|_K + \lambda\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{k-1}\|f_\rho\|_K.
\end{aligned}$$

Thus,

$$\begin{aligned}
\lim_{k\to\infty}\|L_{K,D}f_{D,\lambda}^{(k)} - S_D^T y_D\|_K &\le \lambda^{1/2}\|(L_{K,D}+\lambda I)^{-1/2}(L_{K,D}f_\rho - S_D^T y_D)\|_K\\
&\le \lambda^{1/2}\mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}.
\end{aligned}$$

This verifies (12). To prove (13), it follows from Theorem 3 that

$$\begin{aligned}
&\lim_{k\to\infty}\left\|L_{K,D}^{1/2}(f_{D,\lambda}^{(k)} - f_\rho)\right\|_K \le \mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda} + \mathcal{Q}_{D,\lambda}\mathcal{P}_{D,\lambda}\lim_{k\to\infty}\sum_{j=1}^{k-1}\left(\frac{\lambda}{\sigma_{min}^{\mathbf{x}}+\lambda}\right)^{j-1/2}\\
=\;& \left(1+\frac{(\sigma_{min}^{\mathbf{x}}+\lambda)^{1/2}\lambda^{1/2}}{\sigma_{min}^{\mathbf{x}}}\right)\mathcal{P}_{D,\lambda}\mathcal{Q}_{D,\lambda}.
\end{aligned}$$

This completes the proof of Corollary 5. ∎

Proof of Theorem 6. Since

$$\|f_{D,\lambda}^{(\hat{k})} - f_\rho\|_\rho \le \|f_{D,\lambda}^{(\hat{k},*)} - f_\rho\|_\rho + \|f_{D,\lambda}^{(\hat{k})} - f_{D,\lambda}^{(\hat{k},*)}\|_\rho =: A(D,\lambda,\hat{k}) + S(D,\lambda,\hat{k}). \tag{49}$$

We divide the proof into four steps.

Step 1: Bounding $A(D,\lambda,\hat{k})$. Define

$$\widetilde{\mathcal{Q}}_{D,\lambda} = \|(L_{K,D}+\lambda I)^{-1}(L_K+\lambda I)\|. \tag{50}$$

We obtain from $\|L_K^{1/2}(L_K+\lambda I)^{-1/2}\| \le 1$ and $\|(L_K+\lambda I)^{-\frac{1}{2}}\| \le \lambda^{-\frac{1}{2}}$ that

$$\begin{aligned}
A(D,\lambda,\hat{k}) &= \|L_K^{1/2}(f_{D,\lambda}^{(\hat{k},*)} - f_\rho)\|_K \le \lambda^{-1/2}\|(L_K+\lambda I)(f_{D,\lambda}^{(\hat{k},*)} - f_\rho)\|_K\\
&\le \lambda^{-1/2}\widetilde{\mathcal{Q}}_{D,\lambda}\|(L_{K,D}+\lambda I)(f_{D,\lambda}^{(\hat{k},*)} - f_\rho)\|_K\\
&\le \lambda^{-1/2}\widetilde{\mathcal{Q}}_{D,\lambda}\|L_{K,D}(f_{D,\lambda}^{(\hat{k},*)} - f_\rho)\|_K + \lambda^{1/2}\widetilde{\mathcal{Q}}_{D,\lambda}\|f_{D,\lambda}^{(\hat{k},*)} - f_\rho\|_K.
\end{aligned} \tag{51}$$

By Lemma 19, with confidence at least $1-\delta$, there holds

$$\|L_{K,D}(f_{D,\lambda}^{(\hat{k},*)} - f_\rho)\|_K \le \frac{3}{2}\lambda^{1/2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta}. \tag{52}$$

Let F_λ be the orthogonal projection onto the subspace of $\mathcal{H}_K$ spanned by the eigenvectors of $L_{K,D}$ associated with eigenvalues less than λ and $F_\lambda^\perp = I - F_\lambda$. We have

$$\|f_{D,\lambda}^{(\hat{k},*)} - f_\rho\|_K \le \lambda^{-1}\|F_\lambda[\lambda(f_{D,\lambda}^{(\hat{k},*)} - f_\rho)]\|_K + \lambda^{-1}\|F_\lambda^\perp[\lambda(f_{D,\lambda}^{(\hat{k},*)} - f_\rho)]\|_K =: A_1 + A_2. \tag{53}$$

By the definition of $F_\lambda^\perp$ and Lemma 19, it follows with confidence $1-\delta$,

$$A_2 \le \lambda^{-1}\|F_\lambda^\perp[L_{K,D}(f_{D,\lambda}^{(\hat{k},*)} - f_\rho)]\|_K \le \lambda^{-1}\|L_{K,D}(f_{D,\lambda}^{(\hat{k},*)} - f_\rho)\|_K \le \frac{3}{2}\lambda^{-1/2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta}. \tag{54}$$

Due to (43) and (5), we have

$$A_1 \le \|F_\lambda[g_\lambda^{(\hat{k})}(L_{K,D})L_{K,D}f_\rho - f_\rho)]\|_K \le \|F_\lambda[\lambda^{\hat{k}}(L_{K,D}+\lambda I)^{-\hat{k}}L_K^{r-1/2}]\|\|h_\rho\|_\rho.$$

If $\frac{1}{2} \le r \le \frac{3}{2}$, we have

$$A_1 \le \|F_\lambda[\lambda^{\hat{k}}(L_{K,D}+\lambda I)^{-\hat{k}}(L_{K,D}+\lambda I)^{r-1/2}]\|(\widetilde{\mathcal{Q}}_{D,\lambda})^{r-1/2}\|h_\rho\|_\rho \le (\widetilde{\mathcal{Q}}_{D,\lambda})^{r-1/2}\|h_\rho\|_\rho\lambda^{r-1/2}. \tag{55}$$

If $r > 3/2$, (38) implies

$$\begin{aligned} A_1 &\leq \|F_\lambda[\lambda^{\hat{k}}(L_{K,D}+\lambda I)^{-\hat{k}}(L_K^{r-1/2}-L_{K,D}^{r-1/2})]\|\|h_\rho\|_\rho + \|F_\lambda[\lambda^{\hat{k}}(L_{K,D}+\lambda I)^{-\hat{k}}L_{K,D}^{r-1/2}]\|\|h_\rho\|_\rho \\ &\leq \|h_\rho\|_\rho\left((r-1/2)\kappa^{2r-3}\mathcal{R}_D+\lambda^{r-1/2}\right). \end{aligned} \tag{56}$$

Inserting (56), (55) and (54) into (53) and then plugging (53) and (52) into (51), we have from Lemma 22 in Appendix A that with confidence $1-\delta$,

$$\begin{aligned} A(D,\lambda,\hat{k}) &\leq 3\log^6\frac{16}{\delta}\left[\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]\mathcal{W}_{D,\lambda} \\ &+ 4\lambda^{\frac{1}{2}}\|h_\rho\|_\rho\log^4\frac{16}{\delta}\left[\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]^2\left(\lambda^{r-\frac{1}{2}}+\frac{(4r-2)\kappa^{2r-1}}{\sqrt{|D|}}\right). \end{aligned} \tag{57}$$

Step 2: Bounding $S(D,\lambda,\hat{k})$. It follows from (30), (44) and the definitions of $\mathcal{Q}_{D,\lambda}$ and $\mathcal{P}_{D,\lambda}$ that

$$\begin{aligned} S(D,\lambda,\hat{k}) &= \|L_K^{1/2}[f_{D,\lambda}^{(\hat{k})}-f_{D,\lambda}^{(\hat{k},*)}]\|_K \\ &\leq \|(L_K+\lambda I)^{1/2}g_\lambda^{(\hat{k})}(L_{K,D})(L_{K,D}+\lambda I)^{1/2}(L_{K,D}+\lambda I)^{-1/2}(L_{K,D}f_\rho-S_D^Ty_D)\|_K \\ &\leq \mathcal{Q}_{D,\lambda}^2\|g_\lambda^{(\hat{k})}(L_{K,D})(L_{K,D}+\lambda I)\|\mathcal{P}_{D,\lambda} \\ &\leq \mathcal{Q}_{D,\lambda}^2\mathcal{P}_{D,\lambda}\left[\|g_\lambda^{(\hat{k})}(L_{K,D})L_{K,D}\|+\lambda\|g_\lambda^{(\hat{k})}(L_{K,D})\|\right] \\ &\leq (\hat{k}+1)\mathcal{Q}_{D,\lambda}^2\mathcal{P}_{D,\lambda}. \end{aligned}$$

This together with (17) and Lemma 22 in Appendix A implies with confidence $1-\delta$

$$S(D,\lambda,\hat{k}) \leq \frac{\hat{k}+1}{2}\mathcal{W}_{D,\lambda}\log^5\frac{16}{\delta}\left[\sqrt{2}\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}}{\sqrt{\lambda}}\right)+\sqrt{2}\right].$$

Combining the above inequality with (47) yields

$$\begin{aligned} S(D,\lambda,\hat{k}) &\leq \left[\sqrt{2}\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}}{\sqrt{\lambda}}\right)+\sqrt{2}\right]\log\frac{16}{\delta}\left\{(2r+1)\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta}\right. \\ &+ 2\left[\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]\lambda^r\|h_\rho\|_\rho\log^2\frac{16}{\delta} \\ &+ \left.(2r-1)\frac{4\kappa^{2r-1}}{\sqrt{|D|}}\lambda^{1/2}\|h_\rho\|_\rho\log\frac{16}{\delta}\right\}. \end{aligned} \tag{58}$$

Step 3: Bounding $\mathcal{A}_{D,\lambda}$ and $\mathcal{W}_{D,\lambda}$. Since $r \geq 1/2$, and $\lambda = (c/|D|)^{1/(2r+s)}$ with $c \geq 1$, it follows from (6), (48) and $r \geq \frac{1}{2}$ that

$$\mathcal{A}_{D,\lambda} = \frac{1}{\sqrt{|D|}}\left\{\frac{1}{\sqrt{|D|\lambda}}+\sqrt{\mathcal{N}(\lambda)}\right\} \leq (1+\sqrt{C_0})c^{-s/(4r+2s)}|D|^{-r/(2r+s)}. \tag{59}$$

This implies

$$\mathcal{A}_{D,\lambda}^2/\lambda \leq (1+\sqrt{C_0})^2c^{(-s-1)/(2r+s)}|D|^{\frac{1-2r}{2r+s}}$$

and

$$\frac{\mathcal{A}_{D,\lambda}}{\sqrt{\lambda}}+1 \leq \tilde{C}_1 \tag{60}$$

with $\tilde{C}_1 := (1+\sqrt{C_0})c^{\frac{-s-1}{4r+2s}}+1$. Now we turn to bound $\mathcal{W}_{D,\lambda}$. If $\eta_{\delta/4} := 2\log(16/\delta)/\sqrt{|D|\lambda} \leq 1$, we have from Lemma 22 in Appendix A that

$$\sqrt{\max\{\mathcal{N}_D(\lambda),1\}} \leq 5\sqrt{\max\{\mathcal{N}(\lambda),1\}}.$$

Then, it follows from (6) and $\lambda = (c/|D|)^{1/(2r+s)}$ that

$$(\sqrt{|D|\lambda}+9)\sqrt{\max\{\mathcal{N}_D(\lambda),1\}} \leq 5\max\{c^{1/(4r+2s)},9\}\max\{C_0^{1/2}c^{-s/(4r+2s)},1\}|D|^{\frac{2r+2s-1}{4r+2s}}.$$

Thus, it follows from (15) that

$$\mathcal{W}_{D,\lambda} \leq \tilde{C}_2'|D|^{-\frac{r}{2r+s}}, \tag{61}$$

where

$$\tilde{C}_2' := 16\sqrt{2}(\kappa^2+\kappa+1)(\kappa M+\gamma)\left(5(c^{-1/(2r+s)}+1)(c^{1/(4r+2s)}+9)(C_0^{1/2}c^{-s/(4r+2s)}+1)+1\right)^2.$$

If $\eta_{\delta/4} > 1$, we get from Lemma 22 in Appendix A again and $\lambda = (c/|D|)^{1/(2r+s)}$ that with confidence $1-\delta$

$$\sqrt{\max\{\mathcal{N}_D(\lambda),1\}} \leq (1+16c^{-1/(2r+s)})\sqrt{\max\{\mathcal{N}(\lambda),1\}}\log^2\frac{16}{\delta}.$$

The same argument as above shows that with confidence $1-\delta$,

$$\mathcal{W}_{D,\lambda} \leq \tilde{C}_2''|D|^{-\frac{r}{2r+s}}\log^4\frac{16}{\delta}, \tag{62}$$

where

$$\begin{aligned} \tilde{C}_2'' &:= 16\sqrt{2}(\kappa^2+\kappa+1)(\kappa M+\gamma) \\ &\times \left((1+16c^{-1/(2r+s)})(c^{-1/(2r+s)}+1)(c^{1/(4r+2s)}+9)(C_0^{1/2}c^{-s/(4r+2s)}+1)+1\right)^2. \end{aligned}$$

Combining (61) with (62), we obtain with confidence $1-\delta$

$$\mathcal{W}_{D,\lambda} \leq \tilde{C}_2|D|^{-\frac{r}{2r+s}}\log^4\frac{16}{\delta} \tag{63}$$

with $\tilde{C}_2 := \max\{\tilde{C}_2', \tilde{C}_2''\}$.

Step 4: Deriving the learning rate. Plugging (63) and (60) into (57) and (58), we have with confidence $1-\delta$ that

$$A(D,\lambda,\hat{k}) \leq \tilde{C}_3|D|^{-\frac{r}{2r+s}}\log^{10}\frac{16}{\delta} \tag{64}$$

with $\tilde{C}_3 = 3(4(\kappa^2+\kappa)^2\tilde{C}_1^2+1)\tilde{C}_2 + 4(4(\kappa^2+\kappa)^2\tilde{C}_1^2+1)^2\|h_\rho\|\big(c^{\frac{r}{2r+s}} + (4r-2)\kappa^{2r-1}c^{\frac{1}{4r+2s}}\big)$ and

$$S(D,\lambda,\hat{k}) \le \tilde{C}_4|D|^{-\frac{r}{2r+s}}\log^9\frac{16}{\delta} \tag{65}$$

with $\tilde{C}_4 = \sqrt{2}(2(\kappa^2+\kappa)\tilde{C}_1+1)\big[(2r+1)\tilde{C}_2 + 2(2(\kappa^2+\kappa)^2\tilde{C}_1^2+1)c^{\frac{r}{2r+s}}\|h_\rho\|_\rho + (8r-4)\kappa^{2r-1}\|h_\rho\|_\rho c^{\frac{1}{4r+2s}}\big]$. Putting (64) and (65) into (49), we have

$$\|f_{D,\lambda}^{(\hat{k})} - f_\rho\|_\rho \le (\tilde{C}_3+\tilde{C}_4)|D|^{-\frac{r}{2r+s}}\log^{10}\frac{16}{\delta}.$$

This completes the proof of Theorem 6 with the constant $C := \tilde{C}_3 + \tilde{C}_4$. ■

6. Empirical Studies

In this section, we report experimental results to study the behavior of BKRR and the adaptive stopping rule (16) in practice. We consider two regression problems. For the j-th regression problem ($j = 1, 2$), we assume that training examples are independently drawn from the regression model $y_i = g_j(x_i) + \xi_i, i = 1, \ldots, |D|$, where $\{x_i\}_{i=1}^{|D|}$ are drawn from the uniform distribution on the (hyper)-cube $[0,1]^{d_j}$ (d_j is the input dimension) and $\{\xi_i\}_{i=1}^{|D|}$ are noise components independently drawn from the Gaussian distribution $\mathcal{N}(0, 1/5)$. For the j-th problem, we build the estimator by applying BKRR in the RKHS induced by a Mercer kernel K_j. We consider the following two regression functions

$$g_1(x) = \min(x, 1-x), \quad x \in [0,1],$$

$$g_2(x) = (1-\|x\|_2)_+^6(35\|x\|_2^2 + 18\|x\|_2 + 3), \quad x \in [0,1]^3.$$

The two Mercer kernels are $K_1 : \mathbb{R}\times\mathbb{R} \mapsto \mathbb{R}, K_2 : \mathbb{R}^3\times\mathbb{R}^3 \mapsto \mathbb{R}$ defined by

$$K_1(x,\tilde{x}) = 1 + \min(x,\tilde{x}) \quad \text{and} \quad K_2(x,\tilde{x}) = g_3(x-\tilde{x}),$$

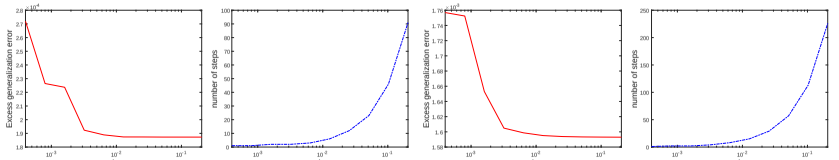
where $g_3(x) = (1-\|x\|_2)_+^4(4\|x\|_2+1), x \in [0,1]^3$. It can be found in Chang et al. (2017) that $g_1 \in \mathcal{H}_{K_1}$ with exponent $r = 1/2$ in (5) and $g_2 \in \mathcal{H}_{K_2}$ with exponent $r > 1/2$. We repeat each experiment 40 times and report the average of these experimental results.

Our numerical results are divided into three parts. In the first part, we study the relation between the generalization ability of BKRR and the regularization parameter to verify our motivation to combine KRR with boosting. In the second part, we validate the empirical behavior of BKRR and its comparison with iterated Tikhonov regularization (ITR). In the last part we show the effectiveness of adaptive stopping rule (16) in practical regression problems.

6.1. Regularization Parameters in BKRR

A common consensus in boosting theory (Friedman, 2001) is that the weak learners should be under-fitting and the boosting iteration will reduce the bias and increase the variance accordingly. For estimators with high-level under-fitting, more boosting iterations are imposed and lead to a similar learning performance as other efficient algorithms. If KRR is used to build up a weak learner, this argument then shows that we should select a relatively large λ, larger than some value, so that BKRR with a suitable number of iterations can reach a similar learning performance as KRR. Our first simulation is to verify this argument and show a relation between the generalization ability and regularization parameters in BKRR.

In this simulation, We traverse the regularization parameter λ over the set $0.0002 \times \{1, 2, 2^2, \ldots, 2^{10}\}$. For each regularization parameter, we run BKRR until k reaches 150 for $f_\rho = g_1$ and 300 for $f_\rho = g_2$, respectively. For each considered k and λ, we estimate the excess generalization error (EGE) $\mathcal{E}(f_{D,\lambda}^{(k)}) - \mathcal{E}(f_\rho)$ by $\frac{1}{2000}\sum_{i=1}^{2000}[f_{D,\lambda}^{(k)}(x_i') - f_\rho(x_i')]^2$, where $\{x_i'\}_{i=1}^{2000}$ are independently drawn from the uniform distribution on the corresponding input space. For each λ, we record the optimal k (selected to be optimal to the test data directly) and the corresponding EGE. Figure 1 reports EGEs and iteration numbers versus regularization parameters. It is shown in Figure 1 that if λ is larger than some value (near 10^{-2} in this simulation), then BKRR with different λ possesses similar learning performances provided the number of iterations is appropriately selected. Figure 1 also shows that the more high-level of under-fitting, the more boosting iterations required, which verifies the previous common consensus. All these results show that using KRR to build up a weak learner for boosting is reasonable and the selection of λ does not affect the generalization ability very much.



(a) $f_\rho = g_1$. (b) $f_\rho = g_2$.

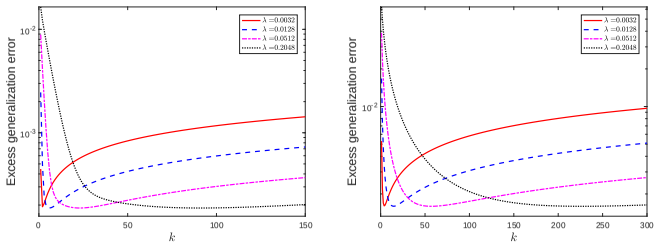
Figure 1: Excess generalization errors of BKRR versus regularization parameters. We also plot the iteration number at which the optimal EGEs are achieved.

6.2. Behavior of BKRR

In this subsection, we want to validate the empirical behavior of BKRR and its comparison with ITR. It should be mentioned that BKRR focuses on fixed regularization parameters and varying iteration numbers, while ITR focuses on fixed iteration numbers and varying regularization parameters. The aim of this simulation is to verify an advantage of BKRR over ITR in the parameter selection, showing that selecting an appropriate number of iterations in BKRR is easier than selecting an appropriate regularization parameter in ITR.

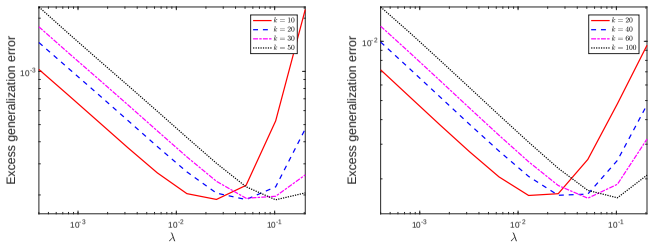
We first study how BKRR would behave along the iterations. In the first experiment, we aim to study how EGEs would change as a function of iteration numbers. We fix regularization parameters $\lambda \in \{0.0032, 0.0128, 0.0512, 0.2048\}$, and show in Figure 2 EGEs versus the iteration number for two regression problems. From Figure 2 we see that EGEs

would typically decrease first as k increases from 1 to some number, after which it increases slowly as a function of k. To be detailed, the increasing curve behaves as a concave function with respect to k, which validates our arguments in Theorem 3 and Theorem 4 that the variance increases with exponentially diminishing terms as k increases. Furthermore, for some large λ ($\lambda = 0.2048$ for example), BKRR shows a rather stable relationship between the generalization performance and iteration numbers. This is consistent with our theoretical findings in Theorem 3 and Theorem 4. We can also see clearly that the iteration number k at which EGEs achieve the minimal value would increase as λ increases.



(a) $f_\rho = g_1$. (b) $f_\rho = g_2$.

Figure 2: Excess generalization errors versus the number of iterations for different regularization parameters. We consider four λ and two regression problems with the regression function being g_1 and g_2 in panel (a) and panel (b), respectively.



(a) $f_\rho = g_1$. (b) $f_\rho = g_2$.

Figure 3: Excess generalization errors versus regularization parameters for different number of iterations. We consider four k and two regression problems with the regression function being g_1 and g_2 in panel (a) and panel (b), respectively.

We then illustrate the behavior of ITR with fixed k versus regularization parameters. We fix the iteration numbers to some predefined values, and report EGEs of $f_{D,\lambda}^{(k)}$ as a function of regularization parameters in Figure 3. According to Figure 3, we see that EGEs first decrease and then increase as a function of λ. As compared to the stable relationship between EGEs and the number of iterations for BKRR, EGEs change more rapidly with the regularization parameters for ITR. This gives empirical evidence that the selection of iteration number in BKRR is more tractable than the selection of regularization parameters in ITR. It is also clear that the optimal regularization parameter becomes larger as k increases, which is consistent with Theorem 1.

In Table 1, we record the optimal EGEs achieved by BKRR over all iterates for different regularization parameters. We list in the first row the considered regularization parameters. In the second and third rows, we report the optimal EGEs achieved by BKRR with the fixed regularization parameters on two regression problems. In Table 2, we record the optimal EGEs achieved by ITR over all regularization parameters for different iteration numbers. We list in the first and third rows the considered number of iterations. In the second and fourth rows, we report the optimal EGEs achieved by ITR with fixed iteration number on two regression problems. It can be found that BKRR can achieve similar accuracies to ITR with a much easier parameter-selection strategy.

λ	0.0004	0.0008	0.0016	0.0032	0.0064	0.0128	0.0256	0.0512	0.1024	0.2048
g_1	2.72e–4	2.26e–4	2.24e–4	1.92e–4	1.89e–4	1.87e–4	1.87e–4	1.87e–4	1.87e–4	1.87e–4
g_2	1.76e–3	1.75e–3	1.65e–3	1.60e–3	1.60e–3	1.60e–3	1.59e–3	1.59e–3	1.59e–3	1.59e–3

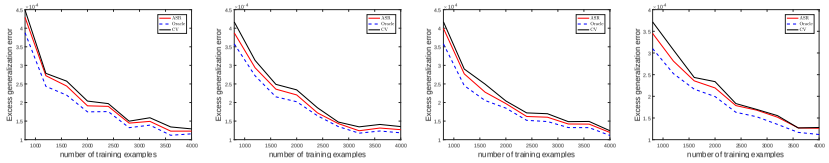
Table 1: Excess generalization errors of BKRR with different regularization parameters.

k	15	30	45	60	75	90	105	120	135	150
g_1	1.91e–4	1.92e–4	1.87e–4	1.92e–4	1.89e–4	1.87e–4	1.88e–4	1.92e–4	1.97e–4	2.02e–4
k	30	60	90	120	150	180	210	240	270	300
g_2	1.59e–3	1.59e–3	1.63e–3	1.59e–3	1.63e–3	1.62e–3	1.60e–3	1.59e–3	1.61e–3	1.63e–3

Table 2: Excess generalization errors of ITR with different iteration numbers.

6.3. BKRR with Early Stopping

In this subsection, we aim to validate the effectiveness of adaptive stopping rule (19) in practical regression problems for some $\theta \in \mathbb{R}_+$. It should be noted that (19) is independent of the confidence level and also different from (16) in the constant term. As discussed in Remark 7, such a modification is reasonable due to the (semi) exponential bias-variance trade-off of BKRR. We apply BKRR to regression problems with different sample sizes ($|D| \in \{800, 1200, 1600, 2000, 2400, 2800, 3200, 3600, 4000\}$) and different regularization parameters ($\lambda \in \{0.016, 0.032, 0.064, 0.128\}$). For each sample size and regularization parameter, we run BKRR with several iterations to get a sequence of candidate models. We record the iteration number $\hat{k}_{\text{ASR}}$ selected by the adaptive stopping rule (ASR) (19) with $\theta = 0.05$, the iteration number $\hat{k}_{\text{CV}}$ selected by the five-fold cross validation (CV) and the iteration number $\hat{k}_{\text{Oracle}}$ with the minimal generalization error over all candidate models.



(a) $\lambda = 0.016$ (b) $\lambda = 0.032$ (c) $\lambda = 0.064$ (d) $\lambda = 0.128$

Figure 4: Excess generalization errors of the model selected by ASR (red color), the model selected by CV (black color) and the best candidate model (blue color) versus the number of training examples. We consider the regression problem $y = g_1(x) + \epsilon$ and four regularization parameters: $\lambda = 0.016$, $\lambda = 0.032$, $\lambda = 0.064$ and $\lambda = 0.128$.

In Figure 4, we fix different regularization parameters and plot the EGEs of $f_{D,\lambda}^{\hat{k}_{\rm ASR}}$, $f_{D,\lambda}^{\hat{k}_{\rm CV}}$ and $f_{D,\lambda}^{\hat{k}_{\rm Oracle}}$ versus the number of training examples for the regression problem with the regression function g_1. According to Figure 4, it is clear that ASR (19) works well in selecting a good model with EGEs comparable to the best candidate model. Furthermore, ASR also behaves slightly better than the CV widely used in practical learning problems. It should be mentioned that the five-fold CV requires the training of an additional five models based on different assignments of validation sets, which can be time-consuming. This repeated training is not required in ASR and therefore ASR requires significantly less computational costs than CV.

Acknowledgments

The authors would like to thank two anonymous referees for their constructive suggestions. The work of Shao-Bo Lin is supported partially by the National Natural Science Foundation of China (Grant Numbers 61876133, 11771021). The work of Yunwen Lei is supported partially by the National Natural Science Foundation of China (Grant No. 61806091) and the Shenzhen Peacock Plan (Grant No. KQTD2016112514355531). The work of Ding-Xuan Zhou is supported partially by the Research Grants Council of Hong Kong [Project No. CityU 11303915] and by National Natural Science Foundation of China under Grant 11461161006. Part of the work was done when the last author visited Shanghai Jiaotong University (SJTU), for which the support from SJTU and the Ministry of Education is greatly appreciated. The corresponding author is Yunwen Lei.

Appendix A. Auxiliary Lemmas

In this appendix, we present some useful lemmas. The first one (Mücke, 2018, Corollary 2.2) (see also Lu et al. (2018)) describes the difference between the effective dimension and its empirical counterpart.

Lemma 21 *For any $0 < \delta < 1$, with confidence $1 - \delta$, there holds*

$$(1 + 4\eta_\delta)^{-1}\sqrt{\max\{\mathcal{N}(\lambda), 1\}} \le \sqrt{\max\{\mathcal{N}_D(\lambda), 1\}} \le (1 + 4\max\{\sqrt{\eta_\delta}, \eta_\delta^2\})\sqrt{\max\{\mathcal{N}(\lambda), 1\}},$$

where $\eta_\delta := 2\log(4/\delta)/\sqrt{|D|\lambda}$.

Based on Lemma 21, we can get the following lemma.

Lemma 22 *Let D be a sample drawn independently according to ρ and $0 < \delta < 1$. With confidence at least $1 - \delta$,*

$$\begin{aligned}
\mathcal{Q}_{D,\lambda}^2 &\le \widetilde{\mathcal{Q}}_{D,\lambda} \le 2\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}\log\frac{8}{\delta}}{\sqrt{\lambda}}\right)^2 + 2,\\
\mathcal{R}_D &\le \frac{4\kappa^2}{\sqrt{|D|}}\log\frac{8}{\delta},\\
\mathcal{P}_{D,\lambda} &\le 2(\kappa M + \gamma)\mathcal{A}_{D,\lambda}\log\big(8/\delta\big),\\
(1 + 4\eta_{\delta/4})^{-1}\sqrt{\max\{\mathcal{N}(\lambda), 1\}} &\le \sqrt{\max\{\mathcal{N}_D(\lambda), 1\}} \le (1 + 4\sqrt{\eta_{\delta/4}} \vee \eta_{\delta/4}^2)\sqrt{\max\{\mathcal{N}(\lambda), 1\}}
\end{aligned}$$

hold simultaneously.

Proof. From Guo et al. (2017, Proposition 1) and (67) in Appendix B below, there exists a subset $\mathcal{Z}_{\delta,1}^{|D|}$ of $\mathcal{Z}^{|D|}$ with measure at least $1 - \delta$ such that for all $D \in \mathcal{Z}_{\delta,1}^{|D|}$

$$\mathcal{Q}_{D,\lambda}^2 \le \widetilde{\mathcal{Q}}_{D,\lambda} \le 2\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}\log\frac{2}{\delta}}{\sqrt{\lambda}}\right)^2 + 2.$$

From Yao et al. (2007, Proposition 5.3), there exists a subset $\mathcal{Z}_{\delta,2}^{|D|}$ of $\mathcal{Z}^{|D|}$ with measure at least $1 - \delta$ such that for all $D \in \mathcal{Z}_{\delta,2}^{|D|}$

$$\mathcal{R}_D \le \frac{4\kappa^2}{\sqrt{|D|}}\log\frac{2}{\delta}.$$

It also follows from Blanchard and Krämer (2016, Lemma 5.1) that there exists a subset $\mathcal{Z}_{\delta,3}^{|D|}$ of $\mathcal{Z}^{|D|}$ with measure at least $1 - \delta$ such that for all $D \in \mathcal{Z}_{\delta,3}^{|D|}$

$$\mathcal{P}_{D,\lambda} \le 2(\kappa M + \gamma)\mathcal{A}_{D,\lambda}\log\big(2/\delta\big).$$

According to Lemma 21, there exists a subset $\mathcal{Z}_{\delta,4}^{|D|}$ of $\mathcal{Z}^{|D|}$ with measure at least $1 - \delta$ such that for all $D \in \mathcal{Z}_{\delta,4}^{|D|}$

$$(1 + 4\eta_\delta)^{-1}\sqrt{\max\{\mathcal{N}(\lambda), 1\}} \le \sqrt{\max\{\mathcal{N}_D(\lambda), 1\}} \le (1 + 4\sqrt{\eta_\delta} \vee \eta_\delta^2)\sqrt{\max\{\mathcal{N}(\lambda), 1\}}.$$

Thus, for $D \in \mathcal{Z}_{\delta,1}^{|D|} \cap \mathcal{Z}_{\delta,2}^{|D|} \cap \mathcal{Z}_{\delta,3}^{|D|} \cap \mathcal{Z}_{\delta,4}^{|D|}$, the above four inequalities hold simultaneously. Then Lemma 22 follows by scaling δ to $\delta/4$. ∎

From Lemma 22, we derive the following estimate.

Lemma 23 *For any $0 < \delta < 1$, with confidence $1-\delta$, there holds*

$$\mathcal{P}_{D,\lambda}\mathcal{Q}_{D\lambda} \le \frac{1}{2}\mathcal{W}_{D,\lambda}\log^4\frac{16}{\delta}.$$

Proof. It follows from Lemma 22 and (48) that with confidence $1-\delta$, there holds

$$\begin{aligned}\mathcal{P}_{D,\lambda}\mathcal{Q}_{D\lambda} &\le \frac{4\sqrt{2}(\kappa^2+\kappa+1)(\kappa M+\gamma)}{\sqrt{|D|}}\left(\frac{1}{\lambda|D|}+\frac{\sqrt{\mathcal{N}(\lambda)}}{\sqrt{|D|\lambda}}+1\right)\left(\frac{1}{\sqrt{\lambda|D|}}+\sqrt{\mathcal{N}(\lambda)}\right)\log^2\frac{16}{\delta}\\ &\le \frac{4\sqrt{2}(\kappa^2+\kappa+1)(\kappa M+\gamma)}{\sqrt{|D|}}\left(\frac{(\sqrt{|D|\lambda}+9)\sqrt{\max\{\mathcal{N}_D(\lambda),1\}}}{|D|\lambda}+1\right)\\ &\times\frac{(\sqrt{|D|\lambda}+9)\sqrt{\max\{\mathcal{N}_D(\lambda),1\}}}{\sqrt{|D|\lambda}}\log^4\frac{16}{\delta}.\end{aligned}$$

This together with (15) completes the proof of Lemma 23. ∎

Our final lemma establishes a relation between the in-sample norm and out-sample norm of functions in $\mathcal{H}_K$.

Lemma 24 *Let $f \in \mathcal{H}_K$. Then*

$$\|f\|_\rho \le \mathcal{Q}_{D,\lambda}\|L_{K,D}^{1/2}f\|_K + \mathcal{Q}_{D,\lambda}\lambda^{1/2}\|f\|_K. \tag{66}$$

Proof. Since $f \in \mathcal{H}_K$, it follows from the definition of $\mathcal{Q}_{D,\lambda}$ that

$$\begin{aligned}\|f\|_\rho &= \|L_K^{1/2}f\|_K \le \|(L_K+\lambda I)^{1/2}f\|_K \le \mathcal{Q}_{D,\lambda}\|(L_{K,D}+\lambda I)^{1/2}f\|_K\\ &\le \mathcal{Q}_{D,\lambda}\|L_{K,D}^{1/2}f\|_K + \mathcal{Q}_{D,\lambda}\lambda^{1/2}\|f\|_K.\end{aligned}$$

This finishes the proof of Lemma 24. ∎

Appendix B. Some Inequalities for Positive Linear Operators

In this part, we recall some basic definitions and properties for linear operators which can be found in Bhatia (2013). Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be two Hilbert spaces. Let $\mathcal{L}(\mathcal{H}_1,\mathcal{H}_2)$ be the space of all bounded linear operators from $\mathcal{H}_1$ to $\mathcal{H}_2$. The adjoint of an operator A is the unique operator A^* in $\mathcal{L}(\mathcal{H}_2,\mathcal{H}_1)$ that satisfies the relation

$$\langle g, Af\rangle_{\mathcal{H}_2} = \langle A^*g, f\rangle_{\mathcal{H}_1}$$

for all $f \in \mathcal{H}_1$ and $g \in \mathcal{H}_2$. For the space $\mathcal{L}(\mathcal{H},\mathcal{H})$, we use the more compact notation $\mathcal{L}(\mathcal{H})$. For $A \in \mathcal{L}(\mathcal{H})$, if $A = A^*$, we then call A a self-adjoint operator. A self-adjoint operator is said to be positive, if $\langle f, Af\rangle_{\mathcal{H}} \ge 0$ for all $f \in \mathcal{H}$. If $\langle f, Af\rangle_{\mathcal{H}} > 0$ for all nonzero f, we say A is strictly positive.

For $A \in \mathcal{L}(\mathcal{H})$, the operator norm is defined by

$$\|A\| = \sup_{\|f\|_{\mathcal{H}}=1}\|Af\|_{\mathcal{H}}.$$

If A is compact and positive, there exists a normalized eigenpairs of A, denoted by $\{(\lambda_i,\varphi_i)\}_{i=1}^\infty$, with eigenvectors $\{\varphi_i\}$ forming an orthonormal basis for $\mathcal{H}$ and eigenvalues satisfying $\lambda_1 \ge \lambda_2 \ge \cdots \ge 0$. There also holds $\|A\| = \lambda_1$. Write the trace of the positive operator A by

$$\mathrm{Tr}(A) = \sum_{i=1}^\infty \lambda_i.$$

The Hilbert-Schmidt norm of A is then defined by

$$\|A\|_{HS} = (\mathrm{Tr}(A^2))^{1/2} = \left(\sum_{j=1}^\infty \lambda_j^2\right)^{1/2}.$$

If $\|A\|_{HS} < \infty$, we then call A a Hilbert-Schmidt operator. If A is Hilbert-Schmidt, it follows from the definition that $\|A\| \le \|A\|_{HS}$. For $F : \mathbf{R}_+ \cap \{0\} \to \mathbf{R}$, we define the operator

$$F(A) = \sum_{i=1}^\infty F(\lambda_i)\varphi_i\otimes\varphi_i = \sum_{i=1}^\infty F(\lambda_i)\langle\cdot,\varphi_i\rangle_{\mathcal{H}}\varphi_i$$

by spectral calculus. For positive operators A and B, there holds $\|AB\| = \|BA\|$. We also need the following two important inequalities, which can be found in Lemma 1 and Lemma 4 in Guo et al. (2017) (see also Bhatia (2013, Lemma VII.5.5) for the second one).

Lemma 25 *Let A and B be positive operators. Then for any $0 \le \tau \le 1$, there holds*

$$\|A^\tau B^\tau\| \le \|AB\|^\tau. \tag{67}$$

If in addition A and B are Hilbert-Schmidt and $\max\{\|A\|,\|B\|\} \le \kappa$, then for arbitrary $\mu \ge 1$, there holds

$$\|A^\mu - B^\mu\|_{HS} \le \mu\kappa^{\mu-1}\|A-B\|_{HS}. \tag{68}$$

References

F. Bauer, S. Pereverzev, and L. Rosasco. On regularization algorithms in learning theory. *J. Complex.*, 23(1):52–72, 2007.

R. Bhatia. *Matrix Analysis*. Vol. 169, Springer Science & Business Media, 2013.

G. Blanchard and N. Krämer. Convergence rates of kernel conjugate gradient for random design regression. *Anal. Appl.*, 14(06):763–794, 2016.

P. Bühlmann and B. Yu. Boosting with the l_2 loss: regression and classification. *J. Amer. Satis. Assoc.*, 98(462):324–339, 2003.

A. Caponnetto and E. De Vito. Optimal rates for the regularized least-squares algorithm. *Found. Comp. Math.*, 7(3):331–368, 2007.

A. Caponnetto and Y. Yao. Cross-validation based adaptation for regularization operators in learning theory. *Anal. Appl.*, 8(02):161–183, 2010.

X. Chang, S. B. Lin, and D. X. Zhou. Distributed semi-supervised learning with kernel ridge regression. *J. Mach. Learn. Res.*, 18(46): 1-22, 2017.

F. Cucker and D. X. Zhou. *Learning Theory: An Approximation Theory Viewpoint*. Cambridge University Press, 2007.

E. De Vito, V. Umanità, and S. Villa. An extension of Mercer theorem to vector-valued measurable kernels. *Appl. Comput. Harmonic Anal.* 34:339-351, 2013.

H. W. Engl. On the choice of the regularization parameter for iterated tikhonov regularization of ill-posed problems. *J. Approx. Theory*, 49(1):55–63, 1987.

T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Adv. Comput. Math.*, 13(1):1–50, 2000.

Y. Freund. Boosting a weak learning algorithm by majority. *Inform. & Comput.*, 121(2): 256–285, 1995.

J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Ann. Stat.*, 29(5):1189–1232, 2001.

J. H. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Ann. Stat.*, 28(2):337–407, 2000.

L. L. Gerfo, L. Rosasco, F. Odone, E. De Vito, and A. Verri. Spectral algorithms for supervised learning. *Neural Comput.*, 20(7):1873–1897, 2008.

Z. C. Guo, S. B. Lin, and D. X. Zhou. Learning theory of distributed spectral algorithms. *Inverse Probl.*, 33(7):074009, 2017.

Z.-C. Guo, D. H. Xiang, X. Guo, and D. X. Zhou. Thresholded spectral algorithms for sparse approximations. *Anal. Appl.*, 15(03):433–455, 2017b.

L. Györfi, M. Kohler, A. Krzyzak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer, Berlin, 2002.

M. Hanke and C. W. Groetsch. Nonstationary iterated tikhonov regularization. *J. Optimiz. Theory Appl.*, 98(1):37–53, 1998.

Q. Jin and L. Stals. Nonstationary iterated tikhonov regularization for ill-posed problems in banach spaces. *Inverse Probl.*, 28(10):104011, 2012.

Q. N. Jin and Z. Y. Hou. On the choice of the regularization parameter for ordinary and iterated tikhonov regularization of nonlinear ill-posed problems. *Inverse Probl.*, 13(3): 815, 1997.

J. T. King and D. Chillingworth. Approximation of generalized inverses by iterated regularization. *Numer. Funct. Anal. Optim.*, 1(5):499–513, 1979.

S. B. Lin and D. X. Zhou. Distributed kernel-based gradient descent algorithms. *Constr. Approx.*, 47(2):249–276, 2018.

S. B. Lin and D. X. Zhou. Optimal learning rates for kernel partial least squares. *J. Fourier Anal. Appl.*, 24(3):908–933, 2018.

S. B. Lin, X. Guo, and D. X. Zhou. Distributed learning with regularized least squares. *J. Mach. Learn. Res.*, 18(92):1–31, 2017.

S. Lu, P. Mathé, and S. Pereverzyev. Balancing principle in supervised learning for a general regularization scheme. *Appl. Comput. Harmon. Anal.*, In Press. 2018

N. Mücke. Adaptivity for Regularized Kernel Methods by Lepskii's Principle. arXiv preprint arXiv:1804.05433, 2018.

B. U. Park, Y. K. Lee, and S. Ha. L_2 boosting in kernel regression. *Bernoulli*, 15(3): 599–613, 2009.

G. Raskutti, M. Wainwright, and B. Yu. Early stopping and non-parametric regression: an optimal data-dependent stopping rule. *J. Mach. Learn. Res.*, 15(1):335–366, 2014.

R. E. Schapire. The strength of weak learnability. *Mach. Learn.*, 5(2):197–227, 1990.

L. Shi, Y. L. Feng, and D. X. Zhou. Concentration estimates for learning with l_1-regularizer and data dependent hypothesis spaces. *Appl. Comput. Harmonic Anal.*, 31(2):286–302, 2011.

S. Smale and D. X. Zhou. Shannon sampling and function reconstruction from point values. *Bull. Amer. Math. Soc.*, 41(3):279–305, 2004.

S. Smale and D. X. Zhou. Learning theory estimates via integral operators and their approximations. *Constr. Approx.*, 26(2):153–172, 2007.

I. Steinwart, D. Hush, and C. Scovel. Optimal rates for regularized least squares regression. In S. Dasgupta and A. Klivan, editors, *Annual Conference on Learning Theory*, pages 79–93, 2009.

I. Steinwart and C. Scovel. Mercer's theorem on general domains: On the interaction between measures, kernels, and RKHSs. *Constr. Approx.*, 35(3):363-417, 2012.

Q. Wu. Bias corrected regularization kernel network and its applications. In *International Joint Conference on Neural Networks*, pages 1072–1079, 2017.

Y. Yao, L. Rosasco, and A. Caponnetto. On early stopping in gradient descent learning. *Constr. Approx.*, 26(2):289–315, 2007.

Y. Ying and D. X. Zhou. Unregularized online learning algorithms with general loss functions. *Appl. Comput. Harmonic Anal.*, 2(42):224–244, 2017.

Journal of Machine Learning Research 20 (2019) 1-47 Submitted 3/18; Revised 12/18; Published 2/19

Analysis of spectral clustering algorithms for community detection: the general bipartite setting

Zhixin Zhou ZHIXIN@G.UCLA.EDU
Department of Statistics
University of California
Los Angeles, CA 90095-1554, USA

Arash A. Amini AAAMINI@UCLA.EDU
Department of Statistics
University of California
Los Angeles, CA 90095-1554, USA

Editor: Ulrike von Luxburg

Abstract

We consider spectral clustering algorithms for community detection under a general bipartite stochastic block model (SBM). A modern spectral clustering algorithm consists of three steps: (1) regularization of an appropriate adjacency or Laplacian matrix (2) a form of spectral truncation and (3) a k-means type algorithm in the reduced spectral domain. We focus on the adjacency-based spectral clustering and for the first step, propose a new data-driven regularization that can restore the concentration of the adjacency matrix even for the sparse networks. This result is based on recent work on regularization of random binary matrices, but avoids using unknown population level parameters, and instead estimates the necessary quantities from the data. We also propose and study a novel variation of the spectral truncation step and show how this variation changes the nature of the misclassification rate in a general SBM. We then show how the consistency results can be extended to models beyond SBMs, such as inhomogeneous random graph models with approximate clusters, including a graphon clustering problem, as well as general sub-Gaussian biclustering. A theme of the paper is providing a better understanding of the analysis of spectral methods for community detection and establishing consistency results, under fairly general clustering models and for a wide regime of degree growths, including sparse cases where the average expected degree grows arbitrarily slowly.

Keywords: Spectral clustering; bipartite networks; stochastic block model; regularization of random graphs; community detection; sub-Gaussian biclustering; graphon clustering.

1. Introduction

Spectral clustering is one of the most popular and successful approaches to clustering and has appeared in various contexts in statistics and computer science among other disciplines. The idea generally applies when one can define a similarity matrix between pairs of objects to be clustered (Ng et al., 2002; Von Luxburg, 2007; Von Luxburg et al., 2008). Recently, there has been a resurgence of interest in the spectral approaches and their analysis in the context of network clustering (Boppana, 1987; McSherry, 2001; Dasgupta et al., 2006; Coja-Oghlan, 2010; Tomozei and Massoulié, 2010; Rohe et al., 2011; Balakrishnan et al., 2011; Chaudhuri et al., 2012; Fishkind et al., 2013; Qin and Rohe, 2013; Joseph and Yu, 2013; Krzakala et al., 2013; Lei et al., 2015; Yun and Proutiere, 2014a,b; Binkiewicz et al., 2014; Lyzinski et al., 2014; Chin et al., 2015; Gao et al., 2017). The interest has been partly fueled by the recent activity in understanding statistical network models for clustering and community detection, and in particular by the flurry of theoretical work on the stochastic block model (SBM)—also called the planted partition model—and its variants (Abbe, 2017).

There has been significant recent advances in analyzing spectral clustering approaches in SBMs. We start by identifying the main components of the analysis in Section 3, and then show how variations can be introduced at each step to obtain improved consistency results and novel algorithms. We will work in the general bipartite setting which has received comparatively less attention, but most results in the paper can be easily extended to the (symmetric) unipartite models (cf. Remark 7). It is worth noting that clustering bipartite SBMs is closely related to the biclustering (Hartigan, 1972) and co-clustering (Dhillon, 2001; Rohe and Yu, 2012) problems.

A modern spectral clustering algorithm often consists of three steps: (a) the regularization and concentration of the adjacency matrix (or the Laplacian) (b) the spectral truncation step, and (c) the k-means step. By using variations in each step one obtains different spectral algorithms, which is then reflected in the variations in the consistency results.

The regularization step is fairly recent and is motivated by the observation that proper regularization significantly improves the performance of spectral methods in sparse networks (Chaudhuri et al., 2012; Amini et al., 2013; Joseph and Yu, 2013; Le et al., 2015; Chin et al., 2015). In particular, regularization restores the concentration of the adjacency matrix (or the Laplacian) around its expectation in the *sparse regime*, where the average degree of the network is constant or grows very slowly with the number of nodes. In this paper, building on these recent advances, we introduce a novel regularization scheme for the adjacency matrix that is fully data-driven and avoids relying on unknown quantities such as maximum expected degrees (for rows and columns) of the network. This regularization scheme is introduced as Algorithm 1 in Section 4 and we show that under a general SBM it achieves the same concentration bound (Theorem 3) as its oracle counterpart.

For the spectral truncation step, we will consider three variations, one of which (Algorithm 2) is the common approach of keeping the top k leading eigenvectors as columns of the matrix passed to the k-means step. In this traditional approach, the spectral truncation can be viewed as obtaining a low-dimensional representation of the data, suitable for an application of simple k-means type algorithms. We also consider a recent variant (Algorithm 3) proposed in Yun and Proutiere (2014b) and Gao et al. (2017), in which the spectral truncation step acts more as a denoising step. We then propose a third alternative (Algorithm 4 in Section 5) which combines the benefits of both approaches while improving the computational complexity. One of our novel contributions is to derive consistency results for Algorithms 3 and 4, under a general SBM, showing that the behavior of the two algorithms is the same (but different than Algorithm 2) assuming that the k-means step satisfies a property we refer to as *isometry invariance* (Theorems 5 and 6).

In the final step of spectral clustering, one runs a k-means type algorithm on the output of the truncation step. We discuss this step in some detail since it is often mentioned

©2019 Zhixin Zhou and Arash A. Amini.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-170.html.

briefly in the analyses of spectral clustering, with the exception of a few recent works (Lei et al., 2015; Gao et al., 2017, 2018). By the k-means step, we do not necessarily mean the solution of the well-known k-means problem, although, this step is usually implemented by an algorithm that approximately minimizes the k-means objective. We will consider the k-means step in some generality, by introducing the notation of a *k-means matrix* (cf. Section 3.3). The goal of the final step of spectral clustering is to obtain a k-means matrix which is close to the output of the truncation step. We characterize sufficient conditions on this approximation so that the overall algorithm produces consistent clustering. Any approach that satisfies these conditions can be used in the k-means step, even if it is not an approximate k-means solver.

Most of the above ideas extend beyond SBMs and, in Section 6, we consider various extensions. We first consider some unique aspects of the bipartite setting, for example, the possibility of having clusters only on one side of the network, or having more clusters on one side than the rank of the expected adjacency matrix. We then show how the results extend to general sub-Gaussian biclustering (Section 6.3) and general inhomogeneous random graphs with approximate clusters (Section 6.4).

The organization of the rest of the paper is as follows: After introducing some notation in Section 1.1, we discuss the general (bipartite) SBM in Section 2. An outline of the analysis is given in Section 3, which also provides the high-level intuition of why the spectral clustering works in SBMs and what each step will achieve in terms of guarantees on its output. This section provides a typical blueprint theorem on consistency (Theorem 1 in Section 3.1) which serves as a prelude to later consistency results. The rest of Section 3 provides the details of the last two steps of the analysis sketched in Section 3.1. The regularization and concentration (the first step) is detailed in Section 4 where we also introduce our data-driven regularization. We then give explicit algorithms and their corresponding consistency results in Section 5. Extensions of the results are discussed in Section 6. Some simulations showing the effectiveness of the regularization are provided in Section 7.

Related work. There are numerous papers discussing aspect of spectral clustering and its analysis. Our paper is mostly inspired by recent developments in the field, especially by the consistency results of Lei et al. (2015), Yun and Proutiere (2014a), Chin et al. (2015), Gao et al. (2017) and concentration results for the regularized adjacency (and Laplacian) matrices such as Le et al. (2015) and Bandeira et al. (2016). Theorem 4 on the consistency of the typical adjacency-based spectral clustering algorithm—which we will call SC-1—is generally known (Lei et al., 2015; Gao et al., 2017), though our version is slightly more general; see Remark 5. The spectral algorithm SC-RR is proposed and analyzed in Yun and Proutiere (2014b), Gao et al. (2017) and Gao et al. (2018) for special cases of the SBM (and its degree-corrected version); the new consistency result we give for SC-RR is for the general SBM and reveals the contrast with SC-1. Previous analyses did not reveal this difference due to focusing on the special case; see Examples 1 and 2 in Section 5 for details. We also propose the new SC-RRE which has the same performance as SC-RR (assuming a proper k-means step) but is much more computationally efficient. The results that we prove for SC-RR and SC-RRE can be recast in terms of the *mean parameters* of the SBM (in contrast to SC-1), as demonstrated in Corollary 4 of Section 5.3. For an application of

this result, we refer to our work on optimal bipartite network clustering (Zhou and Amini, 2018).

1.1. Notation

Orthogonal matrices. We write $\mathbb{O}^{n\times k}$ for the set of $n\times k$ matrices with orthonormal columns. The condition $k\le n$ is implicit in defining $\mathbb{O}^{n\times k}$. The case $\mathbb{O}^{n\times n}$ is the set of orthogonal matrices, though with some abuse of terminology we also refer to matrices in $\mathbb{O}^{n\times k}$ as orthogonal even if $k<n$. Thus, $Z\in\mathbb{O}^{n\times k}$ iff $Z^TZ=I_k$. We also note that $Z\in\mathbb{O}^{n\times k_1}$ and $U\in\mathbb{O}^{k_1\times k}$ implies $ZU\in\mathbb{O}^{n\times k}$. The following holds:

$$\|Ux\|_2=\|x\|_2,\quad \forall x\in\mathbb{R}^k,\ U\in\mathbb{O}^{k_1\times k}, \tag{1}$$

for any $k_1\ge k$. On the other hand,

$$\|U^Tx\|_2\le\|x\|_2,\quad \forall x\in\mathbb{R}^{k_1},\ U\in\mathbb{O}^{k_1\times k}, \tag{2}$$

where equality holds for all $x\in\mathbb{R}^{k_1}$, iff $k_1=k$. To see (2), let $u_1,\dots,u_k\in\mathbb{R}^{k_1}$ be the columns of U, constituting an orthonormal sequence which can be completed to an orthonormal basis by adding say $u_{k+1},\dots,u_{k_1}$. Then, $\|U^Tx\|_2^2=\sum_{j=1}^k\langle u_j,x\rangle^2\le\sum_{j=1}^{k_1}\langle u_j,x\rangle^2=\|x\|_2^2$.

Membership matrices and misclassification. We let $\mathbb{H}^{n\times k}$ denote the set of *hard* cluster labels: $\{0,1\}$-valued $n\times k$ matrices where each row has exactly a single 1. A matrix $Z\in\mathbb{H}^{n\times k}$ is also called a membership matrix, where row i is interpreted as the membership of node i to one of k clusters (or communities). Here we implicitly assume that we have a network on nodes in $[n]=\{1,\dots,n\}$, and there is a latent partition of $[n]$ into k clusters. In this sense, $Z_{ik}=1$ iff node i belongs to cluster k. Given, two membership matrices $Z,Z'\in\mathbb{H}^{n\times k}$, we can consider the average misclassification rate between them, which we denote as $\overline{\mathrm{Mis}}(Z,Z')$: Letting z_i^T and $(z_i')^T$ denote the ith row of Z and Z' respectively, we have

$$\overline{\mathrm{Mis}}(Z,Z'):=\min_Q\frac{1}{n}\sum_{i=1}^n 1\{z_i\ne Qz_i'\} \tag{3}$$

where the minimum is taken over $k\times k$ permutations matrices Q. We also let $\mathrm{Mis}_r(Z,Z')$ be the misclassification rate between the two, over the rth cluster of Z, that is, $\mathrm{Mis}_r(Z,Z')=\frac{1}{n_r}\sum_{i:\ z_i=r}1\{z_i\ne Q^*z_i'\}$ where $n_r=\sum_{i=1}^n1\{z_i=r\}$ is the size of the rth cluster of Z, and Q^* is the optimal permutation in (3). Note that in contrast to $\overline{\mathrm{Mis}}$, Mis_r is not symmetric in its two arguments. We also write $\mathrm{Mis}_\infty:=\max_r\mathrm{Mis}_r$. These definitions can be extended to misclassification rates between k-means matrices introduced in Section 3.3.

2. Stochastic Block Model

We consider the general, not necessarily symmetric, Stochastic Block Model (SBM) with bi-adjacency matrix $A\in\{0,1\}^{n_1\times n_2}$. We assume throughout that $n_2\ge n_1$, without loss of generality. We have membership matrices $Z_r\in\mathbb{H}^{n_r\times k_r}$ for each of the two sides $r=1,2$,

where $k_r \le n_r$ denotes the number of communities on side r. Each element of A is an independent draw from a Bernoulli variable, and

$$P := \mathbb{E}[A] = Z_1 B Z_2^T, \quad B = \frac{\Psi}{\sqrt{n_1 n_2}} \tag{4}$$

where $B \in [0,1]^{k_1 \times k_2}$ is the connectivity— or the edge probability—matrix, and Ψ is a rescaled version. We also use the notation

$$A \sim \mathrm{Ber}(P) \iff A_{ij} \sim \mathrm{Ber}(P_{ij}), \text{ independent across } (i,j) \in [n_1] \times [n_2]. \tag{5}$$

Classical SBM which we refer to as *symmetric SBM* corresponds to the following modifications:

(a) A is assumed to be symmetric: Only the upper diagonal elements are drawn independently and the bottom half is filled symmetrically. For simplicity, we allow for self-loops, i.e., draw the diagonal elements from the same model. This will have a negligible effect in the arguments.

(b) $n_1 = n_2 = n,\ k_1 = k_2 = k,\ Z_1 = Z_2 = Z$.

(c) B is assumed symmetric.

We note that (4) still holds over all the elements. Directed SBM is also a special case, where (b) is assumed but not (a) or (c). That is, A is not assumed to be symmetric and all the entries are independently drawn, while B may or may not be symmetric.

We refer to P as the *mean matrix* and note that it is of rank at most $k := \min\{k_1, k_2\}$. Often $k \ll n_1, n_2$, that is P is a low-rank matrix which is the key in why spectral clustering works well for SBMs. However, the case where either $k_1 \gtrsim n_1$ or $k_2 \gtrsim n_2$ is allowed. (Here, $k_1 \gtrsim n_1$ means $k_1 \ge c n_1$ for some universal constant $c > 0$.) An extreme example of such setup can be found in Section 6.1.

We let $N_r = \mathrm{diag}(n_{r1}, \dots, n_{rk_r})$ for $r = 1, 2$ where n_{rj} is the size of the jth cluster of Z_r; that is, N_r is a diagonal matrix whose diagonal elements are the sizes of the clusters on side r. We also consider the normalized version of N_r,

$$\bar{N}_r := N_r / n_r = \mathrm{diag}(\pi_{r1}, \dots, \pi_{rk_r}), \tag{6}$$

collecting the cluster proportions $\pi_{rj} := n_{rj}/n_r$. Let us define

$$\bar{B} := N_1^{1/2} B N_2^{1/2} = \bar{N}_1^{1/2} \Psi \bar{N}_2^{1/2}.$$

For sparse graphs, we expect $\bar{N}_r$ and Ψ to remain stable as $n_r \to \infty$, hence $\bar{B}$ remains stable; in contrast, the entries of B itself vanish under scaling (4). See Remark 2 below for details. Throughout the paper, barred parameters refer to quantities that remain stable or slowly vary with $n_r \to \infty$. The following lemma is key in understanding spectral clustering for SBMs:

Lemma 1 *Assume that $\bar{B} = U_\psi \Sigma V_\psi^T$ is the reduced SVD of $\bar{B}$, where $U_\psi \in \mathbb{O}^{k_1 \times k}, V_\psi \in \mathbb{O}^{k_2 \times k}$, $\Sigma = \mathrm{diag}(\sigma_1, \dots, \sigma_k)$, and $k = \min\{k_1, k_2\}$. Then,*

$$P = (\bar{Z}_1 U_\psi)\, \Sigma\, (\bar{Z}_2 V_\psi)^T \tag{7}$$

is the reduced SVD of P where $\bar{Z}_r := Z_r N_r^{-1/2}$ is itself an orthogonal matrix, $\bar{Z}_r^T \bar{Z}_r = I_{k_r}$.

Proof We first show that $Z_r^T Z_r = N_r$ which then implies that $\bar{Z}_r$ is orthogonal. Let z_{ri}^T be the ith row of Z_r and note that $Z_r^T Z_r = \sum_{i=1}^n z_{ri} z_{ri}^T$. Since $z_{ri} \in \mathbb{H}^{1 \times k_r}$, each $z_{ri} z_{ri}^T$ is a diagonal matrix with a single 1 on the diagonal at the position determined by the cluster assignment of node i on side r. Now, a little algebra on (4) shows that $P = \bar{Z}_1 \bar{B} \bar{Z}_2^T$, hence (7) holds. Since $\bar{Z}_r \in \mathbb{O}^{n_r \times k_r}$, we have $\bar{Z}_1 U_\psi \in \mathbb{O}^{n_1 \times k}$ and $\bar{Z}_2 V_\psi \in \mathbb{O}^{n_2 \times k}$ showing that (7) is in fact a (reduced) SVD of P. ■

When dealing with the symmetric SBM, we will drop the subscript r from all the relevant quantities; for example, we write $N = N_1 = N_2$, $\bar{Z} = \bar{Z}_1 = \bar{Z}_2$, $\pi_j = \pi_{1j} = \pi_{2j}$, and so on.

Remark 1 (Reduced versus truncated) *The term* reduced SVD *in Lemma 1 (also known as compact SVD) means that we reduce the orthogonal matrices in a full SVD by removing the columns corresponding to zero singular values. The number of columns of the resulting matrices will be equal to the rank of the underlying matrix (i.e., both $\bar{Z}_1 U_\psi$ and $\bar{Z}_2 V_\psi$ will have $k = \min\{k_1, k_2\}$ columns in the case of P). Hence, a reduced SVD is still an exact SVD. Later, we will use the term* truncated SVD *to refer to an "approximation" of the original matrix by a lower rank matrix obtained by further removing columns corresponding to small nonzero singular values (starting from a reduced SVD). Hence, a truncated SVD is only an approximation of the original matrix.*

Remark 2 (Scaling and sparsity) *As can be seen from the above discussion, the normalization in (4) is natural for studying spectral clustering. In the symmetric case, where $n_1 = n_2 = n$, the normalization reduces to $B = \Psi/n$, which is often assumed when studying sparse SBMs by requiring that either $\|\Psi\|_\infty$ is $O(1)$ or grows slowly with n. To see why this implies a sparse network, note that the expected average degree of the symmetric SBM (under this scaling) is*

$$\frac{1}{n} 1^T P 1 = \frac{1}{n} 1^T N B N 1 = 1^T \bar{N} \Psi \bar{N} 1 = \sum_{i,j=1}^k \pi_i \pi_j \Psi_{ij} =: d_{av}$$

using $1_n^T Z = 1_k^T N$. (Here and elsewhere, 1 is the vector of all ones of an appropriate dimension; we write 1_n if we want to emphasize the dimension n.) Thus, the growth of the average expected degree, d_{av}, is the same as Ψ, and as long as Ψ is $O(1)$ or grows very slowly with n, the network is sparse. Alternatively, we can view the expected density of the network (the expected number of edges divided by the total number of possible edges) as a measure of sparsity. For the symmetric case, the expected density is $(\frac{1}{2} 1^T P 1)/\binom{n}{2} \sim d_{av}/n$ and is $O(n^{-1})$ if $d_{av} = O(1)$. Similar observations hold in the general bipartite case if we

let $n = \sqrt{n_1 n_2}$, the geometric mean of the dimensions. The expected density of the bipartite network under the scaling of (4) *is*

$$\frac{1^T P 1}{n^2} = \frac{1}{n^2} 1^T N_1 B N_2 1 = \frac{1}{n} 1^T \bar{N}_1 \Psi \bar{N}_2 1 = \frac{d_{av}}{n}, \quad (n = \sqrt{n_1 n_2})$$

where $d_{av} := 1^T \bar{N}_1 \Psi \bar{N}_2 1 = \sum_{i,j} \pi_{1i} \pi_{2j} \Psi_{ij}$ can be thought of as the analog of the expected average degree in the bipartite case. As long as $\|\Psi\|_\infty$ grows slowly relative to $n = \sqrt{n_1 n_2}$, the bipartite network is sparse.

3. Analysis steps

Throughout, we focus on recovering the row clusters. Everything that we discuss goes through, with obvious modifications, for recovering the column clusters. Recalling the decomposition (7), the idea of spectral clustering in the context of SBMs is that $\bar{Z}_1 U_\psi$ has enough information for recovering the clusters and can be obtained by computing a reduced SVD of P. In particular, applying a k-means type clustering on the rows of $\bar{Z}_1 U_\psi$ should recover the cluster labels. On the other hand, the actual random adjacency matrix, A, is concentrated around the mean matrix P, after proper regularization if need be. We denote this potentially *regularized version as* A_{re}. Then, by the spectral perturbation theory, if we compute a reduced SVD of $A_{\rm re} = \hat{Z}_1 \hat{\Sigma} \hat{Z}_2^T$ where $\hat{Z}_r \in \mathbb{O}^{n_r \times k}$, $r = 1, 2$ and $\hat{\Sigma}$ is diagonal, we can conclude that $\hat{Z}_1$ concentrates around $\bar{Z}_1 U_\psi$. Hence, applying a continuous k-means algorithm on $\hat{Z}_1$ should be able to recover the labels with a small error.

3.1. Analysis sketch

Let us sketch the argument above in more details. A typical approach in proving consistency of spectral clustering consists of the following steps:

1. We replace A with a properly regularized version $A_{\rm re}$. We provide the details for one such regularization in Theorem 2 (Section 4). However, the only property we require of the regularized version is that it concentrates, with high probability, around the mean of A, at the following rate (assuming $n_2 \geq n_1$):

$$\|A_{\rm re} - \mathbb{E}[A]\|_{\rm op} \leq C\sqrt{d}, \quad \text{where} \quad d = \sqrt{\frac{n_2}{n_1}} \|\Psi\|_\infty. \tag{8}$$

Here and throughout $\|\cdot\|_{\rm op}$ is the $\ell_2 \to \ell_2$ operator norm and $\|\Psi\|_\infty = \max_{ij} \Psi_{ij}$.

2. We pass from $A_{\rm re}$ and $P = \mathbb{E}[A]$ to their (symmetrically) dilated versions $A_{\rm re}^\dagger$ and $P^\dagger$. The symmetric dilation operator will be given in (13) (Section 3.2) and allows us to use spectral perturbation bounds for symmetric matrices. A typical final result of this step is

$$\|\hat{Z}_1 - \bar{Z}_1 U_\psi Q\|_F \leq \frac{C_2}{\sigma_k} \sqrt{kd}, \quad \text{w.h.p.} \tag{9}$$

for some $Q \in \mathbb{O}^{k \times k}$. We recall that $\|\cdot\|_F$ is the Frobenius norm. Here, σ_k is the smallest nonzero singular value of $\bar{B}$ as defined in Lemma 1. The form of (9) will be different

if instead of $\hat{Z}_1$ one considers other objects as the end result of this step; see Section 5 (e.g., (34)) for instances of such variations. The appearance of Q is inevitable and is a consequence of the necessity of *properly aligning* the bases of spectral subspaces, before they can be compared in Frobenius norm (cf. Lemma 3). Nevertheless, the growing stack of orthogonal matrices on the RHS of $\bar{Z}_1$ has little effect on the performance of row-wise k-means, as we discuss shortly.

3. The final step is to analyze the effect of applying a k-means algorithm to $\hat{Z}_1$. Here, we introduce the concept of a *k-means matrix*, one whose rows take at most k distinct values. (See Section 3.3 for details). A k-means algorithm $\mathscr{K}$ takes a matrix $\hat{X} \in \mathbb{R}^{n \times m}$ and outputs a k-means matrix $\mathscr{K}(\hat{X}) \in \mathbb{R}^{n \times m}$. Our focus will be on k-means algorithms with the following property: If $X^* \in \mathbb{R}^{n \times m}$ is a k-means matrix, then for some constant $c > 0$,

$$\|\hat{X} - X^*\|_F^2 \leq \varepsilon^2 \implies \overline{\rm Mis}(\mathscr{K}(\hat{X}), X^*) \leq c\varepsilon^2/(n\delta^2). \tag{10}$$

where $\delta^2 = \delta^2(X^*)$ is the minimum center separation of X^* (cf. Definition 2), and $\overline{\rm Mis}$ is the average misclassification rate between two k-means matrices. For future reference, we refer to property (10) as the *local quadratic continuity (LQC)* of algorithm $\mathscr{K}$; see Remark 6 for the rationale behind the naming. As will become clear in Section 3.3, k-means matrices encode both the cluster label information and cluster center information, and these two pieces can be recovered from them in a lossless fashion. Thus, it makes sense to talk about misclassification rate between k-means matrices, by interpreting it as a statement about their underlying label information. In Section 3.3, we will discuss k-means algorithms that satisfy (10).

The preceding three steps of the analysis follow the three steps of a general spectral clustering algorithm, which we refer to as *regularization*, *spectral truncation* and *k-means* steps, respectively. Recalling the definition of cluster proportions, let us assume for some $\beta_r \geq 1$,

$$\max_{(t,s):\, t \neq s} \frac{2}{\pi_{rt}^{-1} + \pi_{rs}^{-1}} \leq \frac{\beta_r}{k_r}, \quad r = 1, 2. \tag{A1}$$

The LHS is the maximum *harmonic mean* of pairs of distinct cluster proportions. For balanced clusters, we have $\pi_{rt} = 1/k_r$ for all $t \in [k_r]$ and we can take $\beta_r = 1$. In general, β_r measures the deviation of the clusters (on side r) from balancedness. The following is a prototypical consistency theorem for a spectral clustering algorithm:

Theorem 1 (Prototype SC consistency) *Consider a spectral algorithm with a k-means step satisfying* (10)*, and the "usual" spectral truncation step, applied to a regularized bi-adjacency matrix A_{re} satisfying concentration bound* (8)*. Let $\mathscr{K}(\hat{Z}_1)$ be the resulting estimate for membership matrix Z_1, and assume $k_1 = k =: \min\{k_1, k_2\}$. Then, under the SBM model of Section 2 and assuming* (A1)*, w.h.p.,*

$$\overline{\rm Mis}(\mathscr{K}(\hat{Z}_1), \bar{Z}_1) \lesssim \beta_1 \Big(\frac{d}{\sigma_k^2} \Big).$$

Here, and in the sequel, "with high probability", abbreviated w.h.p., means with probability at least $1 - n^{-c_1}$ for some universal constant $c_1 > 0$. The notation $f \lesssim g$ means $f \leq c_2\, g$ where $c_2 > 0$ is a universal constant. In addition, $f \asymp g$ means $f \lesssim g$ and $g \lesssim f$.

Proof By assumption, concentration bound (8) holds. By Lemma 3 in Section 3.2, (8) implies (9) for the usual truncation step. Let $O := U_\psi Q$ and $\varepsilon^2 = C_2^2\, kd/\sigma_k^2$ so that (9) reads $\|\hat{Z}_1 - \bar{Z}_1 O\|_F \leq \varepsilon^2$. The k-means step satisfies (10) by assumption. Applying (10) with $\hat{X} = \hat{Z}_1$, $X^* = \bar{Z}_1 O$ and ε^2 defined earlier leads to

$$\overline{\text{Mis}}(\mathscr{K}(\hat{Z}_1), \bar{Z}_1) = \overline{\text{Mis}}(\mathscr{K}(\hat{Z}_1), \bar{Z}_1 O) \;\lesssim\; \frac{\varepsilon^2}{n_1 \delta^2}. \tag{11}$$

It remains to calculate the minimum center separation of $X^* = \bar{Z}_1 O \in \mathbb{R}^{n_1 \times k}$, where $\bar{Z}_1 \in \mathbb{O}^{n_1 \times k_1}$ and $O := U_\psi Q \in \mathbb{O}^{k_1 \times k}$. We have

$$\delta^2 = \delta^2(\bar{Z}_1 O) = \delta^2(\bar{Z}_1) = \min_{t \neq s} \|n_{1t}^{-1/2} e_t - n_{1s}^{-1/2} e_s\|_2^2 = \min_{t \neq s} (n_{1t}^{-1} + n_{1s}^{-1})$$

where $e_s \in \mathbb{R}^{k_1}$ is the sth standard basis vector. The second equality uses invariance of δ^2 to right-multiplication by a square orthogonal matrix. This is a consequence of $\|u^T O - v^T O\|_2 = \|u - v\|_2$ for $u, v \in \mathbb{R}^{k_1}$ and $O \in \mathbb{O}^{k_1 \times k}$ when $k_1 = k$; see (1). The third equality is from the definition $\bar{Z}_1 = Z_1 N_1^{-1/2}$. Using (A1),

$$(n_1 \delta^2)^{-1} \leq \max_{t \neq s} (\pi_{1t}^{-1} + \pi_{1s}^{-1})^{-1} \leq \frac{\beta_1}{2k_1}.$$

Thus, $\frac{\varepsilon^2}{n_1 \delta^2} \lesssim \beta_1 \frac{k}{k_1} \frac{d}{\sigma_k^2}$ which gives the result in light of (11) and assumption $k_1 = k$. ■

The case $k_1 > k$ will be discussed in Section 6.2. For (10) to hold for a k-means algorithm, one usually requires some additional constraints on $\varepsilon^2/(n\delta^2)$, ensuring that this quantity is small. We will restate Theorem 1 with such conditions explicitly once we consider the details of some k-means algorithms. For now, Theorem 1 should be thought of as a general blueprint, with specific variations obtained in Section 5 for various spectral clustering algorithms.

Remark 3 *To see that Theorem 1 is a consistency result, consider the typical case where $\beta_1 \asymp 1$, and $\sigma_k \asymp d$, so that $\overline{\text{Mis}}(\mathscr{K}(\hat{Z}_1), \bar{Z}_1) = O(d^{-1})$. Then, as long as $d \to \infty$, i.e., the average degree of the network grows with n, assuming $n_1 \asymp n_2 \asymp n$ (for some n), we have $\overline{\text{Mis}}(\mathscr{K}(\hat{Z}_1), \bar{Z}_1) = o(1)$, i.e., the average misclassification rate vanishes with high probability. One might ask why $\sigma_k \asymp d$ is reasonable. Consider the typical case where $\Psi = \rho_n \Psi'$ for some constant matrix Ψ' and a scalar parameter ρ_n that captures the dependence on n. This setup is common in network modeling (Bickel and Chen, 2009). Then, $d \asymp \|\Psi\|_\infty = \rho_n \|\Psi'\|_\infty$ and $\sigma_k = \rho_n \sigma_k'$ where σ_k' is defined based on Ψ' and hence constant. It follows that $d \asymp \rho_n \asymp \sigma_k$ and $\overline{\text{Mis}}(\mathscr{K}(\hat{Z}_1), \bar{Z}_1) = o(\rho_n^{-1})$. Note that, in general, ρ_n can grow as fast as n (cf. (4)). More specific examples are given in Section 5.*

Remark 4 *We note that bounds similar to (9) appear in the recent literature (Abbe et al., 2017; Cape et al., 2018) but for the $2 \to \infty$ operator norm, as opposed to the Frobenius norm, assuming that the network is dense enough. These bounds can be used to show that spectral clustering achieves exact recovery when the degree grows as $\log n$ for the two-community case (Abbe et al., 2017). A similar exact recovery result for general K is obtained in (Su et al., 2017) assuming a degree growth of $(\log n)^a$ for sufficiently large a. Whether the $2 \to \infty$ bounds can be used to sharpen the misclassification rate of spectral clustering below the $\log n$ regime is open.*

Remark 5 *Condition (A1) is more relaxed that what is commonly assumed in the literature (though the proof is the same). Stating the condition as a harmonic mean allows one to have similar results as the balanced case when one cluster is large, while others remain more or less balanced. For example, let $\pi_{r1} = 1 - c$ for some constant $c \in (0, 1)$, say $c = 0.4$, and let $\pi_{rt} = c/(k_r - 1)$ for $t \neq 1$. Then, we have for $s \neq t$*

$$\frac{2}{\pi_{rt}^{-1} + \pi_{rs}^{-1}} \leq 2 \min\{\pi_{rt}, \pi_{rs}\} = \frac{2c}{k_r - 1} \leq \frac{4}{k_r}$$

assuming $k_r \geq 2$. Hence (A1) holds with $\beta_r = 4$. Note that as k_r is increased, all but one cluster get smaller.

Remark 6 (LQC naming) *The rationale behind the naming of (10) is as follows: Let $\Delta(X, Y) := \frac{1}{\sqrt{n}} \|X - Y\|_F$ be the metric induced by the normalized Frobenius norm on the space of $n \times m$ matrices. Assume that X^* is a k-means matrix and that k-means matrices are fixed points of algorithm $\mathscr{K}$, hence $X^* = \mathscr{K}(X^*)$. Then, by taking the infimum over ε^2, (10) can be written as*

$$\overline{\text{Mis}}\left(\mathscr{K}(\hat{X}), \mathscr{K}(X^*)\right) \leq \omega\big(\Delta(\hat{X}, X^*)\big) \tag{12}$$

where $\omega(t) = ct^2/\delta^2(X^*)$, showing that $\mathscr{K}$ is continuous w.r.t. the two (pseudo)-metrics, locally at X^*, with a quadratic modulus of continuity. Note that this continuity is only required to hold around a k-means matrix X^* (and not in general). Another reason for the "locality" is that such statements often only hold for sufficiently small $\Delta(\hat{X}, X^*)$.

In this rest of this section, we fill in some details of the last two steps of the plan sketched above, deferring Step 1 to Section 4.

3.2. SV truncation (Step 2)

We now show how the concentration bound (8) implies the deviation bound (9) for the usual spectral truncation step. Consider the *symmetric dilation* operator : $\mathbb{R}^{n_1 \times n_2} \to \mathbb{S}^{n_1 + n_2}$, given by

$$P^\dagger := \begin{pmatrix} 0 & P \\ P^T & 0 \end{pmatrix}, \tag{13}$$

where $\mathbb{S}^n$ is the set of symmetric $n \times n$ matrices. This operator will be very useful in translating the results between the symmetric and non-symmetric cases. Let us collect some of its properties:

Lemma 2 (Symmetric Dilation) *Let $P \in \mathbb{R}^{n_1 \times n_2}$ have a reduced SVD given by $P = U\Sigma V^T$ where $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_k)$ is a $k \times k$ nonnegative diagonal matrix . Then,*

(a) $P^\dagger$ has a reduced EVD given by

$$P^\dagger = W \begin{pmatrix} \Sigma & 0 \\ 0 & -\Sigma \end{pmatrix} W^T, \quad W = \frac{1}{\sqrt{2}} \begin{pmatrix} U & U \\ V & -V \end{pmatrix} \in \mathbb{O}^{(n_1+n_2)\times 2k}.$$

(b) $P \mapsto P^\dagger$ is a linear operator with $\|P^\dagger\| = \|P\|$ and $\|P^\dagger\|_F = \sqrt{2}\|P\|_F$.

(c) The gap between k top (signed) eigenvalues of $P^\dagger$ and the rest of its spectrum is $2\sigma_k$.

Proof Part (a) can be verified directly (e.g. $W^TW = I_{2k}$ follows from $U^TU = V^TV = I_k$) and part (c) follows by noting that $\sigma_j \geq 0$ for all j. Part (b) also follows directly from part (a), using unitary-invariance of the two norms. ■

In addition, let us define a singular value (SV) truncation operator $\mathscr{T}_k : \mathbb{R}^{n_1 \times n_2} \to \mathbb{R}^{n_1 \times n_2}$ that takes a matrix A with SVD $A = \sum_i \sigma_i u_i v_i^T$ to the matrix

$$A^{(k)} := \mathscr{T}_k(A) := \sum_{i=1}^{k} \sigma_i u_i v_i^T. \tag{14}$$

In other words, $\mathscr{T}_k$ keeps the largest k singular values (and the corresponding singular vectors) and zeros out the rest. Recall that we order singular values in nonincreasing fashion $\sigma_1 \geq \sigma_2 \geq \cdots$. We also refer to (14) as the *k-truncated SVD* of A. Using the dilation and the Davis–Kahan (DK) theorem for symmetric matrices (Bhatia, 2013, Theorem VII.3.1), we have:

Lemma 3 *Let $A_{re}^{(k)} = \hat{Z}_1 \hat{\Sigma} \hat{Z}_2^T$ be the k-truncated SVD of A_{re} and assume that the concentration bound* (8) *holds. Let $\bar{Z}_1 U_\psi$ be given by the reduced SVD of P in* (7)*. Then, the deviation bound* (9) *holds for some $k \times k$ orthogonal matrix Q, and $C_2 = 2C$.*

Later, in Section 5, we introduce an alternative spectral truncation scheme. It is worth comparing the above lemma to Lemma 6 which establishes a similar result for the alternative truncation.

Remark 7 (Symmetric case) *When P is symmetric one can still use the dilation operator. In this case, since P itself is symmetric, it has an eigenvalue decomposition (EVD), say $P = U\Lambda U^T$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_k)$ is the diagonal matrix of the eigenvalues of P. Since these eigenvalues could be negative, there is a slight modification needed to go from the EVD to the SVD of P. Let s_i be the sign of λ_i and set $S = \mathrm{diag}(s_i, i = 1, \ldots, k)$. Then, it is not hard to see that with $V = US$ and $\Sigma = \Lambda S = \mathrm{diag}(|\lambda_i|, i = 1, \ldots, k)$, we obtain the SVD $P = U\Sigma V^T$. In other words, all the discussion in this section, and in particular Lemma 3 hold with $V = US$ and $\sigma_i = |\lambda_i|$. The special case of Lemmas 2 and 3 for the symmetric case appears in Lei et al. (2015). These observations combined with the fact that the concentration inequality discussed in Section 4 holds in the symmetric case leads to the following conclusion: All the results discussed in this paper apply to the symmetric SBM, for the version of the adjacency-based spectral clustering that* sorts the eigenvalues based on their absolute values. *This is the most common version of spectral algorithms in use. On the other hand, one gets a different behavior for the algorithm that considers the top k (signed) eigenvalues. We have borrowed the term "symmetric dilation" from Tropp (2015) where these ideas have been successfully used in translating matrix concentration inequalities to the symmetric case.*

3.3. k-means algorithms (Step 3)

Let us now give the details of the third and final step of the analysis. We introduce some notations and concepts that help in the discussion of k-means (type) algorithms.

k-means matrices. Recall that $\mathbb{H}^{n\times k}$ denotes the set of *hard* (cluster) labels: $\{0,1\}$-valued $n \times k$ matrices where each row has exactly a single 1. Take $Z \in \mathbb{H}^{n\times k}$. A related notion is that of a *cluster* matrix $Y = ZZ^T \in \{0,1\}^n$ where each entry denotes whether the corresponding pair are in the same cluster. Relative to Z, Y loses the information about the ordering of the cluster labels. We define the class of *k-means matrices* as follows:

$$\begin{aligned} \mathbb{M}_{n,m}^k &:= \{X \in \mathbb{R}^{n\times m} : \ X \text{ has at most } k \text{ distinct rows}\} \\ &= \{ZR : \ Z \in \mathbb{H}^{n\times k}, R \in \mathbb{R}^{k\times m}\}. \end{aligned} \tag{15}$$

The rows of R, which we denote as r_i^T, play the role of cluster centers. Let us also denote the rows of X as x_i^T. The second equality in (15) is due to the following correspondence: Any matrix $X \in \mathbb{M}_{n,m}^k$ uniquely identifies a *cluster* matrix $Y \in \{0,1\}^{n\times n}$ via, $Y_{ij} = 1$ iff $x_i = x_j$. This in turn "uniquely" identifies a label matrix Z up to $k!$ permutation of the labels. From Z, we "uniquely" recover R, with the convention of setting rows of R for which there is no label equal to zero. (This could happen if X has fewer than k distinct rows.)

With these conventions, there is a one-to-one correspondence between $X \in \mathbb{M}_{n,m}^k$ and $(Z, R) \in \mathbb{H}^{n\times k} \times \mathbb{R}^{k\times m}$, up to label permutations. That is, (Z, R) and (ZQ, QR) are considered equivalent for any permutation matrix Q. The correspondence allows us to talk about a (relative) misclassification rate between two k-means matrices: If $X_1, X_2 \in \mathbb{M}_{n,m}^k$ with membership matrices $Z_1, Z_2 \in \mathbb{H}^{n\times k}$, respectively, we set

$$\overline{\mathrm{Mis}}(X_1, X_2) := \overline{\mathrm{Mis}}(Z_1, Z_2). \tag{16}$$

k-means as projection. Now consider a general $\hat{X} \in \mathbb{R}^{n\times m}$. The classical k-means problem can be thought of as projecting $\hat{X}$ onto $\mathbb{M}_{n,m}^k$, in the sense of finding a nearest member of $\mathbb{M}_{n,m}^k$ to $\hat{X}$ in Frobenius norm. Let us write $d_F(\cdot,\cdot)$ for the distance induced by the Frobenius norm, i.e., $d_F(\hat{X}, X) = \|\hat{X} - X\|_F$. The k-means problem is that of solving the following optimization:

$$d_F(\hat{X}, \mathbb{M}_{n,m}^k) := \min_{X \in \mathbb{M}_{n,m}^k} d_F(\hat{X}, X). \tag{17}$$

The arguments to follow go through for any distance on matrices that has a ℓ_2 decomposition over the rows:

$$d_F(\hat{X}, X)^2 = \sum_{i=1}^{n} d(\hat{x}_i, x_i)^2, \tag{18}$$

where x_i^T and $\hat{x}_i^T$ are the rows of X and $\hat{X}$ respectively, and $d(\hat{x}_i, x_i)$ is some distance over vectors in $\mathbb{R}^d$. For the case of the Frobenius norm: $d(\hat{x}_i, x_i) = \|\hat{x} - x_i\|_2$, the usual ℓ_2 distance. This is the primary case we are interested in, though the result should be understood for the general case of (18). Since solving the k-means problem (17) is NP-hard, one can look for approximate solutions:

Definition 1 *A κ-approximate k-means solution for $\hat{X}$ is a matrix $\widetilde{X} \in \mathbb{M}_{n,m}^k$ that achieves κ times the optimal distance:*

$$d_F(\hat{X}, \widetilde{X}) \leq \kappa\, d_F(\hat{X}, \mathbb{M}_{n,m}^k). \tag{19}$$

We write $\mathcal{P}_\kappa : \mathbb{R}^{n\times m} \mapsto \mathbb{M}_{n,m}^k$ for the (set-valued) function that maps matrices $\hat{X}$ to κ-approximate solutions $\widetilde{X}$.

An equivalent restatement of (19) is

$$d_F(\hat{X}, \widetilde{X}) \leq \kappa\, d_F(\hat{X}, X), \quad \forall X \in \mathbb{M}_{n,m}^k. \tag{20}$$

Note that $\mathcal{P}_\kappa(\hat{X}) = \{\widetilde{X} \in \mathbb{M}_{n,m}^k : \ \widetilde{X} \text{ satisfies (20)}\}$. Our goal is to show that whenever $\hat{X}$ is close to some $X^* \in \mathbb{M}_{n,m}^k$, then any κ-approximate k-means solution based on it, namely $\widetilde{X} \in \mathcal{P}_\kappa(\hat{X})$ will be close to X^* as well. This is done in two steps:

1. If the distance $d_F(X, \widetilde{X})$ between two k-means matrices $X, \widetilde{X} \in \mathbb{M}_{n,m}^k$ is small, then their relative misclassification rate $\overline{\text{Mis}}(X, \widetilde{X})$ is so.

2. If a general matrix $\hat{X} \in \mathbb{R}^{n\times d}$ is close to a k-means matrix $X \in \mathbb{M}_{n,m}^k$, then so is its κ-approximate k-means projection. More specifically,

$$d_F(\widetilde{X}, X) \leq (1+\kappa)\, d_F(\hat{X}, X), \quad \forall \widetilde{X} \in \mathcal{P}_\kappa(\hat{X}), \tag{21}$$

   which follows from the triangle inequality $d_F(\widetilde{X}, X) \leq d_F(\widetilde{X}, \hat{X}) + d_F(\hat{X}, X)$ and (20).

Combining the two steps (taking $X = X^*$), we will have the result.

Let us now give the details of the first step above. For this result, we need the key notion of center separation. k-means matrices have more information that just a membership assignment. They also contain an encoding of the relative positions of the clusters, and hence the minimal pairwise distance between them, which is key in establishing a misclassification rate.

Definition 2 (Center separation) *For any $X \in \mathbb{M}_{n,m}^k$, let us denote its centers, i.e. distinct rows, as $\{q_r(X), r \in [k]\}$, and let*

$$\delta_r(X) = \min_{\ell:\, \ell \neq r} d(q_\ell(X), q_r(X)), \quad \delta_\wedge(X) = \min_r \delta_r(X). \tag{22}$$

In addition, let $n_r(X)$ be the number of nodes in cluster r according to X, and $n_\wedge(X) = \min_r n_r(X)$, the minimum cluster size.

If X has $m < k$, the convention would be to let $q_k(X) = 0$ for $k = m+1, \ldots, k$. We usually do not work with these degenerate cases. Implicit in the above definition is an enumeration of the clusters of X. We note that definition of $\delta_r = \delta_r(X)$ in (22) implies

$$d(q_\ell(X), q_r(X)) \geq \max\{\delta_\ell, \delta_r\}, \quad \forall (r, \ell) : r \neq \ell. \tag{23}$$

We are now ready to show that that any algorithm that computes a κ-approximate solution to the k-means problem (17) (where κ is some constant) satisfies the LQC property (10) needed in the last step of the analysis sketched in Section 3.1. Recall that $\text{Mis}_r(X; \widetilde{X})$ is the misclassification rate over the rth cluster of X (Section 1.1).

Proposition 1 *Let $X, \widetilde{X} \in \mathbb{M}_{n,m}^k$ be two k-means matrices, and write $n_r = n_r(X)$, $n_\wedge = n_\wedge(X)$ and $\delta_r = \delta_r(X)$. Assume that $d_F(X, \widetilde{X}) \leq \varepsilon$ and*

(a) X has exactly k nonempty clusters, and

(b) $c_r^{-2}\, \varepsilon^2/(\delta_r^2 n_r) < 1$ for $r \in [k]$, and constants $c_r > 0$ such that $c_r + c_\ell \leq 1$, $r \neq \ell$.

Then, $\widetilde{X}$ has exactly k clusters and

$$\text{Mis}_r(X; \widetilde{X}) \leq \frac{c_r^{-2}\, \varepsilon^2}{n_r\, \delta_r^2}, \quad \forall r \in [k]. \tag{24}$$

In particular, under the conditions of Proposition 1 with $c_r = 1/2$, we have

$$\text{Mis}_\infty(X, \widetilde{X}) \leq \frac{4\,\varepsilon^2}{\min_r n_r \delta_r^2} \leq \frac{4\,\varepsilon^2}{n_\wedge \delta_\wedge^2}, \qquad \overline{\text{Mis}}(X, \widetilde{X}) \leq \frac{4\,\varepsilon^2}{n\, \delta_\wedge^2}.$$

where the second one follows from the identity $\overline{\text{Mis}}(X, \widetilde{X}) = \sum_{r=1}^k (n_r/n)\, \text{Mis}_r(X, \widetilde{X})$. Combining Proposition 1 with (21), we obtain the following corollary:

Corollary 1 *Let $X^* \in \mathbb{M}_{n,m}^k$ be a k-means matrix, and write $n_r = n_r(X^*)$, $n_\wedge = n_\wedge(X^*)$ and $\delta_r = \delta_r(X^*)$. Assume that $\hat{X} \in \mathbb{R}^{n\times m}$ is such that $d_F(X^*, \hat{X}) \leq \varepsilon$ and*

(a) X^* has exactly k nonempty clusters, and

(b) $c_r^{-2}\, (1+\kappa)^2 \varepsilon^2/(\delta_r^2 n_r) < 1$ for $r \in [k]$, and constants $c_r > 0$ such that $c_r + c_\ell \leq 1$, $r \neq \ell$.

Then, any $\widetilde{X} \in \mathcal{P}_\kappa(\hat{X})$ has exactly k clusters and

$$\text{Mis}_r(X^*; \mathcal{P}_\kappa(\hat{X})) \leq \frac{c_r^{-2}\, (1+\kappa)^2 \varepsilon^2}{n_r\, \delta_r^2}, \quad \forall r \in [k]. \tag{25}$$

As before, $\text{Mis}_r(X^*, \mathcal{P}_\kappa(\hat{X}))$ should be interpreted as $\max_{\widetilde{X} \in \mathcal{P}_\kappa(\hat{X})} \text{Mis}_r(X^*, \widetilde{X})$, that is, the result hold for any κ-approximate k-means solution for $\hat{X}$. In particular, under the conditions of Corollary 1 with $c_r = 1/2$, we have

$$\overline{\text{Mis}}(X^*, \mathcal{P}_\kappa(\hat{X})) \ \leq \ 4(1+\kappa)^2 \frac{\varepsilon^2}{n\, \delta_\wedge^2}. \tag{26}$$

A similar bound holds for Mis_∞. Note that (26) is of the desired form needed in (10).

Remark 8 *Corollary 1 shows that any constant-factor approximation to the k-means problem* (17) *satisfies the LCQ property* (10)*. Such approximations can be computed in polynomial time; see for example Kumar et al. (2004). One can also use Lloyd's algorithm with* `kmeans++` *initialization to get a* $\kappa \lesssim \log k$ *approximation (Arthur and Vassilvitskii, 2007). Both Kumar et al. (2004) and Arthur and Vassilvitskii (2007) give constant probability approximations, hence if such algorithms are used in our subsequent results, "w.h.p." should be interpreted as with high constant probability (rather than* $1 - o(1)$*). Recently Lu and Zhou (2016) has shown that Lloyd's algorithm with random initialization can achieve near optimal misclassification rate in certain random models. Their analysis is complementary to ours in that we establish and use the LQC property* (10) *which uniformly holds for any input matrix* $\hat{X}$ *and we do not consider specific algorithms. The core ideas of our analysis in this section are borrowed from Lei et al. (2015) and Jin (2015). It is worth noting that there are other algorithms that turn a general matrix into a k-means matrix, without trying to approximate the k-means problem* (17)*, and still satisfy the LQC. We will give one such example, Algorithm 6, in Appendix B. Interestingly, in contrast to the k-means approximation algorithms, the LQC guarantee for Algorithm 6 is not probabilistic.*

4. Regularization and concentration

Here, we provide the details of Step 1, namely, the concentration of the regularized adjacency matrix. We start by a slight generalization of the results of Le et al. (2017) to the rectangular case:

Theorem 2 *Assume* $n_1 \le n_2$ *and let* $A \in \{0,1\}^{n_1 \times n_2}$ *have independent Bernoulli entries with mean* $\mathbb{E}[A_{ij}] = p_{ij}$*. Take* $d \ge \max_{ij} n_2 p_{ij}$*. Pick any subsets* $\mathcal{I}_1 \subset [n_1]$ *and* $\mathcal{I}_2 \subset [n_2]$ *such that*

$$|\mathcal{I}_1| \le 10 n_2/d, \quad \text{and} \quad |\mathcal{I}_2| \le 10 n_2/d \tag{27}$$

and fix some $d' > 0$*. Define a regularized adjacency matrix* A_{re} *as follows:*

(a) Set $[A_{re}]_{ij} = A_{ij}$ *for all* $(i,j) \in \mathcal{I}_1^c \times \mathcal{I}_2^c$*.*

(b) Set $[A_{re}]_{ij}$ *arbitrarily when* $i \in \mathcal{I}_1$ *or* $j \in \mathcal{I}_2$*, but subject to the constraints* $\|[A_{re}]_{i*}\|_1 \le d'$ *and* $\|[A_{re}]_{*j}\|_1 \le d'$ *for all* $i \in \mathcal{I}_1$ *and* $j \in \mathcal{I}_2$*.*

Then, for any $r \ge 1$*, with probability at least* $1 - n_2^{-r}$*, the new adjacency matrix* A_{re} *satisfies*

$$\|A_{re} - \mathbb{E}[A]\|_{op} \le C_2 r^{3/2}(\sqrt{d} + \sqrt{d'}). \tag{28}$$

The same result holds if in step (b) one uses ℓ_2 *norm instead of* ℓ_1*.*

Theorem 2 follows directly from the non-symmetric (i.e., directed) version of (Le et al., 2017, Theorem 2.1), by padding A with rows of zeros to get a square $n_2 \times n_2$ matrix. The result then follows by the same argument as in (Le et al., 2017, Theorem 2.1). The term "arbitrary" in the statement of the theorem includes any reduction even if the scheme is stochastic and depends on A itself. This feature will be key in developing data-driven schemes.

To apply Theorem 2, take $\mathcal{I}_1 = \{i \in [n_1] : \sum_j A_{ij} > 2d\}$ and $\mathcal{I}_2 = \{j \in [n_2] : \sum_i A_{ij} > 2d\}$, i.e., the set of rows and columns with degrees larger than $2d$. It is not hard to see that if d is any upper bound on the expected row and column degrees, then the sizes of these sets satisfy (27) with high probability. This follows, for example, from the same argument as that leading to (64) in Appendix A.2. Recalling the scaling of the connectivity matrix in (4), taking $d = n_2\|P\|_\infty$ gives an upper bound on the expected row and column degrees, assuming $n_2 \ge n_1$. Thus, we can apply Theorem 2 with $d' = d = \sqrt{n_2/n_1}\|\Psi\|_\infty = n_2\|P\|_\infty$, to obtain the desired concentration bound (8). Note that this concentration result does not require $d = n_2\|P\|_\infty$ to satisfy any lower bound (such as $d = \Omega(\log n_2)$). See Remarks 9 and 10 for the significance of this fact.

A disadvantage of the regularization scheme described in Theorem 2 is the required knowledge of a good upper bound on $d = n_2\|P\|_\infty$. In the next section, under a SBM with some mild regularity assumptions, we develop a fully data-driven scheme with the same guarantees as those of Theorem 2.

4.1. Data-driven regularization

We now describe a regularization scheme that is data-driven and does not require the knowledge of d as in Theorem 2. Let us write $D_i = \sum_{j=1}^{n_2} A_{ij}$ for the degree of node i on side 1 and $\overline{D} = \frac{1}{n_1}\sum_{i=1}^{n_1} D_i$ for the average degree on side 1. Consider the following order statistics for the degree sequence:

$$D_{(1)} \ge D_{(2)} \ge \cdots \ge D_{(n_1)}. \tag{29}$$

The idea is that under a block model, we can achieve concentration (8) by reducing the row degrees that are roughly above $D_{(\alpha)}$ for $\alpha = \lfloor n_1/\overline{D} \rfloor$ (and similarly for the columns). The overall scheme is described in Algorithm 1. The algorithm has a tuning parameter τ; however, we will show that taking $\tau = 3$ is enough. This constant is for convenience and has no special meaning. Note that the regularization described in Step 6 is a special case of that in Step 5; it is equivalent to first reducing the row degrees by setting $[A_{\mathrm{re}}]_{i*} \leftarrow A_{i*}\widehat{d}_1/D_i$ for $i \in \widehat{\mathcal{I}}_1$ followed by reducing the column degrees $[A_{\mathrm{re}}]_{*j} \leftarrow [A_{\mathrm{re}}]_{*j}\widehat{d}_2/D_j$ for $j \in \widehat{\mathcal{I}}_2$, or vice versa. Alternatively, we can replace the ℓ_1 norm constraints in Step 5 with ℓ_2 norm version and replace Step 6 with $[A_{\mathrm{re}}]_{ij} = A_{ij}\sqrt{w_i w'_j}$ for all $i \in [n_1], j \in [n_2]$.

Let $p_{ij} = \mathbb{E}[A_{ij}]$ and define $d_i := \mathbb{E}[D_i] = \sum_j p_{ij}$ and

$$\bar{d} = \frac{1}{n_1}\sum_{i=1}^{n} d_i, \quad d_{\max} := \max_i d_i, \quad d := n_2 \cdot \max_{i,j} p_{ij}. \tag{30}$$

Note that $\bar{d} \le d_{\max} \le d$ and we have $\bar{d} = \mathbb{E}[\overline{D}]$, that is, the expected average degree of the network (on side 1). We need the following mild regularity assumption:

(A2) Consider the SBM model given by (4) and (5) with $n_1 \le n_2$. Assume further that for some $\beta \ge 1$, and for all $t \in [k_1]$ and $s \in [k_2]$,

$$n_{1t} \ge \frac{n_1}{\beta k_1}, \quad n_{2s} \ge \frac{n_2}{\beta k_2}, \quad \text{and} \quad \left(\frac{n_2}{n_1}\right)^2 \max\{8\beta k_1, 8\beta k_2, 90\} \le \bar{d} \le \frac{1}{2}n_1. \tag{31}$$

Algorithm 1 Data-driven adjacency regularization

Input: Adjacency matrix $A \in \mathbb{R}^{n_1 \times n_2}$ and regularization parameter τ. (Default: $\tau = 3$)

Output: Regularized adjacency matrix $A_{\rm re} \in \mathbb{R}^{n_1 \times n_2}$.

1: Form the degree sequence $D_i = \sum_{j=1}^{n_2} A_{ij}$ for $i = 1, \ldots, n$ and the corresponding order statistics: $D_{(1)} \geq D_{(2)} \geq \cdots \geq D_{(n_1)}$.

2: Let $\overline{D} = \frac{1}{n_1} \sum_{i=1}^{n_1} D_i$ and $\alpha = \lfloor n_1 / \overline{D} \rfloor$.

3: Set $\widehat{d}_1 = \tau D_{(\alpha)}$ and $\widehat{\mathcal{I}}_1 = \{i : D_i \geq \widehat{d}_1\}$.

4: Repeat Steps 1–3 on A^T to get $\widehat{\mathcal{I}}_2$ and $\widehat{d}_2$, and let D'_j be the corresponding degrees (i.e., column degrees in the original matrix A).

5: Set $[A_{\rm re}]_{ij} = A_{ij}$ for all $(i,j) \in \widehat{\mathcal{I}}_1^c \times \widehat{\mathcal{I}}_2^c$, and set $[A_{\rm re}]_{ij}$ arbitrarily when $i \in \widehat{\mathcal{I}}_1$ or $j \in \widehat{\mathcal{I}}_2$, but subject to: $\|[A_{\rm re}]_{i*}\|_1 \leq \widehat{d}_1$ for all $i \in \widehat{\mathcal{I}}_1$ and $\|[A_{\rm re}]_{*j}\|_1 \leq \widehat{d}_2$ for all $j \in \widehat{\mathcal{I}}_2$.

6: For example, $[A_{\rm re}]_{ij} = A_{ij} w_i w'_j$ for all $(i,j) \in [n_1] \times [n_2]$, where $w_i = \frac{\widehat{d}_1}{D_i} \wedge 1$ and $w'_j = \frac{\widehat{d}_2}{D'_j} \wedge 1$ satisfies the conditions in Step 5.

The next key lemma shows that $D_{(\alpha)}$ is a good proxy for $d_{\max}$ and d under a SBM:

Lemma 4 *Let $\alpha = \lfloor n_1 / \overline{D} \rfloor$ where $\overline{D}$ is the average degree on side 1 as defined earlier. Under assumption (A2), the following hold, with probability at least $1 - 3e^{-n_1/80} - e^{-n_1/(\beta k_2)}$,*

(a) $d_{\max}/2 \leq D_{(\alpha)} \leq 3 d_{\max}/2$.

(b) $\big|i : D_i > 3 D_{(\alpha)}\big| \leq 10 n_1 / d$.

Conditions in (31) are quite mild: The first and second require that the clusters associated with the maximum degree are not too small. In the balanced case, where all clusters are of equal size, we have $n_{rt} = n_r / k_r$ (for all $t \in [k_r]$ and $r = 1, 2$) hence the condition is satisfied with $\beta = 1$. In general, β measures deviations from balancedness. Note that (A1) and the first two conditions in (31) do not imply each other. The last condition in (31) is satisfied if $\bar{d} \to \infty$ slower than n_1 and faster than $\max\{k_1, k_2\}$ (assuming $\beta = O(1)$ and $n_1 \asymp n_2$).

A result similar to Lemma 4 holds for the column degrees with appropriate modifications: Let $D'_j = \sum_i A_{ij}$ be the column degrees with expectation $d'_j = \sum_i p_{ij}$, and define the corresponding average $\bar{d}' = \frac{1}{n_2} \sum_j d'_j$ and $d'_{\max} = \max_j d'_j$. We, however, keep d as defined in (30).

Lemma 5 *Let $\alpha' = \lfloor n_2 / \overline{D}' \rfloor$ where $\overline{D}'$ is the average degree on side 2 as defined earlier. Under assumption (A2), the following hold, with probability at least $1 - 3e^{-n_2/80} - e^{-n_1/(\beta k_1)}$,*

(a) $d'_{\max}/2 \leq D'_{(\alpha')} \leq 3 d'_{\max}/2$.

(b) $\big|j : D'_j > 3 D'_{(\alpha')}\big| \leq 10 n_2 / d$.

Equipped with Lemmas 4 and 5, we have the following for Algorithm 1:

Theorem 3 *Assume that $A \in \mathbb{R}^{n_1 \times n_2}$ is generated from the SBM model satisfying (A2). Then, for any $r \geq 1$, with probability at least $1 - n_2^{-r} - 6e^{-n_1/80} - 2e^{-n_1/\beta \max\{k_1,k_2\}}$, the regularized output A_{re} of Algorithm 1 (with $\tau = 3$) satisfies*

$$\|A_{re} - \mathbb{E}[A]\|_{op} \leq C_3 r^{3/2} \sqrt{d}, \tag{32}$$

where d is as in (30) and $C_3 > 0$ is a universal constant.

Proof [Proof of Theorem 3] Recall the definitions of $\widehat{\mathcal{I}}_r$ and $\widehat{d}_r$ ($r = 1, 2$) in Algorithm 1. On the union of the events described in Lemmas 4 and 5, both $|\widehat{\mathcal{I}}_1|$ and $|\widehat{\mathcal{I}}_2|$ are bounded by $10 n_2/d$ (since $n_1 \leq n_2$), $\widehat{d}_1 = 3 D_{(\alpha)} \leq 9 d_{\max}/2$ and $\widehat{d}_2 = 3 D'_{(\alpha')} \leq 9 d'_{\max}/2$. Since $d_{\max} \leq n_2 \|P\|_\infty = d$ and $d'_{\max} \leq n_1 \|P\|_\infty \leq d$, we can apply Theorem 2 with $d' = 9d/2$ completing the proof. ■

Theorem 3 thus provides the same concentration guarantee as in Theorem 2 without the knowledge of d. See Appendix A.2 for the proof of the two lemmas of this section.

Remark 9 (Comparison with existing work) *Results of the form (8) hold for A itself without any regularization if one further assumes that $d \gtrsim \log n_2$. These result are often derived for the symmetric case; see for example Tomozei and Massoulié (2010), Lei et al. (2015) and Chen and Xu (2016); or Bandeira et al. (2016) for the more general result with $d = \max_i \sum_j p_{ij}$. In order to break the $\log n_2$ barrier on the degrees, one has to resort to some form of regularization. When d is given, the general regularization for the adjacency matrix is to either to remove the high degree nodes as in Chin et al. (2015) or reduce their effect as in Le et al. (2017) and Theorem 2 above. In contrast, there is little work on data-driven regularization for the adjacency matrix. One such algorithm was investigated in Gao et al. (2017) for the special case of the SPBB model—discussed in Example 1 (Section 5)—under the assumption $a = O(b)$. The algorithm truncates the degrees to a multiple of the average degree, i.e., $\widehat{d}_1 = C\overline{D}$ for some large $C > 0$. A possible choice of $C = 5$ is given in Yun and Proutiere (2014a) assuming that the expected degrees of the nodes are similar.*

In contrast to existing results, our data-driven regularization holds for a general SBM and only requires the mild assumptions in (A2) while preserving the same upper bound (32) that holds with the knowledge of d. In particular, we do not require $\min_{k\ell} \Psi_{k\ell} \asymp \max_{k\ell} \Psi_{k\ell}$ (which is what $a = O(b)$ means in the SPBB model). Algorithm 1 enables effective and provable regularization without knowing any parameters.

5. Consistency results

We now state our various consistency results. The proofs are deferred to Appendix A.4. We start with a refinement of Theorem 1 for the specific algorithm SC-1 given in Algorithm 2.

Theorem 4 *Consider the spectral algorithm* SC-1 *given in Algorithm 2. Assume $k_1 = k =: \min\{k_1, k_2\}$, and for a sufficiently small $C > 0$,*

$$k d\, \sigma_k^{-2} \leq C (1 + \kappa)^{-2}.$$

Then, under the SBM model satisfying (A2), w.h.p.,

$$\overline{\rm Mis}(\mathcal{P}_\kappa(\hat{Z}_1), \bar{Z}_1) \;\lesssim\; (1+\kappa)^2 \beta_1\Big(\frac{d}{\sigma_k^2}\Big).$$

Algorithm 2 SC-1

1: Apply regularization Algorithm 1 to A to obtain $A_{\rm re}$.
2: Obtain the k-truncated SVD of $A_{\rm re}$ as $A_{\rm re}^{(k)} = \hat{Z}_1\hat{\Sigma}\hat{Z}_2^T$. See (14).
3: Output an element of $\mathcal{P}_\kappa(\hat{Z}_1)$, i.e., a κ-approximate k-means solution for input $\hat{Z}_1$.

Algorithm 3 SC-RR

1: Apply regularization Algorithm 1 to A to obtain $A_{\rm re}$.
2: Obtain the best rank k approximation of $A_{\rm re}$, that is, $A_{\rm re}^{(k)} = \mathscr{T}_k(A_{\rm re})$. See (14).
3: Output an element of $\mathcal{P}_\kappa(A_{\rm re}^{(k)})$, i.e., a κ-approximate k-means solution for input $A_{\rm re}^{(k)}$.

where β_1 is given in (A1) *and d is defined in* (8).

One can take κ to be a fixed small constant say 1.5, since there are κ-approximate k-means algorithms for any $\kappa > 1$. In that case, $(1+\kappa)^2$ can be absorbed into other constants, and the bound in Theorem 4 is qualitatively similar to Theorem 1.

5.1. Reduced-rank SC

Theorem 4 implicitly assumes $\sigma_k > 0$, otherwise the bound is vacuous. This assumption is clearly violated if B is rank deficient (or equivalently, P has rank less than k). A variant of SC suggested in Yun and Proutiere (2014a) and Gao et al. (2018) can resolve this issue. The idea is to use the entire rank k approximation of $A_{\rm re}$, and not just the singular vector matrix $\hat{Z}_1$, as the input to the k-means step. This approach, which we call reduced-rank SC, or SC-RR, is detailed in Algorithm 3. Recall the SV truncation operator $\mathscr{T}_k$ given in (14). It is well-known that $\mathscr{T}_k$ maps every matrix to its best rank-k approximation in Frobenius norm, i.e.,

$$\mathscr{T}_k(A_{\rm re}) = \min\big\{\|R - A_{\rm re}\|_F : \ {\rm rank}(R) \le k\big\}$$

with the approximation error satisfying

$$\|\mathscr{T}_k(A_{\rm re}) - A_{\rm re}\|_{\rm op} = \sigma_{k+1}(A_{\rm re}). \tag{33}$$

SC-RR uses this best rank-k approximation as a denoised version of $A_{\rm re}$ and runs a k-means algorithm on its rows. To analyze SC-RR, we need to replace bound (9) in Step 2 with an appropriate modification. The following lemma replaces Lemma 3 and provides the necessary bound in this case.

Lemma 6 *Let $A_{re}^{(k)} = \mathscr{T}_k(A_{re})$ be the k-truncated SVD of A_{re} and assume that the concentration bound* (8) *holds. Then,*

$$\|A_{re}^{(k)} - P\|_F \le C\sqrt{8\,kd}. \tag{34}$$

Comparing with (9), we observe that (34) provides an improvement by removing the dependence on the singular value gap σ_k. However, we note that in terms of the relative

error, i.e., $\|A_{\rm re}^{(k)} - P\|_F/\|P\|_F$ this may or may not be an improvement. There are cases where $\|P\|_F \approx \sqrt{k}\sigma_k$, in which case the relative error predicted by (34) is $O(\sqrt{d}/\sigma_k)$, similar to the relative error bound based on (9) (since $\|\bar{Z}_1 U_\psi Q\|_F = \sqrt{k}$); see Example 1 below.

Following through the three-step analysis of Section 3.1, with (9) replaced with (34), we obtain a qualitatively different bound on the misclassification error of Algorithm 3. The key is that center separation of P treated as a k-means matrix is different from that of $\bar{Z}_1 U_\psi Q$. Note that P is indeed a valid k-means matrix according to Definition (15); in fact, $P \in \mathbb{M}_{n_1,n_2}^{k_1}$. Similarly, $P^T \in \mathbb{M}_{n_2,n_1}^{k_2}$. Let us define

$$\Psi_{1,\wedge}^2 := \min_{(s,t):\, s\neq t} \sum_{\ell=1}^{k_2} \pi_{2\ell}(\Psi_{s\ell} - \Psi_{t\ell})^2, \tag{35}$$

$$\widetilde{\Psi}_{1,\wedge}^2 := \min_{(s,t):\, s\neq t} \Big[\pi_{1t} \sum_{\ell=1}^{k_2} \pi_{2\ell}(\Psi_{s\ell} - \Psi_{t\ell})^2\Big]. \tag{36}$$

Theorem 5 *Consider the spectral algorithm* SC-RR *given in Algorithm 3. Assume that for a sufficiently small $C_1 > 0$,*

$$kd\,\widetilde{\Psi}_{1,\wedge}^{-2} \le C_1(1+\kappa)^{-2}. \tag{37}$$

Then, under the SBM model satisfying (A2), with d as defined in (8)*, w.h.p.,*

$$\overline{\rm Mis}(\mathcal{P}_\kappa(A_{re}^{(k)}), P) \ \le \ C_1^{-1}(1+\kappa)^2\Big(\frac{kd}{\Psi_{1,\wedge}^2}\Big).$$

As is clear from the proof, one can take $C_1 = 1/(32C^2)$ where C is the constant in concentration bound (8). Condition (37) can be replaced with the stronger assumption

$$kd\,(\pi_{1,\wedge}\Psi_{1,\wedge}^2)^{-1} \le C_1(1+\kappa)^{-2} \tag{38}$$

a where $\pi_{1,\wedge} := \min_{t\in[k_1]}\pi_{1t}$, since $\widetilde{\Psi}_{1,\wedge}^2 \ge \pi_{1,\wedge}\Psi_{1,\wedge}^2$.

Although the bounds of Theorems 4 and 5 are different, surprisingly, in the case of the planted partition model, they give the same result as the next example shows.

Example 1 (Planted partition model, symmetric case) *Let us consider the simplest symmetric SBM, the symmetric balanced planted partition (SBPP) model, and consider the consequences of Theorems 4 and 5 in this case. Recall that in the symmetric case we drop index r from k_r, n_r, n_{rj}, $\bar{N}_r$, β_r, $\Psi_{r,\wedge}$ and so on. SBPP is characterized by the following assumptions:*

$$\Psi = bE_k + (a-b)I_k, \ \ a \ge b, \quad \pi_j = n_j/n = \frac{1}{k}, \ \forall j \in [k].$$

Here, $E_k \in \mathbb{R}^{k\times k}$ is the all ones matrix and balanced *refers to all the communities being of equal size, leading to cluster proportions $\pi_j = 1/k$. In particular, $\beta = 1$, as defined in* (A1)*. We have $\bar{B} = \bar{N}^{1/2}\Psi\bar{N}^{1/2} = \Psi/k$, recalling $\bar{N} = {\rm diag}(\pi_j)$. Hence, the smallest singular value of $\bar{B}$ is $\sigma_k = (a-b)/k$. Theorem 4 gives the following result:*

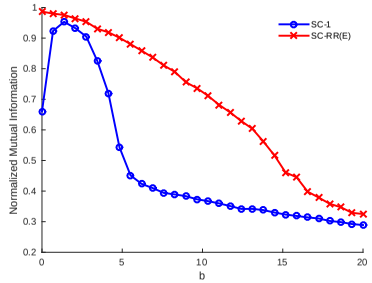


Figure 1: An example of the performance boost of SC-RR (or SC-RRE) relative to SC-1. The data is generated from the bipartite version of Example 2 with $n_2 = 2n_1 = 1000$, $k_1 = k_2 = 4$, $\pi_{r\ell} = n_r/k_r$ for all $\ell \in [k_r]$, $r = 1, 2$, and $\Psi = 2bE_4 + \mathrm{diag}(16, 16, 16, 2)$ similar to (39). The key is the significant difference in the two smallest diagonal elements of Ψ. The plot shows the normalized mutual information (a measure of cluster quality) between the output of the two spectral clustering algorithms and the true clusters, as b varies. Only row clusters are considered. The plot shows a significant improvement for SC-RR(E) relative to SC-1 over a range of b. As b increases, the relative difference between Ψ_{33} and Ψ_{44} reduces and the model approaches that of Example 1, leading to similar performances for both algorithms as expected. It is interesting to note that the monotone nature of the performance of SC-RR(E) as a function of b and the non-monotone nature of that of SC-1 is reflected in the upper bounds (40) and (41).

Corollary 2 *Under the SBPP model, as long as $k^3a/(a-b)^2$ is sufficiently small,* SC-1 *has average misclassification error of $O(k^2a/(a-b)^2)$ with high probability.*

Now consider SC-RR. *Using definitions* (35), *we have $k\widetilde{\Psi}_\wedge^2 = \Psi_\wedge^2 = 2(a-b)^2/k$. Then, Theorem 5 gives the exact same result for* SC-RR*:*

Corollary 3 *Corollary 2 holds with* SC-1 *replaced with* SC-RR.

Results of Corollary 2 and 3 are consistency results as long as $k^2a/(a-b)^2 = o(1)$. A typical example is when $k = O(1)$, $a = a_0 f_n$, $b = b_0 f_n$, $a_0 \asymp 1$ and $b_0 \asymp 1$ for some $f_n \to \infty$ as $n \to \infty$. Then, SC-1 *and* SC-RR *are both consistent at a rate $O(f_n^{-1})$.*

Let us now give an example where SC-1 and SC-RR behave differently.

Example 2 *Consider the symmetric balanced SBM, with*

$$\Psi = bE_k + \mathrm{diag}(\alpha_1, \ldots, \alpha_k), \quad \pi_j = n_j/n = \frac{1}{k}, \ \forall j \in [k]. \tag{39}$$

As in Example 1, we have dropped the index r determining the side of network in the bipartite case. Let us assume that $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_k \geq 0$. We have

$$k\widetilde{\Psi}_\wedge^2 = \Psi_\wedge^2 = k^{-1} \min_{s \neq t} \sum_\ell (\Psi_{s\ell} - \Psi_{t\ell})^2 = k^{-1}(\alpha_{k-1}^2 + \alpha_k^2).$$

Thus, Theorem 5 gives the following: With ρ defined as follows:

$$\rho := k^2 \frac{\alpha_1 + b}{\alpha_{k-1}^2 + \alpha_k^2}, \tag{40}$$

as long as $k\rho$ is sufficiently small, SC-RR *has average misclassification error $O(\rho)$ with high probability.*

To determine the performance of SC-1*, we need to estimate σ_k, the smallest singular value of $\bar{B} = \bar{N}^{1/2}\Psi\bar{N}^{1/2} = \Psi/k$. Since Ψ is obtained by a rank-one perturbation of a diagonal matrix, it is well-known that when $\{\alpha_t\}$ are distinct, the eigenvalues of Ψ are obtained by solving $\sum_{t=1}^k 1/(\alpha_t - \lambda) = -1/b$; the case where some of the $\{\alpha_t\}$ are repeated can be reasoned by the taking the limit of the general case. By plotting $\lambda \mapsto \sum_{t=1}^k 1/(\alpha_t - \lambda)$ and looking at the intersection with $\lambda \mapsto -1/b$, one can see that the smallest eigenvalue of Ψ, equivalently its smallest singular value, is in $[\alpha_k, \alpha_{k-1}]$, and can be made arbitrarily close to α_k by letting $b \to 0$. Letting $\alpha_k + \varepsilon_k(\alpha; b)$ denote this smallest singular value, we have $0 \leq \varepsilon_k(\alpha; b) \to 0$ as $b \to 0$.*

It follows that $\sigma_k = \sigma_k(\bar{B}) = k^{-1}(\alpha_k + \varepsilon_k(\alpha; b))$. Theorem 4 gives the following: With ρ defined as

$$\rho := k^2 \frac{\alpha_1 + b}{(\alpha_k + \varepsilon_k(\alpha; b))^2}, \tag{41}$$

as long as $k\rho$ is sufficiently small, SC-1 *has average misclassification error $O(\rho)$ with high probability.*

Comparing (41) *with* (40)*, the ratio of the two bounds is $(\alpha_{k-1}^2 + \alpha_k^2)/(\alpha_k + \varepsilon_k(\alpha; b))^2 \to 1 + (\alpha_{k-1}/\alpha_k)^2$ as $b \to 0$. This ratio could be arbitrarily large depending on the relative sizes of α_k and α_{k-1}. Thus, when the bounds give an accurate estimate of the misclassification rates of* SC-1 *and* SC-RR*, we observe that* SC-RR *has a clear advantage. This is empirically verified in Figure 1, for moderately dense cases. (In the very sparse case, the difference is not very much empirically.) In general, we expect* SC-RR *to perform better when there is a large gap between σ_k and σ_{k-1}, the two smallest nonzero singular values of $\bar{B}$.*

Example 3 (Rank-deficient connectivity) *Consider an extreme case where Ψ is rank one: $\Psi = uv^T$ for some $u, v \in \mathbb{R}_+^k$, where again for simplicity we have assumed $k_1 = k_2 = k > 1$. Also assume $n_1 \asymp n_2 = n$ and $\pi_{rj} = 1/k$ for $j \in [k]$ and $r = 1, 2$, i.e., the clusters are balanced. In this case, $\bar{B} = \Psi/k$ and $\sigma_k = 0$, hence Theorem 4 does not provide any guarantees for* SC-1*. However, Theorem 3 is still valid. We have $k\widetilde{\Psi}_{1,\wedge}^2 = \Psi_{1,\wedge}^2 = k^{-1}\|v\|_2^2 \min_{s \neq t}(u_s - u_t)^2$ and $d \lesssim \|\Psi\|_\infty \leq \|u\|_\infty \|v\|_\infty$. It follows from Theorem 3 that* SC-RR *has average misclassification rate bounded as*

$$O\Big(\frac{k^2\|u\|_\infty\|v\|_\infty}{\|v\|_2^2 \min_{s \neq t}(u_s - u_t)^2}\Big)$$

whenever k times the above is sufficiently small. This is a consistency result assuming that the coordinates of u are different, all the elements of u and v are growing at the same rate and $k = O(1)$.

Algorithm 4 SC-RRE

1: Apply degree regularization Algorithm 1 to A to obtain $A_{\rm re}$.
2: Obtain $A_{\rm re}^{(k)} = \hat{Z}_1\hat{\Sigma}\hat{Z}_2^T$, the k-truncated SVD of $A_{\rm re}$.
3: Output $\mathscr{K}(\hat{Z}_1\hat{\Sigma})$ where $\mathscr{K}$ is an isometry-invariant κ-approximate k-means algorithm.

5.2. Efficient reduced-rank SC

The SC-RR algorithm discussed above has the disadvantage of running a k-means algorithm on vectors in $\mathbb{R}^n$ (the rows of $A_{\rm re}^{(k)}$, or in the ideal case the rows of P). We now introduce a variant of this algorithm that has the same performance as SC-RR in terms of misclassification rate, while computationally is as efficient as SC-1. This approach which we call efficient reduced-rank spectral clustering, SC-RRE, is detailed in Algorithm 4. The efficiency comes from running the k-means step on vectors in $\mathbb{R}^k$ which is usually a much smaller space than $\mathbb{R}^n$ ($k \ll n$ in applications).

For the k-means step in SC-RRE, we need a k-means (type) algorithm $\mathscr{K}$ that only uses the pairwise distances between the data points. We call such k-means algorithms *isometry-invariant*:

Definition 3 *A k-means (type) algorithm $\mathscr{K}$ is isometry-invariant if for any two matrices $X^{(r)} \in \mathbb{R}^{n\times d_r}, r = 1, 2$, with the same pairwise distances among points—i.e., $d(x_i^{(1)}, x_j^{(1)}) = d(x_i^{(2)}, x_j^{(2)})$ for all distinct $i, j \in [n]$, where $(x_i^{(r)})^T$ is the ith row of $X^{(r)}$—one has*

$$\overline{\rm Mis}(\mathscr{K}(X^{(1)}), \mathscr{K}(X^{(2)})) = 0.$$

Although the rows of $\mathscr{K}(X^{(1)})$ and $\mathscr{K}(X^{(2)})$ lie in spaces of possibly different dimensions, it still makes sense to talk about their relative misclassification rate, since this quantity only depends on the membership information of the k-means matrices and not their center information. We have implicitly assumed that $d(\cdot,\cdot)$ defines a family of distances over all Euclidean spaces $\mathbb{R}^d, d = 1, 2, \ldots$. This is obviously true for the common choice $d(x, y) = \|x - y\|_2$. If algorithm $\mathscr{K}$ is randomized, we assume that the same source of randomness is used (e.g., the same random initialization) when applying to either of the two cases $X^{(1)}$ and $X^{(2)}$.

The following result guarantees that SC-RRE behaves the same as SC-RR when one uses an isometry-invariant approximate k-means algorithm in the final step.

Theorem 6 *Consider the spectral algorithm* SC-RRE *given in Algorithm 4. Assume that for a sufficiently small $C_1 > 0$, (37) holds. Then, under the SBM model of Section 2, w.h.p.,*

$$\overline{\rm Mis}(\mathscr{K}(\hat{Z}_1\hat{\Sigma}), P) \;\le\; C_1^{-1}\,(1+\kappa)^2\Big(\frac{kd}{\Psi_{1,\wedge}^2}\Big).$$

Remark 10 (Comparison with existing results) *The existing results hold under different assumptions. Table 1 provides a summary of some the recent results. The "p vs. q" denotes the assortative case where the diagonal entries of B are above p and off-diagonal entries are below q. The "e.v. dep" denotes whether the consistency result depends on the*

Table 1: Comparison of consistency results (cf. Remark 10).

	min. degree	p vs. q	$p = O(q)$	e.v. dep	given Ψ	k-means
Rohe et al. (2011)	$\Omega(n/\sqrt{\log n})$.	No.	Not needed.	Yes.	No.	$n \times k$.
Yun et al. (2014a)	Not needed.	Yes	Not needed.	No.	No.	$n \times n$.
Lei et al. (2015)	$\Omega(\log n)$.	No.	Not needed.	Yes.	No.	$n \times k$.
Chin et al. (2015)	Not needed.	Yes.	Required.	Yes.	Yes.	$n \times k$.
Gao et al. (2017)	Not needed.	Yes.	Required.	Yes.	No.	$n \times k$.
Gao et al. (2018)	Not needed.	Yes.	Required.	No.	No.	$n \times n$.
SC-1	Not needed.	No.	Not needed.	Yes.	No.	$n \times k$.
SC-RRE	Not needed.	No.	Not needed.	No.	No.	$n \times k$.

kth eigenvalue or singular value of B (or Ψ). The "k-means" column records the dimension of the matrix on which a k-means algorithm is applied. To allow for better comparison, let us consider a typical (special) case of the setting in this paper, where $n_1 = n_2$, $k_1 = k_2$, Ψ is symmetric and $d \asymp \Psi_{1,\wedge}$. Then all the spectral methods in Table 1 have misclassification rate guarantees that are polynomial in d^{-1}. General SBMs without assortative assumption (e.g., "p vs. q") were considered in earlier literature (Rohe et al., 2011; Lei et al., 2015). However, the theoretical guarantees were provided for sufficiently dense networks. The generalization to sparse networks is considered in Yun and Proutiere (2014a), Chin et al. (2015), Gao et al. (2017) and Gao et al. (2018) using some regularization on the adjacency matrix. However, these results only apply to assortative networks and with extra assumptions. Overall, the algorithms SC-1 *and* SC-RRE *require less assumptions than any existing works. We also note that the guarantees of Theorems 5 and 6 in the context of a general SBM are new and have not appeared before (not even in the dense case).*

5.3. Results in terms of mean parameters

One useful aspect of SC-RR(E) is that one can state its corresponding consistency result in terms of the *mean parameters* of the block model. Such results are useful when comparing to the optimal rates achievable in recovering the clusters. The row mean parameters of the SBM in Section 2 are defined as $\Lambda_{s\ell} := B_{s\ell}\, n_{2\ell}$ for $(s, \ell) \in [k_1] \times [k_2]$ which we collect in a matrix $\Lambda = (\Lambda_{s\ell}) \in \mathbb{R}^{k_1\times k_2}$. To get an intuition for Λ note that

$$\mathbb{E}[AZ_2] = PZ_2 = Z_1BN_2 = Z_1\Lambda.$$

Each row of AZ_2 is obtained by summing the corresponding row of A over each of the column clusters to get a k_2 vector. In other words, the rows of AZ_2 are the sufficient statistics for estimating the row clusters, had we known the true column clusters. Note that $\mathbb{E}[(AZ_2)_{i*}] = z_{1i}^T\Lambda$, where the notation $(\cdot)_{i*}$ denotes the ith row of a matrix. In other words, we have $\mathbb{E}[(AZ_2)_{i*}] = \Lambda_{s*}$ if node i belongs to row cluster s. Let us define the minimum separation among these row mean parameters:

$$\Lambda_\wedge^2 := \min_{t\neq s} \|\Lambda_{s*} - \Lambda_{t*}\|^2. \tag{42}$$

We have the following corollary of Theorem 5 which is proved in Appendix A.4.

Corollary 4 *Assume that* $\pi_{r,\wedge} := \min_t \pi_{rt} \geq (\beta_r k_r)^{-1}$ *for* $r = 1, 2$, *and let* $k = \min\{k_1, k_2\}$ *and* $\alpha = n_2/n_1$. *Consider the spectral algorithm* SC-RR *given in Algorithm 3. Assume that for a sufficiently small* $C_1 > 0$,

$$\beta_1\beta_2\, k\, k_1 k_2\, \alpha \frac{\|\Lambda\|_\infty}{\Lambda_\wedge^2} \leq C_1(1+\kappa)^{-2}. \tag{43}$$

Then, under the SBM model of Section 2, w.h.p.,

$$\overline{\mathrm{Mis}}(\mathcal{P}_\kappa(A_{re}^{(k)}), P) \;\leq\; C_1^{-1}\,(1+\kappa)^2\beta_2\, k\, k_2\, \alpha \frac{\|\Lambda\|_\infty}{\Lambda_\wedge^2}.$$

We note that the exact same result as Corollary 4 holds for SC-RRE assuming the k-means step uses an isometry invariant algorithm as discussed in Section 5. We refer to Zhou and Amini (2018) for an application of this result in constructing optimal clusterings in the bipartite setting.

6. Extensions

6.1. Clusters on one side only

The bipartite setting allows for the case where only one side has clusters. Assume that $A \sim \mathrm{Ber}(P)$ in the sense of (5) and, for example, only side 2 has k_2 clusters. Then we can model the problem as P having k_2 distinct columns. However, within columns we do not require any block constant structure, i.e., the k_2 distinct columns of P are general vectors in $[0,1]^{n_1}$. This problem can be considered a special case of the SBM model discussed in Section 2 where $k_1 = n_1$: We recall that $P = \bar{Z}_1 \bar{B} \bar{Z}_2^T$ where $\bar{Z}_1$ is an orthogonal matrix of dimension $n_1 \times k_1$, hence $\bar{Z}_1 = I_{n_1}$. All the consistency results of the paper thus hold, where we set $k_1 = n_1$.

6.2. More clusters than rank

One of the unique features of the bipartite setting relative to the symmetric one is the possibility of having more clusters on one side of the network than the rank of the connectivity matrix. In the notation established so far, this is equivalent to $k_1 > k = \min\{k_1, k_2\}$. Let us first examine the performance of SC-1. Recall the SVD of $\bar{B} = U_\psi \Sigma V_\psi^T$, as given in Lemma 1. In contrast to the case $k_1 = k$, where the singular vector matrix U_ψ has no effect on the results (cf. Theorem 4), in the case $k_1 > k$, these singular vectors play a role. Recall that U_ψ is a $k_1 \times k$ orthogonal matrix, i.e., $U_\psi \in \mathbb{O}^{k_1 \times k}$. For a matrix $k_1 \times k_1$ matrix M, and index set $\mathcal{I} \subset [k_1]$, let $M_\mathcal{I}$ be the principal sub-matrix of M on indices $\mathcal{I} \times \mathcal{I}$. We assume the following *incoherence* condition:

$$\max_{\mathcal{I} \subset [k_1]:\, |\mathcal{I}|=2} \|(U_\psi U_\psi^T - I_{k_1})_\mathcal{I}\|_{\mathrm{op}} \;\leq\; 1 - \rho_1, \tag{44}$$

for some $\rho_1 \in (0, 1]$. Letting u_s^T be the sth row of U_ψ, for $s \in [k_1]$, we note that $(U_\psi U_\psi^T)_{st} = \langle u_s, u_t \rangle$, that is, $U_\psi U_\psi^T$ is the Gram matrix of the vectors u_s, $s \in [k_1]$. We have the following extension of Theorem 4:

Algorithm 5 SC-RRE for sub-Gaussian noise

1: Obtain $A^{(k)} = \hat{Z}_1 \hat{\Sigma} \hat{Z}_2^T$, the k-truncated SVD of A.
2: Output $\mathscr{K}(\hat{Z}_1 \hat{\Sigma})$ where $\mathscr{K}$ is an isometry-invariant κ-approximate k-means algorithm.

Theorem 7 *Consider the spectral algorithm* SC-1 *given in Algorithm 2. Assume that for a sufficiently small* $C > 0$,

$$kd\, \sigma_k^{-2} \leq C(1+\kappa)^{-2}\rho_1.$$

Then, under the SBM model of Section 2, w.h.p.,

$$\overline{\mathrm{Mis}}(\mathcal{P}_\kappa(\hat{Z}_1), \bar{Z}_1) \;\lesssim\; (1+\kappa)^2\beta_1 \Big(\frac{k}{\rho_1 k_1}\Big)\Big(\frac{d}{\sigma_k^2}\Big). \tag{45}$$

The theorem is proven in Appendix A.5. The factor $k/(k_1\rho_1) = k_2/(k_1\rho_1)$ in (45) is the price one pays for the asymmetry of the number of communities, when applying SC-1. (Recall that $k := \min\{k_1, k_2\} = k_2$ by assumption.) Note that increasing k_1 decreases k/k_1, and at the same time, often increases ρ_1 since it is harder to have many nearly orthogonal unit vectors in low dimensions.

Remark 11 *It is interesting to note that in contrast to* SC-1, *the consistency results for* SC-RR(E) *do not need any modification for the case where the number of clusters is larger than the rank. In other words, the same Theorems 5 and 6 hold regardless of whether* $k_1 = k$ *or* $k_1 > k$, *though the difficulty of the latter case will be reflected implicitly via a reduction in* $\Psi_{1,\wedge}^2$ *and* $\tilde{\Psi}_{1,\wedge}^2$.

6.3. General sub-Gaussian case

The analysis presented so far for network clustering problems can be extended to general sub-Gaussian similarity matrices. Consider a random matrix A with block constant mean

$$P := \mathbb{E}[A] = Z_1 B Z_2^T, \tag{46}$$

as defined in (4), and where $Z_r, r = 1, 2$ are again membership matrices. However, here A is not necessarily an adjacency matrix. We assume that A_{ij} are sub-Gaussian random variables independent across $(i, j) \in [n_1] \times [n_2]$ and let $\sigma := \max_{i,j} \|A_{ij} - \mathbb{E}A_{ij}\|_{\psi_2}$. We recall that a univariate random variable X is called sub-Gaussian if its sub-Gaussian norm is finite (Vershynin, 2018):

$$\|X\|_{\psi_2} := \inf\{t > 0 : \mathbb{E}\exp(X^2/t^2) \leq 2\} < \infty \tag{47}$$

Note that we do not assume A_{ij} and $A_{i'j'}$ to have the same distribution or the same sub-Gaussian norm even if $Z_{1i} = Z_{1i'}$ and $Z_{2j} = Z_{2j'}$.

Adapting Algorithm 4 to the general sub-Gaussian case, we have Algorithm 5 with the following performance guarantee:

Theorem 8 *Let $B_{1,\wedge}$ and $\widetilde{B}_{1,\wedge}$ be defined as in (35) with Ψ replaced with B and let*

$$\bar{n} := (n_1^{-1} + n_2^{-1})^{-1}.$$

Consider the spectral algorithm for sub-Gaussian noise given in Algorithm 5. Assume that for a sufficiently small $C_1 > 0$,

$$\frac{\sigma^2}{\bar{n}} k \, \widetilde{B}_{1,\wedge}^{-2} \le C_1 (1+\kappa)^{-2}. \tag{48}$$

Then, under the model defined in this section,

$$\overline{\mathrm{Mis}}(\mathscr{K}(\hat{Z}_1 \hat{\Sigma}), P) \;\le\; C_1^{-1} \, (1+\kappa)^2 \Big(\frac{\sigma^2}{\bar{n}} \frac{k}{B_{1,\wedge}^2} \Big).$$

with probability at least $1 - 2e^{-(n_1+n_2)}$.

See Appendix A.6 for the proof. Consider the typical case where the number of clusters k and the separation between them, $B_{1,\wedge}^2$, remains fixed as $\bar{n} \to \infty$. Then, $\sigma^2/\bar{n}$ plays the role of the signal-to-noise ratio (SNR) for the clustering problem and as long as $\sigma^2/\bar{n} = o(1)$, Algorithm 5 is consistent in recovering the clusters. Of course, there are other conditions under which we have consistency, e.g., cluster separation quantity $B_{1,\wedge}^2$ could go to zero, and k could grow as well (as $\bar{n} \to \infty$) and as long as $\sigma^2/\bar{n} = o(B_{1,\wedge}^2/k)$ the algorithm remains consistent.

6.4. Inhomogeneous random graphs

Up to now, we have stated consistency results for the SBM of Section 2. SBM is often criticized for having constant expected degree for nodes in the same community, in contrast to degree variation observed in real networks. (Although, SBM in the sparse regime $d = O(1)$ can still exhibit degree variation within a community, since the node degrees will be roughly $\mathrm{Poi}(d)$ distributed, showing little concentration around their expectations unless $d \to \infty$.) The popular remedy is to look at the degree-corrected block model (DC-SBM) (Karrer and Newman, 2011; Zhao et al., 2012; Gao et al., 2018). Instead, we consider the more general inhomogeneous random graph model (IRGM) (Söderberg, 2002; Bollobás et al., 2007) which might be more natural in practice, since it does not impose the somewhat parametric restrictions of DC-SBM on the mean matrix.

We argue that consistency results for the spectral clustering can be extended to a general inhomogeneous random graph (IRGM) model $A \sim \mathrm{Ber}(P)$, assuming that the mean matrix can be well-approximated by a block structure. Let us consider a scaling as before:

$$P := \frac{P^0}{\sqrt{n_1 n_2}}, \quad \text{and take} \quad d = \sqrt{\frac{n_2}{n_1}} \|P^0\|_\infty. \tag{49}$$

Note that $P \in [0,1]^{n_1 \times n_2}$ is a not assumed to have any block structure. However, we assume that there are membership matrices $Z_r \in \mathbb{H}_{n_r,k_r}$, $r = 1,2$ such that P over the blocks defined by Z_1 and Z_2 is approximately constant. Let

$$\widetilde{B}_{st} := \frac{1}{n_{1s} n_{2t}} \sum_{ij} P_{ij} (Z_1)_{is} (Z_2)_{jt}, \quad s \in [k_1], \; t \in [k_2] \tag{50}$$

be the mean (or average) of P over these blocks, and let $\widetilde{B} = (\widetilde{B}_{st}) \in [0,1]^{k_1 \times k_2}$. Compactly, $\widetilde{B} = N_1^{-1} Z_1^T P Z_2 N_2^{-1}$, in the notation established in Section 2. We can define the SBM approximation of P as

$$\widetilde{P} := Z_1 \widetilde{B} Z_2^T = \bar{Z}_1 \bar{Z}_1^T P \bar{Z}_2 \bar{Z}_2^T = \Pi_1 P \Pi_2, \tag{51}$$

recalling $\bar{Z}_r = Z_r N_r^{-1/2}$ and introducing the notation $\Pi_r = \bar{Z}_r \bar{Z}_r^T$. Note that Π_r is a rank k_r projection matrix. According to (51), the map $P \mapsto \Pi_1 P \Pi_2$ takes a (mean) matrix to its SBM approximation relative to Z_1 and Z_2. We can thus define a similar approximation to P^0,

$$\widetilde{P}^0 := \Pi_1 P^0 \Pi_2, \quad \text{noting that} \quad \widetilde{P} = \frac{\widetilde{P}^0}{\sqrt{n_1 n_2}}.$$

In order to retain the qualitative nature of the consistency results, the deviation of each entry P_{ij}^0 from its block mean $(\widetilde{P}^0)_{ij}$ should not be much larger than $(\widetilde{P}^0)_{ij}^{1/2}$. In fact, we allow for a potentially larger deviation, assuming that:

$$\frac{1}{\sqrt{n_1 n_2}} \|P^0 - \widetilde{P}^0\|_F \le C_0 \sqrt{d}, \tag{52}$$

for some constant $C_0 > 0$ and a satisfying (49). The key is the following concentration result:

Proposition 2 *Assume that $A \sim \mathrm{Ber}(P)$ as in (5), with $n_1 \le n_2$, and let A_{re} be obtained from A by the regularization procedure in Theorem 2, with $d' = d$ as given in (49). Let $\widetilde{P}$ be as defined in (51), and assume that it satisfies (52). Then, with probability at least $1 - n_2^{-c}$,*

$$\|A_{re} - \widetilde{P}\|_{op} \le c_2 \sqrt{d}. \tag{53}$$

Proposition 2 provides the necessary concentration bound required in Step 1 of the analysis outlined in Section 3.1. In other words, bound (53) replaces (8) by guaranteeing a similar order of deviation for A_{re} around $\widetilde{P}$ instead of P. Thus, all the results of the paper follow under an IRGM with a mean matrix satisfying (52), with the modification that the connectivity matrix and all the related quantities (such as σ_k, $\Psi_{1,\wedge}^2$ and so on) are now based on $\widetilde{B}$, the connectivity matrix of the corresponding approximate SBM, as given in (50).

6.4.1. Graphon clustering

As a concrete example of the application of Proposition 2, let us consider a problem which we refer to as *graphon clustering*. For simplicity, consider the case $n_1 = n_2 = n$. Let $\rho_0 : [0,1]^2 \to \mathbb{R}_+$ be a bounded measurable function, and let $X_1, \dots, X_n, Y_1, \dots, Y_n \sim \mathrm{Unif}([0,1])$ be an i.i.d. sample. Assume that

$$A_{ij} \mid \{X_{i'}\}, \{Y_{j'}\} \;\sim\; \mathrm{Bern}\big(\rho_0(X_i, Y_j)/n \big), \tag{54}$$

independently over $i, j = 1, \dots, n$. Consider two partitions of the unit interval: $\biguplus_{s=1}^{k_1} I_s = \biguplus_{t=1}^{k_2} J_t = [0,1]$ where I_s and J_t are (unknown) measurable sets. Let $\Psi \in [0,1]^{k_1 \times k_2}$ and

consider the block-constant function $\widetilde{\rho}_0$ on $[0,1]^2$ defined by

$$\widetilde{\rho}_0 := \sum_{s=1}^{k_1}\sum_{t=1}^{k_2} \Psi_{st} 1_{I_s \times J_t}, \tag{55}$$

where $1_{I_s \times J_t}(x,y) = 1_{I_s}(x)1_{J_t}(y)$ is the indicator of the set $I_s \times J_t$. Whenever ρ_0 has a block constant approximation such as $\widetilde{\rho}_0$, we can consider I_s and J_t as defining implicit (true) clusters over nodes. More precisely, row node i belongs to community s if $1\{X_i \in I_s\}$ and similarly for the column labels. Note that none of $\{X_i\}$, $\{Y_j\}$, $\{I_s\}$ and $\{J_t\}$ are observed. One can ask whether we can still recover these implicit clusters given an instance of A.

Proposition 3 *Assume that there exists a function $\widetilde{\rho}_0$ of the form (55), such that*

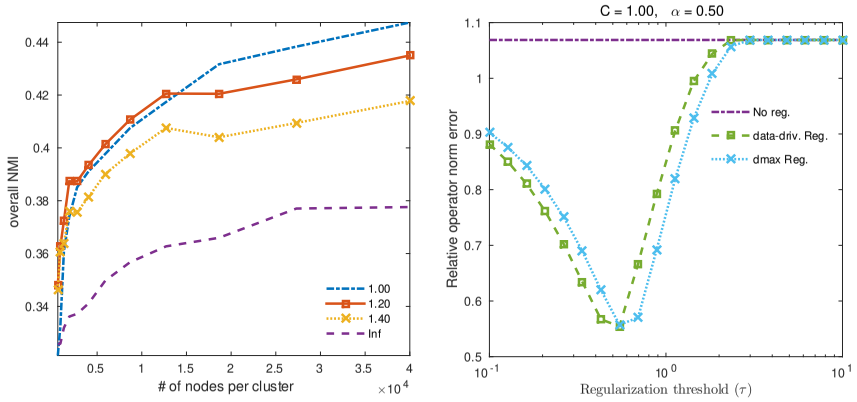
$$\|\widetilde{\rho}_0 - \rho_0\|_{L^4} = o(\sqrt{d}) \tag{56}$$

Consider normalized mean matrices P^0 and $\widetilde{P}^0$ with entries, $P^0_{ij} = \rho_0(X_i, Y_j)$ and $\widetilde{P}^0_{ij} = \widetilde{\rho}_0(X_i, Y_j)$. Then, with high probability $\|P^0 - \widetilde{P}^0\|_F \le n\sqrt{d}$.

Proposition 3 shows that a fourth moment bound of the form (56) is enough to guarantee (52) and as a consequence the result of Proposition 2. In order to apply the results of the paper, we require that (56) holds with $d = \|\rho_0\|_\infty$. Then, all the consistency results of the paper follow with Ψ replaced by that from (55), and regularization Algorithm 1 replaced with that of Theorem 2.

For example, with σ_k as defined in Section 2 (using Ψ from (55) in forming $\bar{B}$), Theorem 4 gives a misclassification rate at most $O(d/\sigma_k^2)$ for recovering the labels $1\{X_i \in I_s\}$ by the SC-1 algorithm. In typical cases, it is plausible to have $\sigma_k^2 \asymp d^2$ (cf. Remark 3), hence a misclassification rate of $O(1/d) = o(1)$ as $d \to \infty$. Proposition 3 is an illustrative example, and one can obtain other conditions by assuming more about the deviation $\widetilde{\rho}_0 - \rho_0$, such as boundedness.

Remark 12 *We note that graphon clustering problem considered above is different from what is typically called graphon estimation. In the latter problem, under a model of the form (54), one is interested in recovering the mean matrix $\mathbb{E}[A]$ or ρ_0 in the MSE sense. This problem has been studied extensively in recent years, often under the assumption of smoothness of ρ_0, for maximum likelihood SBM approximation (Airoldi et al., 2013; Olhede and Wolfe, 2014; Gao et al., 2015; Klopp et al., 2017) and spectral truncation (Xu, 2017). The graphon clustering problem, as far as we know, has not been considered before and is concerned with recovering the underlying clusters, assuming that such* true clusters exist. *We only need the existence of a block constant approximation $\widetilde{\rho}_0$, over the true clusters (of the form (55)) that satisfies $\|\widetilde{\rho}_0 - \rho_0\|_{L^4} = o(\sqrt{d})$ for $d = \|\widetilde{\rho}_0\|_\infty$. Then our results implicitly imply that the underlying clusters are identifiable and consistently recovered by spectral approaches. Note that we do not impose any explicit smoothness assumption on ρ and there is no lower bound requirement on d (such as $d = \Omega(\log n)$ in Xu (2017)).*



(a) (b)

Figure 2: (a) NMI plots for SC-RRE algorithm with degree regularization Algorithm 1 for various values of $\tau = 1, 1.2, 1.4$ and ∞. (b) Relative operator norm error between the adjacency matrix and its expectation, with data-driven and oracle ($d_{\max}$) regularization as well as no regularization.

7. Simulations

We now present some simulation results showing the performance of the data-driven regularization of Section 4.1. We sample from the bipartite SBM model with connectivity matrix

$$B = \sqrt{\frac{\log(n_1 n_2)}{n_1 n_2}} B_0, \quad \text{where} \quad B_0 = \frac{1}{2}\begin{bmatrix} 6 & 1 & 1 & 1 \\ 1 & 6 & 1 & 1 \\ 1 & 1 & 6 & 1 \end{bmatrix}, \tag{57}$$

for which $\Psi = \sqrt{\log(n_1 n_2)} B_0$ as in (4). We let $n_1 = n_0 k_1$ and $n_2 = n_0 k_2$, and we vary n_0. Note that $k_1 = 3$ and $k_2 = 4$. We measure the performance using the normalized mutual information (NMI) between the true and estimated clusters. The NMI belongs to the interval $[0,1]$ and is monotonically increasing with clustering accuracy. We consider Algorithm 1 with regularization parameter $\tau = 1, 1.2, 1.4$ and ∞, where $\tau = \infty$ corresponds to no regularization. Although we have established theoretical guarantees for $\tau = 3$, this constant is not optimal and any scalar $\asymp 1$ might perform well.

In this model, the key parameter $d = \sqrt{n_2/n_1}\|\Psi\|_\infty \sim \sqrt{\log n_0}$ as $n_0 \to \infty$. In other words, the maximum expected degree of the network scales as $\sqrt{\log n_0}$ which is enough for the consistency of spectral clustering; in fact, results in Section 5 predict a misclassification rate of $O(d^{-1}) = O\big((\log n_0)^{-1/2}\big)$ for various spectral algorithms discussed in this paper.

Figure 2(a) shows the NMI plots as a function of n_0 for the SC-RRE algorithm with the regularization scheme of Algorithm 1. In the k-means step, we have used `kmeans++` which as described in Remark 8 satisfies the approximation property of Section 3.3 with $\kappa = O(\log(k_1 \wedge k_2)) = O(1)$ in this case. The results are averaged over 15 replicates. The

plots clearly show that the regularization considerably boosts the performance of spectral clustering for model (57).

Figure 2(b) shows the relative operator norm error between the (regularized) adjacency matrix and its expectation, i.e., $\|A_{\rm re} - \mathbb{E}A\|_{\rm op}/\|\mathbb{E}A\|_{\rm op}$ with and without regularization ($A_{\rm re} = A$). The plots correspond to the same SBM model with $n_0 = 500$ (and the results are averaged over 3 replicates). For the regularization, we consider both the oracle where the degrees are truncated to $\tau d_{\max}$ and $\tau d'_{\max}$ for the rows and columns (see Section 4.1 for the definitions of these quantities) as well as the data-driven one that truncates as in Algorithm 1. The plots show the relative error as a function of τ and we see that the regularization clearly improves the concentration. We also note that the behavior of the data-driven truncation closely follows that of the oracle as predicted by Theorem 3.

The code is available at (Zhou and Amini, 2018).

Acknowledgement

We thank Zahra S. Razaee, Jiayin Guo and Yunfeng Zhang for helpful discussions. We are also grateful to the referees for their constructive suggestions.

Appendix A. Proofs

A.1. Proofs of Section 3.2

Proof [Proof of Lemma 3] Let $\bar{W}$ and $\widehat{W}$ be the W of Lemma 2(a) for $P^\dagger$ and $A^\dagger_{\rm re}$, respectively. Let us also write $\bar{W}_1$ and $\widehat{W}_1$ for the $(n_1+n_2)\times k$ matrices obtained by taking the submatrices of $\bar{W}$ and $\widehat{W}$ on columns $1,\dots,k$. We have

$$\bar{W}_1 = \frac{1}{\sqrt{2}}\begin{pmatrix}\bar{Z}_1 U_\psi \\ \bar{Z}_2 V_\psi\end{pmatrix}, \quad \widehat{W}_1 = \frac{1}{\sqrt{2}}\begin{pmatrix}\hat{Z}_1 \\ \hat{Z}_2\end{pmatrix}.$$

Note that $\bar{W}_1, \widehat{W}_1 \in \mathbb{O}^{(n_1+n_2)\times k}$. Let $\Pi_{\bar{W}_1}$ be the (orthogonal) projection operator, projecting onto $\mathrm{Im}(\bar{W}_1)$, i.e., the column span of $\bar{W}_1$, and similarly for $\Pi_{\widehat{W}_1}$. We have

$$\begin{aligned}
\|\Pi_{\widehat{W}_1} - \Pi_{\bar{W}_1}\|_{\rm op} &\le \frac{2}{2\sigma_k}\|A^\dagger_{\rm re} - P^\dagger\|_{\rm op} && \text{(Symmetric DK and Lemma 2(d))}\\
&= \frac{1}{\sigma_K}\|(A_{\rm re} - P)^\dagger\|_{\rm op} && \text{(Linearity of dilation)}\\
&= \frac{1}{\sigma_K}\|A_{\rm re} - P\|_{\rm op} && \text{(Lemma 2(b)).}
\end{aligned}$$

The next step is to translate the operator norm bound on spectral projections into a Frobenius bound. The key here is the bound on the rank of spectral deviations which leads to a $\sqrt{k}$ scaling as opposed to $\sqrt{n_1+n_2}$, when translating from operator norm to Frobenius:

$$\begin{aligned}
\min_{Q\in\mathbb{O}^{k\times k}} \|\widehat{W}_1 - \bar{W}_1 Q\|_F &\le \|\Pi_{\widehat{W}_1} - \Pi_{\bar{W}_1}\|_F && \text{(By Lemma 8 in Appendix C)}\\
&\le \sqrt{2k}\,\|\Pi_{\widehat{W}_1} - \Pi_{\bar{W}_1}\|_{\rm op} && (\,\mathrm{rank}(\Pi_{\widehat{W}_1} - \Pi_{\bar{W}_1}) \le 2k\,)\\
&\le \frac{\sqrt{2k}}{\sigma_k}\|A_{\rm re} - P\|_{\rm op}.
\end{aligned}$$

Since $2\|\widehat{W}_1 - \bar{W}_1 Q\|_F^2 = \|\hat{Z}_1 - \bar{Z}_1 U_\psi Q\|_F^2 + \|\hat{Z}_2 - \bar{Z}_2 V_\psi Q\|_F^2$, we obtain the desired result after combining with (8). ■

A.2. Proofs of Section 4

Let us start with a relatively well-known concentration inequality:

Proposition 4 (Prokhorov) *Let $S = \sum_i X_i$ for independent centered variables $\{X_i\}$, each bounded by $c < \infty$ in absolute value a.s. and suppose $v \ge \sum_i \mathbb{E}X_i^2$, then*

$$\mathbb{P}\big(S > vt\big) \le \exp[-vh_c(t)], \quad t \ge 0, \quad \textit{where } h_c(t) := \frac{3}{4c}t\log\big(1 + \frac{2c}{3}t\big). \tag{58}$$

Same bound holds for $\mathbb{P}(S < -vt)$.

We often apply this result with $c = 1$. We note that for any $u \ge \frac{2}{3}\alpha$ any $\alpha > 0$,

$$h_1(\alpha u) \ge \frac{3}{4}\alpha u \log\Big(\frac{2}{3}\alpha u\Big) \ge \frac{3}{4}\alpha u \log u^2 \ge \frac{3}{2}\alpha u \log u. \tag{59}$$

Proof [Proof of Lemma 4] We first note that

$$\mathbb{P}(D_i - d_i > vt) \le \exp(-vh_1(t)), \quad \forall v \ge d_i,\ t \ge 0, \tag{60}$$
$$\mathbb{P}(D_i - d_i < -vt) \le \exp(-vh_1(t)), \quad \forall v \ge d_i,\ t \ge 0. \tag{61}$$

Lower bound. Let $Y_i = 1\{D_i < d_{\max}/2\}$. Without loss of generality, assume that cluster 1 achieves the maximum in the definition of $d_{\max}$ (i.e., $t^* = 1$), and fix i such that $z_{1i} = 1$. We have $d_i = d_{\max}$, hence applying (61) with $v = d_i = d_{\max}$ and $t = 1/2$,

$$\mathbb{P}(D_i < d_{\max}/2) \ \le\ \exp(-d_{\max}/10),$$

using $h_1(1/2) = 0.2157 > 1/10$. Let $q := \exp(-d_{\max}/10)$ so that $\mathbb{E}Y_i \le q$. By assumption $d_{\max} \ge \bar{d}$ is sufficiently large that $q \le 1/4$. We then have

$$\begin{aligned}
\mathbb{P}(D_{(\lfloor n_1/\overline{D}\rfloor)} < d_{\max}/2) &\le \mathbb{P}\Big(\sum_{i=1}^{n_1} 1\{D_i \ge d_{\max}/2\} \le n_1/\overline{D}\Big) \le \mathbb{P}\Big(\sum_{i:z_{1i}=1} 1\{D_i \ge d_{\max}/2\} \le n_1/\overline{D}\Big)\\
&= \mathbb{P}\Big(\sum_{i:z_{1i}=1} Y_i > n_{11} - n_1/\overline{D}\Big)\\
&\le \mathbb{P}\Big(\sum_{i:z_{1i}=1} Y_i > n_{11} - 2n_1/\bar{d}\Big) + \mathbb{P}\Big(\overline{D} < \frac{\bar{d}}{2}\Big) \ =: T_1 + T_2
\end{aligned}$$

For the first term we have

$$T_1 \le \mathbb{P}\Big(\sum_{i:z_{1i}=1}(Y_i - \mathbb{E}Y_i) > n_{11}(1-q) - 2n_1/\bar{d}\Big).$$

By assumptions $n_{11} \geq n_1/(\beta k_1)$ and $\beta k_1/\bar{d} \leq 1/8$, we have $n_{11}(1-q) - 2n_1/\bar{d} \geq n_{11}/2$. Taking $c = 1$, $v = n_{11}q$, $t = 1/(2q)$ in Proposition 4, we have $vt = n_{11}/2$ and $t \geq 2$

$$T_1 \leq \mathbb{P}\Big(\sum_{i:\, z_{1i}=1} (Y_i - \mathbb{E}Y_i) > n_{11}/2 \Big) \leq \exp\Big(-n_{11}\, q\, h_1\big(\frac{1}{2q}\big) \Big) \\ \leq \exp\big(-c_1\, n_{11} d_{\max} \big) \leq e^{-3n_1/5}$$

where we have applied (59) with $\alpha = 1/2$ and $u = 1/q = e^{d_{\max}/10}$ to get

$$q\, h_1\big(\frac{1}{2q}\big) \geq q \frac{3}{2} \frac{1}{2q} \log(1/q) = \frac{3}{40} d_{\max}$$

and $n_{11} d_{\max} \geq n_1 d_{\max}/(\beta k_1) \geq 8 n_1$ since $\beta k_1/d_{\max} \leq \beta k_1/\bar{d} \leq 1/8$ by assumption.

For the second term T_2, we note that $n_1(\overline{D} - \bar{d}) = \sum_{i,j}(A_{ij} - \mathbb{E}A_{ij})$. Applying Proposition 4 with $v = n_1\bar{d}$ and $t = 1/2$, we have

$$T_2 = \mathbb{P}\Big(n_1(\overline{D} - \bar{d}) < -n_1\bar{d}/2 \Big) \leq \exp(-n_1\bar{d}/10) \leq e^{-n_1/5}. \tag{62}$$

using $h_1(1/2) \geq 1/10$ and $\bar{d} \geq 2$.

Upper bound. For any $\in [n_1]$, applying (60) with $v = d_{\max} \geq d_i$, and $t = 1/2$, we have

$$\mathbb{P}(D_i > 3d_{\max}/2) \;\leq\; \exp(-d_{\max}/10).$$

Let $Z_i = 1\{D_i > 3d_{\max}/2\}$ and note that $\mathbb{E}Z_i \leq q = \exp(-d_{\max}/10)$, as in the case of the lower bound. Then,

$$\mathbb{P}\Big(D_{(\lfloor n_1/\overline{D} \rfloor)} > \frac{3}{2} d_{\max} \Big) \leq \mathbb{P}\Big(\sum_{i=1}^{n_1} Z_i \geq \frac{n_1}{\overline{D}} - 1 \Big) \\ \leq \mathbb{P}\Big(\sum_{i=1}^{n_1} Z_i \geq \frac{2n_1}{3\bar{d}} - 1 \Big) + \mathbb{P}\Big(\overline{D} > \frac{3\bar{d}}{2} \Big) =: T_1' + T_2'.$$

Using assumption $n_1/\bar{d} \geq 2$, we have $2n_1/(3\bar{d}) - 1 \geq n_1/(6\bar{d}) \geq n_1/(6d_{\max})$, hence

$$T_1' \;\leq\; \mathbb{P}\Big(\sum_{i=1}^{n_1} (Z_i - \mathbb{E}Z_i) \geq \frac{n_1}{6d_{\max}} - n_1 q \Big) =: T_1'' \tag{63}$$

We claim that if $d_{\max} \geq 1$, then for any $\gamma \geq 1$,

$$f(\gamma) := \mathbb{P}\Big(\sum_{i=1}^{n_1} (Z_i - \mathbb{E}Z_i) \geq \frac{n_1}{\gamma d_{\max}} \Big) \leq e^{-3n_1/(20\gamma)}. \tag{64}$$

To see this, taking $c = 1$, $v = n_1 q$, $t = 1/(\gamma q d_{\max})$ in Proposition 4, we have

$$f(\gamma) \leq \exp\Big(-n_1 q\, h_1\Big(\frac{1}{\gamma q d_{\max}} \Big) \Big).$$

Applying (59) with $\alpha = 1/(\gamma d_{\max})$ and $u = 1/q = e^{d_{\max}/10}$, noting that $u \geq (2/3)\alpha$,

$$q\, h_1\big(\frac{1}{\gamma q d_{\max}}\big) \geq q \frac{3}{2} \frac{1}{\gamma q d_{\max}} \log(1/q) = \frac{3}{20\gamma}$$

showing (64).

Going back to bounding T_1'', for sufficiently large $d_{\max}$, we have $1/(6d_{\max}) - q \geq 1/(12 d_{\max})$. Hence, we can apply (64) with $\gamma = 12$ to conclude, $T_1'' \leq e^{-n_1/80}$. For T_2', by the same argument as in (62), we have $T_2 \leq e^{-n_1/5}$.

Proof of part (b). From the lower bound, we have that $3D_{(\lfloor n_1/\overline{D} \rfloor)} > \frac{3}{2} d_{\max}$ on an event $\mathcal{A}$ with probability $\mathbb{P}(\mathcal{A}) \geq 1 - 2e^{-n_1/80}$. Then,

$$W_i := 1\big\{ D_i > 3D_{(\lfloor n_1/\overline{D} \rfloor)} \big\} \;\leq\; 1\{D_i > 3d_{\max}/2\} = Z_i, \quad \text{on } \mathcal{A}.$$

Hence, $\big| i : D_i > 3D_{(\lfloor n_1/\overline{D} \rfloor)} \big| = \sum_i W_i$ and

$$\mathbb{P}\Big(\Big\{ \sum_{i=1}^{n_1} W_i > \frac{10 n_1}{d} \Big\} \cap \mathcal{A} \Big) \leq \mathbb{P}\Big(\sum_{i=1}^{n_1} Z_i > \frac{10n_1}{d} \Big) = \mathbb{P}\Big(\sum_{i=1}^{n_1} (Z_i - \mathbb{E}Z_i) > \frac{10n_1}{d} - n_1 q \Big).$$

Since $d_{\max} \geq n_2 \|P\|_\infty/(\beta k_2) = d/(\beta k_2)$, and $\beta k_2 \leq \bar{d}/8 \leq d_{\max}/8$ we have $d_{\max}^2 \geq 8d$. Recalling that $q = e^{-d_{\max}/10}$

$$qd = (qd_{\max}^2)(d/d_{\max}^2) \leq 1/8. \tag{65}$$

for $d_{\max} \geq 90$. We have

$$\frac{10n_1}{d} - n_1 q \geq \frac{9n_1}{d} \geq \frac{9n_1}{d_{\max}} \frac{1}{\beta k_2}$$

hence we can apply (64) with $\gamma = \beta k_2/9$ to obtain

$$\mathbb{P}\Big(\sum_{i=1}^{n_1} W_i > \frac{10n_1}{d} \Big) \leq \exp\Big(-\frac{27 n_1}{20 \beta k_2} \Big).$$

The proof is complete. ■

Proof [Proof of Lemma 5] The proof of part (a) follows exactly as in the case of row degrees establishing the result with probability at least $1 - 3e^{-n_2/80}$. For part (b), we have by the same argument as in the case of row degrees

$$\mathbb{P}\Big(\big| j : D_j' > 3D'_{(\lfloor n_2/\overline{D}' \rfloor)} \big| > \frac{10n_2}{d} \Big) \leq \mathbb{P}\Big(\sum_{j=1}^{n_2} (Z_j' - \mathbb{E}Z_j') > \frac{10n_2}{d} - n_2 q' \Big) + \mathbb{P}(\mathcal{A}'^c) \\ := T_3 + \mathbb{P}(\mathcal{A}'^c)$$

where $\mathcal{A}'$ is defined similar to $\mathcal{A}$ for column variables and is controlled similarly. We also have $q' := \exp(-d'_{\max}/10)$.

To simplify notation, let $\alpha := n_2/n_1 \geq 1$. We have $d'_{\max} \geq n_1\|P\|_\infty/(\beta k_1) = d/(\alpha\beta k_1)$, recalling $d = n_2\|P\|_\infty$. By assumption, $\bar{d} \geq 8\beta k_1\alpha^2$. Since $n_1\bar{d} = \sum_{ij} p_{ij} = n_2\bar{d}'$, that is, $\bar{d} = \alpha\bar{d}'$, we obtain $\beta k_1 \leq \bar{d}/(8\alpha^2) = \bar{d}'/(8\alpha) \leq d'_{\max}/(8\alpha)$. It follows that $(d'_{\max})^2 \geq 8d$ which is similar to what we had for the rows; hence, $q'd \leq 1/8$ as long as $d'_{\max} \geq 90$ (see (65)), which is true since $d'_{\max} \geq \bar{d}' = \bar{d}/\alpha \geq \bar{d}/\alpha^2 \geq 90$ by assumption. Then,

$$\frac{10n_2}{d} - n_2 q' \geq \frac{9n_2}{d} \geq \frac{9n_2}{d'_{\max}}\frac{1}{\alpha\beta k_1}.$$

Therefore, applying the column counterpart of (64) with $\gamma = \alpha\beta k_1/9$,

$$T_3 \leq \exp\Big(-\frac{27n_2}{20\alpha\beta k_1}\Big) = \exp\Big(-\frac{27n_1}{20\beta k_1}\Big).$$

The proof is complete. ■

A.3. Proofs of Section 3.3

Proof [Proof of Corollary 1] Using (21), we have $d_F(X^*, \widetilde{X}) \leq (1+\kappa)\varepsilon$ for any $\widetilde{X} \in \mathcal{P}_\kappa(\hat{X})$. We now apply Proposition 1 to X^* and $\widetilde{X}$, both k-means matrices, with $(1+\kappa)\varepsilon$ in place of ε. ■

Proof [Proof of Proposition 1] The proof follows the argument in (Lei et al., 2015, Lemma 5.3) which is further attributed to Jin (2015). Let $\mathcal{C}_r$ denote the rth cluster of X, having center $q_r = q_r(X)$. We have $|\mathcal{C}_r| = n_r$. Let x_i^T and $\widetilde{x}_i^T$ be the ith row of X and $\widetilde{X}$, respectively, and let

$$T_r := \{i \in \mathcal{C}_r :\ d(\widetilde{x}_i, q_r) < c_r\delta_r\} = \{i \in \mathcal{C}_r :\ d(\widetilde{x}_i, x_i) < c_r\delta_r\}$$

using $x_i = q_r$ for all $i \in \mathcal{C}_r$ which holds by definition. Let $S_r = \mathcal{C}_r \setminus T_r$. Then,

$$|S_r|c_r^2\delta_r^2 \ \leq \sum_{i\in S_r} d(\widetilde{x}_i, x_i)^2 \leq \varepsilon^2 \implies \frac{|S_r|}{|\mathcal{C}_r|} \leq \frac{c_r^{-2}\varepsilon^2}{n_r\delta_r^2} < 1. \tag{66}$$

where we have used assumption (b). It follows that S_r is a proper subset of $\mathcal{C}_r$, that is, T_r is nonempty for all $r \in [k]$.

Next, we argue that if two elements belong to different $T_r, r \in [k]$, they have different labels according to $\widetilde{X}$. That is, $i \in T_r$, $j \in T_\ell$ for $r \neq \ell$ implies $\widetilde{x}_i \neq \widetilde{x}_j$. Assume otherwise, that is, $\widetilde{x}_i = \widetilde{x}_j$. Then, by triangle inequality and $c_r + c_\ell \leq 1$,

$$d(q_k, q_\ell) \leq d(q_k, \widetilde{x}_i) + d(q_\ell, \widetilde{x}_j) < c_r\delta_r + c_\ell\delta_\ell \leq \max\{\delta_r, \delta_\ell\}$$

contradicting (23). This shows that $\widetilde{X}$ has at least k labels, since all T_r are nonempty, hence exactly k labels, since $\widetilde{X} \in \mathbb{M}^k_{n,m}$ by assumption.

Finally, we argue that if two elements belong to the same T_r, they have the same label according to $\widetilde{X}$. This immediately follows from the previous step since otherwise there will

be at least $k+1$ labels. Thus, we have shown that, for all $r \in [k]$, the labels in each T_r are in the same cluster according to both X and $\widetilde{X}$, that is, they are correctly classified. The misclassification rate over cluster $\mathcal{C}_r$ is then $\leq |S_r|/|\mathcal{C}_r|$ which establishes the result in view of (66). ■

A.4. Proofs of Section 5

Proof [Proof of Theorem 4] Going through the three-step plan of analysis in Section 3, we observe that (8) holds for $A_{\rm re}$ by Theorem 2, and (9) holds by Lemma 3. We only need to verify conditions of Corollary 1, so that κ-approximate k-means operator $\mathcal{P}_\kappa$ satisfies bound (10) of the k-means step. As in the proof of Theorem 1, $X^* = \bar{Z}_1 O \in \mathbb{R}^{n_1\times k}$, where $\bar{Z}_1 \in \mathbb{O}^{n_1\times k_1}$ and $O := U_\psi Q \in \mathbb{O}^{k_1\times k}$. Clearly, X^* has exactly k distinct rows (recalling $k = k_1$). Furthermore, using the calculation in the proof of Theorem 1,

$$n_{1t}\,\delta_t^2 = n_{1t}\min_{s:\ s\neq t}(n_{1t}^{-1} + n_{1s}^{-1}) = \min_{s:\ s\neq t}(1 + \frac{n_{1t}}{n_{1s}}) \geq 1.$$

Recalling that $\varepsilon^2 = C_2^2 kd/\sigma_k^2$, as long as

$$4(1+\kappa)^2\varepsilon^2 = 4C_2^2(1+\kappa)^2\, kd/\sigma_k^2 < 1 \leq n_{1t}\,\delta_t^2$$

condition (b) of Corollary 1 holds and $\mathcal{P}_\kappa$ satisfies (10) with $c = 4(1+\kappa)^2$ as in (26). The rest of the proof follows as in Theorem 1. ■

Proof [Proof of Lemma 6] Throughout the proof, let $\|\cdot\| = \|\cdot\|_{\rm op}$ be the operator norm. Recall that $P = \mathbb{E}A$ is the mean matrix itself, and let $\Delta_{\rm re} := A_{\rm re} - P$. By Weyl's theorem on the perturbation of singular values, $|\sigma_i(A_{\rm re}) - \sigma_i(P)| \leq \|\Delta_{\rm re}\|$ for all i. Since $\sigma_{k+1}(P) = 0$ (see (7)), we have $\sigma_{k+1}(A_{\rm re}) \leq \|\Delta_{\rm re}\|$, hence

$$\begin{aligned}
\|A_{\rm re}^{(k)} - P\| &\leq \|A_{\rm re}^{(k)} - A_{\rm re}\| + \|\Delta_{\rm re}\| && \text{(triangle inequality)}\\
&= \sigma_{k+1}(A_{\rm re}) + \|\Delta_{\rm re}\| && \text{(by (33))}\\
&\leq 2\|\Delta_{\rm re}\| && \text{(Weyl's theorem).}
\end{aligned}$$

Thus, in terms of the operator norm, we lose at most a constant in going from $A_{\rm re}$ to $A_{\rm re}^{(k)}$. However, we gain a lot in Frobenious norm deviation. Since $A_{\rm re}$ is full-rank in general, the best bound on $\Delta_{\rm re}$ based on its operator norm is $\|\Delta_{\rm re}\|_F \leq \sqrt{n_\wedge}\,\|\Delta_{\rm re}\|$ where $n_\wedge = \min\{n_1, n_2\}$. On the other hand, since $A_{\rm re}^{(k)} - P$ is of rank $\leq 2k$, we get

$$\|A_{\rm re}^{(k)} - P\|_F \leq \sqrt{2k}\|A_{\rm re}^{(k)} - P\| \leq 2\sqrt{2k}\|\Delta_{\rm re}\|.$$

Combining with (8), that is, $\|\Delta_{\rm re}\| \leq C\sqrt{d}$, we have the result. ■

Proof [Proof of Theorem 5] We only need to calculate $\delta(P) = \delta_\wedge(P)$ the minimum center separation of P viewed as an element of $\mathbb{M}^{k_1}_{n_1,n_2}$. Recall that

$$P = \bar{Z}_1\bar{B}\bar{Z}_2^T = Z_1 N_1^{-1/2}(\bar{N}_1^{1/2}\Psi\bar{N}_1^{1/2})\bar{Z}_2^T = n_1^{-1/2}Z_1\Psi\bar{N}_1^{1/2}\bar{Z}_2^T.$$

Let e_s be the sth standard basis vector of $\mathbb{R}^{k_1}$. Unique rows of P are $q_s^T := n_1^{-1/2} e_s^T (\Psi \bar{N}_2^{1/2}) \bar{Z}_2^T$ for $s \in [k_1]$. We have

$$\begin{aligned}\|q_s - q_t\|_2^2 &= n_1^{-1} \|\bar{Z}_2 \bar{N}_2^{1/2} \Psi^T (e_s - e_t)\|_2^2 \\ &= n_1^{-1} \|\bar{N}_2^{1/2} \Psi^T (e_s - e_t)\|_2^2 = n_1^{-1} \sum_{\ell=1}^{k_2} \pi_{2\ell} (\Psi_{s\ell} - \Psi_{t\ell})^2.\end{aligned}$$

It follows that $\delta^2(P) = \min_{t \neq s} \|q_s - q_t\|_2^2 = n_1^{-1} \Psi_{1,\wedge}^2$. We apply Corollary 1, with $X^* = P$ and $\hat{X} = A_{\mathrm{re}}^{(k)}$, taking $\varepsilon^2 = 8C^2 kd$ according to Lemma 6. Condition (b) of the corollary holds if

$$32C^2(1+\kappa)^2 kd = 4(1+\kappa)^2 \varepsilon^2 < n_{1t}\, \delta_t^2(P) = \pi_{1t} \min_{s:\, s \neq t} \sum_{\ell=1}^{k_2} \pi_{2\ell} (\Psi_{s\ell} - \Psi_{t\ell})^2$$

for all $t \in [k_1]$, which is satisfied under assumption (37). Corollary 1, and specifically (26) gives the desired bound on misclassification rate $\leq 4(1+\kappa)^2 \varepsilon^2 / (n_1 \delta^2(P))$. ■

Proof [Proof of Theorem 6] Recall that $A_{\mathrm{re}}^{(k)} = \hat{Z}_1 \hat{\Sigma} \hat{Z}_2^T$ is the k-truncated SVD of A_{re}. Let $X^{(1)} = \hat{Z}_1 \hat{\Sigma}$ and $X^{(2)} = A_{\mathrm{re}}^{(k)}$, and let $(x_i^{(1)})^T$ and $(x_i^{(2)})^T$ be their ith rows, respectively. Then,

$$\|x_2^{(i)} - x_2^{(j)}\| = \|\hat{Z}_2 (x_1^{(i)} - x_1^{(j)})\|_2 = \|x_1^{(i)} - x_1^{(j)}\|_2, \quad \forall i \neq j,$$

using $\hat{Z}_2 \in \mathbb{O}^{n_2 \times k}$ and (1). Isometry-invariance of $\mathscr{K}$ implies $\overline{\mathrm{Mis}}(\mathscr{K}(\hat{Z}_1 \hat{\Sigma}), \mathscr{K}(A_{\mathrm{re}}^{(k)})) = 0$. Since $\overline{\mathrm{Mis}}$ is a pseudo-metric on k-means matrices, using the triangle inequality, we get

$$\begin{aligned}\overline{\mathrm{Mis}}\left(\mathscr{K}(\hat{Z}_1 \hat{\Sigma}), P\right) &\leq \overline{\mathrm{Mis}}\left(\mathscr{K}(\hat{Z}_1 \hat{\Sigma}), \mathscr{K}(A_{\mathrm{re}}^{(k)})\right) + \overline{\mathrm{Mis}}\left(\mathscr{K}(A_{\mathrm{re}}^{(k)}), P\right) \\ &= \overline{\mathrm{Mis}}\left(\mathscr{K}(A_{\mathrm{re}}^{(k)}), P\right).\end{aligned}$$

(In fact, using the triangle inequality in the other direction, we conclude that the two sides are equal.) The result now follows from Theorem 5. ■

Proof [Proof of Corollary 4] We have

$$\|\Lambda_{s*} - \Lambda_{t*}\|^2 = \sum_{\ell=1}^{k_2} n_{2\ell}^2 (B_{s\ell} - B_{t\ell})^2 = \sum_{\ell=1}^{k_2} \frac{n_{2\ell}^2}{n_1 n_2} (\Psi_{s\ell} - \Psi_{t\ell})^2 = \alpha \sum_{\ell=1}^{k_2} \pi_{2\ell}^2 (\Psi_{s\ell} - \Psi_{t\ell})^2$$

where we have used $\alpha := n_2/n_1$. Recall from (42) that $\Lambda_\wedge^2 := \min_{t \neq s} \|\Lambda_{s*} - \Lambda_{t*}\|^2$. It follows that $\alpha^{-1} \Lambda_\wedge^2 \leq \Psi_{1,\wedge}^2$, using $\pi_{2,\ell} \leq 1$. We also have $\widetilde{\Psi}_{1,\wedge}^2 \geq \pi_{1,\wedge} \Psi_{1,\wedge}^2 \geq \pi_{1,\wedge} \alpha^{-1} \Lambda_\wedge^2$. Recalling the definition of a from (8), and using $\Psi_{s\ell} = (\sqrt{n_1 n_2}/n_{2\ell}) \Lambda_{s\ell}$, we have

$$d = \sqrt{\frac{n_2}{n_1}} \|\Psi\|_\infty \leq \sqrt{\frac{n_2}{n_1}} \frac{\sqrt{n_1 n_2}}{n_{2,\wedge}} \|\Lambda\|_\infty = \frac{1}{\pi_{2,\wedge}} \|\Lambda\|_\infty \leq \beta_2 k_2 \|\Lambda\|_\infty.$$

Hence, $kd\Psi_{1,\wedge}^{-2} \leq k\, \beta_2 k_2 \|\Lambda\|_\infty (\alpha^{-1} \Lambda_\wedge^2)^{-1} = \beta_2 k k_2 \alpha \|\Lambda\|_\infty \Lambda_\wedge^{-2}$ which is the desired bound. We also note that

$$kd\widetilde{\Psi}_{1,\wedge}^{-2} \leq kd\pi_{1,\wedge}^{-1} \Psi_{1,\wedge}^{-2} \leq (\beta_1 k_1)\, kd\Psi_{1,\wedge}^{-2}$$

which combined with the previous bound shows that the required condition (43) in the statement is enough to satisfy (37). ■

A.5. Proof of Theorem 7

As in the proof of Theorem 4, we only need to verify conditions of Corollary 1, with $X^* = \bar{Z}_1 U_\psi Q \in \mathbb{R}^{n_1 \times k}$, so that κ-approximate k-means operator $\mathcal{P}_\kappa$ satisfies bound (10) of the k-means step. Recall that $\bar{Z}_1 \in \mathbb{O}^{n_1 \times k_1}$, $U_\psi \in \mathbb{O}^{k_1 \times k}$ and $Q \in \mathbb{O}^{k \times k}$. Clearly, X^* has exactly k_1 distinct rows, $u_s^T Q$, $s \in [k_1]$. Furthermore,

$$\delta^2 := \delta_\wedge^2(\bar{Z}_1 U_\psi Q) = \delta_\wedge^2(\bar{Z}_1 U_\psi) = \min_{(t,s):\, t \neq s} \|n_{1t}^{-1/2} u_t - n_{1s}^{-1/2} u_s\|_2^2$$

where the second equality is by (1). Letting $\|\cdot\| = \|\cdot\|_2$, and $x = \pi_{1t}^{-1/2}$ and $y = \pi_{1s}^{-1/2}$ for simplicity, and $z = (x, -y) \in \mathbb{R}^2$, we have

$$\begin{aligned}n_1 \|n_{1t}^{-1/2} u_t - n_{1s}^{-1/2} u_s\|^2 &= \|\pi_{1t}^{-1/2} u_t - \pi_{1s}^{-1/2} u_s\|^2 \\ &= \| \begin{pmatrix} u_t & u_s \end{pmatrix} \begin{pmatrix} x \\ -y \end{pmatrix} \|^2 \\ &= z^T (U_\psi U_\psi^T)_{\mathcal{I}}\, z, \qquad \text{(where } \mathcal{I} = \{t, s\}\text{)}, \\ &= \|z\|^2 - z^T (I_{k_1} - U_\psi U_\psi^T)_{\mathcal{I}}\, z^T.\end{aligned}$$

Since $I_{k_1} - U_\psi U_\psi^T$ is positive semidefinite, using (44), we have

$$0 \leq z^T (I_{k_1} - U_\psi U_\psi^T)_{\mathcal{I}}\, z^T \leq (1 - \rho_1) \|z\|^2,$$

hence, $n_1 \|n_{1t}^{-1/2} u_t - n_{1s}^{-1/2} u_s\|^2 \geq \rho_1 \|z\|^2 = \rho_1 (\pi_{1t}^{-1} + \pi_{1s}^{-1})$. It follows by (A1),

$$(n_1 \delta^2)^{-1} \leq \frac{1}{\rho_1} \max_{t \neq s} (\pi_{1t}^{-1} + \pi_{1s}^{-1})^{-1} \leq \frac{1}{\rho_1} \Big(\frac{\beta_1}{2k_1} \Big).$$

We also have

$$n_{1t}\, \delta_t^2 = n_{1t} \min_{s:\, s \neq t} \|n_{1t}^{-1/2} u_t - n_{1s}^{-1/2} u_s\|_2^2 \;\geq\; \rho_1 \big(1 + \frac{\pi_{1t}}{\pi_{1s}}\big) \;\geq\; \rho_1.$$

We obtain, with $\varepsilon^2 = C_2^2 kd/\sigma_k^2$ and $O = U_\psi Q$,

$$\overline{\mathrm{Mis}}(\mathscr{K}(\hat{Z}_1), \bar{Z}_1) = \overline{\mathrm{Mis}}(\mathscr{K}(\hat{Z}_1), \bar{Z}_1 O) \;\lesssim\; \frac{\varepsilon^2}{n_1 \delta^2} \;\lesssim\; \frac{\beta_1}{\rho_1} \frac{k}{k_1} \frac{d}{\sigma_k^2}$$

as long as $4(1+\kappa)^2 (n_{1t}\, \delta_t^2)^{-1} \varepsilon^2 = 4C_2^2 (1+\kappa)^2 \rho_1^{-1}\, kd/\sigma_k^2 < 1$, to guarantee that condition (b) of Corollary 1 holds, so that $\mathcal{P}_\kappa$ satisfies (10) with $c = 4(1+\kappa)^2$ as in (26). The proof is complete.

A.6. Proof of Theorem 8

The following concentration result for sub-Gaussian random matrices is well-known (Vershynin, 2018):

Lemma 7 *Let $A \in \mathbb{R}^{n_1 \times n_2}$ have independent sub-Gaussian entries, and $\sigma = \max_{i,j} \|A_{ij} - \mathbb{E}A_{ij}\|_{\psi_2}$. Then for any $t > 0$, we have*

$$\|A - \mathbb{E}A\|_{op} \le C\,\sigma(\sqrt{n_1} + \sqrt{n_2} + t), \tag{67}$$

with probability at least $1 - 2\exp(-t^2)$.

Let $t = \sqrt{n_1} + \sqrt{n_2}$ in Lemma 7, then $\|A - \mathbb{E}A\|_{\rm op} \le 2C\sigma(\sqrt{n_1} + \sqrt{n_2})$ with probability at least $1 - 2e^{-(n_1+n_2)}$. We thus have the concentration bound (8) with $\sqrt{d} = 2\sigma(\sqrt{n_1} + \sqrt{n_2})$. The proof of Theorem 5 goes through and the result follows by replacing d with the upper bound $d \le 4\sigma^2(n_1 + n_2)$, and replacing $\Psi^2_{1,\wedge}$ and $\widetilde{\Psi}^2_{1,\wedge}$ with $n_1 n_2 B^2_{1,\wedge}$ and $n_1 n_2 \widetilde{B}^2_{1,\wedge}$, respectively.

A.7. Proofs of Section 6.4

Proof [Proof of Proposition 2] We have with the given probability,

$$\begin{aligned}\|A_{\rm re} - \widetilde{P}\|_{\rm op} &\le \|A_{\rm re} - P\|_{\rm op} + \|P - \widetilde{P}\|_{\rm op} \\ &\le C\sqrt{d} + \frac{1}{\sqrt{n_1 n_2}}\|P^0 - \widetilde{P}^0\|_{\rm op} \\ &\le C\sqrt{d} + \frac{1}{\sqrt{n_1 n_2}}\|P^0 - \widetilde{P}^0\|_F \le (C_0 + C)\sqrt{d}\end{aligned}$$

where we have used Theorem 2 to bound the first term using $d' = d = \sqrt{n_2/n_1}\|P^0\|_\infty = n_2\|P\|_\infty$. The last inequality uses assumption (52). The proof is complete. ■

Proof [Proof of Proposition 3]
Letting $f(x,y) := [\rho_0(x,y) - \widetilde{\rho}_0(x,y)]^2$,

$$\|P^0 - \widetilde{P}^0\|_F^2 = \sum_{i=1}^n \sum_{j=1}^n f(X_i, Y_j) = \sum_{j \in \mathbb{Z}_n} \sum_{i \in \mathbb{Z}_n} f(X_i, Y_{i+j}),$$

where $\mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z}$ is $\{1, \dots, n\}$ viewed as a cyclic group of order n. We have

$$\mathbb{E}[f(X_1, Y_1)^2] = \|f\|_{L^2}^2 = \|\rho_0 - \widetilde{\rho}_0\|_{L^4}^4 = o(d^2)$$

and $\mathbb{E}[f(X_1, Y_1)] = \|f\|_{L^1} \le \|f\|_{L^2} = o(d)$. Note that $\{f(X_i, Y_j) : i, j \in \mathbb{Z}_n\}$ are not independent, however, for each $j \in \mathbb{Z}_n$, $\{f(X_i, Y_{i+j}) : i \in \mathbb{Z}_n\}$ are i.i.d.. Let $Z_j := \sum_{i \in \mathbb{Z}_n} f(X_i, Y_{i+j})$. By independence,

$$\operatorname{var}(Z_j) = n \operatorname{var}(f(X_1, Y_1)) \le n\|f\|_{L^2}^2.$$

Since $\mathbb{E}Z_j = n\|f\|_{L^1}$, by the Chebyshev inequality, for any $j \in \mathbb{Z}_n$,

$$\begin{aligned}\mathbb{P}(Z_j \ge nd) &= \mathbb{P}\big(Z_j - \mathbb{E}Z_j \ge nd - n\|f\|_{L^1}\big) \\ &\le \frac{n\|f\|_{L^2}^2}{(na - n\|f\|_{L^1})^2} = \frac{1}{n}\frac{o(d^2)}{(d - o(d))^2} = o\Big(\frac{1}{n}\Big).\end{aligned}$$

It follows that $\mathbb{P}\big(\|P^0 - \widetilde{P}^0\|_F^2 \ge n^2 d\big) \le \sum_{j \in \mathbb{Z}_n} \mathbb{P}\big(Z_j \ge nd\big) \le n\, o\big(n^{-1}\big) = o(1)$. ■

Appendix B. Alternative algorithm for the k-means step

In this appendix, we present a simple general algorithm that can be used in the k-means step, replacing the κ-approximate k-means solver used throughout the text. The algorithm is based on the ideas in Gao et al. (2017) and Yun and Proutiere (2014b), and the version that we present here acheives the misclassification bound $\varepsilon^2/(n\delta^2)$ needed in Step 3 of the analysis (Section 3.1) without necessarily optimizing the k-means objective function. We present the results using the terminology of the k-means matrices (with rows in $\mathbb{R}^d$) introduced in Section 3.3, although the algorithm and the resulting bound work for data points in any metric space.

Let $X \in \mathbb{M}^k_{n,m}$ be a k-means matrix and let us denote its centers, i.e. distinct rows, as $\{q_r(X), r \in [k]\}$. As in Definition 2, we write $\delta_r(X)$ and $n_r(X)$ for the rth cluster center separation and size, respectively, and $\delta_\wedge(X) = \min_r \delta_r(X)$ and $n_\wedge = \min_r n_r(X)$. Assume that we have an estimate $\hat{X} \in \mathbb{R}^{n \times d}$ of X, and let us write $d(i,j) := d(\hat{x}_i, \hat{x}_j)$, $i, j \in [n]$ for the pairwise distances between the rows of $\hat{X}$.

Algorithm 6 which is a variant of the one presented in Gao et al. (2017), takes these pairwise distances and outputs cluster estimates $\widehat{\mathcal{C}}_1, \dots, \widehat{\mathcal{C}}_k \subset [n]$, after k recursive passes through the data. A somewhat more sophisticated version of this algorithm appears in Yun and Proutiere (2014b), where one also repeats the process for $i = 1, \dots, \log n$ and radii $R_i = iR_1$ in an outer loop, producing clusters $\widehat{\mathcal{C}}_r^{(i)}$, $r \in [k]$; one then picks, among these $\log n$ possible clusterings, the one that minimizes the k-means objective. The variant in Yun and Proutiere (2014b) also leaves no unlabeled nodes by assigning the unlabeled to the cluster whose estimated center is closest. In the rest of this section, we will focus on the simple version presented in Algorithm 6 as this is enough to establish our desired bound. The following theorem provides the necessary guarantee:

Theorem 9 *Consider the cluster model above and let $n_r = n_r(X)$, $n_\wedge = n_\wedge(X)$ and $\delta_\wedge = \delta_\wedge(X)$. Assume that we have approximate data $\hat{x}_1, \dots, \hat{x}_n$ such that $\sum_{i=1}^n d(x_i, \hat{x}_i)^2 \le \varepsilon^2$. In addition, assume that for some $\gamma \in (0,1)$ and $\beta \ge 1$:*

(i) $n_r \le \beta n_\wedge$ *for all* $r \in [k]$ *(Clusters are β-balanced.),*

(ii) $\dfrac{2\varepsilon}{\sqrt{\gamma n_\wedge}} < \dfrac{\delta_\wedge}{3}$ *(ε^2 small enough compared to $n_\wedge \delta_\wedge^2$.),*

(iii) $\xi\beta + \gamma < 1 - \gamma$ *where $\xi := \gamma/(1-\gamma)$. (Gamma small enough relative to β.)*

Algorithm 6 k-means replacement

Input: Pairwise distance $d(i,j)$, $i,j \in [n]$ and radius ρ.

$S \leftarrow [n]$
for $r = 1, \dots, k$ **do**
  For every $i \in S$, let $B_d(i;\rho) := \{j \in S : \ d(i,j) \le \rho\}$.
  Pick $i_0 \in S$ that maximizes $i \mapsto |B_d(i;\rho)|$.
  Let $\widehat{\mathcal{C}}_r = B_d(i_0;\rho)$.
  $S \leftarrow S \setminus \widehat{\mathcal{C}}_r$.
end for

Output: Return clusters $\widehat{\mathcal{C}}_r : 1, \dots, k$ and output remaining S as unlabeled.

Let $M_n(\rho)$ be the (average) misclassification rate of Algorithm 6 with input radius ρ. Then,

$$M_n(\rho) \le \frac{8\varepsilon^2}{n\rho^2}, \quad \forall \rho \in \Big[\frac{2\varepsilon}{\sqrt{\gamma n_\wedge}}, \frac{\delta_\wedge}{3}\Big).$$

Applying the algorithm with $\rho \ge \alpha\delta_\wedge$ for $\alpha < 1/3$ we obtain the misclassication bound $c_\alpha\, \varepsilon^2/(n\delta_\wedge^2)$ where $c_\alpha = 8/\alpha^2$. Thus, Algorithm 6 with a proper choice of the radius ρ satisfies the desired bound (10) of the k-means step.

Proof [Proof of Theorem 9] The proof follows the argument in Gao et al. (2017). As in the proof of Proposition 1, let $\mathcal{C}_r$ denote the rth cluster of X, having center $q_r = q_r(X)$. We have $|\mathcal{C}_r| = n_r$. Let x_i and $\hat{x}_i$ be the ith row of X and $\hat{X}$, respectively, and let

$$T_r \coloneqq \{i \in \mathcal{C}_r : \ d(\hat{x}_i, q_r) < \rho/2\} = \{i \in \mathcal{C}_r : \ d(\hat{x}_i, x_i) < \rho/2\}$$

using $x_i = q_r$ for all $i \in \mathcal{C}_r$ which holds by definition. $\{T_r\}$ are disjoint and clearly $T_r \subset \mathcal{C}_r$. Let $T := \biguplus_r T_r$, a disjoint union, and $T^c = [n] \setminus T$. We have

$$|T^c|\rho^2/4 \ \le \ \sum_{i \in T^c} d(\hat{x}_i, x_i)^2 \le \varepsilon^2 \implies |T^c| \le 4\varepsilon^2/\rho^2. \tag{68}$$

As a consequence of assumption (ii) and our choice of ρ, we have $4\varepsilon^2/(n_\wedge\rho^2) \le \gamma$, hence

$$|T_r| = |\mathcal{C}_r| - |\mathcal{C}_r \setminus T_r| \ \ge \ |\mathcal{C}_r| - |T^c| \ \ge \ n_\wedge\Big(1 - \frac{4\varepsilon^2}{n_\wedge\rho^2}\Big) \ge n_\wedge(1-\gamma) \tag{69}$$

for all $r \in [k]$. On the other hand, $|T^c| \le \gamma n_\wedge$. In particular, combining the two estimates

$$|T^c| \le \xi\, |T_r|, \quad \forall r \in [k] \tag{70}$$

where $\xi = \gamma/(1-\gamma)$. These size estimates will be used frequently in the course of the proof.

Recall that $d(i,j) := d(\hat{x}_i, \hat{x}_j)$, $i,j \in [n]$, the collection of pairwise distances between the data points $\hat{x}_1, \dots, \hat{x}_n$. Thus, with some abuse of notation, $(i,j) \mapsto d(i,j)$ defines a pseudo-metric on $[n]$ (and a proper metric if $\{\hat{x}_i\}$ are distinct). For any two subsets $A, B \subset [n]$ we write $d(A,B) = \inf\{d(i,j) : i \in A,\ j \in B\}$. For any $i \in [n]$, let $d(i,A) = d(\{i\}, A)$.

We say that node i_0 is near T if $d(i_0, T) \le \rho$, i.e., i_0 belongs to the ρ-enlargement of T. Similarly, we say that i_0 is near T_r if $d(i_0, T_r) \le \rho$ and far from T_r otherwise. Note that i_0 can be near at most one of $T_r, r \in [k]$. This is since $d(T_r, T_\ell) \ge \delta_\wedge - \rho$ for $r \ne \ell$, and we are assuming $\delta_\wedge > 3\rho$. In fact, *i_0 is near T iff i_0 is near exactly one of $T_r, r \in [k]$.*

To understand Algorithm 6, let us assume that we are at some iteration of the algorithm and we are picking the center i_0 and the corresponding cluster $\widehat{\mathcal{C}} := \{j : \ d(j, i_0) \le \rho\}$. One of the following happens:

(a) We pick the new center $i_0 \in T_r$ for some r, in which case $\widehat{\mathcal{C}}$ will include the entire T_r, none of $T_\ell, \ell \ne r$, and perhaps some of T^c. That is, $\widehat{\mathcal{C}} \supset T_r$ and $\widehat{\mathcal{C}} \cap T_\ell = \emptyset$ for $\ell \ne r$.

(b) We pick i_0 *near* T_r for some r. In this case, $|\widehat{\mathcal{C}}| \ge |T_r|$, otherwise any member of T_r would have created a bigger cluster by part (a) above. Now, $\widehat{\mathcal{C}}$ cannot contain any of $T_\ell, \ell \ne r$, because i_0 is far from those if it is near T_r. Hence, $\widehat{\mathcal{C}} \subset T_r \cup T^c$. Since $|T^c| \le \xi|T_r|$ by (70), and $|\widehat{\mathcal{C}}| \ge |T_r|$, we have $|\widehat{\mathcal{C}} \cap T_r| \ge (1-\xi)|T_r|$. That is, $\widehat{\mathcal{C}}$ contains a large fraction of T_r.

If either of the two cases above happen, we say that T_r is depleted, otherwise it is intact. If T_r is depleted, it will not be revisited in future iterations, as long as other intact $T_\ell, \ell \ne r$ exist. To see this, first note that $|T_r \cap \widehat{\mathcal{C}}^c| \le \xi|T_r| \le \xi\beta n_\wedge$, using assumption (i). Taking i_0 on or near T_r in a future iteration will give us a cluster of size at most $(\xi\beta+\gamma)n_\wedge < (1-\gamma)n_\wedge$ (by assumption (iii)) which is less that $|T_\ell|$ for an intact cluster.

To simplify notation, if either of (a) and (b) happen, i.e., we pick cluster center i_0 near T_r for some r, we name the corresponding cluster $\widehat{\mathcal{C}}_r$. This is to avoid carrying around a permutation of cluster labels different than the original one, and is valid since each T_r is visited at most once by the above argument. (In fact, each is visited exactly once, as we argue below.) That last possibility is

(c) We pick i_0 *far* from any T_r, that is $d(i_0, T) > \rho$. This gives $\widehat{\mathcal{C}} \subset T^c$, hence $|\widehat{\mathcal{C}}| \le |T^c| \le \gamma n_\wedge < (1-\gamma)n_\wedge \le |T_\ell|$ for any intact T_ℓ. Thus as along as there are intact T_ℓ, this case does not happen.

The above argument gives the following picture of the evolution of the algorithm: At each step $t = 1, \dots, k$, we pick i_0 near T_r for some previously unvisited r, making it depleted, creating estimated cluster $\widehat{\mathcal{C}}_r$ and proceeding to the next iteration. After the k-th iteration all $T_\ell, \ell \in [k]$ will be depleted. We have $|\widehat{\mathcal{C}}_r| \ge |T_r|$, and $\widehat{\mathcal{C}}_r \subset T_r \cap T^c$ for all $r \in [k]$.

By construction $\{\widehat{\mathcal{C}}_\ell\}$ are disjoint. Let $\widehat{\mathcal{C}} := \biguplus_{\ell \in [k]} \widehat{\mathcal{C}}_\ell$, and note that $|\widehat{\mathcal{C}}| \ge |T|$ hence $|\widehat{\mathcal{C}}^c| \le |T^c|$. Since $\widehat{\mathcal{C}}_\ell \cap T_r = \emptyset$ for $\ell \ne r$, we have $T_r \subset \bigcap_{\ell \ne r} \widehat{\mathcal{C}}_\ell^c$, hence $T_r \cap \widehat{\mathcal{C}}_r^c \subset \widehat{\mathcal{C}}^c$. All the misclassified or unclassified nodes produced by the algorithm are contained in $[\bigcup_r (T_r \cap \widehat{\mathcal{C}}_r^c)] \cup T^c$ which itself is contained in $\widehat{\mathcal{C}}^c \cup T^c$. Hence, the misclassification rate is bounded above by

$$\frac{1}{n}|\widehat{\mathcal{C}}^c \cup T^c| \le \frac{1}{n}(|\widehat{\mathcal{C}}^c| + |T^c|) \le \frac{2}{n}|T^c| \le \frac{8\varepsilon^2}{n\rho^2}.$$

where we have used (68). The proof is complete. ∎

Remark 13 *The last part of the argument can be made more transparent as follows: Each $\widehat{\mathcal{C}}_r$ consists of two disjoint part, $\widehat{\mathcal{C}}_r \cap T_r$ and $\widehat{\mathcal{C}}_r \cap T^c$. We have $|\widehat{\mathcal{C}}_r \cap T^c| \geq |T_r \setminus \widehat{\mathcal{C}}_r|$ (equivalent to $|\widehat{\mathcal{C}}_r| \geq |T_r|$). Then,*

$$\sum_k |T_r \setminus \widehat{\mathcal{C}}_r| \leq \sum_r |\widehat{\mathcal{C}}_r \cap T^c| = |\widehat{\mathcal{C}} \cap T^c| \leq |T^c|$$

and total misclassifications are bounded by $\sum_k |T_r \setminus \widehat{\mathcal{C}}_r| + |T^c|$.

Appendix C. Auxiliary lemmas

Lemma 8 *Let $Z, Y \in \mathbb{O}^{n\times k}$ and let $\Pi_Z = ZZ^T$ and $\Pi_Y = YY^T$ be the corresponding projection operators. We have*

$$\min_{Q\in\mathbb{O}^{k\times k}} \|Z - YQ\|_F \;\leq\; \|\Pi_Z - \Pi_Y\|_F.$$

Proof We first note that $\|\Pi_Z\|_F^2 = \mathrm{tr}(\Pi_Z^2) = \mathrm{tr}(\Pi_Z) = k$ (since projections are idempotent), and $\|Z\|_F^2 = \mathrm{tr}(Z^TZ) = \mathrm{tr}(\Pi_Z) = k$. Let $Z^TY = U\Sigma V^T$ be the SVD of Z^TY where $U, V \in \mathbb{O}^{k\times k}$ and $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_k) \succeq 0$. Then, using the change of variable $O = V^TQU$,

$$\begin{aligned}\frac{1}{2}\min_Q \|Z - YQ\|_F^2 &= k - \max_Q \mathrm{tr}(Z^TYQ)\\ &= k - \max_{O\,\in\,\mathbb{O}^{k\times k}} \mathrm{tr}(\Sigma O) = k - \|\Sigma\|_*,\end{aligned}$$

where $\|\Sigma\|_* = \sum_i \sigma_i$ is the nuclear norm of Σ. To see the last equality, we note that since O is orthogonal, we have $|O_{ii}| \leq 1$ for all i, hence $\max_O \mathrm{tr}(\Sigma O) \leq \max_{\forall i,\,|O_{ii}|\leq 1} \sum_i \sigma_i O_{ii} = \sum_i \sigma_i$ by the duality of ℓ_1 and ℓ_∞ norms and $\sigma_i \geq 0$. The equality is achieved by $O = I_k$. On the other hand

$$\frac{1}{2}\|\Pi_Z - \Pi_Y\|_F^2 = k - \mathrm{tr}(\Pi_Z\Pi_Y) = k - \|Z^TY\|_F^2 = k - \|\Sigma\|_F^2.$$

Since $\|\Sigma\|_{\mathrm{op}} = \|Z^TY\|_{\mathrm{op}} = \|Z\|_{\mathrm{op}}\|Y\|_{\mathrm{op}} \leq 1$, we have $\sigma_i \leq 1$ for all i. It follows that $\|\Sigma\|_F^2 \leq \|\Sigma\|_*$ completing the proof. ■

References

Emmanuel Abbe. Community detection and stochastic block models: recent developments. *arXiv preprint arXiv:1703.10146*, 2017.

Emmanuel Abbe, Jianqing Fan, Kaizheng Wang, and Yiqiao Zhong. Entrywise eigenvector analysis of random matrices with low expected rank. *arXiv preprint arXiv:1709.09565*, 2017.

Edo M Airoldi, Thiago B Costa, and Stanley H Chan. Stochastic blockmodel approximation of a graphon: Theory and consistent estimation. In *Advances in Neural Information Processing Systems*, pages 692–700, 2013.

Arash A Amini, Aiyou Chen, Peter J Bickel, Elizaveta Levina, et al. Pseudo-likelihood methods for community detection in large sparse networks. *The Annals of Statistics*, 41 (4):2097–2122, 2013.

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

Sivaraman Balakrishnan, Min Xu, Akshay Krishnamurthy, and Aarti Singh. Noise thresholds for spectral clustering. In *Advances in Neural Information Processing Systems*, pages 954–962, 2011.

Afonso S Bandeira, Ramon Van Handel, et al. Sharp nonasymptotic bounds on the norm of random matrices with independent entries. *The Annals of Probability*, 44(4):2479–2506, 2016.

Rajendra Bhatia. *Matrix analysis*, volume 169. Springer Science & Business Media, 2013.

Peter J Bickel and Aiyou Chen. A nonparametric view of network models and newman–girvan and other modularities. *Proceedings of the National Academy of Sciences*, pages pnas–0907096106, 2009.

Norbert Binkiewicz, Joshua T. Vogelstein, and Karl Rohe. Covariate-assisted spectral clustering. pages 1–48, nov 2014. URL `http://arxiv.org/abs/1411.2158`.

Béla Bollobás, Svante Janson, and Oliver Riordan. The phase transition in inhomogeneous random graphs. *Random Structures & Algorithms*, 31(1):3–122, 2007.

Ravi B Boppana. Eigenvalues and graph bisection: An average-case analysis. In *Foundations of Computer Science, 1987., 28th Annual Symposium on*, pages 280–285. IEEE, 1987.

Joshua Cape, Minh Tang, and Carey E Priebe. On spectral embedding performance and elucidating network structure in stochastic block model graphs. *arXiv preprint arXiv:1808.04855*, 2018.

Kamalika Chaudhuri, Fan Chung, and Alexander Tsiatas. Spectral clustering of graphs with general degrees in the extended planted partition model. In *Conference on Learning Theory*, pages 35–1, 2012.

Yudong Chen and Jiaming Xu. Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices. *The Journal of Machine Learning Research*, 17(1):882–938, 2016.

Peter Chin, Anup Rao, and Van Vu. Stochastic block model and community detection in sparse graphs: A spectral algorithm with optimal rate of recovery. In *Conference on Learning Theory*, pages 391–423, 2015.

Amin Coja-Oghlan. Graph partitioning via adaptive spectral techniques. *Combinatorics, Probability and Computing*, 19(2):227–284, 2010.

Anirban Dasgupta, John Hopcroft, Ravi Kannan, and Pradipta Mitra. Spectral clustering by recursive partitioning. *Algorithms–ESA 2006*, pages 256–267, 2006.

Inderjit S Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–274. ACM, 2001.

Donniell E Fishkind, Daniel L Sussman, Minh Tang, Joshua T Vogelstein, and Carey E Priebe. Consistent adjacency-spectral partitioning for the stochastic block model when the model parameters are unknown. *SIAM Journal on Matrix Analysis and Applications*, 34(1):23–39, 2013.

Chao Gao, Yu Lu, Harrison H Zhou, et al. Rate-optimal graphon estimation. *The Annals of Statistics*, 43(6):2624–2652, 2015.

Chao Gao, Zongming Ma, Anderson Y Zhang, and Harrison H Zhou. Achieving optimal misclassification proportion in stochastic block models. *The Journal of Machine Learning Research*, 18(1):1980–2024, 2017.

Chao Gao, Zongming Ma, Anderson Y Zhang, Harrison H Zhou, et al. Community detection in degree-corrected block models. *The Annals of Statistics*, 46(5):2153–2185, 2018.

John A Hartigan. Direct clustering of a data matrix. *Journal of the american statistical association*, 67(337):123–129, 1972.

Jiashun Jin. Fast community detection by score. *The Annals of Statistics*, 43(1):57–89, 2015.

A Joseph and B Yu. Impact of regularization on Spectral Clustering. *arXiv preprint arXiv:1312.1733*, 2013. URL `http://arxiv.org/abs/1312.1733`.

B. Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Phys. Rev. E*, 83(1):016107, 2011.

Olga Klopp, Alexandre B Tsybakov, Nicolas Verzelen, et al. Oracle inequalities for network models and sparse graphon estimation. *The Annals of Statistics*, 45(1):316–354, 2017.

F. Krzakala, C. Moore, E. Mossel, J. Neeman, A. Sly, Lenka Zdeborová, and Pan Zhang. Spectral redemption in clustering sparse networks. *Proceedings of the National Academy of Sciences of the United States of America*, 110(52):20935–40, dec 2013. ISSN 1091-6490. doi: 10.1073/pnas.1312486110. URL `http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3876200{&}tool=pmcentrez{&}rendertype=abstract`.

Amit Kumar, Yogish Sabharwal, and Sandeep Sen. A simple linear time (1+/spl epsiv/)-approximation algorithm for k-means clustering in any dimensions. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 454–462. IEEE, 2004.

C. M. Le, E. Levina, and R. Vershynin. Sparse random graphs: regularization and concentration of the Laplacian. *arXiv preprint arXiv:1502.03049*, feb 2015. doi: 10.1088/0264-9381/32/11/115009. URL `http://arxiv.org/abs/1502.0304http://arxiv.org/abs/1502.00304http://arxiv.org/abs/1502.03049`.

Can M Le, Elizaveta Levina, and Roman Vershynin. Concentration and regularization of random graphs. *Random Structures & Algorithms*, 2017.

Jing Lei, Alessandro Rinaldo, et al. Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43(1):215–237, 2015.

Yu Lu and Harrison H Zhou. Statistical and computational guarantees of lloyd's algorithm and its variants. *arXiv preprint arXiv:1612.02099*, 2016.

Vince Lyzinski, Daniel L Sussman, Minh Tang, Avanti Athreya, Carey E Priebe, et al. Perfect clustering for stochastic blockmodel graphs via adjacency spectral embedding. *Electronic Journal of Statistics*, 8(2):2905–2922, 2014.

F. McSherry. Spectral partitioning of random graphs. In *Proceedings 2001 IEEE International Conference on Cluster Computing*, pages 529—-537. IEEE Comput. Soc, 2001. ISBN 0-7695-1116-3. doi: 10.1109/SFCS.2001.959929. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=959929`.

Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.

Sofia C Olhede and Patrick J Wolfe. Network histograms and universality of blockmodel approximation. *Proceedings of the National Academy of Sciences*, 111(41):14722–14727, 2014.

Tai Qin and Karl Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. In *Advances in Neural Information Processing Systems*, pages 3120–3128, 2013.

Karl Rohe and Bin Yu. Co-clustering for directed graphs; the stochastic co-blockmodel and a spectral algorithm. *stat*, 1050:10, 2012.

Karl Rohe, Sourav Chatterjee, Bin Yu, et al. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011.

Bo Söderberg. General formalism for inhomogeneous random graphs. *Physical review E*, 66 (6):066121, 2002.

Liangjun Su, Wuyi Wang, and Yichong Zhang. Strong consistency of spectral clustering for stochastic block models. *arXiv preprint arXiv:1710.06191*, 2017.

D.-C. Tomozei and L. Massoulié. Distributed user profiling via spectral methods. In *ACM SIGMETRICS Performance Evaluation Review*, pages 383—-384, 2010. URL `http://www.i-journals.org/ssy/viewarticle.php?id=36http://arxiv.org/abs/1109.3318http://dl.acm.org/citation.cfm?id=1811098`.

Joel A Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, 8(1-2):1–230, 2015.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge University Press, 2018.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4): 395–416, 2007.

Ulrike Von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586, 2008.

Jiaming Xu. Rates of convergence of spectral methods for graphon estimation. *arXiv preprint arXiv:1709.03183*, 2017.

Se-Young Yun and Alexandre Proutiere. Accurate community detection in the stochastic block model via spectral algorithms. *arXiv preprint arXiv:1412.7335*, 2014a.

Se-Young Yun and Alexandre Proutiere. Community detection via random and adaptive sampling. In *Conference on Learning Theory*, pages 138–175, 2014b.

Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. Consistency of community detection in networks under degree-corrected stochastic block models. *The Annals of Statistics*, 40 (4):2266–2292, 2012.

Zhixin Zhou and Arash A. Amini. Optimal bipartite network clustering. *Preprint*, 2018.

Zhixin Zhou and Arash A. Amini *GitHub repository: Bipartite Spectral Clustering*. URL `https://github.com/zhixin0825/bipartite-spectral-clustering`, 2018.

Journal of Machine Learning Research 20 (2019) 1-23 Submitted 4/18; Revised 3/19; Published 3/19

Efficient augmentation and relaxation learning for individualized treatment rules using observational data

Ying-Qi Zhao YQZHAO@FREDHUTCH.ORG
Public Health Sciences Division
Fred Hutchinson Cancer Research Center
Seattle, WA, 98109, USA

Eric B. Laber EBLABER@NCSU.EDU
Department of Statistics
North Carolina State University
Raleigh, NC, 27695, USA

Yang Ning YN265@CORNELL.EDU
Department of Statistical Science
Cornell University
Ithaca, NY, 14853, USA

Sumona Saha SSAHA@MEDICINE.WISC.EDU
School of Medicine and Public Health
University of Wisconsin
Madison, WI, 53705, USA

Bruce E. Sands BRUCE.SANDS@MSSM.EDU
Division of Gastroenterology
Icahn School of Medicine at Mount Sinai
New York, NY, 10029, USA

Editor: XuanLong Nguyen

Abstract

Individualized treatment rules aim to identify if, when, which, and to whom treatment should be applied. A globally aging population, rising healthcare costs, and increased access to patient-level data have created an urgent need for high-quality estimators of individualized treatment rules that can be applied to observational data. A recent and promising line of research for estimating individualized treatment rules recasts the problem of estimating an optimal treatment rule as a weighted classification problem. We consider a class of estimators for optimal treatment rules that are analogous to convex large-margin classifiers. The proposed class applies to observational data and is doubly-robust in the sense that correct specification of either a propensity or outcome model leads to consistent estimation of the optimal individualized treatment rule. Using techniques from semiparametric efficiency theory, we derive rates of convergence for the proposed estimators and use these rates to characterize the bias-variance trade-off for estimating individualized treatment rules with classification-based methods. Simulation experiments informed by these results demonstrate that it is possible to construct new estimators within the proposed framework that significantly outperform existing ones. We illustrate the proposed methods using data from a labor training program and a study of inflammatory bowel syndrome.

©2019 Zhao, Laber, Ning, Saha, Sands.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-191.html.

Keywords: Individualized treatment rules, convex surrogate, double-robustness, classification, personalized medicine

1. Introduction

There is a growing consensus that the best possible care results from treatment decisions that are carefully tailored to individual patient characteristics (Sox and Greenfield, 2009). Individualized treatment rules (ITRs) formalize tailored treatment decisions as a function from patient information to a recommended treatment. We define an optimal ITR as maximizing the mean of a pre-specified clinical outcome if applied to recommend treatments in a population of interest (see Linn et al., 2016, for alternative definitions of optimality). With expanding access to patient-level data through electronic health records, adverse event reporting, insurance claims, and billing records, there is increasing interest in estimating optimal ITRs from observational data. An important use of an estimated optimal ITR is hypothesis-generation whereby the estimated optimal rule is used to discover covariate-treatment interactions or identify subgroups of patients with large treatment effects. In such applications, it is useful to directly control the class of ITRs within which the optimal ITR will be estimated. The form of this class can be chosen to ensure interpretability, enforce logistical or cost constraints, or make the tests of certain clinical hypotheses overt.

One approach to estimating an optimal ITR is to model some or all of the conditional distribution of the outcome given treatments and covariates and then to use this estimated distribution to infer the optimal ITR. These approaches are sometimes called indirect methods as they indirectly specify the form of the optimal ITR through postulated models for components of the conditional outcome distribution. Indirect methods have dominated the literature on estimating optimal ITRs; examples of indirect estimation methods include variations of g-estimation in structural nested models (Robins, 1989, 1997; Murphy, 2003; Robins, 2004); Q- and A-learning (Zhao et al., 2009; Qian and Murphy, 2011; Moodie et al., 2012; Chakraborty and Moodie, 2013; Schulte et al., 2014), and regret regression (Henderson et al., 2009). However, a major drawback with these approaches is that the postulated outcome models dictates the class of possible ITRs. A consequence is that to obtain a simple ITR requires specification of simple outcome models, which may not be correctly specified. Moreover, if these outcome models are misspecified, the foregoing methods may not be consistent for the optimal ITR within the class implied by the outcome models. For example, to ensure a linear ITR using Q-learning, it is common to use a linear conditional mean model. It can be shown that if the linear mean model is misspecified then the estimated optimal ITR using Q-learning need not converge to the optimal linear ITR (Qian and Murphy, 2011). Alternatively, flexible outcome models that mitigate the risk of misspecification (e.g., Zhao et al., 2009; Qian and Murphy, 2011; Moodie et al., 2013) can induce a class of ITRs that is difficult or impossible to interpret (see Section 2 for details).

An alternative to indirect estimation is to decouple models for the conditional outcome distribution from the class of ITRs. One way to do this is to form a flexible estimator of the mean outcome as a function of the ITR that is consistent under a large class of potential generative models and then to use the maximizer of this function over a pre-specified class of ITRs as the estimator of the optimal ITR. These approaches are called direct (Laber et al., 2014), policy-search (Sutton and Barto, 1998; Szepesvári, 2010), policy

learning (Athey and Wager, 2017) or value-search (Davidian et al., 2014) estimators. An advantage of direct estimators is that they permit flexible, e.g., semi- or non-parametric, models for modeled portions of the outcome distribution yet still control the form of the estimated optimal ITR. Direct estimators include outcome weighted learning (Zhao et al., 2012, 2015a, 2015b), robust value-search estimators (Zhang et al., 2012a, 2012b, 2013); marginal structural mean models (Robins et al., 2008; Orellana et al., 2010); and Q-learning with policy-search (Taylor et al., 2015; Zhang et al., 2015, 2017).

While the foregoing methods represent significant progress in direct estimation, computational and theoretical gaps remain. Outcome weighted learning uses a convex relaxation of an inverse-probability weighted estimator (IPWE) of the mean outcome. This convex relaxation makes their method computationally efficient and scalable to large problems; in addition, convexity simplifies derivations of convergence rates and generalization error bounds. However, the IPWE is known to be unstable under certain generative models (Zhang et al., 2012a, 2012b), and theoretical guarantees for outcome weighted learning were developed only for data from a randomized clinical trial. Robust value-search estimators directly maximize an augmented IPWE (AIPWE). The AIPWE is semi-parametric efficient and is significantly more stable than the IPWE. However, the AIPWE is a discontinuous function of the observed data, which makes direct maximization computationally burdensome even in moderate sized problems and complicates theoretical study of these estimators. We establish the theory for both AIPW and its convex relaxation, which fills the gap in the current literature on direct search methods. Marginal structural mean models are best suited for problems where the ITR depends only on a very small number of covariates. Liu et al. (2016) proposed a robust method for estimating optimal treatment rules in a multi-stage setup. At each stage in a multi-stage setup, they proposed a robust weight to replace the original weight in OWL based on the idea of augmentation. However, they still require consistent estimation of the propensity score at the present stage. In particular, their proposal for the single stage problem still relies on an IPWE, and does not possess the double robustness property.

We propose a class of estimators representable as the maximizer of a convex relaxation of the AIPWE; we term this class of estimators Efficient Augmentation and Relaxation Learning (EARL). EARL is computationally efficient, theoretically tractable, and applies to both observational and experimental data. Furthermore, EARL contains outcome weighted learning (OWL) (Zhao et al., 2012) as a special case. However, EARL is considerably more general than OWL, and this generality leads to new insights about classification-based estimation of ITRs, new algorithms, and new theoretical results. Unlike OWL, EARL makes use of both a propensity score and an outcome regression model. Estimators within the EARL framework are doubly-robust in the sense that they consistently estimate the optimal ITR if either the propensity score model or outcome regression model is correctly specified. Within the EARL framework, we are able to characterize convergence rates across a range of convex relaxations, propensity score models, and outcome regression models. In particular, making use of sample splitting, we are able to remove the dependence in estimating the nuisance functions and in constructing the estimated ITR. We show that under all convex relaxations considered, a fast convergence rate of the estimated optimal ITR can be achieved, and that the estimation of the propensity score and outcome regression models need not affect the upper bound of this rate. Our theoretical results complement existing work on

convergence rate for estimating optimal treatment decision rules, which primarily compared the estimated rules to the best-in-class rule (Kitagawa and Tetenov, 2017; Athey and Wager, 2017; **?**). The proposed method has been implemented in R and is freely available through the 'DynTxRegime' package hosted on the comprehensive R network (cran.org).

In Section 2, we introduce the EARL class of estimators. In Section 3, we investigate the theoretical properties of estimators within this class. In Section 4, we use simulation experiments to investigate the finite sample performance of EARL estimators. In Section 5, we present illustrative case studies using data from a labor training program and an inflammatory bowel disease study. In Section 6, we make concluding remarks and discuss potential extensions.

2. Methods

In this section, we first provide background of the proposed method. We then introduce Efficient Augmentation and Relaxation Learning (EARL) in details.

2.1. Background and preliminaries

The observed data, $\{(\boldsymbol{X}_i, A_i, Y_i)\}_{i=1}^n$, comprise n independent, identically distributed copies of $(\boldsymbol{X}, A, Y)$, where: $\boldsymbol{X} \in \mathbb{R}^p$ denotes baseline subject measurements; $A \in \{-1, 1\}$ denotes the assigned treatment; and $Y \in \mathbb{R}$ denotes the outcome, coded so that higher values are better. In this context, an ITR, d, is a map from $\mathbb{R}^p$ into $\{-1, 1\}$ so that a patient presenting with $\boldsymbol{X} = \boldsymbol{x}$ is recommended treatment $d(\boldsymbol{x})$. Let $\mathcal{D}$ denote a class of ITRs of interest. To define the optimal ITR, denoted d^*, we use the framework of potential outcomes (Rubin, 1974; Splawa-Neyman et al., 1990). Let $Y(a)$ denote the potential outcome under treatment $a \in \{-1, 1\}$ and define $Y(d) = \sum_{a\in\{-1,1\}} Y(a) I\{a = d(\boldsymbol{X})\}$ to be the potential outcome under d. The marginal mean outcome $V(d) \triangleq E\{Y(d)\}$ is called the value of the ITR d. The optimal ITR satisfies $d^* \in \mathcal{D}$ and $V(d^*) \geq V(d)$ for all $d \in \mathcal{D}$. Note that this definition of optimality depends on the class $\mathcal{D}$. To express the value in terms of the data generating model, we assume: (i) strong ignorability, $\{Y(-1), Y(1)\} \amalg A \big| \boldsymbol{X}$ (Rubin, 1974; Robins, 1986; Splawa-Neyman et al., 1990); (ii) consistency, $Y = Y(A)$; and (iii) positivity, there exists $\tau > 0$ so that $\tau < P(A = a|\boldsymbol{X})$ for each $a \in \{-1, 1\}$ with probability one. These assumptions are common and well-studied (see Schulte et al., 2014, for a recent review of potential outcomes for treatment rules). Assumption (i) is true in a randomized study but unverifiable in an observational study (Bang and Robins, 2005).

Define $Q(\boldsymbol{x}, a) \triangleq E(Y|\boldsymbol{X} = \boldsymbol{x}, A = a)$, then under the foregoing assumptions, it can be shown that

$$V(d) = E\left[Q\left\{\boldsymbol{X}, d(\boldsymbol{X})\right\}\right], \tag{1}$$

from which it follows that $d^*(\boldsymbol{x}) = \arg\max_{a\in\{-1,1\}} Q(\boldsymbol{x}, a)$. Q-learning is a common regression-based indirect approach for estimating d^* wherein an estimator $\widehat{Q}(\boldsymbol{x}, a)$ of $Q(\boldsymbol{x}, a)$ is constructed and subsequently the estimated optimal rule is $\widehat{d}(\boldsymbol{x}) = \arg\max_a \widehat{Q}(\boldsymbol{x}, a)$. Let $\mathcal{Q}$ denote the postulated class of models for $Q(\boldsymbol{x}, a)$, then the set of possible decision rules obtained using Q-learning is $\mathcal{D} = \{d : d(\boldsymbol{x}) = \arg\max_a Q(\boldsymbol{x}, a),\ Q \in \mathcal{Q}\}$. Thus, there is an inherent trade-off between choosing $\mathcal{Q}$ to be sufficiently rich to reduce the risk of model misspecification and the resultant complexity of the resultant class of ITRs.

Direct estimators specify a class of candidate ITRs independently from postulated models for some or all of the generative model. Let $\mathcal{D}$ denote a class of ITRS; direct search estimators first construct an estimator of the value function, say $\widehat{V}(\cdot)$, and then choose $\widehat{d} = \arg\max_{d\in\mathcal{D}} \widehat{V}(d)$ as the estimator of d^*. Thus, a complex model space for $V(\cdot)$ need not imply a complex class of rules $\mathcal{D}$. However, the class of models for $V(\cdot)$ must be sufficiently rich to avoid implicit, unintended restrictions on $\widehat{d}$. To avoid such restrictions and to avoid model-misspecification, it is common to use a flexible class of semi- or non-parametric models for $V(\cdot)$.

2.2. Augmentation for the value function

Define the propensity score $\pi(a;\boldsymbol{x}) \triangleq P(A = a|\boldsymbol{X} = \boldsymbol{x})$, then

$$V(d) = E\left[\frac{Y}{\pi(A;\boldsymbol{X})} I\{A = d(\boldsymbol{X})\}\right], \tag{2}$$

where $I\{\cdot\}$ denotes the indicator function (e.g., Qian and Murphy, 2011). Unlike (1), the preceding expression does not require an estimator of the Q-function. Given an estimator of the propensity score, $\widehat{\pi}(a;\boldsymbol{x})$, a plug-in estimator for $V(d)$ based on (2) is the inverse probability weighted estimator (IPWE) $\widehat{V}^{\text{IPWE}}(d) \triangleq \mathbb{P}_n[YI\{A = d(\boldsymbol{X})\}/\widehat{\pi}(A;X)]$, where $\mathbb{P}_n$ is the empirical distribution. The IPWE has potentially high variance as it only uses outcomes from subjects whose treatment assignments coincide with those recommended by d.

One approach to reduce variability is to augment the IPWE with a term involving both the propensity score and the Q-function that is estimated using data from all of the observed subjects (Robins et al., 1994; Cao et al., 2009). Let $\widehat{Q}(\boldsymbol{x}, a)$ denote an estimator of $Q(\boldsymbol{x}, a)$. The augmented inverse probability weighted estimator is

$$\widehat{V}^{\text{AIPWE}}(d) \triangleq \mathbb{P}_n\left[\frac{YI\{A = d(\boldsymbol{X})\}}{\widehat{\pi}\{d(X);X\}} - \frac{I\{A = d(\boldsymbol{X})\} - \widehat{\pi}\{d(\boldsymbol{X});\boldsymbol{X}\}}{\widehat{\pi}\{d(\boldsymbol{X});\boldsymbol{X}\}}\widehat{Q}\{\boldsymbol{X}, d(\boldsymbol{X})\}\right]. \tag{3}$$

It can be seen that $\widehat{V}^{\text{AIPWE}}(d)$ is equal to $\widehat{V}^{\text{IPWE}}(d)$ plus an estimator of zero built using outcomes from all subjects regardless of whether or not their treatment assignment is consistent with d. If $\widehat{Q}(\boldsymbol{x}, a) \equiv 0$ then $\widehat{V}^{\text{AIPWE}}(d) = \widehat{V}^{\text{IPWE}}(d)$ for all d.

Hereafter, we use $\widehat{Q}(\boldsymbol{x}, a)$ and $\widehat{\pi}(a;\boldsymbol{x})$ to denote generic estimators of the Q-function and propensity score. The following assumption is used to establish double robustness of $\widehat{V}^{\text{AIPWE}}(d)$.

Assumption 1 *$\widehat{Q}(\boldsymbol{x}, a)$ and $\widehat{\pi}(a;\boldsymbol{x})$ converge in probability uniformly to deterministic limits $Q^m(\boldsymbol{x}, a)$ and $\pi^m(a;\boldsymbol{x})$.*

This assumption does not require that the estimators $\widehat{Q}(\boldsymbol{x}, a)$, $\widehat{\pi}(a;\boldsymbol{x})$ are consistent for the truth, only that they converge to fixed functions. The following result is proved in Web Appendix A.

Lemma 2.1 *Let $d \in \mathcal{D}$ be fixed. If* either $\pi^m(a;\boldsymbol{x}) = \pi(a;\boldsymbol{x})$ *or* $Q^m(\boldsymbol{x}, a) = Q(\boldsymbol{x}, a)$ *for all $(\boldsymbol{x}, a)$ outside of a set of measure zero, then $\widehat{V}^{AIPWE}(d) \to_p V^{AIPWE,m}(d) = V(d)$, where*

$$V^{AIPWE,m}(d) \triangleq E\left[\frac{YI\{A = d(\boldsymbol{X})\}}{\pi^m(A;\boldsymbol{X})} - \frac{I\{A = d(\boldsymbol{X})\} - \pi^m\{d(\boldsymbol{X});\boldsymbol{X}\}}{\pi^m\{d(\boldsymbol{X});\boldsymbol{X}\}}Q^m\{\boldsymbol{X}, d(\boldsymbol{X})\}\right].$$

The preceding result shows that $\widehat{V}^{AIPWE}(d)$ is doubly-robust in the sense that if either the propensity model or the modeled Q-function is consistent, but not necessarily both, then $\widehat{V}^{AIPWE}(d)$ is consistent for $V(d)$. Thus, the maximizer of $\widehat{V}^{AIPWE}(d)$ over $d \in \mathcal{D}$ is termed a doubly-robust estimator of the optimal treatment rule (Zhang et al., 2012a, 2012b, 2013). However, because $\widehat{V}^{AIPWE}(d)$ is not continuous, computing this doubly-robust estimator can be computationally infeasible even in moderate problems (Zhang et al., 2012a). Instead, we form an estimator by maximizing a concave relaxation of $\widehat{V}^{AIPWE}(d)$. Maximizing this concave relaxation is computationally efficient even in very high-dimensional problems. We show that the maximizer of this relaxed criteria remains doubly-robust. Furthermore, we show that the rates of convergence of the proposed estimators depend on the chosen concave relaxation, the chosen propensity model, and the chosen model for the Q-function. The relationships among these choices provides new knowledge about direct search estimators based on concave surrogates (Zhang et al., 2012; Zhao et al., 2012, 2015a, 2015b).

2.3. Efficient augmentation and relaxation learning (EARL)

Let $\mathcal{M}$ be the class of measurable functions from $\mathbb{R}^p$ into $\mathbb{R}$. Any decision rule $d(\boldsymbol{x})$ can be written as $d(\boldsymbol{x}) = \text{sgn}\{f(\boldsymbol{x})\}$ for some function $f \in \mathcal{M}$, where we define $\text{sgn}(0) = 1$. For $d(\boldsymbol{x}) = \text{sgn}\{f(\boldsymbol{x})\}$, $I\{a = d(\boldsymbol{x})\} = I\{af(\boldsymbol{x}) \geq 0\}$. Define $V(f)$, $V^{\text{IPWE},m}(f)$, and $V^{\text{AIPWE},m}(f)$ by substituting $I\{Af(\boldsymbol{X}) \geq 0\}$ for $I\{A = d(\boldsymbol{X})\}$ in their respective definitions. Define

$$W_a^m = W_a(Y, \boldsymbol{X}, A, \pi^m, Q^m) = \frac{YI(A = a)}{\pi^m(a;\boldsymbol{X})} - \frac{I(A = a) - \pi^m(a;\boldsymbol{X})}{\pi^m(a;\boldsymbol{X})}Q^m(\boldsymbol{X}, a), a \in \{-1, 1\}.$$

The following result shows that maximizing $\widehat{V}^{\text{AIPWE}}(f)$ is equivalent to minimizing a sum of weighted misclassification rates; a proof is given in Web Appendix B.

Lemma 2.2 *Assume that $P\{f(\boldsymbol{X}) = 0\} = 0$. Define $\widehat{f}_n = \arg\sup_{f\in\mathcal{M}} \widehat{V}^{\text{AIPWE}}(f)$, then*

$$\widehat{f}_n = \arg\inf_{f\in\mathcal{M}} \mathbb{P}_n\left[|\widehat{W}_1|I\left\{\text{sgn}(\widehat{W}_1)f(\boldsymbol{X}) < 0\right\} + |\widehat{W}_{-1}|I\left\{-\text{sgn}(\widehat{W}_{-1})f(\boldsymbol{X}) < 0\right\}\right],$$

where $\widehat{W}_a = W_a(Y, \boldsymbol{X}, A, \widehat{\pi}, \widehat{Q}), a =\in \{-1, 1\}$.

Lemma 2.2 shows that the estimator, $\widehat{f}_n$, which maximizes $\widehat{V}^{\text{AIPWE}}(f)$ over $f \in \mathcal{M}$, can be viewed as minimizing a sum of weighted 0-1 losses. In this view, the class labels are $\text{sgn}(\widehat{W}_a) \cdot a$ and the misclassification weights are $|\widehat{W}_a|$, $a =\in \{-1, 1\}$ (see Zhang et al., 2012b, 2013). Directly minimizing the combined weighted 0-1 loss is a difficult non-convex optimization problem (Laber and Murphy, 2011). One strategy to reduce computational complexity is to replace the indicator function with a convex surrogate and to minimize the resulting relaxed objective function (Freund and Schapire, 1999; Bartlett et al., 2006; Hastie et al., 2009). This strategy has proved successful empirically and theoretically in classification and estimation of optimal treatment rules (Zhao et al., 2012). However, unlike previous applications of convex relaxations to the estimation of optimal treatment rules, we establish rates of convergence as a function of the: (i) choice of convex surrogate; (ii) convergence rate of the postulated propensity score estimator; and (iii) convergence rate

the postulated Q-function estimator. We characterize the relationship among these three components in Section 3.

The function f is conceptualized as being a smooth function of $\boldsymbol{x}$ that is more easily constrained to possess certain desired structure, e.g., sparsity, linearity, etc. Thus, we will focus on estimation of f within a class of functions $\mathcal{F}$ called the approximation space; we assume that $\mathcal{F}$ is a Hilbert space with norm $\|\cdot\|$. Let $\phi : \mathbb{R} \to \mathbb{R}$ denote a convex function and define EARL estimators as those taking the form

$$\tilde{f}_n^{\lambda_n} = \arg\inf_{f\in\mathcal{F}} \mathbb{P}_n\left[|\widehat{W}_1|\phi\left\{\mathrm{sgn}(\widehat{W}_1)f(\boldsymbol{X})\right\} + |\widehat{W}_{-1}|\phi\left\{-\mathrm{sgn}(\widehat{W}_{-1})f(\boldsymbol{X})\right\}\right] + \lambda_n\|f\|^2, \quad (4)$$

where $\lambda_n\|f\|^2$ is included to reduce overfitting and $\lambda_n \geq 0$ is a (possibly data-dependent) tuning parameter. Throughout, we assume that $\phi(t)$ is one of the following: hinge loss, $\phi(t) = \max(1-t, 0)$; exponential loss, $\phi(t) = e^{-t}$; logistic loss, $\phi(t) = \log(1+e^{-t})$; or squared hinge loss, $\phi(t) = \{\max(1-t,0)\}^2$. However, other convex loss functions are possible provided that they are differentiable, monotone, strictly convex, and satisfy $\phi(0) = 1$ (Bartlett et al., 2006). As noted previously, Zhao et al. (2012) proposed a special case of EARL called outcome weighted learning, which set $\phi(t) = \max(0, 1-t)$, $\widehat{Q}(\boldsymbol{x}, a) \equiv 0$, and assumed that the propensity score was known. Thus, as noted previously, EARL is considerably more general than OWL and, as shown in Section 4, the choice of a non-null model for the Q-function and alternative surrogate loss functions can lead to dramatically improved finite sample performance.

2.4. EARL via sample splitting

To facilitate the analysis of the statistical properties of EARL, we consider the following alternative estimator based on the sample splitting. Let $I_1, I_2, ..., I_K$ denote a random partition of the indices $\{1, 2, ..., n\}$ with $I_j \cap I_k = \emptyset$ for any $j \neq k$ and $\cup_{k=1}^K I_k = \{1, 2, ..., n\}$. We assume the size of the partitions is comparable, that is, $n_k = |I_k|$ with $n_1 \asymp n_2 \asymp ... \asymp n_K$. In practice, K is taken as a small integer (e.g., 2, or 5) and is assumed fixed. Recall that the EARL estimator based on the full sample is defined in (4). In particular, the same samples are used to estimate the nuisance functions $\hat{\pi}, \hat{Q}$ and construct the estimator $\tilde{f}_n^{\lambda_n}$ in (4). This creates the delicate dependence between the estimators $\hat{\pi}, \hat{Q}$ and the samples used in the empirical risk minimization in (4). To remove this dependence, we now modify the procedure via sample splitting. First, for $1 \leq k \leq K$, we construct estimators $\hat{\pi}_k, \hat{Q}_k$ based on the samples in I_k, i.e., $\{(\boldsymbol{X}_i, A_i, Y_i); i \in I_k\}$. Denote $I_{(-k)} = \{1, ..., n\} \backslash I_k$. Then, we use the remaining samples $I_{(-k)}$ for the EARL estimator

$$\widehat{f}_{n,k}^{\lambda_{nk}} = \arg\inf_{f\in\mathcal{F}} \mathbb{P}_n{}^{(-k)}\left[|\widehat{W}_{1k}|\phi\left\{\mathrm{sgn}(\widehat{W}_{1k})f(\boldsymbol{X})\right\} + |\widehat{W}_{-1k}|\phi\left\{-\mathrm{sgn}(\widehat{W}_{-1k})f(\boldsymbol{X})\right\}\right] + \lambda_{nk}\|f\|^2, \quad (5)$$

where $\widehat{W}_{ak} = W_a(Y, \boldsymbol{X}, A, \hat{\pi}_k, \widehat{Q}_k), a = \in \{-1, 1\}$ and $\mathbb{P}_n{}^{(-k)} f = \frac{1}{|I_{(-k)}|}\sum_{i\in I_{(-k)}} f(X_i)$. We note that independent samples are used for estimating the nuisance functions π, Q and the decision rule f. Thus, the dependence between the estimators $\hat{\pi}, \hat{Q}$ and the samples use in (4) is removed. Finally, to obtain a more stable estimator, we can aggregate the estimators

$$\widehat{f}_n^{\lambda_n} = \frac{1}{K}\sum_{k=1}^K \widehat{f}_{n,k}^{\lambda_{nk}}, \quad (6)$$

which is the final estimator based on sample splitting. While the estimator $\widehat{f}_n^{\lambda_n}$ requires more computational cost, it has important advantages over the original EARL estimator $\tilde{f}_n^{\lambda_n}$ in (4). From a theoretical perspective, one can still analyze the EARL estimator $\tilde{f}_n^{\lambda_n}$ based on the empirical process theory. This typically requires the entropy conditions on the function classes of π and Q. In comparison, we show in the following section that the sample splitting estimator $\widehat{f}_n^{\lambda_n}$ does not require this condition. To the best of our knowledge, similar sample splitting technique was first applied by Bickel (1982) in general semiparametric estimation problems; see also Schick (1986). Recently, this approach has received attention in causal inference problems as a means of relaxing technical conditions. We refer to Zheng and van der Laan (2011); Chernozhukov et al. (2016); Robins et al. (2017) for further discussion.

3. Theoretical properties

Let $f^* \in \mathcal{M}$ be such that $d^*(\boldsymbol{x}) = \mathrm{sgn}\{f^*(\boldsymbol{x})\}$, and $V^* \triangleq \sup_{f\in\mathcal{M}} V(f) = V(f^*)$. Define the population risk of function f as

$$\mathcal{R}(f) = E(YI[A \neq \mathrm{sgn}\{f(\boldsymbol{X})\}]/\pi(A; \boldsymbol{X})),$$

and $\mathcal{R}^* \triangleq \inf_{f\in\mathcal{M}} \mathcal{R}(f)$. We define the risk in this way to be consistent with the convention that higher risk is less desirable; however, inspection shows that the risk equals $K - V(f)$ where K is a constant that does not depend on f. Thus, minimizing risk is equivalent to maximizing value, and $V^* - V(f) = \mathcal{R}(f) - \mathcal{R}^*$. Accordingly, for a convex function ϕ, we define the ϕ-risk

$$\mathcal{R}_\phi^m(f) = E[|W_1^m|\phi\left\{\mathrm{sgn}(W_1^m)f(\boldsymbol{X})\right\} + |W_{-1}^m|\phi\left\{-\mathrm{sgn}(W_{-1}^m)f(\boldsymbol{X})\right\}].$$

By construction, $\mathcal{R}_\phi^m(f)$ is convex; we assume that it has a unique minimizer and that $\mathcal{R}_\phi^{m*} \triangleq \inf_{f\in\mathcal{M}} \mathcal{R}_\phi^m(f)$. The following result is proved in Web Appendix C.

Proposition 3.1 *Assume that either $\pi^m(a; \boldsymbol{x}) = \pi(a; \boldsymbol{x})$ or $Q^m(\boldsymbol{x}, a) = Q(\boldsymbol{x}, a)$. Define $\tilde{f} = argmin_{f\in\mathcal{M}} R_\phi^m(f)$ and $c_m(\boldsymbol{x}) = E\{|W_1(Y, \boldsymbol{x}, A, \pi^m, Q^m)| + |W_{-1}(Y, \boldsymbol{x}, A, \pi^m, Q^m)|\}$. Then:*

(a) $d^*(\boldsymbol{x}) = \mathrm{sgn}\{\tilde{f}(\boldsymbol{x})\}$;

(b) and

$$\psi\left\{\frac{V^* - V(f)}{\sup_{\boldsymbol{x}\in\mathbb{R}^p} c_m(\boldsymbol{x})}\right\} \leq \frac{\mathcal{R}_\phi^m(f) - \mathcal{R}_\phi^{m*}}{\inf_{\boldsymbol{x}\in\mathbb{R}^p} c_m(\boldsymbol{x})},$$

where $\psi(\theta) = |\theta|$ for hinge loss, $\psi(\theta) = 1 - \sqrt{1-\theta^2}$ for exponential loss, $\psi(\theta) = (1+\theta)\log(1+\theta)/2 + (1-\theta)\log(1-\theta)/2$ for logistic loss, and $\psi(\theta) = \theta^2$ for squared hinge loss.

Part (a) of the preceding proposition states that if either the model for the propensity score or for the Q-function is correctly specified, then the EARL procedure, optimized over the space of measurable functions, is Fisher consistent for the optimal rule. Part (b) bounds the difference between $V(f)$ and V^* through the surrogate risk difference $\mathcal{R}^m_\phi(f) - \mathcal{R}^{m*}_\phi$. The different forms of $\psi(\cdot)$ are due to the fact that different loss functions induce different distance measures of closeness of $f(x)$ to the true $f^*(x)$. We use these risk bounds to derive bounds on the convergence rates of the value of EARL estimators constructd using sample splitting.

Let Π denote the function spaces to which the postulated models for $\pi(a;\boldsymbol{x})$ belong; that is, the estimator $\widehat{\pi}(a;\boldsymbol{x})$ belongs to Π. Similarly, let $\mathcal{Q}$ denote a postulated class of models for $Q(\boldsymbol{x},a)$. In this section, we allow the approximation space, $\mathcal{F}$, to be arbitrary subject to complexity constraints; our results allow both parametric or non-parametric classes of models. Our primary result is a bound on the rate of convergence of $V^* - V(\widehat{f}^{\lambda_n}_n)$ in terms of the ϕ-risk difference $\mathcal{R}^m_\phi(\widehat{f}^{\lambda_n}_n) - \mathcal{R}^{m*}_\phi$.

For any $\epsilon > 0$ and measure P, let $N\{\epsilon, \mathcal{F}, L_2(P)\}$ denote the covering number of the space $\mathcal{F}$, that is, $N\{\epsilon, \mathcal{F}, L_2(P)\}$ is the minimal number of closed $L_2(P)$-balls of radius ϵ required to cover $\mathcal{F}$ (Kosorok, 2008). Denote $\|f\|^2_{P,2} = Ef^2(\boldsymbol{X})$. We make the following assumptions.

Assumption 2 *There exists $M_Q > 0$ such that $|Y| \le M_Q$ and $|Q(\boldsymbol{x},a)| \le M_Q$ for all $(\boldsymbol{x},a) \in \mathbb{R}^p \times \{-1,1\}$ and $Q \in \mathcal{Q}$; there exists $0 < L_\Pi < M_\Pi < 1$ such that $L_\Pi \le \pi(a;\boldsymbol{x}) \le M_\Pi$ for all $(\boldsymbol{x},a) \in \mathbb{R}^p \times \{-1,1\}$ and $\pi \in \Pi$.*

Assumption 3 *There exist constants $0 < v < 2$ and $c < \infty$ such that for all $0 < \epsilon \le 1$: $\sup_P \log N\{\epsilon, \mathcal{F}, L_2(P)\} \le c\epsilon^{-v}$, where the supremum is taken over all finitely discrete probability measures P.*

Assumption 4 *For some $\alpha, \beta > 0$, $E\|\widehat{\pi}_k(a;\boldsymbol{x}) - \pi(a;\boldsymbol{x})\|^2_{P,2} = O(n^{-2\alpha})$ and $E\|\widehat{Q}_k(\boldsymbol{x},a) - Q(\boldsymbol{x},a)\|^2_{P,2} = O(n^{-2\beta})$ for $a = \pm 1$ and $1 \le k \le K$.*

Assumption 2 assumes outcomes are bounded, which often holds in practice. Otherwise, we can always use a large constant to bound the outcome. We also assume propensity scores are bounded away from 0 and 1, which is a standard condition for the identification of the treatment effect in causal inference. Assumption 3 controls the complexity of the function spaces for estimating an optimal ITR. For example, if $\mathcal{F}$ is composed of linear combinations of elements in a fixed base class, $\mathcal{H}$, where $\mathcal{H}$ has finite Vapnik-Chervonenkis (VC) dimension vc, then there exists a constant c_{vc}, depending on vc, so that $\sup_P \log N\{\epsilon, \mathcal{F}, L_2(P)\} \le c_{vc}\epsilon^{-2vc/(vc+2)}$ (Theorem 9.4, Kosorok (2008)). We note that the entropy conditions on $\mathcal{Q}$ and Π are not needed by using the sample splitting technique, due to the independence between estimating π, Q and estimating f.

Assumption 4 specifies the rate of convergence of the estimators $\widehat{\pi}$ and $\widehat{Q}$ in terms of the $\|\cdot\|_{P,2}$ norm. It is well known that the L_2 rate of convergence is related to the smoothness of the function classes $\mathcal{Q}$ and Π and the dimension of $\boldsymbol{X}$. For instance, if $\mathcal{Q}$ corresponds to the Holder class with smoothness parameter s on the domain $[0,1]^p$, then Theorem 7 of Newey (1997) implies $E\|\widehat{Q}(\boldsymbol{x},a) - Q(\boldsymbol{x},a)\|^2_{P,2} = O_p(K/n + K^{-2s/p})$, where $\widehat{Q}(\boldsymbol{x},a)$ is the regression spline estimator and K is the number of basis functions.

Define the approximation error incurred by optimizing over $\mathcal{F}$ as

$$\mathcal{A}(\lambda_n) = \inf_{f\in\mathcal{F}} \left(\lambda_n\|f\|^2 + \sum_{a=\pm1} E\left[W^m_a \phi\{a\cdot f(\boldsymbol{X})\}\right]\right) - \inf_{f\in\mathcal{M}} \sum_{a=\pm1} E\left[W^m_a \phi\{a\cdot f(\boldsymbol{X})\}\right]. \quad (7)$$

The following result on the risk bound is the main result in this section and is proved in the Web Appendix D.

Theorem 3.1 *Suppose that assumptions 1-4 hold, $\lambda_n \to 0$. Define $c_m(\boldsymbol{x}) = E\{|W^m_1||\boldsymbol{X} = \boldsymbol{x}, A = 1\} + E\{|W^m_{-1}||\boldsymbol{X} = \boldsymbol{x}, A = -1\}$. If $Q^m(\boldsymbol{x},a) = Q(\boldsymbol{x},a)$ and $\pi^m(a;\boldsymbol{x}) = \pi(a;\boldsymbol{x})$, then*

$$\psi\left\{\frac{V^* - V(\widehat{f}^{\lambda_n}_n)}{\sup_{\boldsymbol{x}\in\mathbb{R}^p} c_m(\boldsymbol{x})}\right\} \lesssim \frac{1}{\inf_{\boldsymbol{x}\in\mathbb{R}^p} c_m(\boldsymbol{x})} \cdot \Big[\mathcal{A}(\lambda_n) + n^{-\frac{2}{v+2}}\lambda_n^{-\frac{v}{v+2}} + n^{-1}\lambda_n^{-1} + \lambda_n^{-1/2}n^{-(\alpha+\beta)} + \lambda_n^{-1/2}(n^{-(1/2+\alpha)} + n^{-(1/2+\beta)})\Big].$$

In all cases considered, the function ψ is invertible on $[0,1]$, and its inverse is monotone non-decreasing. Thus, for sufficiently large n (making the right-hand-side of the equation sufficiently small) the inequality can be re-arranged to yield a bound on $V^* - V(\widehat{f}^{\lambda_n}_n)$. The form of ψ^{-1} dictates the tightness of the bound as a function of the ϕ-risk. According to Lemma 3 in Bartlett et al (2006), a flatter loss function leads to better bound on ψ function. In other words, a flatter loss function gives better bounds on $V^* - V(f)$ in terms of $\mathcal{R}^m_\phi(f) - \mathcal{R}^{m*}_\phi$. In this respect, hinge-loss can be seen to provide the tightest bound; however, the ϕ-risk is not directly comparable across different loss functions as they are not on the same scale.

The right hand side of the bound in Theorem 3.1 consists of three parts: the approximation error $\mathcal{A}(\lambda_n)$ due to the size of the approximation space $\mathcal{F}$, the error $n^{-\frac{2}{v+2}}\lambda_n^{-\frac{v}{v+2}} + n^{-1}\lambda_n^{-1}$ due to the estimation in the function space $\mathcal{F}$, and the error $\lambda_n^{-1/2}n^{-(\alpha+\beta)} + \lambda_n^{-1/2}(n^{-(1/2+\alpha)} + n^{-(1/2+\beta)})$ incurred from plugging the estimators $\widehat{\pi}_k$ and $\widehat{Q}_k$. As expected, the approximation error decreases as the complexity of the class $\mathcal{F}$ increases, whereas the estimation error increases with the complexity of the class $\mathcal{F}$ and decreases as the sample size increases.

For the error incurred from plugging the estimators $\widehat{\pi}_k$ and $\widehat{Q}_k$, the component $\lambda_n^{-1/2}(n^{-(1/2+\alpha)} + n^{-(1/2+\beta)})$ converges to 0 faster than $\lambda_n^{-1/2}n^{-(\alpha+\beta)}$ in regular statistical models (i.e., $\alpha, \beta \le 1/2$). Thus, it suffices to only look at the term $\lambda_n^{-1/2}n^{-(\alpha+\beta)}$. This term can shrink to 0 sufficiently fast as long as one of the estimators $\widehat{\pi}_k$ and $\widehat{Q}_k$ has a fast rate, due to the multiplicative form of the estimation error. For example, if $\alpha = \beta = 1/4$, the error from plugging the estimators $\widehat{\pi}$ and $\widehat{Q}$ is $n^{-1/2}\lambda_n^{-1/2}$. Hence, the rate of the proposed method is faster compared with the outcome weighted learning method, which is developed based on an IPWE and does not enjoy this multiplicative form of the errors. This phenomenon can be viewed as a nonparametric version of the double robustness property (see Fan et al., 2016; Benkeser et al., 2017, for additional discussion). Compared with the results in Athey and Wager (2017), we allow for the surrogate loss to replace the 0-1 loss in solving for the optimizer. While the orders in the bound of convergence rates are comparable, the differences in the constants in the bounds might be due to the application of the surrogate function.

Remark 1 *If $\alpha = \beta$ and*

$$n^{2\alpha-1}\lambda_n^{-1/2} \to \infty, \text{ or } n^{2\alpha(v+2)-2}\lambda_n^{1-v/2} \to \infty, \tag{8}$$

then

$$\psi\left\{\frac{V^* - V(\widehat{f}_n^{\lambda_n})}{\sup_{\boldsymbol{x}\in\mathbb{R}^p} c_m(\boldsymbol{x})}\right\} \lesssim \frac{1}{\inf_{\boldsymbol{x}\in\mathbb{R}^p} c_m(\boldsymbol{x})} \cdot \left[\mathcal{A}(\lambda_n) + n^{-\frac{2}{v+2}}\lambda_n^{-\frac{v}{v+2}} + n^{-1}\lambda_n^{-1}\right], \tag{9}$$

where the upper bound is of the same order as that obtained if the conditional mean $Q(\boldsymbol{x}, a)$ and propensity score $\pi(a; \boldsymbol{x})$ are known. We note that the additional constraints on α and v in (8) are necessary to obtain the fast rate of convergence (9). For instance, if the function classes $\mathcal{Q}$ and Π are indexed by finite dimensional parameters, we can obtain $\alpha = \beta = 1/2$ under mild conditions. As a result, the first condition in (8) holds and the fast rate of convergence (9) is applied. On the other hand, if $\mathcal{F}$ is a simple class but $\|\widehat{\pi} - \pi\|_{P,2}$ and $\|\widehat{Q} - Q\|_{P,2}$ converge at slower rates, the rate for $V^* - V(\widehat{f}_n^{\lambda_n})$ will be driven by $\lambda_n^{-1/2} n^{-(\alpha+\beta)}$.

To estimate the value of the optimal treatment rule V^*, one can aggregate the empirical value of the sample splitting estimator $\hat{f}_{n,k}^{\lambda_{n,k}}$ in each subsamples $I_{(-k)}$, that is, $\bar{V} = \frac{1}{K}\sum_{k=1}^K \hat{V}_{(-k)}(\hat{f}_{n,k}^{\lambda_{n,k}})$, where

$$\hat{V}_{(-k)}(f) = \mathbb{P}_n{}^{(-k)}\left[|\widehat{W}_{1k}|\phi\left\{\text{sgn}(\widehat{W}_{1k})f(\boldsymbol{X})\right\} + |\widehat{W}_{-1k}|\phi\left\{-\text{sgn}(\widehat{W}_{-1k})f(\boldsymbol{X})\right\}\right],$$

The following corollary, provides a corresponding bound on the rate for $V^* - \bar{V}$. The proof is given in Web Appendix D.

Corollary 3.1 *Suppose that assumptions 1-4 hold, and $\lambda_n \to 0$. If $Q^m(\boldsymbol{x}, a) = Q(\boldsymbol{x}, a)$ and $\pi^m(a; \boldsymbol{x}) = \pi(a; \boldsymbol{x})$, then*

$$V^* - \bar{V} \lesssim \frac{1}{K}\sum_{k=1}^K [V^* - V(\hat{f}_{n,k}^{\lambda_{n,k}})] + n^{-1/2} + \lambda_n^{-1/2} n^{-(\alpha+\beta)} + \lambda_n^{-1/2}(n^{-(1/2+\alpha)} + n^{-(1/2+\beta)}),$$

where

$$\frac{1}{K}\sum_{k=1}^K [V^* - V(\hat{f}_{n,k}^{\lambda_{n,k}})] \lesssim \sup_{\boldsymbol{x}\in\mathbb{R}^p} c_m(\boldsymbol{x})\psi^{-1}\Big[\frac{1}{\inf_{\boldsymbol{x}\in\mathbb{R}^p} c_m(\boldsymbol{x})} \cdot \Big\{\mathcal{A}(\lambda_n) + n^{-\frac{2}{v+2}}\lambda_n^{-\frac{v}{v+2}} + n^{-1}\lambda_n^{-1} + \lambda_n^{-1/2} n^{-(\alpha+\beta)} + \lambda_n^{-1/2}(n^{-(1/2+\alpha)} + n^{-(1/2+\beta)})\Big\}\Big].$$

Remark 2 *Athey and Wager (2017) and Kitagawa and Tetenov (2017) investigated the binary-action policy learning problem, and established a risk bound of $n^{-1/2}$ for both known propensities (Kitagawa and Tetenov, 2017) and unknown propensities (Athey and Wager, 2017). However, they considered a restricted class of decision rules and subsequent risk bound were established with respect to the optimal rule within this restricted class. Hence, there was not consideration of the approximation error. In contrast, we considered the optimal rule within the space consisting of all measurable functions from $\mathbb{R}^p$ (the covariate space) to $\{-1, 1\}$ (the treatment space). We used a smaller space, for example, a reproducing kernel Hilbert space, to approximate the policy space and to avoid overfitting. This led to a tradeoff between approximation and estimation error, and λ_n was a tuning parameter to control this bias-variance tradeoff. Consequently, the achieved convergence rates are different.*

4. Simulation experiments

We compare EARL estimators with: Q-learning fit using ordinary least squares (QL, Qian and Murphy, 2011); estimating the optimal rule within a restricted class based on an AIPW estimator (AIPWE, Zhang et al., 2012b); and outcome weighted learning (OWL, Zhao et al., 2012). Comparisons are made in terms of the average value of the rule estimated by each method. For Q-learning, we fit a linear model for the Q-function that includes all two-way interactions between predictors and pairwise interactions between these terms and treatment. In the AIPWE method, an AIPW estimator for the value function is constructed and then the optimal linear rule that maximizes the AIPW estimator is identified via a genetic algorithm. Similar to EARL, both a propensity score model and a regression model need to be fitted in AIPWE. We will use the same set of models in EARL and the AIPWE, which are detailed in below. For OWL, we use a linear decision rule; recall that OWL is a special case of EARL with $\widehat{Q}(\boldsymbol{x}, a) \equiv 0$, $\phi(t) = \max(0, t)$, and a known propensity score. All estimation methods under consideration require penalization; we choose the amount of penalization using 10-fold cross-validation of the value. Within the class of EARL estimators, we considered hinge, squared-hinge, logistic, and exponential convex surrogates. An implementation of EARL is available in the R package 'DynTxRegime;' this package also includes implementations of AIPWE and OWL and therefore can be used to replicate the simulation studies presented here. We included an example for implementing EARL method using 'DynTxRegime' package in Web Appendix H.

We consider generative models of the form: $\boldsymbol{X} = (X_1, \ldots, X_p) \sim_{i.i.d.} N(0, 1)$ with $p = 10$; treatments are binary, taking the values in $\{-1, 1\}$ according to the model $p(A = 1|\boldsymbol{X}) = \exp\{\ell(\boldsymbol{X})\}/[1 + \exp\{\ell(\boldsymbol{X})\}]$, where $\ell(\boldsymbol{x}) = x_1 + x_2 + x_1x_2$ in Scenario 1, and $\ell(\boldsymbol{x}) = 0.5x_1 - 0.5$ in Scenario 2; $Y = \sum_{j=1}^p X_j^2 + \sum_{j=1}^p X_j + Ac(\boldsymbol{X}) + \epsilon$, where $\epsilon \sim N(0, 1)$, and $c(\boldsymbol{x}) = x_1 + x_2 - 0.1$. Write $\boldsymbol{X}^2$ to denote $(X_1^2, \ldots, X_p^2)$. The following modeling choices are considered for the propensity and outcome regression models.

CC. A correctly specified logistic regression model for $\pi(A; \boldsymbol{X})$ with predictors X_1, X_2 and X_1X_2 in Scenario 1, and with predictor X_1 in Scenario 2; and a correctly specified linear regression model for $Q(\boldsymbol{X}, A)$ with predictors $\boldsymbol{X}, \boldsymbol{X}^2, A, X_1A$ and X_2A in both scenarios.

CI. A correctly specified logistic regression model for $\pi(A; \boldsymbol{X})$ with predictors X_1, X_2 and X_1X_2 in Scenario 1, and with predictor X_1 in Scenario 2; and an incorrectly specified linear model for $Q(\boldsymbol{X}, A)$ with predictors $\boldsymbol{X}, A, \boldsymbol{X}A$ in both scenarios.

IC. An incorrectly specified logistic regression model for $\pi(A; \boldsymbol{X})$ with predictors $\boldsymbol{X}$ in Scenario 1, and without any predictors in Scenario 2; and a correctly specified linear model for $Q(\boldsymbol{X}, A)$ with predictors $\boldsymbol{X}, \boldsymbol{X}^2, A, X_1A$ and X_2A in both scenarios.

II. An incorrectly specified logistic regression model for $\pi(A; \boldsymbol{X})$ with predictors $\boldsymbol{X}$ in Scenario 1, and without any predictors in Scenario 2; and an incorrectly specified linear model for $Q(\boldsymbol{X}, A)$ with predictors $\boldsymbol{X}, A, \boldsymbol{X}A$ in both scenarios.

We use the same model specifications to carry out AIPWE, and denote them as CC-A, CI-A, IC-A, and II-A correspondingly. For the OWL method, we use correct and incorrect propensity models to construct the ITRs, and denote them as C. and I. respectively. Similarly, we use Q-learning to construct the ITRs based on correct and incorrect regression models, and term them as .C and .I respectively.

We consider sample sizes 200, 500, 1000, 2500, 5000 and 10000. We generate a large validation data set (size 10000) and 500 training sets under each sample size. The ITRs are constructed based on one training set out of 500 replicates using competing methods. For implementing EARL, we use logistic loss. We observe similar patterns for other surrogate loss functions (see the Web Appendix). We carry out cross-validation to select λ_n among a pre-specified set of values $(2^{-5}, 2^{-4}, \ldots, 2^5)$. Then we calculate the mean response had the whole population followed the rule (the value function) by averaging the outcomes over 10000 subjects under the estimated ITRs in the validation data set. Thus, there are 500 values of the estimated rules on the validation set for each sample size. Boxplots of these values are shown in Figures 1 and 3. The performance of OWL was generally worse than that of the EARL estimator or QL. The AIPWE method exhibits a larger bias and a higher variance compared to the proposed method, while running approximately 200 times slower. As expected, the QL method works best when the model is correctly specified but can perform poorly when this model is misspecified.

It appears that misspecification of the model for the Q-function has a bigger impact than misspecification of the propensity score model on the AIPWE and EARL methods. The relatively poor performance when the propensity is correctly specified but the regression model is not might be attributed in part to inverse weighting by the propensity score, which is problematic when some estimated propensity scores are close to zero, yielding large weights and subsequently induces bias (Kang and Schafer, 2007). This is illustrated by contrasting scenarios 1 and 2. Propensity scores in Scenario 2 are bounded away from zero, which yield a better result compared to Scenario 1. Furthermore, the large variability when the regression model is misspecified may be partly a consequence of the method used to estimate the coefficients in the regression model (see Cao et al., 2009).

Finally, we consider an example to illustrate the impact of a severely misspecified propensity score model. In Scenario 3, the data was generated as in Scenario 2 except that the propensity score was set to 0.025 for all subjects. The 'CI' setup outperformed the 'IC' setup, especially when the sample size was small. Furthermore, the performance of the AIPWE method was largely affected by this poorly imposed propensity model. The results of 'CI' and 'II' setups were unsatisfactory even when the sample size was increased to 10000. This example indicates that the performances in the 'CI' and 'IC' setups depend on the degree of misspecification in the outcome regression model and propensity score model.

We also conducted a set of simulation experiments to investigate the role of parametric and nonparametric models for the propensity score and outcome regression. In addition, we compared the performance across different surrogate loss functions, including logistic loss, exponential loss, squared hinge loss, and hinge loss. These additional simulation results can be found in Web Appendix F. In summary, we found that in the examples considered, using

nonparametric working models for propensity scores could improve results over parametric models. Hinge loss has a more robust performance when the regression model is incorrect compared to other smooth losses.

5. Application: Ocean State Crohn's and Colitis Area Registry (OSCCAR)

OSCCAR is a community—based incident cohort of subjects with inflammatory bowel disease (IBD) residing in the state of Rhode Island that was established in 2008 (Sands et al., 2009). Subjects enrolled in OSCCAR have ulcerative colitis (UC), Crohn's disease (CD), or indeterminate colitis (IC). Corticosteroids are commonly used to treat active symptoms. Although, corticosteroids often promptly achieve remission, long-term use is complicated by many potential side effects. One treatment strategy for IBD patients is a "step-up" approach in which patients are prescribed medications with increasing potential toxicity based on the severity of their disease. Alternatively, a "top-down" approach uses aggressive therapy early in the disease course to prevent long-term complications. Both approaches have been shown to be clinically effective, however, there is treatment response heterogeneity and it is not clear which treatment is right for each individual patient. Clinical theory dictates that those likely to experience a more aggressive disease progression would benefit more from "top-down" than "step-up"; whereas those likely to experience a less aggressive progression might benefit more from "step-up."

The primary outcome is the disease activity score measured at the end of the second year, as measured by the Harvey—Bradshaw Index for subjects with CD and the Simple Clinical Colitis Index for subjects with UC. In both measures, higher scores reflect more disease activity. A high-quality treatment rule would reduce disease activity by assigning patients to top-down if it is necessary and step-up otherwise. Among the 274 patients included in the observed data, 32 patients were assigned to the top-down strategy ($A = 1$) and 242 were assigned to step-up ($A = -1$). To remain consistent with our paradigm of maximizing mean response we used the negative disease activity score as the response, Y. 11 patient covariates were used, which included age, gender, ethnicity, marital status, race, body mass index, disease type, antibiotics drug usage, antidiarrheal drug usage, indicator for extra-intestinal manifestation and baseline disease activity scores. We used a linear regression model to estimate the Q-function, and a regularized logistic regression model to estimate the propensity score to avoid overfitting. In addition to the EARL estimators we applied QL and OWL to estimate an optimal treatment rule. Because this is an observational study with unknown propensity scores, we evaluated the estimated treatment rules $\widehat{d}$ using inverse probability weighting $\widehat{V}^{\text{IPWE}}(\widehat{d}) = \mathbb{P}_n \left[Y I\left\{A = \widehat{d}(\boldsymbol{X})\right\} / \widehat{\pi}(A; \boldsymbol{X}) \right] / \mathbb{P}_n \left[I\left\{A = \widehat{d}(\boldsymbol{X})\right\} / \widehat{\pi}(A; \boldsymbol{X}) \right]$, where $\widehat{\pi}$ is the estimated propensity score. Higher values of $\widehat{V}^{\text{IPWE}}(\widehat{d})$, that is, lower disease activity scores, indicate a better overall benefit.

The coefficients of the estimated optimal treatment rules constructed from EARL with logistic loss are presented in Table 1. A permutation test based on 2000 permutation times was conducted to obtain the p-value for each covariate, which showed that body mass index was significant at 0.05 level and gender was significant at 0.1 level. In general, patients with a more severe disease status at baseline are likely to benefit from a top-down therapy.

This is consistent with clinical theory as these symptoms are associated with higher disease severity.

Table 2 describes the agreement between the estimated optimal decision rules constructed using different methods, which shows that the rules estimated using EARL with different loss functions give quite similar treatment recommendations. In this table, we also present the agreement between the estimated decision rules and the observed treatments. Compared to the observed treatment allocations, the estimated rules encourage more patients to receive top-down therapy, where 161 patients are recommended to top-down treatment by EARL methods with all loss functions, 225 patients are recommended by OWL method using logistic loss and 145 patients are recommended by QL method respectively. The estimated disease activity score is 1.75 using logistic loss, compared with 1.80 for the QL estimator, and 1.75 for OWL using logistic loss. Although the achieved benefit of the ITR yielded by OWL and EARL were similar, EARL recommended less patients to the more intensive top-down therapy, which could benefit patients by reducing the side effects. The achieved benefits of the derived ITRs were greater than the benefit that was achieved in the observed dataset, where the average disease activity score was 2.24. Since top-down therapy is relative new in the practice, to be conservative, physicians tend not to provide such therapy to patients. Our analysis encourages the usage of top-down therapy for a greater benefit, which can be tailored according to individual characteristics. By looking into the relationship between the observed treatment and covariates, we found that in current practice, physicians were more likely to follow top-down therapy while giving out antibiotics and antidiarrheals drugs in patients with Crohn's disease. The ITRs resulted from EARL, on the other hand, were more likely to recommend top-down therapy for ulcerative colitis/indeterminate colitis patients while they are not taking antibiotics and antidiarrheals drugs.

We also applied our method to the study of National Supported Work Demonstration, which also showed a superior performance of the proposed method. Results are shown in Web Appendix G.

6. Discussion

We proposed a class of estimators for the optimal treatment rule that we termed EARL. This class of methods is formed by applying a convex relaxation to the AIPWE of the marginal mean outcome. To reduce the risk of misspecification, it is possible to use flexible, e.g., nonparametric, models for the propensity score and the Q-function. However, we showed theoretically and empirically that such flexibility comes at the cost of additional variability and potentially poor small sample performance.

We demonstrated that extreme propensity scores may lead to a large variance in the augmented inverse probability weighted estimator. To alleviate this issue, weight stabilization could potentially help. In particular, we can consider a stabilized weight of the form

$$SW_a^m = W_a^m \Big/ \frac{I(A=a)}{\pi^m(a;\boldsymbol{X})}\,.$$

Using modified weight SW_a^m leads to consistent estimator of the optimal decision rule if positivity assumption holds, that is, Lemma 2.2 still holds under the modified weight.

Table 1: Coefficients for the estimated optimal decision rules by EARL with logistic loss (*: significant at 0.05 level).

	Coefficient	p-value
Intercept	2.466	-
Age	-0.001	0.905
Gender (Male = 1)	0.756	0.015*
Ethnicity (Hispanic = 1)	-1.045	0.144
Marital status (Single = 1)	-0.320	0.318
Race (White = 1)	-0.233	0.478
Body mass index	-0.063	0.037*
Disease type (UC or IC = 1)	0.309	0.234
Antibiotics drug usage (Yes = 1)	-0.156	0.563
Antidiarrheals drug usage (Yes = 1)	-0.580	0.167
Extra-intestinal manifestation (Yes = 1)	0.273	0.286
Baseline disease activity scores	0.050	0.427

Table 2: Agreements between the estimated optimal decision rule yield by different methods and the observed treatment. OWL-logit: OWL using logistic loss; EARL: EARL using logistic loss; EARL: EARL using exponential oss; EARL-hinge: EARL using hinge loss; EARL-sqhinge: EARL using squared hinge loss; QL: Q-learning.

	OWL-Logit	EARL-logit	EARL-exp	EARL-hinge	EARL-sqhinge	QL
OWL-Logit	1	0.642	0.821	0.639	0.639	0.577
EARL-logit		1	0.588	0.996	0.996	0.920
EARL-exp			1	0.591	0.591	0.529
EARL-hinge				1	1	0.916
EARL-sqhinge					1	0.916
QL						1
Observed	0.193	0.449	0.117	0.453	0.453	0.507

Alternatively, we may also consider an estimator which achieves the smallest variance among its class of doubly robust estimators when the propensity score model is correctly specified. Such an estimator can be derived following the techniques used in Cao et al. (2009).

There are several important ways this work might be extended. The first is to handle time-to-event outcomes wherein the observed data are subject to right-censoring. In this setting, efficient methods for augmentation to adjust for censoring might be folded into the EARL framework. Another extension is to multi-stage treatment rules, also known as, dynamic treatment regimes (Murphy, 2003; Robins, 2004; Moodie et al., 2007). A challenging component of this extension is that the variability of the AIPWE increases dramatically as the number of treatment stages increases. We believe that the convex relaxation may help in this setting not only in terms of computation but also by reducing variance.

7. Supplementary Materials

The Web Appendix referenced in Sections 3, 4 and 5 is available online.

Acknowledgements This work was partially supported by R01DK108073, P01CA142538, R01DE024984, P30CA015704, S10 OD020069, DMS-1555141, and grants from the Crohn's and Colitis Foundation and the Centers for Disease Control and Prevention (DP004785-05).

References

Susan Athey and Stefan Wager. Efficient policy learning. *arXiv preprint arXiv:1702.02896*, 2017.

Heejung Bang and James M Robins. Doubly robust estimation in missing data and causal inference models. *Biometrics*, 61(4):962–973, 2005.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *J. of American Statistical Association*, 101(473):138–156, 2006.

David Benkeser, Marco Carone, MJ Van Der Laan, and PB Gilbert. Doubly robust nonparametric inference on the average treatment effect. *Biometrika*, 104(4):863–880, 2017.

Peter J Bickel. On adaptive estimation. *The Annals of Statistics*, pages 647–671, 1982.

Weihua Cao, Anastasios A Tsiatis, and Marie Davidian. Improving efficiency and robustness of the doubly robust estimator for a population mean with incomplete data. *Biometrika*, 96(3):723–734, 2009.

Bibhas Chakraborty and Erica EM Moodie. *Statistical Methods for Dynamic Treatment Regimes*. Springer, 2013.

Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, and Whitney K Newey. Double machine learning for treatment and causal parameters. Technical report, cemmap working paper, Centre for Microdata Methods and Practice, 2016.

M. Davidian, A.A. Tsiatis, and E.B. Laber. Value search estimators. In *Dynamic Treatment Regimes*, pages 1–40. Springer, 2014.

Jianqing Fan, Kosuke Imai, Han Liu, Yang Ning, and Xiaolin Yang. Improving covariate balancing propensity score: A doubly robust and efficient approach. Technical report, 2016.

Yoav Freund and Robert E Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296, 1999.

T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer-Verlag New York, Inc., New York, second edition, 2009.

R. Henderson, P. Ansell, and D. Alshibani. Regret-Regression for Optimal Dynamic Treatment Regimes. *Biometrics*, 66(4), 2009.

Joseph DY Kang and Joseph L Schafer. Demystifying double robustness: A comparison of alternative strategies for estimating a population mean from incomplete data. *Statistical science*, pages 523–539, 2007.

Toru Kitagawa and Aleksey Tetenov. Who should be treated? empirical welfare maximization methods for treatment choice. 2017.

M. R. Kosorok. *Introduction to empirical processes and semiparametric inference*. Springer-Verlag, New York, 2008.

Eric B Laber and Susan A Murphy. Adaptive confidence intervals for the test error in classification. *Journal of the American Statistical Association*, 106(495):904–913, 2011.

Eric B Laber, Daniel J Lizotte, Min Qian, William E Pelham, Susan A Murphy, et al. Dynamic treatment regimes: Technical challenges and applications. *Electronic Journal of Statistics*, 8:1225–1272, 2014.

Kristin A Linn, Eric B Laber, and Leonard A Stefanski. Interactive q-learning for quantiles. *Journal of the American Statistical Association*, (just-accepted):1–37, 2016.

Ying Liu, Yuanjia Wang, Michael R Kosorok, Yingqi Zhao, and Donglin Zeng. Robust hybrid learning for estimating personalized dynamic treatment regimens. *arXiv preprint arXiv:1611.02314*, 2016.

Erica E. M. Moodie, Thomas S. Richardson, and David A. Stephens. Demystifying optimal dynamic treatment regimes. *Biometrics*, 63(2):447–455, 2007.

Erica EM Moodie, Bibhas Chakraborty, and Michael S Kramer. Q-learning for estimating optimal dynamic treatment rules from observational data. *Canadian Journal of Statistics*, 40(4):629–645, 2012.

Erica EM Moodie, Nema Dean, and Yue Ru Sun. Q-learning: Flexible learning about useful utilities. *Statistics in Biosciences*, pages 1–21, 2013.

S. A. Murphy. Optimal dynamic treatment regimes. *Journal of the Royal Statistical Society, Series B*, 65:331–366, 2003.

Whitney K Newey. Convergence rates and asymptotic normality for series estimators. *Journal of econometrics*, 79(1):147–168, 1997.

L. Orellana, A. Rotnitzky, and J. Robins. Dynamic regime marginal structural mean models for estimation of optimal dynamic treatment regimes, part i: Main content. *Int. Jrn. of Biostatistics*, 6(2):1–19, 2010.

Min Qian and S. A. Murphy. Performance guarantees for individualized treatment rules. *The Annals of Statistics*, 39:1180–1210, 2011.

James Robins. A new approach to causal inference in mortality studies with a sustained exposure periodapplication to control of the healthy worker survivor effect. *Mathematical Modelling*, 7:1393–1512, 1986.

James Robins. Causal inference from complex longitudinal data. *Lect. Notes Statist.*, 120: 69–117, 1997.

James M Robins. The analysis of randomized and non-randomized aids treatment trials using a new approach to causal inference in longitudinal studies. *Health service research methodology: a focus on AIDS*, 113:159, 1989.

James M. Robins. Optimal structural nested models for optimal sequential decisions. In *In Proceedings of the Second Seattle Symposium on Biostatistics*, pages 189–326. Springer, 2004.

James M Robins, Andrea Rotnitzky, and Lue Ping Zhao. Estimation of regression coefficients when some regressors are not always observed. *Journal of the American Statistical Association*, 89(427):846–866, 1994.

James M Robins, Lingling Li, Rajarshi Mukherjee, Eric Tchetgen Tchetgen, Aad van der Vaart, et al. Minimax estimation of a functional on a structured high-dimensional model. *The Annals of Statistics*, 45(5):1951–1987, 2017.

J.M. Robins, L. Orellana, and A. Rotnitzky. Estimation and extrapolation of optimal treatment and testing strategies. *Statistics in Medicine*, pages 4678–4721, 2008.

D. B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66:688–701, 1974.

Bruce E Sands, Neal LeLeiko, Samir A Shah, Renee Bright, and Stacey Grabert. Osccar: ocean state crohn's and colitis area registry. *Medicine and Health Rhode Island*, 92(3): 82, 2009.

Anton Schick. On asymptotically efficient estimation in semiparametric models. *The Annals of Statistics*, pages 1139–1151, 1986.

Phillip J. Schulte, Anastasios A. Tsiatis, Eric B. Laber, , and Marie Davidian. Q- and a-learning methods for estimating optimal dynamic treatment regimes. *Statistical Science*, 29:640–661, 2014.

Harold C Sox and Sheldon Greenfield. Comparative effectiveness research: a report from the institute of medicine. *Annals of Internal Medicine*, 151(3):203–205, 2009.

J. Splawa-Neyman, DM Dabrowska, and TP Speed. On the application of probability theory to agricultural experiments (engl. transl. by d.m. dabrowska and t.p. speed). *Statistical Science*, 5:465–472, 1990.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning I: Introduction*. MIT Press, Cambridge,MA, 1998.

Csaba Szepesvári. Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 4(1):1–103, 2010.

Jeremy MG Taylor, Wenting Cheng, and Jared C Foster. Reader reaction to ga robust method for estimating optimal treatment regimesh by zhang et al.(2012). *Biometrics*, 71 (1):267–273, 2015.

Baqun Zhang, Anastasios A Tsiatis, Marie Davidian, Min Zhang, and Eric Laber. Estimating optimal treatment regimes from a classification perspective. *Stat*, 1(1):103–114, 2012a.

Baqun Zhang, Anastasios A Tsiatis, Eric B Laber, and Marie Davidian. A robust method for estimating optimal treatment regimes. *Biometrics*, 68(4):1010–1018, 2012b.

Baqun Zhang, Anastasios A Tsiatis, Eric B Laber, and Marie Davidian. Robust estimation of optimal dynamic treatment regimes for sequential treatment decisions. *Biometrika*, 100:681–695, 2013.

Yichi Zhang, Eric B Laber, Anastasios Tsiatis, and Marie Davidian. Using decision lists to construct interpretable and parsimonious treatment regimes. *Biometrics*, 71(4):895–904, 2015.

Yichi Zhang, Eric B Laber, Marie Davidian, and Anastasios A Tsiatis. Estimation of optimal treatment regimes using lists. *Journal of the American Statistical Association*, (just-accepted), 2017.

Y. Q. Zhao, Donglin Zeng, A. John Rush, and Michael R. Kosorok. Estimating individualized treatment rules using outcome weighted learning. *Journal of American Statistical Association*, 107:1106–1118, 2012.

Yufan Zhao, Michael R. Kosorok, and Donglin Zeng. Reinforcement learning design for cancer clinical trials. *Statistics in Medicine*, 28:3294–3315, 2009.

Wenjing Zheng and Mark J van der Laan. Cross-validated targeted minimum-loss-based estimation. In *Targeted Learning*, pages 459–474. Springer, 2011.

Figure 1: Boxplots for Scenario 1 results under QL, AIPWE, and OWL and EARL using logistic loss.

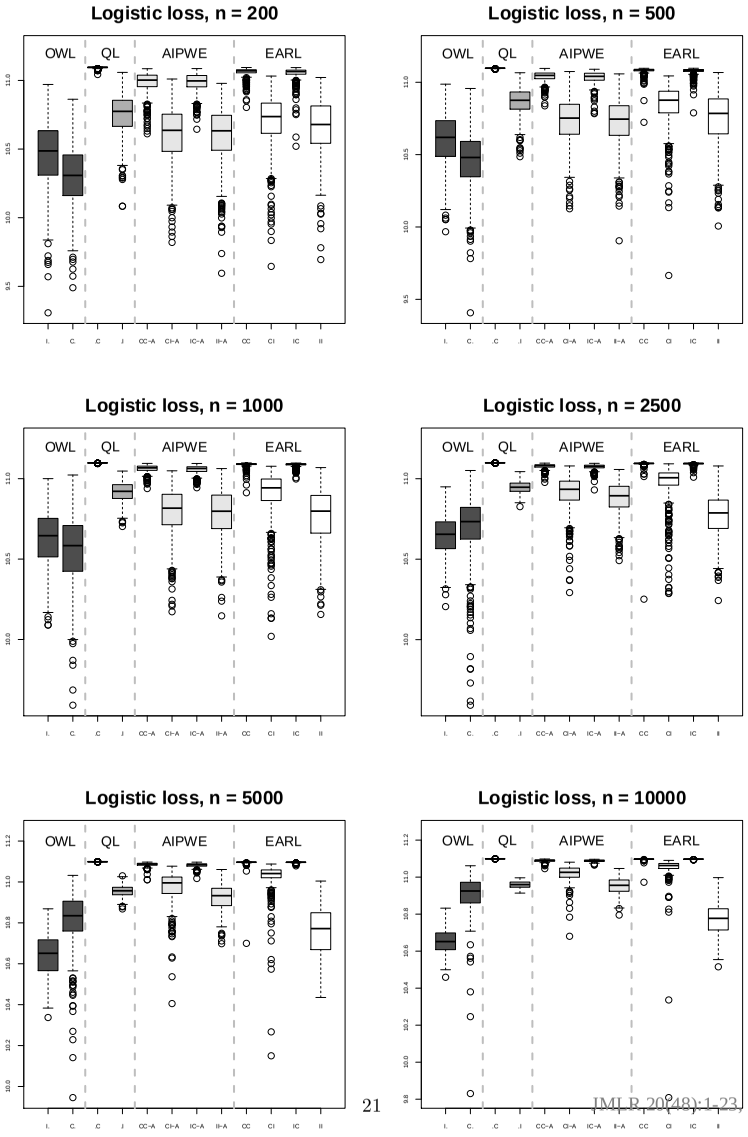


Figure 2: Boxplots for Scenario 2 results under QL, AIPWE, and OWL and EARL using logistic loss.

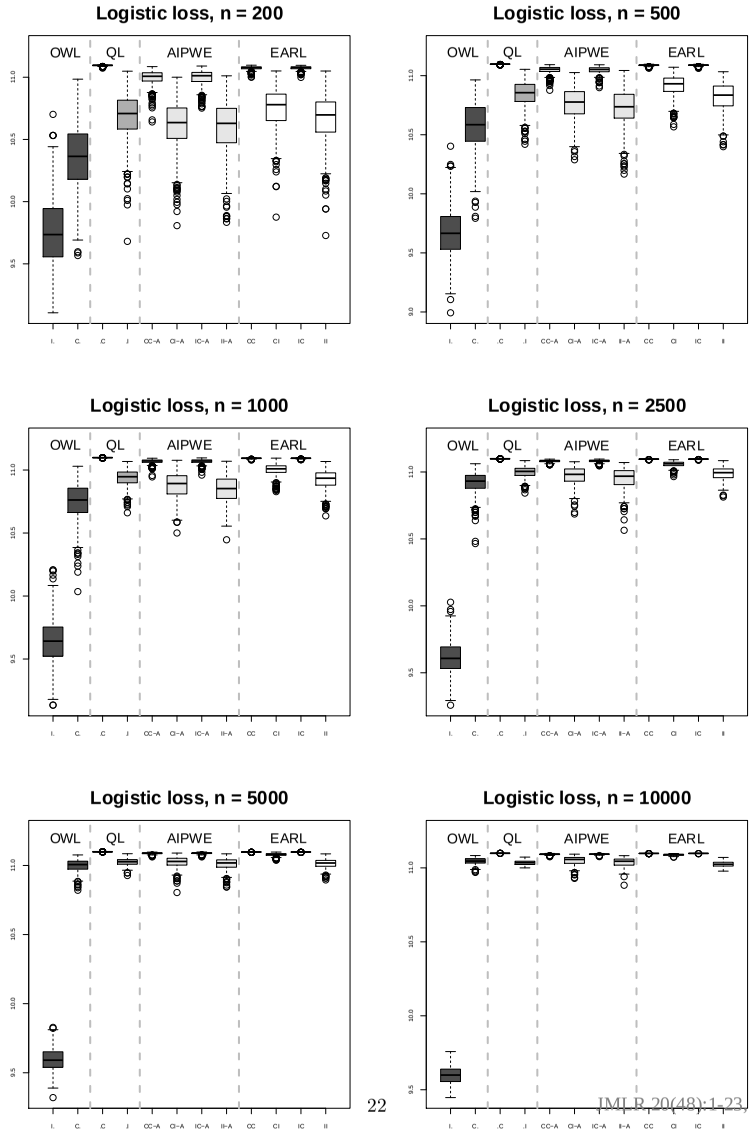
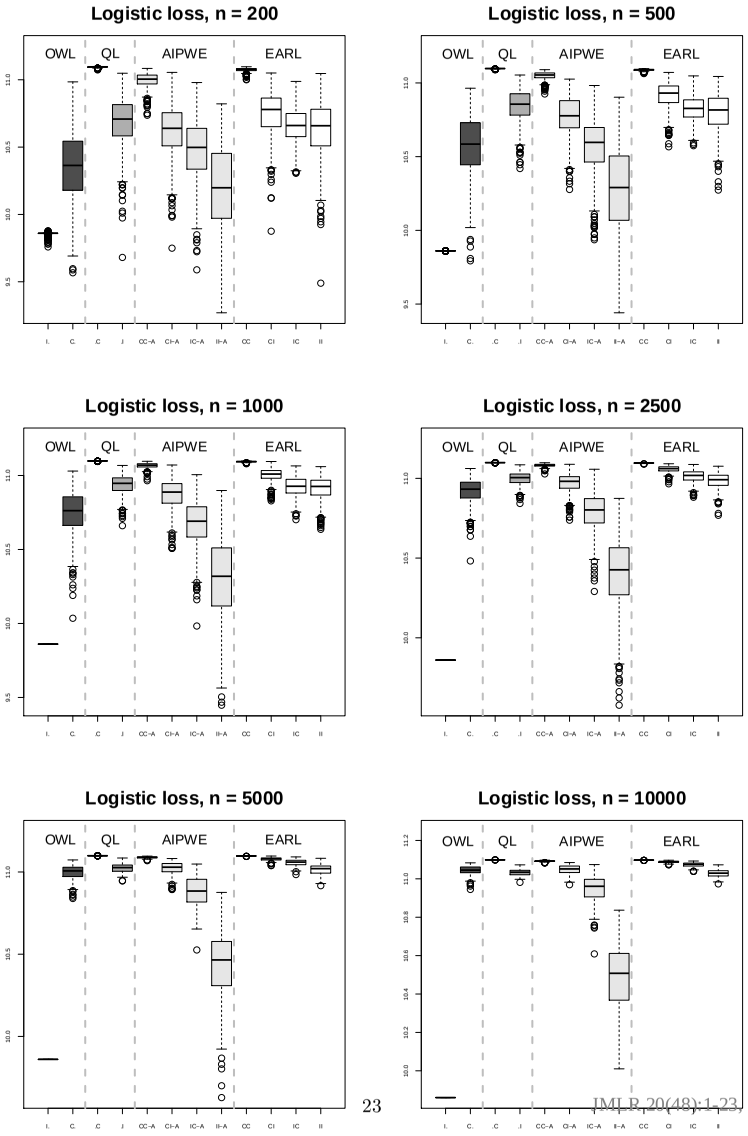


Figure 3: Boxplots for Scenario 3 results under QL, AIPWE, and OWL and EARL using logistic loss.



JMLR 20(48):1-23, 2019

Journal of Machine Learning Research 20 (2019) 1-24 Submitted 4/18; Revised 12/18; Published 2/19

Using Simulation to Improve Sample-Efficiency of Bayesian Optimization for Bipedal Robots

Akshara Rai* ARAI@CS.CMU.EDU
Robotics Institute, School of Computer Science
Carnegie Mellon University, PA, USA

Rika Antonova* ANTONOVA@KTH.SE
Robotics, Perception and Learning, CSC
KTH Royal Institute of Technology, Stockholm, Sweden

Franziska Meier FRANZI.MEIER@GMAIL.COM
Paul G. Allen School of Computer Science & Engineering
University of Washington, Seattle, WA, USA

Christopher G. Atkeson CGA@CS.CMU.EDU
Robotics Institute, School of Computer Science
Carnegie Mellon University, PA, USA

*These authors contributed equally.

Editor: Bayesian Optimization Special Issue

Abstract

Learning for control can acquire controllers for novel robotic tasks, paving the path for autonomous agents. Such controllers can be expert-designed policies, which typically require tuning of parameters for each task scenario. In this context, Bayesian optimization (BO) has emerged as a promising approach for automatically tuning controllers. However, sample-efficiency can still be an issue for high-dimensional policies on hardware. Here, we develop an approach that utilizes simulation to learn structured feature transforms that map the original parameter space into a domain-informed space. During BO, similarity between controllers is now calculated in this transformed space. Experiments on the ATRIAS robot hardware and simulation show that our approach succeeds at sample-efficiently learning controllers for multiple robots. Another question arises: What if the simulation significantly differs from hardware? To answer this, we create increasingly approximate simulators and study the effect of increasing simulation-hardware mismatch on the performance of Bayesian optimization. We also compare our approach to other approaches from literature, and find it to be more reliable, especially in cases of high mismatch. Our experiments show that our approach succeeds across different controller types, bipedal robot models and simulator fidelity levels, making it applicable to a wide range of bipedal locomotion problems.

Keywords: Bayesian Optimization, Bipedal Locomotion, Transfer Learning

1. Introduction

Machine learning can provide methods for learning controllers for robotic tasks. Yet, even with recent advances in this field, the problem of automatically designing and learning controllers for robots, especially bipedal robots, remains difficult. It is expensive to do learning experiments that require a large number of samples with physical robots. Specifically, legged robots are not robust to falls and failures, and are time-consuming to work with and repair. Furthermore, commonly used cost functions for optimizing controllers are noisy to evaluate, non-convex and non-differentiable. In order to find learning approaches that can be used on real robots, it is thus important to keep these considerations in mind.

Deep reinforcement learning approaches can deal with noise, discontinuities and non-convexity of the objective, but they are not data-efficient. These approaches could take on the order of a million samples to learn locomotion controllers (Peng et al., 2016), which would be infeasible on a real robot. For example, on the ATRIAS robot, 10,000 samples would take 7 days, in theory. But practically, the robot needs to be "reset" between trials and repaired in case of damage. Using structured expert-designed policies can help minimize damage to the robot and make the search for successful controllers feasible. However, the problem is black-box, non-convex and discontinuous. This eliminates approaches like PI^2 (Theodorou et al., 2010) which make assumptions about the dynamics of the system and PILCO (Deisenroth and Rasmussen, 2011) which assumes a continuous cost landscape. Evolutionary approaches like CMA-ES (Hansen, 2006) can still be prohibitively expensive, needing thousands of samples (Song and Geyer, 2015).

In comparison, Bayesian optimization (BO) is a sample-efficient optimization technique that is robust to non-convexity and noise. It has been recently used in a range of robotics problems, such as Calandra et al. (2016b), Marco et al. (2017) and Cully et al. (2015). However, sample-efficiency of conventional BO degrades in high dimensions, even for dimensionalities commonly encountered in locomotion controllers. Because of this, hardware-only optimization becomes intractable for flexible controllers and complex robots. One way of addressing this issue is to utilize simulation to optimize controller parameters. However, simulation-only optimization is vulnerable to learning policies that exploit the simulation and, because of that, perform well in simulation but poorly on the actual robot. This motivates the development of hybrid approaches that can incorporate simulation-based information into the learning method and then optimize with few samples on hardware.

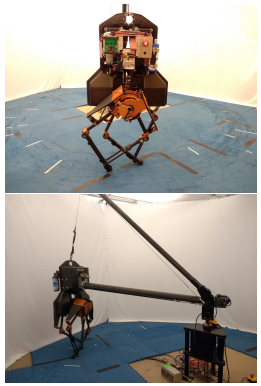


Figure 1: ATRIAS robot.

Towards this goal, our previous work in Antonova, Rai, and Atkeson (2016), Antonova et al. (2017), Rai et al. (2018) presents a framework that uses information from high-fidelity simulators to learn sample-efficiently on hardware. We use simulation to build informed feature transforms that are used to measure controller similarity during BO. Thus, during optimization on hardware, the similarity between controller parameters is informed by how they perform in simulation. With this, it becomes possible to quickly infer which regions of the input space are likely to perform well on hardware. This method has been tested on the ATRIAS biped robot (Figure 1) and shows considerable improvement in sample-efficiency over traditional BO.

In this article, we present in-depth explanations and empirical analysis of our previous work. Furthermore, for the first time, we present a procedure for systematically evaluating

©2019 Akshara Rai, Rika Antonova, Franziska Meier and Christopher G. Atkeson.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-196.html.

robustness of such approaches to simulation-hardware mismatch. We extend our previous work incorporating mismatch estimates (Rai et al., 2018) to this setting. We also conduct extensive comparisons with competitive baselines from prior work, e.g. Cully et al. (2015).

The rest of this article is organized as follows: Section 2 provides background for BO, then gives an overview of related work on optimizing locomotion controllers. Section 3.1 describes the idea of incorporating simulation-based transforms into BO; Section 3.2 explains how we handle simulation-hardware mismatch. Sections 4.1-4.5 describe the robot and controllers we use for our experiments; Section 4.6 explains the motivation and construction of simulators with various kinds of simulation-hardware mismatch. Section 5 summarizes hardware experiments on the ATRIAS robot. Section 5.2 shows generalization to a different robot model in simulation. Section 5.3 presents empirical analysis of the impact of simulator quality on the performance of the proposed algorithms and alternative approaches.

2. Background and Related Work

This section gives a brief overview of Bayesian optimization (BO), the state-of-the-art research on optimizing locomotion controllers, and utilizing simulation information in BO.

2.1. Background on Bayesian Optimization

Bayesian optimization (BO) is a framework for online, black-box, gradient-free global search (Shahriari et al. (2016) and Brochu et al. (2010) provide a comprehensive introduction). The problem of optimizing controllers can be interpreted as finding controller parameters $\boldsymbol{x}^*$ that optimize some cost function $f(\boldsymbol{x})$. Here $\boldsymbol{x}$ contains parameters of a pre-structured policy; the cost $f(\boldsymbol{x})$ is a function of the trajectory induced by controller parameters $\boldsymbol{x}$. For brevity, we will refer to 'controller parameters $\boldsymbol{x}$' as 'controller $\boldsymbol{x}$'. We use BO to find controller $\boldsymbol{x}^*$, such that: $f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}} f(\boldsymbol{x})$.

BO is initialized with a prior that expresses the *a priori* uncertainty over the value of $f(\boldsymbol{x})$ for each $\boldsymbol{x}$ in the domain. Then, at each step of optimization, based on data seen so far, BO optimizes an auxiliary function (called *acquisition function*) to select the next $\boldsymbol{x}$ to evaluate. The acquisition function balances exploration vs exploitation. It selects points for which the posterior estimate of the objective f is promising, taking into account both mean and covariance of the posterior. A widely used representation for the cost function f is a Gaussian process (GP):

$$f(\boldsymbol{x}) \sim \mathcal{GP}(\mu(\boldsymbol{x}), k(\boldsymbol{x}_i, \boldsymbol{x}_j))$$

The prior mean function $\mu(\cdot)$ is set to 0 when no domain-specific knowledge is provided. The kernel function $k(\cdot,\cdot)$ encodes similarity between inputs. If $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is large for inputs $\boldsymbol{x}_i, \boldsymbol{x}_j$, then $f(\boldsymbol{x}_i)$ strongly influences $f(\boldsymbol{x}_j)$. One of the most widely used kernel functions is the Squared Exponential (SE):

$$k_{SE}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_k^2 \exp\big(-\tfrac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \operatorname{diag}(\boldsymbol{\ell})^{-2}(\boldsymbol{x}_i - \boldsymbol{x}_j)\big),$$

where σ_k^2, $\boldsymbol{\ell}$ are signal variance and a vector of length scales respectively. σ_k^2, $\boldsymbol{\ell}$ are referred to as 'hyperparameters' in the literature.

The SE kernel belongs to a broader class of Matérn kernels, which in general have more free parameters. In some cases, carefully choosing kernel parameters improves performance of BO (Snoek et al., 2012). However, domain-informed kernels can easily out-perform even

well-tuned Matérn kernels (Cully et al., 2015). SE and Matérn kernels are stationary: $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ depend only on $r = \boldsymbol{x}_i - \boldsymbol{x}_j \ \forall \boldsymbol{x}_{i,j}$. We aim to remove this limitation in a manner informed by simulation. In future work, incorporating other approaches that relax stationarity (e.g. Martinez-Cantin (2018)) could further improve BO.

2.2. Optimizing Locomotion Controllers

Parametric locomotion controllers can be represented as $\boldsymbol{u} = \pi_{\boldsymbol{x}}(\boldsymbol{s})$, where π is a policy structure that depends on parameters $\boldsymbol{x}$. For example, π can be parameterized by feedback gains on the center of mass (CoM), reference joint trajectories, etc. Vector $\boldsymbol{s}$ is the state of the robot, such as joint angles and velocities, used in closed-loop controllers. Vector $\boldsymbol{u}$ represents the desired control action, for example: torques, angular velocities or positions for each joint on the robot. The sequence of control actions yields a sequence of state transitions, which form the overall 'trajectory' $[\boldsymbol{s}_0, \boldsymbol{u}_1, \boldsymbol{s}_1, \boldsymbol{u}_2, \boldsymbol{s}_2, ...]$. This trajectory is used in the cost function to judge the quality of the controller $\boldsymbol{x}$. In our work, we use structured controllers designed by experts. State of the art research on walking robots featuring such controllers includes Feng et al. (2015) and Kuindersma et al. (2016). The overall optimization then includes manually tuning the parameters $\boldsymbol{x}$. An alternative to manual tuning is to use evolutionary approaches, like CMA-ES, as in Song and Geyer (2015). However, these require a large number of samples and can usually be conducted only in simulation. Optimization in simulation can produce controllers that perform well in simulation, but not on hardware. In comparison, BO is a sample-efficient technique which has become popular for direct optimization on hardware. Recent successes include manipulation (Englert and Toussaint, 2016) and locomotion (Calandra et al., 2016b).

BO for locomotion has been previously explored for several types of mobile robots. These include: snake robots (Tesch et al., 2011), AIBO quadrupeds (Lizotte et al., 2007), and hexapods (Cully et al., 2015). Tesch et al. (2011) optimize a 3-dimensional controller for a snake robot in 10-40 trials (for speeds up to $0.13m/s$). Lizotte et al. (2007) use BO to optimize gait parameters for a AIBO robot in 100-150 trials. Cully et al. (2015) learn 36 controller parameters for a hexapod. Even with hardware damage, they can obtain successful controllers for speeds up to $0.4m/s$ in 12-15 trials.

Hexapods, quadrupeds and snakes spend a large portion of their gaits with their center of mass within their support polygon (convex hull of the feet on the ground). Ignoring velocity, this is statically stable. In contrast, bipedal walking can be highly dynamic, especially for point-feet robots like ATRIAS. ATRIAS, like most bipeds, spends a significant time of its gait being "unstable", or dynamically stable. In our experiments on hardware, ATRIAS goes up to speeds of $1m/s$. All of this leads to a challenging optimization setting and discontinuous cost function landscape. Calandra et al. (2016b) use BO for optimizing gaits of a dynamic biped on a boom, needing 30-40 samples for finding walking gaits for a 4-dimensional controller. While this is promising, optimizing a higher-dimensional controller needed for complex robots would be even more challenging. If significant number of samples lead to unstable gaits and falls, they could damage the robot. Hence, it is important to develop methods that can learn complex controllers fast, without damaging the robot.

2.3. Incorporating Simulation Information into Bayesian Optimization

The idea of using simulation to speed up BO on hardware has been explored before. Information from simulation can be added as a prior to the GP used in BO, such as in Cully et al. (2015). While such methods can be successful, one needs to carefully tune the influence of simulation points over hardware points, especially when simulation is significantly different from hardware.

More generally, approaches such as Kandasamy et al. (2017) and Marco et al. (2017) address trading off computation vs simulation accuracy when selecting the source for the next trial/evaluation (with real world viewed as the most expensive source). Instead, we consider a setting with ample compute resources for simulation, but an extremely small number of experiments on a real robot. This is appropriate for bipedal locomotion with full-scale robots, since these can be expensive to run. Hence, our work does not fall into the 'multi-fidelity' BO paradigm, and we instead take a two step approach: pre-computing information needed for kernel transforms in the first stage, then running an ultra-sample efficient version of BO in the second stage on a real robot.

Recently, several approaches proposed incorporating Neural Networks (NNs) into the Gaussian process (GP) kernels (Wilson et al. (2016), Calandra et al. (2016a)). The strength of these approaches is that they can jointly update the GP and the NN. Calandra et al. (2016a) demonstrated how this added flexibility can handle discontinuities in the cost function landscape. However, these approaches do not directly address the problem of incorporating a large amount of data from simulation in hardware BO experiments.

Wilson et al. (2014) explored enhancing GP kernel with trajectories. Their Behavior Based Kernel (BBK) computes an estimate of a symmetric variant of the KL divergence between trajectories induced by two controllers, and uses this as a distance metric in the kernel. However, getting an estimate would require samples for each controller $\boldsymbol{x}_i, \boldsymbol{x}_j$ whenever $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is needed. This can be impractical, as it involves an evaluation of every controller considered. To overcome this, the authors suggest combining BBK with a model-based approach. But building an accurate model from hardware data might be an expensive process in itself.

Cully et al. (2015) utilize simulation by defining a behavior metric and collecting best performing points in simulation. This behavior metric then guides BO to quickly find controllers on hardware, and can even compensate for damage to the robot. The search on hardware is conducted in behavior space, and limited to pre-selected "successful" points from simulation. This helps make their search faster and safer on hardware. However, if an optimal point was not pre-selected, BO cannot sample it during optimization.

In our work, we utilize trajectories from simulation to build feature transforms that can be incorporated in the GP kernel used for BO. Our approaches incorporate trajectory/behavior information, but ensure that $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is computed efficiently during BO. Our work is related, in part, to input space warping, as described in Snoek et al. (2014), but we construct our transforms from simulated data. Our simulation-informed kernels bias the search towards regions that look promising, but are able to 'recover' and search in other parts of the space if simulation-hardware mismatch becomes apparent.

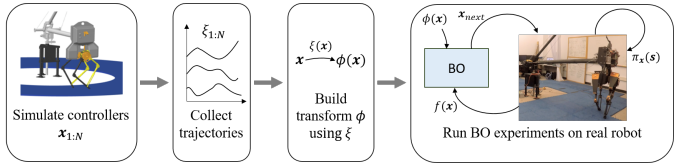


Figure 2: Overview of our proposed approach. Here, $\pi_{\boldsymbol{x}}(\boldsymbol{s})$ is the policy (Section 2.2); $\boldsymbol{x}$ is a vector of controller parameters; $\boldsymbol{s}$ is the state of the robot; $\xi(\boldsymbol{x})$ is a trajectory observed in simulation for $\boldsymbol{x}$; $\phi(\cdot)$ is the transform built using $\xi(\cdot)$; $f(\boldsymbol{x})$ is the cost of $\boldsymbol{x}$ evaluated on hardware. BO uses $\phi(\boldsymbol{x})$ and evaluated costs $f(\boldsymbol{x})$ to propose next promising controller x_{next}.

3. Proposed Approach: Bayesian Optimization with Informed Kernels

In this section, we offer in-depth explanation of approaches from our work in Antonova, Rai, and Atkeson (2016), Antonova et al. (2017), and Rai et al. (2018). This work proposes incorporating domain knowledge into BO with the help of simulation. We evaluate locomotion controllers in simulation, and collect their induced trajectories, which are then used to build an informed transform. This can be achieved by using a domain-specific feature transform (Section 3.1.1) or by learning to reconstruct short trajectory summaries (Section 3.1.2). This feature transform is used to construct an informed distance metric for BO, and helps BO discover promising regions faster. Figure 2 gives an overview. In Section 3.2 we discuss how to incorporate simulation-hardware mismatch in to the transform, ensuring that BO can benefit from inaccurate simulations as well.

3.1. Constructing Flexible Kernels using Simulation-based Transforms

High dimensional problems with discontinuous cost functions are very common with legged robots, where slight changes to some parameters can make the robot unstable. Both of these factors can adversely affect BO's performance, but informed feature transforms can help BO sample high-performing controllers even in such scenarios.

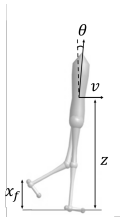
In this section, we demonstrate how to construct such transforms $\phi(\boldsymbol{x})$ utilizing simulations for a given controller $\boldsymbol{x}$. We then use ϕ to create an informed kernel $k_\phi(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for BO on hardware:

$$\begin{aligned} \boldsymbol{t}_{ij} &= \phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j) \\ k_\phi(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \sigma_k^2 \exp\Big(-\tfrac{1}{2} \boldsymbol{t}_{ij}^T \operatorname{diag}(\boldsymbol{\ell})^{-2} \boldsymbol{t}_{ij} \Big) \end{aligned} \tag{1}$$

Note that the functional form above is the same as that of Squared Exponential kernel, if considered from the point of view of the transformed space, with $\phi(\boldsymbol{x})$ as input. Using a low-dimensional ϕ could improve sample efficiency by reducing dimensionality. More notably, crucial gains arise when ϕ brings controllers that perform similar in simulation closer together, as compared to the original parameter space. For locomotion, this could bring failing controllers close together to occupy only a small portion of the transformed space, as illustrated in Rai et al. (2018). In essence, this means that the resultant kernel, though stationary in ϕ, is non-stationary in $\boldsymbol{x}$.

3.1.1. The Determinants of Gait Transform

We propose a feature transform for bipedal locomotion derived from physiological features of human walking called Determinants of Gaits (DoG) (Inman et al., 1953). ϕ_{DoG} was originally developed for human-like robots and controllers (Antonova, Rai, and Atkeson, 2016), and then generalized to be applicable to a wider range of bipedal locomotion controllers and robot morphologies (Rai et al., 2018). It is based on the features in Table 1.



M_1 (Swing leg retraction) – If the maximum ground clearance of the swing foot x_f is more than a threshold, $M_1 = 1$ (0 otherwise); ensures swing leg retraction.
M_2 (Center of mass height) – If CoM height z stays about the same at the start and end of a step, $M_2 = 1$ (0 otherwise); checks that the robot is not falling.
M_3 (Trunk lean) – If the average trunk lean θ is the same at the start and end of a step, $M_3 = 1$ (0 otherwise); ensures that the trunk is not changing orientation.
M_4 (Average walking speed) – Average forward speed v of a controller per step, $M_4 = v_{avg}$; M_4 helps distinguish controllers that perform similar on M_{1-3}.

Table 1: Illustration of the features used to construct DoG transform.

ϕ_{DoG} combines features M_{1-4} per step and scales them by the normalized simulation time to obtain the DoG score of controller $\boldsymbol{x}$:

$$score_{DoG} = \frac{t_{sim}}{t_{max}} \cdot \sum_{s=1}^{N} \sum_{k=1}^{4} M_{ks} \tag{2}$$

Here, subscript k identifies the feature metric, s is the step number, N is the total number of steps taken in simulation, t_{sim} is time at which simulation terminated (possibly due to a fall), t_{max} is total time allotted for simulation. Since larger number of steps lead to higher DoG, some controllers that chatter (step very fast before falling) could get misleadingly high scores; we scale the scores by $\frac{t_{sim}}{t_{max}}$ to prevent that. $\phi_{DoG}(\boldsymbol{x})$ for controller parameters $\boldsymbol{x}$ now becomes the computed $score_{DoG}$ of the resulting trajectories when $\boldsymbol{x}$ is simulated. ϕ_{DoG} essentially aids in (soft) clustering of controllers based on their behaviour in simulation. High scoring controllers are more likely to walk than low scoring ones. Since M_{1-4} are based on intuitive gait features, they are more likely to transfer between simulation and hardware, as compared to direct cost. The thresholds in M_{1-3} are chosen according to values observed in nominal human walking from Winter and Yack (1987).

3.1.2. Learning a Feature Transform with a Neural Network

While domain-specific feature transforms can be extremely useful and robust, they might be difficult to generate when a domain expert is not present. This motivates directly learning such feature transforms from trajectory data. In this section we describe our approach to train neural networks to reconstruct trajectory summaries (Antonova et al., 2017) that achieves this goal of minimizing expert involvement.

Trajectory summaries are a convenient choice for reparametrizing controllers into an easy to optimize space. For example, controllers that fall would automatically be far away from controllers that walk. If these trajectories can be extracted from a high-fidelity simulator, we would not have to evaluate each controller on hardware. However, conventional implementations of BO evaluate the kernel function for a large number of points per iteration, requiring thousands of simulations each iteration. To avoid this, a Neural Network (NN) can be trained to reconstruct trajectory summaries from a large set of pre-sampled data points. NN provides flexible interpolation, as well as fast evaluation (controller $\rightarrow$ trajectory summary). Furthermore, trajectories are agnostic to the specific cost used during BO. Thus the data collection can be done offline, and there is no need to re-run simulations in case the definition of the cost is modified.

We use the term 'trajectory' in a general sense, referring to several sensory states recorded during a simulation. To create trajectory summaries for the case of locomotion, we include measurements of: walking time (time before falling), energy used during walking, position of the center of mass and angle of the torso. With this, we construct a dataset for NN to fit: a Sobol grid of controller parameters ($\boldsymbol{x}_{1:N}$, $N \approx 0.5$ million) along with trajectory summaries $\xi_{\boldsymbol{x}_i}$ from simulation. NN is trained using mean squared loss:

NN input: $\boldsymbol{x}$ – a set of controller parameters

NN output: $\phi_{trajNN}(\boldsymbol{x}) = \hat{\xi}_{\boldsymbol{x}}$ – reconstructed trajectory summary

NN loss: $\frac{1}{2}\sum_{i=1}^{N} ||\hat{\xi}_{\boldsymbol{x}_i} - \xi_{\boldsymbol{x}_i}||^2$

The outputs $\phi_{trajNN}(\boldsymbol{x})$ are then used in the kernel for BO:

$$k_{trajNN}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_k^2 \exp\left(-\frac{1}{2}\boldsymbol{t}_{ij}^T \operatorname{diag}(\boldsymbol{\ell})^{-2}\boldsymbol{t}_{ij}\right), \qquad \boldsymbol{t}_{ij} = \phi_{trajNN}(\boldsymbol{x}_i) - \phi_{trajNN}(\boldsymbol{x}_j) \tag{3}$$

Appendix A describes our data collection and training. We did not carefully select the sensory traces used in the trajectory summaries. Instead, we used the most obvious states, aiming for an approach that could be easily adapted to other domains. To apply this approach to a new setting, one could simply include information that is customarily tracked, or used in costs. For example, for a manipulator, the coordinates of the end effector(s) could be recorded at relevant points. Force-torque measurements could be included, if available.

3.2. Kernel Adjustment for Handling Simulation-Hardware Mismatch

Approaches described in previous sections could provide improvement for BO when a high-fidelity simulator is used in kernel construction. In Rai et al. (2018) we presented promising results of experimental evaluation on hardware. However, it is unclear how the performance changes when simulation-hardware mismatch becomes apparent.

In Rai et al. (2018), we also proposed a way to incorporate information about simulation-hardware mismatch into the kernel from the samples evaluated so far. We augment the simulation-based kernel to include this additional information about mismatch, by expanding the original kernel by an extra dimension that contains the predicted mismatch for each controller $\boldsymbol{x}$.

A separate Gaussian process is used to model the mismatch experienced on hardware, starting from an initial prior mismatch of 0: $g(\boldsymbol{x}) \sim \mathcal{GP}(0, k_{SE}(\boldsymbol{x}_i, \boldsymbol{x}_j))$. For any evaluated controller $\boldsymbol{x}_i$, we can compute the difference between $\phi(\boldsymbol{x}_i)$ in simulation and on hardware: $d_{\boldsymbol{x}_i} = \phi^{sim}(\boldsymbol{x}_i) - \phi^{hw}(\boldsymbol{x}_i)$. We can now use mismatch data $\{d_{\boldsymbol{x}_i} | i = 1...n\}$ to construct a model for the expected mismatch: $\bar{g}(\boldsymbol{x})$. In the case of using a GP-based model, $\bar{g}(\cdot)$ would denote the posterior mean. With this, we can predict simulation-hardware mismatch in the original space of controller parameters for unevaluated controllers. Combining this with

kernel k_ϕ we obtain an adjusted kernel:

$$\boldsymbol{\phi}_{\boldsymbol{x}}^{adj} = \begin{bmatrix} \phi^{sim}(\boldsymbol{x}) \\ \bar{g}(\boldsymbol{x}) \end{bmatrix}, \qquad \boldsymbol{t}_{ij}^{adj} = \boldsymbol{\phi}_{\boldsymbol{x}_i}^{adj} - \boldsymbol{\phi}_{\boldsymbol{x}_j}^{adj}$$
$$k_{\phi_{adj}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_k^2 \exp\Big(-\tfrac{1}{2} (\boldsymbol{t}_{ij}^{adj})^T \operatorname{diag}\Big(\begin{bmatrix} \boldsymbol{\ell_1} \\ \boldsymbol{\ell_2} \end{bmatrix}\Big)^{-2} \boldsymbol{t}_{ij}^{adj} \Big) \tag{4}$$

The similarity between points $\boldsymbol{x}_i, \boldsymbol{x}_j$ is now dictated by two components: representation in ϕ space and expected mismatch. This construction has an intuitive explanation: Suppose controller $\boldsymbol{x_i}$ results in walking when simulated, but falls during hardware evaluation. $k_{\phi_{adj}}$ would register a high mismatch for $\boldsymbol{x}_i$. Controllers would be deemed similar to $\boldsymbol{x}_i$ only if they have both similar simulation-based $\phi(\cdot)$ and similar estimated mismatch. Points with similar simulation-based $\phi(\cdot)$ and low predicted mismatch would still be 'far away' from the failed $\boldsymbol{x}_i$. This would help BO sample points that still have high chances of walking in simulation, but are in a different region of the original parameter space. In the next section, we present a more mathematically rigorous interpretation for $k_{\phi_{adj}}$.

3.2.1. Interpretation of Kernel with Mismatch Modeling

Let us consider a controller $\boldsymbol{x}_i$ evaluated on hardware. The difference between simulation-based and hardware-based feature transform for $\boldsymbol{x}_i$ is $d_{\boldsymbol{x}_i} = \phi^{sim}(\boldsymbol{x}_i) - \phi^{hw}(\boldsymbol{x}_i)$. The 'true' hardware feature transform for $\boldsymbol{x_i}$ is $\phi^{hw}(\boldsymbol{x_i}) = \phi^{sim}(\boldsymbol{x_i}) - d_{\boldsymbol{x_i}}$. After n evaluations on hardware, $\{d_{\boldsymbol{x}_i} | i = 1...n\}$ can serve as data for modeling simulation-hardware mismatch. In principle, any data-efficient model $g(\cdot)$ can be used, such as GP (a multi-output GP in case $\phi(\cdot) > 1D$). With this, we can obtain an adjusted transform: $\hat{\phi}^{hw}(\boldsymbol{x}) = \phi^{sim}(\boldsymbol{x}) - \bar{g}(\boldsymbol{x})$, where $\bar{g}(\cdot)$ is the output of the model fitted using $\{d_{\boldsymbol{x}_i} | i = 1, ...n\}$.

Suppose $\boldsymbol{x}_{new}$ has not been evaluated on hardware. We can use $\hat{\phi}^{hw}(\boldsymbol{x}_{new}) = \phi^{sim}(\boldsymbol{x}_{new}) - \bar{g}(\boldsymbol{x}_{new})$ as the adjusted estimate of what the output of ϕ should be, taking into account what we have learned so far about simulation-hardware mismatch.

Let's construct kernel $k_{\phi_{adj}}^{v_2}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ that uses these hardware-adjusted estimates directly:

$$\begin{aligned} \boldsymbol{q}_{ij}^{adj} &= \hat{\phi}^{hw}(\boldsymbol{x}_i) - \hat{\phi}^{hw}(\boldsymbol{x}_j) \\ &= (\phi^{sim}(\boldsymbol{x}_i) - \bar{g}(\boldsymbol{x}_i)) - (\phi^{sim}(\boldsymbol{x}_j) - \bar{g}(\boldsymbol{x}_j)) \\ &= \underbrace{(\phi^{sim}(\boldsymbol{x}_i) - \phi^{sim}(\boldsymbol{x}_j))}_{\boldsymbol{v}_\phi} - \underbrace{(\bar{g}(\boldsymbol{x}_i) - \bar{g}(\boldsymbol{x}_j))}_{\boldsymbol{v}_g} \end{aligned}$$

$$\begin{aligned} k_{\phi_{adj}}^{v_2}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \sigma_{k_{v_0}}^2 \exp\Big(-\tfrac{1}{2} (\boldsymbol{q}_{ij}^{adj})^T \operatorname{diag}(\boldsymbol{\ell})^{-2} \boldsymbol{q}_{ij}^{adj} \Big) \\ &= \sigma^2 \exp\Big(-\tfrac{1}{2} \big[(\boldsymbol{v}_\phi - \boldsymbol{v}_g)^T \operatorname{diag}(\boldsymbol{\ell})^{-2} (\boldsymbol{v}_\phi - \boldsymbol{v}_g) \big] \Big) \\ &= \sigma^2 \exp\Big(-\tfrac{1}{2} \big[\boldsymbol{v}_\phi^T \operatorname{diag}(\boldsymbol{\ell})^{-2} \boldsymbol{v}_\phi + \boldsymbol{v}_g^T \operatorname{diag}(\boldsymbol{\ell})^{-2} \boldsymbol{v}_g - prod_{ij} \big] \Big) \\ &\qquad \text{where } prod_{ij} = 2 \boldsymbol{v}_\phi^T \operatorname{diag}(\boldsymbol{\ell})^{-2} \boldsymbol{v}_g \end{aligned}$$

Using $\exp(a + b + c) = \exp(c) \cdot \exp(a + b)$, we have:

$$k_{\phi_{adj}}^{v_2}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \exp(prod_{ij}) \exp\Big(-\tfrac{1}{2} \big[\boldsymbol{v}_\phi^T \operatorname{diag}(\boldsymbol{\ell})^{-2} \boldsymbol{v}_\phi + \boldsymbol{v}_g^T \operatorname{diag}(\boldsymbol{\ell})^{-2} \boldsymbol{v}_g \big] \Big)$$

Which can be written as:

$$k_{\phi_{adj}}^{v_2}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma^2 \exp(prod_{ij}) \exp\Big(-\tfrac{1}{2} (\boldsymbol{t}_{ij}^{adj})^T \operatorname{diag}\Big(\begin{bmatrix} \boldsymbol{\ell} \\ \boldsymbol{\ell} \end{bmatrix}\Big)^{-2} \boldsymbol{t}_{ij}^{adj} \Big)$$
$$\boldsymbol{t}_{ij}^{adj} = \begin{bmatrix} \boldsymbol{v}_\phi \\ \boldsymbol{v}_g \end{bmatrix} = \begin{bmatrix} \phi^{sim}(\boldsymbol{x}_i) - \phi^{sim}(\boldsymbol{x}_j) \\ \bar{g}(\boldsymbol{x}_i) - \bar{g}(\boldsymbol{x}_j) \end{bmatrix}$$

Compare this to $k_{\phi_{adj}}$ from Equation 4:

$$k_{\phi_{adj}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_k^2 \exp\Big(-\tfrac{1}{2} (\boldsymbol{t}_{ij}^{adj})^T \operatorname{diag}\Big(\begin{bmatrix} \boldsymbol{\ell_1} \\ \boldsymbol{\ell_2} \end{bmatrix}\Big)^{-2} \boldsymbol{t}_{ij}^{adj} \Big) \tag{5}$$

Now we see that $k_{\phi_{adj}}^{v_2}$ and $k_{\phi_{adj}}$ have a similar form. Hyperparameters $\boldsymbol{\ell_1}, \boldsymbol{\ell_2}$ provide flexibility in $k_{\phi_{adj}}$ as compared to having only vector $\boldsymbol{\ell}$ in $k_{\phi_{adj}}^{v_2}$. They can be adjusted manually or with Automatic Relevance Determination. For $k_{\phi_{adj}}^{v_2}$, the role of signal variance is captured by $\sigma^2 \exp(-prod_{ij})$. This makes the kernel nonstationary in the transformed ϕ space. Since $k_{\phi_{adj}}$ is already non-stationary in $\boldsymbol{x}$, it is unclear whether non-stationarity of $k_{\phi_{adj}}^{v_2}$ in the transformed ϕ space has any advantages.

The above discussion shows that $k_{\phi_{adj}}$ proposed in Rai et al. (2018) is motivated both intuitively and mathematically. It aims to use a transform that accounts for the hardware mismatch, without adding extra non-stationarity in the transformed space. While the above analysis shows the connection between $k_{\phi_{adj}}$ and $k_{\phi_{adj}}^{v_2}$, a more systematic empirical analysis would be beneficial as part of future work.

4. Robots, Simulators and Controllers Used

In this section we give a concise description of the robots, controllers and simulators used in experiments with BO for bipedal locomotion. Our approach is applicable to a wide range of bipedal robots and controllers, including state-of-the-art controllers (Feng et al., 2015). We work with two different types of controllers – a reactively stepping controller and a human-inspired neuromuscular controller (NMC). The reactively stepping controller is model-based: it uses inverse-dynamics models of the robot to compute desired motor torques. In contrast, NMC is model-free: it computes desired torques using hand-designed policies, created with biped locomotion dynamics in mind. These controllers exemplify two different and widely used ways of controlling bipedal robots. In addition to this, we show results on two different robot morphologies – a parallel bipedal robot ATRIAS, and a serial 7-link biped model. Our hardware experiments are conducted on ATRIAS; the 7-link biped is only used in simulation. Our success on both robots shows that the approaches developed in this paper are widely applicable to a range of bipedal robots and controllers.

4.1. ATRIAS Robot

Our hardware platform is an ATRIAS robot (Figure 1). ATRIAS is a parallel bipedal robot, weighing $\approx 64kg$. The legs are 4-segment linkages actuated by 2 Series Elastic Actuators (SEAs) in the sagittal plane and a DC motor in the lateral plane. Details can be found in Hubicki et al. (2016). In this work we focus on planar motion around a boom. ATRIAS is a highly dynamic system due to its point feet, with static stability only in double stance on the boom.

4.2. Planar 7-link Biped

The second robot used in our experiments is a 7-link biped (Thatte and Geyer, 2016). It has a trunk and segmented legs with ankles, with actuators in the hip, knees and ankles. The inertial properties of its links are similar to an average human (Winter and Yack, 1987). This 7-link model is a canonical simulator for testing bipedal walking algorithms, for example in Song and Geyer (2015). It is a simplified two-dimensional simulator for a large range of humanoid robots, like Atlas (Feng et al., 2015). We use this simulator to study the generalizability of our proposed approaches to systems different from ATRIAS.

4.3. Feedback Based Reactive Stepping Policy

We design a parametric controller for controlling the CoM height, torso angle and the swing leg by commanding desired ground reaction forces and swing foot landing location.

$$F_x = K_{pt}(\theta_{des} - \theta) - K_{dt}\dot{\theta} \qquad F_z = K_{pz}(z_{des} - z) - K_{dz}\dot{z} \qquad x_p = k(v - v_{tgt}) + Cd + 0.5vT$$

Here, F_x is the desired horizontal ground reaction force (GRF), K_{pt} and K_{dt} are the proportional and derivative feedback gains on the torso angle θ and velocity $\dot{\theta}$. F_z is the desired vertical GRF, K_{pz} and K_{dz} are the proportional and derivative gains on the CoM height z and vertical velocity $\dot{z}$. z_{des} and θ_{des} are the desired CoM height and torso lean. x_p is the desired foot landing location for the end of swing; v is the horizontal CoM velocity, k is the feedback gain that regulates v towards the target velocity v_{tgt}. C is a constant and d is the distance between the stance leg and the CoM; T is the swing time.

The desired GRFs are sent to ATRIAS inverse dynamics model that generates desired motor torques τ_f, τ_b. Details can be found in Rai et al. (2018). This controller assumes no double-support, and the swing leg takes off as soon as stance is detected. This leads to a highly dynamic gait, posing a challenging optimization problem.

To investigate the effects of increasing dimensionality on our optimization, we construct two controllers with different number of free parameters:

- *5-dimensional controller* : optimizing 5 parameters $[K_{pt}, K_{dt}, k, C, T]$ (z_{des}, θ_{des} and the feedback on z are hand tuned)
- *9-dimensional controller* : optimizing all 9 parameters of the high-level policy $[K_{pt}, K_{dt}, \theta_{des}, K_{pz}, K_{dz}, z_{des}, k, C, T]$

4.4. 16-dimensional Neuromuscular Controller

We use neuromuscular model policies, as introduced in Geyer and Herr (2010), as our controller for the 7-link planar human-like biped model. These policies use approximate models of muscle dynamics and human-inspired reflex pathways to generate joint torques, producing gaits that are similar to human walking.

Each leg is actuated by 7 muscles, which together produce torques about the hip, knee and ankle. Most of the muscle reflexes are length or force feedbacks on the muscle state aimed at generating a compliant leg, keeping knee from hyperextending and maintaining torso orientation in stance. The swing control has three main components – target leg angle, leg clearance and hip control due to reaction torques. Together with the stance control, this leads to a total of 16 controller parameters, described in details in Antonova et al. (2016).

Though originally developed for explaining human neural control pathways, this controller has recently been applied to prosthetics and bipeds, for example Thatte and Geyer (2016) and Van der Noot et al. (2015). This controller is capable of generating a variety of locomotion behaviours for a humanoid model – walking on rough ground, turning, running, and walking upstairs, making it a very versatile controller (Song and Geyer, 2015). This is a model-free controller as compared to the reactive-stepping controller, which was model-based.

4.5. 50-dimensional Virtual Neuromuscular Controller

Another model-free controller we use on ATRIAS is a modified version of Batts et al. (2015). VNMC maps a neuromuscular model, similar to the one described in Section 4.4 to the ATRIAS robot's topology and emulates it to generate desired motor torques. We adapt VNMC by removing some biological components while preserving its basic functionalities. The final version of the controller consists of 50 parameters including low-level control parameters, such as feedback gains, as well as high level parameters, such as desired step length and desired torso lean. Details can be found in Rai et al. (2018). When optimized using CMA-ES, it can control ATRIAS to walk on rough terrains with height changes of $\pm$20 cm in planar simulation (Batts et al., 2015).

4.6. Increasingly Inaccurate Simulators

To compare the performance of different methods that can be used to transfer information from simulation to hardware, we create a series of increasingly approximate simulators. These simulators emulate increasing mismatch between simulation and hardware and its effect on the information transfer. In this setting, the high-fidelity ATRIAS simulator (Martin et al., 2015), which was used in all the previous simulation experiments becomes the *"hardware"*. Next we make dynamics approximations to the original simulator, which are used commonly in simulators to decrease fidelity and increase simulation speed. For example, the complex dynamics of harmonic drives are approximated as a torque multiplication, and the boom is removed from the simulation, leading to a two-dimensional simulator. These approximate simulators now become the *"simulators"*. As the approximations in these simulators are increased, we expect the performance of methods that utilize simulation for optimization on hardware to deteriorate.

The details of the approximate simulators are described in the two paragraphs below:

1. Simulation with simplified gear dynamics : We replace the original high-fidelity gear model with a commonly used approximation for geared systems – multiplying the rotor torque by the gear ratio. This reduces the simulation time to about a third of the original simulator, but leads to an approximate gear dynamics model.

2. Simulation with no boom and simplified gear dynamics : The ATRIAS robot walks on a boom in our hardware experiments. In our second approximation, we remove the boom from the original simulator and constraint the motion of the robot to a 2-dimensional plane. This is a common approximation for two-dimensional robots. However, the boom leads to lateral forces on the robot, which have vertical force components that are not modelled anymore.



Figure 3: ATRIAS during BO with DoG-based kernel (video: `https://youtu.be/hpXNFREgaRA`)

The advantage of such an arrangement is that we can test the effect of un-modelled and wrongly modelled dynamics on information transfer between simulation and hardware. Even in our high-fidelity original simulator, there are several un-modelled components of the actual hardware. For example, the non-rigidness of the robot parts, misaligned motors and relative play between joints. In our experiments, we find that the 50-dimensional VNMC is a sensitive controller, with little hope of directly transferring from simulation to hardware. Anticipating this, we can now test several methods of compensating for this mismatch using our increasingly approximate simulators.

5. Experiments

We will now present our experiments on optimizing controllers that are 5, 9, 16 and 50 dimensional. We split our experiments into three categories: hardware experiments on the ATRIAS robot, simulation experiments on the 7-link biped and experiments using simulators with different levels of mismatch. We demonstrate that our proposed approach is able to generalize to different controllers and robot structures and is also robust to simulation inaccuracies. The sections below present experimental results, while Appendix A gives further details on data collection, BO implementation and kernel generation.

5.1. Hardware Experiments on the ATRIAS Robot

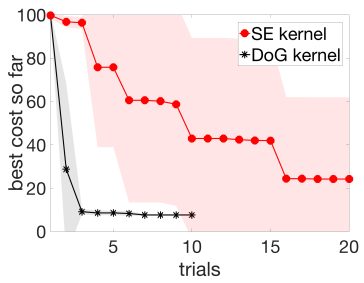
In this section we describe experiments conducted on the ATRIAS robot (hardware from Section 4.1). These experiments were conducted around a boom. The cost function used in our experiments is a slight modification of the cost used in Song and Geyer (2015):

$$cost = \begin{cases} 100 - x_{fall}, \text{if fall} \\ ||v_{avg} - v_{tgt}||, \text{if walk} \end{cases} \quad (6)$$

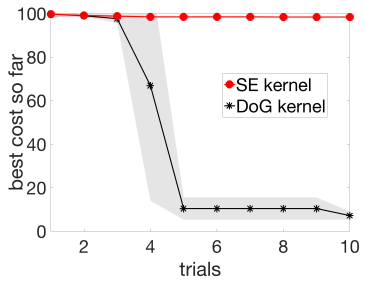
where x_{fall} is distance covered before falling, v_{avg} is average speed per step and v_{tgt} contains target velocity profile, which can be variable. This cost function heavily penalizes falls, and encourages walking controllers to track target velocity.

We do multiple runs of each algorithm on the robot. Each run typically consists of 10 experiments on the robot. All BO runs start from scratch, with an uninformed GP prior. At the end of the run, the GP posterior has 10 data points, depending on the experiment. Each robot trial is designed to be between $30s$ to $60s$ long and the robot is reset to its "home" position between trials.

We will present two sets of hardware experiments in the following subsections. First we describe experiments with the DoG-based kernel on the 5 and 9 dimensional controllers



(a) BO for 5D controller. BO with SE finds walking points in 4/5 runs within 20 trials. BO with DoG-based kernel finds walking points in 5/5 runs within 3 trials.



(b) BO for 9D controller. BO with SE doesn't find walking points in 3 runs. BO with DoG-based kernel finds walking points in 3/3 runs within 5 trials.

Figure 4: BO for 5D and 9D controller on ATRIAS robot hardware. Plots show mean best cost so far. Shaded region shows ± one standard deviation. From Rai et al. (2018).

(first reported in Rai et al. (2018)). The second set describes a new set of experiments for optimizing a 9-dimensional controller using a Neural Network based kernel on hardware.

5.1.1. Experiments with a 5-dimensional controller and DoG-based kernel

In our first set of experiments on the robot, we investigated optimizing the 5-dimensional controller from Section 4.3. For these experiments we picked a challenging variable target speed profile: $0.4m/s$ (15 steps) - $1.0m/s$ (15 steps) - $0.2m/s$ (15 steps) - $0m/s$ (5 steps). The controller was stopped after the robot took 50 steps.

To evaluate the difficulty of this setting, we sampled 100 random points on hardware. 10% of these randomly sampled controllers could walk. In contrast, in simulation the success rate of random sampling was ≈27.5%. This indicates that the simulation was easier, which could be potentially detrimental to algorithms that rely heavily on simulation, because a large portion of controllers that walk in simulation fall on hardware. Nevertheless, using a DoG-based kernel offered significant improvements over a standard SE kernel (Figure 4a).

We conducted 5 runs of BO with DoG-based kernel and 5 runs of BO with SE, 10 trials for DoG-based kernel per run, and 20 for SE kernel. In total, this led to 150 experiments on the robot (excluding the 100 random samples, which were not used during BO). BO with DoG-based kernel found walking points in 100% of runs within 3 trials. In comparison, BO with SE found walking points in 10 trials in 60% runs, and in 80% runs in 20 trials (Figure 4a). Although BO could find walking controllers as early as the second trial (with no prior hardware information), it is worth noting that the optimization did not converge after only a few trials. Sampling more could possibly lead to better walking controllers, but our goal was to find the best controller with a budget of only 10 to 20 trials.

5.1.2. Experiments with a 9-dimensional controller and DoG-based kernel

Our next set of experiments optimized the 9-dimensional controller from Section 4.3. First, we sampled 100 random points for the variable speed profile described above, but this led

to no walking points. To ensure that we have a reasonable baseline we decided to simplify the speed profile for this setting: $0.4m/s$ for 30 steps. We evaluated 100 random points on hardware, and 3 walked for the easier speed profile. In comparison, the success rate in simulation was 8% for the tougher variable-speed profile, implying an even greater mismatch between hardware and simulation than the 5-dimensional controller. For experiments in Sections 5.1.1 and 5.1.2 the inertial measurement unit (IMU) of the robot was broken, and we replaced its functionality with external boom sensors. These were lower resolution than the IMU, leading to noisier readings and larger time delays. As a result, the system did not have a good estimation of vertical height of the CoM, leading to poor control authority. However, the IMU on ATRIAS is a very expensive fiber-optic IMU that is not commonly used on humanoid robots, and most robots use simple state estimation methods. So, this is a common setting for humanoid robots, even if it presents a challenge for the optimization methods.

We conducted 3 runs of BO with DoG-based kernel and BO with SE, 10 trials for DoG-based kernel per run, and 10 for SE. In total, this led to 60 experiments on the hardware (excluding the random samples, which were not used for BO). BO with DoG-based kernel found walking points in 5 trials in 3/3 runs. BO with SE did not find any walking points in 10 trials in all 3 runs. These results are shown in Figure 4b.

Based on these results, we concluded that BO with DoG-based kernel was indeed able to extract useful information from simulation and speed up learning on hardware, even when there was mismatch between simulation and hardware.

5.1.3. Experiments with a 9-dimensional controller and NN-based kernel

In the next set of experiments, we evaluated performance of the NN-based kernel described in Section 3.1.2. We optimize the 9-dimensional controller from Section 4.3.

The target of hardware experiments was to walk for 30 steps at $0.4m/s$, similar to Section 5.1.2.

For these experiments the IMU was repaired, leading to better state estimation on the robot. For a fair comparison, we re-ran experiments with the baseline for this setting and the baseline performed slightly better than the baseline of earlier experiments (because of improved sensing).

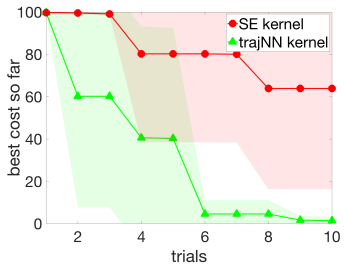
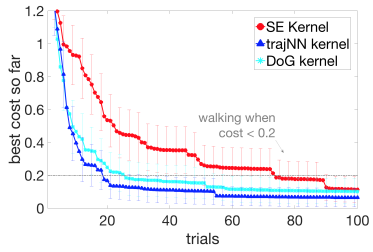
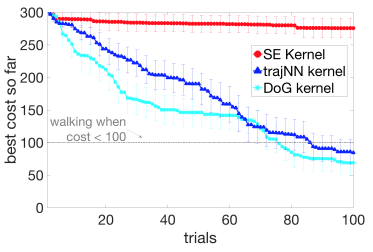


Figure 5: BO for 9D controller on ATRIAS robot hardware.

Figure 5 shows comparison of BO with NN-based kernel and SE kernels. We conducted 5 runs of both algorithms with 10 trials in each run, leading to a total of 100 robot trials. BO with the NN-based kernel found walking points in all 5 runs within 6 trials, while BO with SE kernel only found walking points in 2 of 5 runs in 10 trials. Hence, even without explicit hand-designed domain knowledge, like the DoG-based kernel, the NN-based kernel is able to extract useful information from simulation and successfully guide hardware experiments.



(a) Using smooth cost from Equation 7.



(b) Using non-smooth cost from Equation 8.

Figure 6: BO for the Neuromuscular controller. *trajNN* and *DoG* kernels were constructed with undisturbed model on flat ground. BO is run with mass/inertia disturbances on different rough ground profiles. Plots show means over 50 runs, 95% CIs. From Antonova et al. (2017).

5.2. Simulation Experiments on a 7-link Biped

This section describes simulation experiments with a 16-dimensional Neuromuscular controller (Section 4.4) on a 7-link biped model. These experiments, from Antonova et al. (2017), highlight the cost-agnostic nature of our approach by optimizing two very different costs.

Figure 6 shows BO with DoG-based kernel, NN-based kernel and SE kernel for two different costs from prior literature. The first cost promotes walking further and longer before falling, while penalizing deviations from the target speed (Antonova et al., 2016):

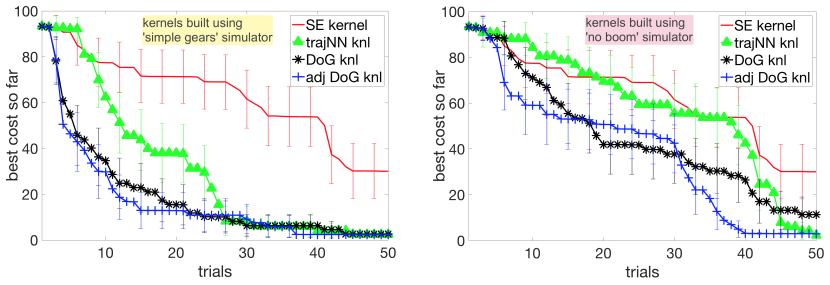
$$cost_{smooth} = 1/(1+t) + 0.3/(1+d) + 0.01(s - s_{tgt}), \tag{7}$$

where t is seconds walked, d is the final CoM position, s is speed and s_{tgt} is the desired walking speed ($1.3m/s$ in our case). The second cost function is similar to the cost used in Section 5. It penalizes falls explicitly, and encourages walking at desired speed and with lower cost of transport:

$$cost_{non\text{-}smooth} = \begin{cases} 300 - x_{fall}, \text{if fall} \\ 100||v_{avg} - v_{tgt}|| + c_{tr}, \text{if walk} \end{cases} \tag{8}$$

where x_{fall} is the distance covered before falling, v_{avg} is the average speed of walking, v_{tgt} is the target velocity, and c_{tr} captures the cost of transport. The changed constants is to account for a longer simulation time. Figure 6a shows that the NN-based kernel and the DoG-based kernel offer a significant improvement over BO with the SE kernel in sample efficiency when using the $cost_{smooth}$, with more than 90% of runs achieving walking after 25 trials. BO with the SE kernel takes 90 trials to get 90% success rate. Figure 6b shows that similar performance by the two proposed approaches is observed on the non-smooth cost. With the NN-based kernel, 70% of the runs find walking solutions after 100 trials, similar to the DoG-based kernel. However, optimizing non-smooth cost is very challenging for BO with the SE kernel: a walking solution is found only in 1 out of 50 runs after 100 trials.

We attribute the difference in performance of the SE kernel on the two costs to the nature of the costs. If a point walks some distance d, Equation 7 reduces in terms of $\frac{1}{d}$ and Equation 8 reduces by $-d$. A sharper fall in the first cost encourages BO to exploit around points that walk some distance, quickly finding points that walk forever. BO with the second cost continues to explore, as the signal is too weak. However the success of both



(a) Informed kernels generated using simulator with simplified gear dynamics.

(b) Informed kernels generated using simplified gear dynamics, without boom model.

Figure 7: BO is run on the original simulator. Informed kernels perform well despite significant mismatch, when kernels are generated using simulator with simplified gear dynamics (left). In the case of severe mismatch, when the boom model is also removed, informed kernels still improve over baseline SE (right). Plots show best cost for mean over 50 runs for each algorithm, 95% CIs.

NN-based and DoG-based kernels on both costs shows that the same kernel can indeed be used for optimizing multiple costs robustly, without any further tuning needed.

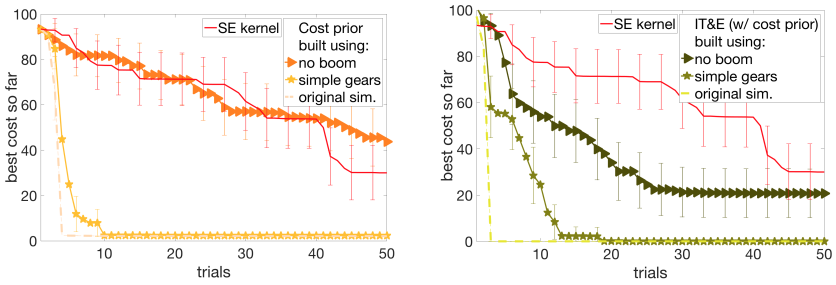
5.3. Experiments with Increasing Simulation-Hardware Mismatch

In this section, we describe our experiments with increasing simulation-hardware mismatch and its effect on approaches that use information from simulation during hardware optimization. The quality of information transfer between simulation and hardware depends not only on the mismatch between the two, but also on the controller used. For a robust controller, small dynamics errors would not cause a significant deterioration in performance, while for a sensitive controller this might be much more detrimental.

In the rest of this section, we provide experimental analyses of settings with increasing simulated mismatch and their effect on optimization of the 50-dimensional VNMC from Section 4.5. We compare several approaches that improve sample-efficiency of BO and investigate if the improvement they offer is robust to mismatch between the simulated setting used for constructing kernel/prior and the setting on which BO is run.

First, we examine the performance of our proposed approaches with informed kernels: k_{DoG}, k_{trajNN} and $k_{DoG_{adj}}$. Figure 7a shows the case when informed kernels are generated using the simulator with simplified gear dynamics while BO is run on the original simulator. After 50 trials, all runs with informed kernels find walking solutions, while for SE only 70% have walking solutions.

Next, Figure 7b shows performance of k_{DoG}, k_{trajNN} and $k_{DoG_{adj}}$ when the kernels are constructed using a simulator with simplified dynamics and without a the boom. In this case the mismatch with the original simulator is larger than before and we see the advantage of using adjustment for DoG-based kernel: $k_{DoG_{adj}}$ finds walking points in all runs after 35 trials. k_{trajNN} also achieves this, but after 50 trials. k_{DoG} finds walking points in $\approx 90\%$



(a) BO with cost prior: straightforward approach useful for low-to-medium mismatch; but no improvement if mismatch is severe.

(b) Performance of IT&E algorithm (our implementation of Cully et al. (2015), adapted to bipedal locomotion).

Figure 8: BO using prior-based approaches. Mean over 50 runs for each algorithm, 95% CIs.

of the runs after 50 trials. The performance of SE stays the same, as it uses no prior information from any simulator.

This illustrates that while the original DoG-based kernel can recover from slight simulation-hardware mismatch, the adjusted DoG-kernel is required for higher mismatch. k_{trajNN} seems to recover from the mismatch, but might benefit from an adjusted version. We leave this to future work.

5.3.1. Comparisons of Prior-based and Kernel-based Approaches

In this section, we classify approaches that use simulation information in hardware optimization as prior-based or kernel-based. Prior-based approaches use costs from the simulation in the prior of the GP used in the BO. This can help BO a lot if the costs are similar between simulation and hardware, and the cost function is fixed. However, in the presence of large mismatch, controllers that perform well in simulation might fail on hardware. A prior-based method can be biased towards sampling promising points from simulation, resulting in worse performance than uninformed BO. Kernel-based approaches consist of methods that incorporate information from simulation into the kernel of the GP. These can be less sample-efficient as compared to prior-based method, but not as likely to be biased towards unpromising regions in the presence of mismatch. They also easily generalize to multiple costs, so that there is no need to re-run simulations for data collection if the cost is changed. This is important because a lot of these approaches can take several days of computation to generate a cost prior or informed kernel. For example, Cully et al. (2015) report taking 2 weeks on a 16-core computer to generate their map.

It is possible to also combine both prior-based and kernel-based methods, as in Cully et al. (2015). We classify these as 'prior-based' methods, since in our experiments prior outweighs the kernel effects for such cases. In our comparison with Cully et al. (2015), we implement a version with and without the prior points. We do not add a cost prior to BO using DoG-based kernel, as this would limit us to a particular cost and high-fidelity simulators, and both of these can be major obstacles in real robot experiments.

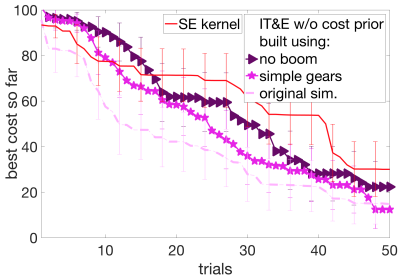
Figure 8a shows the performance when using simulation cost in the prior during BO. BO with a cost prior created using the original version of the simulator illustrates what would happen in the best case scenario. When the simulator with simplified gear dynamics is used for constructing the prior, we observe significant improvements over uninformed BO prior. However, when the prior is constructed from simplified gear dynamics and no boom setting, the approach performs slightly worse than uninformed BO. This shows that while an informed prior can be very helpful when created from a simulator close to hardware, it can hurt performance if simulator is significantly different from hardware.

Next, we discuss experiments with our implementation of Intelligent Trial and Error (IT&E) algorithm from Cully et al. (2015). This algorithm combines adding a cost prior from simulated evaluations with adding simulation information into the kernel. IT&E defines a behavior metric and tabulates best performing points from simulation on their corresponding behavior score. The behavior metric used in our experiments is duty-factor of each leg, which can go from 0 to 1.0. We discretize the duty factor into 21 cells of 0.05 increments, leading to a 21×21 grid. We collect the 5 highest performing controllers for each square in the behavior grid, creating a $21 \times 21 \times 5$ grid. Next, we generate 50 random combinations of a 21×21 grid, selecting 1 out of the 5 best controllers per grid cell. Care was taken to ensure that all 5 controllers had comparable costs in the simulator used for creating the map. Cost of each selected controller is added to the prior and BO is performed in the behavior space, like in Cully et al. (2015).

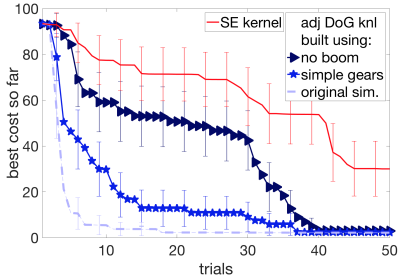
Figure 8b shows BO with IT&E constructed using different versions of the simulator. IT&E constructed using simplified gear dynamics simulator is slightly less sample-efficient than the straightforward 'cost prior' approach. When constructed with the simulator with no boom, IT&E is able to improve over uninformed BO. However, it only finds walking points in 77% of the runs in 50 trials in this case, as some of the generated maps contained no controllers that could walk on the 'hardware'. This is a shortcoming of the IT&E algorithm, as it eliminates a very large part of the search space and if the pre-selected space does not contain a walking point, no walking controllers can be sampled with BO. This problem could possibly be avoided by using a finer grid, or a different behavior metric. However tuning such hyper-parameters can turn out to be expensive, in computation and hardware experiment time.

To separate the effects of using simulation information in prior mean vs kernel, we evaluated a kernel-only version of IT&E algorithm (Figure 9a). It shows that the cost prior is crucial for the success of IT&E and performance deteriorates without it. Hence, it is not practical to use IT&E on a cost different than what it was generated for. Nonetheless, Figure 7 showed that BO with adjusted DoG kernel is able to handle both moderate and severe mismatch with kernel-only information, collected in Figure 9b.

Summarizing this section, we created two simulators with increasing modelling approximations, and studied the effect of using these to aid optimization on the original simulator. We found that while methods that use cost in the prior of BO can be very sample-efficient in low mismatch, their performance worsens as mismatch increases. IT&E, introduced in Cully et al. (2015), uses simulation information in both prior mean and kernel, and is very sample-efficient in cases of low mismatch. Even with high mismatch, it performed better than just prior-based BO but doesn't find walking controllers reliably. In comparison, adjusted DoG-based kernel performed well in all the tested scenarios, and can reliably improve sample-efficiency of BO even when the mismatch between simulation and hardware is high.



(a) BO using our implementation of IT&E without cost prior (from Cully et al. (2015)).



(b) BO using $k_{DoG_{adj}}$ constructed from simulators with various levels of mismatch.

Figure 9: BO using kernel-based approaches. Mean over 50 runs for each algorithm, 95% CIs.

6. Conclusion

In this paper, we presented and analyzed in detail our work from Antonova et al. (2016), Antonova et al. (2017) and Rai et al. (2018). These works introduce domain-specific feature transforms that can be used to optimize locomotion controllers on hardware efficiently. The feature transforms project the original controller space into a space where BO can discover promising regions quickly. We described a transform for bipedal locomotion designed with the knowledge of human walking and a neural network based transform that uses more general information from simulated trajectories. Our experiments demonstrate success at optimizing controllers on the ATRIAS robot. Further simulation-based experiments also indicate potential for other bipedal robots. For optimizing sensitive high-dimensional controllers, we proposed an approach to adjust simulation-based kernel using data seen on hardware. To study the performance of this, as well as compare our approach to other methods, we created a series of increasingly approximate simulators. Our experiments show that while several methods from prior literature can perform well with low simulation-hardware mismatch (sometimes even better than our proposed approach), they suffer when this mismatch increases. In such cases, our proposed kernels with hardware adjustment can yield reliable performance across different costs, simulators and robots.

Acknowledgments

This research was supported in part by National Science Foundation grant IIS-1563807, the Max-Planck-Society, & the Knut and Alice Wallenberg Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding organizations.

Appendix A: Implementation Details

In this appendix we provide a summary of data collection and implementation details. Our implementation of BO was based on the framework in Gardner et al. (2014). We used Expected Improvement (EI) acquisition function (Mockus et al., 1978). We also experimented with Upper Confidence Bound (UCB) (Srinivas et al., 2010), but found that performance was not sensitive to the choice of acquisition function. Hyper-parameters for BO were initialized to default values: 0 for mean offset, 1.0 for kernel length scales and signal variance, 0.1 for σ_n (noise parameter). Hyperparameters were optimized using the marginal likelihood (Shahriari et al. (2016), Section V-A). For all algorithms, we started optimizing hyperparameters after a low-cost controller was found (to save compute resources and avoid premature hyperparameter optimization). The search space boundaries for controller parameters was designed with physical constraints of the ATRIAS robot in mind.

Kernel type	Controller dim	# Sim points	Sim duration	Kernel dim	Features in kernel
k_{DoG}	5	20K	3.5s	1	$score_{DoG}$
	9	100K	5s	1	$score_{DoG}$
	50	200K	5s	1	$score_{DoG}$
k_{trajNN}	9	100K	5s	4	t_{walk}, x_{end}, θ_{avg}, $v_{x,avg}$
	16	100K	5s	8	t_{walk}, x_{end}, θ_{end}, $v_{x,end}$ c_τ, y_{end}, $v_{y,end}$, $\dot{\theta}_{end}$
	50	200K	5s	13	t_{walk}, x_{end}, c_τ, $\boldsymbol{traj}_x$, $\boldsymbol{traj}_\theta$

Table 2: Simulation data collection details. $score_{DoG}$ was described in Section 3.1.1. For k_{trajNN}: t_{walk} is time walked in simulation before falling, x_{end} and y_{end} are the x and y positions of Center of Mass (CoM) at the end of the short simulation, θ is the torso angle, $\dot{\theta}$ is the torso velocity, v is the CoM speed (v_x is the horizontal and v_y is the vertical component), c_τ is the squared sum of torques applied; $\boldsymbol{traj}_x$, $\boldsymbol{traj}_\theta$ denote vectors with mean CoM and θ measurements every second.

k_{trajNN} discussed in Section 3.1.2 was constructed by training a fully connected neural network (NN) with 3 hidden layers, using L1 loss to reconstruct features encoded by trajectory summaries. For experiments with 16D controller in Section 5.2, for example, the hidden layers contained 512, 128, 32 units; NN was trained on 100K simulated examples to reconstruct 8D trajectory summaries (see next-to-last row of Table 2). We experimented with various hidden layer sizes for 9D and 50D controllers, but did not find the overall BO performance to be sensitive to size or other training parameters, like NN learning rate. This was likely because NN used to construct an informed kernel only needs to approximately learn to separate well-performing parts from failing parts of the control parameter space. This is the benefit of our choice not to put the output of NN directly into a GP posterior.

To create cost prior for experiments in Section 5.3 we collected 50,000 evaluations of 30s trials for a range of controller parameters. Then we conducted 50 runs, using random subsets of 35,000 evaluations to construct the prior. The numbers were chosen such that this approach used similar amount of computation as our kernel-based approaches. To accommodate GP prior with a large number of points we used a sparse GP construction provided by Rasmussen and Nickisch (2010).

References

Rika Antonova, Akshara Rai, and Christopher G Atkeson. Sample efficient optimization for learning controllers for bipedal locomotion. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 22–28. IEEE, 2016.

Rika Antonova, Akshara Rai, and Christopher G Atkeson. Deep Kernels for Optimizing Locomotion Controllers. In *Conference on Robot Learning (CoRL), PMLR 78*, pages 47–56, 2017.

Zachary Batts, Seungmoon Song, and Hartmut Geyer. Toward a virtual neuromuscular control for robust walking in bipedal robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6318–6323. IEEE, 2015.

Eric Brochu, Vlad M Cora, and Nando De Freitas. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. *arXiv preprint arXiv:1012.2599*, 2010.

Roberto Calandra, Jan Peters, Carl Edward Rasmussen, and Marc Peter Deisenroth. Manifold Gaussian processes for regression. In *International Joint Conference on Neural Networks (IJCNN)*, pages 3338–3345. IEEE, 2016a.

Roberto Calandra, André Seyfarth, Jan Peters, and Marc Peter Deisenroth. Bayesian Optimization for Learning Gaits Under Uncertainty. *Annals of Mathematics and Artificial Intelligence*, 76(1-2):5–23, 2016b.

Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

Marc Deisenroth and Carl E Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *International Conference on Machine Learning (ICML)*, pages 465–472, 2011.

Peter Englert and Marc Toussaint. Combined Optimization and Reinforcement Learning for Manipulation Skills. In *Robotics: Science and Systems*, 2016.

Siyuan Feng, Eric Whitman, X Xinjilefu, and Christopher G Atkeson. Optimization-based Full Body Control for the DARPA Robotics Challenge. *Journal of Field Robotics*, 32(2): 293–312, 2015.

Jacob R Gardner, Matt J Kusner, Zhixiang Eddie Xu, Kilian Q Weinberger, and John Cunningham. Bayesian Optimization with Inequality Constraints. In *International Conference on Machine Learning (ICML)*, pages 937–945, 2014.

Hartmut Geyer and Hugh Herr. A Muscle-reflex Model that Encodes Principles of Legged Mechanics Produces Human Walking Dynamics and Muscle Activities. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 18(3):263–273, 2010.

Nikolaus Hansen. The CMA evolution strategy: a comparing review. In *Towards a new evolutionary computation*, pages 75–102. Springer, 2006.

Christian Hubicki, Jesse Grimes, Mikhail Jones, Daniel Renjewski, Alexander Spröwitz, Andy Abate, and Jonathan Hurst. ATRIAS: Design and validation of a tether-free 3D-capable spring-mass bipedal robot. *The International Journal of Robotics Research (IJRR)*, 35(12):1497–1521, 2016.

Verne T Inman, Howard D Eberhart, et al. The major determinants in normal and pathological gait. *Journal of Bone and Joint Surgery (JBJS)*, 35(3):543–558, 1953.

Kirthevasan Kandasamy, Gautam Dasarathy, Jeff Schneider, and Barnabás Póczos. Multi-fidelity Bayesian Optimisation with Continuous Approximations. In *International Conference on Machine Learning (ICML)*, pages 1799–1808, 2017.

Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots*, 40(3):429–455, 2016.

Daniel J Lizotte, Tao Wang, Michael H Bowling, and Dale Schuurmans. Automatic Gait Optimization with Gaussian Process Regression. In *International Joint Conference on Artificial Intelligence (IJCAI)*, volume 7, pages 944–949, 2007.

Alonso Marco, Felix Berkenkamp, Philipp Hennig, Angela P Schoellig, Andreas Krause, Stefan Schaal, and Sebastian Trimpe. Virtual vs. real: Trading off simulations and physical experiments in reinforcement learning with Bayesian optimization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1557–1563. IEEE, 2017.

William C Martin, Albert Wu, and Hartmut Geyer. Robust spring mass model running for a physical bipedal robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6307–6312. IEEE, 2015.

Ruben Martinez-Cantin. Funneled Bayesian optimization for design, tuning and control of autonomous systems. *IEEE transactions on cybernetics*, (99):1–12, 2018.

J Mockus, V Tiesis, and A Zilinskas. Toward Global Optimization, Volume 2, Chapter: Bayesian Methods for Seeking the Extremum. 1978.

Xue Bin Peng, Glen Berseth, and Michiel van de Panne. Terrain-adaptive locomotion skills using deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 35(4):81, 2016.

Akshara Rai, Rika Antonova, Seungmoon Song, William Martin, Hartmut Geyer, and Christopher Atkeson. Bayesian optimization using domain knowledge on the ATRIAS biped. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1771–1778. IEEE, 2018.

Carl Edward Rasmussen and Hannes Nickisch. Gaussian processes for machine learning (gpml) toolbox. *J. Mach. Learn. Res.*, 11:3011–3015, December 2010. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1756006.1953029`.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando de Freitas. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems (NIPS)*, pages 2951–2959, 2012.

Jasper Snoek, Kevin Swersky, Rich Zemel, and Ryan Adams. Input warping for Bayesian optimization of non-stationary functions. In *International Conference on Machine Learning (ICML)*, pages 1674–1682, 2014.

Seungmoon Song and Hartmut Geyer. A Neural Circuitry that Emphasizes Spinal Feedback Generates Diverse Behaviours of Human Locomotion. *The Journal of Physiology*, 593(16): 3493–3511, 2015.

Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. In *International Conference on Machine Learning (ICML)*, pages 1015–1022. Omnipress, 2010.

Matthew Tesch, Jeff Schneider, and Howie Choset. Using response surfaces and expected improvement to optimize snake robot gait parameters. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1069–1074. IEEE, 2011.

Nitish Thatte and Hartmut Geyer. Toward Balance Recovery with Leg Prostheses Using Neuromuscular Model Control. *IEEE Transactions on Biomedical Engineering*, 63(5): 904–913, 2016.

Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research (JMLR)*, 11 (Nov):3137–3181, 2010.

Nicolas Van der Noot, Luca Colasanto, Allan Barrea, Jesse van den Kieboom, Renaud Ronsse, and Auke J Ijspeert. Experimental validation of a bio-inspired controller for dynamic walking with a humanoid robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 393–400. IEEE, 2015.

Aaron Wilson, Alan Fern, and Prasad Tadepalli. Using Trajectory Data to Improve Bayesian Optimization for Reinforcement Learning. *The Journal of Machine Learning Research (JMLR)*, 15(1):253–282, 2014.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378, 2016.

DA Winter and HJ Yack. EMG profiles during normal human walking: stride-to-stride and inter-subject variability. *Electroencephalography and clinical neurophysiology*, 67(5): 402–411, 1987.

Journal of Machine Learning Research 20 (2019) 1-24 Submitted 4/18; Revised 1/19; Published 2/19

No-Regret Bayesian Optimization with Unknown Hyperparameters

Felix Berkenkamp BEFELIX@INF.ETHZ.CH
Department of Computer Science
ETH Zurich
Zurich, Switzerland

Angela P. Schoellig SCHOELLIG@UTIAS.UTORONTO.CA
Institute for Aerospace Studies
University of Toronto
Toronto, Canada

Andreas Krause KRAUSEA@ETHZ.CH
Department of Computer Science
ETH Zurich
Zurich, Switzerland

Editor: Bayesian Optimization Special Issue

Abstract

Bayesian optimization (BO) based on Gaussian process models is a powerful paradigm to optimize black-box functions that are expensive to evaluate. While several BO algorithms provably converge to the global optimum of the unknown function, they assume that the hyperparameters of the kernel are known in advance. This is not the case in practice and misspecification often causes these algorithms to converge to poor local optima. In this paper, we present the first BO algorithm that is provably no-regret and converges to the optimum without knowledge of the hyperparameters. During optimization we slowly adapt the hyperparameters of stationary kernels and thereby expand the associated function class over time, so that the BO algorithm considers more complex function candidates. Based on the theoretical insights, we propose several practical algorithms that achieve the empirical sample efficiency of BO with online hyperparameter estimation, but retain theoretical convergence guarantees. We evaluate our method on several benchmark problems.

Keywords: Bayesian optimization, Unknown hyperparameters, Reproducing kernel Hilbert space (RKHS), Bandits, No regret

1. Introduction

The performance of machine learning algorithms often critically depends on the choice of tuning inputs, e.g., learning rates or regularization constants. Picking these correctly is a key challenge. Traditionally, these inputs are optimized using grid or random search (Bergstra and Bengio, 2012). However, as data sets become larger the computation time required to train a single model increases, which renders these approaches less applicable. Bayesian optimization (BO, Mockus (2012)) is an alternative method that provably determines good inputs within few evaluations of the underlying objective function. BO methods construct a statistical model of the underlying objective function and use it to evaluate inputs that are informative about the optimum. However, the theoretical guarantees, empirical performance, and data efficiency of BO algorithms critically depend on their own choice of hyperparameters and, in particular, on the prior distribution over the function space. Thus, we effectively shift the problem of tuning inputs one level up, to the tuning of hyperparameters of the BO algorithm.

In this paper, we use a Gaussian processes (GP, Rasmussen and Williams (2006)) for the statistical model. We present the first BO algorithm that does not require knowledge about the hyperparameters of the GP's stationary kernel and provably converges to the global optimum. To this end, we adapt the hyperparameters of the kernel and our BO algorithm, so that the associated function space grows over time. The resulting algorithm provably converges to the global optimum and retains theoretical convergence guarantees, even when combined with online estimation of hyperparameters.

Related work General BO has received a lot of attention in recent years. Typically, BO algorithms suggest inputs to evaluate by maximizing an acqusition function that measures informativeness about the optimum. Classical acquisition functions are the *expected improvement* over the best known function value encountered so far given the GP distribution (Mockus et al., 1978) and the *Upper Confidence Bound* algorithm, GP-UCB, which applies the 'optimism in the face of uncertainty' principle. The latter is shown to provably converge by Srinivas et al. (2012). Durand et al. (2018) extend this framework to the case of unknown measurement noise. A related method is *truncated variance reduction* by Bogunovic et al. (2016), which considers the reduction in uncertainty at candidate locations for the optimum. Hennig and Schuler (2012) propose *entropy search*, which approximates the distribution of the optimum of the objective function and uses the reduction of the entropy in this distribution as an acquisition function. Alternative information-theoretic methods are proposed by Hernández-Lobato et al. (2014); Wang and Jegelka (2017); Ru et al. (2018). Other alternatives are the *knowledge gradient* (Frazier et al., 2009), which is one-step Bayes optimal, and *information directed sampling* by Russo and Van Roy (2014), which considers a tradeoff between regret and information gained when evaluating an input. Kirschner and Krause (2018) extend the latter framework to heteroscedastic noise.

These BO methods have also been successful empirically. In machine learning, they are used to optimize the performance of learning methods (Brochu et al., 2010; Snoek et al., 2012). BO is also applicable more broadly; for example, in reinforcement learning to optimize a parametric policy for a robot (Calandra et al., 2014; Lizotte et al., 2007; Berkenkamp et al., 2016) or in control to optimize the energy output of a power plant (Abdelrahman et al., 2016). It also forms the backbone of Google vizier, a service for tuning black-box functions (Golovin et al., 2017).

Some of the previous BO algorithms provide theoretical guarantees about convergence to the optimum. These theoretical guarantees only hold when the kernel hyperparameters are known *a priori*. When this is not the case, hyperparameters are often inferred using either *maximum a posteriori* estimates or sampling-based inference (Snoek et al., 2012). Unfortunately, methods that estimate the hyperparameters online are known to get stuck in local optima (Bull, 2011). Instead, we propose to adapt the hyperparameters online in order to enlarge the function space over time, which allows us to provide guarantees in terms of convergence to the global optimum without knowing the hyperparameters. Wang and de

©2019 Felix Berkenkamp, Angela P. Schoellig, Andreas Krause.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-213.html.

Freitas (2014) analyze this setting when a lower bound on the kernel lengthscales is known *a priori*. They decrease the lengthscales over time and bound the regret in terms of the known lower-bound on the lengthscales. Empirically, similar heuristics are used by Wang et al. (2016); Wabersich and Toussaint (2016). In contrast, this paper considers the case where the hyperparameters are *not known*. Moreover, the scaling of the hyperparameters in the previous two papers did not depend on the dimensionality of the problem, which can cause the function space to increase too quickly.

Considering larger function classes as more data becomes available is the core idea behind structural risk minimization (Vapnik, 1992) in statistical learning theory. However, there data is assumed to be sampled independently and identically distributed. This is not the case in BO, where new data is generated actively based on past information.

Our contribution In this paper, we present Adaptive GP-UCB (A-GP-UCB), the first algorithm that provably converges to the globally optimal inputs when BO hyperparameters are *unknown*. Our method expands the function class encoded in the model over time, but does so slowly enough to ensure sublinear regret and convergence to the optimum. Based on the theoretical insights, we propose practical variants of the algorithm with guaranteed convergence. Since our method can be used as an add-on module to existing algorithms with hyperparameter estimation, it achieves similar performance empirically, but avoids local optima when hyperparameters are misspecified. In summary, we:

- Provide theoretical convergence guarantees for BO with unknown hyperparameters;
- Propose several practical algorithms based on the theoretical insights;
- Evaluate the performance in practice and show that our method retains the empirical performance of heuristic methods based on online hyperparameter estimation, but leads to significantly improved performance when the model is misspecified initially.

The remainder of the paper is structured as follows. We state the problem in Sec. 2 and provide relevant background material in Sec. 3. We derive our main theoretical result in Sec. 4 and use insights gained from the theory to propose practical algorithms. We evaluate these algorithms experimentally in Sec. 5 and draw conclusions in Sec. 6. The technical details of the proofs are given in the appendix.

2. Problem Statement

In general, BO considers global optimization problems of the form

$$\mathbf{x}^* = \operatorname*{argmax}_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x}), \tag{1}$$

where $\mathcal{D} \subset \mathbb{R}^d$ is a compact domain over which we want to optimize inputs $\mathbf{x}$, and $f\colon \mathcal{D} \to \mathbb{R}$ is an objective function that evaluates the reward $f(\mathbf{x})$ associated with a given input configuration $\mathbf{x}$. For example, in a machine learning application, $f(\mathbf{x})$ may be the validation loss and $\mathbf{x}$ may be the tuning inputs (e.g., regularization parameters) of the training algorithm. We do not have any significant prior knowledge about the structure of f. Specifically, we cannot assume convexity or that we have access to gradient information. Moreover, evaluations of f are corrupted by σ-sub-Gaussian noise, a general class of noise models that includes, for example, bounded or Gaussian noise.

Regret We aim to construct a sequence of input evaluations $\mathbf{x}_t$, that eventually maximizes the function value $f(\mathbf{x}_t)$. One natural way to prove this convergence is to show that an algorithm has sublinear regret. The instantaneous regret at iteration t is defined as $r_t = \max_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x}) - f(\mathbf{x}_t) \geq 0$, which is the loss incurred by evaluating the function at $\mathbf{x}_t$ instead of at the *a priori unknown* optimal inputs. The cumulative regret is defined as $R_T = \sum_{0<t\leq T} r_t$, the sum of regrets incurred over T steps. If we can show that the cumulative regret is sublinear for a given algorithm, that is, $\lim_{t\to\infty} R_t \,/\, t = 0$, then eventually the algorithm evaluates the function at inputs that lead to close-to-optimal function values most of the time. We say that such an algorithm has *no-regret*. Intuitively, if the average regret approaches zero then, on average, the instantaneous regret must approach zero too, since r_t is strictly positive. This implies that there exists a $t > 0$ such that $f(\mathbf{x}_t)$ is arbitrarily close to $f(\mathbf{x}^*)$ and the algorithm converges. Thus, we aim to design an optimization algorithm that has sublinear regret.

Regularity assumptions Without further assumptions, it is impossible to achieve sublinear regret on (1). In the worst case, f could be discontinuous at every input in $\mathcal{D}$. To make the optimization problem in (1) tractable, we make regularity assumptions about f. In particular, we assume that the function f has low complexity, as measured by the norm in a reproducing kernel Hilbert space (RKHS, Christmann and Steinwart (2008)). An RKHS $\mathcal{H}_k$ contains well-behaved functions of the form $f(\mathbf{x}) = \sum_{i\geq 0} \alpha_i\, k(\mathbf{x}, \mathbf{x}_i)$, for given representer points $\mathbf{x}_i \in \mathbb{R}^d$ and weights $\alpha_i \in \mathbb{R}$ that decay sufficiently quickly. The kernel $k(\cdot,\cdot)$ determines the roughness and size of the function space and the induced RKHS norm $\|f\|_k = \sqrt{\langle f,\, f\rangle}$ measures the complexity of a function $f \in \mathcal{H}_k$ with respect to the kernel.

In the following, we assume that f in (1) has bounded RKHS norm $\|f\|_{k_\theta} \leq B$ with respect to a kernel k_θ that is parameterized by hyperparameters θ. We write $\mathcal{H}_\theta$ for the corresponding RKHS, $\mathcal{H}_{k_\theta}$. For known B and θ, no-regret BO algorithms for (1) are known, e.g., GP-UCB (Srinivas et al., 2012). In practice, these hyperparameters need to be tuned. In this paper, we consider the case where θ and B are unknown. We focus on stationary kernels, which measure similarity based on the distance of inputs, $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x} - \mathbf{x}')$. The most commonly used hyperparameters for these kernels are the lengthscales $\theta \in \mathbb{R}^d$, which scale the inputs to the kernel in order to account for different magnitudes in the different components of $\mathbf{x}$ and effects on the output value. That is, we scale the difference $\mathbf{x} - \mathbf{x}'$ by the lengthscales θ,

$$k_\theta(\mathbf{x}, \mathbf{x}') = k\left(\frac{[\mathbf{x}]_1 - [\mathbf{x}']_1}{[\theta]_1}, \ldots, \frac{[\mathbf{x}]_d - [\mathbf{x}']_d}{[\theta]_d}\right), \tag{2}$$

where $[\mathbf{x}]_i$ denotes the ith element of $\mathbf{x}$. Typically, these kernels assign larger similarity scores to inputs when the scaled distance between these two inputs is small. Another common hyperparameter is the prior variance of the kernel, a multiplicative constant that determines the magnitude of the kernel. We assume $k(\mathbf{x}, \mathbf{x}) = 1$ for all $\mathbf{x} \in \mathcal{D}$ without loss of generality, as any multiplicative scaling can be absorbed by the norm bound B.

In summary, our goal is to efficiently solve (1) via a BO algorithm with sublinear regret, where f lies in some RKHS $\mathcal{H}_\theta$, but neither the hyperparameters θ nor the norm-bound $\|f\|_{k_\theta}$ are known.

3. Background

In this section, we review Gaussian processes (GPs) and Bayesian optimization (BO).

3.1. Gaussian processes (GP)

Based on the assumptions in Sec. 2, we can use GPs to infer confidence intervals on f. The goal of GP inference is to infer a posterior distribution over the nonlinear map $f(\mathbf{x}) : D \to \mathbb{R}$ from an input vector $\mathbf{x} \in D$ to the function value $f(\mathbf{x})$. This is accomplished by assuming that the function values $f(\mathbf{x})$, associated with different values of $\mathbf{x}$, are random variables and that any finite number of these random variables have a joint Gaussian distribution (Rasmussen and Williams, 2006). A GP distribution is parameterized by a prior mean function and a covariance function or kernel $k(\mathbf{x}, \mathbf{x}')$, which defines the covariance of any two function values $f(\mathbf{x})$ and $f(\mathbf{x}')$ for $\mathbf{x}, \mathbf{x}' \in D$. In this work, the mean is assumed to be zero without loss of generality. The choice of kernel function is problem-dependent and encodes assumptions about the unknown function.

We can condition a $GP(0, k(\mathbf{x}, \mathbf{x}'))$ on a set of t past observations $\mathbf{y}_t = (y_1, \dots, y_t)$ at inputs $\mathcal{A}_t = \{\mathbf{x}_1, \dots, \mathbf{x}_t\}$ in order to obtain a posterior distribution on $f(\mathbf{x})$ for any input $\mathbf{x} \in D$. The GP model assumes that observations are noisy measurements of the true function value, $y_t = f(\mathbf{x}_t) + \omega_t$, where $\omega_t \sim \mathcal{N}(0, \sigma^2)$. The posterior distribution is again a $GP(\mu_t(\mathbf{x}), k_t(\mathbf{x}, \mathbf{x}'))$ with mean μ_t, covariance k_t, and variance σ_t, where

$$\mu_t(\mathbf{x}) = \mathbf{k}_t(\mathbf{x})(\mathbf{K}_t + \mathbf{I}\sigma^2)^{-1}\mathbf{y}_t, \tag{3}$$

$$k_t(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') - \mathbf{k}_t(\mathbf{x})(\mathbf{K}_t + \mathbf{I}\sigma^2)^{-1}\mathbf{k}_t^{\mathrm{T}}(\mathbf{x}'), \tag{4}$$

$$\sigma_t^2(\mathbf{x}) = k_t(\mathbf{x}, \mathbf{x}). \tag{5}$$

The covariance matrix $\mathbf{K}_t \in \mathbb{R}^{t \times t}$ has entries $[\mathbf{K}_t]_{(i,j)} = k(\mathbf{x}_i, \mathbf{x}_j)$, $i, j \in \{1, \dots, t\}$, and the vector $\mathbf{k}_t(\mathbf{x}) = \big[k(\mathbf{x}, \mathbf{x}_1), \dots, k(\mathbf{x}, \mathbf{x}_t)\big]$ contains the covariances between the input $\mathbf{x}$ and the observed data points in $\mathcal{A}_t$. The identity matrix is denoted by $\mathbf{I}_t \in \mathbb{R}^{t \times t}$.

3.2. Learning RKHS functions with GPs

The GP framework uses a statistical model that makes different assumptions from the ones made about f in Sec. 2. In particular, we assume a different noise model, and samples from a $\mathrm{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ are rougher than RKHS funtions and are not contained in $\mathcal{H}_k$. However, GPs and RKHS functions are closely related (Kanagawa et al., 2018) and it is possible to use GP models to infer reliable confidence intervals on f in (1).

Lemma 1 (Abbasi-Yadkori (2012); Chowdhury and Gopalan (2017)) *Assume that f has bounded RKHS norm $\|f\|_k \leq B$ and that measurements are corrupted by σ-sub-Gaussian noise. If $\beta_t^{1/2} = B + 4\sigma\sqrt{I(\mathbf{y}_t; f) + 1 + \ln(1/\delta)}$, then for all $\mathbf{x} \in D$ and $t \geq 0$ it holds jointly with probability at least $1 - \delta$ that $|\, f(\mathbf{x}) - \mu_t(\mathbf{x})\,| \leq \beta_t^{1/2}\sigma_t(\mathbf{x})$.*

Lemma 1 implies that, with high probability, the true function f is contained in the confidence intervals induced by the posterior GP distribution that uses the kernel k from Lemma 1 as a covariance function, scaled by an appropriate factor β_t. Here, $I(\mathbf{y}_t; f)$ denotes the mutual information between the GP prior on f and the t measurements $\mathbf{y}_t$. Intriguingly, for

GP models this quantity only depends on the inputs $\mathbf{x}_t$ and not the corresponding measurement y_t. Specifically, for a given set of measurements $\mathbf{y}_\mathcal{A}$ at inputs $\mathbf{x} \in \mathcal{A}$, the mutual information is given by

$$I(\mathbf{y}_\mathcal{A}; f) = 0.5 \log |\mathbf{I} + \sigma^{-2}\mathbf{K}_\mathcal{A}|, \tag{6}$$

where $\mathbf{K}_\mathcal{A}$ is the kernel matrix $[k(\mathbf{x}, \mathbf{x}')]_{\mathbf{x},\mathbf{x}' \in \mathcal{A}}$ and $|\cdot|$ is the determinant. Intuitively, the mutual information measures how informative the collected samples $\mathbf{y}_\mathcal{A}$ are about the function f. If the function values are independent of each other under the GP prior, they will provide large amounts of new information. However, if measurements are taken close to each other as measured by the kernel, they are correlated under the GP prior and provide less information.

3.3. Bayesian Optimization (BO)

BO aims to find the global maximum of an unknown function (Mockus, 2012). The framework assumes that evaluating the function is expensive in terms of time required or monetary costs, while other computational resources are comparatively inexpensive. In general, BO methods model the objective function f with a statistical model and use it to determine informative sample locations. A popular approach is to model the underlying function with a GP, see Sec. 3.1. GP-based BO methods use the posterior mean and variance predictions in (3) and (5) to compute the next sample location.

One commonly used algorithm is the GP-UCB algorithm by Srinivas et al. (2012). It uses confidence intervals on the function f, e.g., from Lemma 1, in order to select as next input the point with the largest plasuble function value according to the model,

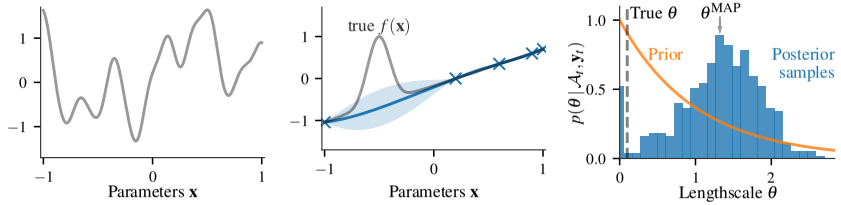
$$\mathbf{x}_{t+1} = \operatorname*{argmax}_{\mathbf{x} \in \mathcal{D}} \mu_t(\mathbf{x}) + \beta_t^{1/2}\sigma_t(\mathbf{x}). \tag{7}$$

Intuitively, (7) selects new evaluation points at locations where the upper bound of the confidence interval of the GP estimate is maximal. Repeatedly evaluating the function f at inputs $\mathbf{x}_{t+1}$ given by (7) improves the mean estimate of the underlying function and decreases the uncertainty at candidate locations for the maximum, so that the global maximum is provably found eventually (Srinivas et al., 2012). While (7) is also an optimization problem, it only depends on the GP model of f and solving it therefore does not require any expensive evaluations of f.

Regret bounds Srinivas et al. (2012) show that the GP-UCB algorithm has cumulative regret $R_t = \mathcal{O}(\sqrt{t\beta_t\gamma_t})$ for all $t \geq 1$ with the same $(1 - \delta)$ probability as the confidence intervals, e.g., in Lemma 1, hold. Here γ_t is the largest amount of mutual information that could be obtained by any algorithm from at most t measurements,

$$\gamma_t = \max_{\mathcal{A} \subset D, |\mathcal{A}| \leq t} I(\mathbf{y}_\mathcal{A}; f). \tag{8}$$

We refer to γ_t as the *information capacity*, since it can be interpreted as a measure of complexity of the function class associated with a GP prior. It was shown by Srinivas et al. (2012) that γ_t has a sublinear dependence on t for many commonly used kernels such as the Gaussian kernel. As a result, R_t has a sublinear dependence on t so that $R_t/t \to 0$ and therefore GP-UCB converges to function evaluations close to $f(\mathbf{x}^*)$. These regret bounds

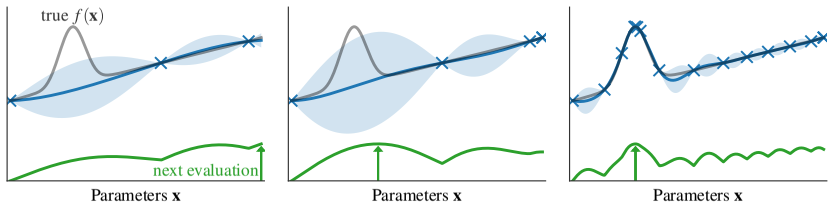


(a) Sample from GP prior. (b) GP estimate (RKHS). (c) Lengthscale distribution.

Figure 1: A sample from the GP prior in Fig. 1a typically varies at a consistent rate over the input space. However, RKHS functions with the same kernel may be less consistent and can have bumps, as in Fig. 1b (gray). As a result, inferring the posterior lengthscales based on measurements (blue crosses in Fig. 1b) can lead to erroneous results. In Fig. 1c, most of the probability mass of the posterior lengthscales has concentrated around large lengthscales that encode smooth functions. Consequently, the GP's 2σ confidence intervals in Fig. 1b (blue shaded) based on the posterior samples do not contain the true function.

were extended to Thompson sampling, an algorithm that uses samples from the posterior GP as the acquisition function, by Chowdhury and Gopalan (2017).

Online hyperparameter estimation In the previous section, we have seen that the GP-UCB algorithm provably converges. However, it requires access to a RKHS norm bound $\|f\|_\theta \leq B$ under the correct kernel hyperparameters θ in order to construct reliable confidence intervals using Lemma 1. In practice, these are unknown and have to be estimated online, e.g., based on a prior distribution placed on θ. Unfortunately, it is well-known that online estimation of the inputs, be it via maximum a posteriori (MAP) or sampling methods, does not always converge to the optimum (Bull, 2011). The problem does not primarily lie with the inference scheme, but rather with the assumptions made by the GP. In particular, typical samples drawn from a GP with a stationary kernel tend to have a similar rate of change throughout the input space, see Fig. 1a. In contrast, the functions inside the RKHS, as specified in Sec. 2, can have different rates of change and are thus improbable under the GP prior. For example, the grey function in Fig. 1b is almost linear but has one bump that defines the global maximum, which makes this function an improbable sample under the GP prior even though it belongs to the RKHS induced by the same kernel. This property of GPs with stationary kernels means that, for inference, it is sufficient to estimate the lengthscales in a small part of the state-space in order to make statements about the function space globally. This is illustrated in Fig. 1c, where we show samples from the posterior distribution over the lengthscales based on the measurements obtained from the GP-UCB algorithm in Fig. 1b (blue crosses). Even though the prior distribution on the lengthscales θ is suggestive of short lengthscales, most of the posterior probability mass is concentrated around lengthscales that are significantly larger than the true ones. As a result, even under model averaging over the samples from the posterior distribution of the lengthscales, the GP confidence intervals do not contain the true function in Fig. 1b. This is not a problem of the inference method applied, but rather a direct consequence of



(a) Stuck in local optimum. (b) Expanding the function class. (c) Global optimum found.

Figure 2: BO algorithms get stuck in local optima when the hyperpararameters of the model are misspecified. In Fig. 2a, the true function is not contained within the GP's confidence intervals (blue shaded), so that GP-UCB only collects data at the local optimum on the right (green arrow), see also Fig. 1. Our method expands the function class over time by scaling the hyperparameters, which encourages additional exploration in Fig. 2b. The function class grows slowly enough, so that the global optimum is provably found in Fig. 2c.

the probabilistic model that we have specified based on the stationary kernel, which does not consider functions with different rates of change to be likely.

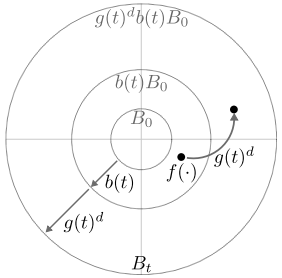
4. The Adaptive GP-UCB Algorithm

In this section, we extend the GP-UCB algorithm to the case where neither the norm bound B nor the lengthscales θ are known. In this case, it is always possible that the local optimum is defined by a local bump based on a kernel with small lengthscales, which has not been encountered by the data points as in Fig. 1b. The only solution to avoid this problem is to keep exploring to eventually cover the input space $\mathcal{D}$ (Bull, 2011). We consider expanding the function space associated with the hyperparameters slowly over time, so that we obtain sublinear regret once the true function class has been identified. Intuitively, this can help BO algorithms avoid premature convergence to local optima caused by misspecified hyperparameters θ and B. For example, in Fig. 2a, the GP-UCB algorithm has converged to a local maximum. By decreasing the lengthscales, we increase the underlying function class, which means that the GP confidence intervals on the function increase. This enables GP-UCB to explore further so that the global optimum is found, as shown in Fig. 2c.

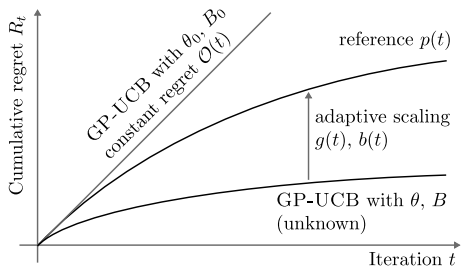
Specifically, we start with an initial guess θ_0 and B_0 for the lengthscales and norm bound on f, respectively. Over the iterations, we scale down the lengthscales and scale up the norm bound,

$$\theta_t = \frac{1}{g(t)}\,\theta_0, \qquad B_t = b(t)g(t)^d\,B_0, \tag{9}$$

where $g\colon \mathbb{N} \to \mathbb{R}_{>0}$ and $b\colon \mathbb{N} \to \mathbb{R}_{>0}$ with $b(0) = g(0) = 1$ are functions that can additionally depend on the data collected up to iteration t, $\mathcal{A}_t$ and $\mathbf{y}_t$. As $g(t)$ increases, the lengthscales θ_t of the kernel become shorter, which enlarges the underlying function space:



(a) Scaling of the norm bound.



(b) Cumulative regret with scaling.

Figure 3: The function f in Fig. 3a has RKHS norm $\|f\|_{\theta_0} > B_0$. To account for this, we expand the norm ball by $b(t)$ over time. When we scale down the lengthscales by $g(t)$, the norm of f in the resulting RKHS is larger, see Lemma 2. We account for this when defining the norm ball B_t in (9). In Fig. 3b, the GP-UCB algorithm based on the misspecified hyperparameters B_0 and θ_0 does not converge (constant regret). Our method scales the lengthscales and norm bound by $g(t)$ and $b(t)$, so that we eventually capture the true model. Scaling the hyperparameters beyond the true ones leads to additional exploration and thus larger cumulative regret than GP-UCB with the true, unknown hyperparameters θ and B. However, as long as the cumulative regret is upper bounded by a sublinear function p, ultimately the A-GP-UCB algorithm converges to the global optimum.

Lemma 2 *(Bull, 2011, Lemma 4) If $f \in \mathcal{H}_\theta$, then $f \in \mathcal{H}_{\theta'}$ for all $0 < \theta' \leq \theta$, and*

$$\|f\|^2_{\mathcal{H}_{\theta'}} \leq \left(\prod_{i=1}^{d} \frac{[\theta]_i}{[\theta']_i} \right) \|f\|^2_{\mathcal{H}_\theta} . \tag{10}$$

Lemma 2 states that when decreasing the lengthscales θ, the resulting function space contains the previous one. Thus, as $g(t)$ increases we consider larger RKHS spaces as candidate spaces for the function f. In addition, as we increase $b(t)$, we consider larger norm balls within the function space $\mathcal{H}_{\theta_t}$, which corresponds to more complex functions. However, it follows from (10) that, as we increase $g(t)$, we also increase the norm of any existing function in $\mathcal{H}_{\theta_0}$ by at most a factor of $g(t)^d$. This is illustrated in Fig. 3a: as we scale up the norm ball to $b(t)B_0$, we capture f under the initial lengthscales θ_0. However, by shortening the lengthscales by $g(t)$, the function f has a larger norm in the new function space $\mathcal{H}_{\theta_t} = \mathcal{H}_{\theta_0/g(t)}$. We account for this through the additional scaling factor $g(t)^d$ in the norm bound B_t in (9).

Theoretical analysis Based on the previous derivations together with Lemma 2, it is clear that, if $g(t)$ and $b(t)$ are monotonically increasing functions and $f \in \mathcal{H}_{\theta_{t^*}}$ with $\|f\|_{\theta_{t^*}} \leq B_{t^*}$ for some $t^* > 0$, then $f \in \mathcal{H}_{\theta_t}$ and $\|f\|_{\theta_t} \leq B_t$ for all $t \geq t^*$. That is, once the function f is contained within the norm ball of B_{t^*} for the lengthscales θ_{t^*}, then, for any further increase in $b(t)$ or $g(t)$, the function f is still contained in the candidate space $\{f \in \mathcal{H}_{\theta_t} \,|\, f \leq B_t\}$.

Based on this insight, we propose A-GP-UCB in Algorithm 1. At iteration t, A-GP-UCB sets the GP lengthscales to θ_t and selects new inputs $\mathbf{x}_{t+1}$ similar to the GP-UCB algorithm, but based on the norm bound B_t. We extend the analysis of GP-UCB and Lemma 1 to obtain our main result.

Theorem 1 *Assume that f has bounded RKHS norm $\|f\|^2_{k_\theta} \leq B$ in a RKHS that is parametrized by a stationary kernel $k_\theta(\mathbf{x}, \mathbf{x}')$ with unknown lengthscales θ. Based on an initial guess, θ_0 and B_0, define monotonically increasing functions $g(t) > 0$ and $b(t) > 0$ and run* A-GP-UCB *with $\beta_t^{1/2} = b(t)g(t)^d B_0 + 4\sigma\sqrt{I_{\theta_t}(\mathbf{y}_t; f) + 1 + \ln(1/\delta)}$ and GP lengthscales $\theta_t = \theta_0/g(t)$. Then, with probability at least $(1-\delta)$, we obtain a regret bound of*

$$R_t \leq 2B \max\left(g^{-1}\left(\max_i \frac{[\theta_0]_i}{[\theta]_i} \right), b^{-1}\left(\frac{B}{B_0} \right) \right) + \sqrt{C_1 t \beta_t I_{\theta_t}(\mathbf{y}_t; f)}, \tag{11}$$

where I_{θ_t} is the mutual information in (6) *based on the GP model with lengthscales θ_t and $C_1 = 8/\log(1+\sigma^{-2})$.*

The proof is given in the appendix. Intuitively, the regret bound in (11) splits the run of the algorithm into two distinct phases. In the first one, either the RKHS space $\mathcal{H}_{\theta_t}(\mathcal{D})$ or the norm bound B_t are too small to contain the true function f. Thus, the GP confidence intervals scaled by $\beta_t^{1/2}$ do not necessarily contain the true function f, as in Fig. 1b. In these iterations, we obtain constant regret that is bounded by $2B$, since $\|f\|_\infty \leq \|f\|_\theta \leq B$. After both g and b have grown sufficiently in order for the considered function space to contain the true function, the confidence bounds are reliable and we can apply the theoretical results of the GP-UCB algorithm. This is illustrated in Fig. 3b: If the initial hyperparameters θ_0 and B_0 are misspecified, the confidence intervals do not contain f and GP-UCB does not converge. We avoid this problem by increasing $b(t)$ and $g(t)$ over time, so that we eventually contain f in our function class. However, increasing the norm ball and decreasing the lengthscales beyond the true ones causes additional exploration and thus additional cumulative regret relative to GP-UCB with the true, unknown hyperparameters. This additional regret represents the cost of not knowing the hyperparameters in advance. As long as the overall regret remains bounded by a sublinear function $p(t)$, our method eventually converges to the global optimum. The regret bound in (11) depends on the true hyperparameters θ and B. However, the algorithm does not depend on them. Theorem 1 provides an instance-specific bound, since the mutual information depends on the inputs in $\mathcal{A}_t$. One can obtain a worst-case upper bound by bounding $I_{\theta_t}(\mathbf{y}_t; f) \leq \gamma_t(\theta_t)$, which is the worst-case mutual information as in (8), but based on the GP model with lengthscales θ_t. While Theorem 1 assumes that the noise properties are known, the results can be extended to estimate the noise similar to Durand et al. (2018).

For arbitrary functions $g(t)$ and $b(t)$, the candidate function space $\{f \in \mathcal{H}_{\theta_t} \,|\, f \leq B_t\}$ can grow at a faster rate than it contracts by selecting informative measurements y_t according to (7). In particular, in the regret term $\sqrt{C_1 t \beta_t \gamma_t}$ both β_t and γ_t depend on the scaling factors $g(t)$ and $b(t)$. If these factors grow at a faster rate than $\sqrt{t}$, the resulting algorithm does not enjoy sublinear regret. We have the following result that explicitly states the dependence of γ_t on the scaling factor $g(t)$.

Proposition 2 *Let k_θ be a stationary kernel parameterized by lengthscales θ as in* (2) *and define $\gamma_t(\theta)$ for lengthscales θ as in* (8)*. Define the lengthscales as $\theta_t = \theta_0/g(t)$ as in* (9)*.*

Algorithm 1 Adaptive GP-UCB(A-GP-UCB)

Input: Input space $\mathcal{D}$, $GP(0, k(\mathbf{x}, \mathbf{x}'))$, functions $g(t)$ and $b(t)$
Set $B_0 = 1$ and $\theta_0 = \text{diam}(\mathcal{D})$
for all $t = 0, 1, 2, \ldots$ **do**
  Set the GP kernel lengthscsales to $\theta_t = \theta_0 / g(t)$
  $\beta_t^{1/2} \leftarrow B(t) + 4\sigma\sqrt{I_{\theta_t}(\mathbf{y}_t; f) + 1 + \ln(1/\delta)}$ with $B(t) = b(t)g(t)^d B_0$
  Choose $\mathbf{x}_{t+1} = \text{argmax}_{\mathbf{x}\in\mathcal{D}}\ \mu_t(\mathbf{x}) + \beta_t^{1/2}\sigma_t(\mathbf{x})$
  Evaluate $y_{t+1} = f(\mathbf{x}_{t+1}) + \epsilon_{t+1}$
  Perform Bayesian update to obtain μ_{t+1} and σ_{t+1}

- *If* $k(\mathbf{x}, \mathbf{x}') = \exp(-\frac{1}{2}\|\mathbf{x} - \mathbf{x}'\|_2^2)$ *is the squared exponential (Gaussian) kernel, then*

$$\gamma_t(\theta_t) = \mathcal{O}\left(g(t)^d (\log t)^{d+1}\right) \tag{12}$$

- *If* $k(\mathbf{x}, \mathbf{x}') = \left(2^{1-\nu}/\Gamma(\nu)\right) r^\nu B_\nu(r)$ *is the Matérn kernel, where* $r = \sqrt{2\nu}\|\mathbf{x} - \mathbf{x}'\|_2$, B_ν *is the modified Bessel function with* $\nu > 1$, *and* Γ *is the gamma function. Then*

$$\gamma_t(\theta_t) = \mathcal{O}\left(g(t)^{2\nu+d} t^{\frac{d(d+1)}{2\nu+d(d+1)}} \log t\right) \tag{13}$$

Proposition 2 explicitly states the relationship between γ_t and $g(t)$. For the Gaussian kernel, if we scale down the lengthscales by a factor of two, the amount of mutual information that we can gather in the worst case, γ_t, grows by 2^d. Given the dependence of γ_t on $g(t)$, we can refine Theorem 1 to obtain concrete regret bounds for two commonly used kernels.

Corollary 3 *If, under the assumptions of Theorem 1,* $g(t)$ *and* $b(t)$ *grow unbounded, then we obtain the following, high-probability regret bounds for Algorithm 1:*

- *Squared exponential kernel:* $R_t \leq \mathcal{O}\left(b(t)\sqrt{t g(t)^{3d}\gamma_t(\theta_0)} + g(t)^d \gamma_t(\theta_0)\sqrt{t}\right)$;
- *Matérn kernel:* $R_t \leq \mathcal{O}\left(b(t)\sqrt{t g(t)^{2\nu+3d}\gamma_t(\theta_0)} + g(t)^{\nu+d}\gamma_t(\theta_0)\sqrt{t}\right)$.

If $b(t)$ and $g(t)$ grow unbounded, the first term of the cumulative regret in (11) can be upper bounded by a constant. The remaining result is obtained by plugging in β_t and the bounds from (8). Thus, any functions $g(t)$ and $b(t)$ that render the regret bounds in Corollary 3 sublinear allow the algorithm to converge, even though the true lengthscales and norm bound are unknown.

The specific choices of $b(t)$ and $g(t)$ matter for the regret bound in Theorem 1 in practice. Consider the one-dimensional case $d = 1$ for the Gaussian kernel. Given the true hyperparameters B and θ, if we set $g(t) = \theta_0/\theta$ and $b(t) = B/B_0$ to be constant, we recover the non-adaptive regret bounds of GP-UCB with known hyperparameters. If $g(t)$ depends on t and grows slowly, then the algorithm incurs constant regret during the initial rounds when the model is misspecified, while functions g that grow to values larger than the optimal ones lead to additional exploration and incur an additional $\mathcal{O}(b(t)g(t)^{3d/2}))$ factor in the cumulative regret in later rounds, as in Corollary 3. In the following section, we discuss appropriate choices for these functions in practice.

4.1. Choosing the scaling functions $g(t)$ and $b(t)$

It follows from Theorem 1 that A-GP-UCB achieves no-regret for any functions $g(t)$ and $b(t)$ that increase without bound and render (11) sublinear in t. Thus, the corresponding BO routine converges to the optimal value eventually. For example, $b(t) = g(t) = \log(t)$ satisfy this condition. However, the convergence guarantees in Theorem 1 are only meaningful once t has grown sufficiently so that the true function is contained in the confidence intervals. In practice, BO is often used with objective functions f that are expensive to evaluate, which imposes a hard constraint on the number of evaluations. For the regret bounds to be meaningful in this setting, we must choose functions g and b that grow fast enough to ensure that the constant regret period in (11) is small, yet slow enough that the effect of the sublinear regret is visible for small enough t. In the following, we propose two methods to choose $g(t)$ and $b(t)$ *adaptively*, based on the observations seen so far.

For convenience, we fix the relative magnitude of $g(t)$ and $b(t)$. In particular, we define $b(t) = 1 + \epsilon_b(t)$ and $g(t)^d = 1 + \epsilon_g(t)$ together with a weighting factor $\lambda = \epsilon_b(t)/\epsilon_g(t)$ that encodes whether we prefer to scale up the norm bound using $b(t)$ or decrease the lengthscales using $g(t)$. This allows us to reason about the overall magnitude of the scaling $h(t) = (1 + \epsilon_g(t))(1 + \epsilon_b(t)) \geq 1$, which can be uniquely decomposed into $g(t)$ and $b(t)$ given λ. For $\lambda = 0$ we have $g(t) = h(t)$, $b(t) = 1$ and the algorithm prefers to attribute an increase in $h(t)$ to $g(t)$ and shorten the lengthscales, while for $\lambda \to \infty$ the algorithm prefers to scale up the RKHS norm. The assumptions in Corollary 3 hold for any $\lambda \in (0, \infty)$ if $h(t)$ grows unbounded. Moreover, we have that $g(t)^d \leq h(t)$ and $b(t) \leq h(t)$.

Reference regret While any function $h(t)$ that grows unbounded and renders the cumulative regret in Theorem 1 sublinear makes our method to converge to the optimum eventually, we want to ensure that our method performs well in finite time too. For fixed hyperparameters with $h(t) = 1$, which implies $g(t) = b(t) = 1$, our algorithm reduces to GP-UCB with hyperparameters θ_0 and B_0 and the regret bound term $\sqrt{C_1\beta_t\gamma_t(\theta_0)}$ is sublinear, which is illustrated by the bottom curve in Fig. 3b. However, this does not imply no-regret if hyperparameters are misspecified as in Fig. 2a, since the first term in Theorem 1 is unbounded in this case. To avoid this, we must increase the scaling factor $h(t)$ to consider larger function classes.

We propose to define a sublinear reference regret $p(t)$, see Fig. 3b, and to scale $h(t)$ to match an estimate of the regret with respect to the current hyperparameters to this reference. As GP-UCB converges, the regret estimate with respect to the current hyperparameters levels off and drops below the reference $p(t)$. In these cases, we increase $h(t)$ to consider larger function classes and explore further. The choice of $p(t)$ thus directly specifies the amount of additional regret one is willing to incur for exploration. Specifically, given a regret estimate $\bar{R}_t(h)$ that depends on the data collected so far and the selected scaling h, we obtain $h(t)$ from matching the reference, $\bar{R}_t(h) = p(t)$, as

$$h^*(t) = \bar{R}_t^{-1}(p(t)), \qquad h(t) = \max(h^*(t),\, h(t-1)). \tag{14}$$

Here we explicitly enforce that $h(t)$ must be an increasing function. In the following, we consider estimators $\bar{R}_t$ that are increasing functions of h, so that (14) can be solved efficiently via a line search.

Whether choosing $h(t)$ according to (14) leads to a sublinear function depends on the regret estimator $\bar{R}_t$. However, it is always possible to upper bound the $h(t)$ obtained from (14) by a fixed sublinear function. This guarantees sublinear regret eventually. In the following, we consider two estimators that upper bound the cumulative regret experienced so far with respect to the hyperparameters suggested by $h(t)$.

Regret bound As a first estimator for the cumulative regret, we consider the regret bound on R_t in (11). We focus on the Gaussian kernel, but the arguments transfer directly to the case of the Matérn kernel. The term $\sqrt{C_1 t\, \beta_t\, I_{\theta_t}(\mathbf{y}_t; f)}$ bounds the regret with respect to the current function class specified by θ_t. In addition to the direct dependence on $b(t)g(t)^d$ in β_t, the regret bound also depends on $g(t)$ implicitly through the mutual information $I_{\theta_t}(\mathbf{y}_t; f)$, where $\theta_t = \theta_0/g(t)$. To make the dependence on $g(t)$ more explicit, we use Theorem 2 and rewrite the mutual information as $(g(t)/g(t-1))^d I_{\theta_{t-1}}(\mathbf{y}_t; f)$ instead. Note that the scaling factor was derived for γ_t, but remains a good indicator of increase in mutual information in practice. With this replacement we use

$$\bar{R}_t(h) = \sqrt{C_1 t\, \beta_t\, (b(t), g(t))\; g(t)^d I_{\theta_{t-1}}(\mathbf{y}_t; f)} \tag{15}$$

to estimate the regret, where the term $\beta_t(b, g)$ is as in Theorem 1, but with the mutual information similarly replaced with the explicit dependence on $g(t)$. Solving (14) together with (15) is computationally efficient, since computing $\bar{R}_t$ does not require inverting the kernel matrix.

One step predictions While (15) is fast to compute, it requires us to know the dependence of $\gamma_t(\theta_t)$ on $h(t)$. Deriving analytic bounds can be infeasible for many kernels. As an alternative, we estimate the regret one-step ahead directly. In particular, if the considered function class is sufficiently large and our confidence intervals hold at all time steps $t > 0$, then the one-step ahead cumulative regret R_{t+1} for our algorithm at iteration t is bounded from above by

$$\bar{R}_t = 2 \sum_{j=1}^{t} \beta_j^{1/2} \sigma_j(\mathbf{x}_{j+1}), \tag{16}$$

where each β_t and σ_t is based on the corresponding hyperparameters θ_t. In Theorem 1, R_{t+1} is further upper-bounded by (11). The regret estimate in (16) depends on $\mathbf{x}_{t+1}$, which is the next input that would be evaluated based on the UCB criterion with GP hyperparameters scaled according to $h(t)$. As the hyperparameters for previous iterations are fixed, the only term that depends on $h(t)$ is the bound on the instantaneous regret, $r_t \leq 2\beta_t \sigma_t(\mathbf{x}_{t+1})$. Unlike (15), (16) is not able to exploit the known dependence of γ_t on $h(t)$, so that it cannot reason about the long-term effects of changing $h(t)$. This means that, empirically, the cumulative regret may overshoot the reference regret, only to settle below it later.

Scaling $h(t)$ according to (16) provides an interesting perspective on the method by Wang and de Freitas (2014). They decrease the kernel lengthscales whenever $\sigma_t(\mathbf{x}_{t+1}) \leq \kappa$. In our framework, this corresponds to $p(t) = \sum_{j=1}^{t} 2\beta_j \max(\kappa, \sigma_j(\mathbf{x}_{j+1})) \geq \kappa t$, which is not sublinear. As a consequence, while they ultimately bound the cumulative regret using the smallest possible lengthscale, the choice for $p(t)$ forces too much exploration to achieve sublinear regret before the lower bound is reached. In contrast, if we choose $p(t)$ to be sublinear,

then the function class grows slowly enough to ensure more careful exploration. This allows us to achieve sublinear regret in the case when a lower bound on the hyperparameters it not known.

4.2. Practical Considerations and Discussion

In this section, we discuss additional practical considerations and show how to combine the theoretical results with online inference of the hyperparameters.

Online inference and exploration strategies The theoretical results presented in the previous sections extend to the case where the initial guess θ_0 of the GP's lengthscale is improved online using any estimator, e.g., with MAP estimation to obtain θ_t^{MAP}. Theoretically, as long as the change in θ_0 is bounded, the cumulative regret increases by at most a constant factor. In practice, this bound can always be enforced by truncating the estimated hyperparameters. Moreover, the scaling induced by online inference can be considered to be part of $g(t)$ according to Lemma 2, in which case the norm bound can be adapted accordingly. In practice, online inference improves performance drastically, as it is often difficult to specify an appropriate relative initial scaling of the lengthscales θ_0.

In more than one dimension, $d > 1$, there are multiple ways that MAP estimation can be combined with the theoretical results of the paper. The simplest one is to enforce an upper bound on the lengthscales based on $g(t)$,

$$\theta_t = \min(\theta_t^{\mathrm{MAP}},\, \theta_0 \,/\, g(t)), \tag{17}$$

where the min is taken elementwise. This choice is similar to the one by Wang et al. (2016). If all entries of θ_0 have the same magnitude, this scaling can be understood as encouraging additional exploration in the smoothest direction of the input space first. This often makes sense, since MAP estimates tend to assume functions that are too smooth, see Fig. 1. However, it can be undesirable in the case when the true function only depends on a subset of the inputs. In these cases, the MAP estimate would correctly eliminate these inputs from the input space by assigning long lengthscales, but the scaling in (17) would encourage additional exploration in these directions first. Note that eventually exploring the entire input space is unavoidable to avoid getting stuck in local optima (Bull, 2011).

An alternative approach is to instead scale down the MAP estimate directly,

$$\theta_t = \theta_t^{\mathrm{MAP}} \,/\, \max(g(t),\, 1). \tag{18}$$

This scaling can be understood as evenly encouraging additional exploration in all directions. While (18) also explores in directions that have been eliminated by the MAP estimate, unlike (17) it simultaneously explores all directions relative to the MAP estimate. From a theoretical point of view, the choice of exploration strategy does not matter, as in the limit as $t \to \infty$ all lengthscales approach zero. In the one-dimensional case, the two strategies are equivalent. Both strategies use the MAP lengthscales for BO in the nominal case, but the $g(t)$ factor eventually scales down the lengthscales further. This ensures that our method only improves on the empirical performance of BO with MAP estimation.

In practice, maximum likelihood estimates for the inputs are often good enough when the underlying function resembles a sample from a GP. Thus, the approach presented in

this paper is most relevant when the underlying function has some 'nonstationarity'. In the literature, other approaches to deal with nonstationarity have been proposed. For example, Snoek et al. (2013) scale the input inputs through a beta function and infer its hyperparameters online. Our approach can easily be combined with any such method, as it works on top of any estimate provided by the underlying inference scheme. Moreover, in high-dimensional spaces one can combine our algorithm with methods to automatically identify a low-dimensional subspace of $\mathcal{D}$ (Djolonga et al., 2013; Wang et al., 2016).

In this paper, we have considered the kernel to be fixed, and only adapted the lengthscales and norm bound. However, often the kernel structure itself is a critical hyperparameter (Duvenaud et al., 2011). The strategy presented in this paper could be used to add rougher kernels over time or, for example, to adapt the ν input of the Matérn kernel, which determines its roughness.

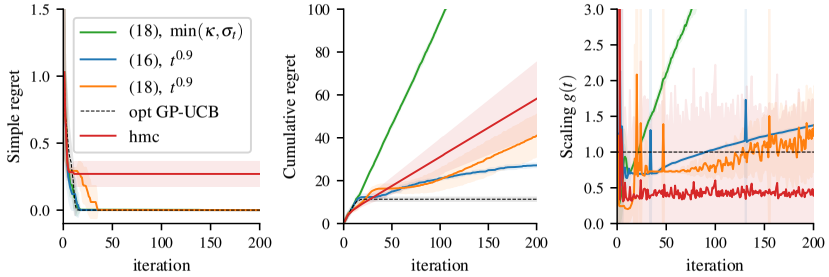
Confidence intervals Empirically, β_t is often set to a constant rather than using the theoretical bounds in Lemma 1, which leads to (point-wise) confidence intervals when f is sampled from a GP model. In particular, typically measurement data is standardized to be zero mean and unit variance and β_t is set to two or three. This often works well in practice, but does not provide any guarantees. However, if one were to believe the resulting confidence bounds, our method can be used to avoid getting stuck in local optima, too. In this case on can set $h(t) = g(t)$ and apply our method as before.

General discussion Knowing how the sample complexity of the underlying BO algorithm depends on the lengthscales also has implications in practice. For example, Wang et al. (2016) and Wabersich and Toussaint (2016) suggest to scale down the lengthscales by a factor of 2 and roughly 1.1, respectively, although not at every iteration. As shown in Sec. 4, this scales the regret bound by a factor of g^d, which quickly grows with the number of dimensions. Exponentiating their factors with $1/d$ is likely to make their approaches more robust when BO is used in high-dimensional input spaces $\mathcal{D}$.

Lastly, in a comparison of multiple BO algorithms (acquisition functions) on a robotic platform, Calandra et al. (2014) conclude that the GP-UCB algorithm shows the best empirical performance for their problem. They use the theoretical version of the algorithm by Srinivas et al. (2012), in which β_t grows with an additional factor of $\mathcal{O}(\sqrt{\log(t^2)})$ relative to Lemma 1. In our framework with the bounds in Lemma 1, this is equivalent to scaling up the initial guess for the RKHS norm bound for f by the same factor at every iteration, which increases the function class considered by the algorithm over time. We conjecture that this increase of the function class over time is probably responsible for pushing the MAP estimate of the lengthscales out of the local minima, which in turn led to better empirical performance.

5. Experiments

In this section, we evaluate our proposed method on several benchmark problems. As baselines, we consider algorithms based on the UCB acquisition function. We specify a strong gamma prior that encourages short lengthscales, and consider both maximum a posteriori (MAP) point-estimates of the hyperparameters and a Hamiltonian Monte Carlo (HMC) approach that samples from the posterior distribution of the hyperparameters and



(a) Simple regret. (b) Cumulative regret. (c) Scaling $g(t)$.

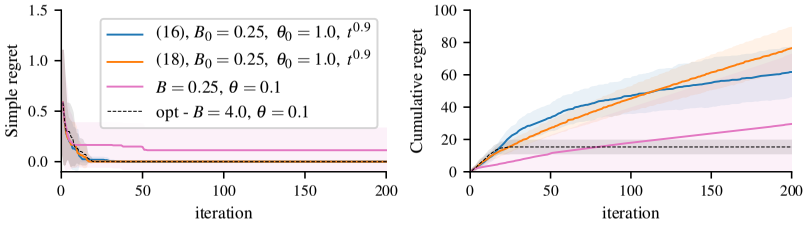
Figure 4: Mean and standard deviation of the empirical simple and cumulative regret over ten different random initializations for the function in Fig. 2. The HMC baseline (red) gets stuck in a local optimum and obtains constant regret in Fig. 4a. GP-UCB with the true hyperparameters (gray dashed) obtains the lowest cumulative regret in Fig. 4b. However, our methods (orange/blue) increase the function class over time, see Fig. 4c, and thus obtain sublinear regret without knowing the true hyperparameters.

marginalizes them out. Unless otherwise specified, the initial lengthscales are set to $\theta_0 = \mathbf{1}$, the initial norm bound is $B_0 = 2$, the confidence bounds hold with probability at least $\delta = 0.9$, and the tradeoff factor between $b(t)$ and $g(t)$ is $\lambda = 0.1$.

We follow several best-practices in BO to ensure a fair comparison with the baselines. We rescale the input space $\mathcal{D}$ to the unit hypercube in order to ensure that both the initial lengthscales and the prior over lengthscales are reasonable for different problems. As is common in practice, the comparison baselines use the empirical confidence intervals suggested in Sec. 4.2, instead of the theoretical bounds in Lemma 1 that are used for our method. Lastly, we initialize all GPs with 2^d measurements that are collected uniformly at random within the domain $\mathcal{D}$. To measure performance, we use the cumulative regret that has been the main focus of this paper. In addition, we evaluate the different methods in terms of simple regret, which is the regret of the best inputs evaluated so far, $\max_{x \in \mathcal{D}} f(x) - \max_{t' <= t} f(\mathbf{x}_{t'})$. This metric is relevant when costs during experiments do not matter and BO is only used to determine high-quality inputs by the end of the optimization procedure.

5.1. Synthetic Experiments

Example function We first evaluate all proposed methods on the example function in Fig. 2, which lives inside the RKHS associated with a Gaussian kernel with $\theta = 0.1$ and has norm $\|f\|_{k_\theta} = 2$. We evaluate our proposed method for the sublinear reference function $p(t) = t^{0.9}$ together with maximum a posteriori hyperparameter estimation. We compare against both GP-UCB with the fixed, correct hyperparameters and HMC hyperparameter estimation. Additionally, we consider a modified variant of the method suggested by Wang and de Freitas (2014), see Sec. 4.1. Rather than scaling the lengthscales by a fixed constant, we conduct a line search to find the smallest possible scaling factor that ren-



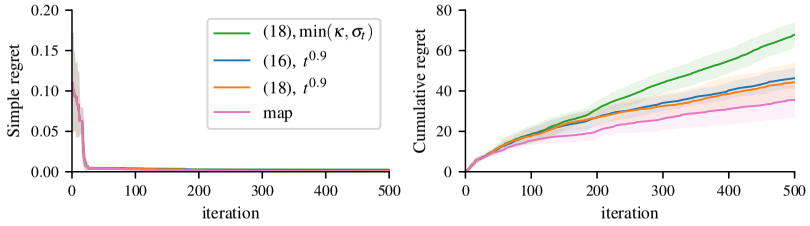
(a) Simple regret. (b) Cumulative regret.

Figure 5: Simple and cumulative regret over 10 random seeds for samples from a GP with bounded RKHS norm. The GP-UCB algorithm with misspecified hyperparameters (magenta) fails to converge given only a wrong choice of B_0. In contrast, our methods (blue/orange) converge even though θ_0 is misspecified in addition.

ders $\sigma_t(\mathbf{x}_{t+1}) \geq \kappa = 0.1$. This is the most conservative variant of the algorithm. Note that we do not know a lower bound on the hyperparameters and therefore do not enforce it.

The results of the experiments are shown in Fig. 4. The simple regret plot in Fig. 4a shows that all methods based on hyperparameter adaptation evaluate close-to-optimal inputs eventually, and do so almost as quickly as GP-UCB based on the true hyperparameters (black, dashed). However, the method based on HMC hyperparameter estimation (red) considers functions that are too smooth and gets stuck in local optima, as in Fig. 2. This can also be seen in Fig. 4c, which plots the effective scaling $g(t)$ based on the combination of Bayesian hyperparameter estimation and hyperparameter adaptation through $h(t)$. The HMC hyperparameters consistenly over-estimate the lengthscales by a factor of roughly two. In contrast, while the MAP estimation leads to the wrong hyperparameters initially, the adaptation methods in (15) and (16) slowly increase the function class until the true lengthscales are found eventually. It can be seen that the one step estimate (16) (orange) is more noisy than the upper bound in (15) (blue).

While all adaptation methods determine good inputs quickly according to the simple regret, they perform differently in terms of the cumulative regret in Fig. 4b. As expected, the HMC method (red line) converges to a local optimum and experiences constant regret increase equal to the simple regret at every time step. The modified method of Wang and de Freitas (2014) (green line) expands the function class too aggressively and also achieves constant regret. Empirically, their method always explores and never repeatedly evaluates close-to-optimal inputs that would decrease cumulative regret. While the method works well in terms of simple regret, without a lower bound on the hyperparameters it never converges to sublinear regret. As expected from Theorem 1, GP-UCB based on the optimal hyperparameters achieves the lowest cumulative regret. Our two methods expand the function class over time, which allows them to converge to close-to-optimal inputs, even though MAP estimation estimates the hyperparameters wrongly initially. While the regret is sublinear, the additional exploration caused by $g(t)$ means that the cumulative regret is larger. This is the additional cost we incur for not knowing the hyperparameters in advance.



(a) Simple regret. (b) Cumulative regret.

Figure 6: Simple and cumulative regret over 5 random seeds for a logistic regression problem. All methods determine close-to-optimal parameters. However, our methods explore more to counteract misspecified hyperparameters.

Samples from a GP As a second experiment, we compare GP-UCB to A-GP-UCB on samples drawn from a GP when the norm bound B_0 is misspecified. Samples from a GP are not contained in the RKHS. To avoid this technical issue, we sample function values from the posterior GP at only a finite number of discrete gridpoints and interpolate between them using the kernel with the correct lengthscales θ. We rescale these functions to have RKHS norm of $B = 4$, but use $B_0 = 0.25$ as an initial guess for both BO algorithms and do not use any hyperparameter estimation. Even though we use the correct kernel lengthscales for GP-UCB, $\theta_0 = \theta = 0.1$, this discrepancy means that the true function is not contained in the initial confidence intervals. As before, for our method we use the reference regret $p(t) = t^{0.9}$ and additionally misspecify the lengthscales, $\theta_0 = 1$.

The results are shown in Fig. 5. GP-UCB with the correct hyperparameters (black, dashed) obtains the lowest cumulative regret. However, it fails to converge when hyperparameters are misspecified (magenta), since the confidence intervals are too small to encourage any exploration. In contrast, our methods (blue/orange) converge to close-to-optimal inputs as in the previous example.

5.2. Logistic Regression Experiment

Lastly, we use our method to tune a logistic regression problem on the MNIST data set (LeCun, 1998). As in the experiment in Klein et al. (2016), we consider four training inputs: the learning rate, the l_2 regularization constant, the batch size, and the dropout rate. We use the validation loss as the optimization objective.

The results are shown in Fig. 6. Even though the input space is fairly high-dimensional with $d = 4$, all algorithms determine close-to-optimal inputs quickly. In particular, MAP estimation determines that both the dropout rate and the batch size do not influence the validation loss significantly. Since the theoretical results in A-GP-UCB are compatible with MAP estimation, our approach achieves the same empirical performance, but has theoretical worst-case regret bounds. After convergence, the BO baselines repeatedly evaluate the same inputs, without gaining any new information. In contrast, our method continues to explore in order to potentially find better inputs. While it does not occur in this case, this allows us

to be more confident that the global optimum has been identified as t increases. For standard BO methods, there is no guarantee of convergence with misspecified hyperparameters.

6. Conclusion and Future Work

We introduced A-GP-UCB, a BO algorithm that is provably no-regret when hyperparameters are unknown. Our method adapts the hyperparameters online, which causes the underlying BO algorithm to consider larger function spaces over time. Eventually, the function space is large enough to contain the true function, so that our algorithm provably converges. We evaluated our method on several benchmark problems, confirming that, on the one hand, it provably converges even in cases where standard BO algorithms get stuck in local optima, and, on the other hand, enjoys competitive performance as standard BO algorithms that do not have theoretical guarantees in this setting.

The main idea behind our analysis is that adapting the hyperparameters increases the cumulative regret bound, but we do so slowly enough to converge eventually. This idea is fairly general and could also be applied to other no-regret algorithms. Another potential future direction is to investigate alternative strategies to select the scaling factors $b(t)$ and $g(t)$ and consider adapting other parameters such as the kernel structure.

Acknowledgments

This research was supported in part by SNSF grant 200020_159557, ERC grant no. 815943, NSERC grant RGPIN-2014-04634, the Vector Institute, and an Open Philantropy Project AI fellowship. We would like to thank Johannes Kirschner for valueable discussions.

References

Yasin Abbasi-Yadkori. *Online learning of linearly parameterized control problems*. PhD thesis, 2012.

Hany Abdelrahman, Felix Berkenkamp, and Andreas Krause. Bayesian optimization for maximum power point tracking in photovoltaic power plants. In *2016 European Control Conference (ECC)*, pages 2078–2083, 2016.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

Felix Berkenkamp, Andreas Krause, and Angela P. Schoellig. Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics. *arXiv:1602.04450 [cs.RO]*, 2016.

Ilija Bogunovic, Jonathan Scarlett, Andreas Krause, and Volkan Cevher. Truncated variance reduction: a unified approach to Bayesian optimization and level-set estimation. In *Advances in Neural Information Processing Systems 29*, pages 1507–1515. Curran Associates, Inc., 2016.

Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv:1012.2599 [cs]*, 2010.

Adam D. Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12(Oct):2879–2904, 2011.

Roberto Calandra, André Seyfarth, Jan Peters, and Marc Peter Deisenroth. An experimental comparison of Bayesian optimization for bipedal locomotion. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1951–1958, 2014.

Sayak Ray Chowdhury and Aditya Gopalan. On kernelized multi-armed bandits. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 844–853. PMLR, 2017.

Andreas Christmann and Ingo Steinwart. *Support Vector Machines*. Information Science and Statistics. Springer, New York, NY, 2008.

Josip Djolonga, Andreas Krause, and Volkan Cevher. High-dimensional Gaussian process bandits. In *Advances in Neural Information Processing Systems 26*, pages 1025–1033, 2013.

Audrey Durand, Odalric-Ambrym Maillard, and Joelle Pineau. Streaming kernel regression with provably adaptive mean, variance, and regularization. *Journal of Machine Learning Research*, 19(17):1–34, 2018.

David K. Duvenaud, Hannes Nickisch, and Carl Edward Rasmussen. Additive Gaussian processes. In *Advances in Neural Information Processing Systems 24*, pages 226–234, 2011.

Peter Frazier, Warren Powell, and Savas Dayanik. The knowledge-gradient policy for correlated normal beliefs. *INFORMS Journal on Computing*, 21(4):599–613, 2009.

Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D. Sculley. Google vizier: a service for black-box optimization. In *Proc. of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 1487–1495, New York, NY, USA, 2017. ACM.

I. S. Gradshteĭn, I. M. Ryzhik, and Alan Jeffrey. *Table of integrals, series, and products*. Academic Press, Amsterdam, Boston, 7th ed edition, 2007.

Philipp Hennig and Christian J. Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13(1):1809–1837, 2012.

José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 918–926. Curran Associates, Inc., 2014.

Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K. Sriperumbudur. Gaussian processes and kernel methods: a review on connections and equivalences. *arXiv:1807.02582 [stat.ML]*, 2018.

Johannes Kirschner and Andreas Krause. Information directed sampling and bandits with heteroscedastic noise. In *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pages 358–384. PMLR, 2018.

Aaron Klein, Stefan Falkner, Jost Tobias Springenberg, and Frank Hutter. Bayesian neural network for predicting learning curves. In *NIPS 2016 Bayesian Neural Network Workshop*, 2016.

Yann LeCun. The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*, 1998.

Daniel J. Lizotte, Tao Wang, Michael H. Bowling, and Dale Schuurmans. Automatic gait optimization with Gaussian process regression. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, volume 7, pages 944–949, 2007.

Jonas Mockus. *Bayesian approach to global optimization: theory and applications*. Springer Science & Business Media, 2012.

Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2:117–129, 1978.

Carl Edward Rasmussen and Christopher K.I Williams. *Gaussian processes for machine learning*. MIT Press, Cambridge MA, 2006.

Binxin Ru, Michael A. Osborne, Mark Mcleod, and Diego Granziol. Fast information-theoretic Bayesian optimisation. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4384–4392. PMLR, 2018.

Daniel Russo and Benjamin Van Roy. Learning to optimize via information-directed sampling. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1583–1591. Curran Associates, Inc., 2014.

M. W. Seeger, S. M. Kakade, and D. P. Foster. Information consistency of nonparametric Gaussian process methods. *IEEE Transactions on Information Theory*, 54(5):2376–2382, 2008.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical Bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2951–2959. Curran Associates, Inc., 2012.

Jasper Snoek, Kevin Swersky, Richard S. Zemel, and Ryan P. Adams. Input warping for Bayesian optimization of non-stationary functions. In *NIPS Workshop on Bayesian Optimization*, 2013.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. *IEEE Transactions on Information Theory*, 58(5):3250–3265, 2012.

V. Vapnik. Principles of risk minimization for learning theory. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 831–838. Morgan-Kaufmann, 1992.

Kim Peter Wabersich and Marc Toussaint. Advancing Bayesian optimization: the mixed-global-local (mgl) kernel and length-scale cool down. *arXiv:1612.03117 [cs, stat]*, 2016.

Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient Bayesian optimization. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3627–3635. PMLR, 2017.

Ziyu Wang and Nando de Freitas. Theoretical analysis of Bayesian optimisation with unknown Gaussian process hyper-parameters. *arXiv:1406.7758 [cs, stat]*, 2014.

Ziyu Wang, Frank Hutter, Masrour Zoghi, David Matheson, and Nando de Freitas. Bayesian optimization in a billion dimensions via random embeddings. *Journal of Artificial Intelligence Research*, 55:361–387, 2016.

Appendix A. Proof of Main Theorem

Lemma 3 *Let $f \in \mathcal{H}_{\theta_{t^*}}$ with $\|f\|_{\theta_{t^*}} \leq B_{t^*}$. Then, for any monotonically increasing functions $g(t) \geq 1$ and $b(t) \geq 1$ and for all $t \geq t^*$: $f \in \mathcal{H}_{\theta_t}$ with $\|f\|_{\theta_t} \leq B_t$*

Proof Lemma 2 together with monotonicity of g yields $\mathcal{H}_{\theta_t} \supseteq \mathcal{H}_{\theta_{t^*}}$ so that $f \in \mathcal{H}_{\theta_t}$ and

$$\|f\|_{\theta_t} \leq \prod_{1 \leq i \leq d} \frac{[\theta_{t^*}]_i}{[\theta_t]_i} \|f\|_{\theta_{t^*}} \leq \frac{g(t)^d}{g(t^*)^d} B_{t^*} = \frac{g(t)^d}{g(t^*)^d} g(t^*)^d b(t^*) B_0 = g(t)^d b(t^*) B_0 \leq B_t$$

■

Lemma 4 *Under the assumptions of Lemma 1, let θ_t be a predictable sequence of kernel hyperparameters such that $\|f\|_{k_{\theta_t}} \leq B_t$ and let the GP predictions μ_t and σ_t use the prior covariance k_{θ_t}. If $\beta_t^{1/2} = B_t + 4\sigma\sqrt{I_{\theta_t}(\mathbf{y}_t; f) + 1 + \ln(1/\delta)}$, then $|\, f(\mathbf{x}) - \mu_t(\mathbf{x}) \,| \leq \beta_t^{1/2} \sigma_t(\mathbf{x})$ holds for all $\mathbf{x} \in D$ and iterations $t \geq 0$ jointly with probability at least $1 - \delta$.*

Proof The proof is the same as the one by Abbasi-Yadkori (2012); Chowdhury and Gopalan (2017), except that the kernel is time-dependent. ■

We are now ready to prove the main result:

Proof [Theorem 1] We split the regret bound into two terms, $R_t = t_0\, r_c + r_s(t)$. In the initial rounds, where either $B_t \leq g(t)^d B_0$ or $\max_i [\theta]_i/[\theta]_0 > 1$, the regret is trivially bounded by $r_t \leq 2\|f\|_\infty \leq 2\|f\|_\theta \leq B$. Thus $r_c \leq 2B$. Let $t_0 \in (0, \infty]$ be the first iteration such that $f \in \mathcal{H}_{\theta_{t_0}}$ with $\|f\|_{\theta_{t_0}} \leq B_{t_0}$. From Lemma 3, we have that $f \in \mathcal{H}_{\theta_t}$ with $\|f\|_{\theta_t} \leq B_t$ for all $t \geq t_0$. Thus we can use Lemma 4 to conclude $|f - \mu_t(\mathbf{x})| \leq \beta_t^{1/2}\sigma_t(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{D}$ and $t \geq t_0$ jointly with probability at least $(1-\delta)$. We use Lemmas 5.2-5.4 in Srinivas et al. (2012) to conclude that the second stage has a regret bound of $r_s^2(t) \leq C_1 \beta_t I(\mathbf{y}_t; f)$, which concludes the proof. ■

Appendix B. Bound on the information capacity γ_t

Theorem 4 (Theorem 8 in Srinivas et al. (2012)) *Suppose that $D \subset \mathbb{R}^d$ is compact, and $k(\mathbf{x}, \mathbf{x}')$ is a covariance function for which the additional assumption of Theorem 2 in Srinivas et al. (2012) hold. Moreover, let $B_k(T_*) = \sum_{s>T_*} \lambda_s$, where $\{\lambda_s\}$ is the operator spectrum of k with respect to the uniform distribution over D. Pick $\tau > 0$, and let $n_T = C_4 T^\tau (\log T)$ with $C_4 = 2\mathcal{V}(D)(2\tau+1)$. Then, the following bound holds true:*

$$\gamma_T \leq \frac{1/2}{1-e^{-1}} \max_{r \in \{1,\ldots,T\}} T_* \log\left(\frac{r n_T}{\sigma^2}\right) + C_4 \sigma^{-2}(1 - \frac{r}{T})(B_k(T_*) T^{\tau+1} + 1) \log T + \mathcal{O}(T^{1-\frac{\tau}{d}}). \tag{19}$$

Theorem 4 allows us to bound γ_t through the operator spectrum of the kernel with respect to the uniform distribution. We now consider this quantity for two specific kernels.

B.1. Bounds for the Squared Exponential Kernel

Lemma 5 *For all $x \in [0, x_{\max}^2]$ it holds that $\log(1+x^2) \geq \frac{\log(1+x_{\max}^2)}{x_{\max}^2} x^2$*

In this section, we use Theorem 4 to obtain concrete bounds for the Gaussian kernel. From Seeger et al. (2008), we obtain a bound on the eigenspectrum that is given by

$$\lambda_s \leq c B^{s^{1/d}}, \text{ where } c = \sqrt{\frac{2a}{A}}, \quad b = \frac{1}{2\theta_t^2}, \quad B = \frac{b}{A}, \quad \text{and} \quad A = a + b + \sqrt{a^2 + 2ab}.$$

The constant $a > 0$ parameterizes the distribution $\mu(\mathbf{x}) \sim \mathcal{N}(\mathbf{0}, (4a)^{-1}\mathbf{I}_d)$. As a consequence of $\theta_t > 0$, we have that $b \geq 0$, $0 < B < 1$, $c > 0$, and $A > 0$. In the following, we bound the eigenspectrum. The steps follow the outline of Seeger et al. (2008), but we provide more details and the dependence on the lengtscales θ_t is made explicit:

$$\begin{aligned} B_k(T_*) &= \sum_{s>T_*} \lambda_s \leq c \sum_{s \geq T_*+1} B^{s^{1/d}} = c \sum_{s \geq T_*+1} \exp\log(B^{s^{1/d}}) = c \sum_{s \geq T_*+1} \exp(s^{1/d} \log B), \\ &= c \sum_{s \geq T_*+1} \exp(-s^{1/d}\alpha) \leq c \int_{T_*}^\infty \exp(-\alpha s^{1/d})\, \mathrm{d}s, \end{aligned}$$

where $\alpha = -\log B$. Now substitute $s = \phi(t) = (t/\alpha)^d$. Then $\mathrm{d}s = \frac{dt^{d-1}}{\alpha}\,\mathrm{d}t$ and

$$B_k(T_*) \leq c \int_{\alpha T_*^{1/d}}^\infty \exp(-t) \frac{dt^{d-1}}{\alpha}\,\mathrm{d}t = cd\alpha^{-d}\Gamma(d, \alpha T_*^{1/d}),$$

where $\Gamma(d,\beta) = \int_\beta^\infty e^{-t} t^{d-1}\,dt = (d-1)! e^{-\beta} \sum_{k=0}^{d-1} \beta^k/k!$ for $d \in \mathbb{N}$ as in Gradshteĭn et al. (2007, (8.352.4)). Then, with $\beta = \alpha T_*^{1/d}$,

$$B_k(T_*) \leq cd\alpha^{-d}(d-1)! e^{-\beta} \sum_{k=0}^{d-1} \beta^k/k! = c(d!)\alpha^{-d} e^{-\beta} \sum_{k=0}^{d-1} (k!)^{-1} \beta^k.$$

Before we bound the information gain, let us determine how α^{-d} and c depend on the lengthscales. In particular, we want to quantify their upper bounds in terms of $g(t)$.

$$\begin{aligned} \alpha^{-d} &= \log^{-d}(1/B) = \log^{-d}\left(2\theta_t^2 A\right) = \log^{-d}\left(1 + 2\theta_t^2 a + 2\theta_t \sqrt{a^2 + \frac{a}{\theta_t^2}}\right) && (20) \\ &\leq \log^{-d}\left(1 + 2\theta_t^2 a\right) \leq \left(\frac{\log(1 + 2\theta_0^2 a)}{2\theta_0^2 a} 2\theta_t^2 a\right)^{-d} \qquad \text{by Lemma 5} && (21) \\ &= \mathcal{O}\left(\theta_t^{-2d}\right) = \mathcal{O}\left(g^{2d}(t)\right), && (22) \end{aligned}$$

where (21) follows from Lemma 5, since $g(t) \geq 1$ for all $t > 0$. Similarly,

$$c = \left(\frac{2a}{a + \frac{1}{2\theta_t^2} + \sqrt{a^2 + \frac{a}{\theta_t^2}}}\right)^{d/2} \leq \left(\frac{2a}{\frac{1}{2\theta_t^2}}\right) = \left(4a\theta_t^2\right)^{d/2} = \mathcal{O}(g(t)^{-d}). \tag{23}$$

As in Srinivas et al. (2012), we choose $T_* = (\log(T n_T)/\alpha)^d$, so that $\beta = \log(T n_T)$ and therefore does not depend on g_t. Plugging into (19), the first term of (19) dominates and

$$\gamma_T = \mathcal{O}\left(\left[\log(T^{d+1}(\log T))\right]^{d+1} c\alpha^{-d}\right)^{d/2} = \mathcal{O}\left((\log T)^{d+1} g(t)^d\right). \tag{24}$$

B.2. Matérn kernel

Following the proof for Theorem 2 in the addendum to Seeger et al. (2008), we have that

$$\lambda_s^{(T)} \leq C(1+\delta) s^{-(2\nu+d)/d} \;\forall s \geq s_0, \tag{25}$$

For the leading constant we have $C = C_3^{(2\nu+d)/d}$ with $\alpha = \frac{2\pi\theta_t}{\sqrt{2\nu}}$. Hiding terms that do not depend on α and therefore $g(t)$, we have

$$\begin{aligned} &C_t(\alpha,\nu) = \frac{\Gamma(\nu + d/2)}{\pi^{d/2}\Gamma(\nu)} \alpha^d = \mathcal{O}(g(t)^{-d}) \quad c_1 = \frac{1}{(2\pi)^d C_t(\alpha,\nu)} = \mathcal{O}(g(t)^d) \\ &C_2 = \frac{\alpha^{-d}}{2^d \pi^{d/2}\Gamma(d/2)} = \mathcal{O}(g(t)^d) \qquad C_3 = C_2 \frac{2\tilde{C}}{d} c_1^{\frac{-d}{2\nu+d}} = \mathcal{O}(g(t)^d g(t)^{\frac{-d^2}{2\nu+d}}) = \mathcal{O}(g(t)^d), \end{aligned}$$

so that $C = \mathcal{O}(g(t)^{2\nu+d})$. The second term in C_3 must be over-approximated as a consequence of the proof strategy. It follows that $B_k(T_*) = \mathcal{O}(g(t)^{2\nu d} T_*^{1-(2\nu+d)/d})$ and, as in Srinivas et al. (2012), that $\gamma_T = \mathcal{O}(T^{\frac{d(d+1)}{2\nu+d(d+1)}} (\log T) g(t)^{2\nu d})$.

Journal of Machine Learning Research 20 (2019) 1-18 Submitted 4/18; Revised 1/19; Published 3/19

Bayesian Combination of Probabilistic Classifiers using Multivariate Normal Mixtures

Gregor Pirš gregor.pirs@fri.uni-lj.si
Erik Štrumbelj erik.strumbelj@fri.uni-lj.si
University of Ljubljana
Faculty of Computer and Information Science
Večna pot 113, 1000 Ljubljana, Slovenia

Editor: Ryan Adams

Abstract

Ensemble methods are a powerful tool, often outperforming individual prediction models. Existing Bayesian ensembles either do not model the correlations between sources, or they are only capable of combining non-probabilistic predictions. We propose a new model, which overcomes these disadvantages. Transforming the probabilistic predictions with the inverse additive logistic transformation allows us to model the correlations with multivariate normal mixtures. We derive an efficient Gibbs sampler for the proposed model and implement a regularization method to make it more robust. We compare our method to related work and the classical linear opinion pool. Empirical evaluation on several toy and real-world data sets, including a case study on air-pollution forecasting, shows that the method outperforms other methods, while being robust and easy to use.

Keywords: correlated classifiers, ensemble, probabilistic models, additive logistic transformation, Bayesian inference

1. Introduction

Classifier combination (or ensemble learning) is an important area of applied machine learning and statistics. The idea behind these methods is that combining several prediction models should achieve better performance. Intuitively, if the models make different mistakes, we can combine them and overcome their individual drawbacks. Zhou (2012) highlights the importance of diversity between individual models and argues that understanding such diversity is an open problem in ensemble learning. A theoretical result on how the generalization error of a combination (weighted average of individual methods) decreases as the correlations between individual errors decrease can be found in Ueda and Nakano (1996). That ensembles achieve better performance in practice proved to be the case, as they are extensively used in various machine learning challenges and competitions to great effect—for example an ensemble method won the KDD Cup 2017 (SIGKDD, 2017). The classifiers in an ensemble can be trained on the same data, or on different data sets. In practice, one might want to combine predictions from various sources, for example, a classifier and expert predictions.

Our work was motivated by Kim and Ghahramani (2012) who proposed a Bayesian approach to combining classifiers—the independent Bayesian classifier combination (IBCC) and its dependent extension, the dependent Bayesian classifier combination (DBCC). Their method is able to combine categorical predictions of classifiers, humans, and other sources, not necessarily trained on the same data. They used Markov networks to model the dependencies between classifiers. One drawback of their method is that it can only be used to combine non-probabilistic (categorical) classifiers. Another drawback is that the method's time complexity grows exponentially in the number of classifiers. Simpson et al. (2013) applied the IBCC to a crowdsourcing problem. They observed that the sampler for IBCC can have difficulties with convergence, and proposed a variational IBCC. Additionally Simpson (2014) presented a simplification of Kim and Ghahramani's IBCC, which led to a more efficient Gibbs sampler for parameter inference. Venanzi et al. (2014) extended the IBCC specifically for crowdsourcing problems. They proposed CommunityBCC, where each source (worker) is assumed to belong to some community with a common confusion matrix. The method is able to model the dependencies implicitly through community membership. The results show that CommunityBCC outperforms IBCC on sparse data sets. Nazábal et al. (2016) extended the IBCC model to combine probabilistic models by modeling the predictions with the Dirichlet distribution, however they do not model correlations explicitly.

Another approach that is closely related to IBCC are supra-Bayesian methods, which rely on Bayes' rule to combine the information gathered from several experts and thus improve the predictions. They assume a multivariate normal (MVN) distribution of the predictions and are therefore able to model correlations. Lindley (1985) presented a method for combining probabilistic predictions of categorical data. The author proposed modeling of log-odds instead of original data to deal with the non-normality. The method also uses a common covariance matrix, independent of true labels. Due to the use of a MVN distribution and the assumption of a common covariance matrix the described method lacks flexibility to model the latent space well.

Hoeting et al. (1999) presented Bayesian model averaging that combines several probabilistic models, weighted by their posterior probability. However this approach assumes that one of the combining models is the true data generating model and that the models represent mutually exclusive situations. These assumptions can lead to severe loss of performance, as Cerquides and De Mántaras (2005) have shown.

Lacoste et al. (2014) proposed the agnostic Bayesian learning of ensembles, where they produced ensembles of predictors based on holdout estimations of their generalization performances. Their ideas are based on the inductive learning paradigm and they use Bayesian treatment to find the posterior probability of each hypothesis being the best. They defined the risk of each hypothesis as its expected loss and examined various priors over the joint risk. The inputs to the model can be probabilistic and they account for correlations between inputs. Agnostic Bayes differs from IBCC, its extensions, and supra-Bayesian methods, as it focuses on finding the best performing models and weighting them accordingly, as opposed to relying on finding a latent structure of the predictions.

We propose a new model based on the IBCC that is able to combine probabilistic predictions by learning the latent structure of the sources. We extend the IBCC to probabilistic predictions with a change of the model's distribution for the predictions. First we transform the predictions with the inverse of the additive logistic transformation and then model them with MVN mixtures to account for correlation explicitly. The correlation matrices grow quadratically in the number of classifiers and the number of classes. We construct

©2019 Gregor Pirš and Erik Štrumbelj.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-241.html.

the model with a fully Bayesian framework and use Gibbs sampling (Casella and George, 1992) for parameter inference. Since the complexity of the model grows with the number of classifiers, we implement a regularization method, which makes the model more robust. Our method can also be viewed as an extension of the supra-Bayesian method, where we use mixtures of normals as the likelihood and condition the means and covariance matrices on the true label—increasing the method's flexibility.

We empirically evaluated our method on several toy and real-world data sets, and compared it to related methods. The results on toy data sets highlight that none of the ensembles works well for all data sets. Overall, our method compares favourably to related methods. Additionally, we provide a case study on combining predictions of air-pollutant concentration in Slovenia, where we show that our method is well suited for combining machine learning models with human expert predictions.

The paper is organized as follows. In Section 2 we formulate the problem, present our main methodological contribution, and provide a description of related work. We present the data sets, empirical results, and a case study in Section 3. We discuss the results and provide directions for future work in Section 4.

2. Methods

Let $\{t_i\}_{i=1}^{n}$ be our data set of n observations that can take one of m different values $t_i \in \{1, \ldots, m\}$. Additionally, we have r sources of probabilistic predictions and, for each source, a probabilistic prediction for each observation $\{p_i^{(1)}, \ldots, p_i^{(r)}\}_{i=1}^{n}$, where $p_i^{(k)} = \begin{bmatrix} p_{i1}^{(k)} & \cdots & p_{im}^{(k)} \end{bmatrix}^T$ is the $k-$th source's prediction for the $i-$th observation.

The task is to learn how to combine individual sources into more accurate probabilistic predictions, so that we are able to produce probabilistic predictions $\{\hat{p}_i\}_{i=1}^{n+n^*}$ for n observed and n^* future/unobserved data, represented with categorical random variables $\{T_i\}_{i=n+1}^{n+n^*}$, for which only the sources' predictions $\{p_i^{(1)}, \ldots, p_i^{(r)}\}_{i=n+1}^{n+n^*}$ are known.

2.1. MVN Mixture Conditional Likelihood Model (MM)

Aitchison (1982) proposed the modeling of correlated data on a simplex with the logit-normal distribution. The data on a simplex are considered as data drawn from a MVN distribution, transformed by the additive logistic transformation. This transformation transforms a vector $x \in \mathbb{R}^z$ into a vector $f(x) \in S^{z+1}$, where S^{z+1} represents a $(z+1)$-dimensional simplex. To model the correlations between probabilistic predictions we first transform each source's prediction with the inverse of the additive logistic transformation. Applying it to $\{p_i^{(1)}, \ldots, p_i^{(r)}\}_{i=1}^{n+n^*}$, we get $\{u_i^{(1)}, \ldots, u_i^{(r)}\}_{i=1}^{n+n^*}$, where $u_i^{(k)} = \begin{bmatrix} u_{i1}^{(k)} & \cdots & u_{i(m-1)}^{(k)} \end{bmatrix}^T$. Let $u_i = \begin{bmatrix} u_i^{(1)T} & \cdots & u_i^{(r)T} \end{bmatrix}$, be the concatenated vector of $u_i^{(k)}$s, where $k = 1, ..., r$. The dimension of u_i is then $r(m-1)$, the number of sources times the number of classes minus one. Then the vectors u_i, can be modeled by a MVN distribution, as described above. However, the transformation does not guarantee a MVN distribution of u_i and in practice multi-modal distributions can often arise. Therefore we model the transformed data with MVN mixtures of dimension $r(m-1)$.

Our generative model can be described as follows. The probabilistic predictions $p_i^{(k)} = \begin{bmatrix} p_{i1}^{(k)} & \cdots & p_{im}^{(k)} \end{bmatrix}^T$ are generated by applying the additive logistic transformation to realizations of MVN mixtures, whose parameters depend on the true label. The probability for a new observation to be in true label j is then proportional to the density of the corresponding MVN mixture. The true labels $\{t_i\}_{i=1}^{n}$ and $\{T_i\}_{i=n+1}^{n+n^*}$ are assumed to be generated by a categorical distribution with parameter ρ, and we set a Dirichlet prior on ρ, same as Nazábal et al. (2016). The mixture memberships g_i are assumed to be generated by a categorical distribution of dimension d. For the parameters of MVN distributions we set semi-conjugate MVN and inverse-Wishart priors. We write the likelihood and the priors as

$$u_i | t_i = j, g_i = h, \mu, \Sigma \sim \mathcal{N}_{r(m-1)}(\mu_{jh}, \Sigma_{jh}), \tag{1}$$
$$t_i | \rho \sim \text{Cat}(\rho), \tag{2}$$
$$\rho | \gamma_0 \sim \text{Dir}(\gamma_0), \tag{3}$$
$$g_i \sim \text{Cat}(\tau_0), \tag{4}$$
$$\mu_{jh} \sim \mathcal{N}_{r(m-1)}(\mu_0, \Sigma_0), \tag{5}$$
$$\Sigma_{jh} \sim \text{inv-Wishart}(\nu_0, S_0^{-1}), \tag{6}$$

where $\mu = \{\mu_{jh} : j = 1, ..., m \wedge h = 1, ..., d\}$, $\Sigma = \{\Sigma_{jh} : j = 1, ..., m \wedge h = 1, ..., d\}$. Figure 1 shows the proposed model in plate notation. Let vectors $u_i = \begin{bmatrix} u_i^{(1)T} & \cdots & u_i^{(r)T} \end{bmatrix}$ form the rows of matrix U. Our goal is to sample from

$$\int p(T, \rho, g, \mu, \Sigma | U, t) d\rho = p(T, g, \mu, \Sigma | U, t). \tag{7}$$

To simplify the sampling we marginalize over ρ. Therefore, we need to sample T, μ, Σ, and g. In the remainder of the section we derive full-conditional distributions for these variables and construct a Gibbs sampler. We infer the parameters μ, Σ, and g only on the data where the true label is known, however, the method would also allow inference over unlabelled data.

First we observe that full-conditional distributions of μ and Σ are conditionally independent of ρ and proportional to the product of likelihood and the respective prior. Let $U^{(jh)}$ be the matrix of observations where the true label is j and that belong to the $h-$th mixture. Using the standard formulas for semi-conjugate priors and using Eq. (1), (5), and (6), we get the following full-conditionals

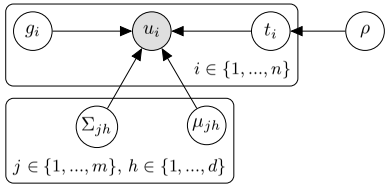


Figure 1: Bayesian plate notation for the proposed model. The transformed predictions u_i for the $i-$th observation are assumed to be generated conditional on the true label $t_i = j$ from the corresponding MVN distribution with parameters μ_{jh} and Σ_{jh}, which also depend on the latent group $g_i = h$. The true label is distributed categorically with parameter ρ. Priors are omitted for brevity.

$$\mu_{jh}|U, t, \rho, g, \Sigma \sim \mathcal{N}_{r(m-1)}(\mu^*, \Sigma^*), \tag{8}$$

$$\Sigma_{jh}|U, t, \rho, g, \mu \sim \text{inv-Wishart}(\nu^*, S^{*-1}), \tag{9}$$

$$\begin{aligned}
\mu^* &= \Sigma^*(\Sigma_0^{-1}\mu_0 + n_{jh}\Sigma_{jh}^{-1}\bar{u}^{(jh)}),\\
\Sigma^* &= (\Sigma_0^{-1} + n_{jh}\Sigma_{jh}^{-1})^{-1},\\
\nu^* &= \nu_0 + n_{jh},\\
S^* &= S_0 + \sum_{l=1}^{n_{jh}} (u_l^{(jh)} - \mu_{jh})(u_l^{(jh)} - \mu_{jh})^T,
\end{aligned}$$

where n_{jh} is the number of true labels equal to j in group h and $\bar{u}^{(jh)}$ are the column means of $U^{(jh)}$.

The mixture membership variable g is also conditionally independent of ρ. Using Eq. (1) and (4), the full-conditional of g is

$$\begin{aligned}
p(g|U, t, \rho, \mu, \Sigma) &\propto p(U|g, t, \rho, \mu, \Sigma)p(g, t, \rho, \mu, \Sigma)\\
&\propto \Big(\prod_{i=1}^{n} p(u_i|g_i, t_i, \rho, \mu, \Sigma)\Big)p(g)\\
&\propto \prod_{i=1}^{n} p(u_i|g_i, t_i, \mu, \Sigma)p(g_i).
\end{aligned} \tag{10}$$

For the predictive distribution of a new observation T_i, we do not have samples of g_i. Therefore we need to marginalize over g_i. Using Eq. (1), (2), and (3), the full-conditional for T is

$$\begin{aligned}
p(T_i = j|U, t, \rho, g, \mu, \Sigma) &\propto \int \sum_{l=1}^{d} p(T_i = j, g_i = l, U, t, \rho, g, \mu, \Sigma)d\rho\\
&\propto \int \sum_{l=1}^{d} p(u_i|T_i = j, g_i = l, t, \rho, g, \mu, \Sigma)p(T_i = j, g_i = l, t, \rho, g, \mu, \Sigma)d\rho\\
&\propto \int \sum_{l=1}^{d} p(u_i|\mu_{jl}, \Sigma_{jl})p(g_i = l|T_i = j, g)p(T_i = j, t, \rho)d\rho\\
&\propto \sum_{l=1}^{d} p(u_i|\mu_{jl}, \Sigma_{jl})p(g_i = l|T_i = j, g) \int p(T_i = j, \rho)p(t|\rho)p(\rho)d\rho,
\end{aligned} \tag{11}$$

where we integrate over all $\rho \in S^m$. The integral in Eq. (11) can be solved by observing that the expression is the multivariate beta function. Some algebra with gamma functions leads to $(\gamma_{0j} + n_j)$, where $n_j = \sum_{l=1}^{n} \mathrm{I}(t_l = j)$. Additionally let $n_{g_{jl}} = p(g_i = l|T_i = j, g) = |\{g_s = l \wedge t_s = j : s = 1, \ldots, n\}|/|\{t_s = j : s = 1, \ldots, n\}|$, which represents the probability of the new observation being in a specific mixture. Inserting this into Eq. (11) and using Eq. (8), (9), and (10) we construct a Gibbs sampler for Eq. (7)

$$\begin{aligned}
\mu_{jh}|U, t, \rho, g, \Sigma &\sim \mathcal{N}_{r(m-1)}(\mu^*, \Sigma^*),\\
\Sigma_{jh}|U, t, \rho, g, \mu &\sim \text{inv-Wishart}(\nu^*, S^{*-1}),\\
p(g_i = h|U, t_i = j, \rho, \mu, \Sigma) &\propto p(u_i|\mu_{jh}, \Sigma_{jh})\tau_{0h},\\
p(T_i = j|U, t, \rho, g, \mu, \Sigma) &\propto \sum_{l=1}^{d} n_{g_{jl}} p(u_i|\mu_{jl}, \Sigma_{jl})(\gamma_{0j} + n_j),
\end{aligned}$$

where we use the conditional independence of group memberships in Eq. (10) and sample each g_i separately. Due to an efficient group collapsing property, the number of mixture groups d can be set to some arbitrary high number and the model finds the suitable number automatically. Some groups tend to be closer together at the beginning. This causes the data points to interchange between groups and pairs of groups often merge into one, resulting in fewer mixture components.

If we constrain the covariance matrices Σ to be diagonal, the model is still able to find mixture-components where the correlations are relatively low. This way the correlations are still modelled, while the method becomes less complex. Therefore, we can exchange some flexibility for simplicity, which results in faster inference (inverse of the covariance matrices becomes trivial), while still remaining flexible enough to model the correlations well in most situations. We included this method in the empirical evaluation (MM-diag).

2.1.1. Regularization

To make the method more robust, we implemented regularization by discounting individual dimensions in transformed observation space U. Let λ^* be a vector of length $r(m-1)$ and

$\lambda_i^* \geq 0$. We modify Eq. (11) by changing $p(u_i|\mu_{jl}, \Sigma_{jl})$ to $p(u_i|\mu_{jl}, \Sigma_{jl} + \text{diag}(\lambda^*))$. Adding a positive number to an element on the diagonal increases the variance of the density, reducing the variance of the data in that dimension (and their covariance with all other dimensions). This effectively decreases the differences in density (between observations) in that dimension. That is, it reduces the influence of that dimension on the distribution of T_i.

For each Gibbs iteration, the λ^* for that iteration is determined before predicting for the unobserved data T using

$$p(\lambda|U, t, \rho, g, \mu, \Sigma) \propto p(t_i = j_i|U, t, \rho, g, \mu, \Sigma, \lambda)p(\lambda),$$

where j_i is the true class label for example i.
In general, we could make our predictions by integrating over λ. However, the posterior distribution for λ is intractable and in practice typically very difficult to sample from efficiently. For the purposes of this work, we use only the MAP estimate

$$\lambda^* = \arg\max_{\lambda} \log p(\lambda) + \sum_{i=1}^{n} \log p(t_i = j_i|U, t, \rho, g, \mu, \Sigma, \lambda).$$

2.2. Related Work

To empirically evaluate our method, we also implemented four Bayesian approaches to combining classifiers and the classical linear opinion pool.

2.2.1. Supra-Bayesian

Lindley (1985) presented the supra-Bayesian method for combining probabilistic predictions of a categorical variable of r experts. The method relies on Bayes' theorem to calculate the posterior distribution of predictions of the decision maker, given the predictions of the experts. Let $A_1, A_2, ..., A_m$ be the possible outcomes of the categorical variable and let $S_1, S_2, ..., S_r$ be classifiers. Let H be the decision maker's prior probability for the response and $q_{ij} = \log(\Pr_i(A_j))$ the log-probability assigned to true label A_j by the $i-$th classifier. Then the decision maker updates his probabilities via the Bayes' theorem

$$p(A_j|Q, H) \propto p(Q|A_j, H)p(A_j|H). \tag{12}$$

To model the correlations between given predictions, Lindley proposes a MVN distribution for $p(Q|A, H)$. Additionally, Lindley argues that the classifiers' belief in their prediction is independent of the label of their prediction, therefore the MVN distributions share a common covariance matrix between predictions for different labels. The author also proposes the modeling of log-odds instead of log-probabilities, as log-odds are expected to be distributed normally. We implemented a ML estimation of the multivariate parameters for Eq. (12), where the matrix Q represented log-odds.

Note that the described method is the one proposed by Lindley (1985) and that supra-Bayesian methods are a very general term for combination methods that rely on the Bayes' theorem. By selecting the appropriate likelihoods we arrive at IBCC and (unregularized) MM as special cases.

2.2.2. IBCC

IBCC (Kim and Ghahramani, 2012) assumes that the true labels t are generated by a categorical distribution with parameter κ. The prediction $c_i^{(k)}$ of individual classifier k is also generated by a categorical distribution with parameters $\pi_j^{(k)}$ for each possible value j of the true label. Let ν be the prior for κ and $\alpha_{0,j}^{(k)}$ prior for $\pi_j^{(k)}$. Kim and Ghahramani additionally put an exponential prior on $\alpha_{0,j}^{(k)}$, but Simpson (2014) observed that the model without this additional flexibility performs comparably well and derived the Gibbs sampler for κ, Π, and the true labels

$$p(\kappa|t, \nu_0) = \frac{1}{B(\nu)} \prod_{j=1}^{m} \kappa_j^{\nu_j},$$

$$p(\pi_j^{(k)}|t, c, \alpha_{0,j}^{(k)}) = \frac{1}{B(\alpha_{t_i}^{(k)})} \prod_{l=1}^{m} \left(\pi_{j,l}^{(k)}\right)^{\alpha_{j,l}^{(k)}},$$

$$p(t_i = j|\Pi, \kappa, c) \propto \kappa_j \prod_{k=1}^{r} \pi_{j,c_i^{(k)}}^{(k)},$$

where $\nu_j = \nu_{0,j} + \sum_{i=1}^{n+n^*} \delta(t_i - j)$ and $\alpha_{j,l}^{(k)} = \alpha_{0,j,l}^{(k)} + \sum_{i=1}^{n+n^*} \delta(t_i - j)\delta(c_i^{(k)} - l)$.

2.2.3. IBCC QPI

The IBCC method can be extended to allow for quasi-probabilistic inputs (QPI). Probabilistic classifiers provide us with probabilities of each outcome $p_i^{(k)} = \begin{bmatrix} p_{i1}^{(k)} & \cdots & p_{im}^{(k)} \end{bmatrix}^T$. For the IBCC to exploit probabilistic predictions, we can first take a sample from a categorical distribution with parameter $p_i^{(k)}$ for each source k. The categorical samples can then be represented as binary vectors, which are of appropriate form for the inputs of IBCC. We then take n_{tsamp} such samples and use them to create a new training set of size $n \times n_{\text{tsamp}}$.

For prediction, we use the same procedure. We sample according to the sources' probabilistic predictions. We retrieve the final prediction as the average probabilistic prediction over all samples. This allows the IBCC to take advantage of probabilistic predictions, instead of simply using the class with the highest probability, as the sources' predictions.

2.2.4. Agnostic Bayes

The agnostic Bayes approach (Lacoste et al., 2014) combines sources based on the probability of being the *best* source, where best is defined in terms of some task-dependent measure. The combined prediction is defined as

$$\hat{p}_i = \sum_{k=1}^{r} P(k^\star = k|p, t)p_i^{(k)},$$

where $P(k^\star = k|p, t)$ is the probability of source k being the best source, given the available data (all predictions p and observed values t). In our case of categorical target variables and

log-score performance measure, we can simplify $P(k^\star = k|p,t) = P(\forall k' \in 1..r : E[l^{(k)}] \geq E[l^{(k')}]|l) = P(k|l)$, where $E[l^{(k)}]$ is the $k-$th source's expected log-score and l all the observed log-scores (for this model, a sufficient statistic for p and t).

We draw samples from $P(k|l)$ with bootstrap inference (this was shown to be very effective in Lacoste et al. 2014, at least as effective as placing priors over joint risk and performing Bayesian inference). That is, we draw a replacement set of observations and compute the average log-score. This gives us the best source on this replacement set—one sample from $P(k|l)$. The process is repeated for several iterations to obtain a set of samples that approximate $P(k|l)$.

2.2.5. Linear Opinion Pool (LOP)

The linear opinion pool (Cooke et al., 1991) is the classical approach to combining predictions. The combined prediction $\hat{p}_i$ is expressed as a linear combination of individual sources

$$\hat{p}_i = X_i \beta,$$

where $X_i = \begin{bmatrix} p_i^{(r)} & p_i^{(r-1)} & \cdots & p_i^{(1)} & p_{\cdot}^{(0)} \end{bmatrix}$ and $\beta = \begin{bmatrix} \beta_r & \beta_{r-1} & \cdots & \beta_1 & \beta_0 \end{bmatrix}^T$.

The parameters β_i and $p_{\cdot}^{(0)}$ were fit by maximizing the likelihood, subject to $0 \leq \beta_i \leq 1$, $\sum_{i=0}^{r} \beta_i = 1$, $0 \leq p_{ij}^{(0)} \leq 1$, and $\sum_{j=1}^{m} p_{\cdot j}^{(0)} = 1$.

We also included a combination of LOP and MM, where the probabilistic predictions obtained by MM are added to the sources before LOP is applied (LOP+MM).

2.2.6. Discussion of Related Work

We can divide the methods for classifier combination into two groups. Methods that aim to estimate the performance of each classifier and then weigh their predictions by their performance (LOP, agnostic-Bayes, etc.). And methods that aim to learn the latent structure of the classifications and then provide probabilistic classifications (IBCC and extensions, supra-Bayesian, etc.).

The former are expected to perform poorly when there is a latent structure to learn, for example, biased classifiers or systematic errors in classifiers. On the other hand, these methods have an advantage that they are not expected to perform discernibly worse than the best source. The latter are the exact opposite—they can learn complex latent structure and perform better than the first group, but can also perform extremely poorly, discernibly worse than the best source. In Section 3 we present 5 toy data sets, which serve to show the differences in performance of these groups, depending on the structure of the data.

The main difference between IBCC and our method (MM) lies in the conditional likelihood for the true label. IBCC models sources' predictions with a multinomial distribution, while MM assumes a latent space of transformed predictions and models it with multivariate normal mixtures. The advantage of this is that it allows us to model both probabilistic predictions and correlations between sources, while the computational complexity is quadratic in the number of sources and outcomes (as opposed to DBCC, where it is exponential).

IBCC QPI is similar to MM in that it is able to exploit probabilistic nature of predictions, however it does not model correlations.

Nazábal et al. (2016) used a Dirichlet distribution to model probabilistic outputs of the sources. A similar (but not equivalent) model would result if we used diagonal covariances without mixtures in the proposed MM. Therefore the proposed MM improves on the drawbacks of Nazábal et al. as it allows for the modeling of correlations between sources, and additionally adds flexibility through the use of mixtures.

MM could also be interpreted as a regularized supra-Bayesian method with MVN-mixture likelihood. The modeling of log-odds described by Lindley (1985) is equivalent to the modeling of the transformed classifiers by MM. A major difference between the methods is that MM does not assume the same covariances over all true labels. Furthermore, each mixture component has its own covariance matrix, which further improves the flexibility of our method.

3. Empirical Evaluation

We empirically evaluated and compared the methods on several toy and real-world data sets. We estimated out-of-sample log-score using train-test splits. To compare the best-performing method with the rest, we used the differences between log-scores. Let a be the vector of length n_t of log-scores of the best performing method, b the vector of log-scores of another method, and $d = a - b$. If

$$\left| \frac{1}{n_t} \sum_{i=1}^{n_t} d_i \right| > 2 \sqrt{\frac{\text{VAR}[d]}{n_t}},$$

we argue that there is a discernible difference in performance. Details on how the data were generated and split are provided below.

For IBCC and its extension, we selected priors that assume each class has the same prior probability. We used the same priors ($\alpha_{0,j} = 1$) for all confusion matrices, which represents a weak belief that the models are random. For MM we selected vague priors that put prior belief that the mean values of transformed predictions are zero with large variances. However, MM has proved to be somewhat robust to the prior selection. We set the same priors for Bayesian methods over all data sets. For λ, we use uniform priors $\lambda_i \sim_{\text{iid}} \text{U}(0, 10^5)$. We set the maximum number of mixture components for MM to 15, however this number dynamically falls, depending on the problem, as we described in Section 2.1.

3.1. Data Sets

In this section we describe the data sets we used for empirical evaluation.

3.1.1. Toy Data Sets

Each of the 5 toy data sets is a combination of the same underlying data set and two sources of probabilistic predictions from a set of sources with different properties. The purpose of the toy data sets is to illustrate some of the shortcomings of the methods for combining categorical predictions.

The data set has a target variable with three possible outcomes. The target variable's value is generated for each sample i separately by first drawing the underlying probability vector p_i^* from Dirichlet$(1.5, 0.9, 0.6)$ and then drawing the target value from Categorical(p_i^*). We generated 500 samples for training and 5000 samples for testing.

We then generated the following sources of probabilities, based on the underlying probability vectors p^*:

- **Random 1 & 2:** The probabilities are drawn independently for each sample from Dirichlet$(1, 1, 1)$.
- **Good 1 & 2:** The probabilities are drawn independently for each sample from Dirichlet$(10p_i^*)$.
- **Permuted 1:** The probabilities are drawn independently for each sample from Dirichlet$(100p_i^*)$ and permuted (2, 3, 1). This results in a source that gives very accurate probabilities, but does not correctly assign them to the possible values of the target variable.
- **Permuted 2:** Identical to Good 2, but permuted (2, 3, 1).

The two sources, Source 1 and Source 2, were assigned as follows: **toy A** (Random 1, Random 2), **toy B** (Good 1, Random 1), **toy C** (Good 1, Good 2), **toy D** (Good 1, Permuted 1), and **toy E** (Good 1, Permuted 2).

3.1.2. Bookies

Each sample in this data set is a football game and the four sources of probabilistic predictions are four major online bookmakers. Probabilistic predictions were calculated by normalizing the reciprocals of odds offered for the outcome of the game (home, draw, away). Therefore, the target variable has 3 outcomes. The data set has 5434 samples, we used 2000 for training and 3434 for testing. Bookmakers are very good sources of probabilistic predictions (Forrest et al., 2005) and their predictions are highly correlated—the lowest correlation is 0.92. We do not expect any ensemble method to outperform the best source.

3.1.3. DNA

Two data sets were constructed using the StatLog DNA data set in a way similar to the experiments in Kim and Ghahramani (2012).

DNA A: 2000 samples were partitioned into 5 partitions, 400 samples each, and a C4.5 decision tree classifier was trained on each partition. These classifiers were used to classify the remaining 1186 samples, 400 of which were used for training and 786 for testing.

DNA B: 2000 samples were used to train 5 different classifiers: a multinomial logistic regression (Source 1), linear discriminant analysis (Source 2), a decision tree classifier (Source 3), a random forest classifier (Source 4), and a k-nearest-neighbour classifier (Source 5, k = 50). These classifiers were used to classify the remaining 1186 samples, 400 of which were used for training and 786 for testing.

3.2. Results

The results are shown in Table 1. No method clearly outperforms other methods over all data sets. LOP and MM perform well on most data sets. For LOP the exceptions are data sets with a more complex relationship between the sources' predictions and the outcome (toy D). And for MM the exceptions are the two simple toy data sets with a continuous relationship between the sources' predictions and the probability of the outcome (toy B, toy C). MM-diag performs worse than the MM on all data sets, however, the differences are small on average. A combination of LOP and MM nullifies the drawbacks of each individual method and performs the best on average. Toy D, with systematic errors in otherwise good sources, is a good example of a data set where the methods that rely on finding the best source are inferior to models that learn the latent structure, providing a good argument for the latter.

Agnostic Bayes will assign a lot of weight to the best source, unless it is difficult to discern the best source. The latter is in cases where there is not a lot of data, relative to the magnitude of the differences between the sources in terms of performance (toy A, DNA A). Therefore, as expected, it performs well when the most accurate source performs well (toy B, toy C, toy E, bookies), and performs poorly, when it does not (DNA B). Note that there is no correlation between the true labels and sources in the toy A data set—the best possible model would assign the same probabilities to all outcomes. Agnostic Bayes performs poorly because it tries to identify the best performing model, but none of the sources perform well. The same would happen to LOP if we did not include an intercept term.

IBCC performs poorly on the toy data sets but performs well on the DNA data sets. However, it is strictly outperformed by MM. The quasi-probabilistic IBCC outperforms the IBCC on DNA data sets and performs similarly on most of the toy data sets. Furthermore, it performs well on the toy A data set, where it is the best performing model, with the proposed MM showing a similar level of performance. IBCC QPI therefore proved to be better suited for probabilistic input than the IBCC, as expected. The supra-Bayesian approach of Lindley (1985), which can be viewed as a very simple case of the proposed approach, performs the worst on average, among all the Bayesian approaches.

As expected, all methods fail to outperform the best source on the bookies data set, which has very accurate and highly correlated sources. This relates heavily to the theoretical results mentioned in the introduction—the data set lacks diversity to exploit. Methods that do not model correlations perform exceptionally poorly. However, methods that weigh/select sources perform at least as good as the best source.

toy A	toy B	toy C	toy D	toy E	bookies	DNA A	DNA B
IBCC QPI	agnostic	LOP	MM	LOP+MM	Source 1	LOP+MM	LOP+MM
-1.03 ± 0.005	-0.831 ± 0.009	-0.81 ± 0.008	-0.001 ± 0	-0.821 ± 0.008	-0.965 ± 0.008	-0.197 ± 0.022	-0.129 ± 0.017
MM	Source 1	LOP+MM	MM-diag	agnostic	agnostic	MM	MM
-1.031 ± 0.005	-0.831 ± 0.009	-0.811 ± 0.008	-0.002 ± 0	-0.826 ± 0.009	-0.966 ± 0.008	-0.206 ± 0.025	-0.132 ± 0.028
MM-diag	LOP	agnostic	LOP+MM	Source 1	LOP+MM	LOP	IBCC QPI
-1.032 ± 0.006	-0.831 ± 0.009	-0.822 ± 0.009	-0.007 ± 0	-0.826 ± 0.009	-0.967 ± 0.008	-0.22 ± 0.026	-0.134 ± 0.023
LOP	LOP+MM	Source 1	IBCC	MM	LOP	agnostic	MM-diag
-1.032 ± 0.005	-0.831 ± 0.009	-0.826 ± 0.009	-0.018 ± 0	-0.83 ± 0.007	-0.967 ± 0.007	-0.22 ± 0.02	-0.139 ± 0.027
LOP+MM	Supra	MM	Supra	LOP	Source 3	MM-diag	LOP
-1.032 ± 0.006	-0.892 ± 0.008	-0.828 ± 0.007	-0.111 ± 0.001	-0.83 ± 0.008	-0.968 ± 0.007	-0.222 ± 0.03	-0.157 ± 0.016
IBCC	MM	Source 2	IBCC QPI	MM-diag	MM	IBCC QPI	IBCC
-1.043 ± 0.006	-0.893 ± 0.006	-0.832 ± 0.009	-0.204 ± 0.001	-0.838 ± 0.007	-0.968 ± 0.008	-0.248 ± 0.033	-0.181 ± 0.033
Supra	MM-diag	MM-diag	agnostic	Supra	Source 4	IBCC	agnostic
-1.107 ± 0.002	-0.906 ± 0.006	-0.839 ± 0.008	-0.824 ± 0.009	-0.883 ± 0.007	-0.968 ± 0.007	-0.271 ± 0.037	-0.182 ± 0.033
agnostic	IBCC	Supra	Source 1	IBCC QPI	Source 2	Supra	Source 2
-1.259 ± 0.009	-0.921 ± 0.008	-0.883 ± 0.007	-0.824 ± 0.009	-0.908 ± 0.005	-0.969 ± 0.007	-0.275 ± 0.017	-0.182 ± 0.033
Source 2	IBCC QPI	IBCC QPI	LOP	IBCC	MM-diag	Source 4	Supra
-1.469 ± 0.015	-0.953 ± 0.005	-0.907 ± 0.005	-0.835 ± 0.008	-0.92 ± 0.012	-0.969 ± 0.008	-0.461 ± 0.055	-0.214 ± 0.013
Source 1	Source 2	IBCC	Source 2	Source 2	IBCC QPI	Source 5	Source 4
-1.508 ± 0.016	-1.469 ± 0.015	-0.919 ± 0.012	-3.081 ± 0.006	-1.43 ± 0.011	-0.993 ± 0.006	-0.767 ± 0.074	-0.311 ± 0.011
					Supra	Source 3	Source 3
					-1.008 ± 0.007	-0.784 ± 0.075	-0.321 ± 0.046
					IBCC	Source 2	Source 5
					-1.245 ± 0.02	-0.846 ± 0.078	-0.587 ± 0.01
						Source 1	Source 1
						-0.879 ± 0.079	-4.305 ± 0.717

Table 1: Estimated log-scores and standard errors on toy and real-world data sets. For each data set, the methods are ordered in descending order of performance. Highlighted methods are not discernibly worse than the best-performing method for that data set.

3.3. Case Study—Prediction of Air-Pollutant Concentration

Air pollution poses a great risk for public health (World Health Organization, 2014). Forecasting excessive air pollution is an important task, also regulated by a EU directive (European Council, 2008). In Slovenia, the Slovenian Environment Agency (ARSO) is tasked with forecasting daily average concentration of particulate matter (PM_{10}) and daily maximum concentration of tropospheric ozone (O_3) for the same day (in the morning) and the next day. Currently, models and human experts are used for this task. Faganeli Pucer et al. (2018) proposed a Bayesian methodology for this problem and empirically compared several machine learning methods and human experts. Gaussian processes performed the best on average, but the question remains whether an ensemble method can be used to improve on individual models. With that aim, we evaluate methods from Section 2 on four air-pollution data sets used in Faganeli Pucer et al. (2018).

The data sets consist of measurements of daily levels of PM_{10} and O_3 on several stations across Slovenia from 2013 to 2016. The data were provided by ARSO. According to reporting guidelines, PM_{10} levels are categorized into three categories: 0-35 $\mu g/m^3$, 35-50 $\mu g/m^3$, greater than 50 $\mu g/m^3$, and O_3 levels are categorized into four categories: 0-60 $\mu g/m^3$, 60-120 $\mu g/m^3$, 120-180 $\mu g/m^3$, greater than 180 $\mu g/m^3$.

Three prediction models—Bayesian lasso, random forests, and Gaussian processes—and human expert predictions were used to predict the levels for the same and next day, depending on covariates available at the current day. Expert predictions were non-probabilistic (categorical). The models were trained in a time-respecting manner (predictions for 2014 were obtained by training the models on data from 2013, predictions for 2015 were obtained by training the models on data from 2013 and 2014, etc.). We used two thirds of observations for training and the rest for testing.

The empirical results for the case study are shown in Table 2. The combination of MM and LOP performs the best over all data sets. MM and LOP are the only methods that show no discernible differences to LOP+MM over all data sets. This provides a strong argument for our method, compared to other methods which learn the latent structure of the data. MM also strictly outperforms MM-diag, as expected. IBCC QPI performs well on the PM_{10} data sets, however it does not outperform MM. Agnostic Bayes performs slightly better than the best source on all data sets, however it is discernibly worse than the best method on two data sets. The other methods are unable to outperform the best performing individual classifier on all data sets.

The major arguments for our method are twofold. First, it is a step forward in the state-of-the-art methods which rely on learning the latent structure of the data (IBCC, IBCC QPI, Supra). This can be seen as it is the only such method which is never discernibly worse than the best performing method. Second, LOP+MM performs the best on all data sets in the case study. It also outperforms LOP on all real-world data sets, which is discernibly better—it is highly unlikely that this is due to chance.

4. Conclusion

We proposed a new Bayesian method for combining probabilistic predictions, based on MVN mixtures. Our method improves on the IBCC (and DBCC) as it is able to model probabilistic classifiers, and on the work of Nazábal et al. (2016) as it also models the

PM_{10} tomorrow	**PM_{10} today**	**O_3 today**	**O_3 tomorrow**
LOP+MM	LOP+MM	LOP+MM	LOP+MM
-0.613 ± 0.022	-0.406 ± 0.019	-0.375 ± 0.025	-0.439 ± 0.027
MM	MM	LOP	LOP
-0.613 ± 0.023	-0.409 ± 0.021	-0.378 ± 0.024	-0.442 ± 0.028
MM-diag	LOP	MM	MM
-0.62 ± 0.023	-0.419 ± 0.019	-0.389 ± 0.028	-0.446 ± 0.029
IBCC QPI	MM-diag	agnostic	agnostic
-0.62 ± 0.024	-0.419 ± 0.021	-0.393 ± 0.027	-0.463 ± 0.028
LOP	IBCC QPI	Source 1	Source 1
-0.634 ± 0.02	-0.436 ± 0.024	-0.393 ± 0.027	-0.463 ± 0.028
agnostic	agnostic	MM-diag	MM-diag
-0.684 ± 0.026	-0.442 ± 0.023	-0.393 ± 0.027	-0.471 ± 0.034
Source 2	Source 1	IBCC QPI	IBCC QPI
-0.684 ± 0.026	-0.442 ± 0.023	-0.394 ± 0.028	-0.49 ± 0.032
Source 1	Source 2	Source 2	Source 2
-0.703 ± 0.019	-0.516 ± 0.015	-0.462 ± 0.02	-0.508 ± 0.021
Supra	Supra	Source 3	Source 3
-0.732 ± 0.017	-0.533 ± 0.018	-0.511 ± 0.033	-0.53 ± 0.022
IBCC	Source 3	IBCC	Supra
-0.791 ± 0.038	-0.641 ± 0.041	-0.563 ± 0.053	-0.808 ± 0.031
Source 3	IBCC	Supra	IBCC
-0.896 ± 0.042	-0.673 ± 0.041	-0.683 ± 0.036	-0.83 ± 0.067
Source 4	Source 4	Source 4	Source 4
-2.23 ± 0.09	-1.578 ± 0.081	-1.18 ± 0.101	-1.371 ± 0.108

Table 2: Estimated log-scores and standard errors on air-pollution data sets. For each data set, the methods are ordered in descending order of performance. Highlighted methods are not discernibly worse than the best-performing method for that data set. The order of magnitude of the differences is in percent, which shows a practical significance in the usefulness of the best performing methods.

correlations between the classifiers. We derived an efficient Gibbs sampler for our method, along with a regularization method for robustness.

The results on toy data sets highlighted that there is no single method that performs well over several diverse data sets. However, our method outperforms related Bayesian methods on all but one real-world data sets. The method proved to be especially useful in the case study, where we combined three machine learning methods and human expert predictions for air-pollutant concentration forecasting. This case study illustrates the main use-case for our method—a relatively small number of probabilistic or crisp (0/1) sources, which are potentially flawed (biased, etc.) and correlated. Our method is also very robust and requires practically no tuning. Additionally, a combination of linear opinion pool and our method proved to be very successful, outperforming all methods on all but one real-world data sets, and also being robust on the difficult toy data sets.

Our method still has some drawbacks, however. The original model is based on full covariance matrices, so the number of the parameters grows quadratically with the number of possible outcomes and the number of classifiers. While this is an improvement over the exponential growth in the number of classifiers of the DBCC model, it still leads to very complex models in high dimensions. One possibility of at least partially regulating this drawback is to constrain the covariance matrices to being diagonal—as in our empirical evaluation. Alternatively, we could assume the same covariance matrix over all mixture components in a true label, or assume the same variances/covariances over several predictions. We leave the analysis of possible constraints for future work. As part of future work, we will also explore if the predictive performance or robustness of the method could be improved by using Gaussian processes and/or variational autoencoders instead of multivariate normal mixtures to model the sources' transformed probabilities. Additionally, we could limit the computational complexity by using sparse Gaussian processes.

As is the case in other areas of learning, there is no single best method of combining probabilistic predictions. In some cases, the best approach is to combine the sources based on their predictive performance or to directly maximize the predictive performance of the combination, such as agnostic Bayes or the linear opinion pool. However, sometimes there are more complex relationships between classifiers and the target variable and methods that are able to learn these patterns will perform better. In such cases, our method is in several ways a superior alternative to existing Bayesian approaches. It is able to model complex relationships, while still being robust and easy to tune, thus providing us with a useful approach for combining probabilistic predictions.

Acknowledgments

The authors would like to acknowledge Jana Faganeli Pucer and the Slovenian Environment Agency (ARSO), who provided environmental pollution data sets. Our research was partially supported by the Slovenian Research Agency (ARRS project grant L1-7542 and research core funding P5-0410). Gregor Pirš is supported by the ARRS Young researcher grant.

References

John Aitchison. The statistical analysis of compositional data. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 139–177, 1982.

George Casella and Edward I. George. Explaining the Gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

Jesús Cerquides and Ramon López De Mántaras. Robust Bayesian linear classifier ensembles. In *European Conference on Machine Learning*, pages 72–83. Springer, 2005.

Roger Cooke et al. *Experts in uncertainty: opinion and subjective probability in science*. Oxford University Press on Demand, 1991.

European Council. Directive 2008/56/EC of the European Parliament and of the Council. *Decision of Council*, 2008.

Jana Faganeli Pucer, Gregor Pirš, and Erik Štrumbelj. A Bayesian approach to forecasting daily air-pollutant levels. *Knowledge and Information Systems*, pages 1–20, 2018.

David Forrest, John Goddard, and Robert Simmons. Odds-setters as forecasters: The case of English football. *International Journal of Forecasting*, 21(3):551–564, 2005.

Jennifer A. Hoeting, David Madigan, Adrian E. Raftery, and Chris T. Volinsky. Bayesian model averaging: a tutorial. *Statistical Science*, pages 382–401, 1999.

Hyun-Chul Kim and Zoubin Ghahramani. Bayesian Classifier Combination. In *International Conference on Artificial Intelligence and Statistics*, pages 619–627, 2012.

Alexandre Lacoste, Mario Marchand, François Laviolette, and Hugo Larochelle. Agnostic Bayesian learning of ensembles. In *International Conference on Machine Learning*, pages 611–619, 2014.

Dennis V. Lindley. *Bayesian Statistics 2*, chapter Reconciliation of discrete probability distributions, pages 375–391. North-Holland, 1985.

Alfredo Nazábal, Pablo García-Moreno, Antonio Artés-Rodríguez, and Zoubin Ghahramani. Human activity recognition by combining a small number of classifiers. *IEEE Journal of Biomedical and Health Informatics*, 20(5):1342–1351, 2016.

SIGKDD. KDD Cup 2017: Highway tollgates traffic flow prediction. `http://www.kdd.org/kdd2017/News/view/announcing-kdd-cup-2017-highway-tollgates-traffic-flow-prediction`, 2017. Accessed: 2018-04-05.

Edwin Simpson. *Combined decision making with multiple agents*. PhD thesis, University of Oxford, 2014.

Edwin Simpson, Stephen Roberts, Ioannis Psorakis, and Arfon Smith. Dynamic Bayesian combination of multiple imperfect classifiers. In *Decision Making and Imperfection*, pages 1–35. Springer, 2013.

Naonori Ueda and Ryohei Nakano. Generalization error of ensemble estimators. In *IEEE International Conference on Neural Networks*, pages 90–95. IEEE, 1996.

Matteo Venanzi, John Guiver, Gabriella Kazai, Pushmeet Kohli, and Milad Shokouhi. Community-based Bayesian aggregation models for crowdsourcing. In *International Conference on World Wide Web*, pages 155–164. ACM, 2014.

World Health Organization. Ambient (outdoor) air quality and health. *Fact sheet*, (313), 2014.

Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman and Hall/CRC, 2012.

Journal of Machine Learning Research 20 (2019) 1-24 Submitted 5/18; Revised 12/18; Published 03/19

Thompson Sampling Guided Stochastic Searching on the Line for Deceptive Environments with Applications to Root-Finding Problems*

Sondre Glimsdal sondre.glimsdal@uia.no
Centre for Artificial Intelligence Research (CAIR)
University of Agder Postboks 422, 4604 Kristiansand, Norway

Ole-Christoffer Granmo ole.granmo@uia.no
Centre for Artificial Intelligence Research (CAIR)
University of Agder Postboks 422, 4604 Kristiansand, Norway

Editor: Avi Pfeffer

Abstract

The multi-armed bandit problem forms the foundation for solving a wide range of online stochastic optimization problems through a simple, yet effective mechanism. One simply casts the problem as a gambler who repeatedly pulls one out of N slot machine arms, eliciting random rewards. Learning of reward probabilities is then combined with reward maximization, by carefully balancing reward exploration against reward exploitation. In this paper, we address a particularly intriguing variant of the multi-armed bandit problem, referred to as the *Stochastic Point Location (SPL)* problem. The gambler is here only told whether the optimal arm (point) lies to the "left" or to the "right" of the arm pulled, with the feedback being erroneous with probability $1 - \pi$. This formulation thus targets optimization in continuous action spaces with both *informative* and *deceptive* feedback. To tackle this class of problems, we formulate a compact and scalable Bayesian representation of the solution space that simultaneously captures both the location of the optimal arm as well as the probability of receiving correct feedback. We further introduce the accompanying Thompson Sampling guided Stochastic Point Location (TS-SPL) scheme for balancing exploration against exploitation. By learning π, TS-SPL also supports *deceptive* environments that are lying about the direction of the optimal arm. This, in turn, allows us to address the fundamental Stochastic Root Finding (SRF) problem. Empirical results demonstrate that our scheme deals with both deceptive and informative environments, significantly outperforming competing algorithms both for SRF and SPL.

Keywords: thompson sampling, searching on the line, probabilistic bisection search, deceptive environment, stochastic point location

1. Introduction

Research on the *Stochastic Point Location* (SPL) problem (Oommen, 1997) has delivered increasingly efficient schemes for locating the optimal point on a line. In all brevity, the optimal point must be found based on iteratively proposing candidate points, with each candidate revealing whether the optimal point lies to the candidate's left or to its right. The provided directions can be erroneous, and the goal is to locate the optimal point with as few non-optimal candidate proposals as possible. The SPL problem can also be cast as an agent that moves on a line, attempting to locate a particular location λ^*. The agent communicates with a teacher that notifies the agent whether its current location λ is greater or lower than λ^*. However, the teacher is of a stochastic nature and feeds the agent erroneous feedback with probability $1 - \pi$.

Despite the simplicity of the SPL problem, SPL schemes have provided novel solutions for a wide range of problems. Intriguing applications include the estimation of non-stationary binomial distributions (Yazidi et al., 2012b), communication network routing (Oommen et al., 2007), and meta-optimization (Oommen et al., 2009). Furthermore, recent research that addresses the related *Stochastic Root-Finding* (SRF) problem provides promising solutions for parameter estimation, transportation system optimization, as well as supply chain optimization (Chen and Schmeiser, 2001; Pasupathy and Kim, 2011).

State-of-the-art. Adaptive Step Searching (ASS) (Tao et al., 2013) is currently the leading approach to solving SPL problems, although it is outperformed by Hierarchical Stochastic Searching on the Line (HSSL) (Yazidi et al., 2012a) in highly volatile non-stationary environments (Tao et al., 2013). Optimal Computing Budget Allocation (OCBA) has also been applied to SPL (Zhang et al., 2015) and provides stable solutions while converging slightly slower than ASS. Unfortunately, these state-of-the-art schemes fail when noise increases beyond a certain degree, which happens when the majority of obtained directions mislead rather than guide. Indeed, by naively following the directions provided under such circumstances, one is systematically led away from the optimal point. We refer to these kinds of problem environments as *deceptive* environments, as opposed to *informative* ones, which are explained in more detail below.

To the best of the authors' knowledge, the pioneering CPL-AdS (Oommen et al., 2003) scheme was the first known approach handling deceptive SPL environments. CPL-AdS relies on two consecutive phases. In the first phase, a sequence of intelligently selected questions is used to classify the environment as either informative or deceptive. By spending a sufficient amount of time in this phase, the classification can be made arbitrarily accurate. In the second phase, a regular SPL scheme is applied, except that the directions obtained are reversed if the problem environment was classified as deceptive in the first phase. This means that the scheme may have to remain in the first phase for an extensive amount of time to ensure that the problem environment is correctly classified, otherwise, one risks being systematically misled in the second phase. These properties largely render CPL-AdS inappropriate for online or anytime problem solving.

Recently, HSSL has been extended by Zhang et al. to cover both informative and deceptive environments, using a Symmetric HSSL (SHSSL) scheme (Zhang et al., 2016). This scheme essentially runs two HSSL schemes in conjunction: one regular, which handles informative environments, and one that inverts all feedback from the environment, to handle deceptive environments. The hierarchy navigation capabilities of HSSL are then exploited to allow SHSSL to switch between the two HSSLs, depending on the nature of the environment. However, a significant limitation of HSSL, namely, that π must be larger than the conjugate of the golden ratio, carries over to SHSSL. Indeed, SHSSL fails to converge for $\pi \in [0.382, 0.682]$, which amounts to approximately 30% of the feasible values for π. This is in contrast to the approach we propose in this paper, as well as to CPL-AdS (Oommen

*. A preliminary version of some of the results of this paper appears in the Proceedings of AIAI'15.

©2019 Sondre Glimsdal, Ole-Christoffer Granmo .

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-263.html.

et al., 2003), since both of these schemes operate along the whole range of π (apart from $\pi = 0.5$).

To cast further light on the challenges lined out above, we here introduce the *N-Door Puzzle* as a framework for modeling deception. We also propose an accompanying novel solution scheme — *Thompson Sampling guided Stochastic Point Location* (TS-SPL). The TS-SPL scheme handles both SPL and SRF problems, and is capable of *simultaneously* solving the problem as well as determining whether we are dealing with an informative or a deceptive environment. As we shall see, not only does this scheme handle an arbitrary level of noise, but it also outperforms current state-of-the-art techniques in both informative and deceptive environments.

The N-Door Puzzle. In the book "To Mock a Mockingbird" (Smullyan, 1988) the following puzzle is formulated: *"Someone was sentenced to death, but since the king loves riddles, he threw this guy into a room with two doors. One leading to death, one leading to freedom. There are two guards, each one guarding one door. One of the guards is a perfect liar, the other one will always tell the truth. The man is allowed to ask one guard a single yes-no question and then has to decide, which door to take. What single question can he ask to guarantee his freedom?"* To avoid spoiling the puzzle for the reader, we omit the solution here and note that asking a double negative question will often be the correct course of action for these types of puzzles.

The above puzzle can be generalized by increasing the number of doors. Instead of deciding between merely two doors, the prisoner now faces N doors, with a guard posted between each pair of doors. Only a single door leads to freedom, the remaining doors lead to death. Every day at sunrise, the prisoner is allowed to ask one of the guards whether the door leading to freedom is to the left or to the right of the guard. However, only a fixed proportion of the guards answers truthfully, the rest are compulsive liars. Further, the guards are randomly assigned a position at each sunrise, and thus, knowing who lies and who tells the truth is impossible. As an additional complication, depending on the mood of the king, the prisoner may be ordered to walk through a door of his choice at an arbitrary day. Therefore, to save his life, it is imperative that the prisoner determines as quickly as possible which door leads to freedom.

Specifically, let $\pi = \frac{\#\text{truthful guards}}{\#\text{guards}}$ be the fraction of truthful guards. Since the guards are randomly assigned a position each day, the probability of obtaining a truthful answer is governed by π. If $\pi < 0.5$ then the majority of the guards are compulsive liars, and the guards as an entity can be characterized as being *deceptive*. Conversely, if $\pi > 0.5$ then the majority of the guards are truthful and the guards can be seen as *informative*. For completeness, we mention that the puzzle is unsolvable for the case where π is exactly equal to $\frac{1}{2}$, since it is then impossible to obtain any information on neither the nature of the doors nor the guards.

Thompson Sampling. The Thompson Sampling (TS) principle was introduced by Thompson already in 1933 (Thompson, 1933) and now forms the basis for several state-of-the-art approaches to the Multi-Armed Bandit (MAB) problem — a fundamental sequential resource allocation problem that has challenged researchers for decades. At each time step in the MAB problem, one is offered to pull one out of N bandit arms, which in turn triggers a stochastic reward. Each arm has an underlying probability of providing a reward, however,

these probabilities are unknown to the decision maker. The challenge is thus to decide which of the arms to pull at each time step, to maximize the expected total number of rewards obtained (Bubeck and Cesa-Bianchi, 2012).

In all brevity, TS seeks to achieve the above goal by quickly shifting from exploring reward probabilities to maximizing the number of rewards obtained. This is achieved by recursively estimating the underlying reward probability of each arm, using Bayesian filtering of the rewards obtained so far. TS then simply selects the next arm to pull based on the Bayesian estimates of the reward probabilities (one reward probability density function per arm).

The arm selection strategy of TS is rather straightforward, yet surprisingly efficient. To determine which arm to pull, a single candidate reward probability is sampled from the probability density function of each arm. *The arm with the highest sample value is the one pulled next.* The outcome of pulling this arm is in turn used to perform the next Bayesian update of the arm's reward probability estimate. It is this simple scheme that makes TS select arms with frequency proportional to the posterior probability of being optimal, leading to quick convergence towards always selecting the optimal arm.

TS has turned out to be among the top performers for traditional MAB problems (Granmo, 2010; Chapelle and Li, 2011), supported by theoretical regret bounds (Agrawal and Goyal, 2012, 2013a; Dong and Van Roy, 2018). It has also been successfully applied to contextual MAB problems (Agrawal and Goyal, 2013b), constrained Gaussian process optimization (Glimsdal and Granmo, 2013), distributed quality of service control in wireless networks (Granmo and Glimsdal, 2013), cognitive radio optimization (Jiao et al., 2016), as well as a foundation for solving the maximum a posteriori estimation problem (Tolpin and Wood, 2015).

Pure Exploration Bandits. Throughout this paper we assume that each SPL problem potentially takes part in a larger system consisting of multiple SPL problems, and not necessarily operating in isolation. From existing applications in the literature, such as web crawler load balancing (Granmo et al., 2007), it is clear that the value of an SPL scheme does hinge upon its ability to cooperate and interact with other decision makers. Such cooperation demands predictable behaviour from the individual decision makers, as well as coordinated balancing of exploring new solution candidates against maintaining good solution candidates. Without such an ability, the system as a whole will not be able to systematically move towards the more promising areas of the search space, gradually focusing in on an optimal configuration. Therefore, in this paper we omit a direct comparison with schemes that rely on a "fixed sampling *then* decide" approach, such as unimodal bandits (Jia and Mannor, 2011). For the same reason, we will not investigate purely exploitative bandits (Even-Dar et al., 2006; Jamieson et al., 2014; Audibert and Bubeck, 2010; Gabillon et al., 2011; Karnin et al., 2013), bandits that have a predefined finite time horizon and whose performance is only measured at the end of that horizon. Such algorithms are free to explore without any negative impact, and this allows them to outperform traditional exploitation-exploration bandits such as TS and UCB in scenarios where exploitation is not required.[1]

1. There also exists a wide spectrum of techniques and schemes in the literature on the topic of searching with noise. See for instance (Pelc, 2002) for a comprehensive survey. These are unable to handle unknown and deceptive environments, with stochastic directional feedback, and are therefore not directly

Paper Contributions. In this paper, we introduce a novel scheme for solving the SPL problem, namely, TS-SPL. At the core of TS-SPL, we find a compact and scalable Bayesian representation of the SPL solution space. This Bayesian representation simultaneously captures both the location of the optimal point (bandit arm) as well as the probability of receiving correct feedback. We further introduce an accompanying scheme for balancing exploration against exploitation, based on TS. By learning π, TS-SPL also supports *deceptive* environments that are lying about the direction of the optimal arm. This, in turn, allows us to solve the fundamental SRF problem. More specifically, the contributions of the paper can be summarized as follows:

1. We introduce a novel TS-SPL scheme that represents the solution space of N-Door Puzzles, and thus SPL problems, in terms of a Bayesian model. As opposed to competing solutions that merely maintain and refine a single candidate solution, our Bayesian model encompasses the complete space of candidate solutions at every time instant.

2. We formulate a compact and scalable Bayesian representation of the solution space that simultaneously captures both the location of the optimal point (arm), as well as the probability of receiving correct feedback. This Bayesian representation of the problem opens up for efficient exploration and exploitation of the solution space with TS.

3. We link TS-SPL to so-called stochastic bisection search; and unify accompanying methods under the umbrella of TS.

4. Similarly, we enhance the Soft Generalized Binary Search (SGBS), Probabilistic Bisection Search (PBS) and Burnashev-Zigangirov Algorithm (BZ) by introducing novel parameter free solutions that take advantage of our Bayesian model of the N-door puzzle and the SPL problem. This approach eliminates previous reliance on knowing the exact degree of noise affecting the system to be optimized.

5. We provide the first unified empirical comparison of the key state-of-the-art SPL- and SRF solvers.

6. We finally demonstrate the empirical performance of TS-SPL for both SPL and SRF problems. TS-SPL outperforms the state-of-the-art algorithms in both informative and deceptive environments, except for the SGBS and BZ schemes with correctly specified observation noise.

Paper Outline. The paper is organized as follows. In Section 2, we present our scheme for TS guided SPL (TS-SPL). We first introduce the Bayesian model of the N-door puzzle. Based on the Bayesian model, we then formulate our TS-based scheme that balances solution space exploration against reward maximization. We further extend selected state-of-the-art solution schemes with our Bayesian N-door puzzle model. This extension removes the need for knowing the observation noise beforehand. In Section 3, we provide extensive empirical results comparing TS-SPL with state-of-the-art schemes for both SPL and SRF. We conclude in Section 4 and point to promising directions for further work.

comparable to SPL solution schemes. We have therefore not included this class of techniques in the present paper.

2. Thompson Sampling Guided Stochastic Point Location

In this section, we introduce the TS-SPL scheme. The scheme can be summarized as follows. At the core of TS-SPL we find a Bayesian model of the N-Door Puzzle. Formally, we represent an instance of the N-door puzzle as a tuple $(\lambda^*, \pi^*) \in D \times T$, where $D = \{d_1, \ldots, d_N\}$ is the set of doors and $T \in [0, 1]$ is the truthfulness of the guards. Let (λ^*, π^*) be the particular N-door puzzle faced. A novel aspect of TS-SPL is that instead of maintaining a single or a limited set of candidate solutions, we instead maintain a posterior distribution over the whole solution space, $(\lambda, \pi) \in D \times T$. This distribution is conditioned on the feedback already obtained up to time step n, allowing us to single in on (λ^*, π^*) as the number of time steps increases, ultimately converging to (λ^*, π^*).

Assuming no prior information, we assign a uniform distribution over $D \times T$, i.e., all puzzle instances are equally probable. By gradually refining the posterior distribution over $D \times T$, we can select guards to question in a goal-oriented manner. In all brevity, we sample a solution candidate (λ^c, π^c) from $D \times T$, selecting the guard to the left or to right of λ^c. The answer of the selected guard is then used to update our posterior distribution. By repeating this procedure, the expected probability of the underlying N-door puzzle instance (λ^*, π^*) increases monotonically, reducing the probability of other puzzle instances. In effect, given enough iterations, TS-SPL will correctly identify the door leading to freedom as the posterior probability of (λ^*, π^*) approaches unity.

2.1 Bayesian Model of the N-Door Puzzle

The main purpose of the Bayesian model is to facilitate the efficient calculation of a posterior distribution over the possible N-door puzzle instances, $D \times T$. Since the prisoner does not initially know which problem instance he is facing, and since the observations are stochastic, we cast D and T as two random variables. We further assume that D and T are independent of each other. Furthermore, the information we obtain from questioning the guards is represented as a set of random variables $Q = \{Q_1, \ldots, Q_n\}$, with each random variable Q_k representing the answer from question k. Finally, we assume that the outcomes of the individual questions $Q_k \in Q$ are independent when conditioned on D and P. For each question Q_k, we can then compute the probability of the answer ("left" or "right") that we received from the guard, as summarized in Table 1.

Guard to the left of door to freedom	$P(\text{left} \mid \text{guard, door}, t) = t$
	$P(\text{right} \mid \text{guard, door}, t) = 1 - t$
Guard to the right of door to freedom:	$P(\text{left} \mid \text{guard, door}, t) = 1 - t$
	$P(\text{right} \mid \text{guard, door}, t) = t$

Table 1: Conditional door probabilities

As an example, let us assume that the truthfulness of the guards is $t = 0.75$. If for instance the guard to the left of door d_4 replies that the door leading to freedom lies to his left, we can infer that all doors to the left have the likelihood of $t = 0.75$ of leading to freedom, and all the doors to the right have the likelihood $1 - t = 0.25$ of leading to freedom.

Applying Bayes Theorem to $P(Q|d,t)$, defined in Table 1, we are able to derive closed-form expressions for the posterior distributions of both D and T. The derivation of $P(d|Q)$, $d \in D$, follows [the derivation of $P(t|Q)$, $t \in T$, is analogous, and is left out here for the sake of brevity]:

$$P(d|Q) = \sum_{t \in T} P(d,t|Q) \tag{1}$$
$$\propto \sum_{t \in T} P(Q|d,t)P(d,t) \tag{2}$$
$$= \sum_{t \in T} P(Q|d,t)P(d)P(t) \tag{3}$$
$$= \sum_{t \in T} \hat{Q}Q^{+}P(d)P(t) \tag{4}$$

Above, $\hat{Q} = \prod_{k=1}^{n-1} P(Q_k|d,t)$ and $Q^{+} = P(Q_n|d,t)$. Further, (2) follows directly from Bayes Theorem. We obtain (3) as a result of the independence of D and T, and (4) from the independence between the questions in Q. This leads to the following two equations for updating our knowledge about both the door probabilities (5) and the truthfulness of the guards (6).

$$P(d|Q) \propto \sum_{t \in T} \hat{Q}Q^{+}P(d)P(t) \tag{5}$$

$$P(t|Q) \propto \sum_{d \in D} \hat{Q}Q^{+}P(d)P(t) \tag{6}$$

2.2 Guard Selection

We have now formally determined how we can transform information from the guards into a probability distribution over which door leads to freedom. However, as mentioned previously, we also face a trade-off between exploring different doors and zeroing in on the best door found so far. To handle this trade-off, we model the door selection as a so-called Global Information MAB (GI-MAB) (Atan et al., 2015).

To decide which door should be selected at each iteration, we solve the GI-MAB by utilizing the principle of TS. Here, the selection process is simply to select a random door proportional to the probability that this door is the one that leads to freedom. Once the door has been selected, we need to decide which of the guards to query: the guard to the left or to the right of the door selected. We do this by randomly selecting one of the guards, again proportional to the sum of the probabilities of the doors next to each guard. Let us assume for instance that we have three doors $d_k, 1 \leq k \leq 3$ with the probabilities of leading to freedom: $P(d_1) = 0.1, P(d_2) = 0.2, P(d_3) = 0.7$. Then, according to the TS principle, these are also the probabilities we use to sample a particular door. Note that since the answer obtained from each guard affects the complete probability distribution over D (the probability associated with every door is updated), we have a GI-MAB as opposed to a traditional MAB.

2.3 Improving State-of-the-Art Schemes with the Bayesian Model of the N-Door Puzzle

A main advantage of TS-SPL compared to similar schemes is the utilization of the Bayesian model that enables TS-SPL to operate without knowing the problem parameters in advance. Due to TS-SPL's close connection to the Probabilistic Bisection Search (PBS) (Horstein, 1963), Noisy Generalized Binary Search (NGBS) (Nowak, 2008) and the BZ algorithm (Burnashev and Zigangirov, 1974), we will here use our Bayesian TS-SPL model to also make these other schemes parameter free.

Probabilistic Bisection Search

The goal of PBS[2] (Waeber et al., 2013; Nowak, 2008) is to locate an unknown point $X^* \in [0,1]$. To acquire intelligence on the location of X^*, one queries an oracle of the relation between a point x and X^*. The oracle responds by informing whether x is on the left or the right side of X^*. If we assume that the oracle is always telling the truth, then the well-known deterministic bisection search, which halves the search space with each query, can be employed to efficiently find X^*. However, in PBS we assume that the Oracle provides correct answers with probability $p \in (0.5, 1.0]$ and erroneous ones with probability $1-p$.

The PBS can be traced back to Horstein (Horstein, 1963). In PBS a probability distribution is mapped over the search space and is gradually updated using a Bayesian methodology under the assumption that the environment noise p is known a priori. The search space is then continuously explored using the median of the posterior distribution as the point of interest. It has been shown that PBS has a geometric rate of convergence under the latter assumptions (Waeber et al., 2013).

As the noise p is assumed to be given, one can simply invoke (7) to calculate the posterior distribution:

$$P(d \mid Q) \propto P(Q \mid d)\ P(d). \tag{7}$$

Here, $P(Q \mid d)$ is the conditional probability of obtaining answer Q (point to the right). That is, for every location d to the left of X^*, the probability that the oracle directs the decision maker to the right is $P(Q \mid d) = p$. And conversely, $P(Q \mid d) = 1-p$ for d to the right of X^*.

To explicitly represent PBS' dependence on knowing p beforehand, we can cast (7) in terms of (5) and (6). The resulting model is identical to TS-SPL, with the major difference that PBS employs the median to explore the search space. We denote this new and improved scheme PBS-M.

We also observe that due to its simple nature, PBS is particularly well-suited for parallel computing environments (Pallone et al., 2014), as opposed to more traditional stochastic approximation methods (Kushner and Yin, 1987).

PowerTest-Probabilistic Bisection Search

In a recent paper, Frazier et al. (Frazier et al., 2019) demonstrated an alternative approach to removing the dependency of PBS on knowing the fixed noise probability p. Instead of

2. In this context this scheme also covers the Stochastic *Binary* Search

applying a Bayesian prior over p, as done in TS-SPL, they introduce a frequency-based approach, referred to as PowerTest-PBS (PT-PBS). PT-PBS is based on repeatedly sampling the underlying function $g(x)$ until a pre-specified confidence α is archived on a hypothesis test over the sign of the feedback of $g(x)$. They further demonstrated that the asymptotic convergence of PT-PBS is similar to that of Stochastic Approximation (SA).

Generalized Binary Search

The Generalized Binary Search (GBS) problem can be formulated as follows (Nowak, 2008, 2011). Consider a collection of unique binary-valued functions H defined on a domain X. Each $h \in H$ is defined as a mapping from X to $\{-1, 1\}$. Assume that there exists an optimal function $h^* \in H$ that produces the correct binary labeling for each $x \in X$. For each query $x \in X$, the value of $h^*(x)$ is observed, possibly corrupted by independent binary noise. The objective is then to determine the function h^* using as few queries as possible. In this paper, we restrict H to the class of threshold binary functions with the effect of turning the GBS into an informative N-door puzzle.

If the feedback is noiseless then the problem simplifies to the combinatorial problem of finding an optimal decision tree in the H space, a problem that Hyafil and Rivest showed to be NP-complete (Hyafil and Rivest, 1976; Nowak, 2011).

The Soft-Decision Generalized Binary Search (SDGB-Search) (Nowak, 2008, 2011) is the *state-of-art* algorithm for finding $h^*(x) \in H$ when the probability of binary noise is less than 1/2, that is, for informative environments.

Similar to TS-SPL, SDBG-search employs a probabilistic model that for time step n assigns a probability $p_n(h)$ to each $h \in H$. However, for each time-step, it decides which $x \in X$ is queried next based on a deterministic heuristic:

$$\arg\min_{x \in X} \sum_{h \in H} |p(h)h(x)| \tag{8}$$

SDGB uses the following equation to determine and update $p_n(h)$ at each time step:

$$p_{i+1}(h) \propto p_i(h)\beta^{(1-z_i(h))/2}(1-\beta)^{(1+z_i(h))/2}. \tag{9}$$

Here, $z_i(h) = h(x_i)y_i$ and $y_i \in \{-1, 1\}$ are the responses from $h^*(x_i)$. Simplifying (9), we observe that $z_i(h)$ represents an *AND* operator that takes on the value 1 if $h(x_i)$ is equal to $h^*(x_i)$ and -1 otherwise. Furthermore, we note that since $z_h(i) \in \{-1, 1\}$, then one of $1 - z_i(h)$ and $1 + z_i(h)$ will have to take the value 2, while the other takes the value 0.

By applying the transformation $\pi = 1 - \beta$, we can rewrite (9) as:

$$p_{i+1}(h) \propto= \begin{cases} p_i(h) \times \pi & \text{if } y_i = h^*(x_i) \\ p_i(h) \times (1-\pi) & \text{else} \end{cases} \tag{10}$$

.

This update scheme is identical to the one in PBS and thus suffers from the same limitation (noise probability is assumed to be known a priori). In the same manner as we enhanced PBS by employing our Bayesian TS-SPL scheme, we can make SDGB parameter-free using (5,6). In the following, we will denote this improved version of SDGB as SDGB-M.

Burnashev-Zigangirov Algorithm

The Burnashev-Zigangirov (BZ) algorithm (Burnashev and Zigangirov, 1974) is one of the most widely used algorithms for solving the discrete PBS problem and has in particular been used in active learning (Singh et al., 2006; Castro and Nowak, 2006). The BZ algorithm searches for a point θ^* that is located on a line. This line is discretized into m bins and we are only allowed to query the borders of the bins for the direction of θ^*. The BZ algorithm suffers from the same practical limitation as PBS and SDGB, namely a dependency on knowing the exact noise level.

We will now show how the BZ algorithm can be improved in a similar fashion as PBS and SDGB, leveraging our Bayesian model. Let $a_i(j)$ denote the probability of θ^* residing in bin I_i at time-step j. The probability mass function (pmf) of all the bins is therefore $\mathbf{a}(j) = \{a_1(j), a_2(j), \ldots, a_m(j)\}$ with its cumulative density function (cdf) denoted as $\mathbf{A}(j)$.

To decide which point to investigate next (i.e., decide a value for X_{j+1}), the BZ algorithm selects one of the two closest points to the median of $\mathbf{a}(j)$. We denote this point $k = k(j+1)$. The binary response variable $Y_{j+1} = \mathbb{1}\{X_{(j+1)} \geq \theta^*\}$ is observed with probability $1 - \alpha$, whereas $Y_{j+1} = \mathbb{1}\{X_{(j+1)} < \theta^*\}$ is observed with probability α (the noise probability).

To update the probability distribution over $\mathbf{a}(j)$, we introduce $\beta = 1 - \alpha$ and $\tau = 2A(k(j+1)) - 1$. For $i \leq k$, we then have

$$a_i(j+1) = a_i(j) \begin{cases} \frac{2\alpha}{1-\tau(\beta-\alpha)} & \text{if } Y_{j+1} = 0 \\ \frac{2\beta}{1+\tau(\beta-\alpha)} & \text{if } Y_{j+1} = 1 \end{cases}$$

and for $i > k$ we have:

$$a_i(j+1) = a_i(j) \begin{cases} \frac{2\beta}{1-\tau(\beta-\alpha)} & \text{if } Y_{j+1} = 0 \\ \frac{2\alpha}{1+\tau(\beta-\alpha)} & \text{if } Y_{j+1} = 1 \end{cases}$$

To make the BZ algorithm parameter-free, we first note that for any given noise $t \in T$, we have that $\beta = t$, $\alpha = 1 - t$, $\beta - \alpha = 2t - 1$, and $\tau = A_k(j) - (1 - A_k(j))$. After some simple algebraic manipulations, it turns out that the updating scheme of the BZ algorithm is identical to PBS except that:

1. The BZ algorithm calculates the normalizing factor as a part of the updating rule instead of using the likelihood value, and then later normalizes as PBS does.

2. The BZ algorithm samples on the interval edges while PBS samples the midpoints of each interval.

To obtain an enhanced parameter-free version of the BZ algorithm, we simply replace α as a pre-determined constant with a prior distribution that we marginalize out using (5) and (6). We denote the resulting scheme BZ-M.

3. Empirical Results

In this section, we evaluate the performance of TS-SPL empirically, in comparison with competing schemes. We investigate both the effect that the various parameter settings

have on the behavior, as well as the capability of TS-SPL to handle different applications, including SPL and SRF problems. Unless otherwise noted, the empirical results report the average of 10 000 independent trials.

For some of the applications we investigate here, we do not find any existing scheme that handles deceptive environments. Instead, the schemes we identified assume that feedback is on average informative. To make the comparison fair, we thus introduce TS-SPL-INF, which is configured with the precondition that the feedback is informative. This modification also serves to exemplify one of the advantages of our Bayesian approach — we can tailor the the prior distribution of the noisy probability for the task at hand. Note that this informed prior is equivalent to the priors we use for the other probability theory based schemes we introduced in this paper, namely, PBS-M and SDGB-M.

Further note that we apply a fixed set of parameter values across the whole suite of experiments, set to optimize overall performance. For SHSSL (Zhang et al., 2016) and HSSL (Yazidi et al., 2012a) we used a tree branching factor of $D = 8$, and for ASS (Tao et al., 2013) we set $N_{\max} = 256$ and $N_{\min} = 1$. For OCBA (Zhang et al., 2016), we set $n_0 = 15$ and $\theta = 1/256$. We additionally set the confidence γ of PT-PBS (Frazier et al., 2019) to 0.55 based on a comprehensive brute force search for the best value. The prior used for TS-SPL is uniform over the unit interval and is discretized as $|D| = 201$ and $|T| = 101$. For the informative schemes TS-SPL-INF, PGA-M, SGDB-M, BZ-M, we use the same prior for the doors as for TS-SPL, $|D| = 201$ however, we use a uniform prior over the interval $(0.5, 1]$ for truthfulness, with $|T| = 51$.

We will in the following subsections investigate (1) the effect of different priors on TS-SPL; (2) TS-SPL's ability to identify the nature of the underlying stochastic environment; (3) the ability to solve the SPL problem; and (4) performance on SRF problems – a particularly intriguing class of deceptive environments that arises naturally as a result of the properties of stochastic root finding.

3.1 Sensitivity to Discretization and Distribution of Prior

Although TS-SPL is a parameter free scheme, it depends on defining $D \times T$, the set of all possible N-Door Puzzles, and then formulating a prior distribution over this space. We here investigate to what degree the performance of TS-SPL is affected by the degree of discretization, that is, the cardinality of $D \times T$.

To measure performance, we count how many time steps passes before 95% of the probability mass is contained within the target interval I, that is, $P(I|\text{Observed History}) \geq 0.95$. We refer to this event as convergence of the learning process.

From Table 2, we observe that the cardinality of D, in fact, does affect the performance of TS-SPL. As $|D|$ increases, so does the time it takes before TS-SPL converges. However, it is evident that the relationship between convergence time and $|D|$ is non-linear. Indeed, the increase in convergence time is insignificant even when doubling the number of doors from 3200 to 6400. The behaviour reported in the table thus indicates a logarithmic relation between $|D|$ and convergence time.

To see how the cardinality $|T|$ of T affects performance, we increase $|T|$ from 50 to 3200, fixing $|D|$ to 100. From Table 3, it is clear that $|T|$ does not significantly affect performance.

$\lvert D\rvert$:	100	200	400	800	1600	3200	6400
Convergence Steps:	31.4	36.0	38.9	39.4	39.3	40.2	40.9

Table 2: Convergence steps for TS-SPL solving the N-Door Puzzle with $\lambda^* = 0.15$, $I = \{0.15 \pm 0.01\}$, $T = \{0.8\}$, and $\pi = 0.8$.

$\lvert T\rvert$:	50	100	200	400	800	1600	3200
Convergence Steps:	51.6	50.8	48.4	52.1	51.0	52.4	52.1

Table 3: Convergence steps for TS-SPL solving the N-Door Puzzle with $\lambda^* = 0.15$, $I = \{0.15 \pm 0.01\}$, $|D| = 101$, and $\pi = 0.8$.

Another advantage of our Bayesian scheme is the ability to incorporate prior information to guide the algorithm. On the other hand, specifying an incorrect prior can potentially deteriorate performance instead of enhancing it. In Table 4, we provide performance results from employing an informed prior over T and D. With the correct underlying values $\lambda^* = \pi = 0.85$, we specify three types of priors: Correct $\propto N(\mu = 0.85, \sigma = 0.3)$, Incorrect $\propto N(\mu = 0.15, \sigma = 0.3)$ and Flat (all solutions equally probable), denoted C, I, and F, respectively. We can see the effect of these different priors in Table 4. In brief, having a correct prior over the doors contributes more to convergence time than having a correct prior over the truthfulness of the guards. The disadvantage of setting an incorrectly biased prior is also evident, as the flat prior performs better than any combination involving a incorrectly biased prior.

3.2 Tracking the Truthfulness of the Environment

An interesting property of TS-SPL is its ability to provide a distribution over the truthfulness π of the environment. This can be a significant advantage because information on π can be leveraged in various ways. As an example, information on π can be used in the case of repeated trials, where the information from previous trials can be used as a prior in subsequent trials, greatly increasing convergence speed (cf. Section 3.1). Figure 1 plots the probability of each level of noise as the TS-SPL progresses with noise probability $\pi = 0.15$ (a highly deceptive environment). As seen, TS-SPL is capable of quickly estimating π accurately.

3.3 Stochastic Point Location

The N-Door Puzzle, as outlined in the introduction, is dependent on two variables λ^* and π^*, with λ^* specifying the door leading to freedom and π^* the truthfulness of the guards. Since the N-Door Puzzle does not pose any spatial requirements on the placements of the doors we can generate a mapping from the N-Door Puzzle to the SPL problem by uniformly placing the doors over the unit interval.

We here use so-called regret to measure performance because not all of the schemes evaluated in this section are Bayesian. Regret is further typical for evaluating multi-armed bandit algorithms Regret can be stated as the cumulative penalty from selecting sub-optimal

Door	Truthfulness	Convergence
F	F	36.4
F	C	35.7
F	I	41.2
C	F	30.2
C	C	30.0
C	I	40.5
I	F	46.4
I	C	45.2
I	I	113.1

Table 4: Convergence steps for TS-SPL solving the N-Door Puzzle with different priors: C - Correct Prior, F - Flat Prior, I - Incorrect prior. Here, $\lambda^* = 0.85$, $I = \{0.15 \pm 0.01\}$, $|T| = |D| = 101$, and $\pi = 0.85$.

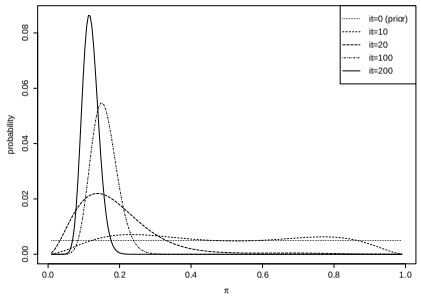


Figure 1: TS-SPL maintains a posterior distribution over π. Here, the true underlying value of π is 0.15. The figure shows the posterior distribution of π after various number of iterations during a single run of TS-SPL. As seen, TS-SPL obtains a sharply peaked posterior over π after only 20 iterations.

actions. In the case of SPL we define regret as the (unsigned) distance between the selected point x and the optimal point λ^*.

3.3.1 Informative SPL

We first evaluate the performance of TS-SPL and TS-SPL-INF on an informative SPL problem, in comparison with algorithms designed to handle informative environments. To the best of our knowledge this is the first time both the family of PBS based schemes and the family of SPL based schemes are compared.

The performance of the different schemes is summarized in Table 5. One significant observation is the performance difference between the Learning Automata (LA) based schemes (HSSL and SHSSL) and the Bayesian schemes. It is clear that performance-wise, Bayesian schemes significantly outperform the LA based schemes. However, it should be noted that the LA based schemes require less memory and run faster than the Bayesian ones due to their simplicity.

As seen in Table 5, the distance $|\frac{1}{2} - \lambda^*|$ is an important metric for how hard a particular SPL problem is to solve. This can be explained by the fact that most schemes start exploring from the center. Thus, if λ^* is far from the center, such a scheme needs more evidence before it starts exploring the peripheral regions of the search space. This is particularly apparent for PBS-M as its performance peaks in the case where $\lambda^* = 0.25$, even when faced with significant noise ($\pi = 0.65$).

Since PBS-M pursues the median of the probability distribution, we can say that PBS-M is conservative in its exploration. This is because it takes significant more evidence to move the point of exploration compared to TS-SPL. TS-SPL, on the other hand, has a tendency to explore too much. Indeed, as noted by Lattimore (Lattimore, 2015), using TS for exploration can lead to over-exploration when facing high variance distributions. In the low noise scenarios, however, NGBS-M is the most efficient scheme, exploring deterministically. For PT-PBS, we observe throughout all the experiments significantly worse performance than PBS-M. We thus omit the PT-PBS results for the remaining experiments, focusing on PBS-M instead.

Finally, from Table 6 we observe that TS-SPL-INF exhibits the lowest standard deviation overall, and is consequentially the scheme that consistently perform closest to its expected regret for every trial. This is in sharp contrast to PBS-M who outperform TS-SPL when it comes to average regret, but is unable to do so consistently. NGBS-M also displays significant variance in high noise scenarios.

3.3.2 SPL in Deceptive Environments

With the underlying π taking on values in the interval $[0, 1]$, we test TS-SPL, CPL-AdS (Oommen et al., 2003) and SHSSL (Zhang et al., 2016) for speed of convergence as well as accumulation of regret. However, since CPL-AdS operates in a two-phase manner, direct comparison with TS-SPL and SHSSL is inappropriate (the latter schemes also operate online). Oommen et al. states that this decision phase needs approximately 200 time steps (Oommen et al., 2003), and by this time TS-SPL is already close to converging to the actual solution. Table 7 further explores this difference. As seen, TS-SPL outperforms CPL-AdS by several orders of magnitude, also outperforming SHSSL.

	Avg Regret $\lambda^* = 0.25$	Avg Regret $\lambda^* = 0.85$	Avg Regret $\lambda^* = 0.95$
TS-SPL	29.2 / 9.8 / 5.1	36.4 / 12.9 / 6.2	57.3 / 20.3 / 10.1
TS-SPL-INF	22.2 / 7.3 / 3.7	**22.5** / 7.7 / 3.8	**23.9** / 8.7 / 4.3
PBS-M	**9.8** / 4.0 / 2.6	32.7 / 14.2 / 8.5	52.1 / 29.6 / 16.9
PT-PBS	245.4 / 247.5 / 247.4	548.2 / 751.5 / 785.2	551.4 / 815.0 / 828.4
BZ-M	23.5 / 5.9 / 2.2	27.5 / 6.3 / 2.5	35.1 / 9.6 / 3.4
NGBS-M	36.9 / **3.5** / **1.0**	48.9 / **4.5** / **1.5**	68.5 / **7.1** / **2.3**
ASS	45.8 / 17.0 / 6.7	30.4 / 8.9 / 3.6	38.8 / 11.7 / 3.9
OCBA	70.8 / 47.4 / 35.2	89.9 / 55.8 / 37.1	112.1 / 78.4 / 48.8
HSSL	117.3 / 23.1 / 8.2	111.7 / 16.7 / 4.8	131.5 / 19.1 / 5.3
SHSSL	152.2 / 32.6 / 11.8	151.8 / 23.5 / 6.5	175.1 / 26.1 / 7.3

Table 5: Average regret for the different schemes in an informative SPL. The result is reported in the format $a/b/c$, where a is the average regret for $\pi = 0.65$, b for $\pi = 0.75$, and c for $\pi = 0.85$. The number of time steps per trial is 1000.

	Std. dev. $\lambda^* = 0.25$	Std. dev. $\lambda^* = 0.85$	Std. dev. $\lambda^* = 0.95$
TS-SPL	16.8 / 5.9 / 2.6	20.5 / 6.5 / 3.1	30.9 / 10.3 / 4.6
TS-SPL-INF	**13.8** / **4.2** / 2.0	**14.2** / **4.4** / 2.5	**15.7** / **5.7** / 2.4
PBS-M	15.2 / 10.3 / 10.2	69.1 / 40.9 / 31.5	94.2 / 71.1 / 56.6
PT-PBS	20.1 / 19.0 / 13.6	145.6 / 31.0 / 22.8	192.1 / 60.5 / 55.8
BZ-M	30.4 / 8.9 / 3.1	40.8 / 9.8 / 4.9	48.8 / 15.3 / 5.3
NGBS-M	68.5 / 8.9 / **0.9**	83.7 / 13.6 / **1.4**	108.7 / 19.4 / **1.6**
ASS	51.6 / 22.4 / 10.1	47.8 / 15.7 / 5.4	62.3 / 23.1 / 4.6
OCBA	46.2 / 27.6 / 19.4	63.9 / 43.9 / 25.6	76.1 / 64.6 / 41.9
HSSL	71.7 / 16.1 / 4.6	83.6 / 16.1 / 4.2	94.8 / 19.4 / 4.5
SHSSL	89.7 / 23.5 / 6.2	108.5 / 23.4 / 5.8	126.5 / 27.7 / 6.4

Table 6: Standard deviation for the different schemes in an informative SPL. The result is reported in the format $a/b/c$, where a is the standard deviation for $\pi = 0.65$, b for $\pi = 0.75$, and c for $\pi = 0.85$. The number of time steps per trial is 1000.

	$\pi = 0.85$	$\pi = 0.15$
TS-SPL ($\lambda^* = 0.85$)	6.2	**6.2**
CPL-AdS ($\lambda^* = 0.85$)	501.6 / 354.9	842.8/502.3
PBS-M ($\lambda^* = 0.85$)	31.5	77.5
BZ-M ($\lambda^* = 0.85$)	4.9	352.5
NGBS-M ($\lambda^* = 0.85$)	**1.4**	191.2
SHSSL ($\lambda^* = 0.85$)	6.5	6.5

Table 7: Cumulative regret for the deceptive SPL problem after $N = 1000$ time steps. For CPL-AdS, we report both the total accumulated regret, as well as regret obtained after the nature of the environment has been decided.

Another interesting observation is that the performance of TS-SPL is symmetric around 0.5. Further note that SHSSL fails to converge for $\pi \in [0.382, 0.682]$, as stated earlier. Hence, SHSSL is effectively operating with a 30% smaller search space for π than both TS-SPL and CPL-AdS.

After modifying PBS, NGBS and BZ to support a Bayesian model of truthfulness, we can use the same prior that we apply in TS-SPL also for these schemes, leading to PBS-M, NGBS-M and BZ-M. The effect of this enhancement to existing schemes is summarized in Table 7. As clearly seen, the query selection method for these schemes is not suited to handle deceptive environments.

3.4 Stochastic Root-Finding

The deterministic root finding problem concerns locating a root x^* of a function $g(x)$, defined over an interval (a, b) [i.e., finding x^*, $g(x^*) = 0$]. We assume that $g(x)$ is unknown, however, an oracle returns the value of $g(x)$ at any point x queried. The problem is then how to determine the root x^* using as few queries as possible. One approach to solving the deterministic root finding problem is the Bisection Method. This approach halves the search space in each iteration by continually querying the oracle using the midpoint of the remaining search space.

If the response from the oracle is noisy, we obtain the SRF problem (Pasupathy and Kim, 2011). We define the SRF problem as follows. For any $x \in (0, 1)$, the oracle generates a sample $Y(x) = g(x) + w_{\text{noise}}$, where w_{noise} is a random variable with mean zero. Let $S(x)$ denote the sign of $Y(x)$: $S(x) = \text{sgn}[Y(x)]$. Notice that the noise w_{noise} may render $\text{sgn}[Y(x)]$ different from $\text{sgn}[g(x)]$. Thus, with noisy feedback, the Bisection Method may discard the wrong half of the search space. The challenge is then how to select a sequence of queries $x_1, x_2, \ldots, x_n$ to gather information on x^*, so that the final query x_n is close to x^*, $|x_n - x^*| < \epsilon$, despite the noise (Waeber et al., 2011). Note that in the SRF problems we investigate here, $S(x)$ returns $\text{sgn}[g(x)]$ with probability π and $-\text{sgn}[g(x)]$ with probability $1 - \pi$.

The traditional approach to solving SRF problems is to apply a variant of Stochastic Approximation (SA) (Robbins and Monro, 1951; Kiefer and Wolfowitz, 1952). Implementation-

wise SA methods[3] extend or modify the iterative Newton-Raphson algorithm to handle noise:

$$x_{n+1} = x_n - a_n Y_n(x_n)$$

where $\{a_n\}$ is a sequence of step lengths that decreases as n increases.

Approaches for applying SA to the SRF problem has been extensively studied in the literature. It is outside the scope of this article to give a full literature review, however, interested readers are referred to surveys in recent studies (Lai, 2003; Asmussen and Glynn, 2007; Pasupathy and Kim, 2011). As there exists a myriad of different SA approaches, we have selected one of the more fundamental ones to form a basis for contrasting the different schemes.

Note that the SA approach we use here requires that $g(x)$ is monotonic. This can be explained as follows. The main difference between SRF and SPL is that, unlike SPL, the SRF oracle does not directly provide feedback on the direction of the root x^* from the query location x. However, for monotonic $g(x)$, one can obtain this direction from the sign of $Y(x)$, $S(x) \in \{-1, 1\}$ and from the derivative $g'(x)$ of $g(x)$. If $g(x)$ is increasing and $S(x) = 1$, then the direction derived from the oracle is "to the left of x" [and "to the right" if $S(x) = -1$]. Conversely, if $g(x)$ is decreasing and $S(x) = 1$, then the direction obtained is "to the right of x" [and "to the left" if $S(x) = -1$].

TS-SPL, on the other hand, does not need to know the derivative of $g(x)$. Indeed, $g(x)$ does not even need to be monotonic. Instead, TS-SPL merely requires an arbitrary mapping of the sign $S(x)$ to a direction. One could, for instance, define positive to mean "left", $S(x) = 1 \Rightarrow$ *left*, and negative to mean "right", $S(x) = -1 \Rightarrow$ *right*. If it turns out that the opposite is the case, TS-SPL will recognize that the feedback is deceptive and still solve the problem. An informative scheme, on the other hand, will be misled in such a deceptive environment.

Informative SPL schemes can also be used for SRF problems. However, then we need an initial sampling step that decides the nature of the function $g(x)$. Learning whether the function $g(x)$ starts above zero and falls below zero, or vice versa, can be done by repeatedly querying a single point on the edge of the interval $(0, 1)$, obtaining multiple samples from either $S(x)$. In brief, by estimating $E[S(x)]$, we can decide the nature of $g(x)$. To gain insight into how many repeated samples are sufficient for estimating $E[S(x)]$ accurately, we employ the two sided Hoeffding's inequality $P(|\bar{X} - E[\bar{X}]| \geq \delta) \leq 2e^{-2n\delta^2}$. Here, $\bar{X}$ is the average of n queries at x and δ is a value such that $|\pi - \frac{1}{2}| \geq \delta$. Setting the rhs. equal to p and solving for n, we obtain $n \geq -\frac{\log(p/2)}{2\delta^2}$. Plugging in for $\delta = 0.05$ and $p = 0.99$ we obtain $\lceil n \rceil = 62$. Thus we are 99% sure of our estimate of $S(x)$, given that $|\pi - 0.5| \geq 0.05$.

The functions that we use to measure performance and compare schemes are illustrated in Figure 2. From Table 8, 9 and 10 it is clear that TS-SPL is the most efficient root solver among state-of-the-art schemes. We believe this largely comes from the fact that it simultaneously learns the nature of the oracle (informative or deceptive), as well as trying to locate the root x^*. In addition, there is the risk that the sampling procedure that the other schemes apply to determine which direction $g(x)$ is increasing may conclude with the wrong answer. If this happens, none of the schemes depending on the sampling will

3. The form of SA shown here is also referred to as Classical Stochastic Approximation (CSA) as it closely resembles the original form proposed by Robbins and Monro (Pasupathy and Schmeiser, 2010).

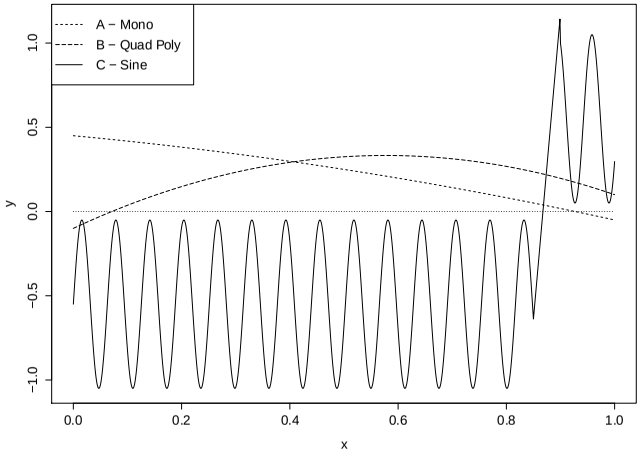


Figure 2: The three functions A, B and C for benchmarking stochastic root finding schemes.

Func A	Avg Regret. $\pi = 0.65$	Avg Regret. $\pi = 0.75$	Avg Regret $\pi = 0.85$
TS-SPL	46.0	17.1	8.6
TS-SPL-INF	55.2 (24.3)	39.1 (8.1)	35.0 (4.1)
PBS-M	55.1 (24.2)	40.3 (9.3)	35.2 (4.2)
NGBS-M	53.4 (22.4)	35.3 (4.3)	32.9 (1.9)
BZ-M	63.8 (32.8)	39.9 (8.9)	33.8 (2.8)
SA	**32.4**	**14.8**	**5.4**
ASS	80.4 (49.4)	38.7 (7.7)	33.5 (2.5)
HSSL	60.4 (29.4)	45.5 (14.6)	35.8 (4.8)
SHSSL	62.8	20.2	6.7
CPL-AdS	162.1 (107.9)	146.3 (97.4)	135.3 (90.1)

Table 8: Average residuals for the different schemes when finding the root of the monotonic function A under various noise levels. The root is $x^* = 0.07104$. The results are given in the format "average residuals (average residuals after sampling)" for each scheme. For CPL-AdS the sampling period is the estimation period (epoch 0) as defined by the scheme. The number of iterations per trial is 250.

Func B	Avg Regret. $\pi = 0.65$	Avg Regret. $\pi = 0.75$	Avg Regret $\pi = 0.85$
TS-SPL	47.1	**17.8**	**8.5**
TS-SPL-INF	53.8 (22.9)	39.5 (8.5)	35.1 (4.1)
PBS-M	**41.1** (10.2)	35.3 (4.31)	33.3 (2.3)
SGBS-M	50.6 (19.7)	35.7 (4.7)	33.0 (2.0)
BZ-M	60.3 (29.4)	39.6 (8.7)	33.6 (2.6)
SA	175.1	204.5	223.3
ASS	81.6 (50.6)	39.5 (8.7)	40.2 (9.0)
HSSL	85.3 (54.4)	50.6 (19.6)	39.0 (8.0)
SHSSL	75.4	30.8	12.7
CPL-AdS	117.9 (109.3)	116.7 (107.1)	144.9 (96.5)

Table 9: Average residuals for the different schemes when finding the root of the quadric function B under various noise levels. The root is $x^* = 0.9270$. The results are given in the format "average residuals (average residuals after sampling)" for each scheme. For CPL-AdS the sampling period is the estimation period (epoch 0) as defined by the scheme. The number of iterations per trial is 250.

Func C	Avg Regret. $\pi = 0.65$	Avg Regret. $\pi = 0.75$	Avg Regret $\pi = 0.85$
TS-SPL	**36.9**	**13.7**	**6.4**
TS-SPL-INF	52.8 (21.9)	38.8 (7.8)	34.9 (3.9)
PBS-M	49.6 (18.7)	39.2 (8.3)	34.3 (3.3)
SGBS-M	47.2 (16.2)	34.6 (3.6)	32.5 (1.6)
BZ-M	58.8 (27.9)	38.4 (7.4)	33.7 (2.7)
SA	149.0	178.0	185.0
ASS	54.3 (23.4)	39 (8.0)	33.5 (2.5)
HSSL	75.2 (44.3)	44.3 (13.4)	35.6 (4.6)
SHSSL	56.5	18.4	**6.4**
CPL-AdS	153.0 (101.6)	156.2 (103.7)	165.0 (109.6)

Table 10: Average residuals for the different schemes when finding the root of the sinusoidal function C under various noise levels. The root is $x^* = 0.8675$. The results are given in the format "average residuals (average residuals after sampling)" for each scheme. For CPL-AdS the sampling period is the estimation period (epoch 0) as defined by the scheme. The number of iterations per trial is 250.

converge towards the root x^*. A perhaps even stronger advantage of TS-SPL is that it can be applied to a wide range of functions without regards to the presence of local extrema. SA, on the other, only performs well for monotonic functions as exemplified in Table 8.

4. Conclusions and Further Work

In this paper, we investigated a novel reinforcement learning problem derived from the so-called "N-Door Puzzle". This puzzle has the fascinating property that it involves stochastic *compulsive liars*. Feedback is erroneous on average, systematically misleading the decision maker. This renders traditional reinforcement learning (RL) based approaches ineffective due to their dependency on *"on average"* correct feedback.

To solve the problem of deceptive feedback, we recast the problem as a challenging variant of the multi-armed bandit problem, referred to as the *Stochastic Point Location* (SPL) problem. In SPL, the decision maker is only told whether the optimal point on a line lies to the "left" or to the "right" of a current guess, with the feedback being erroneous with probability $1-\pi$. Solving this problem opens up for optimization in continuous action spaces with both *informative* and *deceptive* feedback.

Our solution to the above problem, introduced in the present paper, is based on a novel Bayesian representation of the solution space that is both compact and scalable. This model simultaneously captures both the location of the optimal point, as well as the probability of receiving correct feedback π. We further introduced an accompanying Thompson Sampling (TS) guided Stochastic Point Location (TS-SPL) scheme for balancing exploration against exploitation. By learning π, TS-SPL supports deceptive environments that are lying about the direction of the optimal point.

We used TS-SPL to solve the Stochastic Point Location (SPL) problem and outperformed all of the Learning Automata driven methods. However, by enhancing the Soft Generalized Binary Search (SGBS) scheme with our Bayesian representation of the solution space, SGBS was able to outperform TS-SPL under informative feedback. For deceptive SPL problems, TS-SPL outperformed all of the existing state-of-art schemes by several orders of magnitude, even when the latter schemes were supported by our Bayesian model.

We also applied TS-SPL to the *Stochastic Root Finding* (SRF) problem. We further demonstrated that SRF can be seen as a deceptive problem, allowing TS-SPL to outperform existing dedicated state-of-art SRF schemes by an order of magnitude. Thus, TS-SPL can be considered state-of-the-art for both deceptive SPL and for SRF, while yielding comparable results to the top performing schemes in the case of informative SPLs.

Despite the above performance gains, TS-SPL is based on Thompson Sampling, which is known to have a tendency to over-explore high variance reward distributions (Lattimore, 2015). In future work, it is therefore interesting to investigate mechanisms that eliminate or reduce this tendency, to further increase convergence speed.

Another important avenue for future work is the establishment of theoretical results, including proof of convergence, to corroborate the purely empirical findings presented in this paper. We suggest that a promising starting point for such an endeavour would be to combine the theoretical properties of TS (Agrawal and Goyal, 2012, 2013a; Dong and Van Roy, 2018) with the theoretical results of PBS (Waeber et al., 2013; Frazier et al., 2019), as they are closely related.

References

Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory*, pages 39–1, 2012.

Shipra Agrawal and Navin Goyal. Further optimal regret bounds for thompson sampling. In *Proceedings of the 16th Conference on Artificial Intelligence and Statistics*, pages 99–107, 2013a.

Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 127–135, 2013b.

Soren Asmussen and Peter W Glynn. *Stochastic simulation: Algorithms and Analysis*, volume 57. Springer Science & Business Media, 2007.

Onur Atan, Cem Tekin, and Mihaela van der Schaar. Global multi-armed bandits with hölder continuity. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, pages 28–36, 2015.

Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In *Conference on Learning Theory*, pages 13–p, 2010.

Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Machine Learning*, 5(1):1–122, 2012.

Marat Valievich Burnashev and Kamil'Shamil'evich Zigangirov. An interval estimation problem for controlled observations. *Problemy Peredachi Informatsii*, 10(3):51–61, 1974.

Rui M Castro and Robert D Nowak. Upper and lower error bounds for active learning. In *In Proceedings of the 44th Conference on Communication, Control and Computing*, volume 2, page 1, 2006.

Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Proceedings of the Advances in Neural Information Processing Systems 24*, pages 2249–2257. Curran Associates, Inc., 2011.

Huifen Chen and Bruce W Schmeiser. Stochastic root finding via retrospective approximation. *IIE Transactions*, 33(3):259–275, 2001.

Shi Dong and Benjamin Van Roy. An information-theoretic analysis for thompson sampling with many actions. In *Proceedings of the Advances in Neural Information Processing Systems 31*, pages 4157–4165, 2018.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research*, 7(Jun):1079–1105, 2006.

Peter I Frazier, Shane G Henderson, and Rolf Waeber. Probabilistic bisection converges almost as quickly as stochastic approximation. *Mathematics of Operations Research*, 2019.

Victor Gabillon, Mohammad Ghavamzadeh, Alessandro Lazaric, and Sébastien Bubeck. Multi-bandit best arm identification. In *Proceedings of the Advances in Neural Information Processing Systems 24*, pages 2222–2230, 2011.

Sondre Glimsdal and Ole-Christoffer Granmo. Gaussian process based optimistic knapsack sampling with applications to stochastic resource allocation. In *Proceedings of the 24th Midwest Artificial Intelligence and Cognitive Science Conference 2013*, pages 43–50. CEUR Workshop Proceedings, 2013.

Ole-Christoffer Granmo. Solving two-armed bernoulli bandit problems using a bayesian learning automaton. *International Journal of Intelligent Computing and Cybernetics*, 3 (2):207–234, 2010.

Ole-Christoffer Granmo and Sondre Glimsdal. Accelerated bayesian learning for decentralized two-armed bandit based decision making with applications to the goore game. *Applied intelligence*, 38(4):479–488, 2013.

Ole-Christoffer Granmo, B John Oommen, Svein Arild Myrer, and Morten Goodwin Olsen. Learning Automata-based Solutions to the Nonlinear Fractional Knapsack Problem with Applications to Optimal Resource Allocation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 37(1):166–175, 2007.

Michael Horstein. Sequential transmission using noiseless feedback. *IEEE Transactions on Information Theory*, 9(3):136–143, 1963.

Laurent Hyafil and Ronald L Rivest. Constructing optimal binary decision trees is np-complete. *Information Processing Letters*, 5(1):15–17, 1976.

Kevin Jamieson, Matthew Malloy, Robert Nowak, and Sébastien Bubeck. lil'ucb: An optimal exploration algorithm for multi-armed bandits. In *Conference on Learning Theory*, volume 35, pages 423–439, 2014.

Y Yu Jia and Shie Mannor. Unimodal bandits. In *International Conference on Machine Learning*, pages 41–48, 2011.

Lei Jiao, Xuan Zhang, B. John Oommen, and Ole-Christoffer Granmo. Optimizing channel selection for cognitive radio networks using a distributed bayesian learning automata-based approach. *Applied Intelligence*, 44(2):307–321, 2016. ISSN 1573-7497.

Zohar Shay Karnin, Tomer Koren, and Oren Somekh. Almost optimal exploration in multi-armed bandits. *International Conference on Machine Learning*, 28:1238–1246, 2013.

Jack Kiefer and Jacob Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.

HJ Kushner and G Yin. Stochastic approximation algorithms for parallel and distributed processing. *Stochastics: An International Journal of Probability and Stochastic Processes*, 22(3-4):219–250, 1987.

Tze Leung Lai. Stochastic approximation. *Annals of Statistics*, pages 391–406, 2003.

Tor Lattimore. Optimally confident ucb: Improved regret for finite-armed bandits. *arXiv preprint arXiv:1507.07880*, 2015.

Robert Nowak. Generalized binary search. In *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, pages 568–574. IEEE, 2008.

Robert D Nowak. The geometry of generalized binary search. *Information Theory, IEEE Transactions on*, 57(12):7893–7906, 2011.

B John Oommen. Stochastic Searching on the Line and its Applications to Parameter Learning in Nonlinear Optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 27(4):733–739, 1997.

B John Oommen, Govindachari Raghunath, and Benjamin Kuipers. On how to learn from a stochastic teacher or a stochastic compulsive liar of unknown identity. In *AI 2003: Advances in Artificial Intelligence*, pages 24–40. Springer, 2003.

B John Oommen, Sudip Misra, and Ole-Christoffer Granmo. Routing bandwidth-guaranteed paths in mpls traffic engineering: A multiple race track learning approach. *IEEE Transactions on Computers*, 56(7):959–976, 2007.

B John Oommen, Ole-Christoffer Granmo, and Zuoyuan Liang. A novel multidimensional scaling technique for mapping word-of-mouth discussions. In *Opportunities and Challenges for Next-Generation Applied Intelligence*, pages 317–322. Springer, 2009.

Stephen Pallone, Peter I Frazier, and Shane G Henderson. Multisection: Parallelized bisection. In *Simulation Conference, Winter*, pages 3773–3784. IEEE, 2014.

Raghu Pasupathy and Sujin Kim. The stochastic root-finding problem: Overview, solutions, and open questions. *ACM Transactions on Modeling and Computer Simulation*, 21(3): 19, 2011.

Raghu Pasupathy and Bruce W Schmeiser. Root finding via darts – dynamic adaptive random target shooting. In *Simulation Conference (WSC), Proceedings of the 2010 Winter*, pages 1255–1262. IEEE, 2010.

Andrzej Pelc. Searching games with errors – fifty years of coping with liars. *Theoretical Computer Science*, 270(1):71–109, 2002.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

Aarti Singh, Robert Nowak, and Parmesh Ramanathan. Active learning for adaptive mobile sensing networks. In *In Proceedings of the 5th international conference on Information processing in sensor networks*, pages 60–68. ACM, 2006.

Raymond Smullyan. *To Mock a Mockingbird and Other Logic Puzzles: Including an Amazing Adventure in Combinatory Logic*. Knopf, 1988. ISBN 0-19-280142-2.

Tongtong Tao, Hao Ge, Guixian Cai, and Shenghong Li. Adaptive step searching for solving stochastic point location problem. In *Intelligent Computing Theories*, pages 192–198. Springer, 2013.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

David Tolpin and Frank Wood. Maximum a posteriori estimation by search in probabilistic programs. In *Proceedings of the 8th Annual Symposium on Combinatorial Search*, 2015.

Rolf Waeber, Peter Frazier, and Shane G Henderson. A bayesian approach to stochastic root finding. In *Proceedings of the 2011 Winter Simulation Conference*, pages 4033–4045. IEEE, 2011.

Rolf Waeber, Peter I Frazier, and Shane G Henderson. Bisection search with noisy responses. *SIAM Journal on Control and Optimization*, 51(3):2261–2279, 2013.

Anis Yazidi, Ole-Christoffer Granmo, B John Oommen, and Morten Goodwin. A hierarchical learning scheme for solving the stochastic point location problem. In *Advanced Research in Applied Artificial Intelligence*, pages 774–783. Springer, 2012a.

Anis Yazidi, B John Oommen, and Ole-Christoffer Granmo. A novel stochastic discretized weak estimator operating in non-stationary environments. In *Proceedings of the International Conference on Computing, Networking and Communications*, pages 364–370. IEEE, 2012b.

Junqi Zhang, Liang Zhang, and Mengchu Zhou. Solving stationary and stochastic point location problem with optimal computing budget allocation. In *Proceedings of the 2015 IEEE International Conference on Systems, Man, and Cybernetics*, pages 145–150, Oct 2015. doi: 10.1109/SMC.2015.38.

Junqi Zhang, Yuheng Wang, Cheng Wang, and Mengchu Zhou. Symmetrical hierarchical stochastic searching on the line in informative and deceptive environments. *IEEE Transactions on Cybernetics*, PP(99):1–10, 2016. ISSN 2168-2267. doi: 10.1109/TCYB.2016.2521859.

Journal of Machine Learning Research 20 (2019) 1-32 Submitted 7/18; Revised 2/19; Published 3/19

Tunability: Importance of Hyperparameters of Machine Learning Algorithms

Philipp Probst PROBST@IBE.MED.UNI-MUENCHEN.DE
Institute for Medical Information Processing, Biometry and Epidemiology, LMU Munich
Marchioninistr. 15, 81377 München, Germany

Anne-Laure Boulesteix BOULESTEIX@IBE.MED.UNI-MUENCHEN.DE
Institute for Medical Information Processing, Biometry and Epidemiology, LMU Munich
Marchioninistr. 15, 81377 München, Germany

Bernd Bischl BERND.BISCHL@STAT.UNI-MUENCHEN.DE
Department of Statistics, LMU Munich
Ludwigstraße 33, 80539 München, Germany

Editor: Ryan Adams

Abstract

Modern supervised machine learning algorithms involve hyperparameters that have to be set before running them. Options for setting hyperparameters are default values from the software package, manual configuration by the user or configuring them for optimal predictive performance by a tuning procedure. The goal of this paper is two-fold. Firstly, we formalize the problem of tuning from a statistical point of view, define data-based defaults and suggest general measures quantifying the tunability of hyperparameters of algorithms. Secondly, we conduct a large-scale benchmarking study based on 38 datasets from the OpenML platform and six common machine learning algorithms. We apply our measures to assess the tunability of their parameters. Our results yield default values for hyperparameters and enable users to decide whether it is worth conducting a possibly time consuming tuning strategy, to focus on the most important hyperparameters and to choose adequate hyperparameter spaces for tuning.

Keywords: machine learning, supervised learning, classification, hyperparameters, tuning, meta-learning

1. Introduction

Machine learning (ML) algorithms such as gradient boosting, random forest and neural networks for regression and classification involve a number of *hyperparameters* that have to be set before running them. In contrast to direct, first-level model parameters, which are determined during training, these second-level *tuning parameters* often have to be carefully optimized to achieve maximal performance. A related problem exists in many other algorithmic areas, e.g., control parameters in evolutionary algorithms (Eiben and Smit, 2011).

In order to select an appropriate hyperparameter *configuration* for a specific dataset at hand, users of ML algorithms can resort to default values of hyperparameters that are specified in implementing software packages or manually configure them, for example, based on recommendations from the literature, experience or trial-and-error.

©2019 Philipp Probst, Anne-Laure Boulesteix and Bernd Bischl.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-444.html.

Alternatively, one can use hyperparameter *tuning strategies*, which are data-dependent, second-level optimization procedures (Guyon et al., 2010), which try to minimize the expected generalization error of the inducing algorithm over a hyperparameter search space of considered candidate configurations, usually by evaluating predictions on an independent test set, or by running a resampling scheme such as cross-validation (Bischl et al., 2012). For a recent overview of tuning strategies, see, e.g., Luo (2016). These search strategies range from simple grid or random search (Bergstra and Bengio, 2012) to more complex, iterative procedures such as Bayesian optimization (Hutter et al., 2011; Snoek et al., 2012; Bischl et al., 2017b) or iterated F-racing (Birattari et al., 2010; Lang et al., 2017).

In addition to selecting an efficient tuning strategy, the set of tunable hyperparameters and their corresponding ranges, scales and potential prior distributions for subsequent sampling have to be determined by the user. Some hyperparameters might be safely set to default values, if they work well across many different scenarios. Wrong decisions in these areas can inhibit either the quality of the resulting model or at the very least the efficiency and fast convergence of the tuning procedure. This creates a burden for:

1. ML users—Which hyperparameters should be tuned and in which ranges?

2. Designers of ML algorithms—How do I define robust defaults?

We argue that many users, especially if they do not have years of practical experience in the field, here often rely on heuristics or spurious knowledge. It should also be noted that designers of fully automated tuning frameworks face at least very similar problems. It is not clear how these questions should be addressed in a data-dependent, automated, optimal and objective manner. In other words, the scientific community not only misses answers to these questions for many algorithms but also a systematic framework, methods and criteria, which are required to answer these questions.

With the present paper we aim at filling this gap and formalize the problem of parameter tuning from a statistical point of view, in order to simplify the tuning process for less experienced users and to optimize decision making for more advanced processes.

After presenting related literature in Section 2, we define theoretical measures for assessing the impact of tuning in Section 3. For this purpose we (i) define the concept of *default hyperparameters*, (ii) suggest measures for quantifiying the tunability of the whole algorithm and specific hyperparameters based on the differences between the performance of default hyperparameters and the performance of the hyperparameters when this hyperparameter is set to an optimal value. Then we (iii) address the tunability of hyperparameter combinations and joint gains, (iv) provide theoretical definitions for an appropriate hyperparameter space on which tuning should be executed and (v) propose procedures to estimate these quantities based on the results of a benchmark study with random hyperparameter configurations with the help of surrogate models. In sections 4 and 5 we illustrate these concepts and methods through an application. For this purpose we use benchmark results of six machine learning algorithms with different hyperparameters which were evaluated on 38 datasets from the OpenML platform. Finally, in the last Section 6 we conclude and discuss the results.

2. Related Literature

To the best of our knowledge, only a limited amount of articles address the problem of tunability and generation of tuning search spaces. Bergstra and Bengio (2012) compute the relevance of the hyperparameters of neural networks and conclude that some are important on all datasets, while others are only important on some datasets. Their conclusion is primarily visual and used as an argument for why random search works better than grid search when tuning neural networks.

A specific study for decision trees was conducted by Mantovani et al. (2016) who apply standard tuning techniques to decision trees on 102 datasets and calculate differences of accuracy between the tuned algorithm and the algorithm with default hyperparameter settings.

A different approach is proposed by Hutter et al. (2013), which aims at identifying the most important hyperparameters via forward selection. In the same vein, Fawcett and Hoos (2016) present an *ablation analysis* technique, which aims at identifying the hyperparameters that contribute the most to improved performance after tuning. For each of the considered hyperparameters, they compute the performance gain that can be achieved by changing its value from the initial value to the value specified in the target configuration which was determined by the tuning strategy. This procedure is iterated in a greedy forward search.

A more general framework for measuring the importance of single hyperparameters is presented by Hutter et al. (2014). After having used a tuning strategy such as sequential model-based optimization, a functional ANOVA approach is used for measuring the importance of hyperparameters.

These works concentrate on the importance of hyperparameters on single datasets, mainly to retrospectively explain what happened during an already concluded tuning process. Our main focus is the generalization across multiple datasets in order to facilitate better general understanding of hyperparameter effects and better decision making for future experiments. In a recent paper van Rijn and Hutter (2017) pose very similar questions to ours to assess the importance of hyperparameters across datasets. We compare it to our approach in Section 6.

Our framework is based on using surrogate models, also sometimes called empirical performance models, which allow estimating the performance of arbitrary hyperparameter configurations based on a limited number of prior experiments. The idea of surrogate models is far from new (Audet et al., 2000), as it constitutes the central idea of Bayesian optimization for hyperparameter search but is also used, for example, in Biedenkapp et al. (2017) for increasing the speed of an ablation analysis and by Eggensperger et al. (2018) for speeding up the benchmarking of tuning strategies.

3. Methods for Estimation of Defaults, Tunability and Ranges

In this section we introduce theoretical definitions for defaults, tunability and tuning ranges, then describe how to estimate them and finally discuss the topic of reparametrization.

3.1. General Notation

Consider a target variable Y, a feature vector X, and an unknown joint distribution P on (X, Y), from which we have sampled a dataset $\mathcal{T}$ of n observations. A machine learning (ML) algorithm now learns the functional relationship between X and Y by producing a prediction model $\hat{f}(X, \theta)$, controlled by the k-dimensional hyperparameter configuration $\theta = (\theta_1, ..., \theta_k)$ from the hyperparameter search space $\Theta = \Theta_1 \times ... \times \Theta_k$. In order to measure prediction performance pointwise between the true label Y and its prediction $\hat{f}(X, \theta)$, we define a loss function $L(Y, \hat{f}(X, \theta))$. We are naturally interested in estimating the expected risk of the inducing algorithm, w.r.t. θ on new data, also sampled from $\mathcal{P}$: $R(\theta) = E(L(Y, \hat{f}(X, \theta))|\mathcal{P})$. This mapping encodes, given a certain data distribution, a certain learning algorithm and a certain performance measure, the numerical quality for any hyperparameter configuration θ. Given m different datasets (or data distributions) $\mathcal{P}_1, ..., \mathcal{P}_m$, we arrive at m hyperparameter risk mappings

$$R^{(j)}(\theta) := E(L(Y, \hat{f}(X, \theta))|\mathcal{P}_j), \qquad j = 1, ..., m. \tag{1}$$

For now, we assume all $R^{(j)}(\theta)$ to be known, and show how to estimate them in Section 3.7.

3.2. Optimal Configuration per Dataset and Optimal Defaults

We first define the best hyperparameter configuration for dataset j as

$$\theta^{(j)\star} := \arg\min_{\theta \in \Theta} R^{(j)}(\theta). \tag{2}$$

Defaults settings are supposed to work well across many different datasets and are usually provided by software packages, in an often ad hoc or heuristic manner. We propose to define an optimal default configuration, based on an extensive number of empirical experiments on m different benchmark datasets, by

$$\theta^{\star} := \arg\min_{\theta \in \Theta} g(R^{(1)}(\theta), ..., R^{(m)}(\theta)). \tag{3}$$

Here, g is a summary function that has to be specified. Selecting the mean (or median as a more robust candidate) would imply minimizing the average (or median) risk over all datasets.

The measures $R^{(j)}(\theta)$ could potentially be scaled appropriately beforehand in order to make them more commensurable between datasets, e.g., one could scale all $R^{(j)}(\theta)$ to $[0, 1]$ by subtracting the result of a very simple baseline like a featureless dummy predictor and dividing this difference by the absolute difference between the risk of the best possible result (as an approximation of the Bayes error) and the result of the very simple baseline predictor. Or one could produce a statistical z-score by subtracting the mean and dividing by the standard deviation from all experimental results on the same dataset (Feurer et al., 2018).

The appropriateness of the scaling highly depends on the performance measure that is used. One could, for example, argue that the AUC does not have to be scaled by using the probabilistic interpretation of the AUC. Given a randomly chosen observation x belonging to class 1, and a randomly chosen observation x' belonging to class 0, the AUC is the probability that the evaluated classification algorithm will assign a higher score to x than to x'. Thus,

an improvement from 0.5 to 0.6 on one dataset could be seen as equally important to an improvement from 0.8 to 0.9 on another dataset. On the other hand, averaging the mean squared error on several datasets does not make a lot of sense, as the scale of the outcome of different regression problems can be very different. Then scaling or using another measure such as the R^2 is reasonable. As our main risk measure is the AUC, we do not use any scaling.[1]

3.3. Measuring Overall Tunability of a ML Algorithm

A general measure of the tunability of an algorithm per dataset can then be computed based on the difference between the risk of an overall reference configuration (e.g., either the software defaults or definition (3)) and the risk of the best possible configuration on that dataset:

$$d^{(j)} := R^{(j)}(\theta^\star) - R^{(j)}(\theta^{(j)\star}), \text{ for } j = 1, ..., m. \tag{4}$$

For each algorithm, this gives rise to an empirical distribution of performance differences over datasets, which might be directly visualized or summarized to an aggregated tunability measure d by using mean, median or quantiles.

3.4. Measuring Tunability of a Specific Hyperparameter

The best hyperparameter value for one parameter i on dataset j, when all other parameters are set to defaults from $\theta^\star := (\theta_1^\star, ..., \theta_k^\star)$, is denoted by

$$\theta_i^{(j)\star} := \underset{\theta \in \Theta, \theta_l = \theta_l^\star \forall l \neq i}{\arg\min} R^{(j)}(\theta). \tag{5}$$

A natural measure for tunability of the i-th parameter on dataset j is then the difference in risk between the above and our default reference configuration:

$$d_i^{(j)} := R^{(j)}(\theta^\star) - R^{(j)}(\theta_i^{(j)\star}), \text{ for } j = 1, ..., m, i = 1, ..., k. \tag{6}$$

Furthermore, we define $d_i^{(j),\text{rel}} = \frac{d_i^{(j)}}{d^{(j)}}$ as the fraction of performance gain, when we only tune parameter i compared to tuning the complete algorithm, on dataset j. Again, one can calculate the mean, the median or quantiles of these two differences over the n datasets, to get a notion of the overall tunability d_i of this parameter.

3.5. Tunability of Hyperparamater Combinations and Joint Gains

Let us now consider two hyperparameters indexed as i_1 and i_2. To measure the tunability with respect to these two parameters, we define

$$\theta_{i_1,i_2}^{(j)\star} := \underset{\theta \in \Theta, \theta_l = \theta_l^\star \forall l \notin \{i_1, i_2\}}{\arg\min} R^{(j)}(\theta), \tag{7}$$

1. We also tried out normalization (z-score) and got qualitatively similar results to the non-normalized results presented in Section 5.

i.e., the θ-vector containing the default values for all hyperparameters other than i_1 and i_2, and the optimal combination of values for the i_1-th and i_2-th components of θ.

Analogously to the previous section, we can now define the tunability of the set (i_1, i_2) as the gain over the reference default on dataset j as

$$d_{i_1,i_2}^{(j)} := R^{(j)}(\theta^*) - R^{(j)}(\theta_{i_1,i_2}^{(j)\star}). \tag{8}$$

The joint gain which can be expected when tuning not only one of the two hyperparameters individually, but both of them jointly, on a dataset j, can be expressed by

$$g_{i_1,i_2}^{(j)} := \min\{(R^{(j)}(\theta_{i_1}^{(j)\star})), (R^{(j)}(\theta_{i_2}^{(j)\star}))\} - R^{(j)}(\theta_{i_1,i_2}^{(j)\star}). \tag{9}$$

Furthermore, one could be interested in whether this joint gain could simply be reached by tuning both parameters i_1 and i_2 in a univariate fashion sequentially, either in the order $i_1 \to i_2$ or $i_2 \to i_1$, and what order would be preferable. For this purpose one could compare the risk of the hyperparameter value that results when tuning them together $R^{(j)}(\theta_{i_1,i_2}^{(j)\star})$ with the risks of the hyperparameter values that are obtained when tuning them sequentially, that means $R^{(j)}(\theta_{i_1 \to i_2}^{(j)\star})$ or $R^{(j)}(\theta_{i_2 \to i_1}^{(j)\star})$, which is done for example in Waldron et al. (2011).

Again, all these measures should be summarized across datasets, resulting in d_{i_1,i_2} and g_{i_1,i_2}. Of course, these approaches can be further generalized by considering combinations of more than two parameters.

3.6. Optimal Hyperparameter Ranges for Tuning

A reasonable hyperparameter space $\Theta^\star$ for tuning should include the optimal configuration $\theta^{(j)\star}$ for dataset j with high probability. We denote the p-quantile of the distribution of one parameter regarding the best hyperparameters on each dataset $(\theta^{(1)\star})_i, ..., (\theta^{(m)\star})_i$ as $q_{i,p}$. The hyperparameter tuning space can then be defined as

$$\Theta^\star := \{\theta \in \Theta | \forall i \in \{1, ..., k\} : \theta_i \geq q_{i,p_1} \wedge \theta_i \leq q_{i,p_2}\}, \tag{10}$$

with p_1 and p_2 being quantiles which can be set for example to the 5 % quantile and the 95 % quantile. This avoids focusing too much on outlier datasets and makes the definition of the space independent from the number of datasets.

The definition above is only valid for numerical hyperparameters. In case of categorical variables one could use similar rules, for example only including hyperparameter values that were at least once or in at least 10 % of the datasets the best possible hyperparameter setting.

3.7. Practical Estimation

In order to practically apply the previously defined concepts, two remaining issues need to be addressed: a) We need to discuss how to obtain $R^{(j)}(\theta)$; and b) in (2) and (3) a multivariate optimization problem (the minimization) needs to be solved.[2]

2. All other previous optimization problems are univariate or two-dimensional and can simply be addressed by simple techniques such as a fine grid search.

For a) we estimate $R^{(j)}(\theta)$ by using surrogate models $\hat{R}^{(j)}(\theta)$, and replace the original quantity by its estimator in all previous formulas. Surrogate models for each dataset j are based on a meta dataset. This is created by evaluating a large number of configurations of the respective ML method. The surrogate regression model then learns to map a hyperparameter configuration to estimated performance. For b) we solve the optimization problem—now cheap to evaluate, because of the surrogate models—through black-box optimization.

3.8. Reparametrization

All tunability measures mentioned above can possibly depend on and be influenced by a reparametrization of hyperparameters. For example, in the case of the elastic net the parameters λ and α could be reparametrized as $\lambda_1 = \alpha\lambda$ and $\lambda_2 = (1-\alpha)\lambda$. Formally, such a reparametrization could be defined as a (bijective) function $\phi : \Theta \to \tilde{\Theta}$, such that $\phi(\theta)$ maps an original configuration θ to a new representation $\tilde{\theta} = \phi(\theta)$ from $\tilde{\Theta}$, in a one-to-one manner. Then defaults (calculated by the approach in Section 3.2) are naturally transformed via $\tilde{\theta}^\star = \phi(\theta^\star)$ into the new space $\tilde{\Theta}$, but will stay logically the same. Moreover, the general tunability of the algorithm does (obviously) not change. Depending on the parameters that are involved in the reparametrization, the tunability of the parameters can change. If, for example, only one parameter is involved, all tunabilities remain the same. If two or more parameters are involved, the single tunabilities of the parameters could change but the tunability of the set of the transformed parameters remains the same.

One might define a reparametrization as ideal (in the sense of simplified tuning) if the tunability is concentrated on one (or only few) hyperparameter(s), so that only this parameter has to be optimized and all remaining hyperparameters can remain at their (optimal) default values, reducing a multivariate optimization problem to a 1-dimensional or at least lower dimensional one. Using the definition above, this would imply that the joint gain of the new parameter(s) is (close to) 0. For example, imagine that the optimal hyperparameter values per dataset of two hyperparameters θ_1 and θ_2 lie on the line of equation $\theta_1 = \theta_2$. A useful reparametrization would then be $\tilde{\theta}_1 = \theta_1 + \theta_2$ and the orthogonal $\tilde{\theta}_2 = \theta_1 - \theta_2$. It would then only be necessary to tune $\tilde{\theta}_1$, while $\tilde{\theta}_2$ would be set to the default value 0.

A more general formulation is possible if we use the definition of 3.4. We could, for example, search for a bijective and invertible function $\phi^\star(.)$, across a certain parameterized function space, such that the mean tunability is concentrated on and therefore maximal for the first parameter and minimal for the other parameters, i.e.:

$$\phi^\star := \arg\min_{\phi \in \Phi} \frac{1}{m} \sum_{j=1}^{m} \min_{\tilde{\theta} \in \tilde{\Theta}, \tilde{\theta}_l = \tilde{\theta}_l^\star \forall l \neq 1} R^{(j)} \left(\phi^{-1} \left(\tilde{\theta} \right) \right). \tag{11}$$

We could select a restricted function space for ϕ, e.g., restrict ourselves to the space of all linear (invertible) transformations $\{\phi : \mathbb{R}^k \to \mathbb{R}^k | \phi(x) = Ax, A \in \mathbb{R}^{k \times k}, \det(A) \neq 0\}$. If concentrating the whole tunability on only one parameter is not possible, we could try a similar approach by concentrating it on a combination of two hyperparameters.

Note that such a reparametrization is not always helpful. For example, imagine we have two binary parameters and transform them such that (i) one of them has 4 levels that correspond to all possible combinations of these two parameters and (ii) the other parameter is set to a fixed constant. This reparametrization would not be useful: all the tunability is contained in the first parameter, but there is no real advantage, as still four evaluations have to be executed in the tuning process to get the best hyperparameter combination.

Finally, note that it can also be useful to reparametrize a single hyperparameter for the purpose of tuning. Imagine, for example, that most of the optimal parameters on the different datasets are rather small and only a few are large. A transformation of this parameter such as a log-transformation may then be useful. This is very similar to using prior probabilities for tuning (based on results on previous datasets) which could be seen as a useful alternative to a reparametrization and which is already proposed in van Rijn and Hutter (2017).

4. Experimental Setup

In this section we give an overview about the experimental setup that is used for obtaining surrogate models, tunability measures and tuning spaces.

4.1. Datasets from the OpenML Platform

Recently, the OpenML project (Vanschoren et al., 2013) has been created as a flexible online platform that allows ML scientists to share their data, corresponding tasks and results of different ML algorithms. We use a specific subset of carefully curated classification datasets from the OpenML platform called *OpenML100* (Bischl et al., 2017a). For our study we only use the 38 binary classification tasks that do not contain any missing values.

4.2. ML Algorithms

The algorithms considered in this paper are common methods for supervised learning. We examine elastic net (`glmnet` R package), decision tree (`rpart`), k-nearest neighbors (`kknn`), support vector machine (`svm`), random forest (`ranger`) and gradient boosting (`xgboost`). For more details about the used software packages see Kühn et al. (2018b). An overview of their considered hyperparameters is displayed in Table 1, including respective data types, box-constraints and a potential transformation function.

In the case of `xgboost`, the underlying package only supports numerical features, so we opted for a dummy feature encoding for categorical features, which is performed internally by the underlying packages for `svm` and `glmnet`.

Some hyperparameters of the algorithms are dependent on others. We take into account these dependencies and, for example, only sample a value for `gamma` for the support vector machine if the radial kernel was sampled beforehand.

4.3. Performance estimation

Several measures are regarded throughout this paper, either for evaluating our considered classification models that should be tuned, or for evaluating our surrogate regression models. As no optimal measure exists, we will compare several of them. In the classification case, we consider AUC, accuracy and Brier score. In the case of surrogate regression, we consider R^2, which is directly proportional to the regular mean squared error but scaled to [0,1] and explains the gain over a constant model estimating the overall mean of all data points. We also compute Kendall's tau as a ranking based measure for regression.

Algorithm	Hyperparameter	Type	Lower	Upper	Trafo
glmnet					
(Elastic net)	alpha	numeric	0	1	-
	lambda	numeric	-10	10	2^x
rpart					
(Decision tree)	cp	numeric	0	1	-
	maxdepth	integer	1	30	-
	minbucket	integer	1	60	-
	minsplit	integer	1	60	-
kknn					
(k-nearest neighbor)	k	integer	1	30	-
svm					
(Support vector machine)	kernel	discrete	-	-	-
	cost	numeric	-10	10	2^x
	gamma	numeric	-10	10	2^x
	degree	integer	2	5	-
ranger					
(Random forest)	num.trees	integer	1	2000	-
	replace	logical	-	-	-
	sample.fraction	numeric	0.1	1	-
	mtry	numeric	0	1	$x \cdot p$
	respect.unordered.factors	logical	-	-	-
	min.node.size	numeric	0	1	n^x
xgboost					
(Gradient boosting)	nrounds	integer	1	5000	-
	eta	numeric	-10	0	2^x
	subsample	numeric	0.1	1	-
	booster	discrete	-	-	-
	max_depth	integer	1	15	-
	min_child_weight	numeric	0	7	2^x
	colsample_bytree	numeric	0	1	-
	colsample_bylevel	numeric	0	1	-
	lambda	numeric	-10	10	2^x
	alpha	numeric	-10	10	2^x

Table 1: Hyperparameters of the algorithms. p refers to the number of variables and n to the number of observations. The columns *Lower* and *Upper* indicate the regions from which samples of these hyperparameters are drawn. The transformation function in the trafo column, if any, indicates how the values are transformed according to this function. The exponential transformation is applied to obtain more candidate values in regions with smaller hyperparameters because for these hyperparameters the performance differences between smaller values are potentially bigger than for bigger values. The `mtry` value in `ranger` that is drawn from $[0, 1]$ is transformed for each dataset separately. After having chosen the dataset, the value is multiplied by the number of variables and afterwards rounded up. Similarly, for the `min.node.size` the value x is transformed by the formula $[n^x]$ with n being the number of observations of the dataset, to obtain a positive integer values with higher probability for smaller values (the value is finally rounded to obtain integer values).

The performance estimation for the different hyperparameter experiments is computed through 10-fold cross-validation. For the comparison of surrogate models 10 times repeated 10-fold cross-validation is used.

4.4. Random Bot sampling strategy for meta data

To reliably estimate our surrogate models we need enough evaluated configurations per classifier and dataset. We sample these points from independent uniform distributions where the respective support for each parameter is displayed in Table 1. Here, *uniform* refers to the untransformed scale, so we sample uniformly from the interval [*Lower*, *Upper*] of Table 1.

In order to properly facilitate the automatic computation of a large database of hyperparameter experiments, we implemented a so called OpenML bot. In an embarrassingly parallel manner it chooses in each iteration a random dataset, a random classification algorithm, samples a random configuration and evaluates it via cross-validation. A subset of 500000 experiments for each algorithm and all datasets are used for our analysis here.[3] More technical details regarding the random bot, its setup and results can be obtained in Kühn et al. (2018b), furthermore, for simple and permanent access the results of the bot are stored in a figshare repository (Kühn et al., 2018a).

4.5. Optimizing Surrogates to Obtain Optimal Defaults

Random search is also used for our black-box optimization problems in Section 3.7. For the estimation of the defaults for each algorithm we randomly sample 100000 points in the hyperparameter space as defined in Table 1 and determine the configuration with the minimal average risk. The same strategy with 100000 random points is used to obtain the best hyperparameter setting on each dataset that is needed for the estimation of the tunability of an algorithm. For the estimation of the tunability of single hyperparameters we also use 100000 random points for each parameter, while for the tunability of combination of hyperparameters we only use 10000 random points to reduce runtime as this should be enough to cover 2-dimensional hyperparameter spaces.

Of course one has to be careful with overfitting here, as our optimal defaults are chosen with the help of the same datasets that are used to determine the performance. Therefore, we also evaluate our approach via a "10-fold cross-validation across datasets". Here, we repeatedly calculate the optimal defaults based on 90% "training datasets" and evaluate the package defaults and our optimal defaults—the latter induced from the training datasets—on the surrogate models of the remaining 10% "test datasets", and compare their difference in performance.

4.6. The Problem of Hyperparameter Dependency

Some parameters are dependent on other superordinate hyperparameters and are only relevant if the parameter value of this superordinate parameter was set to a specific value. For example `gamma` in `svm` only makes sense if the `kernel` was set to "radial" or `degree` only makes sense if the kernel was set to "polynomial". Some of these subordinate parameters might be invalid/inactive in the reference default configuration, rendering it impossible to

3. Only 30 experiments are used for each dataset for `kknn`, because we only consider the parameter k.

univariately tune them in order to compute their tunability score. In such a case we set the superordinate parameter to a value which makes the subordinate parameter active, compute the optimal defaults for the rest of the parameters and compute the tunability score for the subordinate parameter with these defaults.

4.7. Software Details

All our experiments are executed in R and are run through a combination of custom code from our random bot (Kühn et al., 2018b), the `OpenML` R package (Casalicchio et al., 2017), `mlr` (Bischl et al., 2016) and `batchtools` (Lang et al., 2017) for parallelization. All results are uploaded to the OpenML platform and there publicly available for further analysis. `mlr` is also used to compare and fit all surrogate regression models. The fully reproducible R code for all computations and analyses of our paper can be found on the github page: `https://github.com/PhilippPro/tunability`. We also provide an interactive shiny app under `https://philipppro.shinyapps.io/tunability/`, which displays all results of the following section in a potentially more convenient, interactive fashion and which can simply be accessed through a web browser.

5. Results and Discussion

We calculate all results for AUC, accuracy and Brier score but mainly discuss AUC results here. Tables and figures for the other measures can be accessed in the appendix and in our interactive shiny application.

5.1. Surrogate Models

We compare different possible regression models as candidates for our surrogate models: the linear model (`lm`), a simple decision tree (`rpart`), k nearest-neighbors (`kknn`) and random forest (`ranger`)[4] All algorithms are run with their default settings. We calculate 10 times repeated 10-fold cross-validated regression performance measures R^2 and Kendall's tau per dataset, and average these across all datasets.[5] Results for AUC are displayed in Figure 1. A good overall performance is achieved by `ranger` with qualitatively similar results for other classification performance measures (see Appendix). In the following we use random forest as surrogate model because it performs reasonably well and is already an established algorithm for surrogate models in the literature (Eggensperger et al., 2014; Hutter et al., 2013).

5.2. Optimal Defaults and Tunability

Table 2 displays our mean tunability results for the algorithms as defined in formula (4) w.r.t. package defaults (`Tun.P` column) and our optimal defaults (`Tun.O`). The distribution of the tunability values of the optimal defaults can be seen in Figure 2 in the modified

4. We also tried `cubist` (Kuhn et al., 2016), which provided good results but the algorithm had some technical problems for some combinations of datasets and algorithms. We did not include gaussian process which is one of the standard algorithms for surrogate models as it cannot handle categorical variables.
5. In case of `kknn` four datasets did not provide results for one of the surrogate models and were not used.

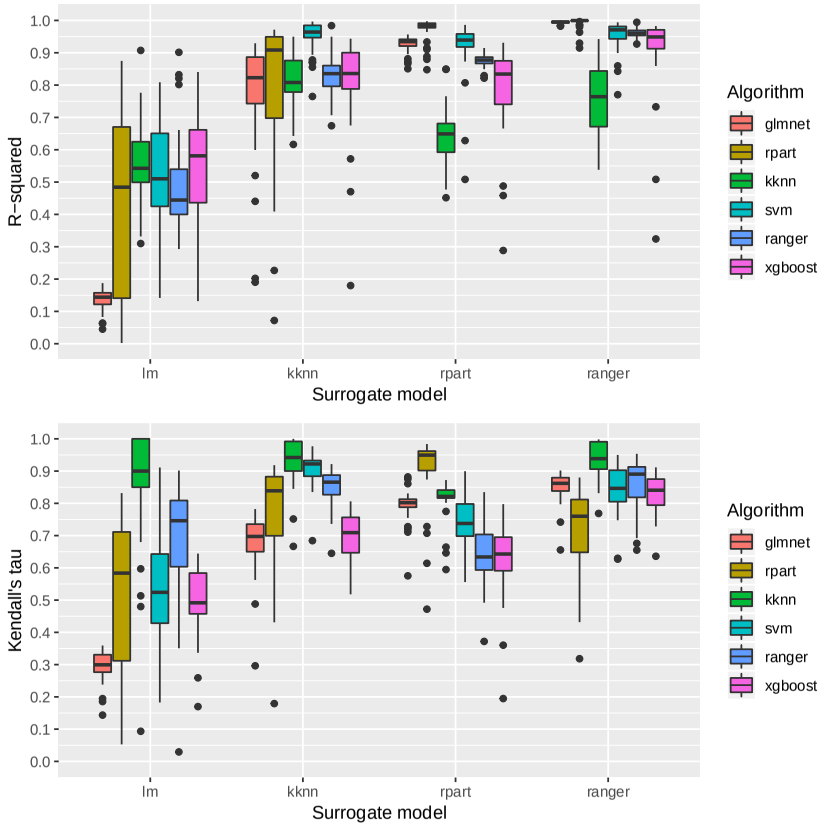


Figure 1: Average performances over the datasets of different surrogate models (target: AUC) for different algorithms (that were presented in 4.2). For an easier comparison of the surrogate models the same graph with exchanged x-axis and legend is available in the appendix in Figure 5.

Algorithm	Tun.P	Tun.O	Tun.O-CV	Improv	Impr-CV
glmnet	0.069 ± 0.019	0.024 ± 0.013	0.037 ± 0.015	0.045 ± 0.015	0.032 ± 0.015
rpart	0.038 ± 0.006	0.012 ± 0.004	0.016 ± 0.004	0.025 ± 0.006	0.022 ± 0.006
kknn	0.031 ± 0.006	0.006 ± 0.004	0.006 ± 0.004	0.025 ± 0.008	0.025 ± 0.008
svm	0.056 ± 0.011	0.042 ± 0.007	0.048 ± 0.008	0.014 ± 0.005	0.008 ± 0.007
ranger	0.010 ± 0.003	0.006 ± 0.001	0.007 ± 0.001	0.004 ± 0.003	0.003 ± 0.003
xgboost	0.043 ± 0.006	0.014 ± 0.006	0.017 ± 0.007	0.029 ± 0.003	0.026 ± 0.003

Table 2: Mean tunability (regarding AUC) with the package defaults (Tun.P) and the optimal defaults (Tun.O) as reference, cross-validated tunability (Tun.O-CV), average improvement (Improv) and cross-validated average improvement (Impr-CV) obtained by using optimal defaults compared to package defaults. The (cross-validated) improvement can be calculated by the (rounded) difference between Tun.P and Tun.O (Tun.O-CV). Standard error of the mean (SEM) is given behind the "±"-sign.

boxplots. Table 2 also displays the average improvement per algorithm when moving from package defaults to optimal defaults (`Improv`), which was positive overall. This also holds for `svm` and `ranger` although the package defaults are data dependent, which we currently cannot model (`gamma` $= 1/p$ for `svm` and `mtry` $= \sqrt{p}$ for `ranger`). As our optimal defaults are calculated based on all datasets, there is a risk of overfitting. So we perform a 5-fold cross-validation on dataset level, always calculating optimal defaults on $\frac{4}{5}$ of datasets and evaluating them on $\frac{1}{5}$ of the datasets. The results in the column `Impr-CV` in Table 2 show that the improvement compared to the package defaults is less pronounced but still positive for all algorithms.

From now on, when discussing tunability, we will only do this w.r.t. our optimal defaults.

Clearly, some algorithms such as `glmnet` and `svm` are much more tunable than the others, while `ranger` is the algorithm with the smallest tunability, which is in line with common knowledge in the web community. In the boxplots in Figure 2 for each ML algorithm, some values that are much bigger than the others are visible, which indicates that tuning has a much higher impact on some specific datasets.

5.3. Tunability of Specific Hyperparameters

In Table 3 the mean tunability (regarding the AUC) of single hyperparameters as defined in Equation (6) in Section 3.4 can be seen. Moreover, in Figure 3 the distributions of the tunability values of the hyperparameters are depicted in boxplots, which makes it possible to detect outliers and to examine skewness. The same results for the Brier score and accuracy can be found in the appendix. In the following analysis of our results, we will refer to tunability only with respect to optimal defaults.

For `glmnet` `lambda` seems to be more tunable than `alpha` regarding the AUC, especially for two datasets tuning seems to be crucial. For accuracy we observe the same pattern, while for Brier score `alpha` seems to be more tunable than `lambda` (see Figure 11 and Figure 13 in the appendix). We could not find any recommendation in the literature for

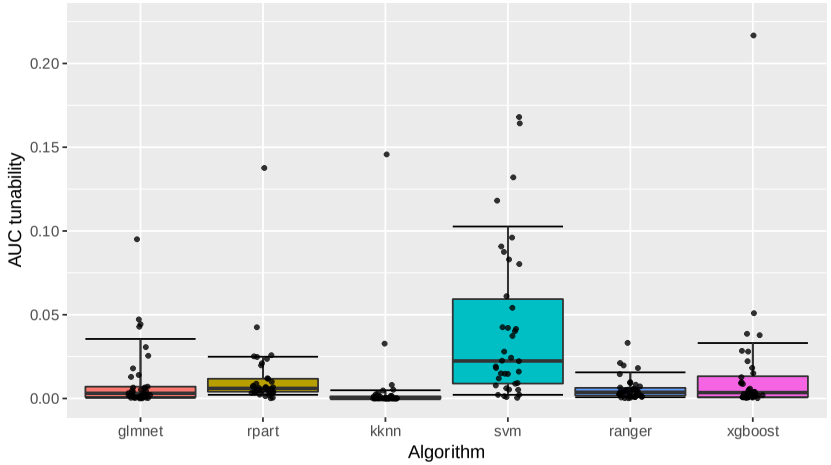


Figure 2: Boxplots of the tunabilities (AUC) of the different algorithms with respect to optimal defaults. The upper and lower whiskers (upper and lower line of the boxplot rectangle) are in our case defined as the 0.1 and 0.9 quantiles of the tunability scores. The 0.9 quantile indicates how much performance improvement can be expected on at least 10% of datasets. One outlier of `glmnet` (value 0.5) is not shown.

these two parameters. In `rpart` the `minbucket` and `minsplit` parameters seem to be the most important ones for tuning which is in line with the analysis of Mantovani et al. (2018). `k` in the `kknn` algorithm is very tunable w.r.t. package defaults, but not regarding optimal defaults. Note that the optimal default is 30 which is at the boundary of possible values, so possibly bigger values can provide further improvements. A classical suggestion in the literature (Lall and Sharma, 1996) is to use $\sqrt{n}$ as default value. This is in line with our results, as the number of observations is bigger than 900 in most of our datasets.

In `svm` the biggest gain in performance can be achieved by tuning the `kernel`, `gamma` or `degree`, while the `cost` parameter does not seem to be very tunable. To the best of our knowledge, this has not been noted in the literature yet. In `ranger` `mtry` is the most tunable parameter which is already common knowledge and is implemented in software packages such as `caret` (Kuhn, 2008). For `xgboost` there are two parameters that are quite tunable: `eta` and `booster`. The tunability of `booster` is highly influenced by an outlier as can be seen in Figure 3. The 5-fold cross-validated results can be seen in Table 10 of the appendix: they are quite similar to the non cross-validated results and for all parameters slightly higher.

Parameter	Def.P	Def.O	Tun.P	Tun.O	$q_{0.05}$	$q_{0.95}$
glmnet			0.069	0.024		
alpha	1	0.403	0.038	0.006	0.009	0.981
lambda	0	0.004	0.034	0.021	0.001	0.147
rpart			0.038	0.012		
cp	0.01	0	0.025	0.002	0	0.008
maxdepth	30	21	0.004	0.002	12.1	27
minbucket	7	12	0.005	0.006	3.85	41.6
minsplit	20	24	0.004	0.004	5	49.15
kknn			0.031	0.006		
k	7	30	0.031	0.006	9.95	30
svm			0.056	0.042		
kernel	radial	radial	0.030	0.024		
cost	1	682.478	0.016	0.006	0.002	920.582
gamma	$1/p$	0.005	0.030	0.022	0.003	18.195
degree	3	3	0.008	0.014	2	4
ranger			0.010	0.006		
num.trees	500	983	0.001	0.001	206.35	1740.15
replace	TRUE	FALSE	0.002	0.001		
sample.fraction	1	0.703	0.004	0.002	0.323	0.974
mtry	$\sqrt{p}$	$p \cdot 0.257$	0.006	0.003	0.035	0.692
respect.unordered.factors	TRUE	FALSE	0.000	0.000		
min.node.size	1	1	0.001	0.001	0.007	0.513
xgboost			0.043	0.014		
nrounds	500	4168	0.004	0.002	920.7	4550.95
eta	0.3	0.018	0.006	0.005	0.002	0.355
subsample	1	0.839	0.004	0.002	0.545	0.958
booster	gbtree	gbtree	0.015	0.008		
max_depth	6	13	0.001	0.001	5.6	14
min_child_weight	1	2.06	0.008	0.002	1.295	6.984
colsample_bytree	1	0.752	0.006	0.001	0.419	0.864
colsample_bylevel	1	0.585	0.008	0.001	0.335	0.886
lambda	1	0.982	0.003	0.002	0.008	29.755
alpha	1	1.113	0.003	0.002	0.002	6.105

Table 3: Defaults (package defaults (Def.P) and optimal defaults (Def.O)), tunability of the hyperparameters with the package defaults (Tun.P) and our optimal defaults (Tun.O) as reference and tuning space quantiles ($q_{0.05}$ and $q_{0.95}$) for different parameters of the algorithms.

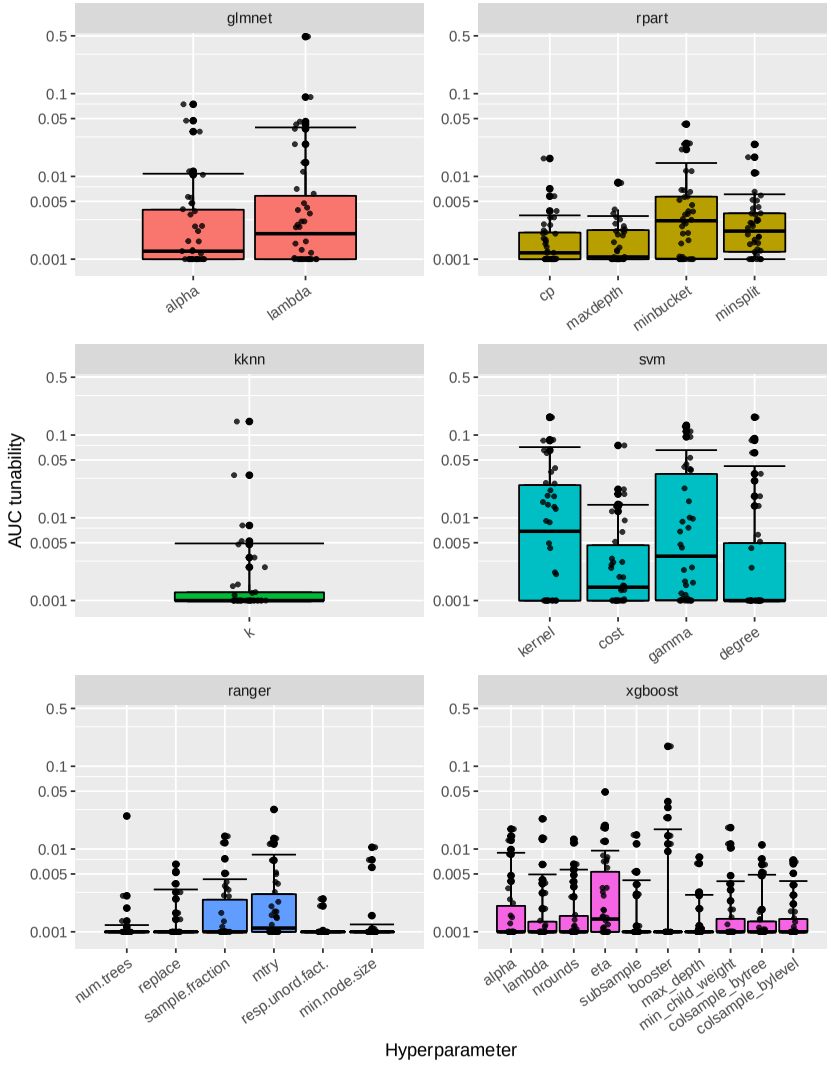


Figure 3: Boxplots of the tunabilities of the hyperparameters of the different algorithms with respect to optimal defaults. The y-axis is on a logarithmic scale. All values below 10^{-3} were set to 10^{-3} to be able to display them. Same definition of whiskers as in Figure 2.

	cp	maxdepth	minbucket	minsplit
cp	0.002	0.003	0.006	0.004
maxdepth		0.002	0.007	0.005
minbucket			0.006	0.011
minsplit				0.004

Table 4: Tunability d_{i_1,i_2} of hyperparameters of `rpart`, diagonal shows tunability of the single hyperparameters.

	maxdepth	minbucket	minsplit
cp	0.0007	0.0005	0.0004
maxdepth		0.0014	0.0019
minbucket			0.0055

Table 5: Joint gain g_{i_1,i_2} of tuning two hyperparameters instead of the most important in `rpart`.

5.4. Tunability of Hyperparameter Combinations and Joint Gains

As an example, Table 4 displays the average tunability d_{i_1,i_2} of all 2-way hyperparameter combinations for `rpart`. Obviously, the increased flexibility in tuning a 2-way combination enables larger improvements when compared with the tunability of one of the respective individual parameters. In Table 5 the joint gain of tuning two hyperparameters g_{i_1,i_2} instead of only the best as defined in Section 3.5 can be seen. The parameters `minsplit` and `minbucket` have the biggest joint effect, which is not very surprising, as they are closely related: `minsplit` is the minimum number of observations that must exist in a node in order for a split to be attempted and `minbucket` the minimum number of observations in any terminal leaf node. If a higher value of `minsplit` than the default performs better on a dataset it is possibly not enough to set it higher without also increasing `minbucket`, so the strong relationship is quite clear. Again, further figures for other algorithms are available through the shiny app. Another remarkable example is the combination of `sample.fraction` and `min.node.size` in `ranger`: the joint gain is very low and tuning `sample.fraction` only seems to be enough, which is concordant to the results of Scornet (2018). Moreover, in `xgboost` the joint gain of `nrounds` and `eta` is relatively low, which is not surprising, as these parameters are highly connected with each other (when setting `nrounds` higher, `eta` should be set lower and vice versa).

5.5. Hyperparameter Space for Tuning

The hyperparameter space for tuning, as defined in Equation (10) in Section 3.6 and based on the 0.05 and 0.95 quantiles, is displayed in Table 3. All optimal defaults are contained in this hyperparameter space while some of the package defaults are not.

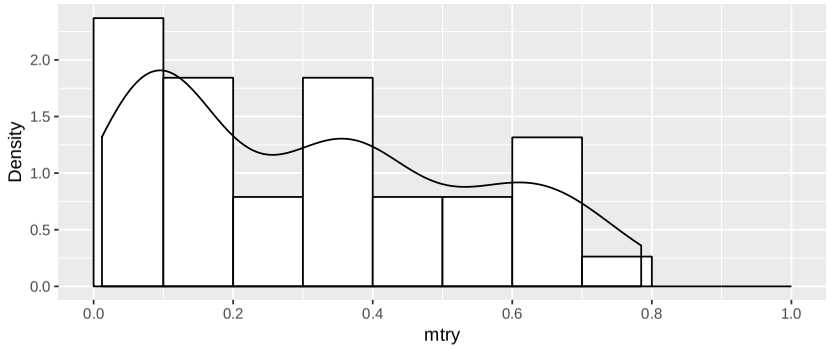


Figure 4: Density and histogram of best parameter values for `mtry` of random forest over all considered datasets.

As an example, Figure 4 displays the full histogram of the best values of `mtry` of the random forest over all datasets. Note that for quite a few datasets much higher values than the package defaults seem advantageous.

6. Conclusion and Discussion

Our paper provides concise and intuitive definitions for optimal defaults of ML algorithms and the impact of tuning them either jointly, tuning individual parameters or combinations, all based on the general concept of surrogate empirical performance models. Tunability values as defined in our framework are easily and directly interpretable as *how much performance can be gained by tuning this hyperparameter?*. This allows direct comparability of the tunability values across different algorithms.

In an extensive OpenML benchmark, we computed optimal defaults for elastic net, decision tree, k-nearest neighbors, SVM, random forest and xgboost and quantified their tunability and the tunability of their individual parameters. This—to the best of our knowledge—has never been provided before in such a principled manner. Our results are often in line with common knowledge from literature and our method itself now allows an analogous analysis for other or more complex methods.

Our framework is based on the concept of default hyperparameter values, which can be seen both as an advantage (default values are a valuable output of the approach) and as an inconvenience (the determination of the default values is an additional analysis step and needed as a reference point for most of our measures).

We now compare our method with van Rijn and Hutter (2017). In contrast to us, they apply the functional ANOVA framework from Hutter et al. (2014) on a surrogate random forest to assess the importance of hyperparameters regarding empirical performance of a support vector machine, random forest and adaboost, which results in numerical importance

scores for individual hyperparameters. Their numerical scores are - in our opinion - less directly interpretable, but they do not rely on defaults as a reference point, which one might see as an advantage. They also propose a method for calculating hyperparameter priors, combine it with the tuning procedure hyperband, and assess the performance of this new tuning procedure. In contrast, we define and calculate ranges for all hyperparameters. Setting ranges for the tuning space can be seen as a special case of a prior distribution - the uniform distribution on the specified hyperparameter space. Regarding the experimental setup, we compute more hyperparameter runs (around 2.5 million vs. 250000), but consider only the 38 binary classification datasets of OpenML100 while van Rijn and Hutter (2017) use all the 100 datasets which also contain multiclass datasets. We evaluate the performance of different surrogate models by 10 times repeated 10-fold cross-validation to choose an appropriate model and to assure that it performs reasonably well.

Our study has some limitations that could be addressed in the future: a) We only considered binary classification, where we tried to include a wider variety of datasets from different domains. In principle this is not a restriction as our methods can easily be applied to multiclass classification, regression, survival analysis or even algorithms not from machine learning whose empirical performance is reliably measurable on a problem instance. b) Uniform random sampling of hyperparameters might not scale enough for very high dimensional spaces, and a smarter sequential technique might be in order here, see Bossek et al. (2015) for an potential approach of sampling across problem instances to learn optimal mappings from problem characteristics to algorithm configurations. c) We currently are learning static defaults, which cannot depend on dataset characteristics (like number of features, or further statistical measures). Doing so might improve performance results of optimal defaults considerably, but would require a more complicated approach. A recent paper regarding this topic was published by van Rijn et al. (2018). d) Our approach still needs initial ranges to be set, in order to run our sampling procedure. Only based on these wider ranges we can then compute more precise, closer ranges.

Acknowledgments

We would like to thank Joaquin Vanschoren for support regarding the OpenML platform and Andreas Müller, Jan van Rijn, Janek Thomas and Florian Pfisterer for reviewing and useful comments. Thanks to Jenny Lee for language editing. This work has been partially funded by grant BO3139/2-3 to ALB from the German Research Foundation and by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibilities for its content.

Appendix A. Additional Graphs and Tables

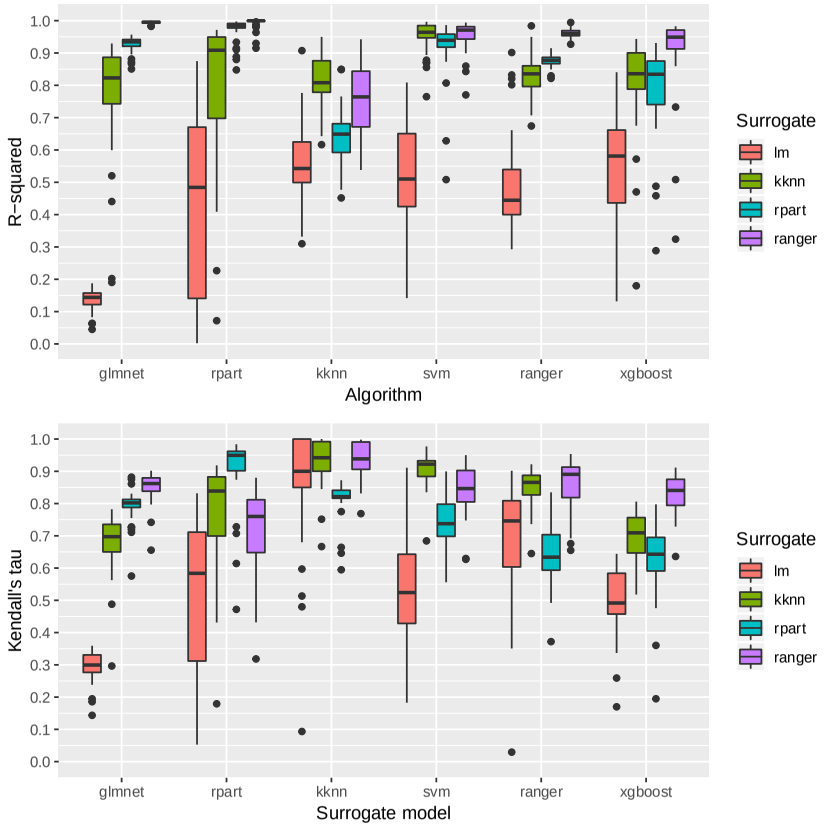


Figure 5: Same as Figure 1 but with exchanged x-axis and legend. Average performances over the datasets of different surrogate models (target: AUC) for different algorithms (that were presented in 4.2).

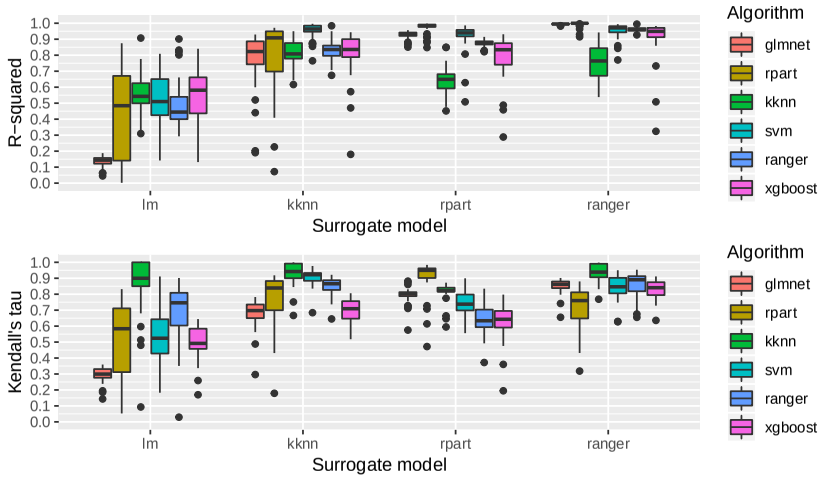


Figure 6: Surrogate model comparison as in Figure 1 but with accuracy as target measure.

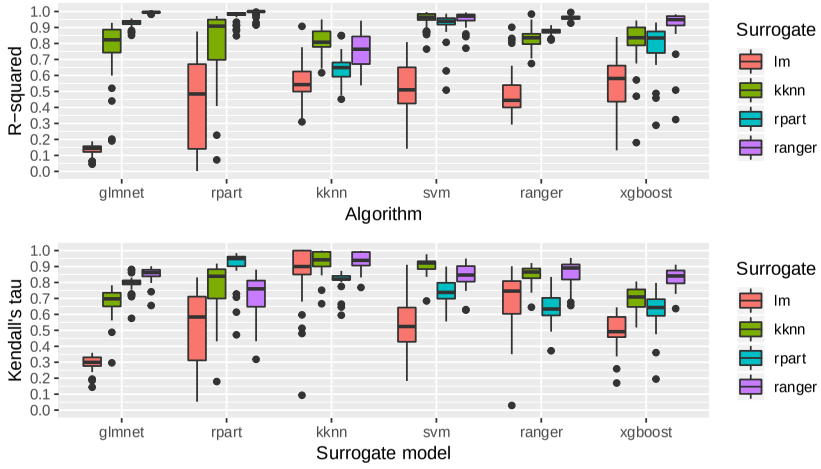


Figure 7: Surrogate model comparison as in Figure 6 (target: accuracy) but with exchanged x-axis and legend.

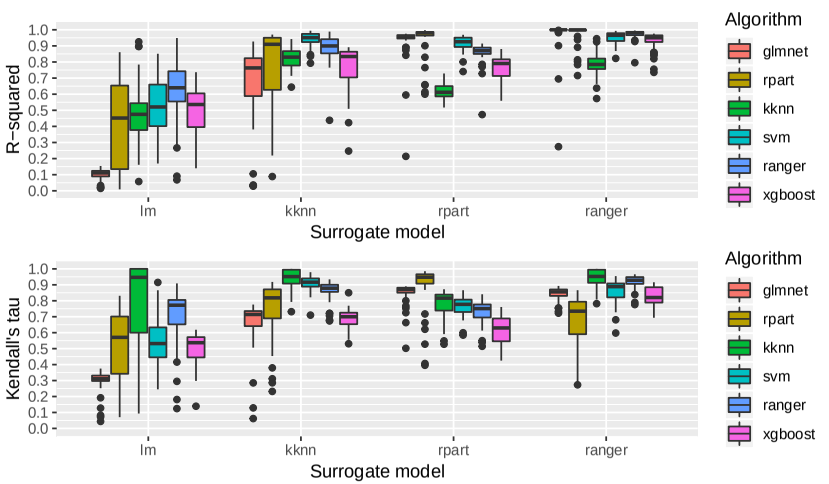


Figure 8: Surrogate model comparison as in Figure 1 but with Brier score as target measure.

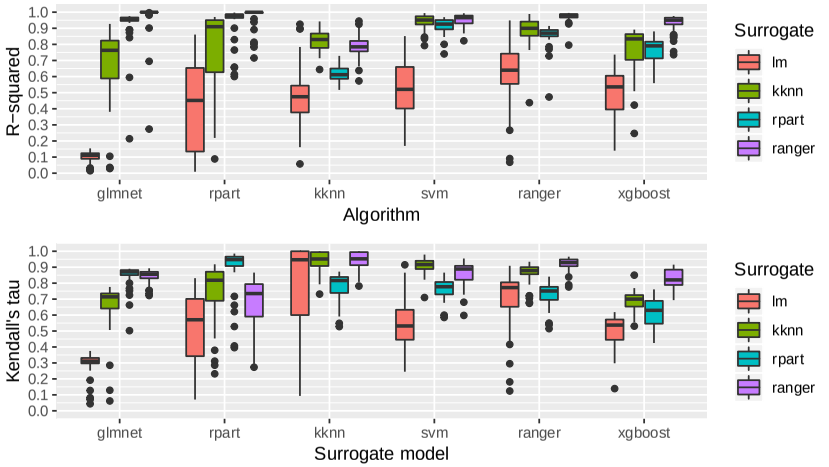


Figure 9: Surrogate model comparison as in Figure 8 but with exchanged x-axis and legend.

Algorithm	Tun.P	Tun.O	Tun.O-CV	Improv	Impr-CV
glmnet	0.042 ± 0.020	0.019 ± 0.010	0.042 ± 0.018	0.023 ± 0.021	0.001 ± 0.013
rpart	0.020 ± 0.004	0.012 ± 0.002	0.014 ± 0.004	0.008 ± 0.003	0.005 ± 0.002
kknn	0.021 ± 0.006	0.008 ± 0.002	0.010 ± 0.004	0.013 ± 0.005	0.010 ± 0.006
svm	0.041 ± 0.009	0.030 ± 0.008	0.041 ± 0.012	0.011 ± 0.004	-0.001 ± 0.011
ranger	0.016 ± 0.004	0.007 ± 0.001	0.009 ± 0.002	0.009 ± 0.004	0.006 ± 0.004
xgboost	0.034 ± 0.005	0.011 ± 0.004	0.012 ± 0.004	0.023 ± 0.004	0.022 ± 0.004

Table 6: Mean tunability as in Table 2, but calculated for the accuracy. Overall tunability (regarding accuracy) with the package defaults (Tun.P) and the optimal defaults (Tun.O) as reference points, cross-validated tunability (Tun.O-CV), average improvement (Improv) and cross-validated average improvement (Impr-CV) obtained by using optimal defaults compared to package defaults. The (cross-validated) improvement can be calculated by the (rounded) difference between Tun.P and Tun.O (Tun.O-CV). Standard error of the mean (SEM) is given behind the "±"-sign.

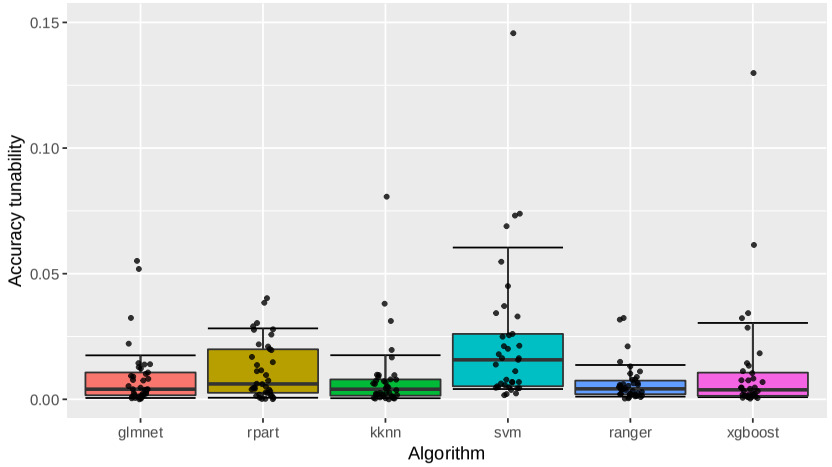


Figure 10: Boxplots of the tunabilities (accuracy) of the different algorithms with respect to optimal defaults.

Parameter	Def.P	Def.O	Tun.P	Tun.O	$q_{0.05}$	$q_{0.95}$
glmnet			0.042	0.019		
alpha	1	0.252	0.022	0.010	0.015	0.979
lambda	0	0.005	0.029	0.017	0.001	0.223
rpart			0.020	0.012		
cp	0.01	0.002	0.013	0.008	0	0.528
maxdepth	30	19	0.004	0.004	10	28
minbucket	7	5	0.005	0.006	1.85	43.15
minsplit	20	13	0.002	0.003	6.7	47.6
kknn			0.021	0.008		
k	7	14	0.021	0.008	2	30
svm			0.041	0.030		
kernel	radial	radial	0.019	0.018		
cost	1	936.982	0.019	0.003	0.025	943.704
gamma	$1/p$	0.002	0.024	0.020	0.007	276.02
degree	3	3	0.005	0.014	2	4
ranger			0.016	0.007		
num.trees	500	162	0.001	0.001	203.5	1908.25
replace	TRUE	FALSE	0.004	0.001		
sample.fraction	1	0.76	0.003	0.003	0.257	0.971
mtry	$\sqrt{p}$	$p \cdot 0.432$	0.010	0.003	0.081	0.867
respect.unordered.factors	TRUE	TRUE	0.001	0.000		
min.node.size	1	1	0.001	0.002	0.009	0.453
xgboost			0.034	0.011		
nrounds	500	3342	0.004	0.002	1360	4847.15
eta	0.3	0.031	0.005	0.005	0.002	0.445
subsample	1	0.89	0.003	0.002	0.555	0.964
booster	gbtree	gbtree	0.008	0.005		
max_depth	6	14	0.001	0.001	3	13
min_child_weight	1	1.264	0.009	0.002	1.061	7.502
colsample_bytree	1	0.712	0.005	0.001	0.334	0.887
colsample_bylevel	1	0.827	0.006	0.001	0.348	0.857
lambda	1	2.224	0.002	0.002	0.004	5.837
alpha	1	0.021	0.003	0.002	0.003	2.904

Table 7: Tunability measures for single hyperparameters and tuning spaces as in Table 3, but calculated for the accuracy. Defaults (package defaults (Def.P) and own optimal defaults (Def.O)), tunability of the hyperparameters with the package defaults (Tun.P) and our optimal defaults (Tun.O) as reference and tuning space quantiles ($q_{0.05}$ and $q_{0.95}$) for different parameters of the algorithms.

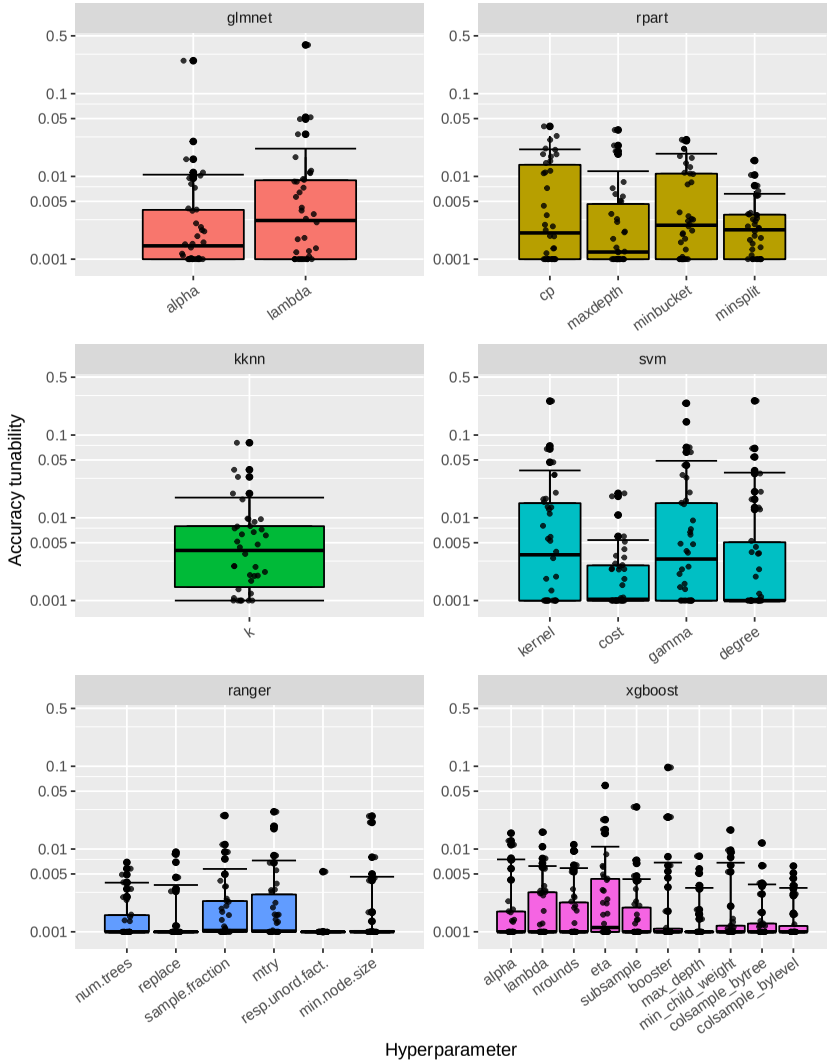


Figure 11: Boxplots of the tunabilities (accuracy) of the hyperparameters of the different algorithms with respect to optimal defaults. The y-axis is on a logarithmic scale. All values below 10^{-3} were set to 10^{-3} to be able to display them. Same definition of whiskers as in Figure 2.

Algorithm	Tun.P	Tun.O	Tun.O-CV	Improv	Impr-CV
glmnet	0.022 ± 0.007	0.010 ± 0.004	0.020 ± 0.014	0.011 ± 0.006	0.001 ± 0.012
rpart	0.015 ± 0.002	0.009 ± 0.002	0.011 ± 0.003	0.006 ± 0.002	0.004 ± 0.002
kknn	0.012 ± 0.003	0.003 ± 0.001	0.003 ± 0.001	0.009 ± 0.003	0.009 ± 0.003
svm	0.026 ± 0.005	0.018 ± 0.004	0.023 ± 0.006	0.008 ± 0.003	0.003 ± 0.005
ranger	0.015 ± 0.004	0.005 ± 0.001	0.006 ± 0.001	0.010 ± 0.004	0.009 ± 0.004
xgboost	0.027 ± 0.003	0.009 ± 0.002	0.011 ± 0.003	0.018 ± 0.002	0.016 ± 0.002

Table 8: Mean tunability as in Table 2, but calculated for the Brier score. Overall tunability (regarding Brier score) with the package defaults (Tun.P) and the optimal defaults (Tun.O) as reference points, cross-validated tunability (Tun.O-CV), average improvement (Improv) and cross-validated average improvement (Impr-CV) obtained by using optimal defaults compared to package defaults. The (cross-validated) improvement can be calculated by the (rounded) difference between Tun.P and Tun.O (Tun.O-CV). Standard error of the mean (SEM) is given behind the "±"-sign.

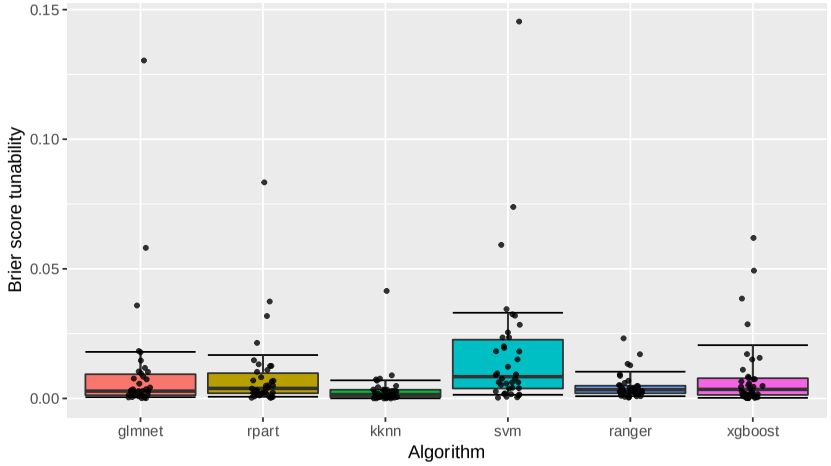


Figure 12: Boxplots of the tunabilities (Brier score) of the different algorithms with respect to optimal defaults.

Parameter	Def.P	Def.O	Tun.P	Tun.O	$q_{0.05}$	$q_{0.95}$
glmnet			0.022	0.010		
alpha	1	0.997	0.009	0.005	0.003	0.974
lambda	0	0.004	0.014	0.007	0.001	0.051
rpart			0.015	0.009		
cp	0.01	0.001	0.009	0.003	0	0.035
maxdepth	30	13	0.002	0.002	9	27.15
minbucket	7	12	0.004	0.006	1	44.1
minsplit	20	18	0.002	0.002	7	49.15
kknn			0.012	0.003		
k	7	19	0.012	0.003	4.85	30
svm			0.026	0.018		
kernel	radial	radial	0.013	0.011		
cost	1	950.787	0.012	0.002	0.002	963.81
gamma	$1/p$	0.005	0.015	0.012	0.001	4.759
degree	3	3	0.003	0.009	2	4
ranger			0.015	0.005		
num.trees	500	198	0.001	0.001	187.85	1568.25
replace	TRUE	FALSE	0.002	0.001		
sample.fraction	1	0.667	0.002	0.003	0.317	0.964
mtry	$\sqrt{p}$	$p \cdot 0.666$	0.010	0.002	0.072	0.954
respect.unordered.factors	TRUE	TRUE	0.000	0.000		
min.node.size	1	1	0.001	0.001	0.008	0.394
xgboost			0.027	0.009		
nrounds	500	2563	0.004	0.002	2018.55	4780.05
eta	0.3	0.052	0.004	0.005	0.003	0.436
subsample	1	0.873	0.002	0.002	0.447	0.951
booster	gbtree	gbtree	0.009	0.004		
max_depth	6	11	0.001	0.001	2.6	13
min_child_weight	1	1.75	0.007	0.002	1.277	5.115
colsample_bytree	1	0.713	0.004	0.002	0.354	0.922
colsample_bylevel	1	0.638	0.004	0.001	0.363	0.916
lambda	1	0.101	0.002	0.003	0.006	28.032
alpha	1	0.894	0.003	0.004	0.003	2.68

Table 9: Tunability measures for single hyperparameters and tuning spaces as in Table 3, but calculated for the Brier score. Defaults (package defaults (Def.P) and own optimal defaults (Def.O)), tunability of the hyperparameters with the package defaults (Tun.P) and our optimal defaults (Tun.O) as reference and tuning space quantiles ($q_{0.05}$ and $q_{0.95}$) for different parameters of the algorithms.

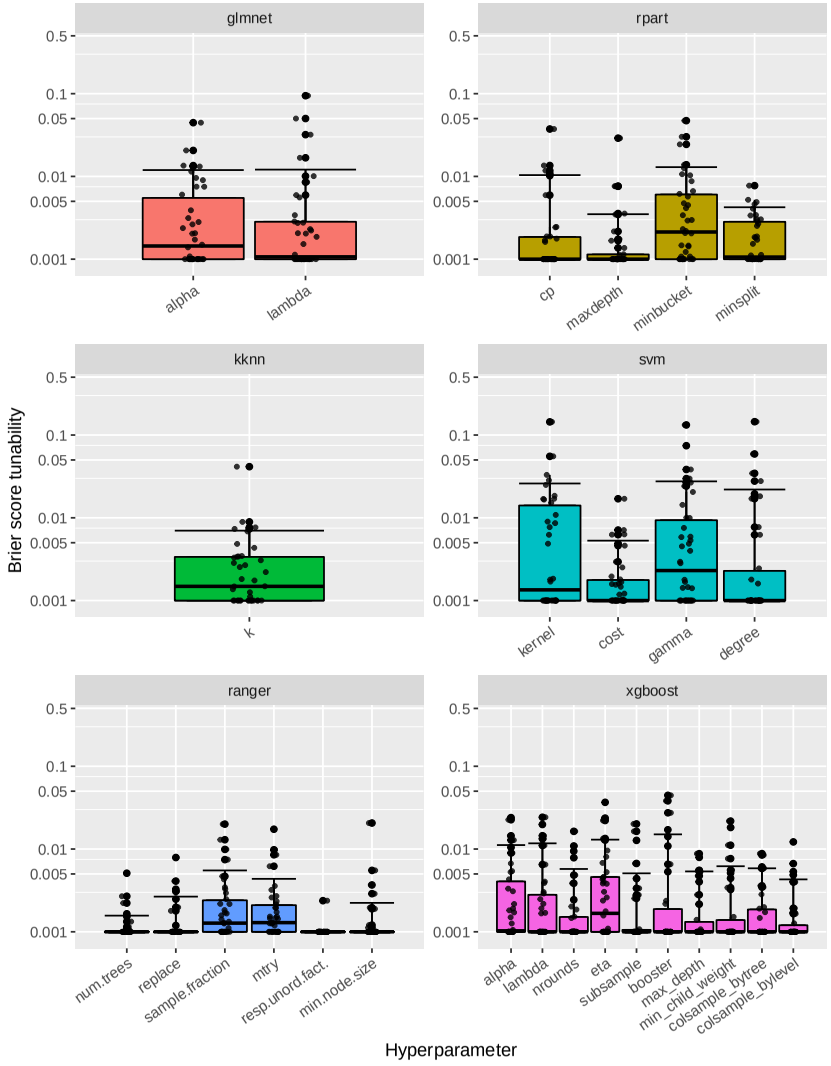


Figure 13: Boxplots of the tunabilities (Brier score) of the hyperparameters of the different algorithms with respect to optimal defaults. The y-axis is on a logarithmic scale. All values below 10^{-3} were set to 10^{-3} to be able to display them. Same definition of whiskers as in Figure 2.

Measure	AUC		Accuracy		Brier score	
Parameter	Tun.O	Tun.O-CV	Tun.O	Tun.O-CV	Tun.O	Tun.O-CV
glmnet	0.024	0.037	0.019	0.042	0.010	0.020
alpha	0.006	0.006	0.010	0.026	0.005	0.015
lambda	0.021	0.034	0.017	0.039	0.007	0.018
rpart	0.012	0.016	0.012	0.014	0.009	0.011
cp	0.002	0.002	0.008	0.008	0.003	0.005
maxdepth	0.002	0.002	0.004	0.004	0.002	0.003
minbucket	0.006	0.009	0.006	0.007	0.006	0.006
minsplit	0.004	0.004	0.003	0.003	0.002	0.003
kknn	0.006	0.006	0.008	0.010	0.003	0.003
k	0.006	0.006	0.008	0.010	0.003	0.003
svm	0.042	0.048	0.030	0.041	0.018	0.023
kernel	0.024	0.030	0.018	0.031	0.011	0.016
cost	0.006	0.006	0.003	0.003	0.002	0.002
gamma	0.022	0.028	0.020	0.031	0.012	0.016
degree	0.014	0.020	0.014	0.027	0.009	0.014
ranger	0.006	0.007	0.007	0.009	0.005	0.006
num.trees	0.001	0.002	0.001	0.003	0.001	0.001
replace	0.001	0.002	0.001	0.002	0.001	0.001
sample.fraction	0.002	0.002	0.003	0.003	0.003	0.003
mtry	0.003	0.004	0.003	0.005	0.002	0.003
respect.unordered.factors	0.000	0.000	0.000	0.001	0.000	0.000
min.node.size	0.001	0.001	0.002	0.002	0.001	0.001
xgboost	0.014	0.017	0.011	0.012	0.009	0.011
nrounds	0.002	0.002	0.002	0.003	0.002	0.002
eta	0.005	0.006	0.005	0.006	0.005	0.006
subsample	0.002	0.002	0.002	0.002	0.002	0.002
booster	0.008	0.008	0.005	0.005	0.004	0.004
max_depth	0.001	0.001	0.001	0.001	0.001	0.001
min_child_weight	0.002	0.003	0.002	0.002	0.002	0.003
colsample_bytree	0.001	0.002	0.001	0.001	0.002	0.002
colsample_bylevel	0.001	0.001	0.001	0.001	0.001	0.002
lambda	0.002	0.003	0.002	0.003	0.003	0.004
alpha	0.002	0.004	0.002	0.003	0.004	0.004

Table 10: Tunability with optimal defaults as reference without (Tun.O) and with (Tun.O-CV) cross-validation for AUC, accuracy and Brier score

References

Charles Audet, John E. Dennis, Douglas Moore, Andrew Booker, and Paul Frank. A surrogate-model-based method for constrained optimization. In *8th Symposium on Multidisciplinary Analysis and Optimization*, page 4891, 2000.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.

André Biedenkapp, Marius Thomas Lindauer, Katharina Eggensperger, Frank Hutter, Chris Fawcett, and Holger H Hoos. Efficient parameter importance analysis via ablation with surrogates. In *AAAI*, pages 773–779, 2017.

Mauro Birattari, Zhi Yuan, Prasanna Balaprakash, and Thomas Stützle. F-Race and iterated F-Race: An overview. In *Experimental Methods for the Analysis of Optimization Algorithms*, pages 311–336. Springer, 2010.

Bernd Bischl, Olaf Mersmann, Heike Trautmann, and Claus Weihs. Resampling methods for meta-model validation with recommendations for evolutionary computation. *Evolutionary Computation*, 20(2):249–275, 2012.

Bernd Bischl, Michel Lang, Lars Kotthoff, Julia Schiffner, Jakob Richter, Erich Studerus, Giuseppe Casalicchio, and Zachary M. Jones. mlr: Machine learning in R. *Journal of Machine Learning Research*, 17(170):1–5, 2016.

Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Frank Hutter, Michel Lang, Rafael G. Mantovani, Jan N. van Rijn, and Joaquin Vanschoren. OpenML benchmarking suites and the OpenML100. *ArXiv preprint arXiv:1708.03731*, 2017a. URL `https://arxiv.org/abs/1708.03731`.

Bernd Bischl, Jakob Richter, Jakob Bossek, Daniel Horn, Janek Thomas, and Michel Lang. mlrMBO: A modular framework for model-based optimization of expensive black-box functions. *ArXiv preprint arXiv:1703.03373*, 2017b. URL `https://arxiv.org/abs/1703.03373`.

Jakob Bossek, Bernd Bischl, Tobias Wagner, and Günter Rudolph. Learning feature-parameter mappings for parameter tuning via the profile expected improvement. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 1319–1326. ACM, 2015.

Giuseppe Casalicchio, Jakob Bossek, Michel Lang, Dominik Kirchhoff, Pascal Kerschke, Benjamin Hofner, Heidi Seibold, Joaquin Vanschoren, and Bernd Bischl. OpenML: An R package to connect to the machine learning platform OpenML. *Computational Statistics*, 32(3):1–15, 2017.

Katharina Eggensperger, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Surrogate benchmarks for hyperparameter optimization. In *Proceedings of the 2014 International Conference on Meta-learning and Algorithm Selection-Volume 1201*, pages 24–31. CEUR-WS.org, 2014.

Katharina Eggensperger, Marius Lindauer, Holger H Hoos, Frank Hutter, and Kevin Leyton-Brown. Efficient benchmarking of algorithm configurators via model-based surrogates. *Machine Learning*, pages 1–27, 2018.

Agoston E Eiben and Selmar K Smit. Parameter tuning for configuring and analyzing evolutionary algorithms. *Swarm and Evolutionary Computation*, 1(1):19–31, 2011.

Chris Fawcett and Holger H Hoos. Analysing differences between algorithm configurations through ablation. *Journal of Heuristics*, 22(4):431–458, 2016.

Matthias Feurer, Benjamin Letham, and Eytan Bakshy. Scalable meta-learning for bayesian optimization. *arXiv preprint 1802.02219*, 2018. URL `https://arxiv.org/abs/1802.02219`.

Isabelle Guyon, Amir Saffari, Gideon Dror, and Gavin Cawley. Model selection: Beyond the bayesian/frequentist divide. *Journal of Machine Learning Research*, 11(Jan):61–87, 2010.

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International Conference on Learning and Intelligent Optimization*, pages 507–523. Springer, 2011.

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Identifying key algorithm parameters and instance features using forward selection. In *International Conference on Learning and Intelligent Optimization*, pages 364–381. Springer, 2013.

Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. An efficient approach for assessing hyperparameter importance. In *ICML*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 754–762, 2014.

Max Kuhn. Building predictive models in R using the caret package. *Journal of statistical software*, 28(5):1–26, 2008.

Max Kuhn, Steve Weston, Chris Keefer, and Nathan Coulter. *Cubist: Rule- and instance-based regression modeling*, 2016. R package version 0.0.19.

Daniel Kühn, Philipp Probst, Janek Thomas, and Bernd Bischl. OpenML R bot benchmark data (final subset), 2018a. URL `https://figshare.com/articles/OpenML_R_Bot_Benchmark_Data_final_subset_/5882230/2`.

Daniel Kühn, Philipp Probst, Janek Thomas, and Bernd Bischl. Automatic Exploration of Machine Learning Experiments on OpenML. *ArXiv preprint arXiv:1806.10961*, 2018b. URL `https://arxiv.org/abs/1806.10961`.

Upmanu Lall and Ashish Sharma. A nearest neighbor bootstrap for resampling hydrologic time series. *Water Resources Research*, 32(3):679–693, 1996.

Michel Lang, Bernd Bischl, and Dirk Surmann. batchtools: Tools for R to work on batch systems. *The Journal of Open Source Software*, 2(10), 2017.

Gang Luo. A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Network Modeling Analysis in Health Informatics and Bioinformatics*, 5(1):1–16, 2016.

Rafael G. Mantovani, Tomáš Horváth, Ricardo Cerri, Andre C.P.L.F. Carvalho, and Joaquin Vanschoren. Hyper-parameter tuning of a decision tree induction algorithm. In *Brazilian Conference on Intelligent Systems (BRACIS 2016)*, 2016.

Rafael Gomes Mantovani, Tomáš Horváth, Ricardo Cerri, Sylvio Barbon Junior, Joaquin Vanschoren, André Carlos Ponce de de Carvalho, and Leon Ferreira. An empirical study on hyperparameter tuning of decision trees. *ArXiv preprint arXiv:1812.02207*, 2018. URL `https://arxiv.org/abs/1812.02207`.

Erwan Scornet. Tuning parameters in random forests. *ESAIM: Procs*, 60:144–162, 2018.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

Jan N. van Rijn and Frank Hutter. Hyperparameter importance across datasets. *ArXiv preprint arXiv:1710.04725*, 2017. URL `https://arxiv.org/abs/1710.04725`.

Jan N van Rijn, Florian Pfisterer, Janek Thomas, Andreas Muller, Bernd Bischl, and J Vanschoren. Meta learning for defaults: Symbolic defaults. In *Neural Information Processing Workshop on Meta-Learning*, 2018.

Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. OpenML: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013.

Levi Waldron, Melania Pintilie, Ming-Sound Tsao, Frances A Shepherd, Curtis Huttenhower, and Igor Jurisica. Optimized application of penalized regression methods to diverse genomic data. *Bioinformatics*, 27(24):3399–3406, 2011.

Journal of Machine Learning Research 20 (2019) 1-31 Submitted 7/18; Revised 1/19; Published 2/19

Deep Reinforcement Learning for Swarm Systems

Maximilian Hüttenrauch MHUETTENRAUCH@LINCOLN.AC.UK
L-CAS
University of Lincoln
LN6 7TS Lincoln, UK

Adrian Šošić ADRIAN.SOSIC@BCS.TU-DARMSTADT.DE
Bioinspired Communication Systems
Technische Universität Darmstadt
64283 Darmstadt, Germany

Gerhard Neumann GNEUMANN@LINCOLN.AC.UK
L-CAS
University of Lincoln
LN6 7TS Lincoln, UK

Editor: George Konidaris

Abstract

Recently, deep reinforcement learning (RL) methods have been applied successfully to multi-agent scenarios. Typically, the observation vector for decentralized decision making is represented by a concatenation of the (local) information an agent gathers about other agents. However, concatenation scales poorly to swarm systems with a large number of homogeneous agents as it does not exploit the fundamental properties inherent to these systems: (i) the agents in the swarm are interchangeable and (ii) the exact number of agents in the swarm is irrelevant. Therefore, we propose a new state representation for deep multi-agent RL based on mean embeddings of distributions, where we treat the agents as samples and use the empirical mean embedding as input for a decentralized policy. We define different feature spaces of the mean embedding using histograms, radial basis functions and neural networks trained end-to-end. We evaluate the representation on two well-known problems from the swarm literature — rendezvous and pursuit evasion — in a globally and locally observable setup. For the local setup we furthermore introduce simple communication protocols. Of all approaches, the mean embedding representation using neural network features enables the richest information exchange between neighboring agents, facilitating the development of complex collective strategies.

Keywords: deep reinforcement learning, swarm systems, mean embeddings, neural networks, multi-agent systems

1. Introduction

In swarm systems, many identical agents interact with each other to achieve a common goal. Typically, each agent in a swarm has limited capabilities in terms of sensing and manipulation so that the considered tasks need to be solved collectively by multiple agents.

A promising application where intelligent swarm systems take a prominent role is swarm robotics (Bayındır, 2016). Robot swarms are formed by a large number of cheap and easy to manufacture robots that can be useful in a variety of situations and tasks, such as search and rescue missions or exploration scenarios. A swarm of robots is inherently redundant towards loss of individual robots since usually none of the robots plays a specific role in the execution of the task. Because of this property, swarm-based missions are often favorable over single-robot missions (or, let alone, human missions) in hazardous environments. Behavior of natural swarms, such as foraging, formation control, collective manipulation, or the localization of a common 'food' source can be adapted to aid in these missions (Bayındır, 2016). Another field of application is routing in wireless sensor networks (Saleem et al., 2011) since each sensor in the network can be treated as an agent in a swarm.

A common method to obtain control strategies for swarm systems is to apply optimization-based approaches using a model of the agents or a graph abstraction of the swarm (Lin et al., 2004; Jadbabaie et al., 2003). Optimization-based approaches allow to compute optimal control policies for tasks that can be well modeled, such as rendezvous or consensus problems (Lin et al., 2007) and formation control (Ranjbar-Sahraei et al., 2012), or to learn pursuit strategies to capture an evader (Zhou et al., 2016). Yet, these approaches typically use simplified models of the agents / the task and often rely on unrealistic assumptions, such as operating in a connected graph (Dimarogonas and Kyriakopoulos, 2007) or having full observability of the system state (Zhou et al., 2016). Rule-based approaches use heuristics inspired by natural swarm systems, such as ants or bees (Handl and Meyer, 2007). Yet, while the resulting heuristics are often simple and can lead to complex swarm behavior, the obtained rules are difficult to adapt, even if the underlying task changes only slightly.

Recently, deep reinforcement learning (RL) strategies have become popular to solve multi-agent coordination problems. In RL, tasks are specified indirectly through a cost function, which is typically easier than defining a model of the task directly or a finding a heuristic for the controller. Having defined a cost function, the RL algorithm aims to find a policy that minimizes the expected cost. Applying deep reinforcement learning within the swarm setting, however, is challenging due to the large number of agents that need to be considered. Compared to single-agent learning, where the agent is confronted only with observations about its own state, each agent in a swarm can make observations of several other agents populating the environment and thus needs to process an entire set of information that is potentially varying in size. Accordingly, two main challenges can be identified in the swarm setting:

1. High state and observation dimensionality, caused by large system sizes.

2. Changing size of the available information set, either due to addition or removal of agents, or because the number of observed neighbors changes over time.

Most current multi-agent deep reinforcement learning methods either concatenate the information received from different agents (Lowe et al., 2017) or encode it in a multi-channel image, where the image channels contain different features based on a local view of an agent (Sunehag et al., 2017; Zheng et al., 2017). However, both types of methods bare major drawbacks. Since neural network policies assume a fixed input dimensionality, a concatenation of observations is unsuitable in the case changing agent numbers. Furthermore, a concatenation disregards the inherent permutation invariance of identical agents in a swarm system and scales poorly to large system sizes. Top-down image based representations alleviate the

©2019 Maximilian Hüttenrauch and Adrian Šošić and Gerhard Neumann. JMLR 20(54):1-31, 2019

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-476.html.

issue of permutation invariance, however, the information obtained from neighboring agents is of mostly spatial nature. While additional information can be captured by adding more image channels, the dimensionality of the representation increases linearly with each feature. Furthermore, the discretization into pixels has limited accuracy due to quantization errors.

In this paper, we exploit the homogeneity of swarm systems and treat the state information perceived from neighboring agents as samples of a random variable. Based on this model, we then use mean feature embeddings (MFE) (Smola et al., 2007) to encode the current distribution of the agents. Each agent gets a local view of this distribution, where the information obtained from the neighbors is encoded in the mean embedding. Due to the sample-based view of the collected state information, we achieve a permutation invariant representation that is furthermore invariant to the number of agents in the swarm / the number of perceived neighbors.

Mean feature embeddings have so far been used mainly for kernel-based feature representations (Gretton et al., 2012), but they can be also applied to histograms or radial basis function (RBF) networks. The resulting models are closely related to the "invariant model" formulated by Zaheer et al. (2017). However, compared to the summation approach described in their paper, the averaging of feature activations proposed in our approach yields the desired invariance with respect to the observed agent number mentioned above. To the best of our knowledge, we are the first to use mean embeddings inside a deep reinforcement learning framework for swarm systems where both the feature space of the mean embedding as well as the policy are learned end-to-end.

We test our state representation on various rendezvous and pursuit evasion problems using Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) as the underlying deep RL algorithm. In the rendezvous problem, the agents need to find a collective strategy that allows them to meet at some arbitrary location. In the pursuit evasion domain, a group of agents collectively tries to capture one or multiple evaders.

Policies are learned in a *centralized-learning / decentralized-execution fashion* fashion, meaning that during learning data from all agents is collected centrally and used to optimize the parameters as if there was only one agent. Nonetheless, each agent only has access to its own perception of the global system state to generate actions from the policy function. We compare our representation to several deep RL baselines as well as to optimization-based solutions, if available. Herein, we perform our experiments both in settings with global observability (i.e., all agents are neighbors) and in settings with local observability (i.e., agents are only locally connected). In the latter setting, we also evaluate different communication protocols (Hüttenrauch et al., 2018) that allow the agents to transmit additional information about their local graph structure. For example, an agent might transmit the number of neighbors within its current neighborhood. Previously, such additional information could not be encoded efficiently due to the poor scalability of the histogram-based approaches.

Our results show that agents using our representation can learn faster and obtain policies of higher quality, suggesting that the representation as mean embedding is an efficient encoding of the global state configuration for swarm-based systems. Moreover, mean embeddings are simple to implement inside existing neural network architectures and can be

applied to any deep RL algorithm, which makes the approach applicable in a wide variety of scenarios. The source code to reproduce our results can be found online.[1]

2. Related Work

The main contribution of this work lies in the development of a compact representation of state information in swarm systems, which can easily be used within deep multi-agent reinforcement learning (MARL) settings that contain homogeneous agent groups. In fact, our work is mostly orthogonal to other research in the field of MARL and the presented ideas can be incorporated into most existing approaches. To provide an overview, we begin with a brief survey of algorithms used in (deep) MARL, we revisit the basics of mean embedding theory, and we summarize some classic approaches to swarm control for the rendezvous and pursuit evasion task.

2.1. Deep RL

Recently, there has been increasing interest in deep reinforcement learning for swarms and multi-agent systems in general. For example, Zheng et al. (2017) provide a many-agent reinforcement learning platform based on a multi-channel image state representation, which uses Deep Q-Networks (DQN) (Mnih et al., 2015) to learn decentralized control strategies in large grid worlds with discrete actions. Gupta et al. (2017) show a comparison of centralized, concurrent and parameter sharing approaches to cooperative deep MARL, using TRPO (Schulman et al., 2015), DDPG (Lillicrap et al., 2015) and DQN. They evaluate each method on three tasks, one of which is a pursuit task in a grid world using bitmap-like images as state representation. A variant of DDPG for multiple agents in Markov games using a centralized action-value function is provided by Lowe et al. (2017). The authors evaluate the method on tasks like cooperative communication, navigation and others. The downside of a centralized action-value function is that the input space grows linearly with the number of agents, and hence, their approach scales poorly to large system sizes. A more scalable approach is presented by Yang et al. (2018). Employing mean field theory, the interactions within the population of agents are approximated by the interaction of a single agent with the average effect from the overall population, which has the effect that the action-value function input space stays constant. Experiments are conducted on a Gaussian squeeze problem, an Ising model, and a mixed cooperative-competitive battle game. Yet, the paper does not address the state representation problem for swarm systems.

Omidshafiei et al. (2017) investigate hysteretic Q-learning (Matignon et al., 2007) and distillation (Rusu et al., 2015). They use deep recurrent Q-networks (Hausknecht and Stone, 2015) to solve single and multi-task Dec-POMDP problems. Following this work, Palmer et al. (2017) add leniency (Panait et al., 2006) to the hysteretic approach to prevent "relative overgeneralization" of agents. The approach is evaluated on a coordinated multi-agent object transportation problem in a grid world with stochastic rewards.

Sunehag et al. (2017) tackle the "lazy agent" problem in cooperative MARL with a single team reward by training each agent with a learned additive decomposition of a value function based on the team reward. Experiments show an increase in performance on cooperative

1. `https://github.com/LCAS/deep_rl_for_swarms`

two-player games in a grid world. Rashid et al. (2018) further develop the idea with the insight that a full factorization of the value function is not necessary. Instead, they introduce a monotonicity constraint on the relationship between the global value function and each local value function. Results are presented on the StarCraft micro management domain.

Finally, Grover et al. (2018) show a framework to model agent behavior as a representation learning problem. They learn an encoder-decoder embedding of agent policies via imitation learning based on interactions and evaluate it on a cooperative particle world (Mordatch and Abbeel, 2018) and a competitive two-agent robo sumo environment (Al-Shedivat et al., 2018). The design of the policy function in the approach of Mordatch and Abbeel (2018) is similar to ours but the model uses a softmax pooling layer. However, instead of applying (model-free) reinforcement learning to optimize the parameters of the policy function, they build an end-to-end differentiable model of all agent and environment state dynamics and calculate the gradient of the return with respect to the parameters via backpropagation.

An application related to our approach can be found in the work by Gebhardt et al. (2018), where the authors use mean embeddings to learn a centralized controller for object manipulation with robot swarms. Here, the key idea is to directly embed the swarm configuration into a reproducing kernel Hilbert space, whereas our approach is based on embedding the agent's local view. Furthermore, using kernel-based feature spaces for the mean embedding scales poorly in the number of samples and in the dimensionality of the embedded information.

2.2. Optimization-Based Approaches for Swarm Systems

To provide a concise summary of the most relevant related work, we concentrate on optimization-based approaches that derive decentralized control strategies for the rendezvous and pursuit evasion problem considered in this paper. Ji and Egerstedt (2007) derive a control mechanism preserving the connectedness of a group of agents with limited communication abilities for the rendezvous and formation control problem. The method focuses on high-level control with single integrator linear state manipulation and provides no rules for agents that are not part of the agent graph. Similarly, Gennaro and Jadbabaie (2006) present a decentralized algorithm to maximize the connectivity (characterized by an exponential model) of a multi-agent system. The algorithm is based on the minimization of the second smallest eigenvalue of the Laplacian of the proximity graph. An approach providing a decentralized control strategy for the rendezvous problem for nonholonomic agents can be found in the work by Dimarogonas and Kyriakopoulos (2007). Using tools from nonsmooth Lyapunov theory and graph theory, the stability of the overall system is examined. A control strategy for the pursuit evasion problem with multiple pursuers and single evader that we investigate in more detail later in this paper was proposed Zhou et al. (2016). The authors derive decentralized control policies for the pursuers and the evader based on the minimization of Voronoi partitions. Again, the control mechanism is for high-level linear state manipulation. Furthermore, the method assumes visibility of the evader at all times. A survey on pursuit evasion in mobile robotics in general is provided by Chung et al. (2011).

2.3. Analytic Approaches

Another line of work concerned with the curse of dimensionality can be found in the area of multi-player reach-avoid games. Chen et al. (2017), for example, look at pairwise interactions between agents. This way, they are able to use the Hamilton-Jacobian-Isaacs approach to solve a partial differential equation in the joint state space of the players. Similar work can be found in (Chen et al., 2014a,b; Zhou et al., 2012).

3. Background

In this section, we give a short overview of Trust Region Policy Optimization and mean embeddings of distributions.

3.1. Trust Region Policy Optimization

Trust Region Policy Optimization is an algorithm to optimize control policies in single-agent reinforcement learning problems (Schulman et al., 2015). These problems are formulated as Markov decision processes (MDPs), which can be compactly written as a tuple $\langle \mathcal{S}, \mathcal{A}, P, R \rangle$. In an MDP, an agent chooses an action $a \in \mathcal{A}$ according to some policy $\pi(a \,|\, s)$ based on its current state $s \in \mathcal{S}$ and progresses to state $s' \in \mathcal{S}$ according to the transition dynamics P, i.e., $s' \sim P(s' \,|\, s, a)$. After each step, the agent receives a reward $r = R(s, a)$, provided by the reward function R, which judges the quality of its decision. The goal of the agent is to find a policy that maximizes the cumulative reward achieved over a certain period of time.

In TRPO, the policy is parametrized by a parameter vector θ containing the weights and biases of a neural network. In the following, we denote this parametrized policy as π_θ. The reinforcement learning objective is expressed as finding a new policy that maximizes the expected advantage function of the current policy π_{old}, i.e., $J^{\text{TRPO}} = \mathbb{E}\left[\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}} A^{\pi_{\text{old}}}(s, a)\right]$, where $A^{\pi_{\text{old}}}(s, a) = Q^{\pi_{\text{old}}}(s, a) - V^{\pi_{\text{old}}}(s)$. Herein, the state-action value function $Q^{\pi_{\text{old}}}(s, a)$ is typically estimated via trajectory rollouts, while for the value function $V^{\pi_{\text{old}}}(s)$ linear or neural network baselines are used that are fitted to the Monte-Carlo returns, resulting in an estimate $\hat{A}(s, a)$ for the advantage function. The objective is to be maximized subject to a fixed constraint on the Kullback-Leibler (KL) divergence of the policy before and after the parameter update, which ensures that the updates to the policy parameters θ are bounded, in order to avoid divergence of the learning process. The overall optimization problem is summarized as

$$\begin{aligned} \underset{\theta}{\text{maximize}} \quad & \mathbb{E}\left[\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}} \hat{A}(s, a)\right] \\ \text{subject to} \quad & \mathbb{E}[D_{\text{KL}}(\pi_{\theta_{\text{old}}} || \pi_\theta)] \leq \delta. \end{aligned}$$

The problem is approximately solved using conjugate gradient optimization, after linearizing the objective and quadratizing the constraint.

3.2. Mean Embeddings

Our work is inspired by the idea of embedding distributions into reproducing kernel Hilbert spaces (Smola et al., 2007) from where we borrow the concept of mean embeddings. A probability distribution $P(X)$ can be represented as an element in a reproducing kernel

Hilbert space by its expected feature map (i.e., the mean embedding),

$$\mu_X = \mathbb{E}_X[\phi(X)],$$

where $\phi(x)$ is a (possibly infinite dimensional) feature mapping. Given a set of observations $\{x_1, \ldots, x_m\}$, drawn i.i.d. from $P(X)$, the empirical estimate of the expected feature map is given by

$$\hat{\mu}_X = \frac{1}{m} \sum_{i=1}^{m} \phi(x_i).$$

Using characteristic kernel functions $k(x, x') = \langle \phi(x),\ \phi(x') \rangle$, such as Gaussian RBF or Laplace kernels, mean embeddings can be used, for example, in two-sample tests (Gretton et al., 2012) and independence tests (Gretton et al., 2008). A characteristic kernel is required to uniquely identify a distribution based on its mean embedding. However, this assumption can be relaxed to using finite feature spaces if the objective is merely to extract relevant information from a distribution such as, in our case, the information needed for the policy of the agents.

4. Deep Reinforcement Learning for Swarms

The reinforcement learning algorithm presented in the last section has been originally designed for single-agent learning. In order to apply this algorithm to the swarm setup, we switch to a different problem domain and show the implications on the learning algorithm. Policies in this context are then optimized in a centralized–learning / decentralized–execution fashion.

4.1. Problem Domain

The problem domain for our swarm system is best described as a swarm MDP environment (Šošić et al., 2017). The swarm MDP can be regarded as a special case of a decentralized partially observable Markov decision process (Dec-POMDP) (Bernstein et al., 2002) and is constructed in two steps. First, an agent prototype is defined as a tuple $\mathbb{A} = \langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \pi \rangle$, determining the local properties of an agent in the system. Herein, $\mathcal{S}$ denotes the set of the agent's local states, $\mathcal{O}$ is the set of possible local observations, $\mathcal{A}$ is the set of actions available to the agent, and $\pi : \mathcal{O} \times \mathcal{A} \to [0, 1]$ is the agent's stochastic control policy. Based on this definition, the swarm MDP is constructed as $\langle N, \mathbb{A}, P, O, R \rangle$, where N is the number of agents in the system and $\mathbb{A}$ is the aforementioned agent prototype. The coupling of the agents is specified through a global state transition model $P : \mathcal{S}^N \times \mathcal{S}^N \times \mathcal{A}^N \to [0, \infty)$ and an observation model $O : \mathcal{S}^N \times \{1, \ldots, N\} \to \mathcal{O}$, which determines the local observation $\boldsymbol{o}^i \in \mathcal{O}$ for agent i at a given swarm state $\boldsymbol{s} \in \mathcal{S}^N$, i.e., $\boldsymbol{o}^i = O(\boldsymbol{s}, i)$. Finally, $R : \mathcal{S}^N \times \mathcal{A}^N \to \mathbb{R}$ is the global reward function, which encodes the cooperative task for the swarm by providing an instantaneous reward feedback $R(\boldsymbol{s}, \boldsymbol{a})$ according to the current swarm state $\boldsymbol{s}$ and the corresponding joint action assignment $\boldsymbol{a} \in \mathcal{A}^N$ of the agents. The specific state dynamics and observation models considered in this paper are described in Section 5.

The model encodes two important properties of swarm networks: First, all agents in the system are assumed to be identical, and accordingly, they are all assigned the same

decentralized policy π. This is an immediate consequence of the two-step construction of the model, which implies that all agents share the same internal architecture. Second, the agents are only partially informed about the global system state, as prescribed by the observation model O. Note that both the transition model and the observation model are assumed to be invariant to permutations of the agents in order to ensure the homogeneity of the system. For details, see (Šošić et al., 2017).

4.2. Local Observation Models

The local observation $\boldsymbol{o}^i$ introduced in the last section is a combination of observations o^i_{loc} an agent makes about local properties (like the agent's current velocity or its distance to a wall) and observations O^i of other agents. In order to describe the observation model used for the agents, we use an interaction graph representation of the swarm. This graph is given by nodes $V = \{v_1, v_2, \ldots, v_N\}$ corresponding to the agents in the swarm and an edge set $E \subset V \times V$, which we assume contains unordered pairs of the form $\{v_i, v_j\}$ indicating that agents i and j are neighbors. The interaction graph is denoted as $\mathcal{G} = (V, E)$. If both the set of nodes and the set of edges are not changing, we call $\mathcal{G}$ a static interaction graph; if either of the set undergoes changes, we instead refer to $\mathcal{G}$ as a dynamic interaction graph.

The set of neighbors of agent i in the graph $\mathcal{G}$ is given by

$$\mathcal{N}_{\mathcal{G}}(i) = \{j \,|\, \{v_i, v_j\} \in E\}.$$

Within this neighborhood, agent i can sense local information about other agents, for example distance or bearing to each neighbor. We denote the information agent i receives from agent j as $o^{i,j} = f(s^i, s^j)$, which is a function of the local states of agent i and agent j. The observation $o^{i,j}$ is available for agent i only if $j \in \mathcal{N}_{\mathcal{G}}(i)$. Hence, the complete state information agent i receives from all neighbors is given by the set $O^i = \left\{o^{i,j} \,|\, j \in \mathcal{N}_{\mathcal{G}}(i)\right\}$.

As the observations of other agents are summarized in form of sets $\{O^i\}$, we require an efficient encoding that can be used as input to a neural network policy. In particular, it must meet the following two properties:

- The encoding needs to be invariant to the indexing of the agents, respecting the unorderedness of the elements in the observation set. Only by exploiting the system's inherent homogeneity we can escape the curse of dimensionality.
- The encoding must be applicable to varying set sizes because the local graph structure might change dynamically. Even if each agent can observe the entire system at all times, the encoding should be applicable for different swarm sizes.

4.3. Local Communication Models

In addition to perceiving local state information of neighboring agents, the agents can also communicate information about the interaction graph $\mathcal{G}$ (Hüttenrauch et al., 2018). For example, agent j can transmit the number of perceived neighbors to agent i. Furthermore, the agents can also perform more complex operations on their local neighborhood graph. For example, they could compute the shortest distance to a target point (such as an evader) that is perceived by at least one agent within their local sub-graph. Hence, by using local

communication protocols, observation $o^{i,j}$ can contain information about both, the local states s^i and s^j as well as the graph $\mathcal{G}$, i.e., $o^{i,j} = f(s^i, s^j, \mathcal{G})$.

4.4. Mean Embeddings as State Representations for Swarms

In the simplest case, the local observation $o^{i,j}$ that agent i receives of agent j is composed of the distance and the bearing angle of agent i to agent j. However, $o^{i,j}$ can also contain more complex information, such as relative velocities or orientations. A straightforward way to represent the information set O^i is to concatenate the local quantities $\{o^{i,j}\}_j$ into a single observation vector. However, as mentioned before, this representation has various drawbacks as it ignores the permutation invariance inherent to a homogeneous agent network. Furthermore, it grows linearly with the number of agents in the swarm and is, therefore, limited to a fixed number of neighbors when used in combination with neural network policies.

To resolve these issues, we treat the elements in the information set O^i as samples from a distribution that characterizes the current swarm configuration, i.e., $o^{i,j} \sim p_i(\cdot \,|\, \mathbf{s})$. We can now use an empirical encoding of this distribution in order to achieve permutation invariance of the elements of O^i as well as flexibility to the size of O^i. As highlighted in Section 3.2, a simple way is to use a mean feature embedding, i.e.,

$$\hat{\mu}_{O^i} = \frac{1}{|O^i|} \sum_{o^{i,j} \in O^i} \phi(o^{i,j}),$$

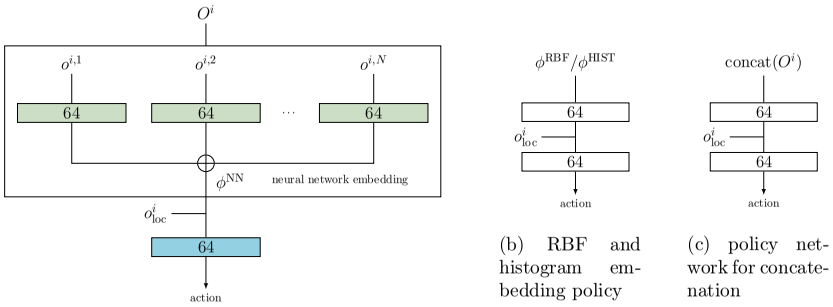
where ϕ defines the feature space of the mean embedding. The input dimensionality to the policy is given by the dimensionality of the feature space of the mean embedding and, hence, it does not depend on the size of the information set O^i any more. This allows us to use the embedding $\hat{\mu}_{O^i}$ as input to a neural network used in deep RL. In the following sections, we describe different feature spaces that can be used for the mean embedding. Figure 1 illustrates the resulting policy architectures with further details given in Appendix F.

4.4.1. Neural Network Feature Embeddings

In line with the deep RL paradigm, we propose to use a neural network as feature mapping ϕ^{NN} whose parameters are determined by the reinforcement learning algorithm. Using a neural network to define the feature space allows us to handle high dimensional observations, which is not feasible with traditional approaches such as histograms (Hüttenrauch et al., 2018). In our experiments, a rather shallow architecture with one layer of RELU units already performed very well, but deeper architectures could be used for more complex applications. To the best of our knowledge, we present the first approach for using neural networks to define the feature space of a mean embedding.

4.4.2. Histograms

An alternative feature space are provided by histograms, which can be related to image-like representations. In this approach, we discretize the space of certain features, such as the distance and bearing to other agents, into a fixed number of bins. This way, we can collect information about neighboring agents in the form of a fixed-size multi-dimensional histogram. Herein, the histogram bins define a feature mapping ϕ^{HIST} using a one-hot-coding for each observed agent. A detailed description of this approach can be found in



(a) neural network embedding policy network

(b) RBF and histogram embedding policy

(c) policy network for concatenation

Figure 1: Illustration of (a) the neural network mean embedding policy, (b) the network architecture used for the RBF and histogram representation, and (c) for the simple concatenation of observations. The numbers inside the boxes denote the dimensionalities of the hidden layers. The color coding in (a) highlights which layers share the same weights. The plus sign denotes the mean of the feature activations.

our previous work (Hüttenrauch et al., 2018). While the approach works well in discrete environments where each cell is only occupied by a single agent, the representation can lead to blurring effects between agents in the continuous case. Moreover, the histogram approach does not scale well with the dimensionality of the feature space.

4.4.3. Radial Basis Functions

A specific problem of the histogram approach is the hard assignment of agents into bins, which results in abrupt changes in the observation space when a neighboring agent moves from one bin to another. A more fine-grained representation can be achieved by using RBF networks with a fixed number of basis functions evenly distributed over the observation space. The resulting feature mapping ϕ^{RBF} is then defined by the activations of each basis function and can be seen as a "soft-assigned" histogram. However, both representations (histogram and RBF) suffer from the curse of dimensionality, as the number of required basis functions typically increases exponentially with the number of dimensions of the observation vector.

4.5. Other Representation Techniques

Inspired by the work of Mordatch and Abbeel (2018), we also investigate a policy function that uses a softmax pooling layer instead of the mean embedding. The elements of the pooling layer $\psi = [\psi_1, \ldots, \psi_K]$ are given by

$$\psi_k = \frac{\sum_{o^{i,j} \in O^i} \exp\left(\alpha \phi_k(o^{i,j})\right) \phi_k(o^{i,j})}{\sum_{o^{i,j} \in O^i} \exp\left(\alpha \phi_k(o^{i,j})\right)}$$

for each feature dimension of $\phi = [\phi_1, \dots, \phi_K]$ with a temperature parameter α. Note that the representation becomes identical to our mean embedding for $\alpha = 0$, while setting $\alpha \gg 1$ results in max-pooling and $\alpha \ll -1$ corresponds to min-pooling. In our experiments, we choose $\alpha = 1$ as a trade-off between a mean embedding and max-pooling and additionally study the performance of max-pooling over each individual feature dimension.

4.6. Adaption of TRPO to the Homogeneous Swarm Setup

Gupta et al. (2017) present a parameter-sharing variant of TRPO that can be used in a multi-agent setup. During the learning phase, the algorithm collects experiences made by all agents and uses these experiences to optimize one policy with a single set of parameters θ. Since, in the swarm setup, we assume homogeneous agents that are potentially indistinguishable to each other, we omit the agent index introduced by Gupta et al. (2017). The optimization problem is expressed using advantage values based on all agents' observations. During execution, however, each agent has only access to its own perception. Hence, the terminology of centralized–learning / decentralized–execution is chosen.

During the trajectory roll-outs, we use a sub-sampling strategy to achieve a trade-off between the number of samples and the variability in advantage values seen by the learning algorithm. Our implementation is based on the OpenAI baselines version of TRPO with 10 MPI workers, where each worker samples 2048 time steps, resulting in $2048N$ samples. Subsequently, we randomly choose the data of 8 agents, yielding $2048 \times 10 \times 8 = 163840$ samples per TRPO iteration. The chosen number of samples worked well throughout our experiments and was not extensively tuned.

5. Experimental Results

Our experiments are designed to study the use of mean embeddings in a cooperative swarm setting. The three main aspects are:

1. How do the different mean embeddings (neural networks, histograms and RBF representation) compare when provided with the same state information content?

2. How does the mean embedding using neural networks perform when provided with additional state information while keeping the dimensionality of the feature space constant?

3. How does the mean embedding of neural network features compare against other pooling techniques?

In this section, we first introduce the swarm model used for our experiments and present the results of different evaluations afterwards. During a policy update, a fixed number of K trajectories are sampled, each yielding a return of $G_k = \sum_{t=1}^{T} r(t)$. The results are presented in terms of the average return, denoted as $\bar{G} = \frac{1}{K} \sum_{k=1}^{K} G_k$. Videos demonstrating the agents' behavior in the different tasks can be found online.[2]

2. http://computational-learning.net/deep_rl_for_swarms

5.1. Swarm Models

Our agents are modeled as unicycles (a commonly used agent model in mobile robotics; see, for example, Egerstedt and Hu, 2001), where the control parameters either manipulate the linear and angular velocities v and ω (single integrator dynamics) or the corresponding accelerations $\dot{v}$ and $\dot{\omega}$ (double integrator dynamics). In the single integrator case, the state of an agent is defined by its location $\boldsymbol{x} = (x, y)$ and orientation ϕ. In case of double integrator dynamics, the agent is additionally characterized by its current velocities. The exact state definition and kinematic models can be found in Appendix A. Note that these agent models are more complex than what is typically considered in optimization-based approaches, which mostly assume single integrator dynamics directly on $\boldsymbol{x}$. Depending on the task, we either opt for a closed state space where the limits act as walls, or a periodic toroidal state space where agents exceeding the boundaries reappear on the opposite side of the space. Either way, the state is bounded by $x_{\max} = y_{\max} = 100$.

We study two different observation scenarios for the agents, i.e., global observability and local observability. In the case of global observability, all agents are neighbors, i.e.

$$\mathcal{N}_{\mathcal{G}}(i) = \{j \in \{1, \dots, N\} \,|\, i \neq j\},$$

which corresponds to a fully connected static interaction graph. For the local observability case, we use Δ-disk proximity graphs, where edges are formed if the distance $d^{i,j} = \sqrt{(x^i - x^j)^2 + (y^i - y^j)^2}$ between agents i and j is less than a pre-defined cut-off distance d_c for communication, resulting in a dynamic interaction graph. The neighborhood set of the graph is then defined as

$$\mathcal{N}_{\mathcal{G}}(i) = \{j \in \{1, \dots, N\} \,|\, i \neq j,\ d^{i,j} \leq d_c\}.$$

For a detailed description of all observational features available to the agents in the tasks, see Appendices B and C.

5.2. Rendezvous

In the rendezvous problem, the goal is to minimize the distances between all agents. The reason why we choose this experiment is because a simple optimization-based baseline controller can be defined by the consensus protocol,

$$\dot{\boldsymbol{x}}^i = -\sum_{j \in \mathcal{N}(i)} (\boldsymbol{x}^i - \boldsymbol{x}^j),$$

where $\boldsymbol{x}^i = (x^i, y^i)$ denotes the location of agent i. To make the solution compatible to the double integrator agent model, we make use of a PD-controller (see Appendix A for details). The reward function for the problem can be found in Appendix E.1.

We evaluate different observation vectors $o^{i,j}$ which are fed into the policy. To compare the histogram and RBF embedding with the proposed neural network approach, we restrict the *basic* observation model (see below) to a set of two features: the distance $d^{i,j}$ between two agents and the corresponding bearing $\phi^{i,j}$. This restriction allows for a comparison to the optimization-based consensus protocol, which is based on displacements (an equivalent formulation of distance and bearing). To show that the neural network embeddings can

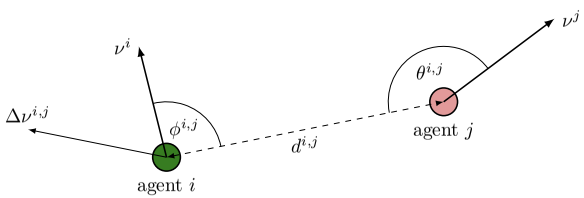


Figure 2: Illustration of two neighboring agents facing the direction of their velocity vectors ν^i and ν^j, along with the observed quantities, shown with respect to agent i. The observed quantities are the bearing $\phi^{i,j}$ to agent j, agent j's relative orientation $\theta^{i,j}$ to agent i, their distance $d^{i,j}$ and a relative velocity vector $\Delta\nu^{i,j} = \nu^i - \nu^j$. In this trivial example, agent i's observed neighborhood size as well as the neighborhood size communicated by agent j are $|\mathcal{N}(i)| = |\mathcal{N}(j)| = 1$.

be used with more informative observations, we further introduce an *extended* set and a communication (*comm*) set. These sets may include relative orientations $\theta^{i,j}$ or relative velocities $\Delta\nu^{i,j}$ (depending on the agent dynamics), as well as the own neighborhood size and those of the neighbors. An illustration of these quantities can be found in Figure 2.

5.2.1. Global Observability

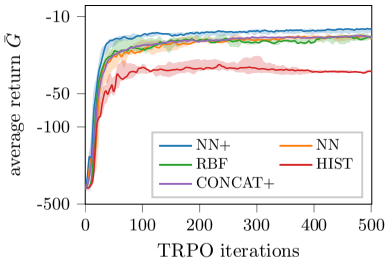
First, we study the rendezvous problem with 20 agents in the global observability setting with double integrator dynamics to illustrate the algorithm's ability to handle complex dynamics. To this end, we compare the performances of policies using histogram, RBF and neural network embeddings on the *basic* set, as well as neural network embeddings on the *extended* set. The observations $o^{i,j}$ in the *basic* set comprise the distance $d^{i,j}$ and bearing $\phi^{i,j}$. In the *extended* set, which is processed only via neural network embeddings, we additionally add neighboring agents' relative orientations $\theta^{i,j}$ and velocities $\Delta\nu^{i,j}$. The local properties o^i_{loc} consist of a shortest distance and orientation to the closest boundary, i.e., d^i_{wall} and ϕ^i_{wall}. The sets are summarized as follows:

$$\begin{aligned} \textit{Basic}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}}\} \\ \textit{Extended}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j},\ \theta^{i,j},\ \Delta\nu^{i,j}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}}\}. \end{aligned}$$

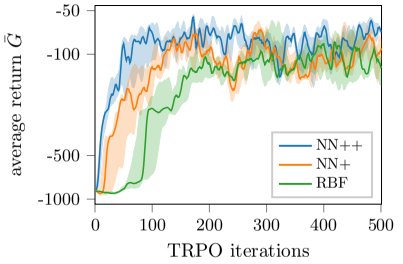
The results are shown in Figure 3a. On first sight, they reveal that all shown methods eventually find a successful strategy, with the histogram approach showing worst performance. Upon a closer look, it can be seen that the best solutions are found with the neural network embedding, in which case the learning algorithm also converges faster, demonstrating that this form of embedding serves as a suitable representation for deep RL. However, there are two important things to note:

- The differences between the approaches seem to be small due to the wide range of obtained reward values, but the NN+ method brings in fact a significant performance gain. Compared to the NN and RBF embedding, the performance of the learned NN+ policy is ∼10% better in terms of the average return of an episode (Figure 3a) and almost twice as good ($\sim 4 \times 10^{-2}$ versus $\sim 8 \times 10^{-2}$) in terms of the mean distance between agents at the steady state solution after around 200 time steps (Figure 5a). Furthermore, the NN+ embedding reaches the mean distance achieved by the NN and RBF embeddings roughly 20 to 30 time steps earlier, which corresponds to an improvement of ∼25%.

- Although the performance gain of NN+ can be partly explained by the use of the extended feature set, experiments with the same feature set using the histogram / RBF approach did *not* succeed to find solutions to the rendezvous problem; hence, the corresponding results are omitted. The reason is that the dimensionality of the input space scales exponentially for the histogram / RBF approach while only linearly for the neural network embedding, which results in a more compact feature representation that keeps the learning problem tractable.



(a) 20 agents with global observability

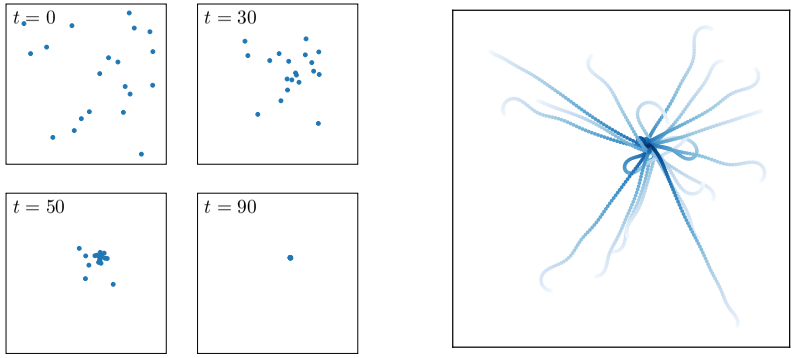


(b) 20 agents with local observability

Figure 3: Learning curves for the rendezvous task with different observation models. The curves show the median of the average return $\bar{G}$ based on the top five trials on a log scale. **Legend:** NN++: neural network mean embedding of *comm* set, NN+: neural network mean embedding of *extended* set, NN: neural network embedding of *basic* set, RBF: radial basis function embedding of *basic* set, HIST: histogram embedding of *basic* set, CONCAT+: simple concatenation of *extended* set.

Together, these two observations suggest that the neural network embedding provides a suitable learning architecture for deep RL, whereas the histogram / RBF approach is only suited for low-dimensional spaces.

Figure 4 shows a visualization of a policy using the neural network mean embedding of the *extended* set. After random initialization, the agents' locations quickly converge to a single point. Figure 5 shows performance evaluations of the best policies found with each of the mean embedding approaches. We plot the evolution of the mean distance between all agents over 1000 episodes with equal starting conditions. We also include the performance of the PD-controller defined in Appendix A. It can be seen in Figures 5a and 5c that the policies using the neural network embeddings decrease the mean distance most quickly and also find



(a) snapshots (b) full episode

Figure 4: Visualization of a learned policy for the pursuit evasion task. The policy is learned and executed by 10 agents using a neural network mean embedding of the *extended* set. Pursuers are illustrated in blue, the evader is highlighted in red. Visualization of a learned policy for the rendezvous task. The policy is learned and executed by 20 agents using a neural network mean embedding of the *extended* set.

the best steady-state solutions among all learning approaches. While the optimization-based solution (PD) eventually drives the mean distance to zero, a small error remains for the learning-based approaches. However, the learned policies are faster in reducing the distance and therefore show a better average reward. Although the optimization-based policy is guaranteed to find an optimal stationary solution, the approach is build for simpler dynamics and hence performs suboptimally in the considered scenario. Note, that the controller gains for this approach have been tuned manually to maximize performance.

In order to show the generalization abilities of the embeddings, we finally evaluate the obtained policies (except for the concatenation) with 100 agents. The results are displayed in Figure 5b. Again, the neural network embedding of the *extended* set is quickest in reducing the inter-agent distances, resulting in the best overall performance.

5.2.2. Local Observability

The local observability case is studied with 20 agents and a communication cut-off distance of $d_c = 40$. Due to the increased difficulty of the task, we resort to single integrator dynamics for this experiment. Again, we evaluate the *basic* and the *extended* set, which in this case contains the single integrator state information. Accordingly, we remove the relative velocities from the information sets. Moreover, we employ a local communication strategy that transmits the number of observed neighbors as additional information. Note that this information can be used by the agents to estimate in which direction the center of mass of the swarm is located.

While the received neighborhood sizes $\{|\mathcal{N}(j)|\}_{j\in\mathcal{N}(i)}$ are treated as part of agent i's local observation of the swarm, the own perceived neighborhood size $|\mathcal{N}(i)|$ is considered as part of the local features o^i_{loc}. The observation models for the local observability case are thus summarized as:

$$\begin{aligned}
\textit{Basic}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}}\} \\
\textit{Extended}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j},\ \theta^{i,j}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}}\} \\
\textit{Comm}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j},\ \theta^{i,j},\ |\mathcal{N}(j)|\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}},\ |\mathcal{N}(i)|\}.
\end{aligned}$$

For the experiment, we limit our comparison to RBF embeddings (which showed best performance among all non-neural-network solutions) of the *basic* set and neural network embeddings of the *extended* set and the *comm* set. The results are illustrated in Figure 3b, which shows that the neural network embeddings lead to a quicker learning progress. Furthermore, by introducing the *comm* model, a higher return is achieved. Compared to the global observability case, however, the learning process exhibits an increased variance caused by the information loss in the reward signal (see Appendix E).

Figure 5c illustrates the performances of the learned policies. Again, the neural network embedding is quicker in reducing the inter-agent distances and converges to better steady-state solutions. In order to test the efficacy of the communication protocol, we further evaluate the learned policies with 10 agents. The results are displayed in Figure 5d. As expected, the performance decreases due to the lower chance of agents seeing each other but we still notice a benefit caused by the communication.

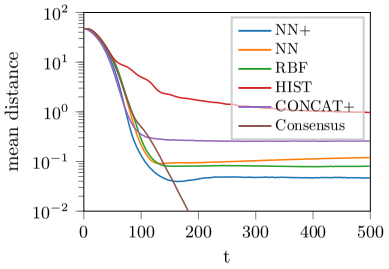
5.3. Pursuit Evasion with a Single Evader

Our implementation of the pursuit evasion scenario is based on the work by Zhou et al. (2016), from which we adopt the evader strategy. The strategy is based on Voronoi regions, which the pursuers try to minimize and the evader tries to maximize. While the original paper considers a closed world, we change the world type from closed to periodic, thereby making it impossible to trap the evader in a corner. In order to encourage a higher level of coordination between the agents, we set the evader's maximum velocity to twice the pursuers' maximum velocity. An episode ends once the evader is caught, i.e., if the distance of the closest pursuer is below a certain threshold. In all our experiments, the evader policy is fixed and not part of the learning process. The reward function for the problem is based on the shortest distance of the closest pursuer and can be found in Appendix E.2.

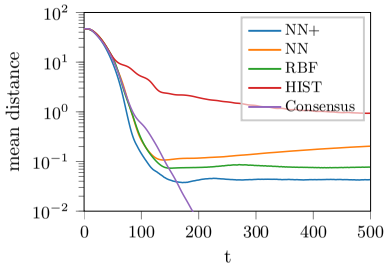
5.3.1. Global Observability

Again, we study the global observability case with ten agents. Since the pursuit of an evader is a more challenging task already, we reduce the movement complexity to single integrator dynamics. The *basic* and *extended* set are equal to those in the rendezvous experiment with single integrator dynamics, with additional information about the evader in the local properties o^i_{loc}. In here, we add the distance $d^{i,e}$ and bearing $\phi^{i,e}$ of agent i to the evader e. Accordingly, the observation sets are given as:

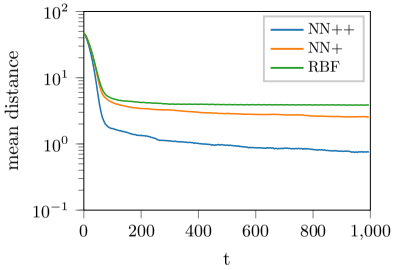
$$\begin{aligned}
\textit{Basic}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}},\ d^{i,e},\ \phi^{i,e}\} \\
\textit{Extended}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j},\ \theta^{i,j}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}},\ d^{i,e},\ \phi^{i,e}\}.
\end{aligned}$$



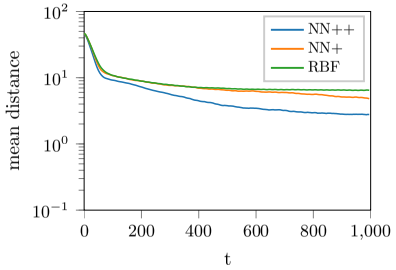
(a) 20 agents (global observability)



(b) 100 agents (global observability)



(c) 20 agents (local observability)



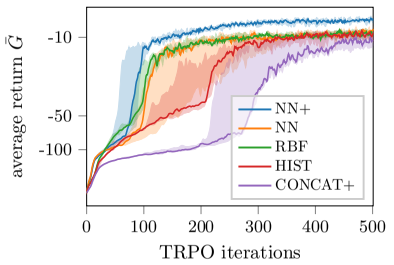
(d) 10 agents (local observability)

Figure 5: Comparison of the mean distance between agents in the rendezvous experiment achieved by the best learned policies and the consensus protocol. In (a) and (b), the policy is learned with 20 agents and executed by 20 and 100 agents, respectively. In (c) and (d), the policy is learned with 20 agents and executed by 20 and 10 agents. Results are averaged over 1000 episodes with identical starting conditions.

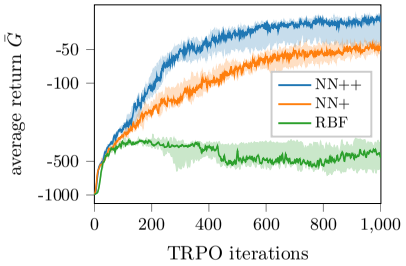
The results in Figure 6a reveal that successful strategies can be obtained with all methods. However, this time, a clear advantage can be seen for the policies using neural network mean embeddings of the *extended* set, both in terms of behavior quality and in the number of samples necessary to find the solution.

Figure 7 illustrates the strategy that such a policy exerts. After random initialization, the agents first spread in a way that leaves no possibility for the evader to increase its Voronoi region, thereby keeping the evader almost on the same spot. Once this configuration is reached, they surround the evader in a circular pattern and start to reduce the distance until one pursuer successfully reaches the distance threshold.

To investigate the performance of the best mean embedding policies (learned with 10 agents), we estimate the corresponding probabilities that the evader is caught within a certain time frame. For the sake of completeness, we also include the method proposed by Zhou

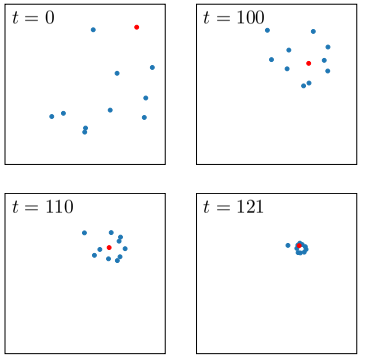


(a) 10 agents with global observability

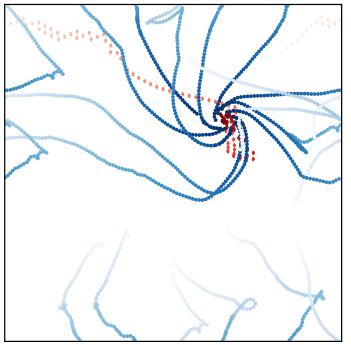


(b) 20 agents with local observability

Figure 6: Learning curves for the pursuit evasion task with different observation models. The curves show the median of the average return $\bar{G}$ based on the top five trials on a log scale. **Legend:** NN++: neural network mean embedding of *comm* set, NN+: neural network mean embedding of *extended* set, RBF: radial basis function embedding of *basic* set, HIST: histogram embedding of *basic* set, CONCAT+: concatenation of *extended* set.



(a) snapshots



(b) full episode

Figure 7: Visualization of a learned policy for the pursuit evasion task. The policy is learned and executed by 10 agents using a neural network mean embedding of the *extended* set. Pursuers are illustrated in blue, the evader is highlighted in red.

et al. (2016), which was originally not designed for a setup with a faster evader, though. The results are plotted in Figure 8 as the fraction of episodes ending at the respective time instant, averaged over 1000 episodes. The plot in Figure 8b reveals that the evader may be caught using all presented methods if the policies are executed for long time periods. As already indicated by the learning curves, using a neural network mean embedding representation yields the quickest capture among all methods. The additional information in the *extended* set further increases performance.

Next, we examine the generalization abilities of the learned policies, this time on scenarios with 5, 20 and 50 agents (Figures 8a,c,d). Increasing the amount of agents leads to a quicker capture for all methods; however, the best performance is still shown by the agents executing a neural network policy based on embeddings of the *extended* set. Interestingly, when using fewer agents than in the original setup (Figure 8a), all methods struggle to capture the evader. After inspection of the behavior, we found that the strategy of establishing a circle around the evader causes too large gaps between the agents through which the evader can escape.

5.3.2. Local Observability

The local observability case is studied with 20 agents and a communication cut-off distance of $d_c = 40$. Additionally, we introduce an observation radius $d_o = 20$ within which the pursuers can observe the distance and bearing to the evader. We reuse the *basic* and *extended* set from last section and modify the *comm* set to include the shortest path information of other agents in the neighborhood of agent i to the evader. This way, each agent i can compute a shortest path to the evader over a graph of connected agents, such that the path $P = (v^1, v^2, \dots, v^M)$ minimizes the sum $d^{i,e}_{\text{min}} = \sum_{m=1}^{M-1} d^{m,m+1}$ where v^1 represents agent i and v^M is the evader. The observation sets are given as:

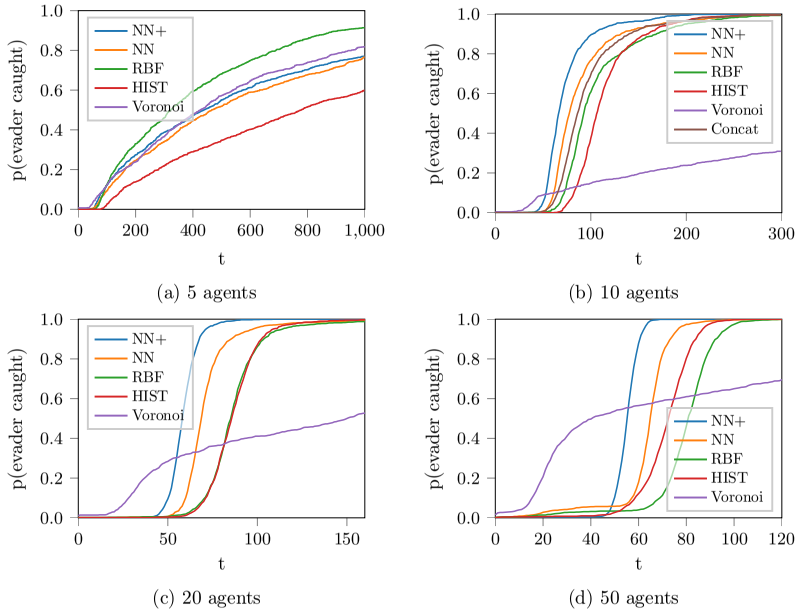
$$
\begin{aligned}
\textit{Basic}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}},\ d^{i,e},\ \phi^{i,e}\} \\
\textit{Extended}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j},\ \theta^{i,j}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}},\ d^{i,e},\ \phi^{i,e}\} \\
\textit{Comm}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j},\ \theta^{i,j},\ d^{j,e}_{\text{min}}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}},\ d^{i,e},\ \phi^{i,e},\ d^{i,e}_{\text{min}}\}.
\end{aligned}
$$

Note that in this case the distance and bearing to an evader are only available if $d^{i,e} \leq d_o$. Furthermore, the correct shortest path is only available if an agent and the evader are in the same sub-graph, otherwise, a pre-defined value is fed into the policy.

Again, we limit the comparison for the local observability case to the more promising methods of neural network and RBF mean embeddings. The results in Figure 6b show that the performance gain of the neural network mean embeddings is even more noticeable than in the global observability case, with a clear advantage in the presence of the local communication protocols. The inspection of the termination probabilities in Figure 9 confirms that the neural network mean embedding results in a significantly improved policy.

5.4. Pursuit Evasion with Multiple Evaders

Lastly, we study a pursuit evasion scenario with multiple evaders, i.e., we assume that agent i receives observation samples $\{o^{i,e}\}$ from several evaders, which are processed using a second mean embedding to account for the variable set size. Where in the previous experiment



(a) 5 agents (b) 10 agents

(c) 20 agents (d) 50 agents

Figure 8: Performance comparison of the best learned policies and the optimization approach minimizing Voronoi regions in the pursuit evasion task with global observability. The curves show the probability that the evader is caught after t time steps. All policies are learned with 10 agents but executed with different agent numbers, as indicated below each subfigure. Results are averaged over 1000 episodes with identical starting conditions.

the agents had precise information about the evader in terms of distance and bearing, they now have to extract this information from the respective embedding. An additional level of difficulty results from the fact that the reward function no longer provides any guidance in terms of the distances to the evaders since it only counts the number of evaders caught in each time step (see Appendix E.3 for details).

We study a scenario with 50 pursuers and 5 evaders using the global observability setup in Section 5.3.1, except that we respawn caught evaders to a new random location instead of terminating the episode. The observation sets, containing the same type of information but arranged according to the inputs of the neural networks, are designed as follows:

$$
\begin{aligned}
\textit{Basic}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j}\} && o^{i,e} = \{d^{i,e},\ \phi^{i,e}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}}\} \\
\textit{Extended}: \quad & o^{i,j} = \{d^{i,j},\ \phi^{i,j},\ \theta^{i,j}\} && o^{i,e} = \{d^{i,e},\ \phi^{i,e}\} && o^i_{\text{loc}} = \{d^i_{\text{wall}},\ \phi^i_{\text{wall}}\}.
\end{aligned}
$$

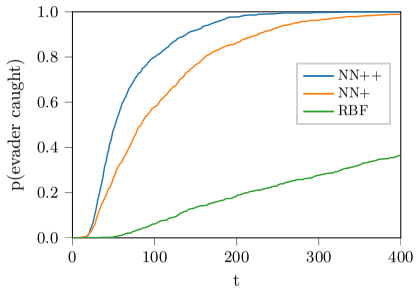


Figure 9: Performance comparison of the best policies in the pursuit evasion task with local observability. The curves show the probability that the evader is caught after t time steps. All policies are learned and executed by 20 agents. Results are averaged over 1000 episodes with identical starting conditions.

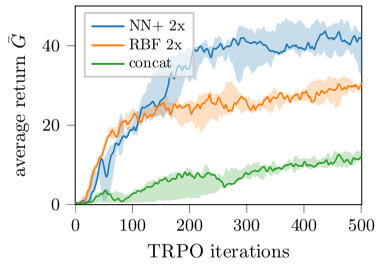
Figure 10a shows the learning curves for policies with neural network and RBF mean embeddings and for the concatenation approach. The return directly relates to the number of evaders caught during an episode. Again, the neural network mean embedding performs significantly better than the RBF embedding. The curves clearly indicate the positive influence of the additional information present in the *extended* set. With this amount of agents, the dimensionality of the concatenation has increased to a point where learning is no longer feasible.

5.5. Evaluation of Pooling Functions

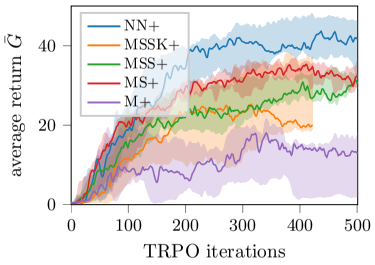
Figure 11 shows learning curves of policies based on mean embeddings, softmax pooling, and max-pooling (as described in Section 4.5) of features of the *extended* set for the rendezvous and pursuit evasion task with global observability.

In the rendezvous task (Figure 11a), all pooling techniques eventually manage to find a good solution. Policies using neural network mean embedding, however, on average converge more quickly while policies using max-pooling show slightly worse performance. Given its reduced computational complexity compared to the softmax-pooling, the mean embedding provides the most effective approach among all proposed architectures.

When examining the results of the pursuit evasion task (Figure 11b), we find that the algorithm produces two distinct solutions. A sub-optimal one, which is only able to circle the evader but is unable to catch it (a catch is realized if the distance of the closest pursuer to the evader is below a certain threshold), and a solution which additionally catches the evader after a short period of time. Therefore, we not only report the performance of the top 5 trials out of 16, but also provide the number of times the algorithm was able to discover the better of the two solution (Table 1). Once that the algorithm finds a good solution, the mean embedding and softmax solutions perform comparably well but the max-pooling approach shows a significantly worse performance. More importantly, however, the algorithm was

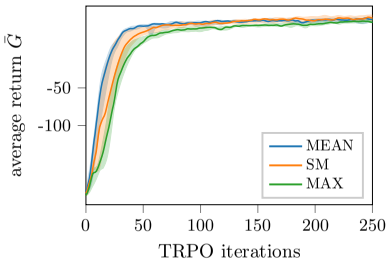


(a) Evaluation of different observation models and embedding techniques.

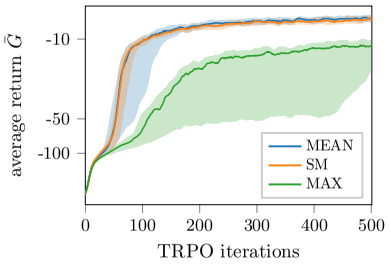


(b) Comparison of the proposed mean embedding policy to policies with statistics of the observations as input.

Figure 10: Learning curves for 50 agent pursuit evasion with 5 evaders. The curves show the median of the average return $\bar{G}$ based on the top five trials. **Legend:** NN+ 2x: two neural network mean embeddings of the *extended* set, RBF 2x: two radial basis function mean embeddings of the *basic* set, concat: simple concatenation of *extended* set. MSSK+, MSS+, MS+ and M+: Combinations of mean, standard deviation, skew and kurtosis of the features in the *extended* set.



(a) Rendezvous with 20 agents and global observability.



(b) Pursuit evasion with 10 pursuers and global observability.

Figure 11: Learning curves of different embedding and pooling architectures based on the *extended* set. The curves show the median of the average return $\bar{G}$ based on the top five trials on a log scale. **Legend:** MEAN: neural network mean embedding, SM: softmax feature pooling, MAX: max feature pooling.

able to find the good solution more often using the mean embedding than using the other pooling approaches.

mean	sm	max
10/16	6/16	4/16

Table 1: Number of times the algorithm discovered policies that led to a successful catch.

5.6. Comparison to Moment-Based Representations

Finally, we compare the mean embedding of neural network features to a representation using statistics of the input. Figure 10b shows an evaluation on the pursuit evasion task with 50 agents and 5 evaders. Here, we use combinations of the empirical mean, standard deviation, skew and kurtosis of each feature of the *extended* set as the input to a policy function. The plot reveals that neural network mean embeddings can capture more relevant information about the characteristics of the distribution of agents than simple statistics of the elements in the *extended* set. Similar results were obtained for the other tasks although the differences in performance were less pronounced.

5.7. Computational Complexity

Unlike in classical optimization-based control, where the controller is derived from an assumed dynamics model, model-free reinforcement learning methods like TRPO find their control policies through interaction with the environment, without requiring explicit knowledge of the underlying system dynamics. While this comes at the cost of an additional exploration phase, learning-based approaches typically offer an increased flexibility in that the same control architecture can adapt to different tasks and environments, without being affected by potential model mismatches. More importantly, considering the final learned policy from a computational perspective, the synthesis of the control signal involves no additional conceptual steps compared to an optimization-based approach.

While a typical experiment with 20 agents in our setup takes between four and six hours of training on a machine with ten cores (sampling trajectories in parallel), a forward pass through the trained neural network to compute the instantaneous control signal takes only about 1 ms, which enables an execution in real time. Furthermore, all control strategies learned through our framework are decentralized, which allows an arbitrary system size scaling in a real swarm network, where the required computations are naturally distributed over all agents. When learning new policies, the memory requirements scale $\mathcal{O}(N(N-1))$ with the number of agents (assuming global observability) since we need to store the local views of all agents. However, decentralized execution after the policy is learned scales linearly in N per agent. An incremental online computation of the mean can be chosen if memory restrictions exist (Finch, 2009).

For comparison, the complexity of calculating Voronoi regions for the pursuit evasion policy scales $\mathcal{O}(N \log N)$ with the number of agents (Aurenhammer, 1991). Concerning the system sizes considered in our experiments, the resulting computation time of both policy types is in the same order of magnitude during task execution.

6. Conclusion

In this paper, we proposed the use of mean feature embeddings as state representations to overcome two major problems in deep reinforcement learning for swarms: the high and possibly changing dimensionality of information perceived by each agent. We introduced three different approaches to realize such embeddings — two manually designed approaches based on histograms / radial basis functions and an end-to-end learned neural network feature representation. We evaluated the approaches on different variations of the rendezvous and pursuit evasion problem and compared their performances to that of a naive feature concatenation method and classical approaches found in the literature. Our evaluation revealed that learning embeddings end-to-end using neural network features scales well with increasing agent numbers, leads to better performing policies, and often results in faster convergence compared to all other approaches. As expected, the naive concatenation approach fails for larger system sizes.

Acknowledgments

The research leading to these results has received funding from EPSRC under grant agreement EP/R02572X/1 (National Center for Nuclear Robotics). Calculations for this research were conducted on the Lichtenberg high performance computer of the TU Darmstadt.

Appendix A. Agent Kinematics

In the single integrator case, the state of an agent is given by $s^i = [x^i, y^i, \phi^i] \in \mathcal{S} = \{[x, y, \phi] \in \mathbb{R}^3 : 0 \leq x \leq x_{\max},\ 0 \leq y \leq y_{\max},\ 0 \leq \phi < 2\pi\}$, and the linear velocity v and angular velocity ω can be directly controlled by the agent. The kinematic model is given by

$$\begin{aligned}\dot{x} &= v\cos\phi\\ \dot{y} &= v\sin\phi\\ \dot{\phi} &= \omega.\end{aligned}$$

In the double integrator case, the state is given by $s^i = [x^i, y^i, \phi^i, v^i, \omega^i] \in \mathcal{S} = \{[x, y, \phi, v, \omega] \in \mathbb{R}^5 : 0 \leq x \leq x_{\max},\ 0 \leq y \leq y_{\max},\ 0 \leq \phi < 2\pi, |v| \leq v_{\max}, |\omega| \leq \omega_{\max}\}$ and the agent can only indirectly change its velocity by acceleration. With the control inputs a_v and a_ω, the model is then given by

$$\begin{aligned}\dot{v} &= a_v\\ \dot{\omega} &= a_\omega\\ \dot{x} &= v\cos\phi\\ \dot{y} &= v\sin\phi\\ \dot{\phi} &= \omega.\end{aligned}$$

For the experiments, we use finite differences to model the system in discrete time.

Appendix B. Observation Model

Irrespective of the task, an agent i can sense the following properties about other agents $j \in \mathcal{N}(i)$ within its neighborhood:

$$
\begin{aligned}
d^{i,j} & && \text{distance to neighboring agents} \\
\phi^{i,j} &= \arctan(\frac{y^j - y^i}{x^j - x^i}) - \phi^i && \text{bearing to neighboring agents} \\
\theta^{i,j} &= \arctan(\frac{y^i - y^j}{x^i - x^j}) - \phi^j && \text{relative orientation} \\
\Delta v^{i,j} &= v^i \left[\cos\phi^i, \sin\phi^i\right] - v^j \left[\cos\phi^j, \sin\phi^j\right] && \text{relative velocity}
\end{aligned}
$$

Furthermore, each agent has access to the following local properties:

$$
\begin{aligned}
d^i_{\text{wall}} &= \min(x^i - x_{\min},\ y^i - y_{\min},\ x_{\max} - x^i, y_{\max} - y^i) && \text{distance to closest wall} \\
\phi^i_{\text{wall}} &= \varphi^i_{\text{wall}} - \phi^i && \text{orientation to closest wall} \\
v^i, \omega^i & && \text{own velocity}
\end{aligned}
$$

where φ^i_{wall} denotes the absolute bearing of agent i to the closest wall segment.

Appendix C. Task Specific Communication Protocols

In the rendezvous task, agent i additionally can sense information about neighborhood sizes:

$$
\begin{aligned}
|\mathcal{N}(i)| & && \text{own neighborhood size} \\
|\mathcal{N}(j)| : j \in \mathcal{N}(i) & && \text{neighborhood size of neighbor } j
\end{aligned}
$$

In pursuit evasion, we additionally have one or multiple evaders with states $s^e = [x^e, y^e] \in \{[x, y] \in \mathbb{R}^2 : 0 \le x \le x_{\max},\ 0 \le y \le y_{\max}\}$. Agents can sense the distance and bearing to an evader, given that the evader is within an observation distance d_o:

$$
\begin{aligned}
d^{i,e} &= \sqrt{(x^i - x^e)^2 + (y^i - y^e)^2} \quad \text{if } d^{i,e} \le d_o && \text{distance to evader} \\
\phi^{i,e} &= \arctan(\frac{y^e - y^i}{x^e - x^i}) - \phi^i \quad \text{if } d^{i,e} \le d_o && \text{bearing to evader}
\end{aligned}
$$

Furthermore, we assume that each agent i can compute a shortest path to the evader over a graph of connected agents, such that the path $P = (v^1, v^2 \ldots, v^M)$ minimizes the sum $\sum_{m=1}^{M-1} d^{m,m+1}$ where v^1 is agent i and v^M is the evader.

Appendix D. Controller for Double Integrator Dynamics

We use a simple PD-controller to transform the consensus protocol with high-level direct state manipulation to the unicycle model with double integrator dynamics. It is given by

$$
\begin{aligned}
a_v &= K_1(v_d - v) \\
a_\omega &= K_2(\phi_d - \phi) + D_2(\omega_d - \omega) \\
v_d &= \|\dot{\boldsymbol{x}}\| \\
\phi_d &= \arctan(\frac{\dot{y}}{\dot{x}}) \\
\omega_d &= 0,
\end{aligned}
$$

where the parameters K_1, K_2 and D_2 are tuned manually to give good performance on the problem.

Appendix E. Reward Functions

E.1. Rendezvous

The reward function is defined in terms of the inter-agent distances $\{d^{i,j}\}$ as

$$
R(\boldsymbol{s}, \boldsymbol{a}) = \alpha \sum_{i=1}^{N} \sum_{j=i+1}^{N} \min(d^{i,j}, d_c) + \beta \|\boldsymbol{a}\|,
$$

where in the global observability case we set the cut-off distance $d_c = \max(x_{\max}, y_{\max})$ to the maximum possible inter-agent distance in the respective environment. The factor $\alpha = -\left(\frac{N(N-1)}{2} d_c\right)^{-1}$ serves as a reward normalization factor and $\beta = -1 \times 10^{-3}$ controls how strongly high action outputs of the policy are penalized.

E.2. Pursuit Evasion

For the case of a single evader, the pursuit evasion objective may be expressed in terms of the distance to the closest pursuer. More specifically, the reward function is given as

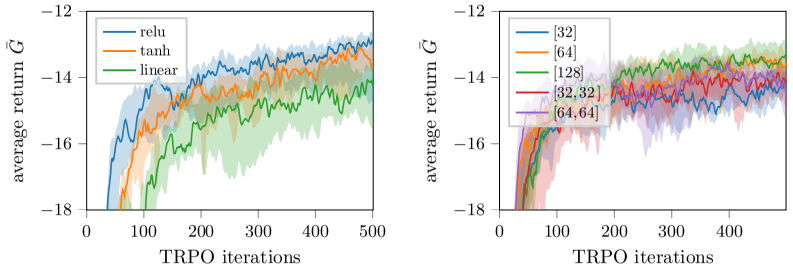
$$
R(\boldsymbol{s}, \boldsymbol{a}) = -\frac{1}{d_o} \min(d_{\min}, d_o),
$$

where $d_{\min} = \min(d^{1,e}, \ldots, d^{N,e})$. For the global observability case, we set d_o to the maximum possible distance of $d^{i,e}$.

E.3. Pursuit Evasion with Multiple Evaders

In the case of multiple evaders, we use a sparser reward function that counts how many evaders are caught per time step, with no additional guidance of inter-agent distances. An evader e is assumed to be caught if the closest pursuer's distance $d_{\min,e} = \min(d^{1,e}, \ldots, d^{N,e})$ is closer than a threshold distance $d_t = 3$. The reward function is given by

$$
R(\boldsymbol{s}, \boldsymbol{a}) = \sum_{e=1}^{E} \mathbf{1}_{[0,d_t]}(d_{\min,e}),
$$



(a) Activation functions evaluation.

(b) Layer size evaluation.

Figure 12: Learning curves for 20 agent rendezvous with (a) different activation functions for the mean embedding and (b) different layer numbers and sizes using a RELU activation function. The curves show the median of the average return $\bar{G}$ based on the top five trials.

where E is the number of evaders and

$$\mathbf{1}_{[a,b]}(x) = \begin{cases} 1 & \text{if } x \in [a,b] \\ 0 & \text{else} \end{cases}$$

is the indicator function.

Appendix F. Policy Architectures

This section briefly summarizes the chosen policy architectures. Illustrations can be found in Figure 1.

F.1. Neural Network Embedding Policy

We evaluated different layer sizes and activation functions on the rendezvous problem and show the results in Figure 12. In all other experiments, the neural network mean feature embedding for agent i, given by

$$\phi^{\text{NN}}(O^i) = \frac{1}{|O^i|} \sum_{o^{i,j} \in O^i} \phi(o^{i,j}),$$

is realized as the empirical mean of the outputs of a single layer feed-forward neural network,

$$\phi(o^{i,j}) = h(W o^{i,j} + b),$$

with 64 neurons and a RELU non-linearity h.

F.2. Histogram Embedding Policy

The histogram embedding is achieved with a two-dimensional histogram over the distance and bearing space to other agents. We use eight evenly spaced bins for each feature, resulting in a 64 dimensional feature vector.

F.3. RBF Embedding Policy

The RBF embedding is given by a vector $\phi^{\text{RBF}}(O^i) = \left[\psi_1(O^i), \ldots, \psi_{M^2}(O^i)\right]$ of M^2 contributions from $M = 8$ radial basis functions whose center points are evenly distributed in the distance and bearing space. With $o^{i,j} = [d^{i,j}, \phi^{i,j}]$, $\mu_m = [\mu_{d,m}, \mu_{\phi,m}]$, and $\sigma = [\sigma_d, \sigma_\phi]$ its components are given by

$$\psi_m(O^i) = \sum_{o^{i,j} \in O^i} \rho_m(o^{i,j}),$$

where we choose

$$\rho_m(o^{i,j}) = \exp\left(-\frac{1}{2}\left[\frac{(d^{i,j} - \mu_{d,m})^2}{\sigma_d^2} + \frac{(\phi^{i,j} - \mu_{\phi,m})^2}{\sigma_\phi^2}\right]\right).$$

The policy network structure used for both, the histogram and the RBF representations, is illustrated in Figure 1b.

F.4. Concatenation Policy

For the concatenation method, we first concatenate agent i's neighborhood observations contained in the set O^i and process them with one hidden layer of 64 neurons and a RELU non-linearity. The resulting feature vector is then concatenated with the local properties o^i_{loc} and fed into a second layer of same size. Finally, the output of the second layer is mapped to the action. The corresponding policy network structure can be seen in Figure 1c.

References

Maruan Al-Shedivat, Trapit Bansal, Yura Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. In *International Conference on Learning Representations*, 2018.

Franz Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.

Levent Bayındır. A review of swarm robotics tasks. *Neurocomputing*, 172:292–321, 2016.

Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.

Mo Chen, Zhengyuan Zhou, and Claire J Tomlin. Multiplayer reach-avoid games via low dimensional solutions and maximum matching. In *American Control Conference (ACC)*, pages 1444–1449, 2014a.

Mo Chen, Zhengyuan Zhou, and Claire J Tomlin. A path defense approach to the multiplayer reach-avoid game. In *IEEE 53rd Annual Conference on Decision and Control (CDC)*, pages 2420–2426, 2014b.

Mo Chen, Zhengyuan Zhou, and Claire J Tomlin. Multiplayer reach-avoid games via pairwise outcomes. *IEEE Transactions on Automatic Control*, 62(3):1451–1457, 2017.

Timothy H. Chung, Geoffrey A. Hollinger, and Volkan Isler. Search and pursuit-evasion in mobile robotics. *Autonomous Robots*, 31(4):299, 2011.

D. V. Dimarogonas and K. J. Kyriakopoulos. On the rendezvous problem for multiple nonholonomic agents. *IEEE Transactions on Automatic Control*, 52(5):916–922, 2007.

M. Egerstedt and Xiaoming Hu. Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation*, 17(6):947–951, 2001.

Tony Finch. Incremental calculation of weighted mean and variance. *University of Cambridge*, 4:11–5, 2009.

G. H. W. Gebhardt, K. Daun, M. Schnaubelt, and G. Neumann. Learning robust policies for object manipulation with robot swarms. In *IEEE International Conference on Robotics and Automation*, 2018.

M. C. De Gennaro and A. Jadbabaie. Decentralized control of connectivity for multi-agent systems. In *IEEE Conference on Decision and Control*, pages 3628–3633, 2006.

Arthur Gretton, Kenji Fukumizu, Choon H Teo, Le Song, Bernhard Schölkopf, and Alex J Smola. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems*, pages 585–592, 2008.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

Aditya Grover, Maruan Al-Shedivat, Jayesh K Gupta, Yura Burda, and Harrison Edwards. Learning policy representations in multiagent systems. `arXiv:1806.06464`, 2018.

Jayesh K. Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 66–83, 2017.

Julia Handl and Bernd Meyer. Ant-based and swarm-based clustering. *Swarm Intelligence*, 1(2):95–113, 2007.

Matthew Hausknecht and Peter Stone. Deep recurrent Q-learning for partially observable MDPs. In *AAAI Fall Symposium Series*, 2015.

Maximilian Hüttenrauch, Adrian Šošić, and Gerhard Neumann. Local communication protocols for learning complex swarm behaviors with deep reinforcement learning. In *International Conference on Swarm Intelligence*, 2018.

A. Jadbabaie, Jie Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

M. Ji and M. Egerstedt. Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics*, 23(4):693–703, 2007.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. `arXiv:1509.02971`, 2015.

J. Lin, A. Morse, and B. Anderson. The multi-agent rendezvous problem. Part 2: The asynchronous case. *SIAM Journal on Control and Optimization*, 46(6):2120–2147, 2007.

Zhiyun Lin, M. Broucke, and B. Francis. Local control strategies for groups of mobile autonomous agents. *IEEE Transactions on Automatic Control*, 49(4):622–629, 2004.

Ryan Lowe, YI WU, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, pages 6379–6390, 2017.

L. Matignon, G. J. Laurent, and N. L. Fort-Piat. Hysteretic Q-learning: An algorithm for decentralized reinforcement learning in cooperative multi-agent teams. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 64–69, 2007.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. In *AAAI Conference on Artificial Intelligence*, 2018.

Shayegan Omidshafiei, Jason Pazis, Christopher Amato, Jonathan P. How, and John Vian. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *International Conference on Machine Learning*, pages 2681–2690, 2017.

Gregory Palmer, Karl Tuyls, Daan Bloembergen, and Rahul Savani. Lenient multi-agent deep reinforcement learning. `arXiv:1707.04402`, 2017.

Liviu Panait, Keith Sullivan, and Sean Luke. Lenient learners in cooperative multiagent systems. In *International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 801–803, 2006.

B. Ranjbar-Sahraei, F. Shabaninia, A. Nemati, and S. D. Stan. A novel robust decentralized adaptive fuzzy control for swarm formation of multiagent systems. *IEEE Transactions on Industrial Electronics*, 59(8):3124–3134, 2012.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. arXiv:1803.11485, 2018.

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. arXiv:1511.06295, 2015.

Muhammad Saleem, Gianni A Di Caro, and Muddassar Farooq. Swarm intelligence based routing protocol for wireless sensor networks: Survey and future directions. *Information Sciences*, 181(20):4597–4624, 2011.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

Alex Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A Hilbert space embedding for distributions. In *International Conference on Algorithmic Learning Theory*, pages 13–31, 2007.

Adrian Šošić, Wasiur R KhudaBukhsh, Abdelhak M Zoubir, and Heinz Koeppl. Inverse reinforcement learning in swarm systems. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 1413–1421, 2017.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning. arXiv:1706.05296, 2017.

Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. Mean field multi-agent reinforcement learning. arXiv:1802.05438, 2018.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in Neural Information Processing Systems*, pages 3391–3401, 2017.

Lianmin Zheng, Jiacheng Yang, Han Cai, Weinan Zhang, Jun Wang, and Yong Yu. MAgent: A many-agent reinforcement learning platform for artificial collective intelligence. arXiv:1712.00600, 2017.

Zhengyuan Zhou, Ryo Takei, Haomiao Huang, and Claire J Tomlin. A general, open-loop formulation for reach-avoid games. In *IEEE 51st Annual Conference on Decision and Control (CDC)*, pages 6501–6506, 2012.

Zhengyuan Zhou, Wei Zhang, Jerry Ding, Haomiao Huang, Dušan M Stipanović, and Claire J Tomlin. Cooperative pursuit with Voronoi partitions. *Automatica*, 72:64–72, 2016.

Journal of Machine Learning Research 20 (2019) 1-21 Submitted 9/18; Revised 3/19; Published 3/19

Neural Architecture Search: A Survey

Thomas Elsken THOMAS.ELSKEN@DE.BOSCH.COM
Bosch Center for Artificial Intelligence
71272 Renningen, Germany
and University of Freiburg

Jan Hendrik Metzen JANHENDRIK.METZEN@DE.BOSCH.COM
Bosch Center for Artificial Intelligence
71272 Renningen, Germany

Frank Hutter FH@CS.UNI-FREIBURG.DE
University of Freiburg
79110 Freiburg, Germany

Editor: Sebastian Nowozin

Abstract

Deep Learning has enabled remarkable progress over the last years on a variety of tasks, such as image recognition, speech recognition, and machine translation. One crucial aspect for this progress are novel neural architectures. Currently employed architectures have mostly been developed manually by human experts, which is a time-consuming and error-prone process. Because of this, there is growing interest in automated *neural architecture search* methods. We provide an overview of existing work in this field of research and categorize them according to three dimensions: search space, search strategy, and performance estimation strategy.

Keywords: Neural Architecture Search, AutoML, AutoDL, Search Space Design, Search Strategy, Performance Estimation Strategy

1. Introduction

The success of deep learning in perceptual tasks is largely due to its automation of the feature engineering process: hierarchical feature extractors are learned in an end-to-end fashion from data rather than manually designed. This success has been accompanied, however, by a rising demand for *architecture engineering*, where increasingly more complex neural architectures are designed manually. *Neural Architecture Search* (NAS), the process of automating architecture engineering, is thus a logical next step in automating machine learning. Already by now, NAS methods have outperformed manually designed architectures on some tasks such as image classification (Zoph et al., 2018; Real et al., 2019), object detection (Zoph et al., 2018) or semantic segmentation (Chen et al., 2018). NAS can be seen as subfield of AutoML (Hutter et al., 2019) and has significant overlap with hyperparameter optimization (Feurer and Hutter, 2019) and meta-learning (Vanschoren, 2019). We categorize methods for NAS according to three dimensions: search space, search strategy, and performance estimation strategy:

©2019 Thomas Elsken, Jan Hendrik Metzen and Frank Hutter.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-598.html.

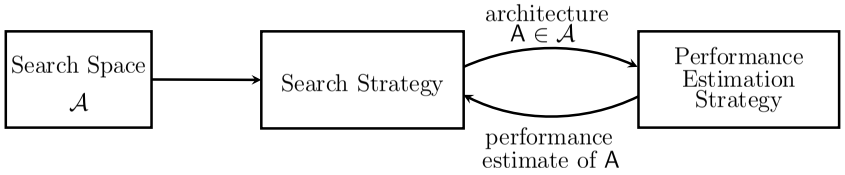


Figure 1: Abstract illustration of Neural Architecture Search methods. A search strategy selects an architecture A from a predefined search space $\mathcal{A}$. The architecture is passed to a performance estimation strategy, which returns the estimated performance of A to the search strategy.

- **Search Space.** The search space defines which architectures can be represented in principle. Incorporating prior knowledge about typical properties of architectures well-suited for a task can reduce the size of the search space and simplify the search. However, this also introduces a human bias, which may prevent finding novel architectural building blocks that go beyond the current human knowledge.
- **Search Strategy.** The search strategy details how to explore the search space (which is often exponentially large or even unbounded). It encompasses the classical exploration-exploitation trade-off since, on the one hand, it is desirable to find well-performing architectures quickly, while on the other hand, premature convergence to a region of suboptimal architectures should be avoided.
- **Performance Estimation Strategy.** The objective of NAS is typically to find architectures that achieve high predictive performance on unseen data. *Performance Estimation* refers to the process of estimating this performance: the simplest option is to perform a standard training and validation of the architecture on data, but this is unfortunately computationally expensive and limits the number of architectures that can be explored. Much recent research therefore focuses on developing methods that reduce the cost of these performance estimations.

We refer to Figure 1 for an illustration. The article is also structured according to these three dimensions: we start with discussing search spaces in Section 2, cover search strategies in Section 3, and outline performance estimation methods in Section 4. We conclude with an outlook on future directions in Section 5.

2. Search Space

The search space defines which neural architectures a NAS approach might discover in principle. We now discuss common search spaces from recent works.

A relatively simple search space is the space of *chain-structured neural networks*, as illustrated in Figure 2 (left). A chain-structured neural network architecture A can be written as a sequence of n layers, where the i'th layer L_i receives its input from layer $i-1$ and

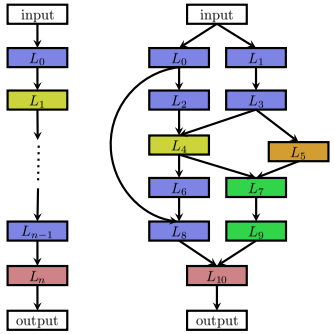


Figure 2: An illustration of different architecture spaces. Each node in the graphs corresponds to a layer in a neural network, e.g., a convolutional or pooling layer. Different layer types are visualized by different colors. An edge from layer L_i to layer L_j denotes that L_j receives the output of L_i as input. Left: an element of a chain-structured space. Right: an element of a more complex search space with additional layer types and multiple branches and skip connections.

its output serves as the input for layer $i+1$, i.e., $\mathsf{A} = L_n \circ \dots L_1 \circ L_0$. The search space is then parametrized by: (i) the (maximum) number of layers n (possibly unbounded); (ii) the type of operation every layer executes, e.g., pooling, convolution, or more advanced operations like depthwise separable convolutions (Chollet, 2016) or dilated convolutions (Yu and Koltun, 2016); and (iii) hyperparameters associated with the operation, e.g., number of filters, kernel size and strides for a convolutional layer (Baker et al., 2017a; Suganuma et al., 2017; Cai et al., 2018a), or simply number of units for fully-connected networks (Mendoza et al., 2016). Note that the parameters from (iii) are conditioned on (ii), hence the parametrization of the search space is not fixed-length but rather a conditional space.

Recent work on NAS (Brock et al., 2017; Elsken et al., 2017; Zoph et al., 2018; Elsken et al., 2019; Real et al., 2019; Cai et al., 2018b) incorporates modern design elements known from hand-crafted architectures, such as skip connections, which allow to build complex, *multi-branch networks*, as illustrated in Figure 2 (right). In this case the input of layer i can be formally described as a function $g_i(L_{i-1}^{out}, \dots, L_0^{out})$ combining previous layer outputs. Employing such a function results in significantly more degrees of freedom. Special cases of these multi-branch architectures are (i) the chain-structured networks (by setting $g_i(L_{i-1}^{out}, \dots, L_0^{out}) = L_{i-1}^{out}$), (ii) Residual Networks (He et al., 2016), where previous layer outputs are summed ($g_i(L_{i-1}^{out}, \dots, L_0^{out}) = L_{i-1}^{out} + L_j^{out}, j < i-1$) and (iii) DenseNets (Huang et al., 2017), where previous layer outputs are concatenated ($g_i(L_{i-1}^{out}, \dots, L_0^{out}) = concat(L_{i-1}^{out}, \dots, L_0^{out})$).

Motivated by hand-crafted architectures consisting of repeated motifs (Szegedy et al., 2016; He et al., 2016; Huang et al., 2017), Zoph et al. (2018) and Zhong et al. (2018a) propose to search for such motifs, dubbed *cells* or *blocks*, respectively, rather than for whole

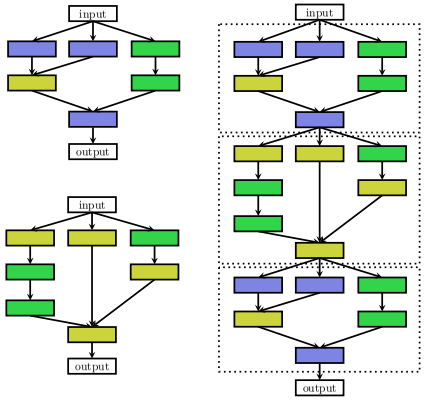


Figure 3: Illustration of the cell search space. Left: Two different cells, e.g., a normal cell (top) and a reduction cell (bottom) (Zoph et al., 2018). Right: an architecture built by stacking the cells sequentially. Note that cells can also be combined in a more complex manner, such as in multi-branch spaces, by simply replacing layers with cells.

architectures. Zoph et al. (2018) optimize two different kind of cells: a *normal cell* that preserves the dimensionality of the input and a *reduction cell* which reduces the spatial dimension. The final architecture is then built by stacking these cells in a predefined manner, as illustrated in Figure 3. This search space has three major advantages compared to the ones discussed above:

1. The size of the search space is drastically reduced since cells usually consist of significantly less layers than whole architectures. For example, Zoph et al. (2018) estimate a seven-times speed-up compared to their previous work (Zoph and Le, 2017) while achieving better performance.

2. Architectures built from cells can more easily be transferred or adapted to other data sets by simply varying the number of cells and filters used within a model. Indeed, Zoph et al. (2018) transfer cells optimized on CIFAR-10 to ImageNet and achieve state-of-the-art performance.

3. Creating architectures by repeating building blocks has proven a useful design principle in general, such as repeating an LSTM block in RNNs or stacking a residual block.

Consequently, this cell-based search space is also successfully employed by many recent works (Real et al., 2019; Liu et al., 2018a; Pham et al., 2018; Elsken et al., 2019; Cai et al., 2018b; Liu et al., 2019b; Zhong et al., 2018b). However, a new design-choice arises when

using a cell-based search space, namely how to choose the *macro-architecture*: how many cells shall be used and how should they be connected to build the actual model? For example, Zoph et al. (2018) build a sequential model from cells, in which each cell receives the outputs of the two preceding cells as input, while Cai et al. (2018b) employ the high-level structure of well-known manually designed architectures, such as DenseNet (Huang et al., 2017), and use their cells within these models. In principle, cells can be combined arbitrarily, e.g., within the multi-branch space described above, by simply replacing layers with cells. Ideally, both the macro-architecture and the micro-architecture (i.e., the structure of the cells) should be optimized jointly instead of solely optimizing the micro-architecture; otherwise, one may easily end up having to do manual macro-architecture engineering after finding a well-performing cell. One step in the direction of optimizing macro-architectures is the hierarchical search space introduced by Liu et al. (2018b), which consists of several levels of motifs. The first level consists of the set of primitive operations, the second level of different motifs that connect primitive operations via a directed acyclic graph, the third level of motifs that encode how to connect second-level motifs, and so on. The cell-based search space can be seen as a special case of this hierarchical search space where the number of levels is three, the second level motifs correspond to the cells, and the third level is the hard-coded macro-architecture.

The choice of the search space largely determines the difficulty of the optimization problem: even for the case of the search space based on a single cell with fixed macro-architecture, the optimization problem remains (i) non-continuous and (ii) relatively high-dimensional (since more complex models tend to perform better, resulting in more design choices).

We note that the architectures in many search spaces can be written as fixed-length vectors; e.g., the search space for each of the two cells by Zoph et al. (2018) can be written as a 40-dimensional[1] search space with categorical dimensions, each of which chooses between a small number of different building blocks and inputs. Unbounded search spaces can be constrained to have a (potentially very large, but finite) number of layers, which again gives rise to fixed-size search spaces with (potentially many) conditional dimensions.

In the next section, we discuss Search Strategies that are well-suited for these kinds of search spaces.

3. Search Strategy

Many different search strategies can be used to explore the space of neural architectures, including random search, Bayesian optimization, evolutionary methods, reinforcement learning (RL), and gradient-based methods. Historically, evolutionary algorithms were already used by many researchers to evolve neural architectures (and often also their weights) decades ago (see, e.g., Angeline et al., 1994; Stanley and Miikkulainen, 2002; Floreano et al., 2008; Stanley et al., 2009; Jozefowicz et al., 2015). Yao (1999) provides a literature review of work earlier than 2000.

1. Each of the 2 cells consists of 5 blocks. For each block, 2 inputs are chosen and for each of these inputs an operation to be applied to the input is chosen, resulting in a $2 \cdot 5 \cdot 2 \cdot 2 = 40$ dimensional representation. Note that originally there was another dimension for either summing or concatenating the operations within a block. However, this choice was discarded in the experiments of the paper and the outputs within a block were always summed.

Bayesian optimization celebrated several early successes in NAS since 2013, leading to state-of-the-art vision architectures (Bergstra et al., 2013), state-of-the-art performance for CIFAR-10 without data augmentation (Domhan et al., 2015), and the first automatically-tuned neural networks to win on competition data sets against human experts (Mendoza et al., 2016). NAS became a mainstream research topic in the machine learning community after Zoph and Le (2017) obtained competitive performance on the CIFAR-10 and Penn Treebank benchmarks with a search strategy based on reinforcement learning. While Zoph and Le (2017) used vast computational resources to achieve this result (800 GPUs for three to four weeks), after their work, a wide variety of methods have been published in quick succession to reduce the computational costs and achieve further improvements in performance.

To frame NAS as a *reinforcement learning* (RL) problem (Baker et al., 2017a; Zoph and Le, 2017; Zhong et al., 2018a; Zoph et al., 2018), the generation of a neural architecture can be considered to be the agent's action, with the action space identical to the search space. The agent's reward is based on an estimate of the performance of the trained architecture on unseen data (see Section 4). Different RL approaches differ in how they represent the agent's policy and how they optimize it: Zoph and Le (2017) use a recurrent neural network (RNN) policy to sequentially sample a string that in turn encodes the neural architecture. They initially trained this network with the REINFORCE policy gradient algorithm (Williams, 1992), but in their follow-up work (Zoph et al., 2018) use Proximal Policy Optimization (Schulman et al., 2017) instead. Baker et al. (2017a) use Q-learning to train a policy which sequentially chooses a layer's type and corresponding hyperparameters.

An alternative view of these approaches is as sequential decision processes in which the policy samples actions to generate the architecture sequentially, the environment's "state" contains a summary of the actions sampled so far, and the (undiscounted) reward is obtained only after the final action. However, since no interaction with an environment occurs during this sequential process (no external state is observed, and there are no intermediate rewards), we find it more intuitive to interpret the architecture sampling process as the sequential generation of a single action; this simplifies the RL problem to a stateless multi-armed bandit problem.

A related approach was proposed by Cai et al. (2018a), who frame NAS as a sequential decision process: in their approach the state is the current (partially trained) architecture, the reward is an estimate of the architecture's performance, and the action corresponds to an application of function-preserving mutations, dubbed network morphisms (Chen et al., 2016; Wei et al., 2017), see also Section 4, followed by a training phase of the network. In order to deal with variable-length network architectures, they use a bi-directional LSTM to encode architectures into a fixed-length representation. Based on this encoded representation, actor networks decide on the sampled action. The combination of these two components constitute the policy, which is trained end-to-end with the REINFORCE policy gradient algorithm. We note that this approach will not visit the same state (architecture) twice.

An alternative to using RL are *neuro-evolutionary* approaches that use evolutionary algorithms for optimizing the neural architecture. The first such approach for designing neural networks we are aware of dates back almost three decades: Miller et al. (1989)

use genetic algorithms to propose architectures and use backpropagation to optimize their weights. Many neuro-evolutionary approaches since then (Angeline et al., 1994; Stanley and Miikkulainen, 2002; Stanley et al., 2009) use genetic algorithms to optimize both the neural architecture and its weights; however, when scaling to contemporary neural architectures with millions of weights for supervised learning tasks, SGD-based weight optimization methods currently outperform evolutionary ones.[2] More recent neuro-evolutionary approaches (Real et al., 2017; Suganuma et al., 2017; Liu et al., 2018b; Real et al., 2019; Miikkulainen et al., 2017; Xie and Yuille, 2017; Elsken et al., 2019) therefore again use gradient-based methods for optimizing weights and solely use evolutionary algorithms for optimizing the neural architecture itself. Evolutionary algorithms evolve a population of models, i.e., a set of (possibly trained) networks; in every evolution step, at least one model from the population is sampled and serves as a parent to generate offsprings by applying mutations to it. In the context of NAS, mutations are local operations, such as adding or removing a layer, altering the hyperparameters of a layer, adding skip connections, as well as altering training hyperparameters. After training the offsprings, their fitness (e.g., performance on a validation set) is evaluated and they are added to the population.

Neuro-evolutionary methods differ in how they sample parents, update populations, and generate offsprings. For example, Real et al. (2017), Real et al. (2019), and Liu et al. (2018b) use tournament selection (Goldberg and Deb, 1991) to sample parents, whereas Elsken et al. (2019) sample parents from a multi-objective Pareto front using an inverse density. Real et al. (2017) remove the worst individual from a population, while Real et al. (2019) found it beneficial to remove the oldest individual (which decreases greediness), and Liu et al. (2018b) do not remove individuals at all. To generate offspring, most approaches initialize child networks randomly, while Elsken et al. (2019) employ Lamarckian inheritance, i.e, knowledge (in the form of learned weights) is passed on from a parent network to its children by using network morphisms. Real et al. (2017) also let an offspring inherit all parameters of its parent that are not affected by the applied mutation; while this inheritance is not strictly function-preserving it might also speed up learning compared to a random initialization. Moreover, they also allow mutating the learning rate which can be seen as a way for optimizing the learning rate schedule during NAS. We refer to Stanley et al. (2019) for a recent in-depth review on neuro-evolutionary methods.

Real et al. (2019) conduct a case study comparing RL, evolution, and random search (RS), concluding that RL and evolution perform equally well in terms of final test accuracy, with evolution having better anytime performance and finding smaller models. Both approaches consistently perform better than RS in their experiments, but with a rather small margin: RS achieved test errors of approximately 4% on CIFAR-10, while RL and evolution reached approximately 3.5% (after "model augmentation" where depth and number of filters was increased; the difference on the non-augmented space actually used for the search was approx. 2%). The difference was even smaller for Liu et al. (2018b), who reported a test error of 3.9% on CIFAR-10 and a top-1 validation error of 21.0% on ImageNet for RS, compared to 3.75% and 20.3% for their evolution-based method, respectively.

2. Some recent work shows that evolving even millions of weights is competitive to gradient-based optimization when only high-variance estimates of the gradient are available, e.g., for reinforcement learning tasks (Salimans et al., 2017; Such et al., 2017; Chrabaszcz et al., 2018). Nonetheless, for supervised learning tasks gradient-based optimization is by far the most common approach.

Bayesian Optimization (BO, see, e.g., (Shahriari et al., 2016)) is one of the most popular methods for hyperparameter optimization, but it has not been applied to NAS by many groups since typical BO toolboxes are based on Gaussian processes and focus on low-dimensional continuous optimization problems. Swersky et al. (2013) and Kandasamy et al. (2018) derive kernel functions for architecture search spaces in order to use classic GP-based BO methods. In contrast, several works use tree-based models (in particular, tree Parzen estimators (Bergstra et al., 2011), or random forests (Hutter et al., 2011)) to effectively search high-dimensional conditional spaces and achieve state-of-the-art performance on a wide range of problems, optimizing both neural architectures and their hyperparameters jointly (Bergstra et al., 2013; Domhan et al., 2015; Mendoza et al., 2016; Zela et al., 2018). While a full comparison is lacking, there is preliminary evidence that these approaches can also outperform evolutionary algorithms (Klein et al., 2018).

Negrinho and Gordon (2017) and Wistuba (2017) exploit the tree-structure of their search space and use *Monte Carlo Tree Search*. Elsken et al. (2017) propose a simple yet well performing *hill climbing* algorithm that discovers high-quality architectures by greedily moving in the direction of better performing architectures without requiring more sophisticated exploration mechanisms.

While the methods above employ a discrete search space, Liu et al. (2019b) propose a *continuous relaxation* to enable direct gradient-based optimization: instead of fixing a single operation o_i (e.g., convolution or pooling) to be executed at a specific layer, the authors compute a convex combination from a set of operations $\{o_1, \dots, o_m\}$. More specifically, given a layer input x, the layer output y is computed as $y = \sum_{i=1}^{m} \alpha_i o_i(x), \alpha_i \geq 0, \sum_{i=1}^{m} \alpha_i = 1$, where the convex coefficients α_i effectively parametrize the network architecture. Liu et al. (2019b) then optimize both the network weights and the network architecture by alternating gradient descent steps on training data for weights and on validation data for architectural parameters such as α. Eventually, a discrete architecture is obtained by choosing the operation i^* with $i^* = \arg\max_i \alpha_i$ for every layer. Instead of optimizing a weighting α of possible operations, Xie et al. (2019); Cai et al. (2019) propose to optimize a parametrized distribution over the possible operations. Shin et al. (2018) and Ahmed and Torresani (2018) also employ gradient-based optimization of neural architectures, however focusing on optimizing layer hyperparameters or connectivity patterns, respectively.

4. Performance Estimation Strategy

The search strategies discussed in Section 3 aim at finding a neural architecture A that maximizes some performance measure, such as accuracy on unseen data. To guide their search process, these strategies need to estimate the performance of a given architecture A they consider. The simplest way of doing this is to train A on training data and evaluate its performance on validation data. However, training each architecture to be evaluated from scratch frequently yields computational demands in the order of thousands of GPU days for NAS (Zoph and Le, 2017; Real et al., 2017; Zoph et al., 2018; Real et al., 2019). This naturally leads to developing methods for speeding up performance estimation, which we will now discuss. We refer to Table 4 for an overview of existing methods.

Performance can be estimated based on *lower fidelities* of the actual performance after full training (also denoted as proxy metrics). Such lower fidelities include shorter training

Speed-up method	How are speed-ups achieved?	References
Lower fidelity estimates	Training time reduced by training for fewer epochs, on subset of data, downscaled models, downscaled data, ...	Li et al. (2017), Zoph et al. (2018), Zela et al. (2018), Falkner et al. (2018), Real et al. (2019), Runge et al. (2019)
Learning Curve Extrapolation	Training time reduced as performance can be extrapolated after just a few epochs of training.	Swersky et al. (2014), Domhan et al. (2015), Klein et al. (2017a), Baker et al. (2017b)
Weight Inheritance/ Network Morphisms	Instead of training models from scratch, they are warm-started by inheriting weights of, e.g., a parent model.	Real et al. (2017), Elsken et al. (2017), Elsken et al. (2019), Cai et al. (2018a,b)
One-Shot Models/ Weight Sharing	Only the one-shot model needs to be trained; its weights are then shared across different architectures that are just subgraphs of the one-shot model.	Saxena and Verbeek (2016), Pham et al. (2018), Bender et al. (2018), Liu et al. (2019b), Cai et al. (2019), Xie et al. (2019)

Table 1: Overview of different methods for speeding up performance estimation in NAS.

times (Zoph et al., 2018; Zela et al., 2018), training on a subset of the data (Klein et al., 2017b), on lower-resolution images (Chrabaszcz et al., 2017), or with less filters per layer and less cells (Zoph et al., 2018; Real et al., 2019). While these low-fidelity approximations reduce the computational cost, they also introduce bias in the estimate as performance will typically be underestimated. This may not be problematic as long as the search strategy only relies on ranking different architectures and the relative ranking remains stable. However, recent results indicate that this relative ranking can change dramatically when the difference between the cheap approximations and the "full" evaluation is too big (Zela et al., 2018), arguing for a gradual increase in fidelities (Li et al., 2017; Falkner et al., 2018).

Another possible way of estimating an architecture's performance builds upon learning curve extrapolation (Swersky et al., 2014; Domhan et al., 2015; Klein et al., 2017a; Baker et al., 2017b; Rawal and Miikkulainen, 2018). Domhan et al. (2015) propose to extrapolate initial learning curves and terminate those predicted to perform poorly to speed up the architecture search process. Swersky et al. (2014), Klein et al. (2017a), Baker et al. (2017b), and Rawal and Miikkulainen (2018) also consider architectural hyperparameters for predicting which partial learning curves are most promising. Training a surrogate model for predicting the performance of novel architectures is also proposed by Liu et al. (2018a), who do not employ learning curve extrapolation but support predicting performance based on architectural/cell properties and extrapolate to architectures/cells with larger size than seen during training. The main challenge for predicting the performances of neural architectures is that, in order to speed up the search process, good predictions in a relatively large search space need to be made based on relatively few evaluations.

Another approach to speed up performance estimation is to initialize the weights of novel architectures based on weights of other architectures that have been trained before. One way of achieving this, dubbed *network morphisms* (Wei et al., 2016), allows modifying an architecture while leaving the function represented by the network unchanged, resulting in methods that only require a few GPU days (Elsken et al., 2017; Cai et al., 2018a,b; Jin et al., 2018). This allows increasing the capacity of networks successively and retaining high performance without requiring training from scratch. Continuing training for a few epochs can also make use of the additional capacity introduced by network morphisms. An advantage of these approaches is that they allow search spaces without an inherent upper bound on the architecture's size (Elsken et al., 2017); on the other hand, strict network morphisms can only make architectures larger and may thus lead to overly complex architectures. This can be attenuated by employing approximate network morphisms that allow shrinking architectures (Elsken et al., 2019).

One-Shot Architecture Search (see Figure 4) treats all architectures as different subgraphs of a supergraph (the one-shot model) and shares weights between architectures that have edges of this supergraph in common (Saxena and Verbeek, 2016; Brock et al., 2017; Pham et al., 2018; Liu et al., 2019b; Bender et al., 2018; Cai et al., 2019; Xie et al., 2019). Only the weights of a single one-shot model need to be trained (in one of various ways), and architectures (which are just subgraphs of the one-shot model) can then be evaluated without any separate training by inheriting trained weights from the one-shot model. This greatly speeds up performance estimation of architectures, since no training is required (only evaluating performance on validation data), again resulting in methods that only require a few GPU days. The one-shot model typically incurs a large bias as it underestimates the actual performance of the best architectures severely; nevertheless, it allows ranking architectures, which would be sufficient if the estimated performance correlates strongly with the actual performance. However, it is currently not clear if this is actually the case (Bender et al., 2018; Sciuto et al., 2019).

Different one-shot NAS methods differ in how the one-shot model is trained: ENAS (Pham et al., 2018) learns a RNN controller that samples architectures from the search space and trains the one-shot model based on approximate gradients obtained through REINFORCE. DARTS (Liu et al., 2019b) optimizes all weights of the one-shot model jointly with a continuous relaxation of the search space, obtained by placing a mixture of candidate operations on each edge of the one-shot model. Instead of optimizing real-valued weights on the operations as in DARTS, SNAS (Xie et al., 2019) optimizes a distribution over the candidate operations. The authors employ the concrete distribution (Maddison et al., 2017; Jang et al., 2017) and reparametrization (Kingma and Welling, 2014) to relax the discrete distribution and make it differentiable, enabling optimization via gradient descent. To overcome the necessity of keeping the entire one-shot model in the GPU memory, ProxylessNAS (Cai et al., 2019) "binarizes" the architectural weights, masking out all but one edge per operation. The probabilities for an edge being either masked out or not are then learned by sampling a few binarized architectures and using BinaryConnect (Courbariaux et al., 2015) to update the corresponding probabilities. Bender et al. (2018) only train the one-shot model once and show that this is sufficient when deactivating parts of this model stochastically during training using path dropout.

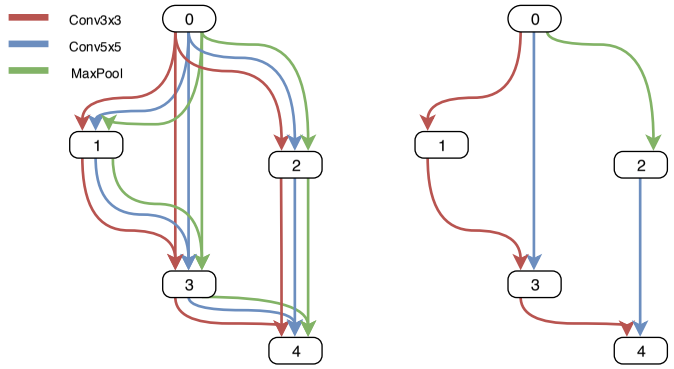


Figure 4: Illustration of one-shot architecture search. Simple network with an input node (denoted as 0), three hidden nodes (denoted as 1,2,3) and one output node (denoted as 4). Instead of applying a single operation (such as a 3x3 convolution) to a node, the one-shot model (left) contains several candidate operations for every node, namely 3x3 convolution (red edges), 5x5 convolution (blue edges) and MaxPooling (green edges) in the above illustration. Once the one-shot model is trained, its weights are shared across different architectures, which are simply subgraphs of the one-shot model (right). Figure inspired by Liu et al. (2019b).

While the previously mentioned approaches optimize a distribution over architectures during training, the approach of Bender et al. (2018) can be seen as using a fixed distribution. The high performance obtainable by the latter indicates that the combination of weight sharing and a fixed (carefully chosen) distribution might (perhaps surprisingly) be the only required ingredients for one-shot NAS. Related to these approaches is meta-learning of hypernetworks that generate weights for novel architectures and thus requires only training the hypernetwork but not the architectures themselves (Brock et al., 2017; Zhang et al., 2019). The main difference here is that weights are not strictly shared but generated by the shared hypernetwork (conditional on the sampled architecture).

A general limitation of one-shot NAS is that the supergraph defined a priori restricts the search space to its subgraphs. Moreover, approaches which require that the entire supergraph resides in GPU memory during architecture search will be restricted to relatively small supergraphs and search spaces accordingly, and are thus typically used in combination with cell-based search spaces. While approaches based on weight sharing have substantially reduced the computational resources required for NAS (from thousands to a few GPU days), it is currently not well understood which biases they introduce into the search if the sampling distribution of architectures is optimized along with the one-shot model instead of fixing it (Bender et al., 2018). For instance, an initial bias in exploring certain parts of the search space more than others might lead to the weights of the one-shot model being better adapted for these architectures, which in turn would reinforce the bias of the search to these parts of the search space. This might result in premature convergence of NAS or little correlation between the one-shot and true performance of an architecture (Sciuto et al., 2019). In general, a more systematic analysis of biases introduced by different performance estimators would be a desirable direction for future work.

5. Future Directions

In this section, we discuss several current and future directions for research on NAS. Most existing work has focused on NAS for image classification. On the one hand, this provides a challenging benchmark since a lot of manual engineering has been devoted to finding architectures that perform well in this domain and are not easily outperformed by NAS. On the other hand, it is relatively easy to define a well-suited search space by exploiting knowledge from manual engineering. This in turn makes it unlikely that NAS will find architectures that substantially outperform existing ones considerably since the found architectures cannot differ fundamentally. We thus consider it important to go beyond image classification problems by applying NAS to less explored domains. Notable first steps in this direction have been taken in image restoration (Suganuma et al., 2018), semantic segmentation (Chen et al., 2018; Nekrasov et al., 2018; Liu et al., 2019a), transfer learning (Wong et al., 2018), machine translation (So et al., 2019), reinforcement learning (Runge et al., 2019)[3], as well as optimizing recurrent neural networks (Greff et al., 2015; Jozefowicz et al., 2015; Zoph and Le, 2017; Rawal and Miikkulainen, 2018), e.g., for language or music modeling. Further promising application areas for NAS would be generative adversarial networks or sensor fusion.

An additional promising direction is to develop NAS methods for multi-task problems (Liang et al., 2018; Meyerson and Miikkulainen, 2018) and for multi-objective problems (Elsken et al., 2019; Dong et al., 2018; Zhou et al., 2018), in which measures of resource efficiency are used as objectives along with the predictive performance on unseen data. We highlight that multi-objective NAS is closely related to network compression (Han et al., 2016; Cheng et al., 2018): both aim at finding well-performing but efficient architectures. Hence, some compression methods can also be seen as NAS methods (Han et al., 2015; Liu et al., 2017; Gordon et al., 2018; Liu et al., 2019c; Cao et al., 2019) and vice versa (Saxena and Verbeek, 2016; Liu et al., 2019b; Xie et al., 2019).

Likewise, it would be interesting to extend RL/bandit approaches, such as those discussed in Section 3, to learn policies that are conditioned on a state that encodes task properties/resource requirements (i.e., turning the setting into a contextual bandit). A similar direction was followed by Ramachandran and Le (2018) in extending one-shot NAS to generate different architectures depending on the task or instance on-the-fly. Moreover, applying NAS to searching for architectures that are more robust to adversarial examples (Cubuk et al., 2017) is an intriguing recent direction.

Related to this is research on defining more general and flexible search spaces. For instance, while the cell-based search space provides high transferability between different image classification tasks, it is largely based on human experience on image classification and does not generalize easily to other domains where the hard-coded hierarchical structure (repeating the same cells several times in a chain-like structure) does not apply (e.g.,

3. Many authors have optimized some architectural choices of deep reinforcement learning algorithms before. Runge et al. (2019) used the largest and most versatile space of policy network architectures so far (e.g., including both convolutional and recurrent building blocks), but a full study of NAS for RL is yet missing.

semantic segmentation or object detection). A search space which allows representing and identifying more general hierarchical structure would thus make NAS more broadly applicable, see Liu et al. (2018b, 2019a) for first work in this direction. Moreover, common search spaces are also based on predefined building blocks, such as different kinds of convolutions and pooling, but do not allow identifying novel building blocks on this level; going beyond this limitation might substantially increase the power of NAS.

The comparison of different methods for NAS and even the reproducibility of published results[4] is complicated by the fact that measurements of an architecture's performance depend on many factors other than the architecture itself. While most authors report results on the CIFAR-10 data set, experiments often differ with regard to search space, computational budget, data augmentation, training procedures, regularization, and other factors. For example, for CIFAR-10, performance substantially improves when using a cosine annealing learning rate schedule (Loshchilov and Hutter, 2017), data augmentation by CutOut (Devries and Taylor, 2017), by MixUp (Zhang et al., 2017) or by a combination of factors (Cubuk et al., 2018), and regularization by Shake-Shake regularization (Gastaldi, 2017) or ScheduledDropPath (Zoph et al., 2018). It is therefore conceivable that improvements in these ingredients have a larger impact on reported performance numbers than the better architectures found by NAS. We thus consider the definition of common benchmarks to be crucial for a fair comparison of different NAS methods. A first step in this direction is the benchmark proposed by Ying et al. (2019), where a search space consisting of approximately 423,000 unique convolutional architectures is considered. Each element of this space was pre-trained and evaluated multiple times, resulting in a data set containing training, validation and test accuracies as well as training times and model sizes for different training budgets for multiple runs. Different search strategies can hence be compared with low computational resources on this benchmark by simply querying the pre-computed data set. In a smaller previous study, Klein et al. (2018) pre-evaluated the joint space of neural architectures and hyperparameters. It would also be interesting to evaluate NAS methods not in isolation but as part of a full open-source AutoML system, where also hyperparameters (Mendoza et al., 2016; Real et al., 2017; Zela et al., 2018), and data augmentation pipeline (Cubuk et al., 2018) are optimized along with NAS.

While NAS has achieved impressive performance, so far it provides little insights into why specific architectures work well and how similar the architectures derived in independent runs would be. Identifying common motifs, providing an understanding why those motifs are important for high performance, and investigating if these motifs generalize over different problems would be desirable.

Acknowledgments

We would like to thank Arber Zela, Esteban Real, Gabriel Bender, Kenneth Stanley, Thomas Pfeil and the anonymous reviewers for feedback on this survey. This work has partly been supported by the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme under grant no. 716721.

4. We refer to Li and Talwalkar (2019) for a detailed discussion on reproducibility for NAS.

References

Karim Ahmed and Lorenzo Torresani. Maskconnect: Connectivity learning by gradient descent. In *European Conference on Computer Vision (ECCV)*, 2018.

Peter J. Angeline, Gregory M. Saunders, and Jordan B. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE transactions on neural networks*, 5 1: 54–65, 1994.

Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. In *International Conference on Learning Representations*, 2017a.

Bowen Baker, Otkrist Gupta, Ramesh Raskar, and Nikhil Naik. Accelerating Neural Architecture Search using Performance Prediction. In *NIPS Workshop on Meta-Learning*, 2017b.

Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *International Conference on Machine Learning*, 2018.

James Bergstra, Dan Yamins, and David D. Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *ICML*, 2013.

James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2546–2554, 2011.

Andrew Brock, Theodore Lim, James M. Ritchie, and Nick Weston. SMASH: one-shot model architecture search through hypernetworks. In *NIPS Workshop on Meta-Learning*, 2017.

Han Cai, Tianyao Chen, Weinan Zhang, Yong Yu, and Jun Wang. Efficient architecture search by network transformation. In *Association for the Advancement of Artificial Intelligence*, 2018a.

Han Cai, Jiacheng Yang, Weinan Zhang, Song Han, and Yong Yu. Path-Level Network Transformation for Efficient Architecture Search. In *International Conference on Machine Learning*, June 2018b.

Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *International Conference on Learning Representations*, 2019.

Shengcao Cao, Xiaofang Wang, and Kris M. Kitani. Learnable embedding space for efficient neural architecture compression. In *International Conference on Learning Representations*, 2019.

Liang-Chieh Chen, Maxwell Collins, Yukun Zhu, George Papandreou, Barret Zoph, Florian Schroff, Hartwig Adam, and Jon Shlens. Searching for efficient multi-scale architectures for dense image prediction. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8713–8724. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/8087-searching-for-efficient-multi-scale-architectures-for-dense-image-prediction.pdf.

Tianqi Chen, Ian J. Goodfellow, and Jonathon Shlens. Net2net: Accelerating learning via knowledge transfer. In *International Conference on Learning Representations*, 2016.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Process. Mag.*, 35(1):126–136, 2018.

François Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv:1610.02357*, 2016.

Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the CIFAR datasets. *CoRR*, abs/1707.08819, 2017.

Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. Back to basics: Benchmarking canonical evolution strategies for playing atari. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 1419–1426. International Joint Conferences on Artificial Intelligence Organization, July 2018.

Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3123–3131. Curran Associates, Inc., 2015.

Ekin D. Cubuk, Barret Zoph, Samuel S. Schoenholz, and Quoc V. Le. Intriguing Properties of Adversarial Examples. In *arXiv:1711.02846*, November 2017.

Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. AutoAugment: Learning Augmentation Policies from Data. In *arXiv:1805.09501*, May 2018.

Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint*, abs/1708.04552, 2017.

T. Domhan, J. T. Springenberg, and F. Hutter. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

Jin-Dong Dong, An-Chieh Cheng, Da-Cheng Juan, Wei Wei, and Min Sun. Dpp-net: Device-aware progressive search for pareto-optimal neural architectures. In *European Conference on Computer Vision*, 2018.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Simple And Efficient Architecture Search for Convolutional Neural Networks. In *NIPS Workshop on Meta-Learning*, 2017.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Efficient multi-objective neural architecture search via lamarckian evolution. In *International Conference on Learning Representations*, 2019.

Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1436–1445, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Matthias Feurer and Frank Hutter. Hyperparameter optimization. In Hutter et al. (2019), pages 3–38. In press, available at http://automl.org/book.

Dario Floreano, Peter Drr, and Claudio Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1):47–62, 2008.

Xavier Gastaldi. Shake-shake regularization. In *International Conference on Learning Representations Workshop*, 2017.

David E. Goldberg and Kalyanmoy Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms*, pages 69–93. Morgan Kaufmann, 1991.

Ariel Gordon, Elad Eban, Ofir Nachum, Bo Chen, Hao Wu, Tien-Ju Yang, and Edward Choi. Morphnet: Fast and simple resource-constrained structure learning of deep networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Klaus Greff, Rupesh Kumar Srivastava, Jan Koutnk, Bas R. Steunebrink, and Jrgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28, 2015.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1135–1143. Curran Associates, Inc., 2015. URL http://papers.nips.cc/paper/5784-learning-both-weights-and-connections-for-efficient-neural-network.pdf.

Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations*, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016.

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely Connected Convolutional Networks. In *Conference on Computer Vision and Pattern Recognition*, 2017.

F. Hutter, H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *LION*, pages 507–523, 2011.

Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors. *Automatic Machine Learning: Methods, Systems, Challenges*. Springer, 2019. In press, available at http://automl.org/book.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

Haifeng Jin, Qingquan Song, and Xia Hu. Auto-keras: Efficient neural architecture search with network morphism, 2018.

Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2342–2350, Lille, France, 07–09 Jul 2015. PMLR.

Kirthevasan Kandasamy, Willie Neiswanger, Jeff Schneider, Barnabas Poczos, and Eric P Xing. Neural architecture search with bayesian optimisation and optimal transport. In *Advances in Neural Information Processing Systems 31*. 2018.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

A. Klein, S. Falkner, J. T. Springenberg, and F. Hutter. Learning curve prediction with Bayesian neural networks. In *International Conference on Learning Representations*, 2017a.

Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 528–536, Fort Lauderdale, FL, USA, 20–22 Apr 2017b. PMLR.

Aaron Klein, Eric Christiansen, Kevin Murphy, and Frank Hutter. Towards reproducible neural architecture and hyperparameter search. In *ICML 2018 Workshop on Reproducibility in ML (RML 2018)*, 2018.

Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. *arXiv preprint*, 2019.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: bandit-based configuration evaluation for hyperparameter optimization. In *International Conference on Learning Representations*, 2017.

Jason Liang, Elliot Meyerson, and Risto Miikkulainen. Evolutionary Architecture Search For Deep Multitask Networks. In *arXiv:1803.03745*, March 2018.

Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive Neural Architecture Search. In *European Conference on Computer Vision*, 2018a.

Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan Yuille, and Li Fei-Fei. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. *arXiv preprint*, 2019a.

Hanxiao Liu, Karen Simonyan, Oriol Vinyals, Chrisantha Fernando, and Koray Kavukcuoglu. Hierarchical Representations for Efficient Architecture Search. In *International Conference on Learning Representations*, 2018b.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*, 2019b.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2755–2763, 2017.

Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2019c.

I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

H. Mendoza, A. Klein, M. Feurer, J. Springenberg, and F. Hutter. Towards Automatically-Tuned Neural Networks. In *International Conference on Machine Learning, AutoML Workshop*, June 2016.

Elliot Meyerson and Risto Miikkulainen. Pseudo-task Augmentation: From Deep Multitask Learning to Intratask Sharing and Back. In *arXiv:1803.03745*, March 2018.

Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Dan Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, and Babak Hodjat. Evolving Deep Neural Networks. In *arXiv:1703.00548*, March 2017.

G.F. Miller, P.M. Todd, and S.U Hedge. Designing neural networks using genetic algorithms. In *3rd International Conference on Genetic Algorithms (ICGA'89)*, 1989.

R. Negrinho and G. Gordon. DeepArchitect: Automatically Designing and Training Deep Architectures. *arXiv:1704.08792*, 2017.

Vladimir Nekrasov, Hao Chen, Chunhua Shen, and Ian D. Reid. Fast neural architecture search of compact semantic segmentation models via auxiliary cells. *arXiv preprint*, 2018.

Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *International Conference on Machine Learning*, 2018.

Prajit Ramachandran and Quoc V. Le. Dynamic Network Architectures. In *AutoML 2018 (ICML workshop)*, 2018.

Aditya Rawal and Risto Miikkulainen. From Nodes to Networks: Evolving Recurrent Neural Networks. In *arXiv:1803.04439*, March 2018.

Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Quoc V. Le, and Alex Kurakin. Large-scale evolution of image classifiers. *International Conference on Machine Learning*, 2017.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Aging Evolution for Image Classifier Architecture Search. In *AAAI*, 2019.

Frederic Runge, Danny Stoll, Stefan Falkner, and Frank Hutter. Learning to design RNA. In *International Conference on Learning Representations*, 2019.

Tim Salimans, Jonathan Ho, Xi Chen, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint*, 2017.

Shreyas Saxena and Jakob Verbeek. Convolutional neural fabrics. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4053–4061. Curran Associates, Inc., 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint*, 2017.

Christian Sciuto, Kaicheng Yu, Martin Jaggi, Claudiu Musat, and Mathieu Salzmann. Evaluating the search phase of neural architecture search. *arXiv preprint*, 2019.

B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, Jan 2016. ISSN 0018-9219. doi: 10.1109/JPROC.2015.2494218.

Richard Shin, Charles Packer, and Dawn Song. Differentiable neural network architecture search. In *International Conference on Learning Representations Workshop*, 2018.

David R. So, Chen Liang, and Quoc V. Le. The evolved transformer. *arXiv preprint*, 2019.

Kenneth Stanley, Jeff Clune, Joel Lehman, and Risto Miikkulainen. Designing neural networks through neuroevolution. volume 1, 01 2019.

Kenneth O Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10:99–127, 2002.

Kenneth O. Stanley, David B. D'Ambrosio, and Jason Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artif. Life*, 15(2):185–212, April 2009. ISSN 1064-5462. doi: 10.1162/artl.2009.15.2.15202. URL http://dx.doi.org/10.1162/artl.2009.15.2.15202.

Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. *arXiv preprint*, 2017.

Masanori Suganuma, Shinichi Shirakawa, and Tomoharu Nagao. A genetic programming approach to designing convolutional neural network architectures. In *Genetic and Evolutionary Computation Conference*, 2017.

Masanori Suganuma, Mete Ozay, and Takayuki Okatani. Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4771–4780, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Kevin Swersky, David Duvenaud, Jasper Snoek, Frank Hutter, and Michael Osborne. Raiders of the lost architecture: Kernels for bayesian optimization in conditional parameter spaces. In *NIPS Workshop on Bayesian Optimization in Theory and Practice*, 2013.

Kevin Swersky, Jasper Snoek, and Ryan Prescott Adams. Freeze-thaw bayesian optimization. 2014.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *Conference on Computer Vision and Pattern Recognition*, 2016.

Joaquin Vanschoren. Meta-learning. In Hutter et al. (2019), pages 39–68. In press, available at http://automl.org/book.

Tao Wei, Changhu Wang, Yong Rui, and Chang Wen Chen. Network morphism. In *International Conference on Machine Learning*, 2016.

Tao Wei, Changhu Wang, and Chang Wen Chen. Modularized morphing of neural networks. *arXiv:1701.03281*, 2017.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, May 1992.

Martin Wistuba. Finding Competitive Network Architectures Within a Day Using UCT. In *arXiv:1712.07420*, December 2017.

Catherine Wong, Neil Houlsby, Yifeng Lu, and Andrea Gesmundo. Transfer learning with neural automl. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8366–8375. Curran Associates, Inc., 2018.

Lingxi Xie and Alan Yuille. Genetic CNN. In *International Conference on Computer Vision*, 2017.

Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. SNAS: stochastic neural architecture search. In *International Conference on Learning Representations*, 2019.

Xin Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, Sept 1999. ISSN 0018-9219. doi: 10.1109/5.784219.

Chris Ying, Aaron Klein, Esteban Real, Eric Christiansen, Kevin Murphy, and Frank Hutter. Nas-bench-101: Towards reproducible neural architecture search. *arXiv preprint*, 2019.

Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. 2016.

Arber Zela, Aaron Klein, Stefan Falkner, and Frank Hutter. Towards automated deep learning: Efficient joint neural architecture and hyperparameter search. In *ICML 2018 Workshop on AutoML (AutoML 2018)*, 2018.

Chris Zhang, Mengye Ren, and Raquel Urtasun. Graph hypernetworks for neural architecture search. In *International Conference on Learning Representations*, 2019.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint*, abs/1710.09412, 2017.

Zhao Zhong, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Practical block-wise neural network architecture generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2423–2432, 2018a.

Zhao Zhong, Zichen Yang, Boyang Deng, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Blockqnn: Efficient block-wise neural network architecture generation. *arXiv preprint*, 2018b.

Yanqi Zhou, Siavash Ebrahimi, Sercan Ark, Haonan Yu, Hairong Liu, and Greg Diamos. Resource-efficient neural architect. In *arXiv:1806.07912*, 2018.

Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*, 2017.

Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2018.

Journal of Machine Learning Research 20 (2019) 1-23 Submitted 12/18; Published 2/19

Near Optimal Frequent Directions for Sketching Dense and Sparse Matrices

Zengfeng Huang HUANGZF@FUDAN.EDU.CN
School of Data Science
Fudan University
Shanghai, China

Editor: Kilian Weinberger

Abstract

Given a large matrix $A \in \mathbb{R}^{n\times d}$, we consider the problem of computing a sketch matrix $B \in \mathbb{R}^{\ell\times d}$ which is significantly smaller than but still well approximates A. We consider the problems in the streaming model, where the algorithm can only make one pass over the input with limited working space, and we are interested in minimizing the *covariance error* $\|A^TA - B^TB\|_2$. The popular Frequent Directions algorithm of Liberty (2013) and its variants achieve optimal space-error tradeoffs. However, whether the running time can be improved remains an unanswered question. In this paper, we almost settle the question by proving that the time complexity of this problem is equivalent to that of matrix multiplication up to lower order terms. Specifically, we provide new space-optimal algorithms with faster running times and also show that the running times of our algorithms can be improved if and only if the state-of-the-art running time of matrix multiplication can be improved significantly.

Keywords: Matrix Sketching, Frequent Directions, Streaming Algorithms

1. Introduction

For large-scale matrix computations, exact algorithms are often too slow, so a large body of work focuses on designing fast randomized approximation algorithms. Matrix sketching is a commonly used algorithmic technique for solving linear algebra problems over massive data matrices, e.g., Sarlos (2006); Clarkson and Woodruff (2013); Avron et al. (2013); Chierichetti et al. (2017). In the sketch-and-solve framework, given a large matrix, we first compute a small sketch using a lightweight algorithm, and then do more expensive computations on the sketch instead of on the original matrix. Thus, the key is how to efficiently compute sketch matrices that are small but still preserve vital properties of the input matrix.

In real-world applications, data often arrives in a streaming fashion and it is often impractical or impossible to store the entire data set in the main memory. This paper studies online streaming algorithms for maintaining matrix sketches with small *covariance errors*. In the *streaming model*, the rows of the input matrix arrive one at a time; algorithm is only allowed to make one pass over the stream with severely limited working space, and is required to maintain a sketch continuously. This problem has received lots of attention recently (Liberty, 2013; Ghashami and Phillips, 2014; Woodruff, 2014a; Ghashami et al., 2016; Wei et al., 2016).

©2019 Zengfeng Huang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-875.html.

The popular *Frequent Directions* algorithms (Liberty, 2013; Ghashami et al., 2016) achieve optimal tradeoffs between space usage and approximation error (Woodruff, 2014a), which have found lots of applications in online learning, e.g., Boutsidis et al. (2015); Karnin and Liberty (2015); Leng et al. (2015); Huang and Kasiviswanathan (2015); Luo et al. (2016); Calandriello et al. (2017), and other important problems (Song et al., 2015; Ye et al., 2016; Kim et al., 2016; Cohen et al., 2017). However, in contrast to the space complexity, it is unclear whether their running times can be improved; one might hope to get linear (in sparsity) time algorithms, which is possible for many matrix problems (see e.g., Clarkson and Woodruff (2013)). This paper is motivated by the following question:

- *Is there an input sparsity time Frequent Directions, which achieves the same optimal space-error tradeoff?*

1.1. Problem Definitions

Given a matrix $A \in \mathbb{R}^{n\times d}$, the problem is to compute a much smaller matrix $B \in \mathbb{R}^{\ell\times d}$, which has low *covariance error*, i.e., $\|A^TA - B^TB\|_2$.

Definition 1 (Covariance Sketch) *For any $0 < \alpha < 1$, and integer $0 \le k \le \mathsf{rank}(A)$, we will call B an (α, k)-*cov*-sketch of A, if the covariance error*[1]

$$\|A^TA - B^TB\|_2 \le \alpha\|A - [A]_k\|_F^2. \tag{1}$$

Here $\|\cdot\|_2$ and $\|\cdot\|_F$ are the *spectral norm* and *Frobenius norm* of matrices; $[A]_k$ is the best rank-k approximation to A. We will use $\pi_B^k(A)$ to denote the projection of A on the top-k singular vectors of B, i.e. $\pi_B^k(A) = AVV^T$, where the columns of V are the top-k right singular vectors of B.

Definition 2 (Projection Error) *The projection error of B with respect to A is defined as $\|A - \pi_B^k(A)\|_F^2$.*

Note $\pi_B^k(A)$ is a rank-k matrix, thereby the projection error is at least $\|A - [A]_k\|_F^2$. It is proved in Ghashami and Phillips (2014) that one can obtain relative projection error from small covariance error. We include a proof of the next lemma in Appendix A.1.

Lemma 3 (Covariance Error to Projection Error (Ghashami and Phillips, 2014))

$$\|A - \pi_B^k(A)\|_F^2 \le \|A - [A]_k\|_F^2 + 2k\cdot\|A^TA - B^TB\|_2.$$

Therefore, any $(\frac{\varepsilon}{2k}, k)$-cov-sketch B has projection error

$$\|A - \pi_B^k(A)\|_F^2 \le (1+\varepsilon)\|A - [A]_k\|_F^2. \tag{2}$$

We will often refer to $(\frac{\varepsilon}{2k}, k)$-cov-sketches as (ε, k)-proj-sketches.

(ε, k)-proj-sketches actually satisfy a more general property, called *Projection-Cost Preserving* property. Specially, suppose B is an (ε, k)-proj-sketch of a matrix A, then

$$\|B - BQQ^T\|_F^2 = (1\pm\varepsilon)\|A - AQQ^T\|_F^2,$$

1. for $k = 0$, we define $[A]_0 = \mathbf{0}$

for all rank-k orthonormal matrices Q (see Feldman et al. (2013); Cohen et al. (2015, 2017) for more details). One immediate application of projection-cost preserving sketches is constrained low rank approximations, which can be formulated as the following optimization problem:

$$\min_{X:\mathsf{rank}(X)\leq k, X\in\mathcal{S}} \|A - AXX^T\|_F^2,$$

where $\mathcal{S}$ is some constraint; special cases include sparse PCA, nonnegative PCA, and k-means clustering (Feldman et al., 2013; Cohen et al., 2015). Given any projection-cost preserving sketch B of A, one can solve the above optimization problem with respect to B to obtain an approximate solution, which significantly reduces the computational costs when B is much smaller than A.

Modern data matrices are often large and sparse, so we will always assume n and d are very large and $\mathsf{nnz}(A) \ll nd$, where $\mathsf{nnz}(A)$ in the number of nonzero entries in A. Moreover, we assume that each entry of A is representable by $O(\log(nd))$ bits. To simplify the analysis, we assume the entries of A are integers of magnitude at most $\mathrm{poly}(nd)$; the general case can be reduced to this, see e.g., Boutsidis et al. (2016).

1.2. Previous Results

In the row-wise update streaming model, Liberty's *Frequent Directions* (FD) algorithm (Liberty, 2013), with an improved analysis by Ghashami and Phillips (2014), maintains an (α, k)-cov-sketch $B \in \mathbb{R}^{\ell\times d}$ at any time, where $\ell = O(k + \alpha^{-1})$. The algorithm uses $O(d\ell)$ space and runs in $O(nd\ell)$ time. For sparse matrices, the running time of FD is improved to $O(\mathsf{nnz}(A)\ell\log d + \mathsf{nnz}(A)\log n + n\ell^2)$ by Ghashami et al. (2016). Setting $\alpha = \varepsilon/2k$ (or $\ell = O(k/\varepsilon)$) and by Lemma 3, B is a (ε, k)-proj-sketch. Now, B contains $O(k/\varepsilon)$ rows, and the space and running time become $O(dk/\varepsilon)$ and $O(\mathsf{nnz}(A)k\varepsilon^{-1}\cdot\log d + \mathsf{nnz}(A)\log n + nk^2\varepsilon^{-2})$ respectively. The $\log d$ factor in the leading term was removed by Luo et al. (2016). It was shown by Woodruff (Woodruff, 2014a) that the space used by FD is optimal for both covariance error and projection error. A natural question is if the running time can be improved. In particular,

- *Is there an input sparsity time algorithm, i.e., in time* $O(\mathsf{nnz}(A) + (n+d)\cdot\mathrm{poly}(k\alpha^{-1}))$, *which achieves the same guarantee as* FD?

1.3. Our Contributions

This paper almost settles the above question. Our main contributions are summarized as follows.

1. We give a new space-optimal streaming algorithm with $O(ndk) + \tilde{O}(d\alpha^{-3})$ running time to compute (α, k)-cov-sketches for dense matrices, which improves the original FD algorithm for small α. The running time is optimal up to lower order terms, provided matrix multiplication cannot be improved significantly.

2. Based on our fast algorithm for sketching dense matrices, we then give a new space-optimal streaming algorithm with $O(\mathsf{nnz}(A)k + nk^3) + \tilde{O}(d\alpha^{-3})$ running time to compute (α, k)-cov-sketches for sparse matrices. We separate the dependence of $1/\alpha$ from $\mathsf{nnz}(A)$, which improves the results of Ghashami et al. (2016) for small α. Therefore, to compute an (ε, k)-proj-sketch, our algorithm only needs $O(\mathsf{nnz}(A)k)$ time (ignoring lower order terms) as opposed to $O(\mathsf{nnz}(A)k\varepsilon^{-1}\cdot\log d)$ in Ghashami et al. (2016) (see Table 1).

3. We show that $o(\mathsf{nnz}(A)k)$ time is likely very difficult to achieve, as it will imply a breakthrough in fast matrix multiplication. Specifically, we prove that, under mild assumptions, the time complexity for computing an $(O(1), k)$-cov-sketch $B \in \mathbb{R}^{O(k)\times d}$ of A in the streaming model is equivalent to the time complexity of left multiplying A by an arbitrary matrix $C \in \mathbb{R}^{k\times n}$.

	Time	
	(α, k)-cov	(ε, k)-proj
FD (Liberty, 2013)	$O(ndk + nd/\alpha)$	$O(ndk/\varepsilon)$
FFDdense (**new**)	$O(ndk)$	$O(ndk)$
Sparse FD(Ghashami et al., 2016)	$O(\mathsf{nnz}(A)k\log d + \mathsf{nnz}(A)\log d/\alpha)$	$O(\mathsf{nnz}(A)k\log d/\varepsilon)$
FFDsparse (**new**)	$O(\mathsf{nnz}(A)k)$	$O(\mathsf{nnz}(A)k)$
Lower bounds (**new**)	$\Omega(\mathsf{nnz}(A)k)$	$\Omega(\mathsf{nnz}(A)k)$

Table 1: Running times and lower bounds for streaming (α, k)-cov-sketch and (ε, k)-proj-sketch algorithms. Lower order terms are omitted and the lower bounds are conditional.

1.4. Other Related Work

The problem of computing (α, k)-cov-sketches was also studied in the sliding window streaming model (Wei et al., 2016) and distributed models (Ghashami et al., 2014; Huang et al., 2017; Zhang et al., 2017). A closely related problem, namely *approximate PCA*, was studied in Kannan et al. (2014); Liang et al. (2014); Boutsidis et al. (2016); Zhang et al. (2015). Clarkson and Woodruff (2009) studied other streaming numerical linear algebra problems.

1.5. Matrix Preliminaries and Notations

We always use n for the number rows and d for the dimension of each row. For a d-dimensional vector x, $\|x\|$ is the ℓ_2 norm of x. We use x_i to denote the ith entry of x, and $\mathsf{Diag}(x) \in \mathbb{R}^{d\times d}$ is a diagonal matrix such that the ith diagonal entry is x_i. For a matrix $A \in \mathbb{R}^{n\times d}$ with $n > d$, we use A_i to denote the ith row of A and $a_{i,j}$ for the (i,j)-th entry of A. $\mathsf{nnz}(A)$ is the number of non-zero entries in A and $\mathsf{rows}(A)$ is the number of rows in A. We write the (reduced) singular value decomposition of A as $(U, \Sigma, V) = \mathsf{SVD}(A)$. The computation time of standard SVD algorithms is $O(nd^2)$. We use $\|A\|_2$ or $\|A\|$ to denote the spectral norm of A, which is the largest singular value of A, and $\|A\|_F$ for the *Frobenius Norm*, which is $\sqrt{\sum_{i,j} a_{i,j}^2}$. For $k \leq \mathsf{rank}(A)$, we use $[A]_k$ to denote the best rank k approximation of A. We define $[A]_0 = \mathbf{0}$. $[A; B]$ is the matrix formed by concatenating the rows of A and B. We use $\tilde{O}()$ to hide $\mathrm{polylog}(ndk)$ factors.

1.6. Tools

Frequent Directions. We will use the Frequent Directions (FD) algorithm by Liberty (Liberty, 2013), denoted as $\mathsf{FD}(A, \alpha, k)$; and the main result is summarized in the following theorem.

Theorem 4 (Liberty (2013)) *Given $A \in \mathbb{R}^{n\times d}$, in one pass, $\mathsf{FD}(A, \alpha, k)$ processes A in $O(nd(k + \alpha^{-1}))$ time and $O(d(k + \alpha^{-1}))$ space. It maintains a matrix $B \in \mathbb{R}^{O(k+\alpha^{-1})\times d}$ such that $\|A^TA - B^TB\|_2 \le \alpha\|A - [A]_k\|_F^2$.*

Row sampling. We provide a result about row sampling, which is analogous to a result from Drineas et al. (2006). The difference is that they use *sampling with replacement*, i.e., each row of B is an iid sample from the rows of A. On the other hand, we use *Bernoulli sampling*, i.e., sample each A_i independently with some probability q_i, and B is the set of sampled rows. *Bernoulli sampling* can easily be combined with FD in the streaming model. The proof is essentially the same as that for sampling with replacement, which can be found in Appendix.

Theorem 5 *For any $A \in \mathbb{R}^{n\times d}$ and $F > 0$, we sample each row A_i with probability $p_i \ge \frac{\|A_i\|^2}{\alpha^2 F}$; if it is sampled, scale it by $1/\sqrt{p_i}$. Let B be the (rescaled) sampled rows, then w.p. 0.99, $\|A^TA - B^TB\|_2 \le 10\alpha\sqrt{F}\|A\|_F$ and $\|B\|_F \le 10\|A\|_F$. The expected number of rows sampled is $O(\frac{\|A\|_F^2}{\alpha^2 F})$ if $p_i = O(\frac{\|A_i\|^2}{\alpha^2 F})$ for each i.*

Input-sparsity time subspace embedding. We will use fast subspace embeddings (Clarkson and Woodruff, 2013) to speedup computation. There are various approaches to achieve input-sparsity time subspace embedding with different guarantees; the following result suffices for our purpose, the proof of which can be found in Woodruff (2014b) (Lemma 4.2 and Remark 4.1).

Theorem 6 (Subspace Embedding) *Given any $A \in \mathbb{R}^{n\times d}$, there exist (random) matrices $J \in \mathbb{R}^{O(k)\times t_1}$ and $C \in \mathbb{R}^{t_1\times d}$, with $t_1 = \min(d, O(k^2))$, such that, with probability 0.99, the column space of $S = AC^TJ^T$ contains an $O(1)$-approximate rank-k approximation to A. More precisely, there exists X with $\mathsf{rank}(X) \le k$ such that*

$$\|A - SX\|_F^2 \le O(1)\|A - [A]_k\|_F^2,$$

Moreover, C only contains $O(d)$ nonzero entries and $S = AC^TJ^T$ can be computed in time $O(\mathsf{nnz}(A)) + O(nk^3)$.

One can choose J to be a matrix with iid Gaussian random variables and C to be a count-sketch matrix (Clarkson and Woodruff, 2013)

2. Algorithm for Dense Matrices

Theorem 7 ($\mathsf{FFDdense}(A, \alpha, k)$) *Given a matrix $A \in \mathbb{R}^{n\times d}$, $0 < \alpha < 1$ and $0 \le k \le d$, $\mathsf{FFDdense}(A, \alpha, k)$ processes A in one pass using $O(ndk) + \tilde{O}(d\alpha^{-3})$ time and $O(dk + d\alpha^{-1}))$ space, and maintains a matrix $B^{O(k+\alpha^{-1})\times d}$. With probability 0.99, it holds that $\|A^TA - B^TB\|_2 \le \alpha\|A - [A]_k\|_F^2$.*

Overview of the algorithm. To speed up FD, we will use the idea of adaptive random sampling. Let us first review the standard FD algorithm. Given an integer parameter $\ell \le d$, the algorithm always maintains a matrix B with at most 2ℓ rows at any time. When a new row v arrives, it processes the row using $\mathsf{FDShrink}(B, v, \ell)$ (Algorithm 3). In this procedure, we first append a after B; if B has no more than 2ℓ rows we do nothing, and otherwise do a $\mathsf{DenseShrink}$ operation (Algorithm 1) on B, which halves the number of rows in B (after removing zero rows). It was proved by Liberty (2013) and Ghashami and Phillips (2014) that for $\ell = k + \alpha^{-1}$, we have

$$\|A^TA - B^TB\|_2 \le \alpha\|A - [A]_k\|_F^2.$$

Since each SVD computation in $\mathsf{DenseShrink}$ takes $O(d\ell^2)$ time and there are totally n/ℓ SVD computations (SVD is applied every ℓ rows), the running time is $O(nd\ell) = O(nd(k + \alpha^{-1}))$. Our goal is to separate nd from α^{-1} in the running time.

Algorithm 1 $\mathsf{DenseShrink}$

Input: $B \in \mathbb{R}^{2\ell\times d}$.

1: Compute $[U, \Sigma, V] = \mathsf{SVD}(B)$, and $\sigma = \Sigma_{\ell,\ell}$.
2: $\hat{\Sigma} = \sqrt{\max(\Sigma^2 - \sigma^2 I_{2\ell}, 0)}$
3: **return** $B = \hat{\Sigma}V^T$

Algorithm 2 $\mathsf{DenseShrinkR}$

Input: $B \in \mathbb{R}^{2\ell\times d}$.

1: Compute $[U, \Sigma, V] = \mathsf{SVD}(B)$, and $\sigma = \Sigma_{\ell,\ell}$.
2: $\hat{\Sigma} = \sqrt{\max(\Sigma^2 - \sigma^2 I_{2\ell}, 0)}$
3: $\bar{\Sigma} = \sqrt{\Sigma^2 - \hat{\Sigma}^2}$ ▶ $\Sigma^2 = \bar{\Sigma}^2 + \hat{\Sigma}^2$
4: **return** $B = \hat{\Sigma}V^T$, $\bar{\Sigma}$ and V^T

Algorithm 3 $\mathsf{FDShrink}$

Input: $B \in \mathbb{R}^{\ell'\times d}$, $\mathbf{v} \in \mathbb{R}^d$, and an integer ℓ ▶ it always holds that $\ell' < 2\ell$.

1: $B = [B; \mathbf{v}]$
2: **If** $\ell' + 1 = 2\ell$, **then** $B = \mathsf{DenseShrink}(B, \ell)$.
3: **return** B

To achieve this, we first compute a coarse approximation using FD by invoking $B = \mathsf{FD}(A, k, \frac{1}{2k})$, which takes $O(ndk)$ time. The key idea here is that in each $\mathsf{DenseShrink}$ operation, after shrinking B, we also return the residual; we call this modified shrinking operation $\mathsf{DenseShrinkR}$ (see Algorithm 2). Let C be the matrix which is the concatenation of all residuals return from $\mathsf{DenseShrinkR}$. We will show $A^TA = B^TB + C^TC$ and $\|C\|_F^2 \le \|A - [A]_k\|_F^2$. We then refine the answer by computing an approximation to C. Since the norm of C is small, random sampling suffices. To save space, the sampled rows will be fed to a standard FD algorithm. See Algorithm 4 for detailed description of the algorithm.

Algorithm 4 FFDdense

Input: $A \in \mathbb{R}^{n\times d}$, $0 < \alpha < 1$, and integer $k \le d$.

1: $F = 0$, $\ell = 3k$, $Q =$ **empty**, $B =$ **empty**
2: **for** $i = 1$ **to** n **do**
3: Append A_i after B
4: **if** $\mathsf{rows}(B) = 2\ell$ **then**
5: $[B, \Sigma, V] = \mathsf{DenseShrinkR}(B)$ ▶ *Here $\ell = 3k$, thus $B = \mathsf{FD}(A, \frac{1}{2k}, k)$*
6: $F = F + \|\Sigma\|_F^2$, ▶ *Next: subsample $C := \Sigma V$, and then compress the sampled rows using standard* FD
7: **for** $j = 1$ **to** 2ℓ **do**
8: $p_j = \frac{\Sigma_j^2}{\alpha^2 F}$
9: Sample C_j with probability p_j.
10: **if** C_j is sampled **then**
11: Set $\mathbf{v} = \frac{C_j}{\sqrt{p_j}}$.
12: $Q = \mathsf{FDShrink}([Q; \mathbf{v}], \frac{1}{\alpha})$ ▶ *Invoking* FD *with $k = 0$*
13: **end if**
14: **end for**
15: **end if**
16: **end for**
17: **return** $[B; Q]$

Correctness. We note that, at the end of Algorithm 4, $B = \mathsf{FD}(A, \frac{1}{2k}, k)$, so $\|A^TA - B^TB\|_2 \le \|A - [A]_k\|_F^2/2k$, or equivalently

$$\max_{x:\|x\|=1} \big|\, \|Ax\|^2 - \|Bx\|^2 \,\big| \le \|A - [A]_k\|_F^2/2k. \tag{3}$$

Let $\Sigma^{(i)}$, $V^{(i)}$, and $B^{(i)}$ be the value of Σ, V, and B respectively returned by ith $\mathsf{DenseShrinkR}$ operation (line 5). Let $C^{(i)} = \Sigma^{(i)}V^{(i)}$. We use $B'^{(i)}$ to denote the value of B right before the ith $\mathsf{DenseShrinkR}$ operation (or the input of the ith $\mathsf{DenseShrinkR}$ operation). From Algorithm 2, we have that

$$B'^{(i)T}B'^{(i)} = B^{(i)T}B^{(i)} + V^{(i)T}\Sigma^{(i)2}V^{(i)} = B^{(i)T}B^{(i)} + C^{(i)T}C^{(i)}.$$

Let $A^{(i)}$ be the rows of A arrived between the $(i-1)$th and the ith $\mathsf{DenseShrinkR}$ operation, which means $B'^{(i)} = [B^{(i-1)}; A^{(i)}]$, and thus

$$B'^{(i)T}B'^{(i)} = B^{(i-1)T}B^{(i-1)} + A^{(i)T}A^{(i)}.$$

Combined with the previous equality, we get

$$A^{(i)T}A^{(i)} + B^{(i-1)T}B^{(i-1)} - B^{(i)T}B^{(i)} = C^{(i)T}C^{(i)}.$$

Let t be the total number of iterations. We define $B^{(0)} = 0$, and $C = [C^{(1)}; \cdots; C^{(t)}]$. Summing the above equality over $i = 1, \cdots, t$, we have

$$\begin{aligned} C^TC = \sum_i C^{(i)T}C^{(i)} &= \sum_i \left(A^{(i)T}A^{(i)} + B^{(i-1)T}B^{(i-1)} - B^{(i)T}B^{(i)}\right) \\ &= A^TA - B^TB. \end{aligned}$$

It follows that

$$\|C\|_F^2 \;=\; \mathsf{trace}(C^TC) = \mathsf{trace}(A^TA) - \mathsf{trace}(B^TB) = \|A\|_F^2 - \|B\|_F^2.$$

Now we bound $\|A\|_F^2 - \|B\|_F^2$ using similar ideas as in Ghashami and Phillips (2014). Let w_j be the jth singular vector of A, we have

$$\begin{aligned} \|C\|_F^2 &= \|A\|_F^2 - \|B\|_F^2 \\ &= \sum_{j=1}^{k} \|Aw_j\|^2 + \sum_{j=k+1}^{d} \|Aw_j\|^2 - \|B\|_F^2 \\ &\le \sum_{j=1}^{k} \|Aw_j\|^2 + \|A - [A]_k\|_F^2 - \sum_{j=1}^{k} \|Bw_j\|^2 \quad \text{because } \sum_{j=1}^{k} \|Bw_j\|^2 \le \|B\|_F^2 \\ &\le \|A - [A]_k\|_F^2 + k \cdot \|A - [A]_k\|_F^2/2k \quad \text{by Eq (3)} \\ &= 1.5\|A - [A]_k\|_F^2. \end{aligned} \tag{4}$$

In the algorithm, each row of C is sampled with probability $\frac{\|C_j\|^2}{\alpha^2 F}$, where F is the current squared F-norm of C. Let C_s be the sampled rows. Given Eq (4), we can prove the following using Theorem 5

$$\|C^TC - C_s^TC_s\|_2 \le \alpha\|A - [A]_k\|_F^2, \text{ and } \|C_s\|_F^2 = O(1) \cdot \|A - [A]_k\|_F^2.$$

At the end of the algorithm, $Q = \mathsf{FD}(C_s, \alpha, 0)$, then

$$\|C_s^TC_s - Q^TQ\|_2 \le \alpha\|C_s\|_F^2 \le O(\alpha) \cdot \|A - [A]_k\|_F^2.$$

Applying triangle inequality, we have $\|C^TC - Q^TQ\|_2 \le O(\alpha) \cdot \|A - [A]_k\|_F^2$, and thus

$$\|A^TA - B^TB - Q^TQ\|_2 = \|C^TC - Q^TQ\|_2 \le O(\alpha) \cdot \|A - [A]_k\|_F^2,$$

which proves the correctness.

Space and running time. The space is dominated by maintaining $B = \mathsf{FD}(A, 1/2k, k)$ and $Q = \mathsf{FD}(C_s, \alpha, 0)$, which is $O(dk + d/\alpha)$ in total.

The running time of computing B is $O(ndk)$, and the running time for Q is $O(\mathsf{rows}(C_s)d/\alpha)$. To bound $\mathsf{rows}(C_s)$, we divide the stream into epochs, where F roughly doubles in each epoch. This means the total number of epochs is bounded by $O(\log(nd))$, since we assume each real number in the input can be represented by $O(\log(nd))$ bits. We emphasize that a rigorous analysis on this will be more subtle; see discussions in the proof of Lemma 15 below. Applying Theorem 5 on the submatrix in each epoch, it is easy to check the expected number of rows sampled in each epoch is $O(1/\alpha^2)$, so $\mathsf{rows}(C_s) = O(\frac{\log(nd)}{\alpha^2})$. Thus the total running time is $O(ndk) + \tilde{O}(d\alpha^{-3})$. We remark that the residual return by $\mathsf{DenseShrinkR}$ is in the form of $C = \Sigma V^T$, where Σ is diagonal and V has orthonormal columns. Therefore, the row norms of C are simply the diagonals of Σ.

3. Algorithm for Sparse Matrices

For sketching sparse matrices, we will use a subroutine (see Algorithm 5) for computing weak low rank approximations based on fast subspace embeddings. In this subroutine, the input matrix is an $\ell \times d$ matrix A and two additional matrices J and C, where J has $O(k)$ rows and C has d columns. In our applications, J, C will always be the subspace embedding matrices from Theorem 6. The algorithm outputs an $O(k) \times \ell$ matrix Z with orthonormal rows.

Algorithm 5 Weak Low Rank Approximation (LRA)

Input: $A \in \mathbb{R}^{\ell \times d}$, $J \in \mathbb{R}^{O(k) \times w}$ and $C \in \mathbb{R}^{w \times d}$

1: Compute $S = AC^T J^T$
2: Compute $Z \in \mathbb{R}^{O(k) \times \ell}$ whose rows form an orthonormal basis for the column space of S
3: **return** Z

The following lemma is a simple corollary of Theorem 6.

Lemma 8 *If J, C are the subspace embedding matrices from Theorem 6 with $w = \min(d, O(k^2))$ and Z is the output of Algorithm 5, then with probability* 0.99

$$\|A - Z^T ZA\|_F^2 \leq O(1)\|A - [A]_k\|_F^2.$$

Moreover, Z can be computed in $O(\mathsf{nnz}(A) + \ell k^3)$ time and $O(\ell k^2)$ extra space.

Proof From Theorem 6, there exists X with $\mathsf{rank}(X) \leq k$ such that

$$\|A - Z^T X\|_F^2 \leq O(1)\|A - [A]_k\|_F^2.$$

Hence,

$$\begin{aligned}\|A - Z^T ZA\|_F^2 &\leq \|A - Z^T ZA\|_F^2 + \|Z^T ZA - Z^T X\|_F^2 \\ &= \|A - Z^T X\|_F^2 \quad \text{Pythagorean theorem} \\ &\leq O(1)\|A - [A]_k\|_F^2.\end{aligned}$$

Note that S can be computed in time $O(\mathsf{nnz}(A) + \ell k^3)$ and Z can be computed from S in $O(\ell k^2)$ time. The space usage is dominated by storing the intermediate result AC^T, which is $O(\ell k^2)$. ■

3.1. Overview of Our Algorithm

Our approach is quite different from Ghashami et al. (2016). Their main idea is to use fast approximate SVD (Musco and Musco, 2015) in the original FD, which leads to suboptimal time. Our approach is summarized as follows.

1. Decompose $A^T A = A'^T A' + R^T R$, such that A' contains small number of rows and $\|A' - [A']_k\|_F^2 = O(1) \cdot \|A - [A]_k\|_F^2$. Moreover, $\|R\|_F^2 = O(1) \cdot \|A - [A]_k\|_F^2$.

2. Compute a sketch B of A' using fast FD algorithm for dense matrices (Theorem 7), which satisfies that $\|A'^T A' - B^T B\|_2 \leq \alpha\|A' - [A']_k\|_F^2 \leq \alpha\|A - [A]_k\|_F^2$.

3. Compute a sketch matrix C of R such that $\|R^T R - C^T C\|_2 \leq \alpha\|R\|_F^2 \leq O(\alpha) \cdot \|A - [A]_k\|_F^2$, which can be done via random sampling (Theorem 5) combined with FD.

4. The final sketch is $S = [B; C]$.

Note that $S = [B; C]$ approximate $[A'; R]$ in the sense that

$$\begin{aligned}\|A'^T A' + R^T R - B^T B - C^T C\|_2 &\leq \|A'^T A' - B^T B\|_2 + \|R^T R - C^T C\|_2 \\ &\leq O(\alpha) \cdot \|A - [A]_k\|_F^2.\end{aligned}$$

From step (1), we have $A^T A = A'^T A' + R^T R$, and thus $[B; C]$ is a good approximation of A. Next we briefly describe how to implement this in one pass and small space.

To achieve (1), we use the following new idea. Let $Z \in \mathbb{R}^{O(k) \times d}$ be an orthonormal matrix satisfying $\|A - Z^T ZA\|_F^2 \leq O(1)\|A - [A]_k\|_F^2$. Let $A' = ZA$ and $R = (I - Z^T Z)A$. It is easy to check that A' and R satisfy the requirement of (1). In the streaming model, we divide A into blocks, each of which contains roughly dk non-zero entries, and thus there are at most $t = \frac{nnz(A)}{dk}$ blocks. We use the above idea for each of the blocks and concatenate the results together. More precisely, for each block $A^{(i)} \in \mathbb{R}^{\ell_i \times d}$, we use an input-sparsity time algorithm (Algorithm 5 and Lemma 8) to compute a matrix $Z^{(i)} \in \mathbb{R}^{O(k) \times \ell_i}$ such that

$$\|A^{(i)} - Z^{(i)T} Z^{(i)} A^{(i)}\|_F^2 \leq O(1)\|A^{(i)} - [A^{(i)}]_k\|_F^2.$$

Let $A'^{(i)} = Z^{(i)} A^{(i)}$, and $R^{(i)} = (I - Z^{(i)T} Z^{(i)})A^{(i)}$. We then set $A' = [A'^{(1)}; \cdots; A'^{(t)}]$ and $R = [R^{(1)}; \cdots; R^{(t)}]$, and prove that A' and R satisfy the requirement of (1), where A' only has $t \times O(k) = O(\frac{\mathsf{nnz}(A)}{d})$ rows (since each block of A' has $O(k)$ rows). Here we do not compute R explicitly, as we will sample a subset of the rows from R. Note that the running time of this step is dominated by computing $Z^{(i)} A^{(i)}$, which is $O(\mathsf{nnz}(A^{(i)})k)$, and thus $O(\mathsf{nnz}(A)k)$ in total.

To compute B of step (2), we may use the standard $\mathsf{FD}(A', \alpha, k)$ (Theorem 4). Since A' has at most $O(\frac{\mathsf{nnz}(A)}{d})$ rows, B can be computed in $O(\mathsf{nnz}(A)(k + \alpha^{-1}))$ time. However, it still has an $\mathsf{nnz}(A)\alpha^{-1}$ term. So, we apply our faster FD algorithm for dense matrices (Theorem 7) on A', which only takes $O(\mathsf{nnz}(A)k) + \tilde{O}(d\alpha^{-3})$ time.

In order to compute a sketch C of R in step (3), we first subsample the rows of R using streaming Bernoulli sampling. One difficulty is that R could be dense, and it may take nd time to compute the row norms. Fortunately, each $R^{(i)}$ is of special form, and we are able to design a fast algorithm to compute its row norms (Algorithm 7). Let Q be the sampled rows, with $\mathsf{rows}(Q) = \tilde{O}(1/\alpha^2)$, and note that each row of Q can be computed in time $O(kd)$ as $Z^{(i)} A^{(i)}$ has already been computed in step (1). We finally use $\mathsf{FD}(Q, \alpha, 0)$ to compute a sketch matrix C of Q in time $\tilde{O}(d\alpha^{-3})$. In all, the running time is roughly $O(\mathsf{nnz}(A)k) + \tilde{O}(d\alpha^{-3} + dk\alpha^{-2})$.

3.2. Our Algorithm

Algorithm 6 FFDsparse

Input: $A \in \mathbb{R}^{n\times d}$, $\alpha \in (0,1)$, and integers $k \le d$.

$F = \eta$, $F' = 0$, $B = \mathbf{0}$, $Q = \mathbf{0}$ ▶ η will be determined in Lemma 15

Divide the rows of A into continuous blocks $A^{(1)}, \cdots, A^{(t)}$: we will put new rows into the current block until: a) the number of non-zero entries exceeds dk, or b) the number of rows is $\frac{d}{k}$. When either a) or b) happens, we start a new block. Note that the total number of blocks $t \le \frac{\mathsf{nnz}(A)}{dk} + \frac{nk}{d}$.

Choose subspace embedding matrices J, C (Theorem 6) ▶ J, C are fixed for all blocks

for $i = 1$ **to** t **do**

  Compute $Z^{(i)} = \mathsf{LRA}(A^{(i)}, J, C)$ (Algorithm 5). Let $\ell_i = \mathsf{rows}(A^{(i)})$.

  Compute $A'^{(i)} = Z^{(i)} A^{(i)}$.

  $w = \mathsf{RowNorms}(A^{(i)}, Z^{(i)})$ ▶ w contains the row norms of $A^{(i)} - Z^{(i)T} Z^{(i)} A^{(i)}$ (Algorithm 7)

  $F' = F' + \|w\|^2$, and if $F' \ge 2F$, $F = F'$.
  ▶ We always have $F = O(1) \cdot \sum_i \|(I - Z^{(i)T} Z^{(i)}) A^{(i)}\|_F^2$

  Let $p \in \mathbb{R}^{\ell_i}$ such that $p_j = \frac{w_j^2}{\alpha^2 F}$ for $j = 1, \cdots, \ell_i$. Let $x \in \mathbb{R}^{\ell_i}$ be a random vector with independent entries: For each j, $x_j = 1/p_j$ w.p. p_j, and $x_j = 0$ w.p. $1 - p_j$.

  Let $R^{(i)} = (I - Z^{(i)T} Z^{(i)}) A^{(i)}$ and $Q^{(i)} = \mathsf{Diag}(x) \cdot R^{(i)}$ (no need to compute $R^{(i)}$ explicitly).

  $B = \mathsf{FFDdense}([B; A'^{(i)}], \alpha, k)$. ▶ Sketching $A' = [A'^{(1)}; \cdots; A'^{(t)}]$ using Theorem 7

  $C = \mathsf{FD}([C; Q^{(i)}], \alpha, 0)$. ▶ Sketching $Q = [Q^{(1)}; \cdots; Q^{(t)}]$ using FD.

end for

return $[B; C]$

Theorem 9 (FFDsparse) *Given any matrix $A \in \mathbb{R}^{n\times d}$, $0 < \alpha < 1$ and $0 \le k \le d$, $\mathsf{FFDsparse}(A, \alpha, k)$ (Algorithm 6) maintains a matrix S in a streaming fashion, such that, with probability 0.9,*

$$\|A^T A - S^T S\|_2 \le \alpha \|A - [A]_k\|_F^2.$$

The algorithm uses $O(d(k + \alpha^{-1}))$ space and $O(\mathsf{nnz}(A)k + nk^3) + \tilde{O}(d\alpha^{-3} + dk\alpha^{-2})$ time.

By Lemma 3, we also have the following result.

Theorem 10 *Given any matrix $A \in \mathbb{R}^{n\times d}$, $0 < \varepsilon < 1$ and $0 < k \le d$, there is a streaming algorithm which maintains a strong (ε, k)-proj-sketch $S \in \mathbb{R}^{O(k/\varepsilon)\times d}$. The algorithm uses $O(dk/\varepsilon)$ space and runs in $O(\mathsf{nnz}(A)k + nk^3) + \tilde{O}(dk^3\varepsilon^{-3}))$ time.*

3.3. Proof of Theorem 9

The detail of our fast algorithm for sparse matrix is described in Algorithm 6. We let $A' = [A'^{(1)}; \cdots; A'^{(t)}]$, $R = [R^{(1)}; \cdots; R^{(t)}]$, and $Q = [Q^{(1)}; \cdots; Q^{(t)}]$. We use $w^{(i)}$ to denote the vector w in ith iteration. We need some technical lemmas.

Lemma 11 *With probability at least 0.99, we have (1) $\|A' - [A']_k\|_F^2 \le \|A - [A]_k\|_F^2$; (2) $\|R\|_F^2 \le O(1) \cdot \|A - [A]_k\|_F^2$.*

Proof We divide A and A' into blocks as defined in Algorithm 6, i.e. $A = [A^{(1)}; \cdots; A^{(t)}]$ and $A' = [A'^{(1)}; \cdots; A^{(t)}]$. For each i, we have $A'^{(i)} = Z^{(i)} A^{(i)}$ for some matrix $Z^{(i)}$ with $O(k)$ orthonormal rows.

Let P be the projection matrix onto the subspace spanned by the top-k right singular vectors of A. So we have

$$\begin{aligned}
\|A' - [A']_k\|_F^2 &\le \|A' - A'P\|_F^2 && \text{since } P \text{ is of rank } k \\
&= \sum_{i=1}^{t} \|A'^{(i)} - A'^{(i)} P\|_F^2 \\
&= \sum_{i=1}^{t} \|Z^{(i)} A^{(i)} - Z^{(i)} A^{(i)} P\|_F^2 \\
&\le \sum_{i=1}^{t} \|A^{(i)} - A^{(i)} P\|_F^2 && Z^{(i)} \text{ is a orthonormal} \\
&= \|A - AP\|_F^2 = \|A - [A]_k\|_F^2, && \text{by definition of } P
\end{aligned}$$

which proves (1).

As defined in Algorithm 6, $R^{(i)} = (I - Z^{(i)T} Z^{(i)}) A^{(i)}$, where $Z^{(i)} = \mathsf{LRA}(A^{(i)}, J, C)$. Since $\|A^{(i)} - Z^{(i)T} Z^{(i)} A^{(i)}\|_F^2 \le O(1) \cdot \|A^{(i)} - [A^{(i)}]_k\|_F^2$ for each i (by Lemma 8), one gets

$$\begin{aligned}
\|R\|_F^2 = \sum_{i=1}^{t} \|R^{(i)}\|_F^2 &= \sum_{i=1}^{t} \|(I - Z^{(i)T} Z^{(i)}) A^{(i)}\|_F^2 \\
&\le O(1) \cdot \sum_{i=1}^{t} \|A^{(i)} - [A^{(i)}]_k\|_F^2 \\
&\le O(1) \cdot \|A - [A]_k\|_F^2
\end{aligned}$$

which proves (2). One caveat: the above argument needs $\|A^{(i)} - Z^{(i)T}Z^{(i)}A^{(i)}\|_F^2 \le O(1) \cdot \|A^{(i)} - [A^{(i)}]_k\|_F^2$ to hold for all i simultaneously. To achieve this, one could use a similar idea as in Boutsidis et al. (2016) to first boost the success probability in Lemma 8 and then apply a union bound, but this results in an extra log factor in the time complexity.

Thus, we use a different argument to bypass this. Let $S = AC^TJ^T$ and write S in the block form as $S = [S^{(1)}; \cdots; S^{(t)}]$.

Note that $S^{(i)} = A^{(i)}C^TJ^T$ and the rows of $Z^{(i)}$ form an orthonormal basis for the column space of $S^{(i)}$, thus for each i

$$\begin{aligned}
&\min_{X:\mathsf{rank}(X)\le k} \|A^{(i)} - S^{(i)}X\|_F^2 = \min_{X:\mathsf{rank}(X)\le k} \|A^{(i)} - Z^{(i)T}X\|_F^2 \\
&= \min_{X:\mathsf{rank}(X)\le k} \|A^{(i)} - Z^{(i)T}Z^{(i)}A^{(i)} + Z^{(i)T}Z^{(i)}A^{(i)} - Z^{(i)T}X\|_F^2 \\
&= \min_{X:\mathsf{rank}(X)\le k} \left(\|A^{(i)} - Z^{(i)T}Z^{(i)}A^{(i)}\|_F^2 + \|Z^{(i)T}Z^{(i)}A^{(i)} - Z^{(i)T}X\|_F^2\right) \quad \text{by Pythagorean} \\
&\ge \|A^{(i)} - Z^{(i)T}Z^{(i)}A^{(i)}\|_F^2
\end{aligned}$$

Therefore, we get

$$\begin{aligned}
\min_{X:\mathsf{rank}(X)\le k} \|A - SX\|_F^2 &= \min_{X:\mathsf{rank}(X)\le k} \sum_{i=1}^t \|A^{(i)} - S^{(i)}X\|_F^2 \\
&\ge \sum_{i=1}^t \min_{X:\mathsf{rank}(X)\le k} \|A^{(i)} - S^{(i)}X\|_F^2. \\
&\ge \sum_{i=1}^t \|A^{(i)} - Z^{(i)T}Z^{(i)}A^{(i)}\|_F^2 \qquad (5)
\end{aligned}$$

One the other hand, by Theorem 6, with probability 0.99

$$\min_{X:\mathsf{rank}(X)\le k} \|A - SX\|_F^2 \le O(1)\|A - [A]_k\|_F^2. \qquad (6)$$

Combining (5) and (6), we finally get

$$\|R\|_F^2 = \sum_{i=1}^t \|R^{(i)}\|_F^2 = \sum_{i=1}^t \|A^{(i)} - Z^{(i)T}Z^{(i)}A^{(i)}\|_F^2 \le O(1)\|A - [A]_k\|_F^2.$$

Note that, in the above proof, we only require one probabilistic event, i.e., (6), to happen, and thus avoid the union bound argument. This finish the proof of part (2). ■

Lemma 12 *$A^TA = R^TR + A'^TA'$; and with probability 0.99, it holds that $\|A^TA - A'^TA'\|_F \le O(1) \cdot \|A - [A]_k\|_F^2$.*

Proof To prove the first part, we only need to prove $A^{(i)T}A^{(i)} = R^{(i)T}R^{(i)} + A'^{(i)T}A'^{(i)}$ holds for all i. Recall that $A'^{(i)} = Z^{(i)}A^{(i)}$. For each i, we have

$$\begin{aligned}
R^{(i)T}R^{(i)} &= A^{(i)T}(I - Z^{(i)T}Z^{(i)}) \cdot (I - Z^{(i)T}Z^{(i)})A^{(i)} \\
&= A^{(i)T}(I - Z^{(i)T}Z^{(i)})A^{(i)} \qquad \text{since } (I - Z^{(i)T}Z^{(i)}) \text{ is a projection} \\
&= A^{(i)T}A^{(i)} - A'^{(i)T}A'^{(i)}.
\end{aligned}$$

This proves the first part, from which, we also get

$$\|A^TA - A'^TA'\|_F = \|R^TR\|_F \le \|R\|_F^2,$$

where the inequality is from the submultiplicative property of the Frobenius norm. Then the second part follows from Lemma 11. ■

Lemma 13 *If the entries of A are integers bounded in magnitude by $\mathrm{poly}(nd)$ and $\mathsf{rank}(A) \ge 1.1k$, then $\|A - [A]_k\|_F^2 \ge 1/\mathrm{poly}(nd)$.*

Proof The lemma directly follows from a result of Clarkson and Woodruff (2009), and here we use the restated version from Boutsidis et al. (2016).

Lemma 14 (Lemma 37 of Boutsidis et al. (2016)) *If an $n \times d$ matrix A has integer entries bounded in magnitude by γ, and has rank ρ, then the k-th largest singular value of A satisfies*

$$\sigma_k \ge (nd\gamma^2)^{-k/2(\rho-k)}.$$

■

Lemma 15 *We set $\eta = \mathrm{poly}^{-1}(nd)$ (see line 1 of Algorithm 6), then with probability at least 0.99, it holds that*

$$\|Q\|_F^2 = O(\|R\|_F^2),\ \|R^TR - Q^TQ\|_2 = O(\alpha) \cdot \|A - [A]_k\|_F^2,$$

$$\text{and } \mathsf{rows}(Q) = O(\log(nd)/\alpha^2).$$

Proof Let us first assume $\mathsf{rank}(A) \ge 1.1k$. Each row R_i of R is sampled with probability $\Theta(\frac{\|R_i\|^2}{\alpha^2 F})$, with F initialized to be η. We have $\eta \le \|A - [A]_k\|_F^2$ (by Lemma 13). When $\|R\|_F^2 \ge \eta$, the sampling probability for each row is at least $\Omega(\frac{\|R_i\|^2}{\alpha^2\|R\|_F^2})$ (since F is at most a constant approximation to $\|R\|_F^2$ at any time), so the first two parts directly follow from Theorem 5 and Lemma 11. Otherwise if $\|R\|_F^2 \le \eta \le \|A - [A]_k\|_F^2$, then the probability for sampling each row is $\Omega(\frac{\|R_i\|^2}{\alpha^2\eta}) = \Omega(\frac{\|R_i\|^2}{\alpha^2\|A-[A]_k\|_F^2})$ and the first two parts follow from Theorem 5.

To bound the number of rows sampled, we divide the stream into epochs, where F roughly doubles in each epoch. So the total number of epochs is bounded by $O(\log \frac{\|R\|_F}{\eta})$, as the final value of F is at most $O(\|R\|_F^2)$. Recall that we assume each entry of A is an integer bounded in magnitude by $\mathrm{poly}(nd)$, which implies the number of epochs is

$$O(\log \frac{\|R\|_F^2}{\eta}) \le O\left(\log\left(\|A\|_F^2 \cdot \mathrm{poly}(nd)\right)\right) = O(\log(nd)).$$

The number of rows sampled in each epoch is at most $O(1/\alpha^2)$: let $a_1, \cdots, a_t$ be the rows of R in the epoch, and thus $\sum_j \|a_j\|^2 \le O(F)$ (since the epoch ends if F doubles); each row

a_j is sampled with probability $\Theta(\frac{\|a_j\|^2}{\alpha^2 F})$, which implies the total number of rows sampled is $O(1/\alpha^2)$. This proves the third part.

The case $\mathsf{rank}(A) \le 1.1k$ is easier: we can set the rank parameter k a little larger (say $k' = 2k$) in our algorithm so that R is always $\mathbf{0}$ by Lemma 11, and thus $\mathsf{rows}(Q) = 0$. In this case, our algorithm is essentially exact. ■

Correctness. By union bound, with probability 0.9, all the above lemmas hold simultaneously, and we assume this happens. Since $C = \mathsf{FD}(Q, \alpha, 0)$, by Theorem 4 and Lemma 15, we have

$$\|Q^TQ - C^TC\|_2 \le \alpha\|Q\|_F^2 \le O(\alpha)\cdot\|R\|_F^2. \tag{7}$$

Since $B = \mathsf{FFDdense}(A', \alpha, k)$, by Theorem 7, we have

$$\|A'^TA' - B^TB\|_2 \le \alpha\|A' - [A']_k\|_F^2 \le \alpha\|A - [A]_k\|_F^2, \tag{8}$$

where the last inequality is from Lemma 11. Let $S = [B; C]$,

$$\begin{aligned}
&\|A^TA - S^TS\|_2 = \|A^TA - B^TB - C^TC\|_2\\
&\le \|A^TA - A'^TA' - C^TC\|_2 + \|A'^TA' - B^TB\|_2\\
&\le \|A^TA - A'^TA' - C^TC\|_2 + \alpha\|A - [A]_k\|_F^2 \qquad \text{by (8)}\\
&= \|R^TR - C^TC\|_2 + \alpha\|A - [A]_k\|_F^2\\
&\le \|R^TR - Q^TQ\|_2 + \|Q^TQ - C^TC\|_2 + \alpha\|A - [A]_k\|_F^2 \quad \text{triangle inequality}\\
&\le O(\alpha)\cdot\|A - [A]_k\|_F^2 + O(\alpha)\cdot\|R\|_F^2 + \alpha\|A - [A]_k\|_F^2 \qquad \text{by (7) and Lemma 15}\\
&\le O(\alpha)\cdot\|A - [A]_k\|_F^2, \qquad \text{by Lemma 11}
\end{aligned}$$

which proves the error bound after adjusting α by a constant.

Running time. Let ℓ_i be the number of rows in ith block $A^{(i)}$. The time to compute $Z^{(i)}$ using Lemma 8 is $O(\mathsf{nnz}(A^{(i)}) + \ell_i k^2)$. Hence the total time used on this step is

$$\sum_i O(\mathsf{nnz}(A^{(i)}) + \ell_i k^2) = O(\mathsf{nnz}(A) + nk^2).$$

The step to compute the matrix multiplication $A'^{(i)} = Z^{(i)}A^{(i)}$ takes $O(\mathsf{nnz}(A^{(i)})k)$ time, since Z has $O(k)$ rows. So the total time spent on this step is $O(\mathsf{nnz}(A)k)$. By definition, there are at most $O(\frac{\mathsf{nnz}(A)}{dk} + \frac{nk}{d})$ blocks. Moreover, after left multiplied by $Z^{(i)}$, each block contributes $O(k)$ rows to A', and thus the total number of rows in A' is at most $O(\frac{\mathsf{nnz}(A)}{d} + \frac{nk^2}{d})$. Computing B by invoking $B = \mathsf{FFDdense}(A', \alpha, k)$ needs $O(\mathsf{rows}(A')dk) + \tilde{O}(d/\alpha^3) = O(\mathsf{nnz}(A)k + nk^3) + \tilde{O}(d/\alpha^3)$ time. Finally, each row of Q can be computed in time $O(dk)$ given A' (which has been computed in line 6). Invoking $C = \mathsf{FD}(Q, \alpha, 0)$ needs $\tilde{O}(d/\alpha^3 + dk/\alpha^2)$ since $\mathsf{rows}(Q) = \tilde{O}(1/\alpha^2)$ by Lemma 15. So far, the total time is $O(\mathsf{nnz}(A)k + nk^3) + \tilde{O}(d\alpha^{-3} + dk\alpha^{-2})$.

The time for naively computing the row norms of R is high. One may use Johnson-Lindenstrauss transforms (Johnson and Lindenstrauss, 1984) to reduce the running time,

since constant approximations are good enough. By standard arguments, it is not hard to verify that the running time will be $O(\mathsf{nnz}(A)\log n)$ with JL transforms. If $k = \Omega(\log n)$, this term is absorbed by the leading term $O(\mathsf{nnz}(A)k)$; if $k = o(\log n)$, however, this becomes the dominating part. We next provide an algorithm, which computes the *exact* row norms of $A - Z^TZA$ in time $O(\mathsf{nnz}(A)k)$. Using this algorithm, the total time of $\mathsf{FFDsparse}$ is $O(\mathsf{nnz}(A)k + nk^3) + \tilde{O}(d\alpha^{-3} + dk\alpha^{-2})$.

Faster algorithm for computing row norms.

Lemma 16 *Given any $A \in \mathbb{R}^{n\times d}$ and $Z \in \mathbb{R}^{O(k)\times n}$, the exact row norms of $A - Z^TZA$ can be computed in time $O(\mathsf{nnz}(A)k + dk^2 + nk^2)$ and in space $O(nk + dk)$.*

Proof The algorithm is presented in Algorithm 7. We first compute ZA in $\mathsf{nnz}(A)k$ time,

Algorithm 7 RowNorms

Input: $A \in \mathbb{R}^{n\times d}$, $Z \in \mathbb{R}^{O(k)\times\ell}$
1: Compute $S = ZA$
2: Compute $[U, \Sigma, V] = \mathsf{SVD}(S)$
3: Compute $L = Z^TU\Sigma$
4: **for** $i = 1$ **to** n **do**
5: Compute $\mathbf{a}^{(1)} = A_iV$
6: Compute $w_i = \left\|\mathbf{a}^{(1)} - L_i\right\|^2 + \|A_i\|^2 - \|\mathbf{a}^{(1)}\|^2$
7: **end for**
8: **return** $w = [w_1, \cdots, w_n]$

then perform SVD on ZA: $ZA = U\Sigma V^T$. Since ZA is a $O(k)\times d$ matrix, this takes $O(dk^2)$ time. Next compute $L = Z^TU\Sigma$, which takes $O(\mathsf{rows}(A)k^2)$ time. Let $\mathbf{a}$ be the ith row in A. We compute $\mathbf{a}^{(1)} = \mathbf{a}V$ (in $O(\mathsf{nnz}(a)k)$ time), and set $\mathbf{b} = \mathbf{a} - \mathbf{a}^{(1)}V^T$ (we don't compute $\mathbf{b}$ explicitly). Then decompose $\mathbf{a}$ into two orthogonal parts as

$$\mathbf{a} = \mathbf{a}^{(1)}V^T + \mathbf{b}.$$

The ith row of $A - Z^TZA$ is

$$\mathbf{a} - L_iV^T = \left(\mathbf{a}^{(1)} - L_i\right)V^T + \mathbf{b}.$$

Note that V^T and $\mathbf{b}$ are orthogonal, and thus

$$\|\mathbf{a} - L_iV^T\|^2 = \|\left(\mathbf{a}^{(1)} - L_i\right)V^T\|^2 + \|\mathbf{b}\|^2 = \left\|\mathbf{a}^{(1)} - L_i\right\|^2 + \|\mathbf{a}\|^2 - \|\mathbf{a}^{(1)}\|^2,$$

which can be computed in time $O(\mathsf{nnz}(\mathbf{a}) + k)$. In all, the total time to compute all the row norms in $A - Z^TZA$ is

$$\mathsf{nnz}(A)k + dk^2 + \mathsf{rows}(A)k^2 + \sum_{i=1}^{\mathsf{rows}(A)} O(\mathsf{nnz}(A_i) + k) = O(\mathsf{nnz}(A)k + dk^2 + \mathsf{rows}(A)k^2).$$

■

Therefore, the time to compute all the row norms in the jth block $A^{(j)} - Z^T Z A^{(j)}$ is $O(\mathsf{nnz}(A^{(j)})k + dk^2 + \mathsf{rows}(A^{(j)})k^2)$, and the total time over all block is thus

$$\sum_{j=1}^{t} O(\mathsf{nnz}(A^{(i)})k + dk^2 + \mathsf{rows}(A^{(i)})k^2) = O(\mathsf{nnz}(A)k + nk^3),$$

where we use the fact that $t = O(\frac{\mathsf{nnz}(A)}{dk} + \frac{nk}{d})$.

Space. For space, we need a buffer to store a new block of A, the size of which is at most $dk + d$, as $\mathsf{nnz}(A^{(i)})$ is at most $dk + d$. When using Algorithm 5, the input matrix always has at most $\frac{d}{k}$ rows, so we need $O(dk)$ space to compute and store each $Z^{(i)}$ (Lemma 8). $A'^{(i)}$ is of dimension $O(k) \times d$, which needs $O(dk)$ space to compute and store. According to Lemma 16, the space needed to compute the row norms of each $R^{(i)}$ is $O(\ell_i k + dk) = O(dk)$, since $\ell_i \leq \frac{d}{k}$ for all i. From Theorem 7, the space used by $\mathsf{FFDdense}(A', \alpha, k)$ is $O(d(k+\alpha^{-1}))$. Note that, in line 12 of Algorithm 6, the rows of $Q^{(i)}$ can be computed one by one and fed to FD directly, and thus compute $C = \mathsf{FD}(Q, \alpha, 0)$ uses $O(d/\alpha)$ space by theorem 4. In all, the total space usage is bounded by $O(d(k + \alpha^{-1}))$.

4. On the Equivalence Between Frequent Directions and Matrix Multiplication

In this section, we show that, under mild assumptions, the time complexity of Frequent Directions is essentially equivalent to that of matrix multiplication modulo lower order terms.

Let $\mathcal{A}$ be an algorithm for multiplying a matrix $A \in \mathbb{R}^{n\times d}$ with an arbitrary k by n matrix; and $T(\mathsf{nnz}(A), k, n, d)$ be the running time. In this section we will assume T is non-decreasing in the first parameter $\mathsf{nnz}(A)$. Moreover, T is additive in the first parameter, i.e., $T(a, k, n, d) + T(b, k, n, d) = T(a + b, k, n, d)$ for any non-negative a and b.

4.1. Faster Matrix Multiplication Implies Faster FD

This direction follows from our $\mathsf{FFDsparse}$ algorithm.

Theorem 17 *Let $\mathcal{A}$ be an algorithm for multiplying a matrix $A \in \mathbb{R}^{n\times d}$ with an arbitrary k by n matrix and assume its running time is $T(\mathsf{nnz}(A), k, n, d)$. If T is additive and non-decreasing in the first parameter $\mathsf{nnz}(A)$, then Algorithm 6 can be implemented in time $T(\mathsf{nnz}(A), O(k), \frac{d}{k}, d) + O(nk^3) + \tilde{O}(d\alpha^{-3} + dk\alpha^{-2})$.*

Proof First, we note that the time complexity of our algorithm for sketching sparse matrices is dominated by computing $Z^{(i)}A^{(i)}$ (Line 6 in Algorithm 6) for $i = 1, 2, \cdots, t$, where $Z^{(i)}$ contains $O(k)$ rows. Using $\mathcal{A}$, this takes $T(\mathsf{nnz}(A^{(i)}), O(k), \frac{d}{k}, d)$ time, since the number of rows in $A^{(i)}$ is at most d/k in the algorithm. By the additive assumption, the total time is

$$\sum_{i=1}^{t} T(\mathsf{nnz}(A^{(i)}), O(k), \frac{d}{k}, d) = T(\mathsf{nnz}(A), O(k), \frac{d}{k}, d).$$

■

4.2. Faster FD Implies Faster Matrix Multiplication

In this subsection, we prove the other direction by showing that the running time of FD is lower bounded by matrix multiplication. The proof is based on the idea of Musco and Woodruff (2017). In particular, we show that the existence an algorithms which computes an $(O(1), k)$-cov-sketch in time $o(\mathsf{nnz}(A)k)$ implies a breakthrough in matrix multiplication, which is likely very difficult. In fact, the lower bound holds even for offline algorithms without constraints on working space.

Theorem 18 *Assume there is an algorithm $\mathcal{A}$, which, given any $A \in \mathbb{R}^{n\times d}$ with polynomially bounded integer entries, returns $B \in \mathbb{R}^{O(k)\times d}$ in time $T(\mathsf{nnz}(A), k, n, d)$ such that*

$$\|A^T A - B^T B\|_2 \leq \Delta \|A - [A]_k\|_F^2,$$

for some constant error parameter Δ. If T is additive and non-decreasing in the first parameter, then there is a $T(\mathsf{nnz}(M), k, n, d) + T(nk, k, n, d) + O(dk^2)$ time algorithm for multiplying arbitrary polynomially bounded integer matrices $M^T \in \mathbb{R}^{(d-k)\times n}$, $C \in \mathbb{R}^{n\times k}$.

Proof For any matrices $M \in \mathbb{R}^{n\times(d-k)}$ and $C \in \mathbb{R}^{n\times k}$ with integer entries in $[-U, U]$, let $A \in \mathbb{R}^{n\times d}$ be the matrix which is a concatenation of the columns of M and wC, i.e., $A = [M, wC]$. Here w is large number will be determined later. We have $\|A - [A]_k\|_F^2 \leq \|M\|_F^2 \leq ndU^2$ and

$$A^T A = \begin{bmatrix} M^T M & wM^T C \\ wC^T M & w^2 C^T C \end{bmatrix}.$$

We assume $\mathcal{A}$ is an algorithm with running time T, which can compute a sketch matrix $B \in \mathbb{R}^{O(k)\times d}$ of A such that

$$\|A^T A - B^T B\|_2 \leq \Delta \|A - [A]_k\|_F^2 \leq \Delta U^2 nd,$$

for some constant error parameter Δ.

The spectral norm of a matrix N is the largest singular value, which can be equivalently defined as $\|N\|_2 = \max_{x,y:\|x\|=\|y\|=1} x^T N y$, thereby $N_{i,j} = e_i^T N e_j \leq \|N\|_2$ for all i, j. It follows that $(A^T A - B^T B)_{i,j} \leq \Delta U^2 nd$, meaning the corresponding block of $B^T B$ is an entry-wise approximation to $wM^T C$ within additive error $\Delta U^2 nd$.

Now if w is a large integer, say $w = \lceil 3\Delta U^2 nd \rceil$, we can recover $M^T C$ from $B^T B$ exactly by rounding the numbers in $B^T B$ to their nearest integer multiple of w (as $M^T C$ is an integer matrix). Note $B^T B$ can be computed in time $O(dk^2)$ given B and the rounding can be done in $O(dk)$, so using $\mathcal{A}$, the exact integer matrix multiplication $M^T C$ can be computed in time

$$T(\mathsf{nnz}(A), k, n, d) + O(dk^2) = T(\mathsf{nnz}(M), k, n, d) + T(nk, k, n, d) + O(dk^2).$$

Here we used the fact that $\mathsf{nnz}(wC) \leq nk$ and the assumption that T is linear in the first parameter. We remark that all the integers in our reduction are at most $\mathrm{poly}(nd)$ in magnitude, as long as $U = \mathrm{poly}(nd)$, so our reduction works for any M, C with polynomially bounded entries. ■

Therefore, if $T = o(\mathsf{nnz}(A)k) = o(\mathsf{nnz}(M)k + nk^2)$, then $M^T C$ can be computed in time $o(\mathsf{nnz}(M)k) + O(nk^2 + dk^2)$, which will be a breakthrough in fast matrix multiplication.

5. Conclusion

This paper studies the problem of computing covariance sketches for matrices in the streaming model. Based on several novel algorithmic techniques, we provide new space-optimal algorithms with improved running time. We also prove that, under mild assumptions, the time complexity of Frequent Directions is essentially equivalent to that of matrix multiplication up to lower order terms. In particular, this implies that the running times of FFDdense and FFDsparse cannot be significantly improved unless the state-of-the-art matrix multiplication algorithms can, and vice versa. Thus, this paper almost settles the time complexity of this problem.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Grant No. 61802069), Shanghai Sailing Program (Grant No. 18YF1401200) and Shanghai Science and Technology Commission (Grant No. 17JC1420200).

Appendix A. Omitted Proofs

A.1. Covariance error to projection error

Lemma 19 $\|A - \pi_B^k(A)\|_F^2 \le \|A - [A]_k\|_F^2 + 2k \cdot \|A^TA - B^TB\|_2$.

Proof For any x with $\|x\| = 1$, we have

$$\left| \|Ax\|^2 - \|Bx\|^2 \right| = \left| x^T(A^TA - B^TB)x \right| \le \|A^TA - B^TB\|_2 \tag{9}$$

Let u_i and w_i be the ith right singular vector of B and A respectively

$$\begin{aligned}
&\|A - \pi_B^k(A)\|_F^2 = \|A\|_F^2 - \|\pi_B^k(A)\|_F^2 \\
&= \|A\|_F^2 - \sum_{i=1}^{k} \|Au_i\|^2 \quad \text{Pathagorean theorem} \\
&\le \|A\|_F^2 - \sum_{i=1}^{k} \|Bu_i\|^2 + k \cdot \|A^TA - B^TB\|_2 \quad \text{by Eq. (9)} \\
&\le \|A\|_F^2 - \sum_{i=1}^{k} \|Bw_i\|^2 + k \cdot \|A^TA - B^TB\|_2 \quad \text{because } \sum_{i=1}^{k} \|Bw_i\|^2 \le \sum_{i=1}^{k} \|Bu_i\|^2 \\
&\le \|A\|_F^2 - \sum_{i=1}^{k} \|Aw_i\|^2 + 2k \cdot \|A^TA - B^TB\|_2 \quad \text{by Eq. (9)} \\
&= \|A - [A]_k\|_F^2 + 2k \cdot \|A^TA - B^TB\|_2.
\end{aligned}$$

■

A.2. Row sampling

Theorem 20 *For any $A \in \mathbb{R}^{n\times d}$ and $F > 0$, we sample each row A_i with probability $p_i \ge \frac{\|A_i\|^2}{\alpha^2 F}$; if it is sampled, scale it by $1/\sqrt{p_i}$. Let B be the (rescaled) sampled rows, then w.p. 0.99, $\|A^TA - B^TB\|_2 \le 10\alpha\sqrt{F}\|A\|_F$, and $\|B\|_F \le 10\|A\|_F$. The expected number of rows sampled is $O(\frac{\|A\|_F^2}{\alpha^2 F})$ if $p_i = O(\frac{\|A_i\|^2}{\alpha^2 F})$ for each i.*

Proof Since spectral norm is no larger than the Frobenius norm, it is sufficient to prove $\|A^TA - B^TB\|_F \le 10\alpha\sqrt{F}\|A\|_F$.

For each $j \in [n]$, let

$$x_j = \begin{cases} 1 & \text{if the } j\text{th rows of } A \text{ is sampled} \\ 0 & \text{otherwise.} \end{cases}$$

We have $(A^TA)_{i,j} = \sum_{t=1}^{n} a_{t,i}a_{t,j}$, while

$$(B^TB)_{i,j} = \sum_{t=1}^{n} \frac{x_t^2 \cdot a_{t,i}a_{t,j}}{p_t}.$$

So $\mathsf{E}[(B^TB)_{i,j}] = (A^TA)_{i,j}$. We also have

$$\mathsf{Var}[(B^TB)_{i,j}] = \mathsf{Var}\left[\sum_{t=1}^{n} \frac{x_t^2 \cdot a_{t,i}a_{t,j}}{p_t}\right] = \sum_{t=1}^{n} \frac{a_{t,i}^2 a_{t,j}^2 \cdot \mathsf{Var}\left[x_t^2\right]}{p_t^2} \leq \sum_{t=1}^{n} \frac{a_{t,i}^2 a_{t,j}^2}{p_t},$$

where we use the fact $\mathsf{Var}[x_t^2] = p_t(1-p_t) \leq p_t$. So we have

$$\mathsf{E}\left[\left((A^TA)_{i,j} - (B^TB)_{i,j}\right)^2\right] = \mathsf{Var}[(B^TB)_{i,j}] \leq \sum_{t=1}^{n} \frac{a_{t,i}^2 a_{t,j}^2}{p_t}.$$

Therefore,

$$\begin{aligned}
\mathsf{E}\left[\|A^TA - B^TB\|_F^2\right] &= \sum_{i,j} \mathsf{E}\left[\left((A^TA)_{i,j} - (B^TB)_{i,j}\right)^2\right] \\
&\leq \sum_{i,j}\sum_{t=1}^{n} \frac{a_{t,i}^2 a_{t,j}^2}{p_t} \\
&= \sum_{t=1}^{n} \frac{\|A_t\|^2\|A_t\|^2}{p_t} \\
&= \sum_{t=1}^{n} \alpha^2 F\|A_t\|^2 = \alpha^2 F\|A\|_F^2.
\end{aligned} \tag{10}$$

We adjust α by a constant, and using Markov's inequality

$$\mathsf{Pr}\left[\|A^TA - B^TB\|_F^2 \geq 100\alpha^2 F\|A\|_F^2\right] \leq 0.01,$$

which is equivalent to

$$\mathsf{Pr}\left[\|A^TA - B^TB\|_F \geq 10\alpha\sqrt{F}\|A\|_F\right] \leq 0.01.$$

The success probability can be boosted by a similar argument as in Drineas et al. (2006) via McDiarmid's inequality (see e.g. Boucheron et al. (2013)).

It is not hard to verify that

$$\mathsf{E}\left[\|B\|_F^2\right] = \|A\|_F^2.$$

So by another Markov inequality, we prove the second part. ■

References

Haim Avron, Vikas Sindhwani, and David Woodruff. Sketching structured matrices for faster nonlinear regression. In *NIPS*, 2013.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

Christos Boutsidis, Dan Garber, Zohar Karnin, and Edo Liberty. Online principal components analysis. In *SODA*. SIAM, 2015.

Christos Boutsidis, D Woodruff, and Peilin Zhong. Optimal principal component analysis in distributed and streaming models. In *STOC*, 2016.

Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Efficient second-order online kernel learning with adaptive embedding. In *NIPS*, 2017.

Flavio Chierichetti, Sreenivas Gollapudi, Ravi Kumar, Silvio Lattanzi, Rina Panigrahy, and David Woodruff. Algorithms for ℓ_p low rank approximation. In *ICML*, 2017.

Kenneth L Clarkson and David P Woodruff. Numerical linear algebra in the streaming model. In *STOC*. ACM, 2009.

Kenneth L Clarkson and David P Woodruff. Low rank approximation and regression in input sparsity time. In *STOC*, 2013.

Michael B Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *STOC*. ACM, 2015.

Michael B Cohen, Cameron Musco, and Christopher Musco. Input sparsity time low-rank approximation via ridge leverage score sampling. In *SODA*. SIAM, 2017.

Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast monte carlo algorithms for matrices i: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1): 132–157, 2006.

Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. In *SODA*. SIAM, 2013.

Mina Ghashami and Jeff M Phillips. Relative errors for deterministic low-rank matrix approximations. In *SODA*. SIAM, 2014.

Mina Ghashami, Jeff M Phillips, and Feifei Li. Continuous matrix approximation on distributed data. *Proceedings of the VLDB Endowment*, 7(10):809–820, 2014.

Mina Ghashami, Edo Liberty, and Jeff M Phillips. Efficient frequent directions algorithm for sparse matrices. In *KDD*, 2016.

Hao Huang and Shiva Prasad Kasiviswanathan. Streaming anomaly detection using randomized matrix sketching. *Proceedings of the VLDB Endowment*, 9(3):192–203, 2015.

Zengfeng Huang, Xuemin Lin, Wenjie Zhang, and Ying Zhang. Efficient matrix sketching over distributed data. In *Proceedings of PODS*. ACM, 2017.

William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

Ravi Kannan, Santosh Vempala, and David Woodruff. Principal component analysis and higher correlations for distributed data. In *COLT*, 2014.

Zohar Karnin and Edo Liberty. Online pca with spectral bounds. In *COLT*, 2015.

Young-Bum Kim, Karl Stratos, and Ruhi Sarikaya. Scalable semi-supervised query classification using matrix sketching. In *ACL*, 2016.

Cong Leng, Jiaxiang Wu, Jian Cheng, Xiao Bai, and Hanqing Lu. Online sketching hashing. In *IEEE CVPR*, 2015.

Yingyu Liang, Maria-Florina F Balcan, Vandana Kanchanapally, and David Woodruff. Improved distributed principal component analysis. In *NIPS*, 2014.

Edo Liberty. Simple and deterministic matrix sketching. In *KDD*. ACM, 2013.

Haipeng Luo, Alekh Agarwal, Nicolo Cesa-Bianchi, and John Langford. Efficient second order online learning by sketching. In *NIPS*, 2016.

Cameron Musco and Christopher Musco. Randomized block krylov methods for stronger and faster approximate singular value decomposition. In *NIPS*, 2015.

Cameron Musco and David Woodruff. Is input sparsity time possible for kernel low-rank approximation? In *NIPS*, 2017.

Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In *FOCS*, 2006.

Qiang Song, Jian Cheng, and Hanqing Lu. Incremental matrix factorization via feature space re-learning for recommender system. In *RecSys*. ACM, 2015.

Zhewei Wei, Xuancheng Liu, Feifei Li, Shuo Shang, Xiaoyong Du, and Ji-Rong Wen. Matrix sketching over sliding windows. In *SIGMOD*, 2016.

David Woodruff. Low rank approximation lower bounds in row-update streams. In *NIPS*, 2014a.

David Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014b.

Qiaomin Ye, Luo Luo, and Zhihua Zhang. Frequent direction algorithms for approximate matrix multiplication with applications in cca. In *AAAI*. AAAI Press, 2016.

Haida Zhang, Zengfeng Huang, Zhewei Wei, Wenjie Zhang, and Xuemin Lin. Tracking matrix approximation over distributed sliding windows. In *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*. IEEE, 2017.

Yuchen Zhang, Martin Wainwright, and Michael Jordan. Distributed estimation of generalized matrix rank: Efficient algorithms and lower bounds. In *ICML*, 2015.

Journal of Machine Learning Research 20 (2019) 1-31 Submitted 12/13; Revised 4/17; Published 4/19

Multi-class Heterogeneous Domain Adaptation

Joey Tianyi Zhou ZHOUTY@IHPC.A-STAR.EDU.SG
Institute of High Performance Computing (IHPC)
*Agency for Science, Technology and Research (A*STAR),*
Singapore 138632

Ivor W. Tsang IVOR.TSANG@UTS.EDU.AU
Centre for Artificial Intelligence
University of Technology Sydney
Australia

Sinno Jialin Pan SINNOPAN@NTU.EDU.SG
School of Computer Science and Engineering
Nanyang Technological University,
Singapore 639798

Mingkui Tan MINGKUITAN@SCUT.EDU.CN
School of Software Engineering
South China University of Technology,
China

Editor: Shie Mannor

Abstract

A crucial issue in heterogeneous domain adaptation (HDA) is the ability to learn a feature mapping between different types of features across domains. Inspired by language translation, a word translated from one language corresponds to only a few words in another language, we present an efficient method named *Sparse Heterogeneous Feature Representation (SHFR)* in this paper for multi-class HDA to learn a sparse feature transformation between domains with multiple classes. Specifically, we formulate the problem of learning the feature transformation as a compressed sensing problem by building multiple binary classifiers in the target domain as various measurement sensors, which are decomposed from the target multi-class classification problem. We show that the estimation error of the learned transformation decreases with the increasing number of binary classifiers. In other words, for adaptation across heterogeneous domains to be successful, it is necessary to construct a sufficient number of incoherent binary classifiers from the original multi-class classification problem. To achieve this, we propose to apply the error correcting output correcting (ECOC) scheme to generate incoherent classifiers. To speed up the learning of the feature transformation across domains, we apply an efficient batch-mode algorithm to solve the resultant nonnegative sparse recovery problem. Theoretically, we present a generalization error bound of our proposed HDA method under a multi-class setting. Lastly, we conduct extensive experiments on both synthetic and real-world datasets to demonstrate the superiority of our proposed method over existing state-of-the-art HDA methods in terms of prediction accuracy and training efficiency.

Keywords: Heterogeneous domain adaptation, multi-class classification, compressed sensing.

©2019 Joey Tianyi Zhou, Ivor W. Tsang, Sinno Jialin Pan, Mingkui Tan.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/13-580.html.

1. Introduction

In many applications, it is often expensive or time consuming to annotate sufficient labeled training data to build machine learning-based information systems. There may therefore be only a few labeled training data in the domain of interest but many labeled training data in another domain that is referred to as a source domain. Because of the differences between domains, a model trained on the labeled data in the source domain cannot be applied directly to the target domain. To address this problem, domain adaptation that aims to adapt a model from a source domain to a target domain with little or no additional human supervision has been proposed (Blitzer et al., 2006; Jiang and Zhai, 2007; Huang et al., 2007; Pan et al., 2008, 2011). Apart from thorough theoretical studies (Ben-David et al., 2006; Ben-David et al., 2010), domain adaptation techniques have been successfully applied to various applications, such as WiFi localization (Pan et al., 2008, 2011), Natural Language Processing applications (Daumé III, 2007; Jiang and Zhai, 2007), sentiment analysis (Blitzer et al., 2007b; Pan et al., 2010; Seah et al., 2011; Zhou et al., 2014, 2016a), object categorization (Duan et al., 2009; Gong et al., 2012; Hoffman et al., 2013), object detection (Donahue et al., 2013; Hoffman et al., 2014) and information retrieval (Zhou et al., 2016b, 2018).

Most existing domain adaptation works are based on the assumption that different domain data can be represented by the same feature space of the same dimensionality (Pan and Yang, 2010). However, this assumption may not hold in many real-world scenarios. For instance, in a cross-language (e.g., English/Spanish) text classification task (Prettenhofer and Stein, 2010), the feature spaces are referred to as vocabularies in different languages, and the features across domains are neither in the same feature space nor of the same dimensionality. To relax this assumption, there has been an increased focus on domain adaptation across heterogeneous feature spaces, which is referred to as Heterogeneous Domain Adaptation (HDA) (Dai et al., 2008; Yang et al., 2009). Besides cross-language text classification, many other real-world applications can be formulated as an HDA problem, such as an image classification task using auxiliary text data (Zhu et al., 2011), where the heterogeneous features are referred to as pixels and bags of textual words, respectively, image classification by using auxiliary images with different sets of features (Saenko et al., 2010; Kulis et al., 2011), where the heterogeneous features are referred to as different computer vision features, and so on.

Existing HDA approaches can be classified into two main categories. In the first category, two dense feature mappings $\mathbf{P}$ and $\mathbf{Q}$ can be learned to transform the source domain data $\mathbf{X}_S$ and target domain data $\mathbf{X}_T$, respectively, to a new latent common feature space such that the difference between the mapped domain data $\mathbf{P}\mathbf{X}_S$ and $\mathbf{Q}\mathbf{X}_T$ is reduced (Shi et al., 2010; Prettenhofer and Stein, 2010; Wang and Mahadevan, 2011; Duan et al., 2012). For example, Shi et al. (2010) proposed a heterogeneous spectral mapping (HeMap) method to learn dense orthogonal mappings based on spectral embedding without using any label information. The resultant optimization problem is a standard eigen-decomposition problem. However, this approach is known to suffer from the scalability issue. Wang and Mahadevan (2011) proposed a manifold alignment method known as DAMA to align heterogeneous features into a latent space based on manifold regularization. Unfortunately, DAMA only works on data that has a manifold structure. This limits its transferability to data on which the manifold assumption is satisfied. Furthermore, the use of DAMA results in a generalized eigen-decomposition problem of a series of matrices whose sizes depend on the dimensionality of the data. Duan et al. (2012) proposed a Heterogeneous Feature Augmentation (HFA) method to augment heterogeneous features with homogeneous common features learned using a maximum-

margin approach from both source and target domains. However, the proposed model results in an expensive semidefinite program (SDP) problem.

In the second category, a dense feature mapping $\mathbf{G}$ can be learned to transform heterogeneous data from one domain to the other domain directly so that the difference between $\mathbf{G}\mathbf{X}_S$ and $\mathbf{X}_T$ can be minimized or the alignment between $\mathbf{G}\mathbf{X}_S$ and $\mathbf{X}_T$ can be maximized. Harel and Mannor (2011) proposed a method named MOMAP to learn dense rotation matrices to match data distributions between the source and target domains. The resultant optimization problem is solved through singular value decomposition (SVD) for each class in an independent way. Kulis et al. (2011) proposed an asymmetric regularized cross-domain transformation (ARC-t) method to learn asymmetric transformation across domains based on metric learning. Similar to DAMA, ARC-t also utilizes the label information to construct the similarity and dissimilarity constraints between instances from the source and target domains, respectively. However, the computational complexities of ARC-t and its kernelized version depend quadratically on the feature dimensions and the data size, respectively.

Though most existing HDA methods have shown promising results, they still suffer from the following three major limitations.

1. **Dense Feature Mappings.** Existing methods tend to recover dense feature mappings, which is not tenable without sufficient constraints. As we will show, dense feature mappings may inevitably lose interpretability and contain significant amount of noise, which may affect the classification performance.

2. **Multi-class problem.** To address multi-class classification problems, most existing HDA methods (Dai et al., 2008; Wang and Mahadevan, 2011; Duan et al., 2012) simply adopt the one-vs-all strategy to learn multiple binary classifiers independently. Unfortunately, this strategy fails to fully explore the underlying structure among multiple classes. Consequently, this heuristic scheme is unable to guarantee good performance for multi-class HDA problems.

3. **High Computational Cost.** Since the size of the feature mapping $\mathbf{G}$ scales with the product of the dimensionality of the source and target domains, the computational cost to estimate the mapping is extremely high, especially for high-dimensional source and target data, e.g. MOMAP, HeMap, DAMA, ARC-t. To address this computational issue, Duan et al. (2012) and Kulis et al. (2011) proposed kernelized versions to learn the feature mapping. However, the kernelized methods still suffer from high computational cost on large-scale data in terms of the number of data instances.

2. Motivations and Contributions

In this work, we propose a **Sparse Heterogeneous Feature Representation** (SHFR) approach to address the above issues under following assumptions.

2.1 Motivations

1. **Sparse feature representation:** The feature mapping $\mathbf{G}$ between two domains is row-sparse, i.e., each target domain feature can be represented by a small subset of the source domain features.

2. **Class-invariance transformation:** Instances belonging to different classes share the same feature mapping $\mathbf{G}$.

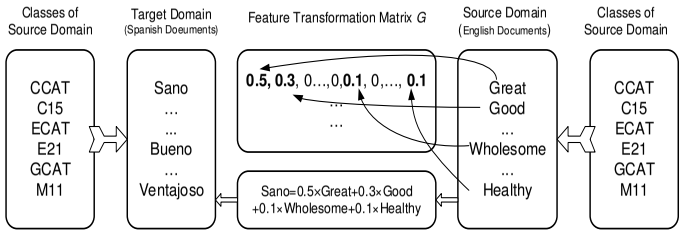


Figure 1: Illustration of the sparse feature representation matrix.

The above two assumptions are common in real-world multilingual text categorization applications. Recall that the sparsity assumption of the feature mapping across domains implies that each feature in one domain can only be represented by a small subset of features in another domain. Here, we still use the multi-language (i.e., English/Spanish) text classification problem as a motivating example. Typically, the word "Sano" in Spanish has a similar meaning to the words "Great", "Good", "Wholesome", and "healthy" in English, but not to all English words. Therefore, by assuming that the feature mapping across domains is linear, a feature or word in the Spanish domain can be represented by a linear combination of only several features or words in the English domain. As illustrated in Figure 2.1, the sparse matrix $\mathbf{G}$ denotes the feature mapping from the English domain to the Spanish domain. Based on the sparse matrix $\mathbf{G}$, the word "Sano" in Spanish can be represented sparsely by only four words in English as "Sano$''$ $=$ $0.5\times$ "Great$''+0.3\times$ "Good$''+0.1\times$ "Wholesome$''+0.1\times$ "Healthy$''$. This sparsity characteristic, which also facilitates a significant reduction in computational cost on very high-dimensional data, has not been explored in existing HDA methods. The feature mapping of the word "Sano" is invariant for different classes, for example, "CCAT" and "C15". To encode the *class-invariance* property into the feature mapping, we propose to learn a common $\mathbf{G}$ underlying all the classes which has a similar spirit to multi-task feature learning (Argyriou et al., 2007) and domain-invariant feature learning (Gong et al., 2013). We first decompose a multi-class classification problem into multiple binary classification tasks, and then jointly optimize all the binary classification tasks and the feature mapping $\mathbf{G}$, such that $\mathbf{G}$ is class-invariant to all the binary classification tasks.

2.2 Our Contributions

Based on these observations, we propose a new approach to learning a feature mapping for HDA with application to cross-lingual multi-class text categorization. To estimate this a feature mapping, we leverage the weight vectors of the binary classifiers learned in the source and target domains. We then formulate the problem of learning the feature mapping between domains as a Compressed Sensing (CS) problem, which aims to recover a sparse signal from a small number of measurements with respect to an underdetermined linear system by taking advantage of the sparseness or compressibility of the signal (Donoho, 2006; Candès et al., 2006). The main contributions of this paper are as follows:

1. We propose a **Sparse Heterogeneous Feature Representation** (SHFR) approach to learn a sparse transformation for HDA by exploring the common underlying structures between multiple classes.

2. Our major finding is that, based on the CS theory, a sparse feature mapping can be learned if and only if a sufficient number of classifiers are provided. Based on this analysis, we propose to generate sufficient classifiers under the **Error Correcting Output Correcting** (ECOC) scheme to accurately estimate the sparse feature mapping.

3. To reduce the high computational cost, we propose a **batch-mode** pursuit algorithm to efficiently solve the resultant nonnegative lasso problem of SHFR. The computational cost of SHFR is independent of the data size and the dimensionality of the source domain data, and scales linearly only with size of the dimensionality of the target domain data.

4. This paper is the first to present a **generalization error bound** for multi-class HDA which is dependent on the number of tasks as well as the degree of sparsity of the feature mapping $\mathbf{G}$.

The remainder of the paper is organized as follows. In Section 3, we first present how to formulate an HDA problem as a compressed sensing problem, and describe the detail of the proposed method, SHFR. We then theoretically analyze a generalization error bound for SHFR in Section 4, and develop a batch-mode algorithm in Section 5 to efficiently solve the resultant compressed sensing problem. In Section 6, we conduct extensive experiments on both toy and real-world datasets to demonstrate the effectiveness and efficiency of SHFR. We conclude this paper and discuss future work in Section 7.

3. Sparse Heterogeneous Feature Representation

We study the HDA problem with one source domain and one target domain under the multi-class setting. Let $\{(\mathbf{x}_{S_i}, y_{S_i})\}_{i=1}^{n_S}$ denote a set of labeled training instances of the source domain, where $\mathbf{x}_{S_i} \in \mathbb{R}^{d_S}$ denotes the i-th instance, and $y_{S_i} \in \{1, 2, \cdots, c\}$ denotes its label. Let $\{(\mathbf{x}_{T_i}, y_{T_i})\}_{i=1}^{n_T}$ be a set of labeled training instances of the target domain, where $n_T \ll n_S$, $\mathbf{x}_{T_i} \in \mathbb{R}^{d_T}$, and $y_{T_i} \in \{1, 2, \cdots, c\}$.

Given a binary task $t \in \{1, 2, \cdots, n_c\}$, we assume that the classifiers for both source and target domain are linear, which can be written as $f^t(\mathbf{x}) = \mathbf{w}^{t\top}\mathbf{x}$, where $\mathbf{w}^t$ is the weight vector of the t-th classifier. Since there are sufficient labeled data in the source domain, i.e., $\{(\mathbf{x}_{S_i}, y_{S_i})\}_{i=1}^{n_S}$, we can learn a robust set of weight vectors $\{\mathbf{w}_S^t\}_{t=1}^{n_c}$ for the source classifiers of the binary tasks decomposed from the multi-class classification problem. Similarly, we can build a corresponding set of weight vectors $\{\mathbf{w}_T^t\}_{t=1}^{n_c}$ with limited labeled data $\{(\mathbf{x}_{T_i}, y_{T_i})\}_{i=1}^{n_T}$ for the target domain.

3.1 Feature Mapping for HDA

Recall that, for HDA problems, the feature dimensions of the source and target domains may not be equal, i.e., $d_S \neq d_T$. To make effective learning across heterogeneous domains possible, a transformation matrix $\mathbf{G} \in \mathbb{R}^{d_T \times d_S}$ was introduced in ARC-t (Kulis et al., 2011; Saenko et al., 2010) to learn the similarity $\mathbf{x}_{T_i}^\top \mathbf{G}\mathbf{x}_{S_i}$ between a source instance $\mathbf{x}_{S_i} \in \mathbb{R}^{d_S}$ and a target instance $\mathbf{x}_{T_i} \in \mathbb{R}^{d_T}$. Data points can be transformed from the source feature space to the target feature

space via $\mathbf{x}_{T_i} = \mathbf{G}\mathbf{x}_{S_i}$, or equivalently, the target domain data can be mapped into the source domain via $\mathbf{G}^\top$ (Saenko et al., 2010). To learn the transformation matrix $\mathbf{G}$, corresponding pairs across domains are usually supposed to be given in advance (Qi et al., 2011) or generated based on the label information in both the source and target domains (Saenko et al., 2010). However, in practice, instance-correspondences between domains are usually missing, or the constructed instance correspondences using label information are not precise. Therefore, instead of learning the transformation by using the input raw data directly, we adapt an idea from a multi-task learning method (Ando and Zhang, 2005)[1], and propose to learn the feature mapping across heterogeneous features based on the source and target predictive structures, i.e., $\{\mathbf{w}_S^t\}$'s and $\{\mathbf{w}_T^t\}$'s.

We can either learn the transformation matrix $\mathbf{G} \in \mathbb{R}^{d_T \times d_S}$ by maximizing the dependency between the transformed weight vectors of source classifiers and the weight vectors of target classifiers via

$$\max_{\mathbf{G}} \mathbf{w}_T^{t\top} \mathbf{G}\mathbf{w}_S^t,$$

or by minimizing the distance between these two vectors via

$$\min_{\mathbf{G}} \|\mathbf{w}_T^t - \mathbf{G}\mathbf{w}_S^t\|.$$

For simplicity in theoretical analysis, we adopt the latter objective to learn the transformation $\mathbf{G}$. Given a binary task $t \in \{1, \cdots, n_c\}$ and a transformation $\mathbf{G} \in \mathbb{R}^{d_T \times d_S}$, the relationship between the weight vectors of the source and target classifiers can be modeled as follows,

$$\mathbf{w}_T^t - \mathbf{G}\mathbf{w}_S^t = \mathbf{w}_\Delta^t,$$

where $\mathbf{w}_\Delta^t$ is referred to as a "delta" weight vector, and its ℓ_2-norm $\|\mathbf{w}_\Delta^t\|_2$ is used to measure the difference between the target weight vector $\mathbf{w}_T^t$ and the transformed source weight vector $\mathbf{G}\mathbf{w}_S^t$. Note that the relationship is linear. This is because, in this paper, we focus on the application of HDA on cross-language text categorization, where using a linear form to model the relationship of predictive structures between domains is more suitable for high-dimensional text data. For modeling a nonlinear relationship between the source and target predictive structures, which may be more useful in other application areas, e.g., image classification or speech recognition, it is possible to apply an explicit nonlinear feature mapping $\phi(\mathbf{x})$ to both the source and target domain data before training the weight vectors for each domain.

3.2 Proposed Formulation

In order to use the robust source domain weight vector $\mathbf{w}_S^t$ to make predictions on the target domain data, the difference between the source and target domains after transformation should be minimized. We therefore propose to learn the transformation $\mathbf{G}$ by minimizing $\|\mathbf{w}_\Delta^t\|_2$. As mentioned in Section 2, the feature mapping $\mathbf{G}$ should be class-invariant and sparse, and for multi-language text classification and many other real-world applications, the feature mapping $\mathbf{G}$ between feature spaces should be nonnegative. The motivation of adding the nonnegative constraint on the feature mapping $\mathbf{G}$ is similar to that of applying nonnegative matrix factorization (NMF) (Lee and Seung,

1. Multi-task learning aims to simultaneously learn the classifiers for multiple tasks by sharing parameters between all the tasks, where each task has only limited labeled data. In contrast, the goal of HDA is to learn a classifier only for a target domain with limited target label data by leveraging the models learned from the source domain.

2001) to text mining. Here, we aim to approximate "input vectors", i.e., target features, by a nonnegative linear combination of a set of nonnegative "basis vectors", i.e., source features, through the mapping $\mathbf{G}$. This is also similar to multiple kernel learning (MKL) (Lanckriet et al., 2004; Gönen and Alpaydın, 2011), where the target kernel is represented by the nonnegative linear combination of base kernels. In the context of multi-lingual text classification, the nonnegative entries in $\mathbf{G}$ can be deemed as explicit "soft" global co-occurrences between the source and target wordbooks, while without the nonnegative constraint on $\mathbf{G}$, it is difficult to interpret the physical or semantic meaning behind $\mathbf{G}$.

Let $\mathbf{G} = [\mathbf{g}_1, \mathbf{g}_2, \cdots, \mathbf{g}_{d_T}]^\top$, by imposing the ℓ_1-regularization and the nonnegative constraint on $\mathbf{g}_i$, the learning of $\mathbf{G}$ can be formulated as the following nonnegative LASSO problem (Yuan and Lin, 2007; Slawski and Hein, 2011),

$$\begin{aligned} \min_{\mathbf{G}} \quad & \frac{1}{n_c}\sum_{t=1}^{n_c}\|\mathbf{w}_T^t - \mathbf{G}\mathbf{w}_S^t\|_2^2 + \sum_{i}^{d_T}\lambda_i\|\mathbf{g}_i\|_1, \\ s.t. \quad & \mathbf{g}_i \succeq \mathbf{0}, \end{aligned} \tag{1}$$

where $\lambda_i > 0$ is the trade-off parameter for the regularization term on each $\mathbf{g}_i$. In (1), the first term in the objective aims to minimize the difference between $\mathbf{w}_T^t$ and $\mathbf{G}\mathbf{w}_S^t$ over all the n_c tasks, the second term in the objective is to enforce the sparsity on each row of $\mathbf{G}$, and the constraints are used to preserve nonnegative linear relations between the source and target predictive structures.

Note that in practice, once the source domain classifiers in terms of the weight vectors $\{\mathbf{w}_S^t\}_{t=1}^{n_c}$ are learned offline, the source domain training data can be discarded. For a new domain of interest, i.e., the target domain, the weight vectors $\{\mathbf{w}_T^t\}_{t=1}^{n_c}$ can first be learned with a few labeled training data, and then $\mathbf{G}$ can be learned with $\{\mathbf{w}_S^t\}_{t=1}^{n_c}$ instead of the original source domain training data, which significantly reduces the learning complexity. This learning scheme is typically different from most of the existing HDA methods that require the original source domain training data to be available to learn the feature mapping across domains. More discussion on the complexity issue can be found in Section 5.1. As $\mathbf{G}$ is learned through all the binary classification tasks, knowledge can be transferred from the easy tasks to the difficult ones. In Section 4.2, we theoretically show that as long as a significant number of base binary tasks are not badly designed, the generalization error of the target classification model is guaranteed to be small.

To solve the optimization problem (1), we can first rewrite it in the following equivalent form,

$$\begin{aligned} \min_{\mathbf{g}_i} \quad & \frac{1}{n_c}\sum_{t=1}^{n_c}\sum_{i=1}^{d_T}(w_{T_i}^t - \mathbf{w}_S^{t\top}\mathbf{g}_i)^2 + \sum_{i}^{d_T}\lambda_i\|\mathbf{g}_i\|_1, \\ s.t. \quad & \mathbf{g}_i \succeq \mathbf{0}, \end{aligned} \tag{2}$$

where $w_{T_i}^t$ is the i-th element of the vector $\mathbf{w}_T^t$. If we exchange the summation sequences in the first term, the above optimization problem can be reformulated as the following CS problem,

$$\begin{aligned} \min_{\mathbf{g}_i} \quad & \sum_{i}^{d_T}\left(\frac{1}{n_c}\|\mathbf{b}_i - \mathbf{D}\mathbf{g}_i\|_2^2 + \lambda_i\|\mathbf{g}_i\|_1\right), \\ s.t. \quad & \mathbf{g}_i \succeq \mathbf{0}, \end{aligned} \tag{3}$$

where $\mathbf{b}_i$ is the concatenated row vector containing $\{w_{T_i}^t\}_{t=1}^{n_c}$, and $\mathbf{D} = [\mathbf{w}_S^1\ \mathbf{w}_S^2\ \cdots\ \mathbf{w}_S^{n_c}]^\top \in \mathbb{R}^{n_c \times d_S}$.

This type of CS problem has been well studied, and a number of algorithms have been proposed to solve it, such as nonnegative least squares (NNLS) (Cantarella and Piatek, 2004), accelerated proximal gradient algorithm (APG) (Toh and Yun, 2010), and orthogonal matching pursuit (OMP) (Zhang, 2009). In this work, we propose an efficient batch-mode algorithm to solve it, which is introduced in detail in Section 5. After the sparse transformation $\mathbf{G}$ has been learned, we can reuse the source domain classifiers to predict its label for any unseen test data $\mathbf{x}_T^*$ from the target domain via

$$y_T^* = F(\{(\mathbf{G}\mathbf{w}_S^t)^\top\mathbf{x}_T^*\}_{t=1}^{n_c}),$$

where $F(\cdot)$ is a decision function that combines the predictive results of all the n_c source classifiers to make a final prediction.

4. Generalization Error Bound for HDA

In the past decade, theoretical analysis for domain adaptation has been widely studied (Blitzer et al., 2007a; Mansour et al., 2009; Ben-David et al., 2010), and has focused on understanding conditions and assumptions of domain adaptation. However, most previous theoretical works were restricted by the assumption that the data from the source and target domains are represented in the same feature space and therefore cannot be applied directly to analyze the generalization error of our proposed method, SHFR, under the HDA setting. In this section, we analyze the theoretical guarantee of SHFR in detail. In contrast to existing generalization error analysis for homogeneous domain adaptation, we derive the generalization error bound for SHFR mainly from the perspectives of ECOC and compressed sensing.

4.1 Estimation Error Bound for Sparse Feature Mapping

Recall that the optimization problem (3) is composed of d_T lasso problems. In general, we have $n_c < d_S$ for relatively high-dimensional data, therefore (3) is an underdetermined linear system (Donoho, 2006). However, based on the compressed sensing theory, if $\mathbf{g}_i$ is sparse, it is possible to obtain a solution with sufficient measurements and a matrix $\mathbf{D}$ that satisfies certain conditions (Donoho, 2006; Candès et al., 2006). In particular, one such condition is the sparse Riesz condition or RIP condition, which requires that any two columns of $\mathbf{D}$ should be as perfectly incoherent as possible (Donoho, 2006; Zhang and Huang, 2008; Candès et al., 2006).

For simplicity in presentation, let k_i denote the number of non-sparse entries of $\mathbf{g}_i$ (or non-sparsity degree), and $\widehat{\mathbf{g}}_i$ denote an estimator of $\mathbf{g}_i$. According to Theorem 3 and Remark 4 in (Zhang and Huang, 2008), under some restricted conditions, the estimation error $||\mathbf{g}_i - \widehat{\mathbf{g}}_i||_2$ for each independent subproblem can be bounded by $O\left(\sqrt{\frac{k_i \log d_S}{n_c}}\right)$, where d_S denotes the dimensionality of the source domain data and n_c is the number of tasks. Hence, we can obtain the following lemma.

Lemma 1 *Under the sparse Riesz condition, the estimation error $\|\Delta\mathbf{G}_{CS}\|_F = \|\mathbf{G} - \widehat{\mathbf{G}}\|_F$ in (1) is bounded by $O\left(d_T\sqrt{\frac{k\log d_S}{n_c}}\right)$, where k, d_S, d_T, n_c, $\widehat{\mathbf{G}}$ denote the largest row non-sparsity degree of $\mathbf{G}$, the dimensionality of the source and target domain data, the number of tasks, and the estimator of $\mathbf{G}$, respectively.*

Note that besides the sparse Riesz condition, Lemma 1 also holds under some other restricted conditions, such as the RIP condition (Zhang and Huang, 2008; Candès et al., 2006). Both the sparse Riesz condition and the RIP condition require that any two columns of $\mathbf{D}$ should be as incoherent as possible (Donoho, 2006; Zhang and Huang, 2008; Candès et al., 2006). However, although the RIP condition for randomly designed matrices has been thoroughly investigated (Baraniuk et al., 2008), it is more difficult to investigate it for deterministically designed matrices in general. In contrast, the sparse Riesz condition for a deterministically designed matrix holds with high probability if the following condition is satisfied (see Proposition 1 in (Zhang and Huang, 2008)):

$$\max_{|A|=k} \inf_{\alpha \geq 1} \left\{ \sum_{j \in A} \left(\sum_{i \in A, i \neq j} |\rho_{ji}|^{\alpha/(\alpha-1)} \right)^{\alpha-1} \right\}^{1/\alpha} \leq \delta < 1,$$

where k denotes the non-sparsity degree and $\rho_{ji} = \mathbf{d}_j' \mathbf{d}_i / n_c$.

Remark 2 *Note that ρ_{ji} can be deemed as the correlation between the columns $\mathbf{d}_j$ and $\mathbf{d}_i$. Therefore, the sparse Riesz condition for a deterministically designed matrix can be evaluated by the correlations between the columns of the dictionary matrix. The smaller the summary of correlations $\sum |\rho_{ji}|$ (or δ) is, the more likely it is that the sparse Riesz condition for $\mathbf{D}$ will hold.*

According to Lemma 1 and Remark 2, to reduce the reconstruction error of the feature mapping $\mathbf{G}$, it is necessary to 1) construct as many classifiers as possible to increase the number of measurements, i.e., n_c needs to be large enough, and 2) ensure the columns of $\mathbf{D}$ constructed from the classifiers are incoherent, i.e., $\mathbf{D}$ is incoherent. For a multi-class classification problem with labels $\{1, 2, \cdots, c\}$, c binary classifiers can be generated using the one-vs-all strategy (Dietterich and Bakiri, 1995). However, when c is small, this strategy is not able to generate sufficient binary classifiers. Alternatively, the one-vs-one strategy may be used to generate $c(c-1)/2$ binary classifiers. The classifiers generated in this way may have large redundancy, however, i.e., some classifiers may be highly correlated to each other. To address this issue, we propose to use the Error Correcting Output Codes (ECOC) scheme (Dietterich and Bakiri, 1995; Zhou et al.) to generate sufficient binary classifiers for estimating the feature mapping $\mathbf{G}$. As will be shown empirically in Section 6.5, greater incoherence for $\mathbf{D}$ over the one-vs-all or one-vs-one strategy can be guaranteed with a proper design of the ECOC coding matrix.

4.2 Generalization Error Analysis of HDA Based on ECOC

By using the ECOC scheme, the multi-class hypothesis h is composed of a set of binary classifiers $\{\widehat{\mathbf{G}}\mathbf{w}_S^t\}_{t=1}^{n_c}$. However, in contrast to the one-vs-all scheme, ECOC is more general and consists of two steps: encoding, i.e., to design a coding matrix M to induce binary classifiers, and decoding, i.e., to combine the results of all the binary classifiers to make predictions based on the coding matrix. In order to analyze the generalization error bound of SHFR for multi-class HDA based on ECOC, we need to first analyze the principles of ECOC for multi-class classification.

As discussed in Dietterich and Bakiri (1995); Allwein et al. (2001), the prediction performance of a multi-class classification model learned based on the ECOC scheme depends on the design of the coding matrix. Given a coding matrix $M = \{-1, 0, +1\}^{c \times n_c}$, each class is associated with a row of the matrix M, and each binary classifier is associated with a column of the matrix M. If

an entry $M(i, j) = +1$, it implies that in the classifier f^j, class i is considered as a positive class, while if $M(i, j) = -1$, it implies that in the classifier f^j, the class i is considered as a negative class, otherwise, the class i is not taken into account for training and testing. In this way, either the one-vs-all scheme or the one-vs-one scheme can be considered as a special case of the ECOC scheme.

As pointed out by Dietterich and Bakiri (1995), a coding matrix is good if each codeword, i.e., each row, is well-separated in the Hamming distance from each of the other codewords, and each bit-position function f^k, i.e., each column, is uncorrelated with the functions to be learned for the other bit positions f^j, $j \neq k$. The power of a coding matrix M for multi-class classification can be measured by the minimum Hamming distance between any pair of its codewords (Allwein et al., 2001), denoted by $\Delta_{min}(M)$. For instance, for the one-vs-all code, $\Delta_{min}(M) = 2$, and for the one-vs-one code, $\Delta_{min}(M) = \left(\binom{c}{2} - 1 \right)/2 + 1$. For a random matrix with components chosen over $\{-1, 0, 1\}$, the expected value of $\Delta_{min}(M)$ for any distinct pair of rows is $n_c/2$.

Here, we first follow the proof procedure on error analysis for multi-class classification described in Allwein et al. (2001) to analyze the condition of a testing error by the ECOC decoding. After that, we derive the generalization error bound for the proposed SHFR with the ECOC scheme.

Given a coding matrix $M \in \{-1, 0, +1\}^{c \times n_c}$, each row corresponds to a class and each column corresponds to a binary task. For some $t \in \{1, 2, \cdots, n_c\}$, we denote $\mathbf{w}_S^t$ and h_S^t as the source domain weight vector and the hypothesis of the t-th task defined by the coding matrix M, respectively. We denote $\widehat{\mathbf{G}}\mathbf{w}_S^t$ and h_T^t as the estimated target domain weight vector and hypothesis of the t-th task, respectively. We also define the loss function,

$$L(z) = (1 - z)_+ = \max\{1 - z, 0\}, \tag{4}$$

in terms of the margin z. For example, given a target domain instance $\mathbf{x}_{T_i}$ with its label y_{T_i}, the corresponding margin $z_{T_i}^t$ for t-th task is defined as

$$z_{T_i}^t = M(y_{T_i}, t) h_T^t(\mathbf{x}_{T_i}) = M(y_{T_i}, t)((\widehat{\mathbf{G}}\mathbf{w}_S^t)^\top \mathbf{x}_{T_i}). \tag{5}$$

For instance, SVM seeks to minimize the margin-based loss under L_2 norm regularization as follows,

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|_2^2 + C \sum_{i=1}^{m} (1 - y_i \mathbf{w}^\top \mathbf{x}_i)_+, \tag{6}$$

where we denote margin $z = y_i \mathbf{w}^\top \mathbf{x}_i$. When the prediction is correct and $M(y_{T_i}, t) \neq 0$ for $\mathbf{x}_{T_i}$, then the margin $z_{T_i}^t > 0$, which results in $0 \leq L(z) < 1$; when the prediction is wrong and $M(y_{T_i}, t) \neq 0$ for $\mathbf{x}_{T_i}$, then the margin $z_{T_i}^t < 0$, which results in $L(z) > 1$; when $M(y_{T_i}, t) = 0$, then the margin $\mathbf{z}_{T_i}^t = 0$, which results in $L(z) = 1$. We define the $L(z)$ loss-based decoding scheme for prediction as follows:

$$d_L(M(r, :), h(\mathbf{x}_{T_i})) = \sum_{t=1}^{n_c} L(M(r, t) h_T^t(\mathbf{x}_{T_i})). \tag{7}$$

The predicted label $\hat{y} \in \{1, 2, \cdots, c\}$ is given by $\hat{y} = \arg\min_r d_L(M(r), h(\mathbf{x}_{T_i}))$.

To understand the error correction ability of ECOC, we first define the distance between the codes in any distinct pair of rows, $M(r_1,:)$ and $M(r_2,:)$, in the coding matrix M as

$$d(M(r_1,:),M(r_2,:)) = \sum_{s=1}^{n_c} d(M(r_1,s),M(r_2,s)), \tag{8}$$

where $d(M(r_1,s),M(r_2,s)) = \frac{1-sign(M(r_1,s)M(r_2,s))}{2}$.

We define $\rho = \min_{r_1 \neq r_2} d(M(r_1,:),M(r_2,:))$ as the minimum distance between any two rows in the coding matrix M, which is pre-calculated before the training.

Proposition 3 *Given a coding matrix $M \in \{-1,0,+1\}^{c\times n_c}$, a set of hypothesis outputs $h^1(\mathbf{x}),\cdots,h^{n_c}(\mathbf{x})$ on a test instance $\mathbf{x}$ generated by n_c base classifiers, and a convex loss function $L(z)$ in terms of the margin z, if $\mathbf{x}$ is misclassified by the loss-based ECOC decoding, then*

$$\sum_{t=1}^{n_c} L(M(y,t)h^t(\mathbf{x})) \geq \rho L(0), \tag{9}$$

where $M(y,t)$ denotes the corresponding label generated by the coding matrix M of ground truth. In other words, the loss of the vector of predictions $f(\mathbf{x})$ is greater than $\rho L(0)$.

Proof Suppose that the loss-based ECOC decoding incorrectly classifies a test instance $\mathbf{x}$ with the ground-truth label y. Then should then exist a label $r \neq y$ such that

$$d_L(M(y,:),h(\mathbf{x})) \geq d_L(M(r,:),h(\mathbf{x})).$$

By using the definitions of the loss function and margin introduced in (4) and (5), we denote

$$z^t = M(y,t)h^t(\mathbf{x}),$$
$$z'^t = M(r,t)h^t(\mathbf{x}),$$

By using the definitions of d_L in (7), we have

$$\sum_{t=1}^{n_c} L(z^t) \geq \sum_{t=1}^{n_c} L(z'^t). \tag{10}$$

Let $S_\Delta = \{t : M(r,t) \neq M(y,t) \wedge M(r,t) \neq 0 \wedge M(y,t) \neq 0\}$ be the set of columns of M whose r-th and y-th row entries are different and nonzero, and let $S_0 = \{t : M(r,t) = 0 \vee M(y,t) = 0\}$ be the set of columns whose r-th or y-th row entry is zero.

If $t \notin S_\Delta \cup S_0$, then $z^t = z'^t$, and (10) implies that

$$\sum_{t\in S_\Delta\cup S_0} L(z^t) \geq \sum_{t\in S_\Delta\cup S_0} L(z'^t),$$

which, in turn, implies that

$$\begin{aligned}\sum_{t=1}^{n_c} L(z^t) \geq & \sum_{t\in S_\Delta\cup S_0} L(z^t)\\ \geq & \frac{1}{2}\sum_{t\in S_\Delta\cup S_0}(L(z^t)+L(z'^t))\\ = & \frac{1}{2}\sum_{t\in S_\Delta}(L(z^t)+L(z'^t)) + \frac{1}{2}\sum_{t\in S_0}(L(z^t)+L(z'^t)).\end{aligned} \tag{11}$$

If $t \in S_\Delta$, then $z'^t = -z^t$ and $(L(-z^t)+L(z^t))/2 \geq L(0)$ due to convexity. If $t \in S_0$, then either $z^t = 0$ or $z'^t = 0$. Hence, we obtain that $L(z'^t)+L(z^t) \geq L(0)$, and the second term of (11) is at least $L(0)|S_0|/2$. Therefore,

$$\begin{aligned}\sum_{t=1}^{n_c} L(z^t) \geq & \frac{1}{2}\sum_{t\in S_\Delta}(L(z^t)+L(z'^t)) + \frac{1}{2}\sum_{t\in S_0}(L(z^t)+L(z'^t))\\ \geq & L(0)\left(|S_\Delta| + \frac{|S_0|}{2}\right)\\ = & L(0)d(M(r,:),M(y,:))\\ \geq & \rho L(0),\end{aligned} \tag{12}$$

where (12) is obtained from the fact that

$$\begin{aligned}d(M(r_1,s),M(r_2,s)) &= \frac{1-sign(M(r_1,s)M(r_2,s))}{2}\\ &= \begin{cases}0 & if\ \ M(r,t)=M(y,t)\wedge M(r,t)\neq 0\wedge M(y,t)\neq 0\\ \frac{1}{2} & if\ \ M(r,t)=0\vee M(y,t)=0\\ 1 & if\ \ M(r,t)\neq M(y,t)\wedge M(r,t)\neq 0\wedge M(y,t)\neq 0.\end{cases}\end{aligned}$$

This completes the proof. ■

Remark 4 *From Proposition 3, it can be seen that misclassification on a test instance $(\mathbf{x},y)$ implies that $\sum_{t=1}^{n_c} L(M(y,t)h^t(\mathbf{x})) \geq \rho L(0)$. In other words, the prediction codes are not required to be exactly the same as ground-truth codes for all the base classifications. As long as the loss is smaller than $\rho L(0)$, ECOC can rectify the error committed by some base classifiers, and is still able to make an accurate prediction. This error-correcting ability is very important, especially when the labeled data is insufficient in the target domain. This proposition holds for any convex margin-based loss function $L(z)$. In this paper, we use hinge loss defined in (4), then $L(0) = 1$.*

Theorem 5 *Let ϵ be the averaged loss of the target domain hypotheses $h_T^1,\ldots,h_T^{n_c}$ on the target test data $\{\{(\mathbf{x}_{T_i},M(y_{T_i},t))\}_{i=n_T+1}^{n_{Te}}\}_{t=1}^{n_c}$ with respect to the coding matrix $M \in \{-1,0,+1\}^{c\times n_c}$, where c is the cardinality of the label set. Let the convex loss function be $L(z) = |1-z|_+$. The multi-class HDA generalization error of SHFR using sparse random encoding and loss-based decoding is then at most $\frac{\epsilon n_c}{\rho}$.*

Proof According to Proposition 3, the loss for any misclassified testing instance $\mathbf{x}_T$ satisfies

$$\rho L(0) \leq \sum_{t=1}^{n_c} L(M(y_T,t)h_T^t(\mathbf{x}_T)).$$

Let a be the number of incorrect predictions for a test sample of size n_{Te}, then we can obtain the following inequality,

$$a\rho L(0) \leq a\sum_{t=1}^{n_c} L(M(y_T,t)h_T^t(\mathbf{x}_T)) \leq n_{Te}\sum_{t=1}^{n_c} L(M(y_T,t)h_T^t(\mathbf{x}_T)). \tag{13}$$

Then we have

$$a \leq \frac{n_{Te}\sum_{t=1}^{n_c} L(M(y_T,t)h_T^t(\mathbf{x}_T))}{\rho L(0)} = \frac{n_{Te}\epsilon n_c}{\rho}, \tag{14}$$

where $\epsilon = \frac{\sum_{t=1}^{n_c} L(M(y_T,t)h_T^t(\mathbf{x}_T))}{n_c}$ and $L(0) = |1-0|_+ = 1$.
Therefore, the generalization error rate is bounded by $\frac{\epsilon n_c}{\rho L(0)}$. The proof is completed. ■

Remark 6 *The above error bound is usually small in the setting studied in this paper for the following reasons. When sufficient classifiers are generated using ECOC and* $\mathbf{G}$ *satisfies the sparsity assumption, the reconstruction error is small and bounded according to the compressive sensing theory. On the other hand, the loss* $L(M(y_T,t)h_T^t(\mathbf{x}_T))$ *for some* $\{h_T^t\}$*s, which rely on* $\mathbf{G}$*, may be large due to there being limited target domain data. The data-independent term* n_c *is usually predefined and the minimum distance* ρ *is determined correspondingly, as* ρ *monotonically increases with* n_c *according to (8). Given the predefined ECOC coding matrix, as long as the averaged loss* ϵ *is small, according to Theorem 5, target classifier is still able to make a precise prediction. In other words, SHFR is only unable to achieve satisfactory performance in terms of classification accuracy if most or all of the binary classifiers generated using ECOC trained on the target domain data are poorly designed.*

In summary, we show that the multi-class HDA generalization error of SHFR based on the ECOC scheme has a linear relationship with the average loss over all generated binary classifiers when using random encoding and loss-based decoding methods. In addition to theoretical analysis, García-Pedrajas and Ortiz-Boyer (2011) conducted comprehensive experiments and concluded that ECOC and one-vs-one are superior to other multi-class classification schemes, and are the best choices for either powerful learners or simple learners. Furthermore, Montazer et al. (2012) claimed that, of several multi-class classification schemes, sparse ECOC is better than dense ECOC, while dense ECOC is better than one-vs-one, and one-vs-all is the worst. Ghani (2000) proved a theoretical bound that a randomly-construct binary matrix is not well row-separated with probability at most $1/c^4$, where c is the number of classes. Therefore, in our experiments, we adopt the sparse and random ECOC scheme for multi-class HDA problems.

4.3 Robust Transformation Learning using ECOC

The robustness of ECOC is another important motivation for using ECOC in SHFR. Recall that the error-correcting codes can be viewed as a compact form of voting, and a certain number of incorrect votes can be corrected through the corrected votes (Dietterich and Bakiri, 1995). Given a total number of T classifiers, the voting-based methods guarantee to make a correct decision as long as there are $\lfloor \frac{T}{2} + 1 \rfloor$ correct classifiers (Dietterich, 2000), where $\lfloor\ \rfloor$ is the *floor* function that returns the largest integer not greater than the input of the function. In other words, even though there are misclassifications due to incorrect base classifiers, good performance can still be achieved in terms of the final classification accuracy by using the ECOC scheme. It has been shown in previous research that although bit error is unavoidable in real-world applications, it is still possible to make correct decisions using the error-correcting codes with a sufficient number of good learners (Ghani, 2000).

This property is particularly important for the proposed multi-class HDA method, SHFR, since some of the learned binary classifiers in the target domain may not be accurate due to insufficient label information. SHFR with ECOC is able to learn a robust transformation matrix $\mathbf{G}$ even when some target binary classifiers are not precise. The robustness of SHFR with ECOC will be verified empirically in Section 6, where we demonstrate that even with inaccurate target classifiers, the class-invariant $\mathbf{G}$ learned by SHFR with the ECOC scheme can still dramatically enhance prediction accuracy for the target domain. By combining all the above theoretical analyses, we can conclude that the generalization error of our proposed method SHFR for multi-class HDA problems is small and bounded with a proper ECOC coding matrix design.

5. Efficient Batch Decoding for SHFR

In this section, we present an algorithm to solve the proposed optimization problem (3) of SHFR in detail. Recall that problem (3) contains d_T LASSO problems with nonnegative constraints, which can be solved by adapting existing algorithms like NNLS solvers (Cantarella and Piatek, 2004; Slawski and Hein, 2011) or LASSO solvers, e.g., APG (Toh and Yun, 2010) and OMP (Zhang, 2009). However, the time complexity of these methods is $O(n_c d_S)$ for each LASSO problem, which makes problem (3) intractable when the source domain data is of very high dimensionality and the number of LASSO problems d_T is very large.

Recall that each of the optimization problems (3) can be cast as the following problem

$$\begin{aligned} \min_{\mathbf{g}_i} \quad & \frac{1}{n_c}\|\mathbf{b}_i - \mathbf{D}\mathbf{g}_i\|_2^2 + \lambda_i\|\mathbf{g}_i\|_1, \\ s.t. \quad & \mathbf{g}_i \succeq \mathbf{0}, \end{aligned} \tag{15}$$

where $\mathbf{b}_i \in \mathbb{R}^{n_c \times 1}$ is the concatenated row vector containing $w_{T_i}^t$ for all the n_c tasks, and $\mathbf{D} = [\mathbf{w}_S^1\ \mathbf{w}_S^2\ \cdots\ \mathbf{w}_S^{n_c}]^\top \in \mathbb{R}^{n_c \times d_S}$. For simplicity and without loss of clarity, we drop the subscript i from $\mathbf{g}_i$ and $\mathbf{b}_i$ in our discussion. Given any $\mathbf{g}$, let

$$\boldsymbol{\xi} = \mathbf{b} - \mathbf{D}\mathbf{g} \tag{16}$$

be the residual. Problem (15) can then be equivalently reformulated as the following problem

$$\begin{aligned} \min_{\mathbf{g}\succeq\mathbf{0},\boldsymbol{\xi}} \quad & \lambda||\mathbf{g}||_1 + \frac{1}{2}||\boldsymbol{\xi}||^2, \\ s.t. \quad & \boldsymbol{\xi} = \mathbf{b} - \mathbf{D}\mathbf{g}. \end{aligned} \tag{17}$$

Solving this problem directly can be very expensive when the source domain data is of very high dimensionality. Following Tan et al. (2015a), we propose to solve problem (15) via a matching pursuit method, called matching pursuit LASSO (MPL), as shown in Algorithm 1.

In Algorithm 1, the residual $\boldsymbol{\xi}^0$ is initialized by $\boldsymbol{\xi}^0 = \mathbf{b}$ since $\mathbf{g}^0 = \mathbf{0}$. At the t-th step, we first choose a set of B components in $\mathbf{g}$ with the B largest values in $\mathbf{D}^\top\boldsymbol{\xi}_t$, and merge their indices $\mathcal{J}_t$ into the active set $\mathcal{I}_t$. We then solve a reduced problem whose scale is only $O(n_c|\mathcal{I}_t|)$:

$$\begin{aligned} \min_{\mathbf{g}\succeq\mathbf{0},\boldsymbol{\xi}} \quad & \lambda||\mathbf{g}||_1 + \frac{1}{2}||\boldsymbol{\xi}||^2, \\ s.t. \quad & \boldsymbol{\xi} = \mathbf{b} - \mathbf{D}\mathbf{g}, \mathbf{g}_{\mathcal{I}_t^c} = \mathbf{0}, \end{aligned} \tag{18}$$

Algorithm 1 Matching pursuit LASSO for solving the nonnegative LASSO problem.

Initialize the residual $\boldsymbol{\xi}^0 = \mathbf{b}$, $\mathcal{I}_0 = \emptyset$, and let $t = 1$.
1: Compute $\mathbf{q} = \mathbf{D}^\top \boldsymbol{\xi}^{t-1}$, choose the B largest q_j, and record their indices by $\mathcal{J}_t$.
2: Let $\mathcal{I}_t = \mathcal{I}_{t-1} \cup \mathcal{J}_t$.
3: Let $\mathbf{g}_{\mathcal{I}_t^c} = \mathbf{0}$, and solve the subproblem (18) to update $\mathbf{g}_{\mathcal{I}_t}$.
4: Compute $\boldsymbol{\xi}^t = \mathbf{b} - \mathbf{D}\mathbf{g}_{\mathcal{I}_t}$.
5: Terminate if the stopping condition is achieved. Otherwise, let $t = t + 1$ and go to step 1.

where $\mathcal{I}_t$ denotes the index set of the selected columns from $\mathbf{D}$ and $\mathcal{I}_t^c$ is the corresponding complementary set of $\mathcal{I}_t$. This subproblem can be solved efficiently using a projected proximal gradient (PG) method. Note that the number B in Step 1 is a small positive integer, e.g., $B = 1$. Due to the nonnegative constraints $\mathbf{g} \succeq 0$, we find the elements with the B largest value in $\mathbf{D}^\top \boldsymbol{\xi}$ rather than $|\mathbf{D}^\top \boldsymbol{\xi}|$.

Batch-mode MPL. In Algorithm 1, solving each subproblem has a complexity of $O(kn_c)$, where k denotes the degree of the sparsity, namely $k = |\mathcal{I}_t|$, while computing $\mathbf{D}^\top \boldsymbol{\xi}$ has a complexity of $O(d_S n_c)$. Note that problem (3) has d_T sub-problems as in (15). Therefore, computing $\mathbf{D}^\top \boldsymbol{\xi}$ will dominate the whole complexity of the decoding if both n_c and d_T are very large.

To address the above computational burden imposed by $\mathbf{D}^\top \boldsymbol{\xi}$, we propose a batch-mode MPL (BMPL) algorithm. Note that $\mathbf{D}^\top \boldsymbol{\xi} = \mathbf{D}^\top(\mathbf{b} - \mathbf{D}\mathbf{g}) = \mathbf{D}^\top \mathbf{b} - \mathbf{D}^\top \mathbf{D}\mathbf{g}$. We can pre-compute $\boldsymbol{\beta} = \mathbf{D}^\top \mathbf{b}$ and $\mathbf{Q} = \mathbf{D}^\top \mathbf{D}$, and store them in the memory. Since $\mathbf{g}_{\mathcal{I}_t^c} = \mathbf{0}$ for the t-th iteration, we have $\mathbf{D}^\top \boldsymbol{\xi}^t = \mathbf{D}^\top \mathbf{b} - [\mathbf{D}^\top \mathbf{D}_{\mathcal{I}_t}]\mathbf{g}_{\mathcal{I}_t} = \boldsymbol{\beta} - \mathbf{Q}_{\mathcal{I}_t}\mathbf{g}_{\mathcal{I}_t}$, where $\mathbf{Q}_{\mathcal{I}_t}$ denotes the columns of $\mathbf{Q}$ indexed by $\mathcal{I}_t$. The computation cost for $\mathbf{D}^\top \boldsymbol{\xi}$ is thus reduced to $O(kn_c)$, which saves considerable computation cost when $k \ll d_S$. Note that $\boldsymbol{\beta}$ and $\mathbf{Q}$ are shared by all the LASSO tasks, thus they only need to be calculated once. With this strategy, the overall computational cost of solving (3) when d_S is large is significantly reduced to $O(d_T n_c k)$ (Tan et al., 2015a,b).

5.1 Complexity Comparison

We use linear SVMs to build the base classifiers for the source and target domains, which are obtained using the Liblinear solver (Fan et al., 2008) and pre-trained offline. The learning of $\mathbf{G}$ is not related to the number of training instances. According to the analysis in Section 5, the computational cost of SHFR is $O(d_T n_c k)$, where the sparsity degree $k \ll d_S$. Compared to state-of-the-art HDA methods, our proposed method is much more efficient. The MOMAP method transforms the problem into c separate singular value decomposition (SVD) problems resulting in complexity of $O(4d_S^2 d_T + 8d_S d_T^2 + 9d_T^3)$. The HeMap method is based on standard eigen-decomposition, whose complexity is $O(n_S + n_T)^3$. The ARC-t method solves an optimization problem that contains $n_S n_T$ constraints by applying an alternating projection method (e.g., Bregman's algorithm (Censor and Zenios, 1997)), resulting in complexity of $d_S d_T n_S n_T$. The HFA method adopts an alternating projection method to solve a semidefinite program (SDP), where the transformation matrix to be learned is in $\mathbb{R}^{(n_S+n_T)\times(n_S+n_T)}$, resulting in time complexity bounded by $O(n_S + n_T)^3$. Therefore, ARC-t and HFA perform inefficiently when the data size is large. DAMA first constructs a series of combinatorial Laplacian matrices in $\mathbb{R}^{(d_S+d_T)\times(d_S+d_T)}$, and then solves a generalized eigen-decomposition problem of time complexity bounded by $O(d_S + d_T)^3$. DAMA is therefore

Table 1: Complexity comparison of HDA methods.

Methods	Complexity
MOMAP (Harel and Mannor, 2011)	$O(4d_S^2 d_T + 8d_S d_T^2 + 9d_T^3)$
HeMap (Shi et al., 2010)	$O(n_S + n_T)^3$
DAMA (Wang and Mahadevan, 2011)	$O(d_S + d_T)^3$
ARC-t (Kulis et al., 2011)	$O(d_S d_T n_S n_T)$
HFA (Duan et al., 2012)	$O(n_S + n_T)^3$
SHFR (our proposed method)	$O(d_T n_c k)$

computationally very expensive when the data dimensionality is high. The comparison in terms of time complexity between different methods is summarized in Table 1.

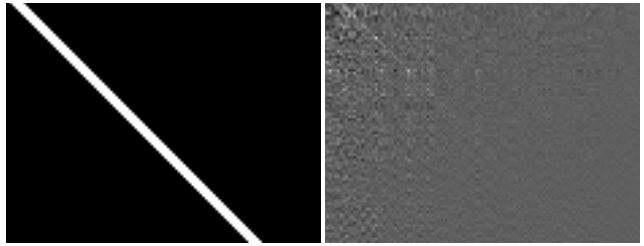
6. Experiments

In this section, we conduct experiments on both toy and real-world datasets to verify the effectiveness and efficiency of our proposed method SHFR for multi-class HDA. The parameter settings of our proposed method and other baseline methods are as follows. For ARC-t and HFA, which require the use of kernel functions to measure data similarity, we use the RBF kernel for learning the transformation. Parameter tuning is still an open research issue, as cross-validation is not applicable to HDA problems due to the limited size of labeled data in the target domain. We therefore tune the parameters of the comparison methods on a predefined range and report their best results on the test data. We use linear SVMs as the base classifiers, the regularization parameter C of which is cross-validated from the range of $\{0.01, 0.1, 1, 10, 100\}$ on the source domain data, and we use it for all the comparison methods. For our proposed method SHFR, we empirically set the maximum number of iterations to 20, $\lambda = 0.01$ and $B = 2$ in the BGMP algorithm, and generate the ECOC as long as possible for each dataset.

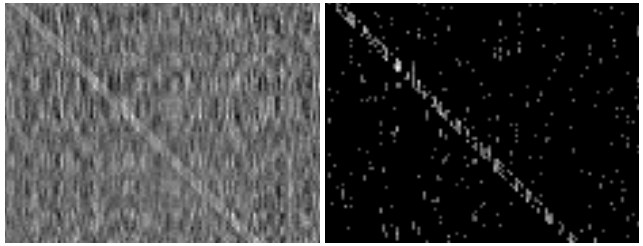
6.1 Experiments on Synthetic Datasets

We first compare the performance of different HDA methods in terms of recovering a ground-truth feature mapping $\mathbf{G}$ on a 20-class toy dataset. To generate the toy dataset, we first randomly generate 150 instances of 150 features for each class from different Gaussian distributions to form a source domain $\mathbf{X}_S \in \mathbb{R}^{150\times 3{,}000}$. We then construct the ground-truth sparse feature mapping $\mathbf{G} \in \mathbb{R}^{100\times 150}$ using the following method: for each row i, we set $\mathbf{G}_{ij} = 1/5$, where $j = i, i+1, \cdots, i+5$, and $\mathbf{G}_{ij} = 0$ otherwise. This generation of $\mathbf{G}$ implies that each target domain feature is represented by five source domain features. The ground-truth feature mapping is displayed in Figure 2(a), where the dark area represents the zero entries and the bright area denotes nonzero values of $\mathbf{G}$. Lastly, we construct the target domain data $\mathbf{X}_T \in \mathbb{R}^{100\times 3{,}000}$ by using $\mathbf{X}_T = \mathbf{G}\mathbf{X}_S$. When conducting the experiment, we randomly select five instances per class from the target domain data $\mathbf{X}_T$ as the labeled training data, and apply different HDA methods on them together with all 3,000 source domain labeled data to recover the feature mapping $\mathbf{G}$.

In this experiment, the HDA methods, DAMA and ARC-t, are adopted as the baselines. For ease of comparison, we present the recovered matrix $\mathbf{G}$ for each of the three methods in Figures 2(b)-2(d). From Figure 2(b), we observe that DAMA fails to recover the structure of $\mathbf{G}$, while ARC-t



(a) Ground Truth (b) DAMA



(c) ARC-t (d) SHFR

Figure 2: Illustration of the recovered feature mappings using different methods on the toy data. Black represents 0 and white represents 1.

shows better performance as can be seen in Figure 2(c). However, the $\mathbf{G}$s recovered by these two methods are not sparse. In contrast, SHFR can perfectly recover the sparse $\mathbf{G}$ with little noise as shown in Figure 2(d). These experimental results demonstrate that by explicitly adding sparsity constraints, SHFR is able to recover the feature mapping $\mathbf{G}$ more accurately.

6.2 Experiments on Real-world Datasets

We conduct experiments on three real-world datasets, Multilingual Reuters Collection, BBC Collection, and Cross-lingual Sentiment Dataset, to verify the effectiveness and efficiency of SHFR. The reported results are averaged over 10 independent data-split procedures.

6.2.1 DATASETS AND EXPERIMENTAL SETUP

Multilingual Reuters Collection[2] is a text dataset with over 11,000 news articles from six categories in five languages, i.e., English, French, German, Italian and Spanish, which are represented by a bag-of-words weighted by TF-IDF. Following the setting in Duan et al. (2012), we use Spanish as the target domain and the other four languages as source domains, which results in four multi-class HDA problems. For each class, we randomly select 100 instances from the source domain

2. http://multilingreuters.iit.nrc.ca/ReutersMultiLingualMultiView.htm

and 10 instances from the target domain for training. We randomly select 10,000 instances from the target domain as the test data. Note that the original data is of very high dimensionality, and the baseline methods cannot handle such high-dimensional features. To conduct comparison experiments, we first perform PCA with 60% energy preserved on the TF-IDF features. We then obtain 1,131 features for the the English documents, 1,230 features for the French documents, 1,417 features for the German documents, 1,041 features for the Italian documents, and 807 features for the Spanish documents. In contrast to the baseline methods, we use original features for SHFR since it can efficiently handle high-dimensional data.

BBC Collection[3] was collected for multi-view learning, and each instance is represented by three views. It was constructed from a single-view BBC corpora by splitting news articles into related "views" of text. We consider **View 3** as the target domain, and **View 1** and **View 2** as source domains. Similar to the pre-processing on the Reuters dataset, we perform PCA on the original data to reduce dimensions such that other baselines can be applied. The reduced dimensions for **View 1**, **View 2** and **View 3** are 203, 205 and 418, respectively. We randomly select 70% source domain instances and 10 target domain instances for each class for training. The remaining target domain instances are considered to be the test data.

Cross-lingual Sentiment Dataset[4] consists of Amazon product reviews of three product categories: books, DVDs and music. These reviews are written in four languages: English, German, French, and Japanese. We treat English reviews as the source domain data and the other language reviews as the target data domain data. After performing PCA, the reviews are of 715, 929, 964, and 874 features for English, German, French and Japanese, respectively. We randomly select 1,500 source domain instances and 10 target domain instances per class for training, and use the rest 5,970 target domain instances for testing.

6.2.2 OVERALL COMPARISON RESULTS

Comparison results between SHFR and other baselines on the three real-world datasets are reported in Tables 2-4. From the tables, we observe that the SVMs conducted on a small number of target domain data only, denoted by SVM-T, using either one-vs-one or one-vs-all strategy, perform the worst on average. Moreover, the results of SVM-T using one-vs-one and one-vs-all, respectively, are not consistent. For instance, on the BBC dataset in Table 3, SVM-T using the one-vs-all strategy performs much better than SVM-T using the one-vs-one strategy, while on the sentiment dataset in Table 4, SVM-T using the one-vs-one strategy performs much better than SVM-T using the one-vs-all strategy. The reason is that the size of the labeled training data is too limited to train a precise and stable classifier in the target domain. Compared to one-vs-all and one-vs-one, SVM-T with ECOC shows much more stable performance and superior results on most problems, which is because of the error correcting ability of ECOC when the labeled data is limited. The performance in terms of classification accuracy of the HDA baseline methods, DAMA, ARC-t and HFA, are comparable on the three datasets except for the BBC dataset, where DAMA performs much worse than the other two methods. This may be because the performance of DAMA is sensitive to the intrinsic manifold structure of the data. If the manifold assumption does not hold on the data, the performance of DAMA drops significantly. Our proposed SHFR method using either the one-vs-one or ECOC scheme performs best on the three datasets. SHFR can further improve the performance

3. http://mlg.ucd.ie/datasets/segment.html
4. http://www.uni-weimar.de/cms/medien/webis/research/corpora/corpus-webis-cls-10.html

Table 2: Multilingual Reuters Collection: comparison results in terms of classification acc (%). Results of SHFR are significantly better than the other baselines, judged by the t-test with a significance level at 0.05.

Source Domain	SVM-T (1 vs all)	SVM-T (1 vs 1)	SVM-T (ECOC)	DAMA	ARC-t	HFA	SHFR (1vs1)	SHFR (ECOC)
English	65.40±4.45	65.21±7.85	68.01±2.48	63.42±2.62	66.72±2.27	68.82±1.68	69.45±1.56	**72.82±1.08**
French				65.12±1.28	67.67±2.05	68.14±1.71	70.77±1.57	**74.01±1.25**
German				66.98±2.45	68.15±1.72	68.42±1.57	71.62±1.69	**74.15±1.14**
Italian				67.56±2.23	67.47±2.32	69.59±2.51	70.75±1.54	**73.35±1.31**

Table 3: BBC Collection: comparison results in terms of classification acc (%). Results of SHFR are significantly better than the other baselines, judged by the t-test with a significance level at 0.05.

Source Domain	SVM-T (1 vs all)	SVM-T (1 vs 1)	SVM-T (ECOC)	DAMA	ARC-t	HFA	SHFR (1vs1)	SHFR (ECOC)
View 1	73.35±4.98	68.78±11.7	76.45±5.47	67.42±2.25	75.78±2.69	72.45±10.65	89.81±1.20	**90.57±1.49**
View 2				66.87±1.73	74.53±2.48	73.75±8.06	88.92±1.67	**91.85±0.96**

Table 4: Cross-lingual Sentiment Dataset: comparison results in terms of classification acc. (%). Results of SHFR are significantly better than the other baselines, judged by the t-test with a significance level at 0.05.

Target Domain	SVM-T (1 vs all)	SVM-T (1 vs 1)	SVM-T (ECOC)	DAMA	ARC-t	HFA	SHFR (1vs1)	SHFR (ECOC)
French	48.40±3.45	58.30±5.01	57.45±3.17	55.18±3.46	53.46±5.18	56.46±3.42	60.80±3.08	**62.12±2.61**
German	49.78±5.12	61.62±6.31	62.18±4.51	56.60±3.72	57.29±2.17	55.14±3.15	63.85±3.46	**65.57±2.32**
Japanese	48.25±6.34	56.13±4.81	57.27±2.61	53.56±2.67	55.75±3.02	55.02±4.67	59.67±3.75	**62.58±2.86**

in terms of classification accuracy, compared to one-vs-one, using the ECOC scheme. The superior performance benefits from both the global transformation of $\mathbf{G}$ and the error correcting ability of ECOC. As discussed in Section 4.1, the recovered feature mapping $\mathbf{G}$ tends to be more accurate with more constructed binary tasks.

6.3 Impact on Training Sample Size of the Target Domain

We verify the impact of the size of labeled training sample in the target domain to the overall HDA performance in terms of classification accuracy. We vary the number of target domain training instances from five to 20. In this experiment, we only report the results on the Reuters dataset, where we use English as the source domain and Spanish as the target domain. The experimental results are shown in Figure 3. From the figure, we observe that SHFR consistently outperforms the baseline methods under different numbers of labeled training instance in the target domain. In particular, SHFR shows significantly better performance than the baseline methods when the size of the target domain labeled data is smaller than 10.

Based on Theorem 5, the overall generalization error of SHFR depends on the accuracy of the binary classifiers generated based on ECOC. To empirically demonstrate this point, we report the

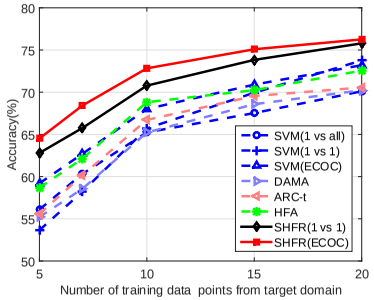
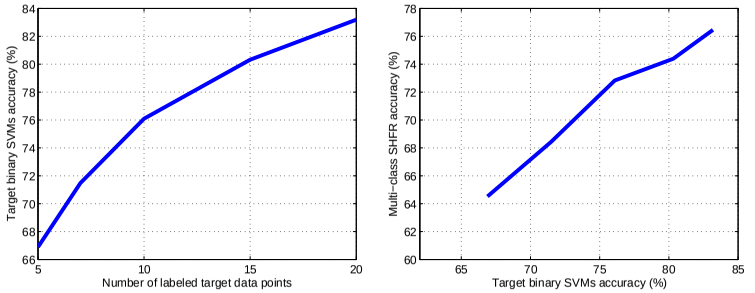


Figure 3: Comparison of different HDA methods under varying size of target labeled data.



(a) Averaged binary accuracy v.s. target data size. (b) Multiclass accuracy v.s. averaged binary accuracy.

Figure 4: Relationships between binary base classifiers and multi-class SHFR (ECOC).

averaged accuracy of all binary classifiers generated based on ECOC on the Reuters dataset under varying numbers of labeled training instances in the target domain in Figure 4(a). We also report the overall multi-class prediction accuracy versus the averaged accuracy of all binary classifiers in Figure 4(b). From the figures, we see that more labeled target data leads to more precise target binary classifiers, and thus leads to improvement in multi-class HDA classification in terms of accuracy. In particular, we observe from Figure 4(b) that the multi-class classification accuracy is proportional to the averaged binary task accuracy. The reason is that more precise target binary classifiers generally induce better alignment between the target and source classifiers.

6.4 Impact on Dimensionality of the Target Domain

In this experiment, we aim to compare the performance of different HDA methods under varying dimensionality of the target domain data. Here, we only report experimental results on the English-Spanish HDA task on the Reuters dataset. We first select 5,000 of the most-frequent features in the English domain as the source domain. We vary the dimensionality of the Spanish domain, i.e.,

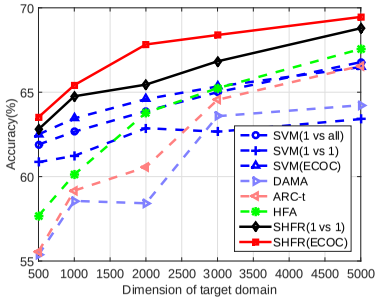


Figure 5: Comparison of different HDA methods under varying dimensions of target labeled data.

Table 5: Comparison in correlation $\sum_{ji} |\rho_{ji}|$.

Coding Scheme	English	French	German	Italian
1 vs all	0.8532 ± 0.1358	0.7557 ± 0.1289	0.2738 ± 0.0245	0.9331 ± 0.5701
1 vs 1	0.3374 ± 0.0290	0.2509 ± 0.0576	0.1168 ± 0.0110	0.3322 ± 0.1679
ECOC	0.1349 ± 0.0142	0.1478 ± 0.0225	0.0732 ± 0.0145	0.1725 ± 0.1047

the target domain, in the range of $\{500, 1000, 2000, 3000, 5000\}$ by selecting the most-frequent features of this domain. The results are shown in the Figure 5, from which we observe that SHFR consistently outperforms other baselines, especially when the dimensionality of the target domain data is low.

6.5 Incoherent Dictionary $\mathbf{D}$ Construction

In this experiment, we adopt the summation of absolute value of all pairs of columns in the dictionary $\mathbf{D}$ to evaluate $\mathbf{D}$'s coherence degree, i.e., $\sum_{ji} |\rho_{ji}|$. In other words, the smaller the value of $\sum_{ji} |\rho_{ji}|$ is, the more incoherent the dictionary is (Zhang and Huang, 2008). The results of different coding schemes on the Reuters dataset are summarized in Table 5. From the table, we observe that ECOC can be used to significantly reduce the coherence degree to enable better construction of the dictionary for the formulated compressed sensing problem.

6.6 Error Correction through Learning a Global $\mathbf{G}$

As discussed in Section 4, the weight vectors of the binary classifiers constructed in the target domain, i.e., $\{\mathbf{w}_T\}$s, may be unreliable due to the lack of target labeled data, which may affect the estimation of $\mathbf{G}$. To verify that SHFR can correct the bias of some binary classifiers, we conduct comparison experiments between SVM-T and SHFR in terms of classification accuracy on each binary task on the Reuters dataset. The results are shown in Table 6, where each column corresponds to a binary task, indexed by $k \in \{1, \cdots, 15\}$. From the table, we observe that the predictions of SVM-T on some binary tasks are inaccurate due to the limited number of labeled data. The accuracy is even below 50% (numbers in boldface). However, by learning the transformation $\mathbf{G}$ jointly among all the binary tasks, we are able to reduce the bias of the weak binary classifiers,

Table 6: Comparison results on each binary task in terms of classification accuracy (%).

Binary Classifiers	1	2	3	4	5	6	7	8
SVM-T	74.42	67.98	85.67	82.53	**43.32**	75.95	66.30	73.92
SHFR	65.39	84.81	82.45	94.28	80.76	76.75	79.74	84.63
Difference	-9.03	16.82	-3.22	11.75	37.44	0.80	13.44	10.70
Binary Classifiers	9	10	11	12	13	14	15	
SVM-T	**42.78**	76.56	78.32	65.43	72.74	**46.56**	**49.71**	
SHFR	84.45	70.07	90.35	81.35	83.97	80.75	86.43	
Difference	41.67	-6.49	12.03	15.92	11.23	34.20	36.72	

and increase accuracy by more than 30% on average. Experiments on the other two datasets return similar results, where the performance in terms of accuracy of the binary classifiers obtained through $\mathbf{G}$ is increased by 26.53% on the BBC dataset and 3.01% on the Sentiment dataset on average, compared to the results generated by SVM-T.

The experimental results in Table 6 also verify why SHFR outperforms ARC-t in the experiments shown in Tables 2-4 and Figure 3. ARC-t aims to align the target data with the source data via a transformation $\mathbf{G}$ that is learned from similarity and dissimilarity constraints. The constraints are constructed from a large number of source domain labeled data and a few target domain labeled data. In the multi-class setting, the number of similarity constraints to be constructed when the number of classes is large is much smaller than the number of dissimilarity constraints due to the limited number of target domain labeled data. In contrast to ARC-t, SHFR tries to align the target classifiers with the source classifiers using transformation $\mathbf{G}$, which is shared by all the induced binary tasks or classifiers. In other words, $\mathbf{G}$ is estimated through all classifiers. By borrowing the idea from multi-task feature learning which jointly optimizes all classifiers to learn a global feature transformation $\mathbf{G}$, is still possible to estimate a stable and precise $\mathbf{G}$ for multi-class HDA. Furthermore, as discussed in Section 4.3, ECOC with a well designed coding matrix is powerful for correcting error, as shown in Table 6.

6.7 Impact of Dichotomizer Size on Classification Error

As proven in Section 4, the generalization error of SHFR for multi-class HDA depends on the averaged loss over all the tasks, which relies on two factors: the number of classifiers (or measurements) and the degree of sparsity on $\mathbf{G}$. When $\mathbf{G}$ is sparse and the constructed binary classifiers are sufficient, it is possible to recover a precise $\mathbf{G}$ by using the dictionary constructed by $\mathbf{w}_S$. To demonstrate how the generalization error of SHFR changes under varying numbers of classifiers or dichotomizers, we conduct experiments on the Reuters dataset.

Experimental results are showed in Figure 6. From this figure we observe that when more dichotomizers are constructed, the predictions are more accurate in the target domain. This verifies that having more dichotomizers provides more information to recover the feature mapping $\mathbf{G}$. Furthermore, the standard deviation decreases with the increasing number of dichotomizers; however, as observed from the figure, multi-class accuracy no longer increases when the number of classifiers reaches 31. There are two reasons for this observation: 1) the redundancy in information among the constructed binary classifiers prevents improvement in estimating $\mathbf{G}$, and 2) according to García-Pedrajas and Ortiz-Boyer (2011), when there are many dichotomizers, the minimum distance in ECOC becomes small, which decreases the error correction ability. In general, SHFR obtains bet-

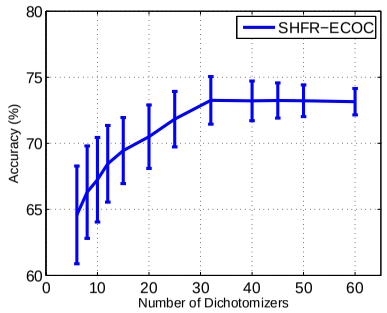


Figure 6: Accuracy vs dichotomizer (or task) size.

Table 7: Impact of Nonnegative Constraints and L_1 Norm Regularization: comparison between the results of different solvers in terms of classification accuracy (%). Results of SHFR are significantly better than the other baselines except for the cases denoted with *, judged by the t-test with a significance level at 0.05.

Dataset	Domain	SHFR-LASSO	SHFR-NNLS	SHFR
Reuters	English	67.08±1.63	71.47±1.56	**72.82±1.08**
	French	67.17±1.26	72.08±1.19	**74.01±1.25**
	German	69.56±1.28	73.62±1.75*	**74.15±1.14**
	Italian	68.19±2.51	71.98±1.78	**73.35±1.31**
BBC	View 1	84.72±2.18	88.66±1.29	**90.57±1.49**
	View 2	87.06±2.01	90.25±1.43	**91.85±0.96**
Sentiment	French	56.12±2.30	60.17±2.05	**62.12±2.61**
	German	58.83±2.24	63.47±1.49	**65.57±2.32**
	Japanese	56.72±1.81	61.51 ±1.76*	**62.58±2.86**

ter and more stable performance in terms of classification accuracy with an increasing number of dichotomizers.

6.8 Impacts of Nonnegative Constraints and ℓ_1 Norm Regularization

As discussed in Section 4, similar to the motivation of NMF, the nonnegative constraints on $\mathbf{G}$ are to approximate the "input vectors" (i.e., target features) by nonnegative linear combinations of nonnegative "basis vectors" (i.e., source features). The idea of nonnegative linear relationship can also be found in multiple kernel learning, where the target kernel is approximated by the nonnegative linear combination of base kernels. To verify the impact of the nonnegative constraints on $\mathbf{G}$, we add a baseline for comparison by moving the nonnegative constraints from the the optimization problem of SHFR, which is denoted by SHFR-LASSO. In this case, the model is reduced to a typical lasso problem.

Regarding the ℓ_1 norm regularization term in SHFR, note that some studies in the literature have shown that Non-Negative Least Squares (NNLS) can indeed achieve sparse recovery of nonnegative signals in a noiseless setting (Bruckstein et al., 2008; Donoho and Tanner, 2010; Wang and Tang, 2009; Wang et al., 2011). However, the noiseless assumption does not hold in practice. As mentioned by Slawski et al. (2013), apart from sign constraints, NNLS only consists of a fitting term, and thus is prone to overfitting, especially when the training data is high-dimensional. The ℓ_1-norm regularizer is therefore necessary to prevent over-adaptation to noisy data and enforce desired structural properties of the solution, which is similar to the idea of adding sparsity to the vector of coefficients of a predictive model (Slawski et al., 2013). In our proposed optimization problem (1), it is insufficient to achieve desirable solution of $\mathbf{G}$ without the ℓ_1-norm term in the objective since both $\mathbf{w}_S^t$ and $\mathbf{w}_T^t$ are of high dimensionality, and some of $\{\mathbf{w}_T^t\}$'s are not precise. As stated in Theorem 1 in Slawski and Hein (2011), exact sparsity, i.e., ℓ_1-norm, is not needed when the matrix satisfies the self-regularizing property, which is not applicable to our optimization problem as $\mathbf{G}$ does not have this property. Using the ℓ_1-norm ensures that a more sparse solution can be achieved by using nonnegative constraints. To verify the impact of the ℓ_1-norm on $\mathbf{G}$, we add another baseline for comparison by moving the ℓ_1-norm from the the optimization problem of SHFR, which is denoted by SHFR-NNLS. In this case, the model is reduced to an NNLS problem. In this paper, we adopt the NNLS solver proposed by Cantarella and Piatek (2004) to solve it.[5]

The results on the three multilingual datasets are summarized in Table 7. We observe from the table that SHFR-LASSO performs the worst and SHFR-NNLS enhances its performances through the nonnegative constraint. This is probably because the feature mapping $\mathbf{G}$ is learned globally and the features are naturally nonnegative for text classification. As SHFR explicitly imposes both nonnegativity and sparsity in learning the transformation $\mathbf{G}$, it can further enhance the performance compared to SHFR-NNLS by avoiding over adapting the noise in the high-dimensional data.

6.9 Experiments on Removing Negative Alignment

We have observed from Table 6 that the accuracy of SHFR for some constructed binary tasks is slightly less than the accuracy of SVM-T after learning and using the transformation $\mathbf{G}$, which can be referred to as *negative transfer* (Pan and Yang, 2010). The problem of negative transfer in SHFR mainly occurs for the following two reasons:

1. *Negative alignment* between the source and target classifiers, which happens when the classification accuracy of the corresponding target classifier is below 50%. This is because some initial target classifiers are less accurate due to the limited number of target domain labeled data.

2. The transformation $\mathbf{G}$ is class-invariant, therefore $\mathbf{G}$ is enforced to generalize well on all classifiers. In this case, $\mathbf{G}$ may correct the bias of some weak classifiers, but may also smooth the strength of good classifiers.

In this experiment, we study how the performance of SHFR is affected by removing negative alignments, and how SHFR performs if only good target classifiers whose accuracy is above 50%

5. In practice, the nonnegative lasso problem has been well studied and applied to many applications such as dimensionality reduction (Wang and Ye, 2014) and index tracking (Wu et al., 2014). Nevertheless, how to solve the nonnegative lasso problem is not the focus of our paper. In this paper, we address it by modifying the algorithm proposed by Tan et al. (2015a,b), which has demonstrated superior performance over many state-of-the-art solvers.

Table 8: Comparison results on each binary task after removing negative alignment in terms of classification accuracy (%).

Binary Classifiers	1	2	3	4	5	6	7	8
SVM-T	74.42	67.98	85.67	82.53	**43.32**	75.95	66.30	73.92
SHFR-RN	85.39	83.26	86.45	92.43	83.79	78.19	77.45	81.28
Difference	10.97	15.28	0.87	9.90	40.47	2.24	11.15	7.36
Binary Classifiers	9	10	11	12	13	14	15	
SVM-T	**42.78**	76.56	78.32	65.43	72.74	**46.56**	**49.71**	
SHFR-RN	84.15	78.45	92.14	82.12	84.58	82.72	88.61	
Difference	41.37	1.89	13.82	16.69	11.84	36.16	38.90	

are used. We investigate whether dropping negative alignments improves accuracy on all the binary tasks as well as the final multi-class HDA classification problem. Experimental results are shown in Table 8 and Figure 7. Compared to the results in Table 6, we discover from Table 8 that by removing the negative alignments we can improve all binary classification performances in terms of accuracy, and thus avoid negative transfers. We sort the binary tasks created from ECOC according to the accuracy on the test data and remove the poorly designed tasks. From Figure 7, we observe that when the number of dichotomizers is small (e.g., less than 30), SHFR with negative alignment removed achieves better performance in terms of multi-class classification accuracy. However, in practice, it is difficult to evaluate target classifiers due to the limit number of target domain labeled data. Interestingly, we can also find from Figure 7 that when the number of dichotomizers or tasks is sufficient, the classification accuracy of SHFR with or without removing negative alignments converges to the same value. This is because sub-class partition in ECOC often leads to higher binary accuracy when creating many dichotomizers or tasks. In this case, there are fewer binary target classifiers whose accuracy is below 50%. Therefore, we can conclude that SHFR with negative alignments removed enhances the performance in terms of classification accuracy when there are insufficient classifiers. In the multi-class HDA setting, however, the difficulty in alignment evaluation makes this post-processing impractical. Fortunately, SHFR can achieve the same performance as SHFR-RNs when there are sufficient dichotomizers or tasks, which makes SHFR more practicable for the multi-class HDA problem.

6.10 Training Time Comparison

Lastly, to verify the computational efficiency of SHFR, we conduct two experiments to compare the training time in learning the feature mapping between SHFR and other baselines under varying numbers of data dimensions and data instances. These experiments are conducted on a toy dataset, where the dimensions of the source and target data are constructed to be the same. In the first experiment, shown in Figure 8(a), we fix the number of training data (including both source and target domain data) to 3,000 and vary the data dimensions in the range of $\{100, 200, 500, 1000, 2000, 5000\}$. We observe that SHFR performs faster than all the baselines, and its training time increases slowly with the number of data dimensions. This is because the time complexity of SHFR has a linear relationship with the target domain dimensions. We can also see that the training time of DAMA increases dramatically when the number of data dimensions increases. In contrast, the training time of HFA and ARC-t does not increase much when the number of data dimensions increases. This

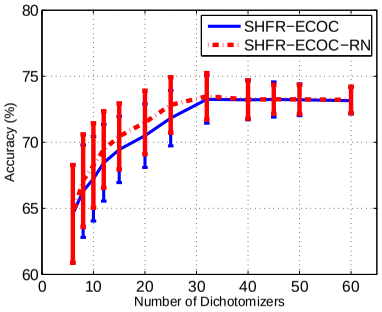
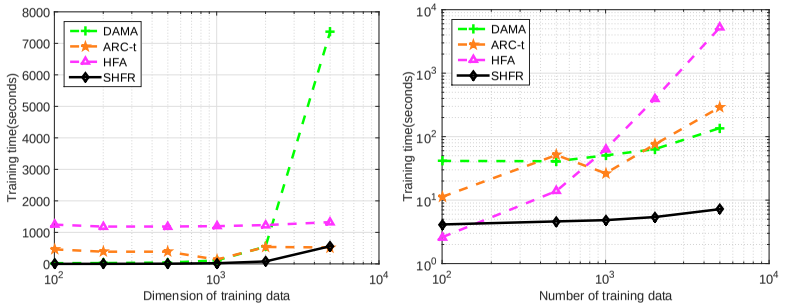


Figure 7: Accuracy v.s. dichotomizers size on removing tasks with negative alignment.



(a) Training time v.s. data dimensions (b) Training time v.s. data size

Figure 8: Time complexity comparison

is because these two methods adopt the kernel technique such that the computational time depends on the number of data instances instead of the data dimensions. In the second experiment, shown in Figure 8(b), we fix the data dimension to 1,000 and vary the number of training instances in the range of $\{100, 500, 1000, 2000, 5000\}$. From the figure, we observe that the training time of SHFR changes slightly when the number of training instances increases. However, the training time of ARC-t and HFA increases polynomially when the number of training instances increases. These two experiments verify the superiority of SHFR in computational efficiency compared with the baseline methods.

7. Conclusion and Future Work

In this paper, we propose a **sparse heterogeneous feature representation** (SHFR) method for learning sparse transformation between heterogeneous features for HDA by exploring the common underlying structures of multiple classes between the source and target domains for multi-class

HDA. SHFR encodes the sparsity and class-invariance properties in learning the feature mapping. Learning feature mapping can be cast as a Compressed Sensing (CS) problem for SHFR. Based on the CS theory, we show that the number of binary learning tasks affects the multi-class HDA performance. In addition, the proposed method has superior scalability over other methods. Extensive experiments demonstrate the effectiveness, efficiency, and stability of SHFR.

In future, we would like to investigate how to extend the proposed method by exploring the nonlinear feature transformation between image and text feature representations.

Acknowledgments

We would like to thank the action editor Shie Mannor and anonymous reviewers for their valuable comments and constructive suggestions that greatly contributed to improving the final version of the paper.

Joey Tianyi Zhou was partially supported by programmatic grant no. A1687b0033, A18A1b0045 from the Singapore government's Research, Innovation and Enterprise 2020 plan (Advanced Manufacturing and Engineering domain). Ivor W. Tsang acknowledges the partial funding from the ARC Future Fellowship FT130100746, ARC grant LP150100671, and DP180100106. Sinno J. Pan thanks the support from NTU Singapore Nanyang Assistant Professorship (NAP) grant M4081532.020, Singapore MOE AcRF Tier-2 grant MOE2016-T2-2-060, and the Data Science & Artificial Intelligence Research Centre at NTU Singapore. Mingkui Tan was partially supported by National Natural Science Foundation of China (NSFC) 61602185, Program for Guangdong Introducing Innovative and Enterpreneurial Teams 2017ZT07X183, Guangdong Provincial Scientific and Technological Funds under Grants 2018B010107001.

References

Erin L. Allwein, Robert E. Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *J. Mach. Learn. Res.*, 1:113–141, September 2001.

Rie K. Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, 6:1817–1853, 2005.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *NIPS*, pages 41–048. MIT Press, 2007.

Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin. A simple proof of the restricted isometry property for random matrices. *Constr. Approx.*, 28(3):253–263, 2008.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *NIPS*, pages 137–144, 2006.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Mach. Learn.*, 79(1-2):151–175, May 2010.

John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *EMNLP*, pages 120–128, 2006.

John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. In *NIPS*, 2007a.

John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 2007b.

Alfred M Bruckstein, Michael Elad, and Michael Zibulevsky. On the uniqueness of nonnegative sparse solutions to underdetermined systems of equations. *IEEE Trans. Inform. Theory*, 54(11): 4813–4820, 2008.

Emmanuel J. Candès, Justin K. Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure Appl. Math.*, 59(8):1207–1223, August 2006.

Jason Cantarella and Michael Piatek. TSNNLS: A solver for large sparse least squares problems with non-negative variables. *arXiv preprint cs/0408029*, 2004.

Yair Censor and Stavros A. Zenios. *Parallel Optimization: Theory, Algorithms and Applications*. Oxford University Press, 1997.

Wenyuan Dai, Yuqiang Chen, Gui-Rong Xue, Qiang Yang, and Yong Yu. Translated learning: Transfer learning across different feature spaces. In *NIPS*, pages 353–360, 2008.

Hal Daumé III. Frustratingly easy domain adaptation. In *ACL*, pages 256–263. ACL, June 2007.

Thomas G. Dietterich. Ensemble methods in machine learning. In *MCS*, pages 1–15. Springer-Verlag, 2000.

Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *J. Artif. Intell. Res.*, 2:263–286, 1995.

Jeff Donahue, Judy Hoffman, Erik Rodner, Kate Saenko, and Trevor Darrell. Semi-supervised domain adaptation with instance constraints. In *CVPR*, 2013.

David L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52:1289–1306, 2006.

David L Donoho and Jared Tanner. Counting the faces of randomly-projected hypercubes and orthants, with applications. *Discrete & Comput. Geom.*, 43(3):522–541, 2010.

Lixin Duan, Ivor W. Tsang, Dong Xu, and Tat-Seng Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *ICML*, pages 289–296, 2009.

Lixin Duan, Dong Xu, and Ivor W. Tsang. Learning with augmented features for heterogeneous domain adaptation. In *ICML*, 2012.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008.

Nicolás García-Pedrajas and Domingo Ortiz-Boyer. An empirical study of binary classifier fusion methods for multiclass classification. *Inf. Fusion*, 12(2):111–130, April 2011.

Rayid Ghani. Using error-correcting codes for text classification. In *ICML*, pages 303–310, 2000.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *J. Mach. Learn. Res.*, 12:2211–2268, 2011.

Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, pages 2066–2073. IEEE, 2012.

Boqing Gong, Kristen Grauman, and Fei Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, pages 222–230, 2013.

Maayan Harel and Shie Mannor. Learning from multiple outlooks. In *ICML*, pages 401–408, 2011.

Judy Hoffman, Erik Rodner, Jeff Donahue, Kate Saenko, and Trevor Darrell. Efficient learning of domain-invariant image representations. In *ICLR*, 2013.

Judy Hoffman, Sergio Guadarrama, Eric S Tzeng, Ronghang Hu, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. LSDA: Large scale detection through adaptation. In *NIPS*, pages 3536–3544, 2014.

Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Correcting sample selection bias by unlabeled data. In *NIPS*, pages 601–608, 2007.

Jing Jiang and ChengXiang Zhai. Instance weighting for domain adaptation in NLP. In *ACL*, pages 264–271. ACL, 2007.

Brian Kulis, Kate Saenko, and Trevor Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, pages 1785–1792, 2011.

G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *J. Mach. Learn. Res.*, 5:27–72, 2004.

Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562, 2001.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *COLT*, 2009.

Gholam Ali Montazer, Sergio Escalera, et al. Error correcting output codes for multiclass classification: Application to two image vision problems. In *AISP*, pages 508–513. IEEE, 2012.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, October 2010. ISSN 1041-4347.

Sinno Jialin Pan, James T. Kwok, and Qiang Yang. Transfer learning via dimensionality reduction. In *AAAI*, pages 677–682, July 2008.

Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Chen Zheng. Cross-domain sentiment classification via spectral feature alignment. In *WWW*, pages 751–760. ACM, 2010.

Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Trans. Neural Netw.*, 22(2):199–210, 2011.

Peter Prettenhofer and Benno Stein. Cross-language text classification using structural correspondence learning. In *ACL*, pages 1118–1127, 2010.

Guojun Qi, Charu C. Aggarwal, and Thomas S. Huang. Towards semantic knowledge propagation from text corpus to web images. In *WWW*, pages 297–306, 2011.

Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226. Springer-Verlag, 2010.

Chun-Wei Seah, Ivor Wai-Hung Tsang, and Yew-Soon Ong. Healing sample selection bias by source classifier selection. In *ICDM*, pages 577–586, 2011.

Xiaoxiao Shi, Qi Liu, Wei Fan, Philip S. Yu, and Ruixin Zhu. Transfer learning on heterogenous feature spaces via spectral transformation. In *ICDM*, pages 1049–1054, 2010.

Martin Slawski and Matthias Hein. Sparse recovery by thresholded non-negative least squares. In *NIPS*, pages 1926–1934, 2011.

Martin Slawski, Matthias Hein, et al. Non-negative least squares for high-dimensional linear models: Consistency and sparse recovery without regularization. *Electron. J. Stat.*, 7:3004–3056, 2013.

Mingkui Tan, Ivor W. Tsang, and Li Wang. Matching pursuit LASSO part I: Sparse recovery over big dictionary. *IEEE Trans. Signal Processing*, 63(3):727–741, 2015a.

Mingkui Tan, Ivor W. Tsang, and Li Wang. Matching pursuit LASSO part II: Applications and sparse recovery over batch signals. *IEEE Trans. Signal Processing*, 63(3):742–753, 2015b.

Kim-Chuan Toh and Sangwoon Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pac. J. Optim.*, 6(615-640):15, 2010.

Chang Wang and Sridhar Mahadevan. Heterogeneous domain adaptation using manifold alignment. In *IJCAI*, pages 1541–1546, 2011.

Jie Wang and Jieping Ye. Two-layer feature reduction for sparse-group lasso via decomposition of convex sets. In *NIPS*, pages 2132–2140, 2014.

Meng Wang and Ao Tang. Conditions for a unique non-negative solution to an underdetermined system. In *Allerton*, pages 301–307. IEEE, 2009.

Meng Wang, Weiyu Xu, and Ao Tang. A unique nonnegative solution to an underdetermined system: From vectors to matrices. *IEEE Trans. Signal Processing*, 59(3):1007–1016, 2011.

Lan Wu, Yuehan Yang, and Hanzhong Liu. Nonnegative-lasso and application in index tracking. *Comput. Stat. Data An.*, 70:116–126, 2014.

Qiang Yang, Yuqiang Chen, Gui-Rong Xue, Wenyuan Dai, and Yong Yu. Heterogeneous transfer learning for image clustering via the socialweb. In *ACL/IJCNLP*, pages 1–9, 2009.

Ming Yuan and Yi Lin. On the non-negative garrotte estimator. *J. Roy. Stat. Soc. B*, 69(2):143–161, 2007.

Cun-Hui Zhang and Jian Huang. The sparsity and bias of the LASSO selection in high-dimensional linear regression. *Ann. Statist.*, 36(4):1567–1594, 2008.

Tong Zhang. On the consistency of feature selection using greedy least squares regression. *J. Mach. Learn. Res.*, 10:555–568, 2009.

Joey Tianyi Zhou, Ivor W Tsang, Shen-Shyang Ho, and Klaus-Robert Müller. N-ary decomposition for multi-class classification. *Mach. Learn.*, pages 1–22.

Joey Tianyi Zhou, Sinno Jialin Pan, Ivor W. Tsang, and Yan Yan. Hybrid heterogeneous transfer learning through deep learning. In *AAAI*, pages 2213–2220, 2014.

Joey Tianyi Zhou, Sinno Jialin Pan, Ivor W. Tsang, and Shen-Shyang Ho. Transfer learning for cross-language text categorization through active correspondences construction. In *AAAI*, pages 2400–2406, 2016a.

Joey Tianyi Zhou, Xinxing Xu, Sinno Jialin Pan, Ivor W. Tsang, Zheng Qin, and Rick Siow Mong Goh. Transfer hashing with privileged information. In *IJCAI*, pages 2414–2420, 2016b.

Joey Tianyi Zhou, Heng Zhao, Xi Peng, Meng Fang, Zheng Qin, and Rick Siow Mong Goh. Transfer hashing: From shallow to deep. *IEEE Trans. Neural Netw. Learning Syst.*, 29(12):6191–6201, 2018.

Yin Zhu, Yuqiang Chen, Zhongqi Lu, Sinno Jialin Pan, Gui-Rong Xue, Yong Yu, and Qiang Yang. Heterogeneous transfer learning for image classification. In *AAAI*, 2011.

Journal of Machine Learning Research 20 (2019) 1-49 Submitted 6/16; Revised 9/18; Published 3/19

The Common-directions Method for Regularized Empirical Risk Minimization

Po-Wei Wang POWEIW@CS.CMU.EDU
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA
Ching-pei Lee CHING-PEI@CS.WISC.EDU
Department of Computer Sciences
University of Wisconsin–Madison
Madison, WI 53706-1613, USA
Chih-Jen Lin CJLIN@CSIE.NTU.EDU.TW
Department of Computer Science
National Taiwan University
Taipei 106, Taiwan

Editor: Tong Zhang

Abstract

State-of-the-art first- and second-order optimization methods are able to achieve either fast global linear convergence rates or quadratic convergence, but not both of them. In this work, we propose an interpolation between first- and second-order methods for regularized empirical risk minimization that exploits the problem structure to efficiently combine multiple update directions. Our method attains both optimal global linear convergence rate for first-order methods, and local quadratic convergence. Experimental results show that our method outperforms state-of-the-art first- and second-order optimization methods in terms of the number of data accesses, while is competitive in training time.

1. Introduction

Consider the general problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} \quad f(\boldsymbol{w}), \tag{1}$$

where $f(\boldsymbol{w})$ is convex and continuously differentiable. Most iterative optimization methods generate iterates by using the first- and second-order derivatives. After a new step is generated, the previous steps and related calculation results are usually considered outdated and thus discarded. In this work, we show that we can reuse previous descent directions in later iterations to reduce computation and to achieve both linear and quadratic convergence.

The idea behind our proposed method is simple. We store all previous directions, and approximately solve the subproblem

$$\min_{\boldsymbol{t}\in\mathbb{R}^m} \quad f(\boldsymbol{w}+P\boldsymbol{t}), \tag{2}$$

©2019 Po-Wei Wang, Ching-pei Lee, and Chih-Jen Lin.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/16-309.html.

where the columns of $P = [\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m]$ form an orthonormal basis of the span of the previous update directions. The resulting direction $P\boldsymbol{t}$ is then used to update the current iterate $\boldsymbol{w}$ for the next iteration. Utilizing all previous directions obtained from the first iteration on seems to be computationally and spatially expensive, especially in the later stage of this method. However, we note that for regularized empirical risk minimization (ERM) of linear models, the method comes with little additional cost by wisely caching the inner products between these $\boldsymbol{p}_i$ and the training data. We will exemplify how to use this caching strategy to significantly accelerate the computation in our method for ERM. Experimental results show that our method outperforms state of the art.

There are other optimization methods that also reuse previous update directions. For example, the heavy-ball method (Polyak, 1964), the (nonlinear) conjugate gradient method (Fletcher and Reeves, 1964), and Nesterov's accelerated gradient method (Nesterov, 1983, 2013) reuse the previous direction $\boldsymbol{w}_k - \boldsymbol{w}_{k-1}$ together with the current gradient to decide the update direction at the k-th iteration. On the other hand, quasi-Newton methods such as BFGS (Dennis and Moré, 1977) and L-BFGS (Liu and Nocedal, 1989) store previous gradients to construct an approximation of the Hessian. More details of these methods and empirical comparisons are provided in later sections.

This paper is organized as follows. We describe the details of our algorithm in Section 2. We prove global linear convergence and local quadratic convergence. Section 3 illustrates important techniques for accelerating the computation when solving ERM problems. We then extend the proposed method to a more general framework and obtain a better linear convergence rate in Section 4. Related works are discussed in Section 5. Experimental results in Section 6 show that our method outperforms state-of-the-art methods empirically in terms of the number of data accesses, and is competitive in terms of training time. We then consider some interesting extensions in Section 7. Section 8 concludes the paper.

Programs used for experiments and a supplementary file including additional results can be found at `http://www.csie.ntu.edu.tw/~cjlin/papers/commdir/`.

2. Common-directions Method

Before starting the description of our algorithm, we first specify the type of problems we consider. For all our theoretical results except Theorem 6, we require that $f(\boldsymbol{w})$ satisfies Assumption 1.

Assumption 1 *The objective function $f(\boldsymbol{w})$ is differentiable, ρ Lipschitz smooth and σ strongly convex with some constants $\rho \geq \sigma > 0$.*[1] *That is,*

$$\|\nabla f(\boldsymbol{u}) - \nabla f(\boldsymbol{v})\| \leq \rho\|\boldsymbol{u}-\boldsymbol{v}\|, \quad \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n, \tag{3}$$

and

$$f(\boldsymbol{u}) - f(\boldsymbol{v}) - \nabla f(\boldsymbol{v})^\top(\boldsymbol{u}-\boldsymbol{v}) \geq \frac{\sigma}{2}\|\boldsymbol{u}-\boldsymbol{v}\|^2, \quad \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n. \tag{4}$$

For some theoretical results, we further need that $f(\boldsymbol{w})$ satisfies Assumption 2.

1. The requirement $\rho \geq \sigma$ comes from the conditions (3) and (4).

Algorithm 1: A framework for the common-directions method

Given $\boldsymbol{w}_0$, compute $\nabla f(\boldsymbol{w}_0)$;
$P = [\nabla f(\boldsymbol{w}_0)/\|\nabla f(\boldsymbol{w}_0)\|]$;
for *k=0,1,... in the outer loop* **do**
  if $\nabla f(\boldsymbol{w}_k) = \mathbf{0}$, or some stopping condition is satisfied **then**
    stop;
  end
  Let $\boldsymbol{t}_k = -(P^\top \nabla^2 f(\boldsymbol{w}_k) P)^{-1} P^\top \nabla f(\boldsymbol{w}_k)$;
  Backtracking line search on $f(\boldsymbol{w}_k + \theta P \boldsymbol{t}_k)$ by Algorithm 2 to obtain θ_k;
  $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \theta_k P \boldsymbol{t}_k$;
  Compute $\nabla f(\boldsymbol{w}_{k+1})$;
  Let $\boldsymbol{p} = \nabla f(\boldsymbol{w}_{k+1}) - P(P^\top \nabla f(\boldsymbol{w}_{k+1}))$;
  if $\boldsymbol{p} \neq \mathbf{0}$ **then**
    $P = [P,\ \boldsymbol{p}/\|\boldsymbol{p}\|]$;
  end
end

Algorithm 2: Backtracking line search

Given $\beta \in (0,1)$, $\lambda > 0$, the current iterate $\boldsymbol{w}$, a descent direction $P\boldsymbol{t}$;
Let $\theta := 1$;
while $f(\boldsymbol{w}) - f(\boldsymbol{w} + \theta P \boldsymbol{t}) < \frac{\lambda}{2}\theta^2 \|P\boldsymbol{t}\|^2$ **do**
  $\theta := \beta\theta$;
end

Assumption 2 *The objective function $f(\boldsymbol{w})$ is twice-differentiable, and the Hessian of $f(\boldsymbol{w})$ is M Lipschitz continuous with some constant $M > 0$. That is,*

$$\|\nabla^2 f(\boldsymbol{u}) - \nabla^2 f(\boldsymbol{v})\| \leq M\|\boldsymbol{u} - \boldsymbol{v}\|, \quad \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n.$$

When minimizing a function, we usually only work on derivatives of the current iterate. For example, a gradient descent step can be derived from searching along the negative gradient direction, and a Newton step can be expressed by the current gradient and Hessian. However, after obtaining the update direction, stale derivatives are usually discarded, which can be inefficient. A simple idea is to store all previous directions $\boldsymbol{p}_j$ in $P = [\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m]$ and consider working on the linear span of $\boldsymbol{p}_j$. If possible, we search for the best linear combination $P\boldsymbol{t}$ such that $f(\boldsymbol{w} + P\boldsymbol{t})$ is the smallest; see the sub-problem (2). However, in practice (2) may be difficult to be solved, so following past optimization development, we consider the second-order Taylor approximation of $f(\boldsymbol{w})$ as

$$\tilde{f}(\boldsymbol{u} \mid \boldsymbol{w}) \equiv f(\boldsymbol{w}) + \nabla f(\boldsymbol{w})^\top (\boldsymbol{u} - \boldsymbol{w}) + \frac{1}{2}(\boldsymbol{u} - \boldsymbol{w})^\top \nabla^2 f(\boldsymbol{w})(\boldsymbol{u} - \boldsymbol{w}), \tag{5}$$

where $\nabla^2 f(\boldsymbol{w})$ is the Hessian at $\boldsymbol{w}$. Then, a descent direction $\boldsymbol{d} = P\boldsymbol{t}$ can be found by optimizing $\tilde{f}$ along all the directions $\boldsymbol{p}_j$

$$\min_{\boldsymbol{t}} \quad \tilde{f}(\boldsymbol{w} + \sum_{j=1}^{m} t_j \boldsymbol{p}_j \mid \boldsymbol{w}) = (P^\top \nabla f(\boldsymbol{w}))^\top \boldsymbol{t} + \frac{1}{2}\boldsymbol{t}^\top (P^\top \nabla^2 f(\boldsymbol{w}) P)\boldsymbol{t}. \tag{6}$$

Problem (6) can be solved by the following linear system.

$$(P^\top \nabla^2 f(\boldsymbol{w}) P)\boldsymbol{t} = -P^\top \nabla f(\boldsymbol{w}).$$

Say we have stored m previous directions in $P \in \mathbb{R}^{m \times n}$, where $m \ll n$. Problem (6) has m variables but if instead we consider the Newton direction by solving the problem in (5), the number of variables is n. Therefore, solving (6) should be easier than finding a Newton direction if the construction of $P\nabla^2 f(\boldsymbol{w})P^\top$ is not too expensive. In Section 3, we will show that when the Hessian is structured, the subproblem (6) can be solved easily.

From the above concept, we propose the common-directions method. In each iteration of this method, given the orthonormal basis P of the span of the past directions, we solve the subproblem (6) with respect to $\boldsymbol{t}$ to obtain the update direction $P\boldsymbol{t}$. A line search along $P\boldsymbol{t}$ is performed to guarantee the strict decrease of the function value. After updating the iterate $\boldsymbol{w}$, a new direction $-\nabla f(\boldsymbol{w})$ is then taken to enlarge the orthonormal basis P by Gram-Schmidt orthogonalization. A description of the common-directions method is given in Algorithm 1, and the line-search sub-routine is described in Algorithm 2. Note that other choices of the directions being added to P are also possible, but for the ease of description and analysis, we confine our choice to the current steepest descent direction.

We define the following quantities before stating and proving the theorems. We denote the condition number of f by

$$\kappa \equiv \frac{\rho}{\sigma},$$

and define an ϵ-accurate solution of f to be a point $\boldsymbol{w}$ such that

$$f(\boldsymbol{w}) - f(\boldsymbol{w}^*) \leq \epsilon,$$

where $\boldsymbol{w}^*$ is the optimal solution of f. Note that under Assumption 1, problem (1) possesses a unique optimal solution. The following theorems present theoretical convergence of the proposed method. In particular, we show that the line search sub-routine has finite termination, and our method is globally linearly convergent and locally quadratically convergent.

Theorem 3 *Under Assumption 1, every time being called by Algorithm 1, Algorithm 2 terminates within $\lceil \log_\beta(\beta\sigma/(\rho+\lambda)) \rceil$ steps, and Algorithm 1 converges Q-linearly with iteration complexity $O(\kappa^3 \log(1/\epsilon))$ to achieve an ϵ-accurate solution. Further, if Algorithm 2 always stops at step size 1, the iteration complexity becomes $O(\kappa \log(1/\epsilon))$.*

In practice, we observed that the common-direction method usually accepts step size 1.

Theorem 4 *Under Assumptions 1 and 2, Algorithm 1 converges quadratically after certain iteration.*

The proofs are presented in Appendix A.

3. Application on Regularized Empirical Risk Minimization Problems

For most optimization problems, computing the $n \times n$ Hessian matrix, or its inverse, whose cost is $O(n^3)$, is much more expensive than obtaining the function value and the n-dimensional gradient vector. To tackle this problem, different approaches have been considered, and in some cases the problem structure can be utilized to reduce the cost. We are interested in solving machine learning problems, and thus we will focus our discussion on differentiable regularized ERM problems. More specifically, given training feature-label pairs $(\boldsymbol{x}_i, y_i)$, $i = 1, \ldots, l$, with $\boldsymbol{x}_i \in \mathbb{R}^n$, $y_i \in \mathbb{R}$, and a parameter $C > 0$, we consider L2-regularized ERM problems

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} \quad f(\boldsymbol{w}), \quad \text{where } f(\boldsymbol{w}) \equiv \frac{1}{2}\boldsymbol{w}^\top\boldsymbol{w} + C\sum_{i=1}^{l}\xi(y_i; \boldsymbol{w}^\top\boldsymbol{x}_i), \tag{7}$$

and ξ is a loss function convex in $\boldsymbol{w}^\top\boldsymbol{x}_i$. The first term in (7) makes f be σ strongly convex with $\sigma \geq 1$, and the choice of loss functions ensures that $\nabla f(\boldsymbol{w})$ is Lipschitz continuous, though the corresponding constant ρ may be unknown. Therefore, Assumption 1 is satisfied.

For problem (7), by defining

$$X \equiv \begin{pmatrix} \boldsymbol{x}_1^\top \\ \vdots \\ \boldsymbol{x}_l^\top \end{pmatrix},$$

we can write the gradient as

$$\nabla f(\boldsymbol{w}) = \boldsymbol{w} + X^\top \boldsymbol{v}_{\boldsymbol{w}}, \text{ where } (v_{\boldsymbol{w}})_i = C\frac{\partial}{\partial z}\xi(y_i; z)\mid_{z=\boldsymbol{w}^\top\boldsymbol{x}_i}, i = 1, \ldots, l, \tag{8}$$

and if $f(\boldsymbol{w})$ is twice-differentiable, the Hessian is

$$\nabla^2 f(\boldsymbol{w}) \equiv I + X^\top D_{\boldsymbol{w}} X, \tag{9}$$

where I is the identity matrix and $D_{\boldsymbol{w}}$ is a diagonal matrix with

$$(D_{\boldsymbol{w}})_{ii} \equiv C\frac{\partial^2}{\partial z^2}\xi(y_i; z)\mid_{z=\boldsymbol{w}^\top\boldsymbol{x}_i}, i = 1, \ldots, l. \tag{10}$$

The structure of (9) allows efficient computations to use the Hessian in various ways. Our method utilizes it to incorporate the Hessian-vector products with low cost as discussed in the following sections.

Some problems of the form (7), such as logistic regression and ridge regression, also satisfy Assumption 2. The loss of ridge regression is

$$\xi(y_i; \boldsymbol{w}^\top\boldsymbol{x}_i) = (y_i - \boldsymbol{w}^\top\boldsymbol{x}_i)^2.$$

It is clearly twice-differentiable with the diagonal matrix $D_{\boldsymbol{w}}$ defined in (10) being the identity matrix. Clearly, this loss satisfies Assumption 2 with any $M \geq 0$ as the Hessian is a constant. The loss of logistic regression is

$$\xi(y_i; \boldsymbol{w}^\top\boldsymbol{x}_i) = \log(1 + \exp(-y_i\boldsymbol{w}^\top\boldsymbol{x}_i)),$$

where $y_i \in \{-1, 1\}$. Because its third derivative is upper-bounded, by applying first-order Taylor expansion on the Hessian, one can see that Assumption 2 holds. On the other hand, problems that satisfy Assumption 1 only, like squared-hinge loss support vector classification/regression (Boser et al., 1992; Vapnik, 1995), can also be solved by our algorithm, although in this case we can only expect linear convergence.[2]

3.1. Subproblem of Common Directions

At the k-th iteration, given the matrix $P_k = [\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_m] \in \mathbb{R}^{n\times m}$ with orthogonal columns, we consider solving (6) along $P_k\boldsymbol{t}$

$$\min_{\boldsymbol{t}\in\mathbb{R}^m} \quad \tilde{f}(\boldsymbol{w}_k + P_k\boldsymbol{t} \mid \boldsymbol{w}_k). \tag{11}$$

The gradient and the Hessian of (11) with respect to the variable $\boldsymbol{t}$ at $\boldsymbol{t} = \boldsymbol{0}$ are respectively

$$\nabla_{\boldsymbol{t}}\tilde{f}(\boldsymbol{w}_k + P_k\boldsymbol{t})\mid_{\boldsymbol{t}=\boldsymbol{0}} = P_k^\top\nabla f(\boldsymbol{w}_k) \tag{12}$$

and

$$\nabla^2_{\boldsymbol{t},\boldsymbol{t}}\tilde{f}(\boldsymbol{w}_k + P_k\boldsymbol{t})\mid_{\boldsymbol{t}=\boldsymbol{0}} = P_k^\top\nabla^2 f(\boldsymbol{w}_k)P_k = P_k^\top(I + X^\top D_{\boldsymbol{w}_k}X)P_k = I + P_k^\top X^\top D_{\boldsymbol{w}_k}XP_k. \tag{13}$$

Observe the term XP_k. The trick to make the computation efficient is to store this term in memory to avoid repetitive computations of $X\boldsymbol{p}_j$ for all previous directions $\boldsymbol{p}_j$. If XP_k, $\nabla f(\boldsymbol{w}_k)$, and $D_{\boldsymbol{w}_k}$ are available, the construction of (12)-(13) involves the following cost.

- $P_k^\top\nabla f(\boldsymbol{w}_k)$: $O(mn)$,
- $(XP_k)^\top D_{\boldsymbol{w}_k}(XP_k)$: $O(lm^2)$.

The optimal solution $\boldsymbol{t}$ of (11) can be obtained by solving the following linear system

$$\left(I + P_k^\top X^\top D_{\boldsymbol{w}_k}XP_k\right)\boldsymbol{t} = -P_k^\top\nabla f(\boldsymbol{w}_k) \tag{14}$$

in $O(m^3)$ cost. Note that the Hessian with respect to $\boldsymbol{t}$ is always invertible because the term I in (13) ensures its positive definiteness.

Let $\boldsymbol{t}_k$ be the solution of (14). We show that to obtain necessary information for the next iteration, it is sufficient to maintain XP_{k+1} and $X\boldsymbol{w}_{k+1}$. To have the new function value, from (7) we mainly need to compute $X\boldsymbol{w}_{k+1}$. If $X\boldsymbol{w}_k$ is available and the full direction $(XP_k)\boldsymbol{t}_k$ is taken, by

$$X\boldsymbol{w}_{k+1} = X\boldsymbol{w}_k + \theta(XP_k)\boldsymbol{t}_k, \tag{15}$$

with $\theta = 1$, in $O(lm)$ cost $X\boldsymbol{w}_{k+1}$ is obtained. In practice, a step size θ is decided by line search, which involves some minor cost as shown in Section 3.2. The vector $X\boldsymbol{w}_{k+1}$ is also used to construct $\boldsymbol{v}_{\boldsymbol{w}_{k+1}}$ needed for gradient calculation. Then from (8),

$$\nabla f(\boldsymbol{w}_{k+1}) = \boldsymbol{w}_{k+1} + X^\top\boldsymbol{v}_{\boldsymbol{w}_{k+1}}$$

2. If the objective function is not twice differentiable, we can use the generalized Hessian (Mangasarian, 2002) to replace the Hessian in the Newton step, as Assumption 1 guarantees the existence of generalized Hessian.

takes $O(\#\text{nnz})$ cost, where #nnz is the number of non-zero elements in X. We then apply the Gram-Schmidt procedure to check if a new column should be added to P_{k+1}. In practice this procedure is computed by

$$\boldsymbol{p}_{m+1} = (I - P_k P_k^T)\nabla f(\boldsymbol{w}_{k+1}) = \nabla f(\boldsymbol{w}_{k+1}) - P_k(P_k^\top \nabla f(\boldsymbol{w}_{k+1})).$$

The cost is $O(mn)$. If $\boldsymbol{p}_{m+1} \neq \boldsymbol{0}$, to maintain XP_{m+1} after $\boldsymbol{p}_{m+1}$ is added, we must compute $X\boldsymbol{p}_{m+1}$, which costs $O(\#\text{nnz})$. Details of the algorithm is shown in Algorithm 3.

In summary, the cost per iteration is

$$O(lm^2 + mn + m^3 + \#\text{nnz}). \tag{16}$$

Because usually we have $n \gg m$ and $l \gg m$ from the first iteration till reaching a good enough solution, in general #nnz is still the dominant term in (16). Therefore, each of our iterations does not cost much more than a typical iteration in an optimization method that often involves the $O(\#\text{nnz})$ cost.

Further, for the same set of directions P_k, we could solve the subproblem (11) and update $\boldsymbol{w}_k$ multiple times in order to approximately solve (2). Note that the update of $\boldsymbol{w}_k$ does not need to go through the data matrix X. The reason is that the $D_{\boldsymbol{w}}$ matrix and the $\boldsymbol{v}_{\boldsymbol{w}}$ vector rely on $\boldsymbol{w}_k$ only through the inner products $\boldsymbol{w}_k^\top \boldsymbol{x}_i$, and this is a linear combination of XP_k, and thus we can compute it without additional inner product computations with the data matrix X. This idea of solving (11) multiple times will be exploited in Section 4 to achieve the optimal linear convergence rate for first-order methods.

3.2. Line Search on Common Directions

Once the solution $\boldsymbol{t}$ of (14) has been obtained, in $O(mn)$ cost we get the descent direction

$$\boldsymbol{d} = P_k \boldsymbol{t}. \tag{17}$$

Next, we conduct line search in Algorithm 2 for the step size θ in

$$f(\boldsymbol{w}_k + \theta\boldsymbol{d}) = \frac{1}{2}\|\boldsymbol{w}_k + \theta\boldsymbol{d}\|^2 + C\sum_{i=1}^{l} \xi(y_i; \boldsymbol{w}_k^\top \boldsymbol{x}_i + \theta \boldsymbol{d}^\top \boldsymbol{x}_i).$$

The sufficient decrease condition is

$$C\sum_{i=1}^{l} \xi(y_i; \boldsymbol{w}_k^\top \boldsymbol{x}_i) - \theta \boldsymbol{w}_k^\top \boldsymbol{d} - \frac{\theta^2}{2}\|\boldsymbol{d}\|^2 - C\sum_{i=1}^{l} \xi(y_i; \boldsymbol{w}_k^\top \boldsymbol{x}_i + \theta \boldsymbol{d}^\top \boldsymbol{x}_i) \geq \frac{\lambda}{2}\theta^2 \|\boldsymbol{d}\|^2.$$

Because XP_k is maintained, after obtaining $X\boldsymbol{d}$ in $O(lm)$ time by

$$X\boldsymbol{d} = (XP_k)\boldsymbol{t}, \tag{18}$$

and calculating $\boldsymbol{w}_k^\top \boldsymbol{d}$, $\boldsymbol{w}_k^\top \boldsymbol{w}_k$, $\boldsymbol{d}^\top \boldsymbol{d}$ in $O(n)$ time, each line search step would only cost $O(l)$ because $\boldsymbol{w}_k^\top \boldsymbol{x}_i$ is already available in our calculation for the gradient and the Hessian. Thus, we could perform line search in

$$O(lm + mn + l \times \#(\text{line search steps})) \text{ time}. \tag{19}$$

By Theorem 3, #(line search) is bounded by a constant, and thus the complexity does not grow infinitely. After finishing the line search procedure, we already have $\boldsymbol{w}_k^\top \boldsymbol{x}_i + \theta \boldsymbol{d}^\top \boldsymbol{x}_i$ as the next $\boldsymbol{w}_{k+1}^\top \boldsymbol{x}_i$ as shown in (15).

Algorithm 3: The common-directions method for solving the ERM problem (7)

Given $\boldsymbol{w}_0$, compute $\boldsymbol{z} = X\boldsymbol{w}_0$;
Use $\boldsymbol{z}$ to calculate $\boldsymbol{v}_{\boldsymbol{w}_0}$ and $D_{\boldsymbol{w}_0}$ in (8) and (10);
Compute $\nabla f(\boldsymbol{w}_0) = \boldsymbol{w}_0 + X^\top \boldsymbol{v}_{\boldsymbol{w}_0}$ from (8);
$P = [\nabla f(\boldsymbol{w}_0)/\|\nabla f(\boldsymbol{w}_0)\|], U = XP$;
for *k=0,1,... in the oute loop* **do**
  if $\nabla f(\boldsymbol{w}_k) = \boldsymbol{0}$, or some stopping condition is satisfied **then**
    stop;
  end
  Obtain $\boldsymbol{t}$ by solving (14) with $\boldsymbol{t} = [I + U^\top D_{\boldsymbol{w}_k} U]^{-1}[-P^\top \nabla f(\boldsymbol{w}_k)]$;
  Compute $\hat{\boldsymbol{z}} = U\boldsymbol{t}$;
  Backtracking line search on $f(\boldsymbol{w}_k + \theta P\boldsymbol{t})$ by Algorithm 2 to obtain θ using the values of $\boldsymbol{z}$ and $\hat{\boldsymbol{z}}$ to compute $\xi(y_i; \boldsymbol{z} + \theta\hat{\boldsymbol{z}})$;
  $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \theta P\boldsymbol{t}$;
  $\boldsymbol{z} = \boldsymbol{z} + \theta\hat{\boldsymbol{z}}$;
  Use $\boldsymbol{z}$ to calculate $\boldsymbol{v}_{\boldsymbol{w}_{k+1}}$ and $D_{\boldsymbol{w}_{k+1}}$;
  Compute $\nabla f(\boldsymbol{w}_{k+1}) = \boldsymbol{w}_{k+1} + X^\top \boldsymbol{v}_{\boldsymbol{w}_{k+1}}$ from (8);
  Let $\boldsymbol{p} = \nabla f(\boldsymbol{w}_{k+1}) - P(P^\top \nabla f(\boldsymbol{w}_{k+1}))$;
  if $\boldsymbol{p} \neq \boldsymbol{0}$ **then**
    $P = [P,\ \boldsymbol{p}/\|\boldsymbol{p}\|], U = [U,\ X\boldsymbol{p}/\|\boldsymbol{p}\|]$;
  end
end

3.3. Total Cost of the Algorithm

Based on the analysis in Sections 3.1 and 3.2, the total cost is

$$O(lm^2 + mn + m^3 + \#\text{nnz} + l \times \#(\text{line search steps})) \times \text{iterations}. \tag{20}$$

In Section 5, we will compare the cost with that of some existing approaches.

4. Common-directions Method with Multiple Inner Iterations

In Section 2, we considered solving the second-order Taylor approximation of $f(\boldsymbol{w}_k + P_k\boldsymbol{t})$ with respect to the coefficients $\boldsymbol{t}$ to get a good combination along the directions of P_k. Now we further extend the idea to directly minimize $f(\boldsymbol{w}_k + P_k\boldsymbol{t})$. We first describe the general framework, and then discuss how it can be applied to the special case of solving regularized ERM problems.

4.1. General Framework

We can consider the following convex optimization problem

$$\min_{\boldsymbol{t}} \quad f(\boldsymbol{w}_k + P_k\boldsymbol{t}), \tag{21}$$

and apply any optimization algorithm to solve it. When applying iterative algorithms, we terminate the optimization procedure after the solution is considered good enough.

Therefore, usually (21) is only solved approximately. In particular, the method discussed in Section 2 can be viewed as using only one Newton iteration to loosely solve (21). Because of two levels of iterations, we refer to the process of approximately solving (21) as an outer iteration, while each step in the process as an inner iteration.

Among the different methods to solve (21), we apply multiple inner Newton steps on the variable $\boldsymbol{t}$. That is, we repeatedly minimize the second-order Taylor expansion of (21).

$$\min_{\Delta \boldsymbol{t} \in \mathbb{R}^m} \quad \tilde{f}(\boldsymbol{w}_k + P_k\boldsymbol{t} + P_k\Delta\boldsymbol{t} \mid \boldsymbol{w}_k + P_k\boldsymbol{t}).$$

Because

$$\begin{aligned}&\tilde{f}(\boldsymbol{w}_k + P_k\boldsymbol{t} + P_k\Delta\boldsymbol{t} \mid \boldsymbol{w}_k + P_k\boldsymbol{t})\\ &= f(\boldsymbol{w}_k + P_k\boldsymbol{t}) + (P_k^\top \nabla f(\boldsymbol{w}_k + P_k\boldsymbol{t}))^\top \Delta\boldsymbol{t} + \frac{1}{2}\Delta\boldsymbol{t}^\top P_k^\top \nabla f(\boldsymbol{w}_k + P_k\boldsymbol{t}) P_k \Delta\boldsymbol{t},\end{aligned}$$

this amounts to solving the following linear system for $\Delta\boldsymbol{t}$.

$$(P_k^\top \nabla^2 f(\boldsymbol{w}_k + P_k\boldsymbol{t}) P_k)\Delta\boldsymbol{t} = -P_k^\top \nabla f(\boldsymbol{w}_k + P_k\boldsymbol{t}). \tag{22}$$

Each time the solution of the linear system will be our update direction, and then we will conduct backtracking line search to ensure the sufficient function decrease. The detailed algorithm is described in Algorithm 4, where we can see that $\boldsymbol{w}_k + P_k\boldsymbol{t}$ is the iterate updated in the inner iterations.

We now show that this general framework has better global convergence rate than the one discussed in Section 2.

Theorem 5 *Under Assumption 1 and a proper inner stopping condition, given any $\epsilon > 0$, Algorithm 4 converges R-linearly and obtains an ϵ-accurate solution in $O(\sqrt{\kappa}\log(1/\epsilon))$ iterations.*

The proof is in Appendix B.

It has been shown in Nesterov (2003) that $O(\sqrt{\kappa}\log(1/\epsilon))$ is the optimal iteration complexity for methods whose directions are obtained from the span of $\nabla f(\boldsymbol{w}_0), \nabla f(\boldsymbol{w}_1), \cdots$. The inner stopping condition used in proving Theorem 5 is not easy to implement in practice. It is possible to devise a practical one to ensure that this optimal linear convergence rate can be guaranteed. However, as we will see in the empirical results in Section 6, the setting of using a single inner iteration in approximately solving (21) tends to outperform that of using multiple inner iterations in terms of overall training time. In light of this understanding, we do not further explore practical inner stopping conditions.

Another interesting property of Algorithm 4 is that if the optimal solution of (21) is obtained at each outer iteration, then the algorithm can reach the optimum of (7) in finite steps even if neither Assumption 1 nor 2 is satisfied.

Theorem 6 *If at each iteration of Algorithm 4, the optimal solution of* (21) *is obtained, the optimum of* (1) *is reached within n iterations.*

The proof is in Appendix C.

These two theorems show that our algorithm can be treated as an analogy of the linear conjugate gradient method to general nonlinear optimization problems. See more details in Section 5.1

Algorithm 4: Common-directions method with multiple inner iterations

Given $\boldsymbol{w}_0$, compute $\nabla f(\boldsymbol{w}_0)$;
$P = [\nabla f(\boldsymbol{w}_0)/\|\nabla f(\boldsymbol{w}_0)\|]$;
for *k=0,1,... in the outer loop* **do**
  if $\nabla f(\boldsymbol{w}_k) = \boldsymbol{0}$, or some stopping condition is satisfied **then**
    stop;
  end
  $\boldsymbol{w} = \boldsymbol{w}_k$;
  while an inner stopping condition is not satisfied **do** `// inner loop`
    Obtain $\Delta\boldsymbol{t}$ by solving (22) with

$$(P^\top \nabla^2 f(\boldsymbol{w}) P)\Delta\boldsymbol{t} = -P^\top \nabla f(\boldsymbol{w});$$

    Backtracking line search on $f(\boldsymbol{w} + \theta P\Delta\boldsymbol{t})$ by Algorithm 2 to obtain θ;
    $\boldsymbol{w} = \boldsymbol{w} + \theta P\Delta\boldsymbol{t}$;
  end
  $\boldsymbol{w}_{k+1} = \boldsymbol{w}$;
  Compute $\nabla f(\boldsymbol{w}_{k+1})$;
  Let $\boldsymbol{p} = \nabla f(\boldsymbol{w}_{k+1}) - P(P^\top \nabla f(\boldsymbol{w}_{k+1}))$;
  if $\boldsymbol{p} \neq \boldsymbol{0}$ **then**
    $P = [P, \boldsymbol{p}/\|\boldsymbol{p}\|]$;
  end
end

4.2. Application on Regularized ERM Problems

For general problems, Algorithm 4 may not be beneficial because the cost of one inner iteration of solving (21) is similar to that of conducting an outer iteration of Algorithm 1, while (1) rather than (21) is what we actually would like to solve. However, for solving the regularized ERM problems (7), we show that one inner iteration of the while-loop in Algorithm 5 can be significantly cheaper than one outer iteration of Algorithm 3. We note that in the complexity analysis (16) for Algorithm 3, the dominant term $O(\#\text{nnz})$ is unavoidable for calculating $\nabla f(\boldsymbol{w}_{k+1})$. A crucial difference here is that this $O(\#\text{nnz})$ term is not needed. In Algorithm 3, the whole $\nabla f(\boldsymbol{w}_{k+1})$ vector is used for the Gram-Schmidt procedure in checking if P_k should be augmented. Instead, here P_k remains the same throughout inner iterations and all we need is $P_k^\top \nabla f(\boldsymbol{w}_k + P_k\boldsymbol{t})$ in (22). When we apply Algorithm 4 to regularized ERM problems, the linear system (22) has the following form.

$$(I + P_k^\top X^\top D_{\boldsymbol{w}_k + P_k\boldsymbol{t}} X P_k)\Delta\boldsymbol{t} = -(P_k^\top(\boldsymbol{w}_k + P_k\boldsymbol{t}) + P_k^\top X^\top \boldsymbol{v}_{\boldsymbol{w}_k + P_k\boldsymbol{t}}). \tag{23}$$

If

$$XP_k \ \text{ and } \ X(\boldsymbol{w}_k + P_k\boldsymbol{t}), \tag{24}$$

are maintained, the right-hand side can be easily calculated by

- $\boldsymbol{v}_{\boldsymbol{w}_k + P_k\boldsymbol{t}}$: $O(l)$,

- $P_k^\top(\boldsymbol{w}_k + P_k\boldsymbol{t})$: $O(mn)$,
- $(P_k^\top X^\top)\boldsymbol{v}_{\boldsymbol{w}_k+P_k\boldsymbol{t}}$: $O(lm)$.

Interestingly, for other operations in Algorithm 3, it has been shown that the two values in (24) can be easily maintained. Here we follow the same setting. A detailed procedure is in Algorithm 5. In summary, the cost per inner iteration is

$$O(lm^2 + mn + m^3). \tag{25}$$

In compared with the cost per outer iteration in Algorithm 3, here we do not have the #nnz term. The main difference comes from that in (23), we use the available XP_k to calculate

$$(P_k^\top X^\top)\boldsymbol{v}_{\boldsymbol{w}_k+P_k\boldsymbol{t}},$$

but in an outer iteration of Algorithm 3, $\nabla f(\boldsymbol{w}_{k+1})$ is calculated by

$$X^\top \boldsymbol{v}_{\boldsymbol{w}_{k+1}},$$

which costs $O(\#\text{nnz})$. Therefore, because in general #nnz is larger than (25), an inner iteration here is cheaper than an outer one in Algorithm 3.

5. Related Works

We compare our algorithm with linear conjugate gradient method for solving a linear system, as well as popular nonlinear optimization methods. The methods being discussed include those considering a linear combination of multiple directions to form the update direction, and those utilizing the Hessian matrix in various ways.

5.1. Linear Conjugate Gradient Method

Linear conjugate gradient (CG) method iteratively solves the following linear system.

$$A\boldsymbol{x} = \boldsymbol{b}, \tag{26}$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric and positive-definite, and $\boldsymbol{b} \in \mathbb{R}^n$. Alternatively, CG can be viewed as an optimization method for the following problem.

$$\min_{\boldsymbol{x}} \quad \frac{1}{2}\boldsymbol{x}^\top A\boldsymbol{x} - \boldsymbol{b}^\top \boldsymbol{x}. \tag{27}$$

The idea behind linear CG is that at each iteration, we generate a new update direction $\boldsymbol{p}_k$ that is conjugate to the existing ones such that

$$\boldsymbol{p}_i^\top A\boldsymbol{p}_j = 0, \ \ \forall i \neq j.$$

Then an exact line search is conducted to decide the next iterate. At each iteration, a matrix-vector product $A\boldsymbol{v}$ for some vector $\boldsymbol{v}$ is conducted to generate the desired step, and this matrix-vector product is the computational dominant part at each CG iteration.

Algorithm 5: Common-directions method with multiple inner iterations for the ERM problem (7)

Given $\boldsymbol{w}_0$, compute $\boldsymbol{z} = X\boldsymbol{w}_0$;
Use $\boldsymbol{z}$ to compute $\boldsymbol{v}_{\boldsymbol{w}_0}$ and $D_{\boldsymbol{w}_0}$ in (8) and (10);
Compute $\nabla f(\boldsymbol{w}_0) = \boldsymbol{w}_0 + X^\top \boldsymbol{v}_{\boldsymbol{w}_0}$ from (8);
$P = [\nabla f(\boldsymbol{w}_0)/\|\nabla f(\boldsymbol{w}_0)\|]$, $U = XP$;
for *k=0,1,... in the outer loop* **do**
  if $\nabla f(\boldsymbol{w}_k) = \boldsymbol{0}$, or some stopping condition is satisfied **then**
    stop;
  end
  $\boldsymbol{w} = \boldsymbol{w}_k, \boldsymbol{v}_{\boldsymbol{w}} = \boldsymbol{v}_{\boldsymbol{w}_k}, D_{\boldsymbol{w}} = D_{\boldsymbol{w}_k}$;
  while an inner stopping condition is not satisfied **do** `// inner loop`
    Obtain $\Delta\boldsymbol{t}$ by solving (23) with

$$\Delta\boldsymbol{t} = (I + U^\top D_{\boldsymbol{w}} U)^{-1}(-P^\top \boldsymbol{w} + U^\top \boldsymbol{v}_{\boldsymbol{w}});$$

    Compute $\hat{\boldsymbol{z}} = U\Delta\boldsymbol{t}$;
    Backtracking line search on $f(\boldsymbol{w} + \theta P\Delta\boldsymbol{t})$ by Algorithm 2 to obtain θ, using the values of $\boldsymbol{z}$ and $\hat{\boldsymbol{z}}$ to compute $\xi(y_i; \boldsymbol{z} + \theta\hat{\boldsymbol{z}})$;
    $\boldsymbol{w} = \boldsymbol{w} + \theta P\Delta\boldsymbol{t}$;
    $\boldsymbol{z} = \boldsymbol{z} + \theta(XP)\Delta\boldsymbol{t}$;
    Use $\boldsymbol{z}$ to compute $\boldsymbol{v}_{\boldsymbol{w}}$ and $D_{\boldsymbol{w}}$ in (8) and (10);
  end
  $\boldsymbol{w}_{k+1} = \boldsymbol{w}, \boldsymbol{v}_{\boldsymbol{w}_{k+1}} = \boldsymbol{v}_{\boldsymbol{w}}, D_{\boldsymbol{w}_{k+1}} = D_{\boldsymbol{w}}$;
  Compute $\nabla f(\boldsymbol{w}_{k+1}) = \boldsymbol{w}_{k+1} + X^\top \boldsymbol{v}_{\boldsymbol{w}_{k+1}}$ from (8);
  $\boldsymbol{p} = \nabla f(\boldsymbol{w}_{k+1}) - P(P^\top \nabla f(\boldsymbol{w}_{k+1}))$;
  if $\boldsymbol{p} \neq \boldsymbol{0}$ **then**
    $P = [P_k, \boldsymbol{p}/\|\boldsymbol{p}\|]$, $U = [U, X\boldsymbol{p}/\|\boldsymbol{p}\|]$;
  end
end

Therefore, the cost per iteration of CG is simply the cost of a matrix-vector product and several $O(n)$ operations on the vectors.

There are two nice properties of linear CG. First, when viewed as a solver for (26), it obtains the exact solution in n steps. Second, when viewed as a solver for (27), its iteration complexity for an ϵ-accurate solution is $O(\sqrt{\kappa}\log(1/\epsilon))$. See, for example, Chapter 5.1 of Nocedal and Wright (2006). Because the method proposed in Section 4 possesses the same two properties in Theorems 5 and 6, it can be viewed as an extension of linear CG to general nonlinear convex optimization.

5.2. First-order Methods and Momentum

For nonlinear convex optimization, the simplest first-order method is gradient descent. In gradient descent, the update direction is always the negative gradient $-\nabla f(\boldsymbol{w})$. It has been

shown in many standard convex optimization books such as Boyd and Vandenberghe (2004); Nocedal and Wright (2006); Nesterov (2003) that with a proper choice of the step size, to obtain an ϵ-accurate solution for a given $\epsilon > 0$, gradient descent requires $O(\kappa \log(1/\epsilon))$ iterations. However, this is far from the aforementioned optimal convergence rate $O(\sqrt{\kappa}\log(1/\epsilon))$ for methods that have their update directions in the span of all previous gradients.

Methods like the heavy-ball method (Polyak, 1964) and nonlinear CG try to improve the convergence by using the "momentum." These methods possess update rules in the form

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k) + \beta_k(\boldsymbol{w}_k - \boldsymbol{w}_{k-1}), \tag{28}$$

where α_k and β_k vary in different methods, and the term $\boldsymbol{w}_k - \boldsymbol{w}_{k-1}$ is the so-called momentum. By induction we can easily see that $\boldsymbol{w}_k - \boldsymbol{w}_{k-1}$ in (28) is a linear combination of $\nabla f(\boldsymbol{w}_0), \nabla f(\boldsymbol{w}_1), \ldots, \nabla f(\boldsymbol{w}_{k-1})$, and therefore these methods fall in the category of picking the update direction from the span of the previous gradients. Nonlinear CG has many different variants that consider different strategies based on $\boldsymbol{w}_k$, $\boldsymbol{w}_{k-1}$, $\nabla f(\boldsymbol{w}_k)$, $\nabla f(\boldsymbol{w}_{k-1})$ and even $\nabla^2 f(\boldsymbol{w}_k)$, and all of them reduce to linear CG when the problem is (27). Interested readers are referred to Hager and Zhang (2006) and the references therein. When f is quadratic and strongly convex, it can be shown that both the heavy-ball method with proper parameters (Lessard et al., 2016) and nonlinear CG methods can achieve the optimal convergence rate for first-order methods,[3] but for non-quadratic functions, the situation is different. Some variants of nonlinear CG can be shown to converge asymptotically, but the rate is unclear (Hager and Zhang, 2006, Theorem 5.1). On the other hand, Ghadimi et al. (2014) showed that for problems satisfying Assumption 1, with properly chosen parameters, the heavy-ball method converges globally R-linearly. If in addition the problem is twice-differentiable, Polyak (1987) showed that when the iterate is close enough to the optimum, the local R-linear convergence rate of the heavy-ball method matches the optimal linear convergence rate of first-order methods for a specific choice of α_k and β_k.

Nesterov's accelerated gradient (Nesterov, 1983) considers two sequences $\{\boldsymbol{w}_k\}$ and $\{\boldsymbol{s}_k\}$ with the following update rules.

$$\begin{aligned}\boldsymbol{w}_{k+1} &\equiv \boldsymbol{s}_k - \frac{1}{\rho}\nabla f(\boldsymbol{s}_k),\\ \boldsymbol{s}_{k+1} &= (1 + \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1})\boldsymbol{w}_{k+1} - \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\boldsymbol{w}_k.\end{aligned}$$

Interestingly, this method uses only the previous gradients in a deterministic way with κ to be the only data-dependent value, but it can guarantee the optimal iteration complexity of first-order methods for problems satisfying Assumption 1. A problem in this method is that we need to know the values ρ and σ in advance, and these values may not be available. To solve this problem, a variant of Nesterov's accelerated gradient is proposed in Nesterov (2013). This algorithm estimates these parameters at each iteration via an iterative procedure similar to backtracking line search, and achieves the same iteration complexity with little additional cost. Note that for all these first-order methods, the main bottleneck is the computation of the gradients, and the cost is proportional to the number of different gradients computed per iteration.

3. Section 1 of Hager and Zhang (2006) suggests that nonlinear CG methods reduce to linear CG when f is quadratic, and the convergence rate follows what we described in Section 5.1.

Our method can be viewed as a generalization of these approaches, in the sense that the same search space for the update direction is considered. The major difference is that these momentum methods are restricted to some deterministic or pre-specified ways of combination, while our method considers combining multiple directions with adaptive weights according to the data that can and will change over iterations. Another difference is that we utilize higher-order derivatives to decide the combination coefficient, so quadratic convergence in Theorem 4 can be obtained.

5.3. Methods Utilizing Multiple Directions

The cutting plane method (Kelley, 1960) and the bundle method (Teo et al., 2010) use all gradients obtained from the first iteration on. They differ from our method from using the previous gradients to construct a piecewise linear function to approximate (the hard-to-compute part of) the actual function.[4] Therefore, the purpose of using previous gradients in these methods and that of ours are different. An advantage of the cutting plane method and the bundle method is the ability to deal with nonsmooth problems by using subgradients. When being applied to problems satisfying Assumption 1, the bundle method converges linearly (Teo et al., 2010). However, optimizing a piecewise linear function with a growing number of pieces is more expensive than dealing with one smooth function. Our method considers easier-to-solve subproblems at each iteration. Moreover, our method converges quadratically while the bundle method and the cutting plane method do not have this property.

Quasi-Newton methods also consider previous momentums and gradients, but the purpose is to approximate the real Hessian matrix via these vectors in order to achieve superlinear convergence. Among different quasi-Newton methods, the BFGS method (Dennis and Moré, 1977) and its limited-memory variant, L-BFGS (Liu and Nocedal, 1989), are the most widely used ones. It is known that the BFGS method admits local superlinear convergence and converges globally; see, for example, (Nocedal and Wright, 2006, Chapter 6). By applying a proper line search procedure, L-BFGS has global linear convergence for problems satisfying Assumption 1.[5] Note that because older information are discarded, L-BFGS does not have superlinear convergence, but it has the advantage of less spatial cost and is more suitable for large-scale problems.

At the k-th iteration for any $k \geq 1$, given the current iterate $\boldsymbol{w}_k$, BFGS solves the following quadratic problem.

$$\min_{\boldsymbol{p}_k} \quad \nabla f(\boldsymbol{w}_k)^\top \boldsymbol{p}_k + \frac{1}{2}(\boldsymbol{p}_k)^\top B_k \boldsymbol{p}_k, \tag{29}$$

where B_k is a positive definite symmetric approximation of $\nabla^2 f(\boldsymbol{w}_k)$ such that

$$B_k^{-1} = (I - \mu_{k-1}\boldsymbol{u}_{k-1}\boldsymbol{s}_{k-1}^\top)^\top B_{k-1}^{-1}(I - \mu_{k-1}\boldsymbol{u}_{k-1}\boldsymbol{s}_{k-1}^\top) + \mu_{k-1}\boldsymbol{s}_{k-1}\boldsymbol{s}_{k-1}^\top, \tag{30}$$

4. It is known in functional analysis that any convex function can be approximated by piecewise linear functions to any precision.

5. Note that the proof that L-BFGS converges globally linearly utilized the property of using a bounded number of historical vectors and thus does not apply to BFGS.

with

$$\mu_{k-1} \equiv \frac{1}{\boldsymbol{u}_{k-1}^\top \boldsymbol{s}_{k-1}}, \quad \boldsymbol{s}_{k-1} = \boldsymbol{w}_k - \boldsymbol{w}_{k-1}, \quad \boldsymbol{u}_{k-1} = \nabla f(\boldsymbol{w}_k) - \nabla f(\boldsymbol{w}_{k-1}).$$

Note that although B_0 can be decided arbitrarily, the most common choice is

$$B_0 = \beta I \tag{31}$$

for some $\beta > 0$. The solution of (29) is clearly

$$\boldsymbol{p}_k = -B_k^{-1} \nabla f(\boldsymbol{w}_k). \tag{32}$$

If (31) is used, this operation can be conducted cheaply with only $O(kn)$ cost by a sequence of vector-vector operations obtained from expanding (30).[6] After deciding $\boldsymbol{p}_k$, a line search procedure is conducted to find a suitable step size θ_k to ensure that the sequence $\{\boldsymbol{w}_k\}$ converges to the optimal solution of f.

With the choice of (31), by induction the vector

$$\boldsymbol{s}_k = \alpha_{k+1} \boldsymbol{p}_k \tag{33}$$

is a linear combination of $\nabla f(\boldsymbol{w}_0), \ldots, \nabla f(\boldsymbol{w}_k)$ for all k.[7] Thus, we can see that $\boldsymbol{p}_k \in \mathtt{span}\{\nabla f(\boldsymbol{w}_0), \ldots, f(\boldsymbol{w}_k)\}$ and thus the BFGS algorithm only utilizes the previous gradients, but is able to achieve superlinear convergence. Interestingly, our algorithm utilizes the same information as BFGS, but because we also consider the Hessian to search for the best update direction in $\mathtt{span}\{\nabla f(\boldsymbol{w}_0), \ldots, \nabla f(\boldsymbol{w}_k)\}$, our method is better than BFGS for being able to achieve quadratic convergence.

A major drawback of the BFGS algorithm is that it requires storing many vectors (or an $O(n^2)$ dense matrix) in memory. The memory consumption can be prohibitively expensive after iterations. This issue also happens in our method, but we notice that in practice, it is not severe. One reason is that the memory capacity is large enough to store the required information unless the number of iterations is high, and both our method and BFGS converge fast enough such that the memory capacity is not reached. For extremely large data, the memory issue may be faced, but this issue is less problematic in our method than in BFGS. The reason is that at each iteration of BFGS, two new vectors are added to memory, while in our method, at most one vector is added.

To handle the memory consumption problem of BFGS, its limited-memory variant, L-BFGS (Liu and Nocedal, 1989), is proposed. L-BFGS uses a different setting of B_k^{-1}. Given an integer $m > 0$, at the k-th iteration, L-BFGS only uses $\boldsymbol{s}_i, \boldsymbol{u}_i, i = k-m, k-m+1, \ldots, k-1$ and $\nabla f(\boldsymbol{w}_k)$ to construct the approximate Hessian inverse. For L-BFGS, in the recursive computation of (30), the matrix B_{k-m}^{-1} can be replaced by any matrix. According

6. If B_0 has neither sparsity nor some special structures, the cost of computing (32) at each iteration is $O(n^2)$ by directly maintaining (30), and it does not fall in the category of algorithms that only use gradients anymore because the vector $B_0^{-1}\nabla f(\boldsymbol{w})$ is considered.
7. By continually expanding (32) using (31) and (30), we can see that $\boldsymbol{p}_k$ is a linear combination of $\boldsymbol{s}_i, \boldsymbol{u}_i, i = 1, \ldots, k-1$ and $\nabla f(\boldsymbol{w}_k)$. By induction, if all $\boldsymbol{s}_i$ are linear combinations of previous gradients, then with the definition of $\boldsymbol{u}_i$, $\boldsymbol{p}_k$ is also a linear combination of previous gradients. The induction is finished by that $\boldsymbol{s}_k$ is a multiple of $\boldsymbol{p}_k$ in (33).

to (Nocedal and Wright, 2006, Chapter 7), a choice that has been shown to be practically effective is

$$\frac{\boldsymbol{s}_{k-1}^T \boldsymbol{u}_{k-1}}{\|\boldsymbol{u}_{k-1}\|^2} I. \tag{34}$$

This is also the choice we use in our implementation for empirical comparisons. As mentioned earlier, L-BFGS has only linear but not superlinear convergence. Nevertheless, it might outperform BFGS practically because the computation cost per iteration and the memory access amount are both lower. Moreover, it is shown in Bubeck et al. (2015) that L-BFGS outperforms those methods mentioned in Section 5.2 empirically, although the coefficient of its linear convergence is not clear.

5.4. Newton Method

At each iteration, given the current iterate $\boldsymbol{w}$, the Newton method obtains the update direction by solving

$$\min_{\boldsymbol{p} \in \mathbb{R}^n} \quad \boldsymbol{p}^\top \nabla f(\boldsymbol{w}) + \frac{1}{2} \boldsymbol{p}^\top \nabla^2 f(\boldsymbol{w}) \boldsymbol{p}. \tag{35}$$

To obtain the solution of (35), the following n by n linear system is solved.

$$\nabla^2 f(\boldsymbol{w}) \boldsymbol{p} = -\nabla f(\boldsymbol{w}) \quad \Rightarrow \quad \boldsymbol{p} = -\nabla^2 f(\boldsymbol{w})^{-1} \nabla f(\boldsymbol{w}). \tag{36}$$

Then a line search procedure is conducted to ensure the sufficient function decrease. It is known that for twice-differentiable functions satisfying Assumption 2, the Newton method with line search has local quadratic convergence (Boyd and Vandenberghe, 2004; Nocedal and Wright, 2006). However, The computational cost per Newton iteration for directly solving the linear system (36) is as expensive as $O(n^3)$. Moreover, solving (36) precisely may not be beneficial when the current iterate is far from optimum. Therefore, truncated Newton methods that utilize, for example, the iterative linear CG method we discussed in Section 5.1 to approximately solve (36) is often used. In general, the computational cost per CG iteration is $O(n^2)$ for conducting the Hessian-vector product provided the Hessian is available. When we only want to solve the problem approximately, usually the required CG iteration is much smaller than n, making the computation of a truncated Newton step much cheaper than that of a full Newton step. It is known that truncated Newton steps combined with a line search procedure or trust region techniques retain local quadratic convergence for problems satisfying Assumption 2 and have superlinear convergence for problems satisfying Assumption 1, if the residual of the solution to (36) goes to zero as the number of truncated Newton iterations goes to infinity. See, for example, Lin and Moré (1999) and (Nocedal and Wright, 2006, Theorem 7.2).

As mentioned above, the cost of directly computing the Hessian-vector product is $O(n^2)$ both spatially and computationally if we store the Hessian matrix explicitly. However, as Keerthi and DeCoste (2005); Lin et al. (2008) proposed, for ERM problems in (7), if we calculate the product between Hessian and a given vector $\boldsymbol{v}$ by

$$\nabla^2 f(\boldsymbol{w}) \boldsymbol{v} = (I + X^\top D_{\boldsymbol{w}} X) \boldsymbol{v} = \boldsymbol{v} + \left(X^\top \left(D_{\boldsymbol{w}} \left(X \boldsymbol{v} \right) \right) \right), \tag{37}$$

this computation costs only $O(\#\text{nnz} + n)$ per CG iteration, where #nnz is at most ln but is usually much smaller for sparse data. Further, no storage space is needed for the

operation in (37). On the other hand, explicitly forming the Hessian matrix in this case costs $O(\#\text{nnz}^2)$. Therefore, for regularized ERM problems, the CG approach without explicitly computing the Hessian matrix is more preferable. In this sense, truncated Newton methods costs

$$O((\#\text{nnz} + n) \times \text{CG iterations}) \tag{38}$$

per Newton iteration. A comparison between (38) and (20) shows that by a careful utilization of the problem structure of ERM, we have a cheaper cost of running one Newton iteration to approximately minimize a sub-problem over a subspace. The cost of our Newton iteration is only about that of one CG iteration here. We will examine the relative training speed between the truncated Newton method and our algorithm by experiments in Section 6.

6. Experiments

We examine the empirical speed of our algorithm for solving two L2-regularized ERM problems. The first one is logistic regression, in which $\xi(\cdot)$ in (7) is defined by

$$\xi(y; z) \equiv \log\left(1 + \exp\left(-yz\right)\right),$$

and the second one is the squared-hinge loss problem, where $\xi(\cdot)$ is

$$\xi(y; z) \equiv \max\left\{1 - yz, 0\right\}^2.$$

In both cases, $y \in \{-1, 1\}$. We first compare the approach of conducting only one inner iteration discussed in Section 2 with that of conducting multiple inner iterations illustrated in Section 4 using the logistic regression problems. The relative efficiency between our methods and the methods reviewed in Section 5 is then examined on both loss functions.

6.1. Experiment Settings

We consider the data sets listed in Table 1 with three different choices of parameters $C = \{10^{-3}, 1, 10^3\}$ to examine the situation of different condition numbers. All data sets except yahoo-japan and yahoo-korea are publicly available.[8] We use the relative distance to the optimal objective value, which is defined below, to compare different optimization approaches.

$$\left|\frac{f(\boldsymbol{w}) - f(\boldsymbol{w}^*)}{f(\boldsymbol{w}^*)}\right|,$$

where $\boldsymbol{w}^*$ is the optimal solution obtained by running our algorithm long enough. All experiments are conducted on a 64-bit machine with Intel Xeon 2.0 GHz CPU (E5504), 4MB cache, and 32GB memory.

We consider two different criteria for comparison. The first one is the empirical training time. This is the most important criterion in single-machine training for most users. We also consider the number of data passes, which is proportional to the number of iterations[9] and is thus an indicator of iteration complexity. This criterion can also serve as a training time estimate for scenarios that the cost is highly dependent to the number of passes through the data, such as distributed machine learning, or disk-level classification (Yu et al., 2012). In addition, this criterion excludes the effect of implementation differences.

8. http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets.
9. For the Newton method, this is the number of inner iterations in CG plus the number of Newton iterations.

Data	Training size (l)	Features (n)	Density ($\#\text{nnz}/(ln)$)
a9a	32,561	123	11.2757%
covtype	581,012	54	21.9979%
news20	19,996	1,355,191	0.0336%
url	2,396,130	3,231,961	0.0036%
epsilon	400,000	2000	100.0000%
webspam	350,000	16,609,143	0.0224%
yahoo-japan	140,963	832,026	0.0160%
yahoo-korea	368,444	3,053,939	0.0111%
rcv1t	677,399	47,236	0.1549%
KDD2010-b	19,264,097	29,890,095	0.0001%

Table 1: Statistics of data sets.

6.2. Using Single or Multiple Inner Iterations

We compare on logistic regression problems the performance of the two variants of our algorithms, namely the approach of using only a single inner iteration and the variant of using multiple inner iterations, in Figures 1-6. Note that the result of KDD2010-b with $C = 10^3$ is not shown because the training time is too long. For the backtracking line search procedure in Algorithm 2, we set $\beta = 0.4, \lambda = 0.25$. The solution to the linear system (14) is obtained by a Cholesky factorization with partial pivoting. For the multiple inner iteration variant, we use the following heuristic stopping condition for the inner loop in Algorithm 4.

$$\|P_k \nabla f(\boldsymbol{w})\| \leq \min(0.1\|\nabla f(\boldsymbol{w}_k)\|, \|\nabla f(\boldsymbol{w}_k)\|^2).$$

We can see that the two approaches perform quite similarly in both criteria, while the single inner iteration approach is slightly better in the training time. The figures of data passes indicate that using a single inner iteration has similar function value decrease per iteration to using multiple inner iterations, so the additional cost of multiple inner iterations sometimes results in slightly longer training time. Therefore, we use the single iteration variant in later experiments of comparing with existing algorithms.

6.3. Comparison Between Different Methods

We compare the following different algorithms for training L2-regularized smooth empirical risk minimization problems.

- CommDir: our common-directions method with single inner iteration.

- Line search truncated Newton method (NEWTON): at each Newton iteration, (35) is approximately solved by linear CG, until CG generates an approximate solution $\boldsymbol{p}$ of (35) that satisfies
$$\|\nabla f(\boldsymbol{w}) - \nabla^2 f(\boldsymbol{w})\boldsymbol{p}\| \leq 0.1\|\nabla f(\boldsymbol{w})\|.$$
We follow the discussion in Section 5 to efficiently compute the Hessian-vector products in $O(n + \#\text{nnz})$ time. The implementation is modified from that in LIBLINEAR (Fan et al., 2008).

- L-BFGS: We implement the L-BFGS algorithm by considering different numbers of history states $m = 5, 10, 15, 20, 30$.[10] For the first step, where no historical information is available, we follow the implementation by the authors of Liu and Nocedal (1989) to take the initial step size for the gradient direction as $1/\|\nabla f(\boldsymbol{w}_0)\|$. For later iterations, we take the matrix defined in (34) as B_{k-m}^{-1}. In general $m = 30$ performs the best, so we report results of using this number of states.

- BFGS: we use our L-BFGS implementation and set m to infinity to obtain a BFGS solver.

- Nesterov's accelerated gradient (AG): we consider the approach proposed by Nesterov (2013) that adaptively estimates the parameters σ and ρ by a procedure similar to backtracking line search. We take the initial estimation to be $\sigma = 1$ and $\rho = 1/Cl$.

For NEWTON, L-BFGS and BFGS that require a line search procedure to ensure the convergence, we follow our method to apply Algorithm 2 with the same parameters $\beta = 0.4$ and $\lambda = 0.25$ used in Section 6.2. We exploit the trick discussed in Section 3.2 to efficiently conduct this line search procedure. Note that for all methods except AG, evaluating $X^T\boldsymbol{v}_{\boldsymbol{w}}$ in (8) for gradient at each iteration requires one data pass, and the line search procedure requires another one to maintain $X\boldsymbol{w}$. For NEWTON, each CG iteration requires one data pass. For AG, each inner iteration for adaptively estimating σ and ρ requires two data passes to compute related values. All algorithms are implemented in C++.

The results on logistic regression problems are shown in Figures 7-9 for running time and Figures 10-12 for number of data passes. Note that the result of KDD2010-b with $C = 10^3$ is not shown because the training time is too long. We can see that our method has the fewest data passes in all settings and all data sets. This observation suggests the usefulness of our method in the scenario that the data passes are expensive. The number of data passes is significantly reduced because we use the ideas in Section 3 to store the inner products XP_k to avoid redundant data accesses. In terms of training time, our method is among the fastest. The exceptions are a9a and rcv1t whose l are much larger than n, and the largest data set KDD2010-b, which from Table 1 is highly sparse. From the result of data passes, we see that our method requires fewer iterations than other methods, in the cost of the $O(lm^2 + mn)$ operations per iteration in addition to the $O(\#\text{nnz})$ cost required by all methods to conduct one data pass; see (20). When $O(lm^2 + mn) \ll O(\#\text{nnz})$, this additional cost is negligible and hence the advantage of fewer iterations dominates.

10. We also tried the implementation by the authors of Liu and Nocedal (1989) at `http://users.iems.northwestern.edu/~nocedal/lbfgs.html`, but found that it is not as efficient as ours. Note that their implementation does not utilize the efficient line search procedure discussed in Section 3.2.

However, when n is relatively small, or when the data set is highly sparse, the $O(lm^2 + mn)$ cost is significant in compared with $O(\#\text{nnz})$, so our method may be slower for these data sets.

We then show results on squared-hinge loss problems in Figures 13-15 for running time and in Figures 16-18 for data passes. Similar trends of superiority of our method in data passes are still observed in squared-hinge loss problems, but in general the squared-hinge loss problems are harder to optimize for all methods. A possible reason is that logistic loss is infinitely continuously differentiable but squared-hinge loss is only once continuously differentiable. This difficulty in optimization of squared-hinge loss problems is reflected in running time as well. Since our common-directions method has cost that grows superlinearly with the number of iterations passed, its relative running time performance is also worse on squared-hinge loss problems than that on logistic regression problems.

The comparison here between NEWTON and L-BFGS is inconsistent with that in Lin et al. (2008). A further investigation showed that the major difference is from the choice of m. In Lin et al. (2008), $m = 5$ is used, while we observe that setting larger m like 30 here in most cases leads to faster convergence in terms of time, as long as the memory capacity is not an issue. Another minor difference is that each line search iteration of L-BFGS in the experiments of Lin et al. (2008) requires one data pass while we use the idea of caching inner products to accelerate the line search in our experiments. Moreover, in Lin et al. (2008), instead of line search, a trust region approach is considered for the truncated Newton method.

An interesting observation in our experimental results is the impressive performance of BFGS. Although BFGS has better theoretical convergence than L-BFGS, in practice people tend to use L-BFGS because BFGS consumes too much memory. Our experiments indicate that with larger memory and better computation power nowadays, BFGS becomes feasible and can outperform L-BFGS in some cases. However, it is also observed that when both the number of iterations to converge and n are large, such as the cases of webspam and url with $C = 1000$, BFGS still suffers from the memory problem.

7. Discussion

As shown in our experiments, our method has a smaller number of data passes than all other methods. This property implies that our method has at least two potential applications for solving extremely large-scale ERM problems. The first one is disk-level classification when the data cannot fit into memory, and the second one is distributed machine learning. In both tasks, each data pass involves expensive operations like intensive disk I/O or between-machine communication. With fewer data passes, our method is suitable for these tasks to tackle the bottleneck.

Each data pass in disk-level classification requires expensive disk I/O to access data because we cannot store all of them in memory. Existing methods for disk-level classification (Yu et al., 2012; Chang and Roth, 2011) consider solving the dual problem with techniques involving shuffling and partitioning data into blocks. Each block can be fully loaded into memory and these methods work on one block at a time. The random shuffling step gives better convergence, but requires lengthy pre-processing time because of heavy disk I/O. In contrast, our method does not require the random shuffling of data.

To reduce the number of data passes, Chang and Roth (2011) considered caching important instances in memory. Our method also uses ideas similar to caching, while the information being kept in memory is not some training instances, but the inner products XP_k. The number of these inner products is roughly the same as the number of iterations, which is usually not too large. Therefore, the memory consumption of our method tends to be smaller, and hence our method can cache more important information to reduce the data I/O.

For distributed machine learning that uses multiple machines, each data pass requires an expensive communication of a vector proportional to the data size to synchronize information among machines. Usually this step is the bottleneck of distributed machine learning. With fewer data passes and thus fewer rounds of communication, our method can be expected to have shorter training time than other algorithms in distributed environments.

8. Conclusions

In this paper, we propose a new optimization algorithm that utilizes the Hessian information to combine different update directions. Theoretically, our method has global linear and local quadratic convergence, while the cost per iteration is not expensive for solving ERM problems. Empirically, extensive experimental results show that for solving ERM problems, our method is competitive with state-of-the-art methods in terms of training time, but outperforms them in terms of data passes. After the first version of this paper, we have studied possible choices of common directions other than the gradients for distributed training in Lee et al. (2017).

Acknowledgments

The implementation of Nesterov's accelerated gradient is modified from an unreleased C# experimental code provided by Lin Xiao. C.-J. Lin was supported in part by the Ministry of Science and Technology of Taiwan via the grants 107-2221-E-002-167-MY3.

References

Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.

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

Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.

Sébastien Bubeck, Yin Tat Lee, and Mohit Singh. A geometric alternative to Nesterov's accelerated gradient descent. Technical report, 2015. arXiv preprint arXiv:1506.08187.

Kai-Wei Chang and Dan Roth. Selective block minimization for faster convergence of limited memory large-scale linear models. In *Proceedings of the Seventeenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2011.

John E. Dennis, Jr and Jorge J. Moré. Quasi-Newton methods, motivation and theory. *SIAM Review*, 19(1):46–89, 1977.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: a library for large linear classification. *Journal of Machine Learning Research*, 9: 1871–1874, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/liblinear.pdf`.

Reeves Fletcher and Colin M Reeves. Function minimization by conjugate gradients. *The Computer Journal*, 7(2):149–154, 1964.

Euhanna Ghadimi, Hamid Reza Feyzmahdavian, and Mikael Johansson. Global convergence of the heavy-ball method for convex optimization. Technical report, 2014. arXiv preprint arXiv:1412.7457.

William W Hager and Hongchao Zhang. A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimization*, 2(1):35–58, 2006.

S. Sathiya Keerthi and Dennis DeCoste. A modified finite Newton method for fast solution of large scale linear SVMs. *Journal of Machine Learning Research*, 6:341–361, 2005.

James E. Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 1960.

Ching-Pei Lee, Po-Wei Wang, Weizhu Chen, and Chih-Jen Lin. Limited-memory common-directions method for distributed optimization and its application on empirical risk minimization. In *Proceedings of SIAM International Conference on Data Mining*, 2017.

Laurent Lessard, Benjamin Recht, and Andrew Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1): 57–95, 2016.

Chih-Jen Lin and Jorge J. Moré. Newton's method for large-scale bound constrained problems. *SIAM Journal on Optimization*, 9:1100–1127, 1999.

Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi. Trust region Newton method for large-scale logistic regression. *Journal of Machine Learning Research*, 9:627–650, 2008. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/logistic.pdf`.

Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528, 1989.

Olvi L. Mangasarian. A finite Newton method for classification. *Optimization Methods and Software*, 17(5):913–929, 2002.

Yurii E. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27:372–376, 1983.

Yurii E. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, 2003.

Yurii E. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer, second edition, 2006.

Boris T. Polyak. *Introduction to Optimization*. Translation Series in Mathematics and Engineering. 1987.

Boris Teodorovich Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

Choon Hui Teo, S.V.N. Vishwanathan, Alex Smola, and Quoc V. Le. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 11:311–365, 2010.

Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, NY, 1995.

Hsiang-Fu Yu, Cho-Jui Hsieh, Kai-Wei Chang, and Chih-Jen Lin. Large linear classification when data cannot fit in memory. *ACM Transactions on Knowledge Discovery from Data*, 5(4):23:1–23:23, February 2012. URL `http://www.csie.ntu.edu.tw/~cjlin/papers/kdd_disk_decomposition.pdf`.

Appendix A. Analysis of Common-directions Method with Single Inner Iteration

We will show that under Assumptions 1, Algorithm 1 converges linearly by Theorem 3. If in addition Assumption 2 is satisfied, Algorithm 1 converges quadratically by Theorem 4.

Lemma 7 *Let P be an n by m real matrix such that $P^\top P = I$. Then*

$$\|P^\top \boldsymbol{v}\| \le \|\boldsymbol{v}\|, \ \forall \boldsymbol{v} \in \mathbb{R}^n.$$

Further, equality holds if and only if

$$\boldsymbol{v} \in \mathtt{span}(P) \equiv \{P\boldsymbol{t} \mid \forall \boldsymbol{t} \in \mathbb{R}^m\}.$$

Proof Note that $\boldsymbol{v}$ can be decomposed as

$$\boldsymbol{v} = PP^\top \boldsymbol{v} + (I - PP^\top)\boldsymbol{v}. \tag{39}$$

Taking norm at both sides and applying $P^\top P = I$, there is

$$\|\boldsymbol{v}\|^2 = \|P^\top \boldsymbol{v}\|^2 + \|(I - PP^\top)\boldsymbol{v}\|^2 \ge \|P^\top \boldsymbol{v}\|^2. \tag{40}$$

Thus, the first result follows. For the second result, we know that $\boldsymbol{v}$ is in the span of P if and only if its orthogonal projection to the span of P is itself. Therefore,

$$\boldsymbol{v} \in \mathtt{span}(P) \iff \boldsymbol{v} = PP^\top \boldsymbol{v}.$$

Combining this with (39), we obtain

$$\boldsymbol{v} \in \mathtt{span}(P) \implies (I - PP^\top)\boldsymbol{v} = \boldsymbol{0} \implies \|P^\top \boldsymbol{v}\| = \|\boldsymbol{v}\|.$$

Similarly, with (40), we have

$$\|P^\top \boldsymbol{v}\| = \|\boldsymbol{v}\| \implies \|(I - PP^\top)\boldsymbol{v}\| = 0 \implies \boldsymbol{v} \in \mathtt{span}(P).$$

Thus, the second result follows from the above two statements. ■

Lemma 8 *For every iteration k of Algorithm 1, we have*

$$P_k^\top P_k = I, \tag{41}$$

and

$$\|P_k^\top \nabla f(\boldsymbol{w}_k)\| = \|\nabla f(\boldsymbol{w}_k)\|. \tag{42}$$

Proof The augmenting of P in Algorithm 1 is simply the Gram-Schmidt process, and thus (41) holds. Further, from the fact that $(I - P_{k-1}P_{k-1}^\top)\nabla f(\boldsymbol{w}_k)$ is used to generate P_k, we have that $\nabla f(\boldsymbol{w}_k) \in \mathtt{span}(P_k)$, so (42) follows from Lemma 7. ■

Lemma 9 *Under Assumption 1,*

$$\sigma I \preceq P_k^\top \nabla^2 f(\boldsymbol{w}_k) P_k \preceq \rho I. \tag{43}$$

Proof Assumption 1 indicates that

$$\sigma I \preceq \nabla^2 f(\boldsymbol{w}) \preceq \rho I, \ \forall \boldsymbol{w}. \tag{44}$$

Combining (44) and (41) in Lemma 8, one has that for any $\boldsymbol{t}$,

$$\begin{aligned} \boldsymbol{t}^\top P_k^\top \nabla^2 f(\boldsymbol{w}_k) P_k \boldsymbol{t} &\ge \sigma \|P_k \boldsymbol{t}\|^2 = \sigma \|\boldsymbol{t}\|^2, \\ \boldsymbol{t}^\top P_k^\top \nabla^2 f(\boldsymbol{w}_k) P_k \boldsymbol{t} &\le \rho \|P_k \boldsymbol{t}\|^2 = \rho \|\boldsymbol{t}\|^2, \end{aligned} \tag{45}$$

which then imply (43). ■

Lemma 10 *Under Assumption 1,*

$$\frac{1}{2\rho}\|\nabla f(\boldsymbol{w})\|^2 \le f(\boldsymbol{w}) - f^* \le \frac{1}{2\sigma}\|\nabla f(\boldsymbol{w})\|^2, \tag{46}$$

where f^* is the optimal objective value.

Proof The first inequality is from Nesterov (2003, Theorem 2.1.5) together with that

$$\nabla f(\boldsymbol{w}^*) = \mathbf{0},$$

where $\boldsymbol{w}^*$ is the optimal solution. The second inequality follows from Nesterov (2003, Theorem 2.1.10). ■

Proof of Theorem 3

We will show that the update direction does not deviate from the steepest descent direction too much, and the line search procedure will then ensure the sufficient decrease of the function value for linear convergence. Let

$$\boldsymbol{t} = -(P_k^\top \nabla^2 f(\boldsymbol{w}_k) P_k)^{-1} P_k^\top \nabla f(\boldsymbol{w}_k). \tag{47}$$

Note that $(P_k^\top \nabla^2 f(\boldsymbol{w}_k) P_k)$ is invertible because of Lemma 9.

We first show that the step size produced in the line search procedure in Algorithm 2 is lower-bounded. From (44) via Assumption 1, for any $\bar{\theta} > 0$, we have

$$f(\boldsymbol{w}_k + \bar{\theta} P_k \boldsymbol{t}) \leq f(\boldsymbol{w}_k) + \bar{\theta} \nabla f(\boldsymbol{w}_k)^\top P_k \boldsymbol{t} + \frac{\rho}{2} \bar{\theta}^2 \|P_k \boldsymbol{t}\|^2. \tag{48}$$

If

$$-\bar{\theta} \nabla f(\boldsymbol{w}_k)^\top P_k \boldsymbol{t} - \frac{\rho}{2} \bar{\theta}^2 \|P_k \boldsymbol{t}\|^2 \geq \frac{\lambda}{2} \bar{\theta}^2 \|P_k \boldsymbol{t}\|^2, \tag{49}$$

then with (48) the stopping condition of Algorithm 2 is satisfied. For $\bar{\theta} > 0$, (49) is equivalent to

$$\bar{\theta} \leq \frac{-\nabla f(\boldsymbol{w}_k)^\top P_k \boldsymbol{t}}{\|P_k \boldsymbol{t}\|^2} \frac{2}{\rho + \lambda}. \tag{50}$$

From (41) of Lemma 8,

$$\|P_k \boldsymbol{t}\| = \|\boldsymbol{t}\|. \tag{51}$$

Thus, by the definition of $\boldsymbol{t}$ in (47) and Lemma 9,

$$\frac{-\nabla f(\boldsymbol{w}_k)^\top P_k \boldsymbol{t}}{\|P_k \boldsymbol{t}\|^2} = \frac{\boldsymbol{t}^\top (P_k^\top \nabla^2 f(\boldsymbol{w}_k) P_k) \boldsymbol{t}}{\|\boldsymbol{t}\|^2} \geq \sigma, \tag{52}$$

whenever $\nabla f(\boldsymbol{w}_k) \neq \mathbf{0}$. Then we have that (49), or equivalently (50), holds whenever

$$\bar{\theta} \leq \frac{2\sigma}{\rho + \lambda}.$$

This ensures the finite termination of Algorithm 2 and guarantees that the generated step size θ_k satisfies

$$\theta_k \geq \min(1, \frac{2\beta\sigma}{\rho + \lambda}) \geq \frac{\beta\sigma}{\rho + \lambda}, \tag{53}$$

where the second inequality is from $\beta < 1$, $\lambda > 0$, and $\sigma \leq \rho$. Therefore, the line-search procedure terminates after at most $\lceil \log_\beta(\beta\sigma/(\rho + \lambda)) \rceil$ steps.

Next we prove the linear convergence. By (51), the definition of $\boldsymbol{t}$ in (47), Lemma 9, and (42) of Lemmas 8, we have

$$\begin{aligned} &\|P_k \boldsymbol{t}\|^2 = \|\boldsymbol{t}\|^2 \\ &= \nabla f(\boldsymbol{w}_k)^\top P_k (P_k^\top \nabla^2 f(\boldsymbol{w}_k) P_k)^{-2} P_k^\top \nabla f(\boldsymbol{w}_k) \\ &\geq \frac{1}{\rho^2} \|P_k^\top \nabla f(\boldsymbol{w}_k)\| = \frac{1}{\rho^2} \|\nabla f(\boldsymbol{w}_k)\|^2. \end{aligned}$$

With this inequality and Lemma 10, the line search stopping condition implies

$$\begin{aligned} f(\boldsymbol{w}_{k+1}) - f^* &\leq f(\boldsymbol{w}_k) - f^* - \frac{\lambda \theta_k^2}{2} \|P_k \boldsymbol{t}\|^2 \\ &\leq f(\boldsymbol{w}_k) - f^* - \frac{\lambda \theta_k^2}{2\rho^2} \|\nabla f(\boldsymbol{w}_k)\|^2 \leq (1 - \frac{\lambda \theta_k^2 \sigma}{\rho^2})(f(\boldsymbol{w}_k) - f^*). \end{aligned} \tag{54}$$

Plugging in the lower bound (53) of the step-size θ_k to (54), we have the linear convergence

$$f(\boldsymbol{w}_{k+1}) - f^* \leq (1 - \frac{\lambda \beta^2 \sigma^3}{(\rho + \lambda)^2 \rho^2})(f(\boldsymbol{w}_k) - f^*).$$

This non-asymptotic linear convergence holds for any λ. Specifically, if we take $\lambda = \rho$, then we have the iteration complexity

$$O(\frac{\rho^3}{\sigma^3} \log(\frac{1}{\epsilon})),$$

Further, if Algorithm 2 always stops at step size $\theta_k = 1$, the rate (54) becomes

$$f(\boldsymbol{w}_{k+1}) - f^* \leq (1 - \frac{\lambda \sigma}{\rho^2})(f(\boldsymbol{w}_k) - f^*).$$

In this case, picking $\lambda = \rho$ recovers the iteration complexity of gradient descent methods (Nesterov, 2003)

$$O\left(\frac{\rho}{\sigma} \log\left(\frac{1}{\epsilon}\right)\right).$$

Thus, all the results are established. ■

Proof of Theorem 4

If $\nabla f(\boldsymbol{w}_0) = \mathbf{0}$, then the initial point is the optimal solution, and there is nothing to prove. Therefore, because $\nabla f(\boldsymbol{w}_0)/\|\nabla f(\boldsymbol{w}_0)\| \neq \mathbf{0}$ is included in P_0, we start our discussion with $\text{rank}(P_k) \geq 1$. The procedure of adding columns to P_k in Algorithm 1 ensures that $\text{rank}(P_k)$ is a monotonically increasing and bounded sequence with

$$1 \leq \text{rank}(P_k) \leq n, \ \forall k \in \mathbb{N}.$$

Thus, there exists a k_0 such that

$$\text{rank}(P_k) = \text{rank}(P_{k_0}), \ \forall k \geq k_0.$$

Together with $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \theta_k P_k \boldsymbol{t}_k$ in Algorithm 1, this implies

$$\boldsymbol{w}_k - \boldsymbol{w}_0 \in \texttt{span}(P_{k_0}),\ \forall k \geq 0.$$

From Theorem 3, we know that $f(\boldsymbol{w}_k)$ converges to $f(\boldsymbol{w}^*)$. This property and the strong convexity of f imply that $\boldsymbol{w}_k$ also converges to $\boldsymbol{w}^*$ globally because

$$f(\boldsymbol{w}_k) - f(\boldsymbol{w}^*) \geq \frac{\sigma}{2}\|\boldsymbol{w}_k - \boldsymbol{w}^*\|^2.$$

Therefore, we have $\boldsymbol{w}^* - \boldsymbol{w}_0 \in \texttt{span}(P_{k_0})$. This means that the minimization procedure takes update directions wholly in the space $\texttt{span}(P_{k_0})$, and each step in the algorithm is equivalent to applying the Newton method with line search in the subspace. Consider the following equivalent reformulation

$$\boldsymbol{w}_k = \boldsymbol{w}_0 + P_{k_0}\hat{\boldsymbol{t}}_k, \boldsymbol{w}^* = \boldsymbol{w}_0 + P_{k_0}\hat{\boldsymbol{t}}^*, \forall k \geq k_0.$$

Our method after the k_0-th iteration is equivalent to applying the Newton method over the variables $\hat{\boldsymbol{t}}$. That is, the following optimization problem is solved

$$\min_{\hat{\boldsymbol{t}}} \quad f(\boldsymbol{w}_0 + P_{k_0}\hat{\boldsymbol{t}}). \tag{55}$$

From Lemma 9, (55) possesses a unique optimal solution $\hat{\boldsymbol{t}}^*$. See also (47), which has the form of the Newton direction of (55). Note that for $g(\boldsymbol{t}) \equiv f(\boldsymbol{w}_0 + P_{k_0}\boldsymbol{t})$, Assumption 2 is still satisfied because with $\|P_{k_0}\| = 1$,

$$\begin{aligned}&\|\nabla^2 g(\boldsymbol{t}_1) - \nabla^2 g(\boldsymbol{t}_2)\| = \|P_{k_0}^\top(\nabla^2 f(\boldsymbol{w}_0 + P_{k_0}\boldsymbol{t}_1) - \nabla^2 f(\boldsymbol{w}_0 + P_{k_0}\boldsymbol{t}_2))P_{k_0}\| \\ &\leq \|P_{k_0}\|^2 M\|P_{k_0}(\boldsymbol{t}_1 - \boldsymbol{t}_2)\| \leq M\|\boldsymbol{t}_1 - \boldsymbol{t}_2\|.\end{aligned}$$

Further, the line search process is scale-invariant because the columns in P_{k_0} are orthogonal due to the Schmidt process. Thus, the quadratic convergence of $\hat{\boldsymbol{t}}_k$ follows from that of the Newton method for problems satisfying Assumption 2 (Nocedal and Wright, 2006, Theorem 3.5).

$$\|\hat{\boldsymbol{t}}_{k+1} - \hat{\boldsymbol{t}}^*\| = O(\|\hat{\boldsymbol{t}}_k - \hat{\boldsymbol{t}}^*\|^2).$$

Because columns in P_{k_0} are orthogonal to each other, we have that

$$\|P_{k_0}(\hat{\boldsymbol{t}}_{k+1} - \hat{\boldsymbol{t}}^*)\| = \|(\hat{\boldsymbol{t}}_{k+1} - \hat{\boldsymbol{t}}^*)\| = O(\|\hat{\boldsymbol{t}}_k - \hat{\boldsymbol{t}}^*\|^2) = O(\|P_{k_0}(\hat{\boldsymbol{t}}_k - \hat{\boldsymbol{t}}^*)\|^2).$$

Therefore, quadratic convergence for the original variables $\boldsymbol{w}_k$ holds. ■

Appendix B. Proof of Theorem 5

We first prove that, if every iteration of Algorithm 4 reaches the optimal solution of (21), then the optimal convergence rate is guaranteed. The main idea of this proof follows from the convergence rate proof of Nesterov's accelerated gradient (Nesterov, 1983) in Bubeck (2015, Theorem 3.17).

Let $\boldsymbol{w}_k$ be the iterate obtained from Algorithm 4 at the k-th iteration. Define

$$\phi_{k+1}(\boldsymbol{w}) \equiv (1-\alpha)\,\phi_k(\boldsymbol{w}) + \alpha\left(f(\boldsymbol{w}_k) + \nabla f(\boldsymbol{w}_k)^\top(\boldsymbol{w} - \boldsymbol{w}_k) + \frac{\sigma}{2}\|\boldsymbol{w} - \boldsymbol{w}_k\|^2\right), k \geq 0, \tag{56}$$

$$\text{with}\ \ \alpha \equiv \sqrt{\frac{\sigma}{\rho}} \in (0,1], \qquad \text{and} \qquad \phi_0(\boldsymbol{w}) \equiv \frac{\sigma}{2}\|\boldsymbol{w} - \boldsymbol{w}_0\|^2 + f(\boldsymbol{w}_0). \tag{57}$$

We will first show that, by induction,

$$\nabla^2\phi_k(\boldsymbol{w}) = \sigma I,\ \forall k. \tag{58}$$

The claim holds trivially at $k = 0$ from (57). Now suppose the claim (58) holds for some $k \geq 0$. For the $(k+1)$-th iteration, by (56) we have

$$\nabla^2\phi_{k+1}(\boldsymbol{w}) = (1-\alpha)\nabla^2\phi_k(\boldsymbol{w}) + \alpha\sigma I = \sigma I.$$

Thus, the induction on (58) is established.

Now we examine the recurrence relation in ϕ_k. Because $\phi_0(\boldsymbol{w})$ and the recurrence (56) are all strongly convex quadratic, we have $\phi_k(\boldsymbol{w})$ is also a strongly convex quadratic by induction. Let $\boldsymbol{v}_k$ be a constant that minimizes ϕ_k, which implies $\nabla\phi_k(\boldsymbol{v}_k) = \boldsymbol{0}$. By (58) and the Taylor expansion, we obtain

$$\begin{aligned}\phi_k(\boldsymbol{w}) &= \phi_k(\boldsymbol{v}_k) + \nabla\phi_k(\boldsymbol{v}_k)^\top(\boldsymbol{w} - \boldsymbol{v}_k) + \frac{\sigma}{2}\|\boldsymbol{w} - \boldsymbol{v}_k\|^2 \\ &= \phi_k(\boldsymbol{v}_k) + \frac{\sigma}{2}\|\boldsymbol{w} - \boldsymbol{v}_k\|^2,\ \forall \boldsymbol{w}.\end{aligned} \tag{59}$$

Further, let $\boldsymbol{v}_{k+1}$ be the minimizer of ϕ_{k+1}. Using the derivative of (56) together with (59), we have

$$\begin{aligned}&\nabla\phi_{k+1}(\boldsymbol{v}_{k+1}) \\ &= (1-\alpha)\nabla\phi_k(\boldsymbol{v}_{k+1}) + \alpha(\nabla f(\boldsymbol{w}_k) + \sigma(\boldsymbol{v}_{k+1} - \boldsymbol{w}_k)) \\ &= (1-\alpha)\sigma(\boldsymbol{v}_{k+1} - \boldsymbol{v}_k) + \alpha(\nabla f(\boldsymbol{w}_k) + \sigma(\boldsymbol{v}_{k+1} - \boldsymbol{w}_k)) = \boldsymbol{0}.\end{aligned}$$

Thus, we obtain a recurrent definition of $\boldsymbol{v}_{k+1}$:

$$\begin{aligned}\boldsymbol{v}_0 &= \boldsymbol{w}_0, \\ \boldsymbol{v}_{k+1} &= (1-\alpha)\boldsymbol{v}_k + \alpha\boldsymbol{w}_k - \frac{\alpha}{\sigma}\nabla f(\boldsymbol{w}_k).\end{aligned} \tag{60}$$

From (60), we have

$$\boldsymbol{v}_{k+1} - \boldsymbol{v}_k = -\alpha(\boldsymbol{v}_k - \boldsymbol{w}_k) - \frac{\alpha}{\sigma}\nabla f(\boldsymbol{w}_k), \tag{61}$$

$$\boldsymbol{v}_{k+1} - \boldsymbol{w}_k = (1-\alpha)(\boldsymbol{v}_k - \boldsymbol{w}_k) - \frac{\alpha}{\sigma}\nabla f(\boldsymbol{w}_k). \tag{62}$$

Next, we obtain a recurrent relation of $\phi_{k+1}(\boldsymbol{v}_{k+1})$.

$$\begin{aligned}\phi_{k+1}(\boldsymbol{v}_{k+1}) = \; & (1-\alpha)\left(\phi_k(\boldsymbol{v}_k) + \frac{\sigma}{2}\|\boldsymbol{v}_{k+1}-\boldsymbol{v}_k\|^2\right) \\ & + \alpha\left(f(\boldsymbol{w}_k) + \nabla f(\boldsymbol{w}_k)^\top(\boldsymbol{v}_{k+1}-\boldsymbol{w}_k) + \frac{\sigma}{2}\|\boldsymbol{v}_{k+1}-\boldsymbol{w}_k\|^2\right) \quad (63) \\ = \; & (1-\alpha)\phi_k(\boldsymbol{v}_k) + \alpha f(\boldsymbol{w}_k) \\ & + \alpha(1-\alpha)\left(\nabla f(\boldsymbol{w}_k)^\top(\boldsymbol{v}_k-\boldsymbol{w}_k) + \frac{\sigma}{2}\|\boldsymbol{v}_k-\boldsymbol{w}_k\|^2\right) - \frac{1}{2\rho}\|\nabla f(\boldsymbol{w}_k)\|^2, \quad (64)\end{aligned}$$

where (63) is by (56) and (59), and (64) is by (61), (62), and applying the definition $\alpha = \sqrt{\sigma/\rho}$ to the coefficient of $\|\nabla f(\boldsymbol{w}_k)\|^2$. Note that $\{\phi_k\}$ is called the estimation sequence in Nesterov (1983).

Now we show by induction that, for $\boldsymbol{w}_k$ minimizing (21) exactly,

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} \quad \phi_k(\boldsymbol{w}) = \phi_k(\boldsymbol{v}_k) \geq f(\boldsymbol{w}_k), \; \forall k \geq 0. \quad (65)$$

First, (65) holds trivially at $k=0$ from the definition (57). Assume (65) holds at the k-th iteration. At the $(k+1)$-th iteration, (64) and (65) imply

$$\phi_{k+1}(\boldsymbol{v}_{k+1}) \geq f(\boldsymbol{w}_k) + \alpha(1-\alpha)\left(\nabla f(\boldsymbol{w}_k)^\top(\boldsymbol{v}_k-\boldsymbol{w}_k) + \frac{\sigma}{2}\|\boldsymbol{v}_k-\boldsymbol{w}_k\|^2\right) - \frac{1}{2\rho}\|\nabla f(\boldsymbol{w}_k)\|^2. \quad (66)$$

By the definition of P_k and induction on (60), both $\boldsymbol{v}_k$ and $\boldsymbol{w}_k$ belong to $\boldsymbol{w}_0 + \texttt{span}(P_{k-1})$. Thus, the optimality of $\boldsymbol{w}_k$ on

$$\min_{\boldsymbol{t}} \quad f(\boldsymbol{w}_{k-1} + P_{k-1}\boldsymbol{t})$$

implies that

$$P_{k-1}^\top \nabla f(\boldsymbol{w}_k) = \boldsymbol{0}. \quad (67)$$

Thus

$$\nabla f(\boldsymbol{w}_k)^\top(\boldsymbol{v}_k - \boldsymbol{w}_k) = 0. \quad (68)$$

This result, (66), and $\alpha \in (0,1]$ from (57) then lead to

$$\phi_{k+1}(\boldsymbol{v}_{k+1}) \geq f(\boldsymbol{w}_k) - \frac{1}{2\rho}\|\nabla f(\boldsymbol{w}_k)\|^2. \quad (69)$$

Further,

$$\begin{aligned} & f(\boldsymbol{w}_k) - \frac{1}{2\rho}\|\nabla f(\boldsymbol{w}_k)\|^2 = f(\boldsymbol{w}_k) - \frac{1}{\rho}\|\nabla f(\boldsymbol{w}_k)\|^2 + \frac{\rho}{2}\frac{1}{\rho^2}\|\nabla f(\boldsymbol{w}_k)\|^2 \\ \geq \; & f(\boldsymbol{w}_k - \frac{1}{\rho}\nabla f(\boldsymbol{w}_k)) \quad (70) \\ \geq \; & f(\boldsymbol{w}_{k+1}). \quad (71)\end{aligned}$$

The inequality (70) comes from the Taylor expansion of $f(\boldsymbol{w}_k - \nabla f(\boldsymbol{w}_k)/\rho)$ and (44):

$$\begin{aligned} f(\boldsymbol{w}_k - \frac{1}{\rho}\nabla f(\boldsymbol{w}_k)) &= f(\boldsymbol{w}_k) - \frac{1}{\rho}\|\nabla f(\boldsymbol{w}_k)\|^2 + \frac{1}{2\rho^2}\nabla f(\boldsymbol{w}_k)^\top \nabla^2 f(\tilde{\boldsymbol{w}})\nabla f(\boldsymbol{w}_k) \\ &\leq f(\boldsymbol{w}_k) - \frac{1}{\rho}\|\nabla f(\boldsymbol{w}_k)\|^2 + \frac{1}{2\rho}\|\nabla f(\boldsymbol{w}_k)\|^2, \end{aligned}$$

where $\tilde{\boldsymbol{w}}$ is between $\boldsymbol{w}_k$ and $\boldsymbol{w}_k - \frac{1}{\rho}\nabla f(\boldsymbol{w}_k)$. For (71), it holds because $\nabla f(\boldsymbol{w}_k) \in \texttt{span}(P_k)$ and $\boldsymbol{w}_{k+1}$ is a minimizer of f over $\boldsymbol{w}_0 + \texttt{span}(P_k)$. Combining (69) and (71), we have (65) holds at iteration $k+1$. Therefore, (65) holds for every $k \geq 0$ by induction.

Now, we arrive at the final steps. From (65), (56), and the strong convexity of f in Assumption 1, we have

$$\begin{aligned} & f(\boldsymbol{w}_{k+1}) - f(\boldsymbol{w}^*) \\ \leq \; & \min_{\boldsymbol{w}} \quad \phi_{k+1}(\boldsymbol{w}) - f(\boldsymbol{w}^*) \\ \leq \; & \phi_{k+1}(\boldsymbol{w}^*) - f(\boldsymbol{w}^*) \quad (72) \\ = \; & (1-\alpha)\phi_k(\boldsymbol{w}^*) + \alpha(f(\boldsymbol{w}_k) + \nabla f(\boldsymbol{w}_k)^\top(\boldsymbol{w}^*-\boldsymbol{w}_k) + \frac{\sigma}{2}\|\boldsymbol{w}^*-\boldsymbol{w}_k\|^2) - f(\boldsymbol{w}^*) \\ \leq \; & (1-\alpha)\phi_k(\boldsymbol{w}^*) + \alpha f(\boldsymbol{w}^*) - f(\boldsymbol{w}^*) \\ = \; & (1-\alpha)(\phi_k(\boldsymbol{w}^*) - f(\boldsymbol{w}^*)) \quad (73) \\ \leq \; & (1-\alpha)^{k+1}(\phi_0(\boldsymbol{w}^*) - f(\boldsymbol{w}^*)). \quad (74)\end{aligned}$$

Note that (74) comes from repeating the steps between (72) and (73). By taking logarithm at both sides, we have proven that Algorithm 4 achieves the optimal rate when (21) is solved exactly.

Next we discuss the situation when (21) is approximately solved. In the above proof we can see that having a $\boldsymbol{w}_{k+1}$ that satisfies (65) is sufficient for proving the linear convergence. Because (65) comes from (66)-(69) and (71),[11] we can consider running inner iterations in Algorithm 4 until the following inner stopping condition is satisfied.

$$\begin{cases} \nabla f(\boldsymbol{w}_{k+1})^\top(\boldsymbol{v}_{k+1}-\boldsymbol{w}_{k+1}) + \frac{\sigma}{2}\|\boldsymbol{v}_{k+1}-\boldsymbol{w}_{k+1}\|^2 \geq 0, & (75a) \\ f(\boldsymbol{w}_{k+1}) \leq f(\boldsymbol{w}_k - \frac{1}{\rho}\nabla f(\boldsymbol{w}_k)). & (75b)\end{cases}$$

Denote $\boldsymbol{t}_k^*$ as the exact solution of (21) and let

$$\boldsymbol{w}_{k+1}^* \equiv \boldsymbol{w}_k + P_k\boldsymbol{t}_k^*.$$

From the discussion in (68) and (71), clearly $\boldsymbol{w}_{k+1}^*$ satisfies (75). The remaining task is to prove that (75) can be achieved in a finite number of iterations. If $\boldsymbol{w}_{k+1}^*$ strictly satisfies inequalities in (75), then by the continuity of f and ∇f, there is a neighborhood of $\boldsymbol{w}_{k+1}^*$ in which every point satisfies (75). Because inner iterations lead to the convergence to $\boldsymbol{w}_{k+1}^*$, the inner loop must terminate finitely. On the other hand, we can easily rule out situations where $\boldsymbol{w}_{k+1}^*$ satisfies one of the conditions in (75) as an equality. When (75a) becomes an equality under $\boldsymbol{w}_{k+1}^*$, (68) implies that

$$\boldsymbol{w}_{k+1}^* = \boldsymbol{v}_{k+1}.$$

When (75b) becomes an equality, the strong convexity of $f(\boldsymbol{w}_k + P_k\boldsymbol{t})$ on $\boldsymbol{t}$, which follows from Lemma 9, implies that the optimum $\boldsymbol{w}_{k+1}^*$ of $\min_{\boldsymbol{t}} f(\boldsymbol{w}_k + P_k\boldsymbol{t})$ is unique. This property, the fact that $\nabla f(\boldsymbol{w}_k)$ lies in the column space of P_k, and the equality in (75b) imply that

$$\boldsymbol{w}_{k+1}^* = \boldsymbol{w}_k - \frac{1}{\rho}\nabla f(\boldsymbol{w}_k).$$

11. Note that we ensure (75) holds for every k, so (75a) with k rather than $k+1$ is what we use for (66).

Therefore, before starting the inner loop of Algorithm 4, we can first check if $\boldsymbol{v}_{k+1}$ or $\boldsymbol{w}_k - (1/\rho)\nabla f(\boldsymbol{w}_k)$ satisfies (75). If one of them satisfies both inequalities, then we can take this vector as $\boldsymbol{w}_{k+1}$ without going into the loop. Otherwise, neither $\boldsymbol{v}_{k+1}$ nor $\boldsymbol{w}_k - (1/\rho)\nabla f(\boldsymbol{w}_k)$ is $\boldsymbol{w}^*_{k+1}$, so $\boldsymbol{w}^*_{k+1}$ must strictly satisfy inequalities in (75).

Therefore, after a finite number of inner iterations we can always find an iterate $\boldsymbol{w}_{k+1}$ satisfying (75). This condition ensures (69) and (71) hold. Therefore, (65) is established.

Appendix C. Proof of Theorem 6

At the k-th iteration, if (21) is solved exactly, the optimality condition (67) implies that any column in P_k is orthogonal to $\nabla f(\boldsymbol{w}_{k+1})$. Therefore, if $\nabla f(\boldsymbol{w}_{k+1}) \neq \boldsymbol{0}$, the new gradient will augment P_{k+1} by one column. By induction, if $\nabla f(\boldsymbol{w}_{k'}) \neq \boldsymbol{0}$, $\forall k' = 0, \ldots, k-1$, then P_k has k orthogonal columns. Because each column of P_k is in $\mathbb{R}^n$, there can be at most n orthogonal directions in P_k. Therefore, when $k \geq n$, $\texttt{span}(P_k) = R^n$ and the solution of (21) is equivalent to the solution of (1). Thus, the optimum is reached within n iterations.

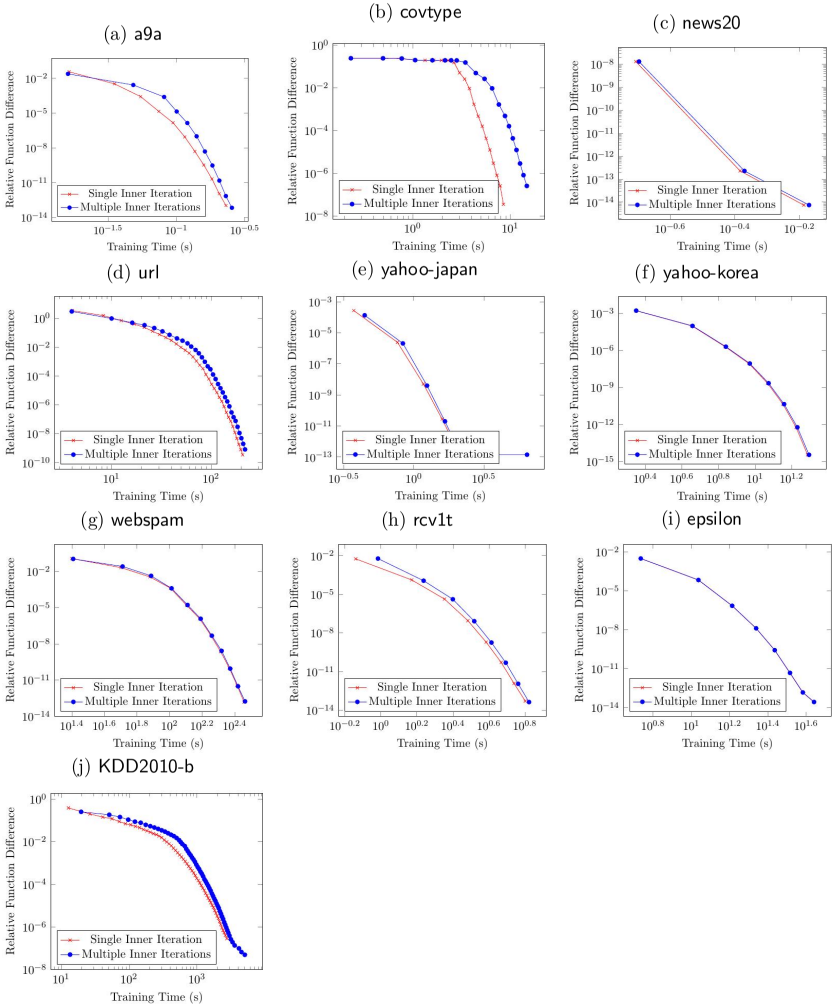


Figure 1: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present training time (in log scale) of logistic regression with $C = 10^{-3}$.

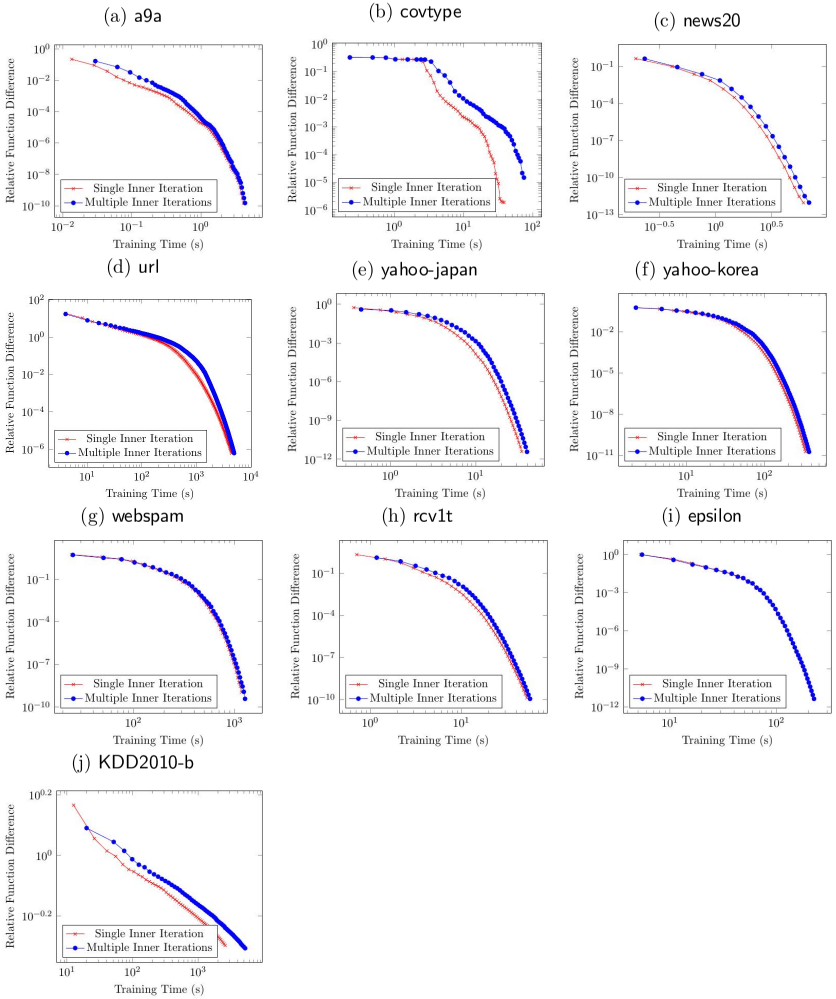


Figure 2: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present training time (in log scale) of logistic regression with $C = 10^1$.

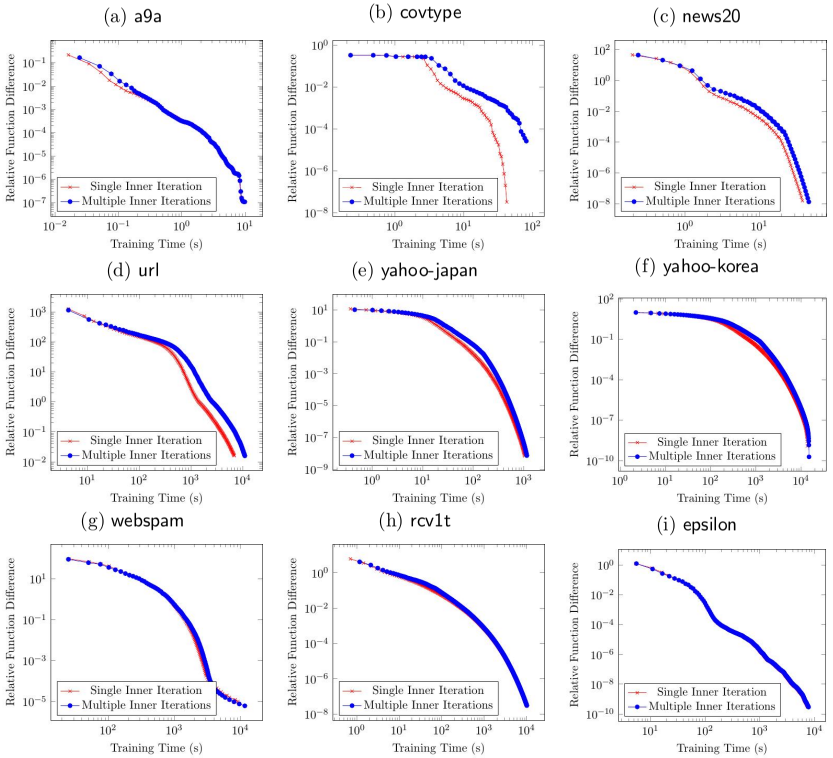


Figure 3: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present training time (in log scale) of logistic regression with $C = 10^3$.

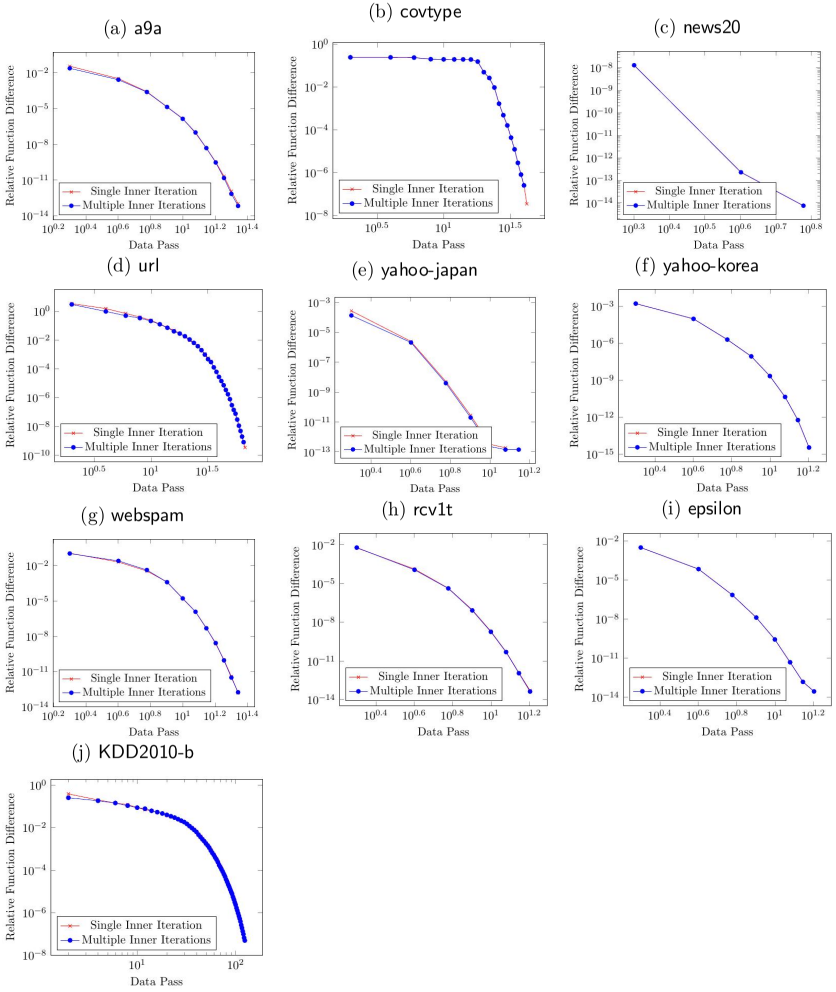


Figure 4: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present data passes (in log scale) of logistic regression with $C = 10^{-3}$.

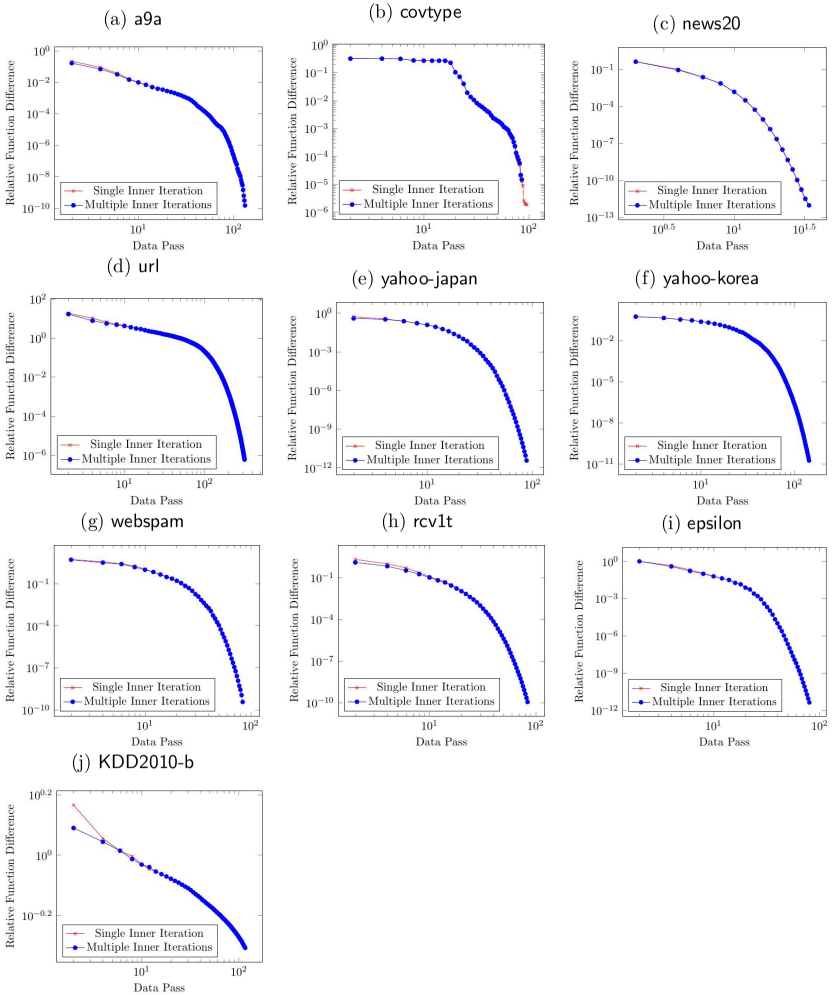


Figure 5: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present data passes (in log scale) of logistic regression with $C = 10^{1}$.

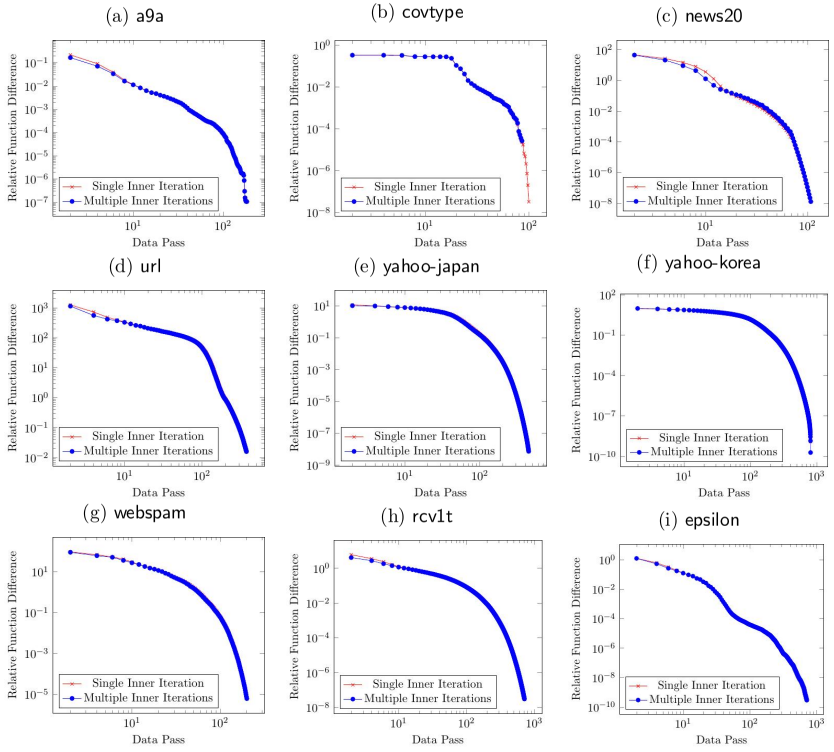


Figure 6: Comparison between single inner iteration and multiple inner iterations variants of the common-directions method. We present data passes (in log scale) of logistic regression with $C = 10^3$.

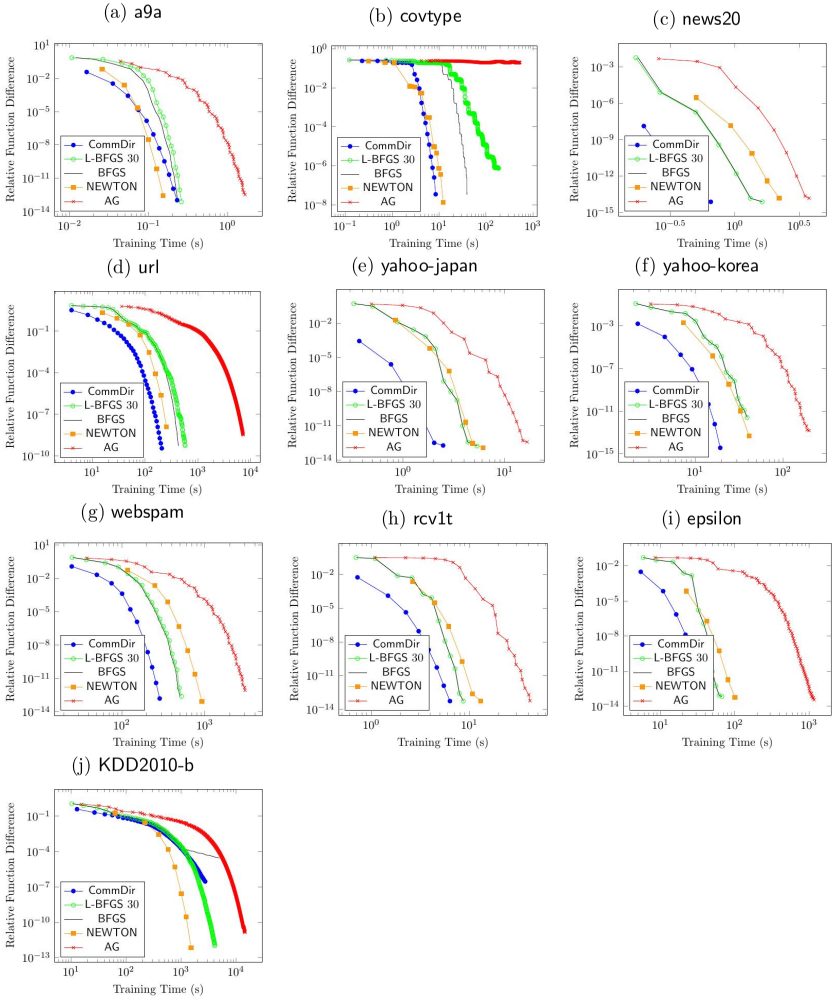


Figure 7: Training time of logistic regression with $C = 10^{-3}$.

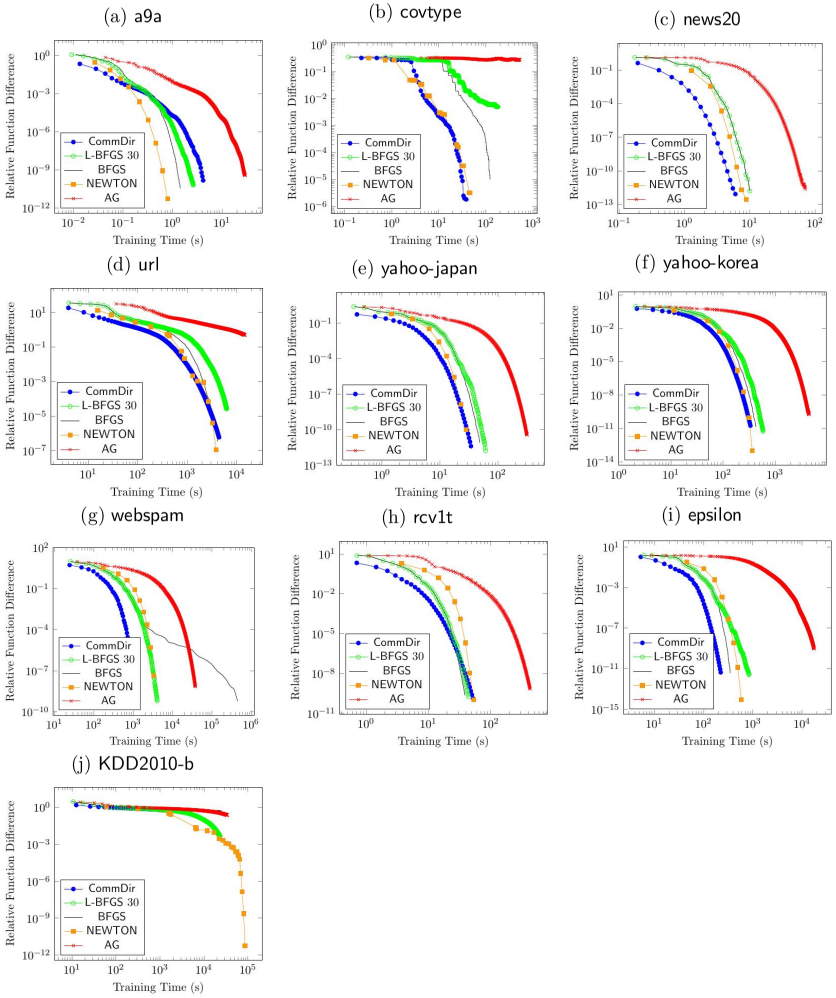


Figure 8: Training time of logistic regression with $C = 1$.

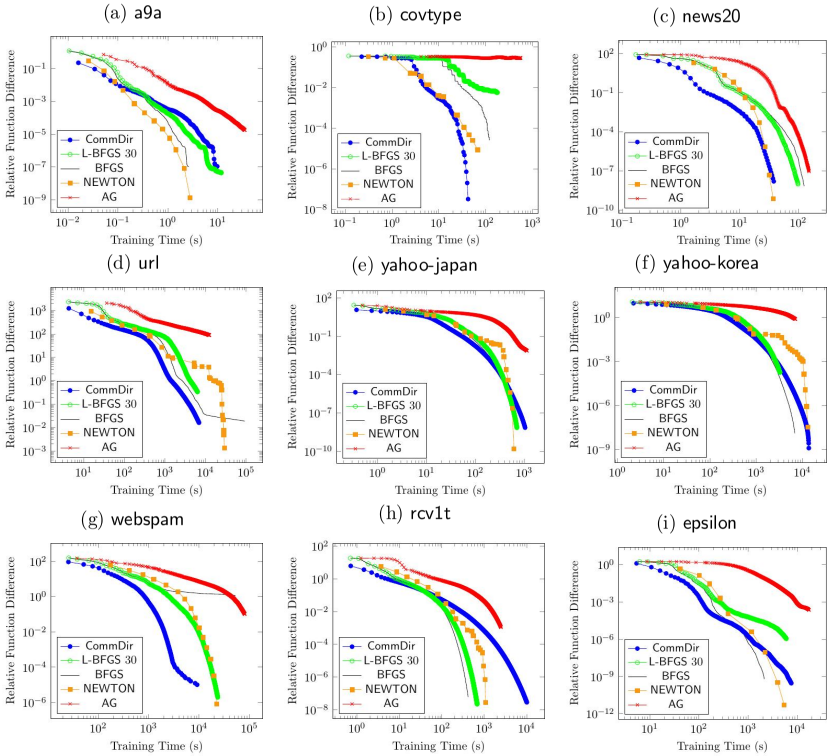


Figure 9: Training time of logistic regression with $C = 10^3$.

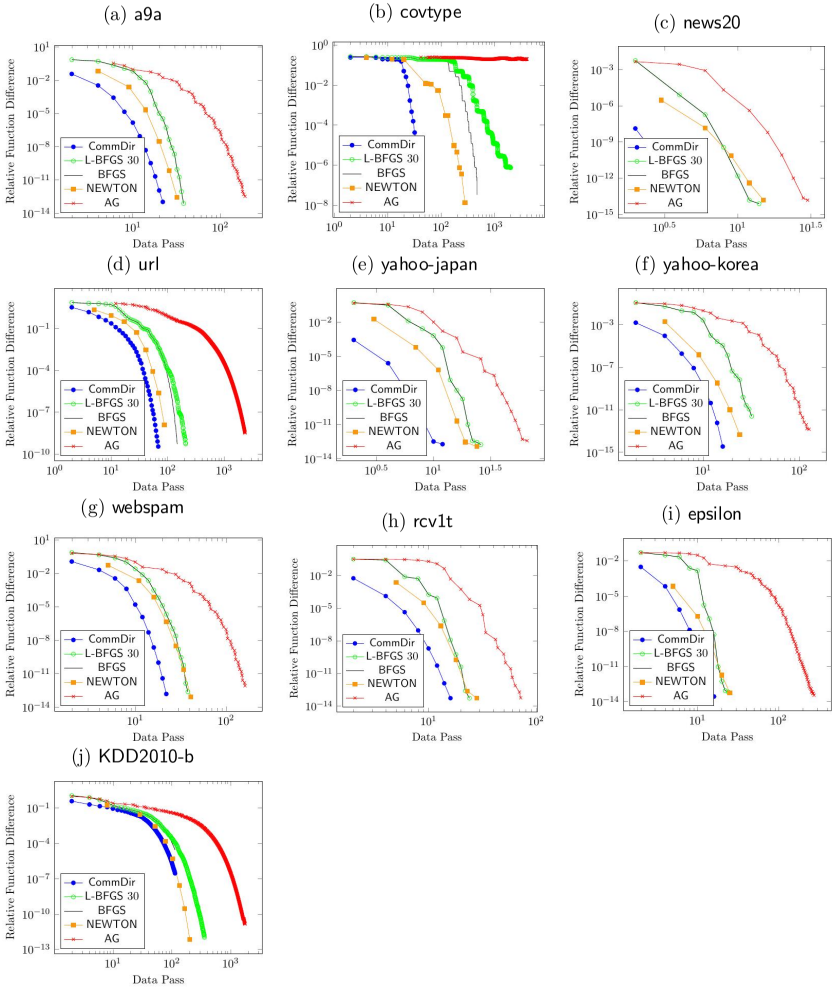


Figure 10: Number of data passes of logistic regression with $C = 10^{-3}$.

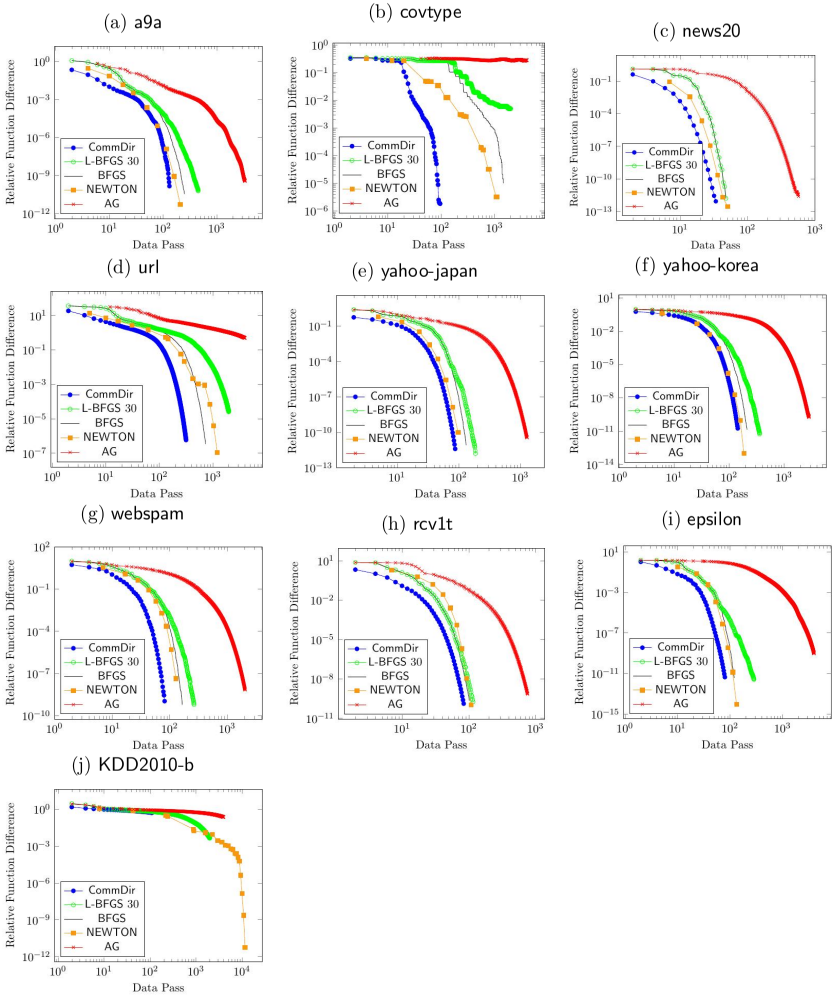


Figure 11: Number of data passes of logistic regression with $C = 1$.

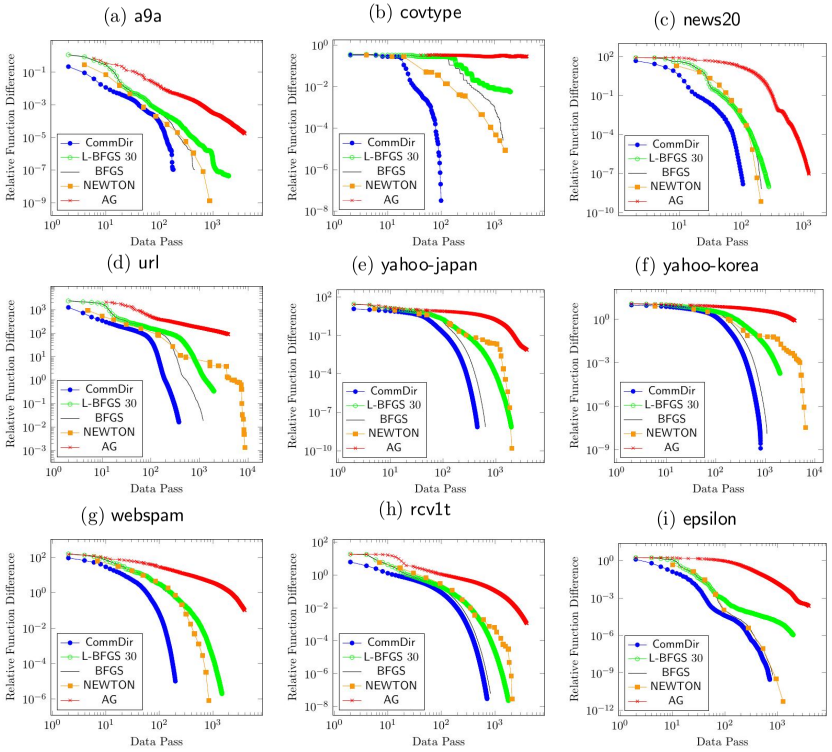


Figure 12: Number of data passes of logistic regression with $C = 10^3$.

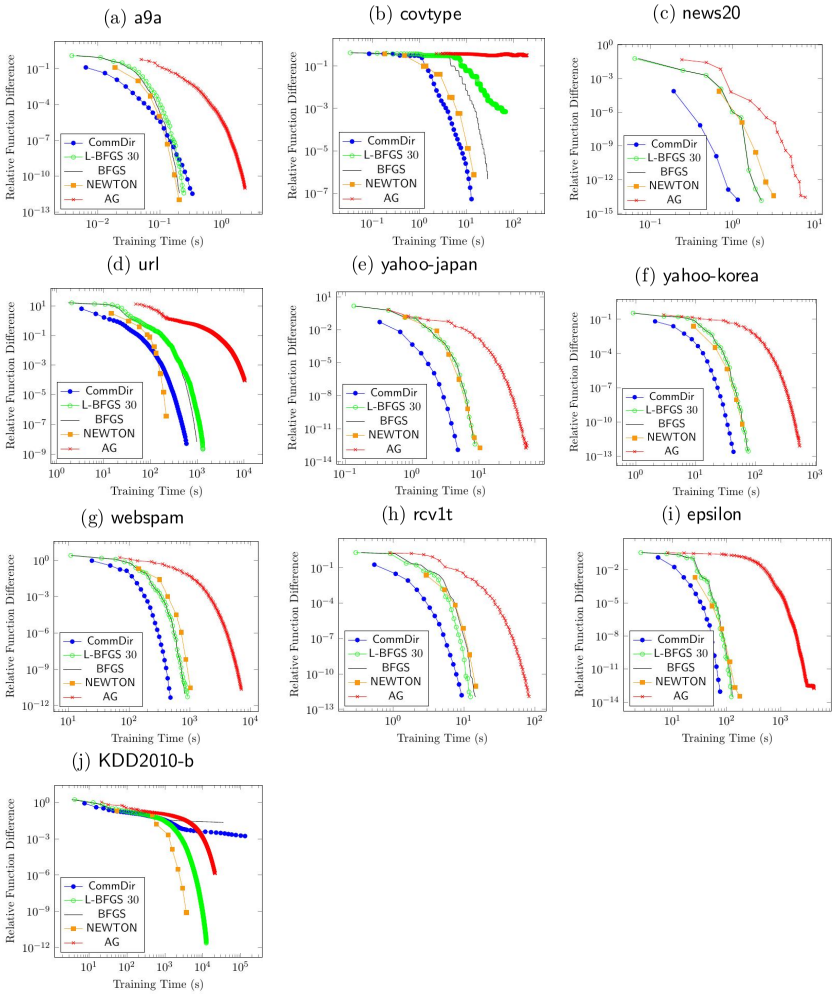


Figure 13: Training time of L2-loss SVM with $C = 10^{-3}$.

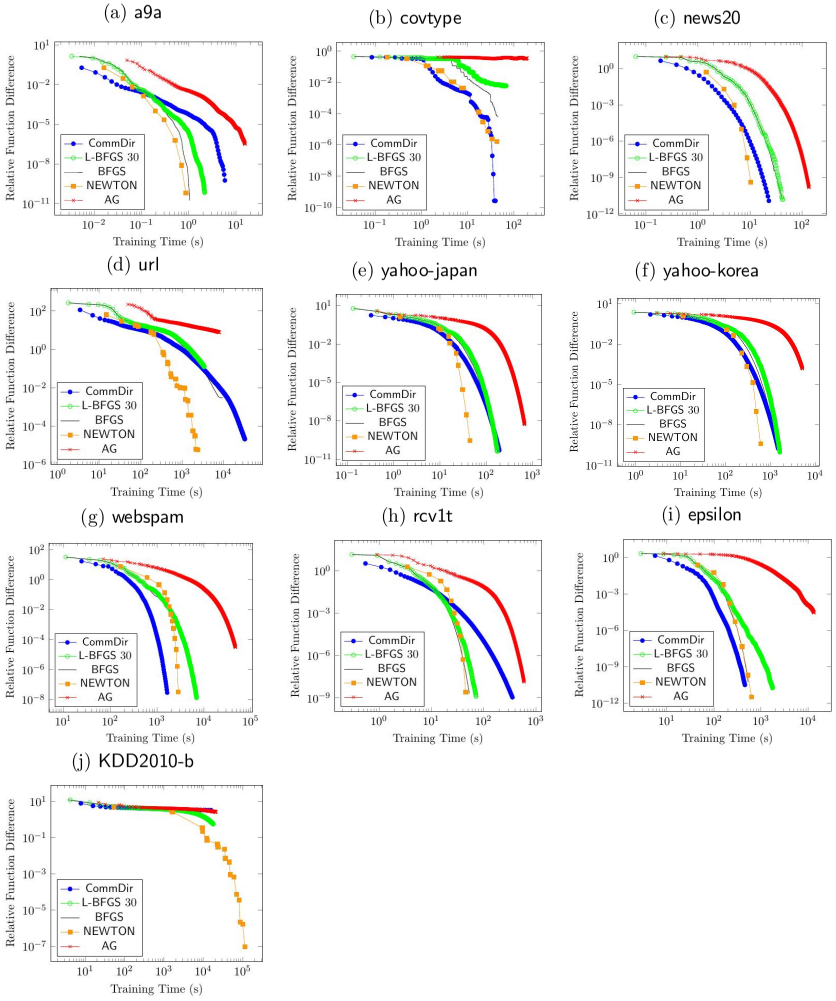


Figure 14: Training time of L2-loss SVM with $C = 1$.

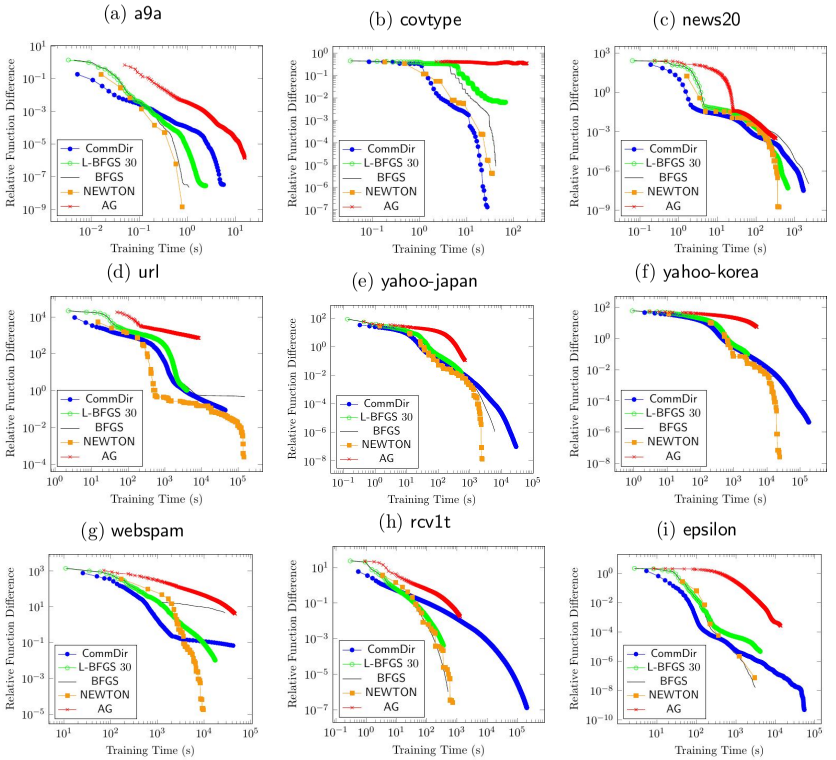


Figure 15: Training time of L2-loss SVM with $C = 10^3$.

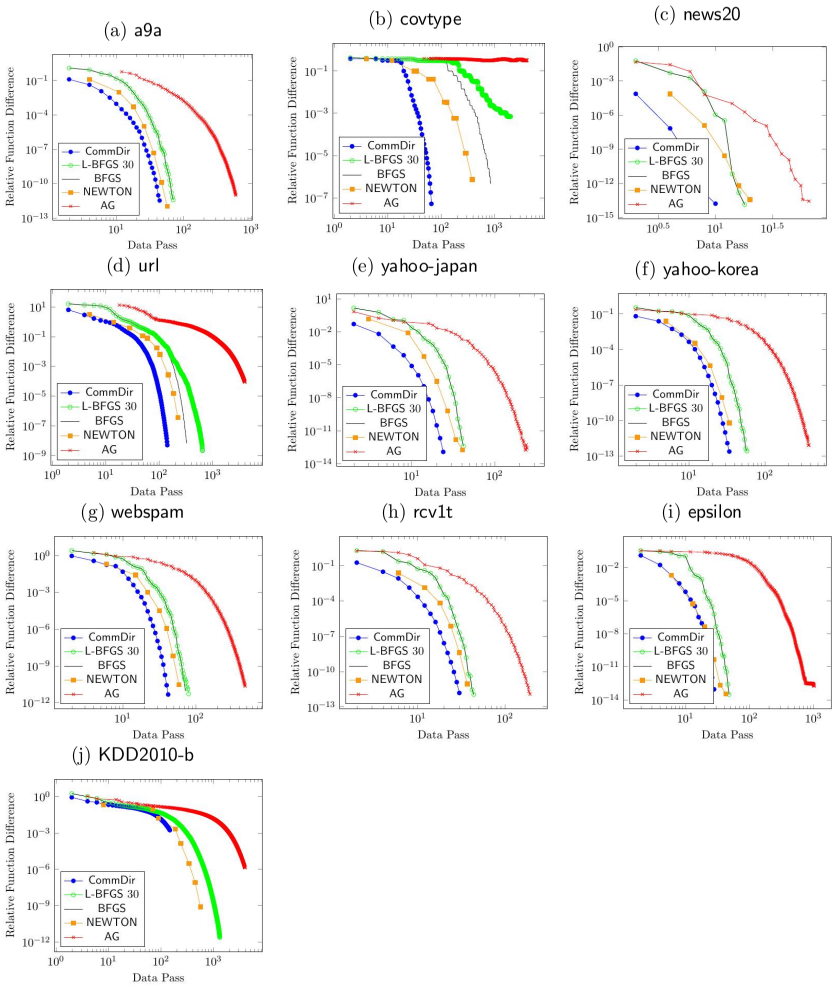


Figure 16: Number of data passes of L2-loss SVM with $C = 10^{-3}$.

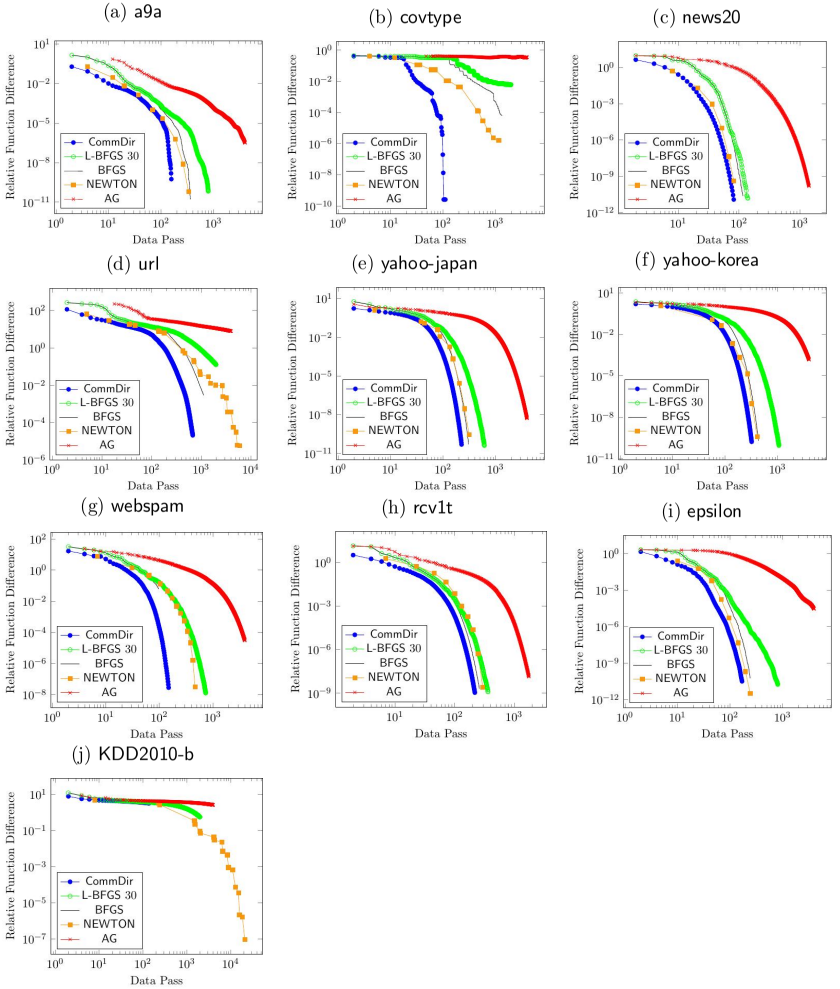


Figure 17: Number of data passes of L2-loss SVM with $C = 1$.

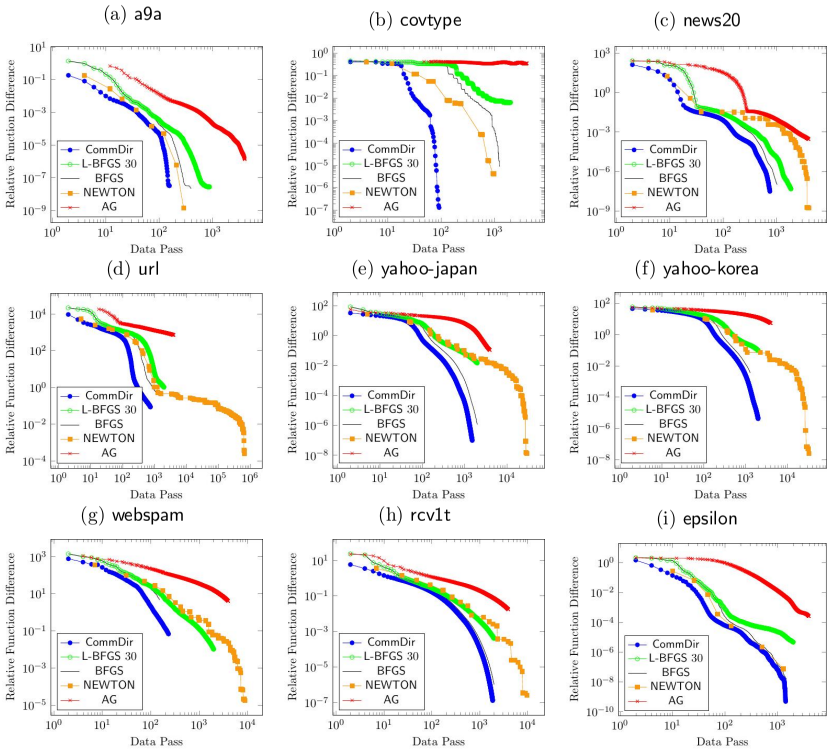


Figure 18: Number of data passes of L2-loss SVM with $C = 10^3$.

Journal of Machine Learning Research 20 (2019) 1-36 Submitted 1/17; Revised 9/17; Published 4/19

Kernel Approximation Methods for Speech Recognition

Avner May[1†] AVNERMAY@CS.STANFORD.EDU
Alireza Bagheri Garakani[2‡] BAGHERIG@USC.EDU
Zhiyun Lu[2‡] ZHIYUNLU@USC.EDU
Dong Guo[2‡] DONGGUO@USC.EDU
Kuan Liu[2,5‡] LIUKUAN@GOOGLE.COM
Aurélien Bellet[3] AURELIEN.BELLET@INRIA.FR
Linxi Fan[1] JIMFAN@CS.STANFORD.EDU
Michael Collins[4,5] MCOLLINS@CS.COLUMBIA.EDU
Daniel Hsu[4] DJHSU@CS.COLUMBIA.EDU
Brian Kingsbury[6] BEDK@US.IBM.COM
Michael Picheny[6] PICHENY@US.IBM.COM
Fei Sha[2] FEISHA@USC.EDU

[1]*Department of Computer Science, Stanford University, Stanford, CA 94305, USA*
[2]*Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA*
[3]*INRIA, 40 Avenue Halley, 59650 Villeneuve d'Ascq, France*
[4]*Department of Computer Science, Columbia University, New York, NY 10027, USA*
[5]*Google Inc, USA*
[6]*IBM Research AI, Yorktown Heights, NY 10598, USA*
† ‡: Contributed equally as the first and second co-authors, respectively

Editor: Benjamin Recht

Abstract

We study the performance of kernel methods on the acoustic modeling task for automatic speech recognition, and compare their performance to deep neural networks (DNNs). To scale the kernel methods to large data sets, we use the random Fourier feature method of Rahimi and Recht (2007). We propose two novel techniques for improving the performance of kernel acoustic models. First, we propose a simple but effective feature selection method which reduces the number of random features required to attain a fixed level of performance. Second, we present a number of metrics which correlate strongly with speech recognition performance when computed on the heldout set; we attain improved performance by using these metrics to decide when to stop training. Additionally, we show that the linear bottleneck method of Sainath et al. (2013a) improves the performance of our kernel models significantly, in addition to speeding up training and making the models more compact. Leveraging these three methods, the kernel methods attain token error rates between 0.5% better and 0.1% worse than fully-connected DNNs across four speech recognition data sets, including the TIMIT and Broadcast News benchmark tasks.

Keywords: kernel methods, deep neural networks, acoustic modeling, automatic speech recognition, feature selection, logistic regression

©2019 Avner May, Alireza Bagheri Garakani, Zhiyun Lu, Dong Guo, Kuan Liu, Aurélien Bellet, Linxi Fan, Michael Collins, Daniel Hsu, Brian Kingsbury, Michael Picheny, Fei Sha.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-026.html

1. Introduction

Large-scale non-linear classification is an important and challenging problem in machine learning. In recent years, deep learning techniques have significantly advanced state-of-the-art performance on classification problems in various domains, including automatic speech recognition (ASR) (Seide et al., 2011a; Hinton et al., 2012; Mohamed et al., 2012; Xiong et al., 2017; Saon et al., 2017), computer vision (Krizhevsky et al., 2012; Simonyan and Zisserman, 2015; He et al., 2016), and natural language processing (NLP) (Mikolov et al., 2013; Sutskever et al., 2014; Andor et al., 2016). Deep neural networks (DNNs) are able to gracefully scale to large data sets, and can successfully leverage this additional data to achieve strong empirical performance. In contrast, kernel methods, which are attractive due to their high capacity, as well as for their theoretical learning guarantees and tractability (Schölkopf and Smola, 2002), do not scale well. In particular, with data sets of size N, the $\Theta(N^2)$ size of the kernel matrix makes training prohibitively slow, while the typical $\Theta(N)$ size of the resulting models (Steinwart, 2003) makes their deployment impractical.

An important technique for scaling kernel methods to large data sets is to use approximation. Kernel approximation methods construct explicit feature maps whose dot-products approximate the kernel function, and then learn linear models with these features. Notable approximation methods include the Nyström method (Williams and Seeger, 2000), and random Fourier features (RFFs) (Rahimi and Recht, 2007). Although these methods make it possible to apply kernel methods to large-scale tasks, there have been very few published attempts applying these methods to the challenging large-scale tasks on which deep learning techniques have truly shined.[1]

The primary contribution of this paper is to demonstrate, in a large-scale setting where deep learning techniques have been known to dominate, that kernel approximation methods can effectively compete with fully-connected DNNs. More specifically:

- We benchmark the performance of kernel approximation methods (RFFs) relative to fully-connected DNNs on the acoustic modeling problem for automatic speech recognition, on four data sets with millions of training points and hundreds/thousands of classes.[2]
- We propose three methods for improving the performance of the kernel acoustic models: a feature selection method, new early stopping criteria for training, and the use of a linear bottleneck layer (Sainath et al., 2013a). We show that using these techniques, the kernel methods attain token error rates (TER)[3] between 0.5% better and 0.1% worse than fully-connected DNNs on the four data sets.

This contribution is important for both practical and theoretical reasons. From a practical perspective, it suggests that kernel methods can be competitive with deep learning

1. See related work section for discussion.
2. We use the IARPA Babel Program Cantonese (IARPA-babel101-v0.4c) and Bengali (IARPA-babel103b-v0.4b) limited language packs, a 50-hour subset of Broadcast News (BN-50) (Kingsbury, 2009), and TIMIT (Garofolo et al., 1993).
3. For our Cantonese data set, 'token error rate' corresponds to 'character error rate.' For our Bengali and Broadcast News data sets, it corresponds to 'word error rate.' For TIMIT, it corresponds to 'phone error rate.'

methods on large-scale tasks. From a theoretical perspective, it adds to our understanding of DNNs and non-linear classification. There is a large open question of why DNNs work, which is being actively investigated from various directions, including optimization (Dauphin et al., 2014; Choromanska et al., 2015; Anandkumar and Ge, 2016; Agarwal et al., 2017; Xie et al., 2017; Pennington and Bahri, 2017), representational power and efficiency (Cybenko, 1989; Hornik et al., 1989; Bengio et al., 2007; Bianchini and Scarselli, 2014; Montúfar et al., 2014; Ba and Caruana, 2014), and generalization performance (Bartlett, 1996; Neyshabur et al., 2015; Zhang et al., 2017; Arpit et al., 2017). The fact that kernel methods can match DNNs on a task this large and challenging gives an important new perspective.

As discussed above, we propose three methods to improve the performance of the kernel acoustic models. First, we propose a simple feature selection algorithm, which effectively reduces the number of random features required to attain a fixed level of performance. We iteratively select features from large pools of random features, using learned weights in the selection criterion. This has two clear benefits: (1) the subsequent training on the selected features is considerably faster than training on the entire pool of random features, and (2) the resulting model is also much smaller. For certain kernels, this feature selection approach—which is applied at the level of the random features—can be regarded as a non-linear method for feature selection at the level of the input features. We use this observation to motivate the design of a new kernel function, the "sparse Gaussian kernel," which performs well in conjunction with the feature selection algorithm.

Second, we present several frame-level metrics which correlate strongly with the TER. We show that we can attain notable gains in TER for both kernels and DNNs by monitoring these metrics on the heldout set during training to determine when to stop training.

Lastly, we demonstrate the importance of using a linear bottleneck (Sainath et al., 2013a) in the parameter matrix of our kernel models. Not only does this method improve the performance of our kernel models significantly, it also makes training faster, and reduces the size of the models learned.

In this paper we unify and extend the previous works of Lu et al. (2016) and May et al. (2016). The most significant additions are as follows: (1) we show that we can attain improved performance by combining the methods from both papers; (2) we present a more extensive set of experiments, including results on the TIMIT benchmark task, and a detailed ablation study to reveal the marginal improvements from each method; (3) we present a larger set of metrics which correlate strongly with TER, and show that we can attain improved performance by using these metrics during training to decide when to decay the learning rate and stop training.

The rest of the paper is organized as follows. We review related work in Section 2. We provide some background for kernel approximation methods, as well as for acoustic modeling, in Section 3. We present our feature selection algorithm in Section 4. In Section 5, we present several novel metrics which correlate strongly with TER, and discuss how they can be used during training to improve TER performance. In Section 6, we report extensive experiments comparing DNNs and kernel methods, including results using the methods discussed above. We conclude in Section 7.

2. Related Work

Scaling up kernel methods has been a long-standing and actively studied problem (Platt, 1998; DeCoste and Schölkopf, 2002; Tsang et al., 2005; Bottou et al., 2007; Clarkson, 2010). Approximating kernels by constructing explicit finite-dimensional feature representations, where dot products between these representations approximate the kernel function, has emerged as a powerful technique (e.g., Williams and Seeger, 2000; Rahimi and Recht, 2007). The Nyström method constructs these feature maps, for arbitrary kernels, via a low-rank decomposition of the kernel matrix (Williams and Seeger, 2000). For shift-invariant kernels, the RFF technique of Rahimi and Recht (2007) uses random projections to generate the features. Random projections can also be used to approximate a wider range of kernels (Kar and Karnick, 2012; Vedaldi and Zisserman, 2012; Hamid et al., 2014; Pennington et al., 2015). Many recent works aim to make RFFs more computationally efficient. One line of work attempts to reduce the time and memory needed to compute the RFFs by imposing structure on the random projection matrix (Le et al., 2013; Yu et al., 2015). It is also possible to use doubly-stochastic methods to speed-up stochastic gradient training of models based on the random features (Dai et al., 2014). For kernels with sparse feature expansions, Sonnenburg and Franc (2010) show how to scale kernel SVMs to data sets with up to 50 million training samples by using sparse vector operations for parameter updates.

Despite much progress in kernel approximation, there have been very few applications of these methods to challenging large-scale problems, or comparisons with DNNs on these tasks. Notable exceptions are the following: on image recognition problems, it has been shown that random Fourier features (Rahimi and Recht, 2007) can be used to replace the fully connected layers in the convolutional neural network (CNN) known as AlexNet (Krizhevsky et al., 2012), and achieve comparable performance on the ImageNet 2012 data set (Dai et al., 2014; Yang et al., 2015). Although these results suggest that kernel methods can replace the fully-connected layers of neural networks, the hybrid CNN-kernel model makes it difficult to disentangle the relative importance of these two components. In ASR, the only existing work applying kernel approximation methods has been quite limited in scope (Huang et al., 2014), using the relatively easy and small-scale TIMIT data set.[4] A detailed evaluation of kernel methods on large-scale ASR tasks, together with a thorough comparison with DNNs, has not been performed. Our work fills this gap, tackling challenging large-scale acoustic modeling problems, where deep neural networks achieve strong performance. Additionally, we provide a number of important improvements to the kernel methods, which boost their performance significantly.

One contribution of our work is to introduce a feature selection method that works well in conjunction with random Fourier features in the context of large-scale multi-class classification problems. Recent work on feature selection methods with random Fourier features, for binary classification and regression problems, includes the Sparse Random Features algorithm of Yen et al. (2014). This algorithm is a coordinate descent method for smooth convex optimization problems in the infinite space of non-linear features; each step involves solving a batch ℓ_1-regularized convex optimization problem over randomly generated non-linear features. Here, the ℓ_1-regularization may cause the learned solution

4. Here, we are excluding the results presented in this paper, some of which have already been published (May et al., 2016; Lu et al., 2016).

to only depend on a subset of the generated features. A drawback of this approach is the computational burden of fully solving many batch optimization problems, which is prohibitive for large data sets. In our attempts to implement an online variant of this method using the forward-backward splitting (FOBOS) algorithm of Duchi and Singer (2009) (using ℓ_1/ℓ_2 mixed-norm regularization for the multi-class setting), we observed that very strong regularization was required to obtain any intermediate sparsity, which in turn severely hurt prediction performance. Our approach for selecting random features is more efficient, and more directly ensures sparsity, than regularization. The basic idea behind our approach is to iteratively train a model over a batch of random features, and to then replace the features whose corresponding rows in the parameter matrix have small ℓ_2 norm. This method bears some similarity to the methods of pruning neural networks which eliminate parameters whose magnitudes are below a certain threshold (Ström, 1997; Han et al., 2015); a difference is that in our method we eliminate entire *rows* of the parameter matrix instead of individual entries.

Another improvement we propose alters the frame-level training of the acoustic model to improve the speech recognition performance of the final model. A set of methods, typically referred to as *sequence training* techniques, share our goal of tuning the acoustic model for the purpose of improving its recognition performance. There are a number of different sequence training criteria which have been proposed, including maximum mutual information (MMI) (Bahl et al., 1986; Valtchev et al., 1997), boosted MMI (BMMI) (Povey et al., 2008), minimum phone error (MPE) (Povey and Woodland, 2002), or minimum Bayes risk (MBR) (Kaiser et al., 2000; Gibson and Hain, 2006; Povey and Kingsbury, 2007). These methods, though originally proposed for training Gaussian mixture model (GMM) acoustic models, can also be used for neural network acoustic models (Kingsbury, 2009; Veselý et al., 2013). Nonetheless, all of these methods are quite computationally expensive and are typically initialized with an acoustic model trained via the frame-level cross-entropy criterion. Our method, by contrast, is very simple, only making a small change to the frame-level training process. Furthermore, it can be used in conjunction with the above-mentioned sequence training techniques by providing a better initial model. Recently, Povey et al. (2016) showed that it is possible to train an acoustic model using *only* sequence-training methods, with the lattice-free version of the MMI criterion. In a similar vein, there has been significant work on end-to-end training of ASR systems, which removes the need for frame-level training of acoustic models altogether (Graves et al., 2006; Amodei et al., 2016; Chan et al., 2016; Chiu et al., 2018). For future work, we would like to see how much kernel models can benefit from the various sequence training methods mentioned above, relative to DNNs.

Recent years have seen huge improvements in the performance of state-of-the-art speech recognition systems. The most important factors leading to this success have been the following: sequence training (Povey et al., 2008, 2016), speaker adaptation through the use of i-vectors (Dehak et al., 2011), training on large data sets (van den Berg et al., 2017; Saon et al., 2017), and improved deep architectures for both language modeling (Mikolov et al., 2010; Sundermeyer et al., 2012; Saon et al., 2017), and acoustic modeling. For acoustic modeling, CNNs (Krizhevsky et al., 2012; Sainath et al., 2013c; Soltau et al., 2014; Simonyan and Zisserman, 2015; Sercu and Goel, 2016; He et al., 2016; Saon et al., 2017) along with Long Short Term Memory (LSTM) networks (Sak et al., 2014; Saon et al., 2017), have been developed to leverage the time-frequency structure of the speech signal,

and achieve better performance than fully-connected feed-forward DNNs. The most recent state-of-the-art systems (Xiong et al., 2017; Saon et al., 2017) use an ensemble of LSTMs and CNNs for acoustic modeling. In Saon et al. (2016) they show an improvement of 1.3% in WER on the Switchboard data set when switching from a sigmoid DNN architecture to an LSTM, while in Xiong et al. (2017) they see that a ResNet CNN (He et al., 2016) improves upon a ReLU DNN by 1.6%.

In the context of these recent advances, our results showing competitive performance with fully-connected DNNs are significant, for a number of reasons. First of all, while no longer being state-of-the-art, fully-connected DNNs still attain strong performance on the acoustic modeling task. Second, fully-connected DNNs remain an important class of models, which are used widely (e.g., Andor et al., 2016). Furthermore, fully-connected layers are an important building block within more complex deep learning architectures (Simonyan and Zisserman, 2015; He et al., 2016). Additionally, we believe it should be important to the research community to discover when and why deep architectures are necessary, while simultaneously working to explore which other families of models might be able to compete; we think kernel methods are an important family of models to consider, as they lend themselves to simpler interpretation, and cleaner theoretical analysis, relative to DNNs. For future work, we would like to develop specialized kernel methods to better leverage the structure in the speech signal, in a manner similar to CNNs and LSTMs.

This work also contributes to the debate on the relative strengths of deep and shallow models. Kernel models can generally be seen as shallow models, given that they involve learning a linear model on top of a fixed transformation of the data. Furthermore, as explained in Section 3.3, many types of kernels (including popular kernels like the Gaussian kernel and the Laplacian kernel) can actually be seen as a special case of a shallow neural network. Conversely, any neural network can be understood as a kernel model, in which the kernel function itself is learned. Classic results show that both deep and shallow neural networks, as well as kernel methods, are "universal approximators," meaning that they can approximate any real-valued continuous function with bounded support to an arbitrary degree of precision (Cybenko, 1989; Hornik et al., 1989; Micchelli et al., 2006). However, a number of papers have argued that there exist functions which deep neural networks can express with exponentially fewer parameters than shallow neural networks (Montúfar et al., 2014; Bianchini and Scarselli, 2014). Other papers have argued that kernel methods may require a number of training samples which is exponential in the intrinsic dimension of the data manifold to generalize well, a problem known as the *curse of dimensionality* (Härdle et al., 2004; Bengio et al., 2007). In Ba and Caruana (2014), the authors show that the performance of shallow neural networks can be improved considerably by training them to match the outputs of deep neural networks. In showing that kernel methods can compete with DNNs on large-scale speech recognition tasks, this paper adds credence to the argument that shallow models can compete with deep networks.

3. Background

In this section, we provide background on kernels and how to approximate them with random Fourier features (Rahimi and Recht, 2007), on acoustic modeling (using neural networks and kernels), and on the linear bottleneck method of Sainath et al. (2013a).

3.1. Kernel Methods and Random Fourier Features

Kernel methods, broadly speaking, are a set of machine learning techniques which either explicitly or implicitly map data from the input space $\mathcal{X}$ to some feature space $\mathcal{H}$, in which a linear model is learned. A kernel function $K\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is then defined[5] as the function which takes as input $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$, and returns the dot-product of the corresponding points in $\mathcal{H}$. If we let $\boldsymbol{\phi}\colon \mathcal{X} \to \mathcal{H}$ denote the map into the feature space, then $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}') \rangle$. Standard kernel methods avoid inference in $\mathcal{H}$, because it is generally a very high-dimensional, or even infinite-dimensional, space. Instead, they solve the dual problem by using the N-by-N kernel matrix containing the values of the kernel function applied to all pairs of N training points. This method of working in the dual space is known as the "kernel trick," and it provides a nice computational advantage when $\dim(\mathcal{H})$ is far greater than N. However, when N is very large, the $\Theta(N^2)$ size of the kernel matrix makes training impractical.

Rahimi and Recht (2007) address this problem by leveraging Bochner's Theorem, a classical result in harmonic analysis, to provide a way to approximate any positive-definite shift-invariant kernel K with finite-dimensional features, known as random Fourier features. A kernel function K is shift-invariant if and only if $K(\boldsymbol{x}, \boldsymbol{x}') = \hat{K}(\boldsymbol{x} - \boldsymbol{x}')\ \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ for some function $\hat{K}\colon \mathbb{R}^d \to \mathbb{R}$. We now present Bochner's Theorem:

Theorem 1 *(Bochner's theorem, adapted from Rahimi and Recht (2007)): A continuous shift-invariant kernel $K(\boldsymbol{x}, \boldsymbol{x}') = \hat{K}(\boldsymbol{x} - \boldsymbol{x}')$ on $\mathbb{R}^d$ is positive-definite if and only if $\hat{K}$ is the Fourier transform of a non-negative measure $\mu(\boldsymbol{\omega})$.*

Thus, for any positive-definite shift-invariant kernel $\hat{K}(\boldsymbol{\delta})$, we have that

$$\hat{K}(\boldsymbol{\delta}) = \int_{\mathbb{R}^d} \mu(\boldsymbol{\omega}) e^{-j\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\delta}}\, d\boldsymbol{\omega}, \tag{1}$$

where

$$\mu(\boldsymbol{\omega}) = (2\pi)^{-d} \int_{\mathbb{R}^d} \hat{K}(\boldsymbol{\delta}) e^{j\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\delta}}\, d\boldsymbol{\delta} \tag{2}$$

is the inverse Fourier transform[6] of $\hat{K}(\boldsymbol{\delta})$, and where $j = \sqrt{-1}$. By Bochner's theorem, $\mu(\boldsymbol{\omega})$ is a non-negative measure. As a result, if we let $Z = \int_{\mathbb{R}^d} \mu(\boldsymbol{\omega}) d\boldsymbol{\omega}$, then $p(\boldsymbol{\omega}) = \frac{1}{Z}\mu(\boldsymbol{\omega})$ is a proper probability distribution, and we get that

$$\frac{1}{Z}\hat{K}(\boldsymbol{\delta}) = \int_{\mathbb{R}^d} p(\boldsymbol{\omega}) e^{-j\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\delta}}\, d\boldsymbol{\omega}.$$

For simplicity, we will assume that $\hat{K}$ is properly-scaled, meaning that $Z = 1$. Now, the above equation allows us to rewrite this integral as an expectation:

$$\hat{K}(\boldsymbol{\delta}) = \hat{K}(\boldsymbol{x} - \boldsymbol{x}') = \int_{\mathbb{R}^d} p(\boldsymbol{\omega}) e^{j\boldsymbol{\omega}^{\mathrm{T}}(\boldsymbol{x}-\boldsymbol{x}')}\, d\boldsymbol{\omega} = \mathbb{E}_{\boldsymbol{\omega}}\left[e^{j\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}} e^{-j\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}'}\right]. \tag{3}$$

5. It is also possible to define the kernel function prior to defining the feature map; then, for positive-definite kernel functions, Mercer's theorem guarantees that a corresponding feature map $\boldsymbol{\phi}$ exists such that $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}') \rangle$.
6. There are various ways of defining the Fourier transform and its inverse. We use the convention specified in Equations (1) and (2), which is consistent with Rahimi and Recht (2007).

Kernel name	$K(\boldsymbol{x}, \boldsymbol{y})$	$p(\boldsymbol{\omega})$	Density name
Gaussian	$e^{-\|\boldsymbol{x}-\boldsymbol{x}'\|_2^2/2\sigma^2}$	$(2\pi(1/\sigma^2))^{-d/2} e^{-\frac{\|\boldsymbol{\omega}\|_2^2}{2(1/\sigma)^2}}$	Normal$(\mathbf{0}_d, \frac{1}{\sigma^2}\mathbb{1}_d)$
Laplacian	$e^{-\lambda\|\boldsymbol{x}-\boldsymbol{x}'\|_1}$	$\prod_{i=1}^{d} \frac{1}{\lambda\pi(1+(\boldsymbol{\omega}_i/\lambda)^2)}$	Cauchy$(\mathbf{0}_d, \lambda)$

Table 1: Gaussian and Laplacian Kernels, together with their sampling distributions $p(\boldsymbol{\omega})$.

This can be further simplified as

$$\hat{K}(\boldsymbol{x} - \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{\omega},b}\left[\sqrt{2}\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x} + b) \cdot \sqrt{2}\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}' + b)\right],$$

where $\boldsymbol{\omega}$ is drawn from $p(\boldsymbol{\omega})$, and b is drawn uniformly from $[0, 2\pi]$. See Appendix B for details on why this specific functional form is correct.

This motivates a sampling-based approach for approximating the kernel function. Concretely, we draw $\{\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_D\}$ independently from the distribution $p(\boldsymbol{\omega})$, and $\{b_1, \ldots, b_D\}$ independently from the uniform distribution on $[0, 2\pi]$. We then use these parameters to approximate the kernel as follows:

$$K(\boldsymbol{x}, \boldsymbol{x}') \approx \frac{1}{D}\sum_{i=1}^{D} \sqrt{2}\cos(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x} + b_i) \cdot \sqrt{2}\cos(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}' + b_i) = z(\boldsymbol{x})^{\mathrm{T}} z(\boldsymbol{x}'),$$

where $z_i(\boldsymbol{x}) = \sqrt{\frac{2}{D}}\cos(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x} + b_i)$ is the i^{th} element of the D-dimensional random vector $z(\boldsymbol{x})$. In Table 1, we list two popular (properly-scaled) positive-definite kernels with their respective inverse Fourier transforms.

Using these random feature maps in conjunction with linear learning algorithms can yield huge gains in efficiency relative to standard kernel methods on large data sets. Learning with a representation $z(\cdot) \in \mathbb{R}^D$ is relatively efficient provided that D is far less than the number of training samples N. For example, in our experiments (Section 6), we have 2 million to 16 million training samples, while $D \approx 25{,}000$ often leads to good performance.

3.2. Neural Networks Acoustic Models

Neural network acoustic models provide a conditional probability distribution $p(y|\boldsymbol{x})$ over C possible acoustic states, conditioned on an acoustic frame $\boldsymbol{x}$ encoded in some feature representation. The acoustic states correspond to context-dependent phoneme states (Dahl et al., 2012), and in modern speech recognition systems, the number of such states is on the order of 10^3 to 10^4. The acoustic model is used within probabilistic systems for decoding speech signals into word sequences. Typically, the probability model used is a hidden Markov model (HMM), where the model's emission and transition probabilities are provided by an acoustic model together with a language model. We use Bayes' rule to compute the probability $p(\boldsymbol{x}|y)$ of emitting a certain acoustic feature vector $\boldsymbol{x}$ from state y, given the output $p(y|\boldsymbol{x})$ of the neural network:

$$p(\boldsymbol{x}|y) \quad = \quad \frac{p(y|\boldsymbol{x})p(\boldsymbol{x})}{p(y)}.$$

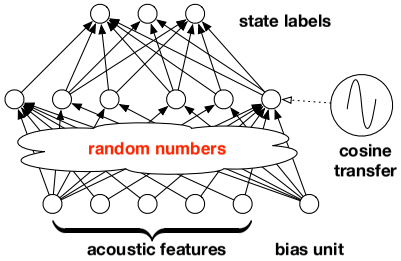


Figure 1: Kernel acoustic model seen as a shallow neural network.

Note that $p(x)$ can be ignored at inference time because it doesn't affect the relative scores assigned to different word sequences, and $p(y)$ is simply the prior probability of HMM state y. The Viterbi algorithm can then be used to determine the most likely word sequence (see Gales and Young (2007) for an overview of using HMMs for speech recognition).

3.3. Kernel Acoustic Models

To train kernel acoustic models, we use random Fourier features and simply plug the random feature vector $z(\boldsymbol{x})$ (for an acoustic frame $\boldsymbol{x}$) into a multinomial logistic regression model:

$$p(y|\boldsymbol{x}) = \frac{\exp\left(\langle \boldsymbol{\theta}_y,\ z(\boldsymbol{x})\rangle\right)}{\sum_{y'} \exp\left(\langle \boldsymbol{\theta}_{y'},\ z(\boldsymbol{x})\rangle\right)}. \tag{4}$$

The label y can take any value in $\{1, 2, \ldots, C\}$, each corresponding to a context-dependent phonetic state label, and the parameter matrix $\Theta = [\boldsymbol{\theta}_1|\ldots|\boldsymbol{\theta}_C]$ is learned. Note that we also include a bias term in our model by appending a 1 to $z(\boldsymbol{x})$ in the equation above.

The model in Equation (4) can be seen as a shallow neural network, with the following properties: (1) the parameters from the inputs (i.e., acoustic feature vectors) to the hidden units are set randomly, and are not learned; (2) the hidden units use $\cos(\cdot)$ as their activation function; (3) the parameters from the hidden units to the output units are learned (can be optimized with convex optimization); and (4) the softmax function is used to normalize the outputs of the network. See Figure 1 for a visual representation of this model architecture. Note that although using sinusoidal activation functions has been proposed previously (Goodfellow et al., 2016), their use has remained quite rare in the deep learning context.

3.4. Linear Bottleneck

When the number of random features D and the number of phonetic state labels C are large, the $D \times C$ size of the kernel acoustic model parameter matrix Θ can lead to memory and computation issues during training. We can significantly reduce the number of parameters in Θ by using a low-rank factorization $\Theta = UV$; this is called a "linear bottleneck" (Sainath et al., 2013a). This strictly decreases the capacity of the resulting model, while unfortunately rendering the optimization problem non-convex. This method can be understood

as a regularization technique, which typically improves the generalization performance of a trained model, as we will show in Section 6.

It is important to note that one can also replace a parameter matrix with a low-rank decomposition after training has completed, for example, using singular value decomposition (Xue et al., 2013). However, in the context of our work it is necessary to impose the low-rank decomposition during training, given GPU memory constraints.

4. Random Feature Selection

In this section, we first motivate and describe our proposed feature selection algorithm. We then introduce a new "sparse Gaussian kernel," which performs well in conjunction with the feature selection algorithm.

4.1. Proposed Feature Selection Algorithm

Our proposed random feature selection method, shown in Algorithm 1, is iterative. In each iteration, a model is trained on a set of features using a single pass of stochastic gradient descent (SGD) on a subset of the training data. Then, the features whose corresponding rows in the parameter matrix have the smallest ℓ_2 norms are discarded and replaced with a new set of random features.

This feature selection method has the following advantages: The overall computational cost is mild, as it requires just T passes of SGD through subsets of the data of size R (equivalent to TR/N full SGD epochs). In fact, in our experiments, we find it sufficient to use $R = O(D)$. Moreover, the method is able to explore a large number of non-linear features, while maintaining a compact model. If the number of features s_t selected in iteration t grows linearly each iteration ($s_t = Dt/T$), then the learning algorithm is exposed to $D(T+1)/2$ random features throughout the feature selection process; this s_t sequence is the selection schedule we use in all our experiments. We show in Section 6 that the acoustic models trained on the selected features generally outperform models trained on random features.

It is important to note the similarities between this method, and the FOBOS method with ℓ_1/ℓ_2-regularization (Duchi and Singer, 2009). In the FOBOS method, one solves the ℓ_1/ℓ_2-regularized problem in a stochastic fashion by alternating between taking unregularized stochastic gradient descent (SGD) steps, and then "shrinking" the rows of the parameter matrix; each time the parameters are shrunk, the rows whose ℓ_2-norms are below a threshold are set to 0. After training completes, the solution will likely have some rows which are all zero, at which point the features corresponding to those rows can be discarded. In our method, on the other hand, we take many consecutive unregularized SGD steps, and only thereafter do we choose to discard the rows whose ℓ_2-norm is below a threshold. As mentioned in the related work section, our attempts at using FOBOS for feature selection failed, because the magnitude of the regularization parameter needed to produce a sparse model was so large that it dominated the learning process; as a result, the learned models performed badly, and the selected features were essentially random.

Algorithm 1 Random feature selection

input Target number of random features D, data subset size R,
Integers $(s_1, \ldots, s_{T-1})$ such that $0 < s_1 < \cdots < s_{T-1} < D$, specifying selection schedule.

initialize set of selected indices $S := \emptyset$.
for $t = 1, 2, \ldots, T$ **do**
  for $i \in \{1, \ldots, D\} \backslash S$ **do**
    $\boldsymbol{\omega}_i \sim p(\boldsymbol{\omega})$.
    $b_i \sim \mathcal{U}(0, 2\pi)$.
  end for
  if $t \neq T$ **then**
    Initialize parameter matrix Θ.[7]
    Learn weights $\Theta \in \mathbb{R}^{D \times C}$ using a single pass of SGD over R randomly selected training examples, using the projection vectors $(\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_D)$, and the biases $(b_1, \ldots, b_D)$, to generate the random Fourier features.
    $S := \{i \mid \Theta_i$ is amongst the s_t rows of Θ with highest ℓ_2 norm$\}$.[8]
  end if
end for
return The selected projection vectors $(\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_D)$, and the selected biases $(b_1, \ldots, b_D)$.

One disadvantage of this method is that the selection criterion may misrepresent the features' actual predictive utilities. For instance, the presence of some random feature may increase or decrease the weights for other random features relative to what they would be if that feature were not present. An alternative would be to consider features in isolation, and add features one at a time (as in stagewise regression methods and boosting), but this would be significantly more computationally expensive. For example, it would require $O(D)$ passes through the data, relative to $O(T)$ passes, which would be prohibitive for large D values. We find empirically that the influence of the additional random features in the selection criterion is tolerable, and it is still possible to select useful features with this method.

4.2. A Sparse Gaussian Kernel

In Section 6 we will show that our proposed feature selection algorithm improves the performance of the Laplacian kernel models significantly more than the Gaussian kernel models. In this section, we leverage this insight in order to design a new kernel, the "sparse Gaussian kernel," which we will show also benefits significantly from the feature selection process.

Recall from Table 1 that for the Laplacian kernel, the sampling distribution used for the random Fourier features is the multivariate Cauchy density. Because the Cauchy distribution is fat-tailed, a d-dimensional Cauchy vector will typically contain some entries

7. See Section 6.3 for details on how Θ is initialized.
8. In the case where we are using a linear bottleneck to decompose the parameter matrix Θ into UV, we perform the SGD training using this decomposition. After we complete the training in a given iteration, we compute $\Theta = UV$, and then select features based on the ℓ_2 norms of the rows of Θ.

much larger than the rest. This property of the sampling distribution implies that many of the random features generated via projections with random Cauchy vectors will effectively concentrate on only a few of the input features. We can thus regard such random features as being non-linear combinations of a small number of the original input features. Thus, the proposed feature selection method effectively picks out useful non-linear interactions between small sets of input features.

We can also directly construct sparse non-linear combinations of the input features. Instead of relying on the properties of the Cauchy distribution, we can actually choose a small number k of coordinates $F \subseteq \{1, 2, \ldots, d\}$, say, uniformly at random, and then choose the random vector so that it is always zero in positions outside of F. Compared to the random Fourier feature approximation to the Laplacian kernel, the random vectors chosen in this way are truly sparse. From a systems perspective, this sparsity can reduce the memory required for the random projection matrix, and make the random features more efficient to compute (if efficient sparse matrix operations are used).

Note that random Fourier features with such sparse sampling distributions in fact correspond to shift-invariant kernels that are rather different from the Laplacian kernel. For instance, if the non-zero entries of $\boldsymbol{\omega}$ are drawn i.i.d. from $\mathcal{N}(0, \sigma^{-2})$, then the corresponding kernel is

$$K(\boldsymbol{x}, \boldsymbol{x}') = \binom{d}{k}^{-1} \sum_{\substack{F \subseteq \{1, \ldots, d\}, \\ |F| = k}} \exp\left(-\frac{\|\boldsymbol{x}_F - \boldsymbol{x}'_F\|_2^2}{2\sigma^2}\right), \tag{5}$$

where $\boldsymbol{x}_F$ is a vector composed of the elements $\boldsymbol{x}_i$ for $i \in F$. We call this kernel the "sparse Gaussian kernel." Note that this kernel puts equal emphasis on all input feature subsets F of size k. However, the feature selection process may effectively bias the distribution of the feature subsets to concentrate on some small family $\mathcal{F}$ of input feature subsets.

5. New Early Stopping Criteria

A challenge for training acoustic models is that the training criterion (e.g., cross-entropy) does not perfectly correlate with the true objective (TER). To partially address this problem, in this section we present several new metrics which we observe correlate with TER significantly better than cross-entropy does. In Section 6 we then show that we can attain improved TER performance by monitoring these metrics on the heldout set during training to decide when to decay the learning rate and stop training. Note that the reason we use these metrics as proxies for the TER, instead of directly using the TER, is that it is very expensive to compute the TER on the development set.

The common thread which unites all the metrics we will present is that they do not penalize very incorrect examples (meaning, examples for which a model assigns a probability very close to 0 to the correct label) as strongly as cross-entropy does. Notice, for instance, that the cross-entropy loss can approach infinity on a single incorrect example. Our metrics are more lenient. We present them now:

1. **Entropy Regularized Log Loss (ERLL)**:[9] This loss rewards models for being confident (low entropy), by considering a weighted sum of the cross entropy loss (CE) and the average entropy (ENT) of the model on the heldout data. Specifically, for any $\beta \in \mathbb{R}$, we define the loss as

$$CE + \beta \cdot ENT = -\frac{1}{N}\sum_{i=1}^{N}\sum_{y=1}^{C}[\mathbb{I}(y = y_i) + \beta \cdot p(y|\boldsymbol{x}_i)] \log p(y|\boldsymbol{x}_i).$$

   This metric encourages models to be more confident, even if it means having a worse cross-entropy loss as a result.

2. **Capped Log Loss**: For any value of $\lambda \geq 0$, we define the "capped log loss" as

$$-\frac{1}{N}\sum_{i=1}^{N}\log(p(y_i|\boldsymbol{x}_i) + \lambda).$$

   Effectively, this loss ensures that no single example contributes more than $-\frac{1}{N}\log(\lambda)$ to the loss. If λ is a small positive number, this loss is very similar to the normal log loss for values of $p(y_i|\boldsymbol{x}_i)$ close to 1, while affecting the loss dramatically for values close to 0 (for example, when $p(y_i|\boldsymbol{x}_i) < \lambda$).

3. **Top-k Log Loss**: For this loss, assume that the heldout examples $(\boldsymbol{x}_i, y_i)$ are sorted in descending order of their $p(y_i|\boldsymbol{x}_i)$ values. Then, for any positive integer $k \leq N$, we can define the "Top-k Log Loss" as

$$-\frac{1}{k}\sum_{i=1}^{k}\log p(y_i|\boldsymbol{x}_i).$$

   This metric judges a model based on how well it does on the k heldout examples to which it assigns highest probabilities.

Notice that for $\beta = 0$, $\lambda = 0$, and $k = N$, these metrics all simplify to the standard cross-entropy loss. In Figure 2, we show plots of the empirical correlations of these metrics with TER values, as a function of each metric's "hyperparameters," based on models we have trained. More specifically, we fully train a large number of kernel and DNN models; we then evaluate the TER performance of these models on the development set, as well as compute the heldout performance of these models in terms of the three metrics described above (for various settings of β, λ, and k). The precise set of models we used are those in Tables 4 and 5. We train models on four data sets: the Cantonese and Bengali data sets for the IARPA Babel program, a 50-hour subset of the broadcast news data set (BN-50), and the TIMIT benchmark task (see Section 6.1 for data set details). We then compute the empirical correlations between the TER values and the different metrics, and plot them as a function of each metric's hyperparameter. Note that for the top-k log loss, we plot the correlation with TER as a function of the fraction $1 - \frac{k}{N}$ of the heldout data set which is *ignored*.

9. In Lu et al. (2016) this metric was called "entropy regularized perplexity" (ERP) for $\beta = 1$.

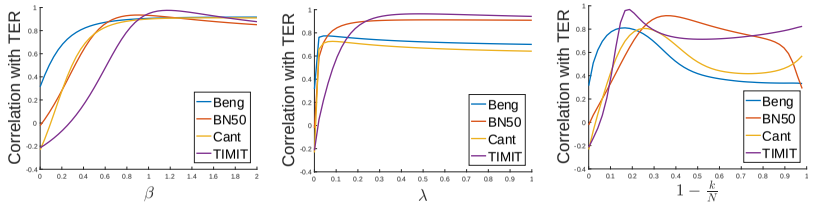


Figure 2: Empirical correlations of (left to right) heldout entropy regularized log loss, capped log loss, and top-k log loss with TER on the development set, as a function of β, λ, and $1 - \frac{k}{N}$, respectively.

As can be seen from these plots, for certain ranges of values of the metric hyperparameters, the correlation of these metrics with TER is quite high. For example, for $\beta = 1$, the correlations between entropy regularized log loss and TER are 0.91, 0.93, 0.90, and 0.95 for Bengali, BN-50, Cantonese, and TIMIT respectively. This is compared to correlations of 0.31, -0.02, -0.23, and -0.21 for the cross-entropy objective.

One conjecture for why these metrics attain higher correlation with ASR performance than heldout cross-entropy is because there is inherent noise in the labels on which the acoustic models are trained. The labels are noisy because they are generated via a forced alignment between the correct transcription and the audio using a GMM/HMM acoustic model, as discussed in Section 6.1. Thus, by not penalizing a model's mistakes on the incorrect labels as harshly, and instead focusing on the model's performance on the clean labels, these metrics better capture the quality of the model for the downstream ASR task. Better understanding this phenomenon is an interesting area for future work.

Because of the high correlation between these metrics and ASR performance, we propose to monitor these metrics on the heldout set during training to decide when to decay the learning rate and stop training (details in Section 6.3). In Section 6.4 we present results using this learning rate decay method with the ERLL metric with $\beta = 1$, and show that it leads to improvements in TER for both kernel and DNN models. We note that we could have also used the other metrics for this purpose, but chose to use ERLL with $\beta = 1$ since we observed that it attained high correlation values with TER across all four data sets.

6. Experiments

We now present our empirical results comparing the performance of fully-connected DNNs with kernel approximation methods for acoustic modeling, across four data sets. We leverage the three methods we have discussed—feature selection, the new early stopping criterion, and linear bottlenecks—and show that using these three methods the kernel models perform very similarly to the DNNs across the four test sets, attaining token error rates between 0.5% better and 0.1% worse than the DNNs.

In this section, we first provide a description of the data sets we use, and our evaluation criteria. We then give an overview of our training procedure, and provide details regarding

hyperparameter choices. We then present our experimental results comparing the performance of kernel approximation methods to DNNs. Lastly, we discuss the impact of the number of random features on performance, and take a deeper look at the dynamics of the feature selection process.

6.1. Data Sets

We train DNN and kernel acoustic models, as described in Section 3, on four data sets: the IARPA Babel Program Cantonese (IARPA-babel101-v0.4c) and Bengali (IARPA-babel103b-v0.4b) limited language packs, a 50 hour subset of the Broadcast News data set (BN-50) (Kingsbury, 2009; Sainath et al., 2011), and the TIMIT benchmark data set (Garofolo et al., 1993). Each data set is partitioned in four: a training set, a heldout set, a development set, and a test set. We use the heldout set to tune the hyperparameters of our training procedure (e.g., the learning rate). We then run decoding on the development set using IBM's Attila speech recognition toolkit (Soltau et al., 2010), to select a small subset of models which perform best in terms of TER (the best kernel and DNN model per data set). We tune the acoustic model weight on the development set to optimize the relative contributions of the language model and the acoustic model to the final score assigned to a given word sequence. Finally, we evaluate the TER performance on the test set using the selected group of models (using, for each model, the best acoustic model weight on the development set), to get a fair comparison between the methods we are using. Having a separate development set helps us avoid the risk of over-fitting to the test set.

The Bengali and Cantonese Babel data sets both include training and development sets of approximately 20 hours, and an approximately 5 hour test set. We designate about 10% of the training data as a heldout set. The training, heldout, development, and test sets all contain different speakers. Babel data is challenging because it is two-person conversations between people who know each other well (family and friends) recorded over telephone channels (in most cases with mobile telephones) from speakers in a wide variety of acoustic environments, including moving vehicles and public places. As a result, it contains many natural phenomena such as mispronunciations, disfluencies, laughter, rapid speech, background noise, and channel variability. An additional challenge in Babel is that the only data available for training language models is the acoustic transcripts, which are comparatively small.

For the Broadcast News data set, we use 45 hours of audio for training, and 5 hours as a heldout set. For the development set, we use the EARS Dev-04f data set (as described by Kingsbury, 2009), which consists of approximately three hours of broadcast news from various news shows. We use the DARPA EARS RT-03 English Broadcast News Evaluation Set (Fiscus et al., 2003) as our test set, consisting of 72 five minute conversations.

The TIMIT data set contains recordings of 630 speakers, of various English dialects, each reciting ten sentences, for a total of 5.4 hours of speech. The training set (from which the heldout set is then taken) consists of data from 462 speakers each reciting 8 sentences (SI and SX sentences). The development set consists of speech from 50 speakers. For evaluation, we use the "core test set," which consists of 192 utterances total from 24 speakers (SA sentences are excluded). For reference, we use the exact same features, labels,

Data set	Train	Heldout	Dev	Test	# features	# classes
Beng.	21 hr (7.7M)	2.8 hr (1.0M)	20 hr (7.1M)	5 hr (1.7M)	360	1000
BN-50	45 hr (16M)	5 hr (1.8M)	2 hr (0.7M)	2.5 hr (0.9M)	360	5000
Cant.	21 hr (7.5M)	2.5 hr (0.9M)	20 hr (7.2M)	5 hr (1.8M)	360	1000
TIMIT	3.2 hr (2.3M)	0.3 hr (0.2M)	0.15 hr (0.1M)	0.15 hr (0.1M)	440	147

Table 2: Data set details. We report the size of each data set partition in terms of the number of hours of speech, and in terms of the number of acoustic frames (in parentheses).

and divisions of the data set as Huang et al. (2014) and Chen et al. (2016), which allows direct comparison of our results with theirs (Table 8).

The acoustic features, representing 25 ms acoustic frames with context, are real-valued dense vectors. A 10 ms shift is used between adjacent frames (except on TIMIT, where a 5 ms shift is used). For the Cantonese, Bengali, and Broadcast News data sets we use a standard 360-dimensional speaker-adapted representation used by IBM (Kingsbury et al., 2013); these feature vectors correspond to the concatenation of nine 40-dimensional vectors corresponding to the four frames before and after the current frame. For the TIMIT experiments, we use 40-dimensional feature space maximum likelihood linear regression (fMLLR) features (Gales, 1998), and concatenate the five neighboring frames in either direction, for a total of eleven frames and 440 features.

The state labels are obtained via forced alignment using a GMM/HMM system. This forced alignment is necessary because there is a mismatch between the ground truth available for training (a word sequence transcribing each utterance) and the actual labels needed for training (precise assignment of HMM states to frames). The Cantonese and Bengali data sets each have 1000 labels, corresponding to quinphone context-dependent HMM states clustered using decision trees. For Broadcast News, there are 5000 such states. The TIMIT data set has 147 context-independent labels, corresponding to the beginning, middle, and end of 49 phonemes.

The language models we use are all n-gram language models estimated using modified Kneser-Ney smoothing, with n values of 2, 4, 3, and 3 for Bengali, Broadcast News, Cantonese, and TIMIT, respectively. The TIMIT language model is a phone-level model. The Bengali and Cantonese language models are particularly small (approximately 60,000 bigrams and 136,000 trigrams, respectively), trained using only the provided audio transcripts. The Broadcast News model is small as well, containing only 3.3 million 4-grams.

In Table 2 we provide details on all the data sets, as well as on their number of features and classes. All our data sets have millions of training points, significantly exceeding the size of typical machine learning tasks tackled by kernel methods. Additionally, the large number of output classes for our data sets presents a scalability challenge, given that the size of the kernel models scales linearly with the number of output classes (if no bottleneck is used).

6.2. Evaluation Metrics

We use five metrics to evaluate the DNN and kernel acoustic models:

1. **Cross-entropy (CE)**: Given examples, $\{(\boldsymbol{x}_i, y_i), i = 1 \dots N\}$, the cross-entropy is defined as
$$-\frac{1}{N}\sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i). \quad (6)$$

2. **Average Entropy (ENT)**: The average entropy of a model is defined as
$$-\frac{1}{N}\sum_{i=1}^{N}\sum_{y=1}^{C} p(y|\boldsymbol{x}_i) \log p(y|\boldsymbol{x}_i).$$
If a model has low average entropy, it is generally confident in its predictions.

3. **Entropy Regularized Log Loss (ERLL)**: Defined in Section 5. We use $\beta = 1$ unless specified otherwise.

4. **Classification Error (ERR)**: The classification error is defined as
$$\frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left[y_i \neq \arg\max_{y \in 1,2,\dots,C} p(y|\boldsymbol{x}_i)\right].$$

5. **Token Error Rate (TER)**: We feed the predictions of the acoustic models, which correspond to probability distributions over the phonetic states, to the rest of the ASR pipeline and calculate the misalignment between the decoder's outputs and the ground-truth transcriptions. For Bengali and BN-50, we measure the error in terms of the word error rate (WER), for Cantonese we use the character error rate (CER), and for TIMIT we use the phone error rate (PER). We use the term "token error rate" (TER) to refer, for each data set, to its corresponding metric.

6.3. Details of Acoustic Model Training

We train all our kernel models with either the Laplacian, the Gaussian, or the sparse Gaussian (Section 4.2) kernels. These kernel models typically have three hyperparameters: the kernel bandwidth (σ for the Gaussian kernels, λ for the Laplacian kernel; see Table 1), the number of random projections, and the initial learning rate. We try various numbers of random features, ranging from 5000 to 400,000. Using more random features leads to a better approximation of the kernel function, as well as to more powerful models, though there are diminishing returns as the number of features increases. The sparse Gaussian kernel additionally has the hyperparameter k which specifies the sparsity of each random projection vector $\boldsymbol{\omega}_i$. For all experiments, we use $k = 5$.

For all DNNs, we tune hyperparameters related to both the architecture and the optimization. This includes the number of layers, the number of hidden units in each layer, and the learning rate. We perform 1 epoch of layer-wise discriminative pre-training (Seide et al., 2011b; Kingsbury et al., 2013), and then train the entire network jointly using SGD. We find that four hidden layers is generally the best setting for our DNNs, so all the DNN results we present in this paper use this setting; in Table 3 we show how depth affects recognition performance on the Broadcast News data set. Additionally, all our DNNs use

	1000	2000	4000
3	17.5	16.8	16.7
4	17.1	**16.4**	16.5
5	16.9	16.5	16.7
6	17.0	16.5	16.6

Table 3: Effect of depth and width on DNN TER (development set): This table shows TER results for DNNs with 1000, 2000, or 4000 hidden units per layer, and 3-6 layers, on the Broadcast News development set. All of these models were trained using a linear bottleneck for the output parameter matrix, and using ERLL for learning rate decay. The best result is in bold.

the tanh activation function. We vary the number of hidden units per layer (1000, 2000, or 4000). We use this same set of DNN architectures for all our data sets.

For both DNN and kernel models, we use stochastic gradient descent (SGD) as our optimization algorithm for minimizing the training cross-entropy objective. We use mini-batches of size of 250 or 256 samples during training, and we use the heldout set to tune the other hyperparameters. We use the learning rate decay scheme described in (Morgan and Bourlard, 1990; Sainath et al., 2013a,b), which monitors performance on the heldout set to decide when to decay the learning rate. This method divides the learning rate in half at the end of an SGD epoch if the heldout cross-entropy doesn't improve by at least 1%; additionally, if the heldout cross-entropy gets *worse*, it reverts the model back to its state at the beginning of the epoch. Instead of using the heldout cross-entropy, in some of our experiments we use the heldout ERLL to decide when to decay the learning rate. We terminate training once we have divided the learning rate 10 times.

As mentioned in Section 3.4, one effective way of reducing the number of parameters in our models is use a linear bottleneck, which is a low-rank factorization of the output parameter matrix (Sainath et al., 2013a). We use bottlenecks of size 1000, 250, 250, and 100 for BN-50, Bengali, Cantonese, and TIMIT, respectively. We train models both with and without this technique; the only exception is that we are unable to train BN-50 kernel models without the bottleneck of size 1000, due to memory constraints on our GPUs.

We initialize our DNN parameters uniformly at random within $[-\frac{\sqrt{6}}{\sqrt{d_{in}+d_{out}}}, \frac{\sqrt{6}}{\sqrt{d_{in}+d_{out}}}]$, as suggested by Glorot and Bengio (2010); here, d_{in} and d_{out} refer to the dimensionality of the input and output of a DNN layer, respectively. For our kernel models, we initialize the random projection matrix as discussed in Section 3, and we initialize the parameter matrix Θ as the zero matrix. When using a linear bottleneck to decompose the parameter matrix, we initialize the resulting two matrices randomly, like we do for our DNNs.

For each iteration of random feature selection, we draw a random subsample of the training data of size $R = 10^6$ (except when $D \geq 10^5$, in which case we use $R = 2\times10^6$, to ensure a safe R to D ratio), but ultimately we use all N training examples once the random features are selected. Thus, each iteration of feature selection has equivalent computational cost to a R/N fraction of an SGD epoch (roughly 1/7 or 2/7 for $D < 10^5$ and $D \geq 10^5$ respectively, on the Babel speech data sets). We use $T = 50$ iterations of feature selection,

and in iteration t, we select $s_t = t \cdot (D/T) = 0.02Dt$ random features. Thus, the total computational cost we incur for feature selection is equivalent to approximately seven (or fourteen) epochs of training on the Babel data sets. For the Broadcast News data set, it corresponds to the cost of approximately six full epochs of training (for $D \geq 10^5$).

All our training code is written in MATLAB, leveraging its GPU features, and executed on Amazon EC2 machines.

6.4. Results

In this section, we report the results from our experiments comparing kernel methods to DNNs on ASR tasks. We first discuss the ablation study showing the marginal improvements to the kernel model performance attained by feature selection, the new early stopping criterion, and the linear bottlenecks; we show analogous results for DNNs for the early stopping and linear bottleneck methods. We then compare for each data set the performance of the best kernel and DNN models from these detailed experiments; the most important result from these comparisons, shown in Table 7, is that the kernel models are competitive with the DNNs, attaining TER values between 0.5% better and 0.1% worse than the DNNs across our four test sets.

For our detailed experiments, for both DNN and kernel methods we train models with and without linear bottlenecks, and with and without using ERLL to determine the learning rate decay. For our kernel methods, we additionally train models with and without using feature selection. We run experiments with all three kernels (Laplacian, Gaussian, sparse Gaussian) and we use 100,000 random features on all data sets except for TIMIT, where we are able to use 200,000 random features (because the output dimensionality is lower). As mentioned in the previous section, for our DNN experiments, we train models with four hidden layers,[10] using the tanh activation function, and using either 1000, 2000, or 4000 hidden units per layer. We focus on comparing the performance of these methods in terms of TER, but we also report results for other metrics. Unless specified otherwise, all TER results are on the development set, and all cross-entropy, entropy, classification error, and ERLL results are on the heldout set.

In Tables 4 and 5, we show the TER results for our DNN and kernel models, respectively, across all data sets. There are many things to notice about these results. For our DNN models, linear bottlenecks almost always lower TER values, though in a few cases they have no effect on TER. Using ERLL to determine when to decay the learning rate generally helps lower TER values for our DNNs, but in a few cases it actually hurts (Cantonese with 4000 hidden units, and TIMIT with 2000 and 4000). The DNNs with 4000 hidden units typically attain the best results, though on a couple of data sets they are matched or narrowly beaten by the 2000 models.

For our kernel models, we see that incorporating a linear bottleneck brings large drops in TER across the board.[11] Performing feature selection generally improves TER as well; we see that it improves TER considerably for the Laplacian kernel, and modestly for the sparse Gaussian kernel. For the Gaussian kernel, it typically helps, though there are several

10. As mentioned in Section 6.3, we find that this is generally the best setting.
11. Recall that we are unable to train BN-50 kernel models *without* using a bottleneck because the resulting models would not fit on our GPUs.

instances in which feature selection hurts TER (see Section 6.6 for discussion). Second, we see that using heldout ERLL to determine when to decay the learning rate helps all our kernel models attain lower TER values. Next, we see that without using feature selection, the sparse Gaussian kernel has the best performance across the board. After we include feature selection, it performs very comparably to the Laplacian kernel with feature selection. It is interesting to note that without using feature selection, the Gaussian kernel is generally better than the Laplacian kernel; however, with feature selection, the Laplacian kernel surpasses the Gaussian kernel (see Section 6.6). In general, the kernel function which performed best, across the majority of settings, was the sparse Gaussian kernel.

To further improve the performance of the kernel models, we train models with up to 400k features on Broadcast News, our most challenging data set. Due to the large computational expense of training these models, we only trained a few, and only used the Laplacian and sparse Gaussian kernels, as these attained the best performance after feature selection, in terms of TER. We report results in Table 6. All of these models were trained with a linear bottleneck of size 1000, and using ERLL for learning rate decay. We include results using 100k random features in this table for comparison. As we can observe, our best kernel model on BN-50 now attains a TER of 16.4%, which is equal to the performance of our best DNN model. Furthermore, we continue to see improvements in the performance of our kernel models, even as we increase the number of features beyond 100k; for the Laplacian kernel we get a gain of 0.3% in TER when increasing from 100k to 400k (both with and without feature selection), while for the sparse Gaussian kernel we get gains of 0.2% and 0.4%, with and without feature selection (respectively). To date, these are the largest models we have trained, though it seems likely we could continue getting performance improvements with even larger models.

In Table 7, we compare for each data set the performance of the best DNN model with the best kernel model, across six metrics. Importantly, for each metric (except for test set TER), we find the kernel and DNN model which performs best for that specific metric; this is in contrast to picking the kernel and DNN models which are best in terms of TER, and reporting all metrics on these models. For Broadcast News, we consider the kernel models from Table 6 with 200k and 400k random features, along with those in Table 5, in the process of finding the best performing models. In terms of heldout cross-entropy, the kernels outperformed the DNNs on Cantonese and TIMIT, while the DNNs beat the kernels on Bengali and BN-50. With regard to classification error, the kernels beat the DNNs on all data sets except for Bengali. In terms of the average heldout entropy of the models, the DNNs were consistently more confident in their predictions (lower entropy) than the kernels. Significantly, we observe that the best development set TER results for our DNN and kernel models are quite comparable; on Cantonese and TIMIT, the kernel models outperform the DNNs by 0.4% absolute, on Broadcast News the kernels exactly match the DNNs, while on Bengali the DNNs do better by 0.1%.

We now discuss the results on the test sets. First of all, to avoid overfitting to the test sets, for each data set we only performed test set evaluations for the DNN and kernel models which performed best in terms of the development set TER. The final row of Table 7 thus contains all the test results we collected.[12] As one can see, the relative test performance

12. The only exception is on the Broadcast News data set. Prior to training the best performing 400k model, we had already evaluated the best 100k model, which attained a TER of 11.9% on the test set.

	1000				2000				4000			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	72.3	71.6	71.7	70.9	71.5	71.1	70.7	70.3	71.1	70.6	70.5	**70.2**
BN-50	18.0	17.3	17.8	17.1	17.4	16.7	17.1	**16.4**	16.8	16.7	16.7	16.5
Cant.	68.4	68.1	67.9	67.5	67.7	67.7	67.2	**67.1**	67.7	**67.1**	67.2	67.2
TIMIT	19.5	19.3	19.4	19.2	19.0	18.9	19.2	19.2	**18.6**	**18.6**	18.7	18.9

Table 4: DNN TER Results (development set): This table shows TER results for 4-hidden layer DNNs with 1000, 2000, or 4000 hidden units per layer. 'NT' specifies that no "tricks" were used during training (no bottleneck, did not use ERLL for learning rate decay). A 'B' specifies that a linear bottleneck was used; an 'R' specifies that ERLL was used for learning rate decay (so 'BR' means both were used). The best result for each data set is in bold.

	Laplacian				Gaussian				Sparse Gaussian			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	74.5	72.1	74.5	71.4	72.6	72.0	72.6	71.8	73.0	71.5	73.0	**70.9**
+FS	72.9	71.1	72.8	70.4	74.1	71.4	74.2	**70.3**	72.9	71.2	72.8	70.7
BN-50	N/A	17.9	N/A	17.7	N/A	17.3	N/A	17.1	N/A	17.3	N/A	**17.0**
+FS	N/A	17.1	N/A	**16.7**	N/A	17.5	N/A	17.0	N/A	17.1	N/A	**16.7**
Cant.	69.9	68.2	69.2	67.4	70.2	67.6	70.0	**67.1**	68.6	67.5	68.1	**67.1**
+FS	68.4	67.5	68.5	**66.7**	69.9	67.7	69.8	66.9	68.6	67.4	68.5	66.8
TIMIT	20.6	19.2	20.4	18.9	19.8	18.9	19.6	18.6	19.9	18.8	19.6	**18.4**
+FS	19.5	18.6	19.3	18.4	19.5	18.6	19.4	18.4	19.3	18.4	19.1	**18.2**

Table 5: Kernel TER Results (development set): This table shows TER results for our kernel experiments using either the Laplacian, Gaussian, or Sparse Gaussian kernels, with 100k random features (except for TIMIT, which uses 200k features). 'NT' specifies that no "tricks" were used during training (no bottleneck, did not use ERLL for learning rate decay). A 'B' specifies that a linear bottleneck was used; an 'R' specifies that ERLL was used for the learning rate decay (so 'BR' means both were used). '+FS' specifies that feature selection was used. The best result for each row is in bold.

	100k	200k	400k
Laplacian	17.7	17.7	17.4
+FS	16.7	**16.4**	**16.4**
Sparse Gaussian	17.0	16.8	16.6
+FS	16.7	16.6	**16.5**

Table 6: Kernel TER Results on Broadcast News development set for models with a very large number of random feature (up to 400k). All models use a bottleneck of size 1000, and use ERLL for learning rate decay.

	Beng. (D/K)	BN-50 (D/K)	Cant. (D/K)	TIMIT (D/K)
CE	**1.243** / 1.256	**2.001** / 2.004	1.916 / **1.883**	1.056 / **0.9182**
ENT	**0.9079** / 1.082	**1.274** / 1.434	**1.375** / 1.516	**0.447** / 0.5756
ERLL	**2.302** / 2.406	**3.548** / 3.552	**3.459** / 3.493	1.671 / **1.607**
ERR	**0.2887** / 0.2936	0.4887 / **0.4881**	0.4353 / **0.4287**	0.324 / **0.3085**
TER (dev)	**70.2** / 70.3	**16.4** / **16.4**	67.1 / **66.7**	18.6 / **18.2**
TER (test)	**69.1** / 69.2	11.7 / **11.6**	63.7 / **63.2**	20.5 / **20.4**

Table 7: Table comparing the Best DNN ('D') and kernel ('K') results, across four data sets and six metrics. The first four metrics are on the heldout set, the fifth is on the development set, and the last metric is reported on the test set. For BN50, the large models from Table 6 are included in the set of models from which the best performing model is picked (for each metric). See Section 6.2 for metric definitions.

	Test TER (DNN)	Test TER (Kernel)
Huang et al. (2014)	20.5	21.3
Chen et al. (2016)	N/A	20.9
This work	20.5	**20.4**

Table 8: Table comparing the Best DNN and kernel results from this work to those from Huang et al. (2014) and Chen et al. (2016), on the TIMIT test set.

of the DNN and kernel models is quite similar to the development set results; the DNNs perform better by 0.5% on Cantonese, 0.1% on TIMIT and Broadcast News, and perform worse by 0.1% on Bengali. For direct comparison on the TIMIT data set, we include in Table 8 the test results for the best DNN and kernel models from Huang et al. (2014) and Chen et al. (2016). As mentioned in Section 6.1, we use the same features, labels, data set partitions (train/heldout/dev/test), and decoding script as these papers, and thus our results are directly comparable. We achieve a 0.9% absolute improvement in TER with our kernel model relative to Huang et al. (2014), and 0.5% improvement relative to Chen et al. (2016); furthermore, our best DNN matches the performance of the best performing DNN from Huang et al. (2014). We believe the most significant of all these results is that the kernels (narrowly) beat the DNNs on Broadcast News, our largest and most challenging data set, and one which has been used extensively in large scale speech recognition research. In Appendix A, we include more detailed tables comparing the various models we trained across all the metrics from Section 6.2.

6.5. Importance of the Number of Random Features

We will now illustrate the importance of the number of random features D on model performance. For this purpose, we trained kernel models on the BN-50 data set using $D \in \{5000, 10000, 25000, 50000, 100000\}$ for the three kernel functions, with and without feature selection, with a linear bottleneck of size 1000, and using heldout cross-entropy for learning rate decay. In Figure 3, we show how increasing the number of features dramat-

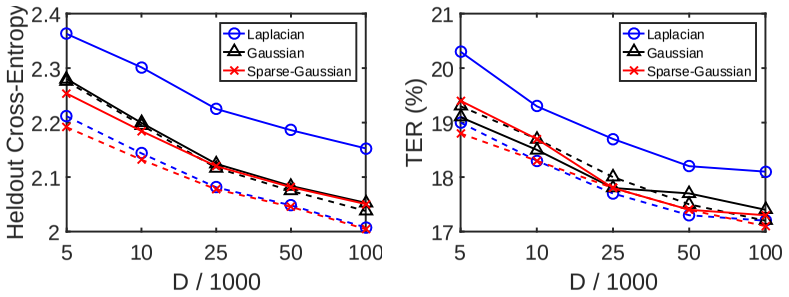


Figure 3: Heldout cross-entropy and development set TER performance of kernel acoustic models on BN-50, as a function of the number of random features D. Dashed lines mean feature selection was performed, while solid lines mean it was not.

ically improves the performance of the learned model, both in terms of cross-entropy and TER; there are diminishing returns, however, with small improvements in TER when increasing D from 50,000 to 100,000. Interestingly, we can also observe in these plots that the benefits from feature selection (gap between the dashed and solid lines) are large for the Laplacian kernel, modest for the sparse Gaussian kernel, and insignificant for the Gaussian kernel. We provide insight into why these trends occur in the following section.

6.6. Dynamics of Random Feature Selection

We now explore the feature selection dynamics. We first show that features selected in early iterations are likely to be selected in all subsequent iterations, demonstrating that the selection criterion is consistent. Second, we show that the feature selection process can be seen as way of learning to upweight important input features. This effect is pronounced for the Laplacian and sparse Gaussian kernels, but not for the Gaussian kernel, helping to explain why the former methods benefit from feature selection more than the latter.

In Figure 4, we plot the fraction of the s_t features selected in iteration t that remain in the model after all T iterations ("survival rate"). We show results for kernel models trained on the Cantonese data set, without using linear bottlenecks, and using heldout cross-entropy for learning rate decay. In nearly all iterations and for all kernels, over half of the selected features survive to the final model. For instance, over 90% of the Laplacian kernel features selected at iteration 10 survive the remaining 40 rounds of selection. For comparison, we plot the expected survival rate if at each iteration the features were chosen uniformly at random; for $s_t = Dt/T$, the expected survival rate for features selected at iteration t is $T!/(t! \cdot T^{T-t})$, which is exponentially small in T when $t \leq \beta T$ for any fixed $\beta < 1$.[13] For example, at $t = 10$ the expected survival rate is approximately 9×10^{-11} with $T = 50$. Thus, our selection criterion is orders of magnitude more consistent than random selection.

13. This can be shown using Stirling's formula. See Jameson (2015) for a useful review.

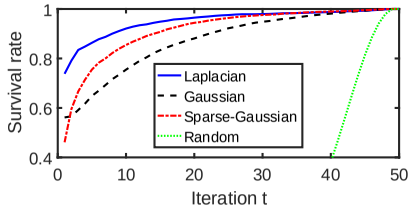


Figure 4: Fraction of the s_t features selected in iteration t that are in the final model (survival rate) for kernel models trained on the Cantonese data set.

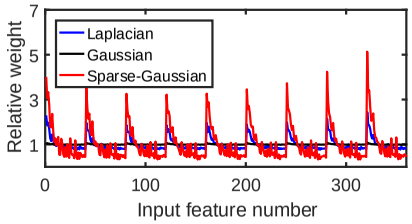


Figure 5: The relative weight of each input feature in the projection matrix Ω after feature selection completes, for kernel models trained on the Cantonese data set.

Finally, we consider how the feature selection process can be seen as a way of learning to assign more weight to the more important input features. Consider the final matrix of random vectors $\Omega := [\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_D] \in \mathbb{R}^{d \times D}$ after feature selection, for the Cantonese models from Figure 4. A coarse measure of how much influence an input feature $i \in \{1, 2, \ldots, d\}$ has in the final feature map is the relative "weight" of the i-th row of Ω. In Figure 5, we plot $\sum_{j=1}^{D} |\Omega_{i,j}|/Z$ for each input feature $i \in \{1, 2, \ldots, d\}$, where $Z = \frac{1}{d} \sum_{i,j} |\Omega_{i,j}|$ is a normalization term.[14] There is a strong periodic effect as a function of the input feature number. An examination of the feature pipeline from Kingsbury et al. (2013) reveals that this is because these features vectors are the concatenation of nine 40-dimensional acoustic feature vectors, where each set of 40 features is ordered by a measure of discriminative quality (via linear discriminant analysis). Thus, it is expected that the features with low $(i-1)$ mod 40 value may be more useful than the others; indeed, this is evident in the plot. Note that this effect exists, but is extremely weak, for the Gaussian kernel. We believe this is because Gaussian random vectors in $\mathbb{R}^d$ are likely to have all their entries be bounded in magnitude by $O(\sqrt{\log(d)})$. This observation helps explain why feature selection significantly improves the Laplacian and sparse Gaussian models, but not the Gaussian models.

14. For the Laplacian kernel, we discard the largest element in each of the d rows of Ω, because there are sometimes outliers which dominate the entire sum for their row.

7. Conclusion

In this paper, we explore the performance of kernel methods on large-scale ASR tasks, leveraging the kernel approximation technique of Rahimi and Recht (2007). We propose two new methods (feature selection, new early stopping criteria) which lead to large improvements in the performance of kernel acoustic models. We further show that using a linear bottleneck (Sainath et al., 2013a) to decompose the parameter matrices of these kernel models leads to significant improvements in performance as well. We validate these findings on four data sets, including the Broadcast News and TIMIT benchmark tasks. Using all these methods together, the kernel methods attain comparable speech recognition performance to DNNs across the four test sets; on Cantonese, TIMIT, and Broadcast News, the kernel models outperform the DNNs by 0.5%, 0.1%, and 0.1% TER absolute, respectively, whereas on Bengali, the DNN does better by 0.1% TER.

For future work, we are interested in continuing to test the limits of kernel methods on challenging empirical tasks. For example, can kernel methods be adapted to compete with convolutional and recurrent neural networks?

Acknowledgments

This research is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD / ARL) contract number W911NF-12-C-0012. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

F.S. is grateful to Lawrence K. Saul (UCSD), Léon Bottou (Facebook), Alex Smola (Amazon), and Chris J.C. Burges (Microsoft Research) for many fruitful discussions and pointers to relevant work. Additionally, A.B.G. is partially supported by a USC Provost Graduate Fellowship. F.S. is partially supported by a NSF IIS-1065243, 1451412, 1139148 a Google Research Award, an Alfred. P. Sloan Research Fellowship, an ARO YIP Award (W911NF-12-1-0241) and ARO Award W911NF-15-1-0484. A.B. is partially supported by a grant from CPER Nord-Pas de Calais/FEDER DATA Advanced data science and technologies 2015-2020. Computation for the work described in this paper was partially supported by the University of Southern California's Center for High-Performance Computing (`http://hpc.usc.edu`).

Appendix A. Detailed Empirical Results

In this appendix, we include tables comparing the models we trained in terms of four different metrics (CE, ENT, ERR, and ERLL). The notation is the same as in Tables 4 and 5. For both DNN and kernel models, 'NT' specifies that no "tricks" were used during training (no bottleneck, did not use ERLL for learning rate decay). A 'B' specifies that a linear bottleneck was used for the output parameter matrix, while an 'R' specifies that ERLL was used for learning rate decay (so 'BR' means both were used). For kernel models, '+FS' specifies that feature selection was performed for the corresponding row. The best result for each metric and data set is in bold.

Some important things to take note of in these tables:

- The linear bottleneck typically causes large drops in the average entropy of kernel models, while not having as strong or consistent an effect on cross-entropy. For DNNs, the bottleneck typically causes increases in cross-entropy, and relatively modest decreases in entropy.
- Using ERLL to determine learning rate decay typically causes increases in cross-entropy, and decreases in entropy, with the decrease in entropy typically being larger than the increase in cross-entropy. As a result, the ERLL is typically lower for models that use this method (with the exception of TIMIT DNN models).
- Feature selection typically results in large drops in cross-entropy, especially for Laplacian and sparse Gaussian kernels, while its effect on entropy is quite small. It thus helps lower heldout ERLL across the board, as well as TER in the vast majority of cases.

	Laplacian				Gaussian				Sparse Gaussian			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	1.34	1.32	1.35	1.39	1.35	1.33	1.36	1.34	1.31	**1.29**	1.34	1.33
+FS	1.28	**1.26**	1.29	1.27	1.35	1.31	1.36	1.35	1.28	1.26	1.31	1.27
BN-50	N/A	2.15	N/A	2.43	N/A	2.05	N/A	2.16	N/A	**2.05**	N/A	2.19
+FS	N/A	2.01	N/A	2.07	N/A	2.04	N/A	2.13	N/A	**2.00**	N/A	2.06
Cant.	**1.93**	1.95	1.95	2.04	1.99	1.98	2.00	2.04	1.93	1.94	1.95	2.00
+FS	**1.88**	1.90	1.89	1.95	1.97	1.97	1.98	2.03	1.90	1.91	1.91	1.96
TIMIT	0.97	0.99	0.97	1.07	0.94	0.96	0.94	1.02	0.94	0.95	**0.94**	1.03
+FS	0.92	0.95	0.92	1.03	0.93	0.96	0.93	1.02	**0.92**	0.96	0.92	1.03

Table 9: Kernel: Metric CE

	1000				2000				4000			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	1.25	1.26	1.24	1.27	**1.24**	1.26	1.26	1.32	1.24	1.25	1.30	1.39
BN-50	2.05	2.05	2.04	2.08	2.01	2.04	2.05	2.22	**2.00**	2.03	2.09	2.27
Cant.	1.92	1.96	1.92	1.98	1.93	1.94	1.97	2.06	**1.92**	1.97	2.03	2.10
TIMIT	**1.06**	1.08	1.20	1.28	1.08	1.09	1.25	1.31	1.10	1.11	1.25	1.33

Table 10: DNN: Metric CE

	Laplacian				Gaussian				Sparse Gaussian			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	1.43	1.23	1.41	**1.08**	1.36	1.31	1.35	1.28	1.35	1.23	1.30	1.10
+FS	1.32	1.21	1.28	1.14	1.44	1.27	1.45	**1.13**	1.32	1.22	1.26	1.14
BN-50	N/A	1.89	N/A	**1.46**	N/A	1.83	N/A	1.53	N/A	1.81	N/A	1.48
+FS	N/A	1.81	N/A	1.56	N/A	1.84	N/A	**1.55**	N/A	1.80	N/A	1.57
Cant.	1.84	1.67	1.76	**1.52**	1.94	1.73	1.91	1.58	1.77	1.69	1.70	1.55
+FS	1.75	1.66	1.73	1.54	1.91	1.72	1.87	1.57	1.75	1.68	1.72	**1.54**
TIMIT	0.95	0.72	0.91	0.61	0.88	0.73	0.86	0.62	0.89	0.76	0.85	**0.61**
+FS	0.86	0.70	0.82	0.58	0.86	0.70	0.83	0.61	0.84	0.69	0.82	**0.58**

Table 11: Kernel: Metric ENT

	1000				2000				4000			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	1.23	1.17	1.18	1.09	1.18	1.13	1.09	0.99	1.14	1.11	1.02	**0.91**
BN-50	1.95	1.77	1.90	1.68	1.76	1.68	1.65	1.40	1.65	1.60	1.48	**1.27**
Cant.	1.71	1.67	1.67	1.57	1.66	1.64	1.55	1.42	1.63	1.55	1.43	**1.38**
TIMIT	0.72	0.70	0.58	0.53	0.63	0.63	0.50	0.48	0.57	0.57	0.48	**0.45**

Table 12: DNN: Metric ENT

	Laplacian				Gaussian				Sparse Gaussian			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	2.77	2.55	2.76	2.47	2.71	2.65	2.71	2.62	2.67	2.52	2.64	**2.44**
+FS	2.60	2.47	2.57	**2.41**	2.79	2.58	2.80	2.48	2.60	2.48	2.57	2.41
BN-50	N/A	4.04	N/A	3.88	N/A	3.88	N/A	3.69	N/A	3.86	N/A	**3.67**
+FS	N/A	3.82	N/A	3.63	N/A	3.88	N/A	3.67	N/A	3.80	N/A	**3.62**
Cant.	3.77	3.62	3.71	3.56	3.94	3.71	3.91	3.62	3.71	3.63	3.65	**3.54**
+FS	3.63	3.56	3.63	**3.49**	3.88	3.69	3.86	3.60	3.64	3.58	3.63	3.50
TIMIT	1.92	1.71	1.87	1.68	1.82	1.70	1.80	1.65	1.83	1.71	1.79	**1.64**
+FS	1.78	1.65	1.74	1.61	1.79	1.67	1.76	1.64	1.76	1.64	1.74	**1.61**

Table 13: Kernel: Metric ERLL

	1000				2000				4000			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	2.48	2.43	2.43	2.37	2.42	2.39	2.35	2.31	2.39	2.37	2.32	**2.30**
BN-50	3.99	3.82	3.94	3.76	3.77	3.72	3.70	3.63	3.65	3.63	3.58	**3.55**
Cant.	3.63	3.63	3.58	3.56	3.59	3.58	3.51	3.48	3.55	3.52	**3.46**	3.47
TIMIT	1.77	1.77	1.77	1.81	1.71	1.72	1.76	1.79	**1.67**	1.68	1.73	1.78

Table 14: DNN: Metric ERLL

	Laplacian				Gaussian				Sparse Gaussian			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	0.30	0.30	0.31	0.31	0.31	0.31	0.31	0.31	0.30	**0.30**	0.30	0.30
+FS	0.29	**0.29**	0.30	0.29	0.31	0.31	0.31	0.31	0.30	0.29	0.30	0.30
BN-50	N/A	0.52	N/A	0.54	N/A	**0.50**	N/A	0.51	N/A	0.50	N/A	0.51
+FS	N/A	0.49	N/A	0.50	N/A	0.50	N/A	0.50	N/A	**0.49**	N/A	0.49
Cant.	**0.43**	0.44	0.44	0.44	0.45	0.45	0.45	0.45	0.43	0.44	0.44	0.44
+FS	**0.43**	0.43	0.43	0.44	0.44	0.44	0.44	0.45	0.43	0.44	0.43	0.44
TIMIT	0.32	0.32	0.32	0.33	0.31	0.32	0.31	0.32	0.31	0.31	**0.31**	0.32
+FS	0.31	0.31	0.31	0.31	0.31	0.31	0.31	0.32	0.31	0.31	**0.31**	0.31

Table 15: Kernel: Metric ERR

	1000				2000				4000			
	NT	B	R	BR	NT	B	R	BR	NT	B	R	BR
Beng.	0.29	0.29	0.29	0.29	**0.29**	0.29	0.29	0.30	0.29	0.29	0.29	0.30
BN-50	0.50	0.50	0.50	0.50	0.49	0.50	0.50	0.51	**0.49**	0.49	0.50	0.51
Cant.	0.44	0.44	**0.44**	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44
TIMIT	0.33	0.33	0.34	0.34	0.33	0.33	0.34	0.34	0.33	**0.32**	0.33	0.33

Table 16: DNN: Metric ERR

Appendix B. Derivation of Functional Form for Random Fourier Features

In this appendix, we will prove that for a properly-scaled (i.e., $Z = 1$) positive-definite shift-invariant kernel k,

$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{\omega},b}\left[\sqrt{2}\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x} + b) \cdot \sqrt{2}\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}' + b)\right], \tag{7}$$

where $\boldsymbol{\omega}$ is drawn from $p(\boldsymbol{\omega})$, the inverse Fourier transform of k, and b is drawn uniformly from $[0, 2\pi]$. We begin this proof using Equation (3) from Section 3.1:

$$\begin{aligned}
k(\boldsymbol{x}, \boldsymbol{x}') &= \int_{R^d} p(\boldsymbol{\omega}) e^{j\boldsymbol{\omega}^{\mathrm{T}}(\boldsymbol{x}-\boldsymbol{x}')}\, d\boldsymbol{\omega} \\
&= \mathbb{E}_{\boldsymbol{\omega}}\left[e^{j\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}} e^{-j\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}'}\right] \\
&= \mathbb{E}_{\boldsymbol{\omega}}\left[\left(\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}) + j\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\right)\left(\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}') - j\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\right)\right] \\
&= \mathbb{E}_{\boldsymbol{\omega}}\left[\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}') + \sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\right] \\
&\quad + j \cdot \mathbb{E}_{\boldsymbol{\omega}}\left[\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}') - \sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\right] \\
&= \mathbb{E}_{\boldsymbol{\omega}}\left[\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}') + \sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\right] \qquad (8)
\end{aligned}$$

Note the Equation (8) is true because we know that $k(\boldsymbol{x}, \boldsymbol{x}')$ is a real-valued function, and thus the imaginary part of the expectation must disappear. We now show that the right-hand side of Equation (7) is equal to this same expression:

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\omega},b}\left[\sqrt{2}\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x} + b) \cdot \sqrt{2}\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}' + b)\right] \\
&\quad = 2 \cdot \mathbb{E}_{\boldsymbol{\omega},b}\Bigg[\left(\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\cos(b) - \sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\sin(b)\right) \cdot \\
&\qquad\qquad \left(\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\cos(b) - \sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\sin(b)\right)\Bigg] \qquad (9) \\
&\quad = 2 \cdot \mathbb{E}_{\boldsymbol{\omega},b}\Big[\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\cos^2(b) \\
&\qquad\qquad - \cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\cos(b)\sin(b) \\
&\qquad\qquad - \sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\cos(b)\sin(b) \\
&\qquad\qquad + \sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\sin^2(b)\Big] \\
&\quad = 2 \cdot \mathbb{E}_{\boldsymbol{\omega}}\left[\frac{1}{2}\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}') + \frac{1}{2}\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\right] \qquad (10) \\
&\quad = \mathbb{E}_{\boldsymbol{\omega}}\left[\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\cos(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}') + \sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x})\sin(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}')\right] \\
&\quad = k(\boldsymbol{x}, \boldsymbol{x}')
\end{aligned}$$

Equation (9) is true by the cosine sum of angles formula, and Equation (10) is true because $\mathbb{E}_b\left[\cos^2(b)\right] = \mathbb{E}_b\left[\sin^2(b)\right] = \int_0^{2\pi} \frac{1}{2\pi}\sin^2(b) = \frac{1}{2}$, and because $\mathbb{E}_b\left[\sin(b)\cos(b)\right] = 0$. This concludes the proof. ■

References

Naman Agarwal, Zeyuan Allen Zhu, Brian Bullins, Elad Hazan, and Tengyu Ma. Finding approximate local minima faster than gradient descent. In *STOC*, 2017.

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Erich Elsen, Jesse Engel, Linxi Fan, Christopher Fougner, Awni Y. Hannun, Billy Jun, Tony Han, Patrick LeGresley, Xiangang Li, Libby Lin, Sharan Narang, Andrew Y. Ng, Sherjil Ozair, Ryan Prenger, Sheng Qian, Jonathan Raiman, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Chong Wang, Yi Wang, Zhiqian Wang, Bo Xiao, Yan Xie, Dani Yogatama, Jun Zhan, and Zhenyao Zhu. Deep Speech 2 : End-to-end speech recognition in English and Mandarin. In *ICML*, 2016.

Animashree Anandkumar and Rong Ge. Efficient approaches for escaping higher order saddle points in non-convex optimization. In *COLT*, 2016.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. Globally normalized transition-based neural networks. In *ACL*, 2016.

Devansh Arpit, Stanislaw K. Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron C. Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In *ICML*, 2017.

Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *NIPS*, 2014.

Lalit R Bahl, Peter F Brown, Peter V De Souza, and Robert L Mercer. Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *ICASSP*, 1986.

Peter L. Bartlett. For valid generalization the size of the weights is more important than the size of the network. In *NIPS*, 1996.

Yoshua Bengio, Yann LeCun, et al. Scaling learning algorithms towards AI. *Large-Scale Kernel Machines*, 34(5):1–41, 2007.

Monica Bianchini and Franco Scarselli. On the complexity of neural network classifiers: A comparison between shallow and deep architectures. *IEEE Trans. Neural Netw. Learning Syst.*, 25(8):1553–1565, 2014.

Léon Bottou, Olivier Chapelle, Dennis DeCoste, and Jason Weston. *Large-Scale Kernel Machines*. MIT Press, 2007.

William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *ICASSP*, pages 4960–4964. IEEE, 2016.

Jie Chen, Lingfei Wu, Kartik Audhkhasi, Brian Kingsbury, and Bhuvana Ramabhadran. Efficient one-vs-one kernel ridge regression for speech recognition. In *ICASSP*, 2016.

Chung-Cheng Chiu, Tara N. Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J. Weiss, Kanishka Rao, Ekaterina Gonina, Navdeep Jaitly, Bo Li, Jan Chorowski, and Michiel Bacchiani. State-of-the-art speech recognition with sequence-to-sequence models. In *ICASSP*, 2018.

Anna Choromanska, Mikael Henaff, Michaël Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *AISTATS*, 2015.

Kenneth L. Clarkson. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. *ACM Trans. Algorithms*, 6(4):63:1–63:30, 2010.

George Cybenko. Approximation by superpositions of a sigmoidal function. *MCSS*, 2(4): 303–314, 1989.

George E. Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Trans. Audio, Speech & Language Processing*, 20(1):30–42, 2012.

Bo Dai, Bo Xie, Niao He, Yingyu Liang, Anant Raj, Maria-Florina Balcan, and Le Song. Scalable kernel methods via doubly stochastic gradients. In *NIPS*, 2014.

Yann N. Dauphin, Razvan Pascanu, Çaglar Gülçehre, KyungHyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *NIPS*, 2014.

Dennis DeCoste and Bernhard Schölkopf. Training invariant support vector machines. *Machine Learning*, 46(1-3):161–190, 2002.

Najim Dehak, Patrick Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *IEEE Trans. Audio, Speech & Language Processing*, 19(4):788–798, 2011.

John C. Duchi and Yoram Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2899–2934, 2009.

Jonathan Fiscus, George Doddington, Audrey Le, Greg Sanders, Mark Przybocki, and David Pallett. 2003 NIST Rich Transcription evaluation data. *Linguistic Data Consortium*, 2003. URL `https://catalog.ldc.upenn.edu/LDC2007S10`.

Mark J. F. Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech & Language*, 12(2):75–98, 1998.

Mark J. F. Gales and Steve J. Young. The application of hidden Markov models in speech recognition. *Foundations and Trends in Signal Processing*, 1(3):195–304, 2007.

John S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathan G. Fiscus, David S. Pallett, Nancy L. Dahlgren, and Victor Zue. TIMIT acoustic phonetic continuous speech corpus. *Linguistic Data Consortium*, 1993. URL `https://catalog.ldc.upenn.edu/LDC93S1`.

Matthew Gibson and Thomas Hain. Hypothesis spaces for minimum Bayes risk training in large vocabulary speech recognition. In *INTERSPEECH*, 2006.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *ICML*, 2006.

Raffay Hamid, Ying Xiao, Alex Gittens, and Dennis DeCoste. Compact random feature maps. In *ICML*, 2014.

Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural network. In *NIPS*, 2015.

Wolfgang Karl Härdle, Marlene Müller, Stefan Sperlich, and Axel Werwatz. *Nonparametric and semiparametric models*. Springer Science & Business Media, 2004.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29, 2012.

Kurt Hornik, Maxwell B. Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

Po-Sen Huang, Haim Avron, Tara N. Sainath, Vikas Sindhwani, and Bhuvana Ramabhadran. Kernel methods match deep neural networks on TIMIT. In *ICASSP*, 2014.

G.J.O. Jameson. A simple proof of Stirling's formula for the gamma function. *The Mathematical Gazette*, 99(544):68–74, 2015.

Janez Kaiser, Bogomir Horvat, and Zdravko Kacic. A novel loss function for the overall risk criterion based discriminative training of HMM models. In *INTERSPEECH*, 2000.

Purushottam Kar and Harish Karnick. Random feature maps for dot product kernels. In *AISTATS*, 2012.

Brian Kingsbury. Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling. In *ICASSP*, 2009.

Brian Kingsbury, Jia Cui, Xiaodong Cui, Mark J. F. Gales, Kate Knill, Jonathan Mamou, Lidia Mangu, David Nolden, Michael Picheny, Bhuvana Ramabhadran, Ralf Schlüter, Abhinav Sethy, and Philip C. Woodland. A high-performance Cantonese keyword search system. In *ICASSP*, 2013.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

Quoc V. Le, Tamás Sarlós, and Alexander J. Smola. Fastfood - computing Hilbert space expansions in loglinear time. In *ICML*, 2013.

Zhiyun Lu, Dong Guo, Alireza Bagheri Garakani, Kuan Liu, Avner May, Aurélien Bellet, Linxi Fan, Michael Collins, Brian Kingsbury, Michael Picheny, and Fei Sha. A comparison between deep neural nets and kernel acoustic models for speech recognition. In *ICASSP*, 2016.

Avner May, Michael Collins, Daniel J. Hsu, and Brian Kingsbury. Compact kernel models for acoustic modeling via random feature selection. In *ICASSP*, 2016.

Charles A. Micchelli, Yuesheng Xu, and Haizhang Zhang. Universal kernels. *Journal of Machine Learning Research*, 7:2651–2667, 2006.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, 2010.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *ICLR Workshop*, 2013.

Abdel-rahman Mohamed, George E. Dahl, and Geoffrey E. Hinton. Acoustic modeling using deep belief networks. *IEEE Trans. Audio, Speech & Language Processing*, 20(1):14–22, 2012.

Guido F. Montúfar, Razvan Pascanu, KyungHyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *NIPS*, 2014.

N. Morgan and H. Bourlard. Generalization and parameter estimation in feedforward nets: Some experiments. In *NIPS*, 1990.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. In *ICLR (Workshop)*, 2015.

Jeffrey Pennington and Yasaman Bahri. Geometry of neural network loss surfaces via random matrix theory. In *ICML*, 2017.

Jeffrey Pennington, Felix X. Yu, and Sanjiv Kumar. Spherical random features for polynomial kernels. In *NIPS*, 2015.

John C. Platt. Fast Training of Support Vector Machines using Sequential Minimal Optimization. In *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.

Daniel Povey and Brian Kingsbury. Evaluation of proposed modifications to MPE for large scale discriminative training. In *ICASSP*, 2007.

Daniel Povey and Philip C. Woodland. Minimum phone error and I-smoothing for improved discriminative training. In *ICASSP*, 2002.

Daniel Povey, Dimitri Kanevsky, Brian Kingsbury, Bhuvana Ramabhadran, George Saon, and Karthik Visweswariah. Boosted MMI for model and feature-space discriminative training. In *ICASSP*, 2008.

Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur. Purely sequence-trained neural networks for ASR based on lattice-free MMI. In *INTERSPEECH*, 2016.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NIPS*, 2007.

Tara N. Sainath, Brian Kingsbury, Bhuvana Ramabhadran, Petr Fousek, Petr Novák, and Abdel-rahman Mohamed. Making deep belief networks effective for large vocabulary continuous speech recognition. In *ASRU*, 2011.

Tara N. Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *ICASSP*, 2013a.

Tara N. Sainath, Brian Kingsbury, Hagen Soltau, and Bhuvana Ramabhadran. Optimization techniques to improve training speed of deep neural networks for large speech tasks. *IEEE Trans. Audio, Speech & Language Processing*, 21(11):2267–2276, 2013b.

Tara N. Sainath, Abdel-rahman Mohamed, Brian Kingsbury, and Bhuvana Ramabhadran. Deep convolutional neural networks for LVCSR. In *ICASSP*, 2013c.

Hasim Sak, Andrew W. Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *INTERSPEECH*, 2014.

George Saon, Tom Sercu, Steven J. Rennie, and Hong-Kwang Jeff Kuo. The IBM 2016 English conversational telephone speech recognition system. In *INTERSPEECH*, 2016.

George Saon, Gakuto Kurata, Tom Sercu, Kartik Audhkhasi, Samuel Thomas, Dimitrios Dimitriadis, Xiaodong Cui, Bhuvana Ramabhadran, Michael Picheny, Lynn-Li Lim, Bergul Roomi, and Phil Hall. English conversational telephone speech recognition by humans and machines. In *INTERSPEECH*, 2017.

B. Schölkopf and A. Smola. *Learning with kernels*. MIT Press, 2002.

Frank Seide, Gang Li, Xie Chen, and Dong Yu. Feature engineering in context-dependent deep neural networks for conversational speech transcription. In *ASRU*, 2011a.

Frank Seide, Gang Li, and Dong Yu. Conversational speech transcription using context-dependent deep neural networks. In *INTERSPEECH*, 2011b.

Tom Sercu and Vaibhava Goel. Advances in very deep convolutional neural networks for LVCSR. In *INTERSPEECH*, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Hagen Soltau, George Saon, and Brian Kingsbury. The IBM Attila speech recognition toolkit. In *SLT*, 2010.

Hagen Soltau, George Saon, and Tara N. Sainath. Joint training of convolutional and non-convolutional neural networks. In *ICASSP*, 2014.

Sören Sonnenburg and Vojtech Franc. COFFIN: a computational framework for linear SVMs. In *ICML*, 2010.

Ingo Steinwart. Sparseness of support vector machines—some asymptotically sharp bounds. In *NIPS*, 2003.

Nikko Ström. Sparse connection and pruning in large dynamic artificial neural networks. In *EUROSPEECH*, 1997.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. LSTM neural networks for language modeling. In *INTERSPEECH*, 2012.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

Ivor W. Tsang, James T. Kwok, and Pak-Ming Cheung. Core vector machines: Fast SVM training on very large data sets. *Journal of Machine Learning Research*, 6:363–392, 2005.

V. Valtchev, J. J. Odell, Philip C. Woodland, and Steve J. Young. MMIE training of large vocabulary recognition systems. *Speech Communication*, 22(4):303–314, 1997.

Ewout van den Berg, Bhuvana Ramabhadran, and Michael Picheny. Training variance and performance evaluation of neural networks in speech. In *ICASSP*, 2017.

Andrea Vedaldi and Andrew Zisserman. Efficient additive kernels via explicit feature maps. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(3):480–492, 2012.

Karel Veselý, Arnab Ghoshal, Lukás Burget, and Daniel Povey. Sequence-discriminative training of deep neural networks. In *INTERSPEECH*, 2013.

Christopher K. I. Williams and Matthias W. Seeger. Using the Nyström method to speed up kernel machines. In *NIPS*, 2000.

Bo Xie, Yingyu Liang, and Le Song. Diverse neural network learns true target functions. In *AISTATS*, 2017.

Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Michael L. Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Toward human parity in conversational speech recognition. *IEEE/ACM Trans. Audio, Speech & Language Processing*, 25(12):2410–2423, 2017.

Jian Xue, Jinyu Li, and Yifan Gong. Restructuring of deep neural network acoustic models with singular value decomposition. In *INTERSPEECH*, 2013.

Zichao Yang, Marcin Moczulski, Misha Denil, Nando de Freitas, Alexander J. Smola, Le Song, and Ziyu Wang. Deep fried convnets. In *ICCV*, 2015.

Ian En-Hsu Yen, Ting-Wei Lin, Shou-De Lin, Pradeep Ravikumar, and Inderjit S. Dhillon. Sparse random feature algorithm as coordinate descent in Hilbert space. In *NIPS*, 2014.

Felix X. Yu, Sanjiv Kumar, Henry A. Rowley, and Shih-Fu Chang. Compact nonlinear maps and circulant extensions. *arXiv preprint arXiv:1503.03893*, 2015.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.

Journal of Machine Learning Research 20 (2019) 1-49 Submitted 6/17; Revised 2/19; Published 5/19

Robust Estimation of Derivatives Using Locally Weighted Least Absolute Deviation Regression

WenWu Wang WENGEWSH@SINA.COM
School of Statistics
Qufu Normal University
Jingxuan West Road, Qufu, Shandong, China
Ping Yu PINGYU@HKU.HK
Faculty of Business and Economics
University of Hong Kong
Pokfulam Road, Hong Kong
Lu Lin LINLU@SDU.EDU.CN
Zhongtai Securities Institute for Financial Studies
Shandong University
Jinan, Shandong, China
Tiejun Tong TONGT@HKBU.EDU.HK
Department of Mathematics
Hong Kong Baptist University
Kowloon Tong, Hong Kong

Editor: Zhihua Zhang

Abstract

In nonparametric regression, the derivative estimation has attracted much attention in recent years due to its wide applications. In this paper, we propose a new method for the derivative estimation using the locally weighted least absolute deviation regression. Different from the local polynomial regression, the proposed method does not require a finite variance for the error term and so is robust to the presence of heavy-tailed errors. Meanwhile, it does not require a zero median or a positive density at zero for the error term in comparison with the local median regression. We further show that the proposed estimator with random difference is asymptotically equivalent to the (infinitely) composite quantile regression estimator. In other words, running one regression is equivalent to combining infinitely many quantile regressions. In addition, the proposed method is also extended to estimate the derivatives at the boundaries and to estimate higher-order derivatives. For the equidistant design, we derive theoretical results for the proposed estimators, including the asymptotic bias and variance, consistency, and asymptotic normality. Finally, we conduct simulation studies to demonstrate that the proposed method has better performance than the existing methods in the presence of outliers and heavy-tailed errors, and analyze the Chinese house price data for the past ten years to illustrate the usefulness of the proposed method.

Keywords: composite quantile regression, differenced method, LowLAD, LowLSR, outlier and heavy-tailed error, robust nonparametric derivative estimation

©2019 WenWu Wang, Ping Yu, Lu Lin, and Tiejun Tong.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-340.html.

1. Introduction

The derivative estimation is an important problem in nonparametric regression and it has applications in a wide range of fields. For instance, when analyzing human growth data (Müller, 1988; Ramsay and Silverman, 2002) or maneuvering target tracking data (Li and Jilkov, 2003, 2010), the first- and second-order derivatives of the height as a function of time are two important parameters, with the first-order derivative representing the speed and the second-order derivative representing the acceleration. The derivative estimates are also needed in change-point problems, e.g., for exploring the structures of curves (Chaudhuri and Marron, 1999; Gijbels and Goderniaux, 2005), for detecting the extremum of derivatives (Newell et al., 2005), for characterizing submicroscopic nanoparticle from scattering data (Charnigo et al., 2007, 2011a), for comparing regression curves (Park and Kang, 2008), for detecting abrupt climate changes (Matyasovszky, 2011), and for inferring the cell growth rates (Swain et al., 2016).

In the existing literature, one usually obtains the derivative estimates as a by-product by taking the derivative of a nonparametric fit of the regression function. There are three main approaches for the derivative estimation: smoothing spline, local polynomial regression, and differenced estimation. For smoothing spline, the derivatives are estimated by taking derivatives of the spline estimation of the regression function (Stone, 1985; Zhou and Wolfe, 2000). For local polynomial regression, a polynomial using the Taylor expansion is fitted locally by the kernel method (Ruppert and Wand, 1994; Fan and Gijbels, 1996; Delecroix and Rosa, 1996). These two methods both require an estimate of the regression function. As pointed out in Wang and Lin (2015), when the regression function estimator achieves the optimal rate of convergence, the corresponding derivative estimators may fail to achieve the rate. In other words, minimizing the mean square error of the regression function estimator does not necessarily guarantee the derivatives be optimally estimated (Wahba and Wang, 1990; Charnigo et al., 2011b).

For the differenced estimation, Müller et al. (1987) and Härdle (1990) proposed a cross-validation method to estimate the first-order derivative without estimating the regression function. Unfortunately, their method may not perform well in practice as the variance of their estimator is proportional to n^2 when the design points are equally spaced. Observing this shortcoming, Charnigo et al. (2011b) and De Brabanter et al. (2013) proposed a variance-reducing estimator for the derivative function called the empirical derivative that is essentially a linear combination of the symmetric difference quotients. They further derived the order of the asymptotic bias and variance, and established the consistency of the empirical derivative. Wang and Lin (2015) represented the empirical derivative as a local constant estimator in locally weighted least squares regression (LowLSR), and proposed a new estimator for the derivative function to reduce the estimation bias in both valleys and peaks of the true derivative function. More recently, Dai et al. (2016) generalized equidistant design to non-equidistant design, and Liu and De Brabanter (2018) further generalized the existing work to random design.

The aforementioned differenced derivative estimators are all based on the least squares (LS) method. Although elegant, the least squares method is not robust to outliers (Huber and Ronchetti, 2009). To overcome this problem, various robust methods have been proposed in the literature to improve the estimation of the regression function, see, for ex-

ample, kernel M-smoother (Härdle and Gasser, 1984), local least absolute deviation (LAD) (Fan and Hall, 1994; Wang and Scott, 1994), and locally weighted least squares (Cleveland, 1979; Ruppert and Wand, 1994) among others. In contrast, little attention has been paid to improving the derivative estimation except for the parallel developments of the above remedies (Härdle and Gasser, 1985; Welsh, 1996; Boente and Rodriguez, 2006), so call for a better solution.

In this paper, we propose a locally weighted least absolute deviation (LowLAD) method by combining the differenced method and the L_1 regression systematically. Over a neighborhood centered at a fixed point, we first obtain a sequence of linear regression representation in which the derivative is the intercept term. We then estimate the derivative by minimizing the sum of weighted absolute errors. By repeating this local fitting over a grid of points, we can obtain the derivative estimates on a discrete set of points. Finally, the entire derivative function is obtained by applying the local polynomial regression or the cubic spline interpolation.

The rest of the paper is organized as follows. Section 2 presents the motivation, the first-order derivative estimator and its theoretical properties, including the asymptotic bias and variance, consistency, and asymptotic normality. Section 3 studies the relation between the LowLAD estimator and the existing estimators. In particular, we show that the LowLAD estimator with random difference is asymptotically equivalent to the (infinitely) composite quantile regression estimator. Section 4 derives the first-order derivative estimation at the boundaries of the domain, and Section 5 generalizes the proposed method to estimate the higher-order derivatives. In Section 6 we conduct extensive simulation studies to assess the finite-sample performance of the proposed estimators and compare them with the existing competitors; we also apply our method to a real data set to illustrate its usefulness in practice. Finally, we conclude the paper with some discussions in Section 7, and provide the proofs of the theoretical results in six Appendices.

A word on notation: $\doteq$ means that the higher-order terms are omitted, and $\approx$ means an approximate result with up to two decimal digits.

2. First-Order Derivative Estimation

Combining the differenced method and the L_1 regression, we propose the LowLAD regression to estimate the first-order derivative. The new method inherits the advantage of the differenced method and also the robustness of the L_1 method.

2.1. Motivation

Consider the nonparametric regression model

$$Y_i = m(x_i) + \epsilon_i, \quad 1 \le i \le n, \tag{1}$$

where $x_i = i/n$ is the design point, Y_i is the response variable, $m(\cdot)$ is an unknown regression function, and ϵ_i are independent and identically distributed (iid) random errors with a continuous density $f(\cdot)$.

We first define first-order symmetric (about i) difference quotient (Charnigo et al., 2011b; De Brabanter et al., 2013) as

$$Y_{ij}^{(1)} = \frac{Y_{i+j} - Y_{i-j}}{x_{i+j} - x_{i-j}}, \quad 1 \le j \le k, \tag{2}$$

where k is a positive integer, and then decompose $Y_{ij}^{(1)}$ into two parts as

$$Y_{ij}^{(1)} = \frac{m(x_{i+j}) - m(x_{i-j})}{2j/n} + \frac{\epsilon_{i+j} - \epsilon_{i-j}}{2j/n}, \quad 1 \le j \le k. \tag{3}$$

On the right hand side of (3), the first term contains the bias information of the true derivative, and the second term contains the variance information. By Wang and Lin (2015), the first-order derivative estimation based on the third-order Taylor expansion usually outperforms the estimation based on the first-order Taylor expansion due to bias correction. For the same reason, we assume that $m(\cdot)$ is three times continuously differentiable on $[0, 1]$. By the Taylor expansion, we obtain

$$\frac{m(x_{i+j}) - m(x_{i-j})}{2j/n} = m^{(1)}(x_i) + \frac{m^{(3)}(x_i)}{6}\frac{j^2}{n^2} + o\left(\frac{j^2}{n^2}\right), \tag{4}$$

where the estimation bias is contained in the remainder term of the Taylor expansion.

By (3) and (4), we have

$$Y_{ij}^{(1)} = m^{(1)}(x_i) + \frac{m^{(3)}(x_i)}{6}\frac{j^2}{n^2} + \frac{\epsilon_{i+j} - \epsilon_{i-j}}{2j/n} + o\left(\frac{j^2}{n^2}\right). \tag{5}$$

In Proposition 11 (see Appendix A), we show that the median of $\epsilon_{i+j} - \epsilon_{i-j}$ is always zero, no matter whether the median of ϵ_i is zero or not. As a result, for any fixed $k = o(n)$, we have

$$\text{Median}[Y_{ij}^{(1)}] = m^{(1)}(x_i) + \frac{m^{(3)}(x_i)}{6}d_j^2 + o\left(d_j^2\right), \quad 1 \le j \le k, \tag{6}$$

where $d_j = j/n$. We treat (6) as a linear regression with d_j^2 and $Y_{ij}^{(1)}$ as the independent and dependent variables, respectively. In the presence of heavy-tailed errors, we propose to estimate $m^{(1)}(x_i)$ as the intercept of the linear regression using the LowLAD method.

2.2. Estimation Methodology

In order to derive the estimation bias, we further assume that $m(\cdot)$ is five times continuously differentiable, that is, the regression function is two degrees smoother than our postulated model due to the equidistant design. Following the paradigm of Draper and Smith (1981) and Wang and Scott (1994), we discard the higher-order terms of $m(\cdot)$ and assume locally that the approximate model is

$$Y_{ij}^{(1)} = \beta_{i1} + \beta_{i3}d_j^2 + \beta_{i5}d_j^4 + \zeta_{ij},$$

where $\beta_i = (\beta_{i1}, \beta_{i3}, \beta_{i5})^T = (m^{(1)}(x_i), m^{(3)}(x_i)/6, m^{(5)}(x_i)/120)^T$ are the unknown coefficients of the true underlying quintic function, and $\zeta_{ij} = (\epsilon_{i+j} - \epsilon_{i-j})/(2j/n)$ with

Median$[\zeta_{ij}]=0$. Under the assumption of the approximate model, β_{i1} can be estimated as

$$\min_b \sum_{j=1}^{k} w_j \left|Y_{ij}^{(1)} - (b_{i1}+b_{i3}d_j^2+b_{i5}d_j^4)\right| = \min_b \sum_{j=1}^{k} \left|\tilde{Y}_{ij}^{(1)} - (b_{i1}d_j+b_{i3}d_j^3+b_{i5}d_j^5)\right|,$$

where $w_j=d_j$ are the weights, $b=(b_{i1},b_{i3},b_{i5})^T$, and $\tilde{Y}_{ij}^{(1)}=(Y_{i+j}-Y_{i-j})/2$. Accordingly, the approximate model can be rewritten as

$$\tilde{Y}_{ij}^{(1)} = \beta_{i1}d_j+\beta_{i3}d_j^3+\beta_{i5}d_j^5+\tilde{\zeta}_{ij},$$

where $\tilde{\zeta}_{ij}=(\epsilon_{i+j}-\epsilon_{i-j})/2$ are iid random errors with Median$[\tilde{\zeta}_{ij}]=0$ and a continuous, symmetric density $g(\cdot)$ which is positive in a neighborhood of zero (see Appendix A).

Rather than the best L_1 quintic fitting, we search for the best L_1 cubic fitting to $\tilde{Y}_{ij}^{(1)}$. Specifically, we estimate the model by LowLAD:

$$(\hat{\beta}_{i1},\hat{\beta}_{i3}) = \arg\min_b \sum_{j=1}^{k} \left|\tilde{Y}_{ij}^{(1)} - b_{i1}d_j - b_{i3}d_j^3\right|$$

with $b=(b_{i1},b_{i3})^T$, and define the LowLAD estimator of $m^{(1)}(x_i)$ as

$$\hat{m}_{\text{LowLAD}}^{(1)}(x_i) = \hat{\beta}_{i1}. \tag{7}$$

The following theorem states the asymptotic behavior of $\hat{\beta}_{i1}$.

Theorem 1 *Assume that ϵ_i are iid random errors with a continuous bounded density $f(\cdot)$. Then as $k\to\infty$ and $k/n\to 0$, $\hat{\beta}_{i1}$ in (7) is asymptotically normally distributed with*

$$\text{Bias}[\hat{\beta}_{i1}] = -\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4}+o\left(\frac{k^4}{n^4}\right), \quad \text{Var}[\hat{\beta}_{i1}] = \frac{75}{16g(0)^2}\frac{n^2}{k^3}+o\left(\frac{n^2}{k^3}\right),$$

where $g(0)=2\int_{-\infty}^{\infty}f^2(x)\mathrm{d}x$. The optimal k that minimizes the asymptotic mean square error (AMSE) is

$$k_{opt} \approx 3.26\left(\frac{1}{g(0)^2 m^{(5)}(x_i)^2}\right)^{1/11} n^{10/11},$$

and, consequently, the minimum AMSE is

$$\text{AMSE}[\hat{\beta}_{i1}] \approx 0.19\left(\frac{m^{(5)}(x_i)^6}{g(0)^{16}}\right)^{1/11} n^{-8/11}.$$

In the local median regression (see Section 3.1 below for its definition), the density $f(\cdot)$ is usually assumed to have a zero median and a positive $f(0)$ value. While in Theorem 1, we only require a continuity condition on the density $f(\cdot)$. In addition, the variance of the LowLAD estimator depends on $g(0)=2\int_{-\infty}^{\infty}f^2(x)\mathrm{d}x$ which is always positive, while the variance of the local median estimator relies on a single value $f(0)$ only. In this sense, the LowLAD estimator is more robust than the local median estimator.

2.3. LowLAD with Random Difference

To further improve the estimation efficiency, we propose the LowLAD with random difference referred to as RLowLAD. First, define the first-order random difference sequence as

$$Y_{ijl} = Y_{i+j}-Y_{i+l}, \quad -k\le j,l\le k,$$

where we implicitly exclude j and l being 0 to be comparable to the LowLAD estimator. Second, define the RLowLAD estimator as

$$\hat{m}_{\text{RLowLAD}}^{(1)}(x_i) = \hat{\beta}_{i1}^{\text{RLowLAD}}, \tag{8}$$

where

$$\begin{aligned}&(\hat{\beta}_{i1}^{\text{RLowLAD}},\hat{\beta}_{i2}^{\text{RLowLAD}},\hat{\beta}_{i3}^{\text{RLowLAD}},\hat{\beta}_{i4}^{\text{RLowLAD}})\\ &= \arg\min_b \sum_{j=-k}^{k}\sum_{l=-k,l<j}^{k}\left|Y_{ijl} - b_{i1}(d_j-d_l) - b_{i2}(d_j^2-d_l^2) - b_{i3}(d_j^3-d_l^3) - b_{i4}(d_j^4-d_l^4)\right|\end{aligned}$$

with the true value $\beta_i=(\beta_{i1},\beta_{i2},\beta_{i3},\beta_{i4})^T=(m^{(1)}(x_i),m^{(2)}(x_i)/2,m^{(3)}(x_i)/6,m^{(4)}(x_i)/24)^T$ and $b=(b_{i1},b_{i2},b_{i3},b_{i4})^T$.

Theorem 2 *Under the assumptions of Theorem 1, the bias and variance of the RLowLAD estimator in (8) are, respectively,*

$$\text{Bias}[\hat{\beta}_{i1}^{\text{RLowLAD}}] = -\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4}+o\left(\frac{k^4}{n^4}\right), \quad \text{Var}[\hat{\beta}_{i1}^{\text{RLowLAD}}] = \frac{75}{24g(0)^2}\frac{n^2}{k^3}+o\left(\frac{n^2}{k^3}\right).$$

By replacing the symmetric difference with the random difference, we improve the estimation accuracy with the variance from $\frac{75}{16g(0)^2}\frac{n^2}{k^3}$ to $\frac{75}{24g(0)^2}\frac{n^2}{k^3}$. While the cost is to increase the computation complexity from $O(n)$ to $O(n^2)$.

3. Comparison with Existing First-Order Derivative Estimators

We first review some existing methods for the first-order derivative estimation which are based on either the least squares regression or the quantile regression. Both methods can be used to estimate the first-order derivative of $m(\cdot)$ due to the special structure of model (1). Note that $\mathrm{E}[Y_i|x_i]=m(x_i)+\mathrm{E}[\epsilon_i]$, and $\mathrm{Q}_\tau(Y_i|x_i)=m(x_i)+\mathrm{Q}_\tau(\epsilon_i)$ because ϵ_i is independent of x_i, where $\mathrm{Q}_\tau(\epsilon_i)$ is the τth unconditional quantile of ϵ_i. Although $\mathrm{E}[Y_i|x_i]$ may not equal to $\mathrm{Q}_\tau(Y_i|x_i)$, their derivatives must be the same as the corresponding derivatives of $m(x_i)$. To make the existing estimators comparable to our LowLAD and RLowLAD estimators, we use the uniform kernel and the same order Taylor expansion throughout this section.

3.1. LS, LowLSR and LAD Estimators

In Fan and Gijbels (1996), the first-order derivative is estimated by the least squares method:

$$(\hat{\alpha}_{i0}^{\text{LS}},\hat{\alpha}_{i1}^{\text{LS}},\hat{\alpha}_{i2}^{\text{LS}},\hat{\alpha}_{i3}^{\text{LS}},\hat{\alpha}_{i4}^{\text{LS}}) = \arg\min_\alpha \sum_{j=-k}^{k}\left(Y_{i+j}-\alpha_{i0}-\alpha_{i1}d_j-\alpha_{i2}d_j^2-\alpha_{i3}d_j^3-\alpha_{i4}d_j^4\right)^2,$$

where $\alpha = (\alpha_{i0}, \alpha_{i1}, \alpha_{i2}, \alpha_{i3}, \alpha_{i4})^T$. For ease of comparison, we exclude the point $j = 0$ so that the same Y_i's are used as in the LowLAD estimation. Define the least squares (LS) estimator as

$$\hat{m}^{(1)}_{\rm LS}(x_i) = \hat{\alpha}^{\rm LS}_{i1}. \tag{9}$$

Corollary 3 *Under the assumptions of Theorem 1, the bias and variance of the LS estimator in* (9) *are, respectively,*

$$\mathrm{Bias}[\hat{\alpha}^{\rm LS}_{i1}] = -\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4} + o\left(\frac{k^4}{n^4}\right), \quad \mathrm{Var}[\hat{\alpha}^{\rm LS}_{i1}] = \frac{75\sigma^2}{8}\frac{n^2}{k^3} + o\left(\frac{n^2}{k^3}\right).$$

To obtain the optimal convergence rate of the derivative estimation, Wang and Lin (2015) proposed the LowLSR estimator:

$$\hat{m}^{(1)}_{\rm LowLSR}(x_i) = \hat{\alpha}^{\rm LowLSR}_{i1}, \tag{10}$$

where

$$(\hat{\alpha}^{\rm LowLSR}_{i1}, \hat{\alpha}^{\rm LowLSR}_{i3}) = \arg\min_{\alpha_{i1},\alpha_{i3}} \sum_{j=1}^{k} \left(\tilde{Y}^{(1)}_{ij} - \alpha_{i1} d_j - \alpha_{i3} d_j^3\right)^2.$$

Corollary 4 *Under the assumptions of Theorem 1, the bias and variance of the LowLSR estimator in* (10) *are, respectively,*

$$\mathrm{Bias}[\hat{\alpha}^{\rm LowLSR}_{i1}] = -\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4} + o\left(\frac{k^4}{n^4}\right), \quad \mathrm{Var}[\hat{\alpha}^{\rm LowLSR}_{i1}] = \frac{75\sigma^2}{8}\frac{n^2}{k^3} + o\left(\frac{n^2}{k^3}\right).$$

Wang and Scott (1994) proposed the local polynomial least absolute deviation (LAD) estimator:

$$\hat{m}^{(1)}_{\rm LAD}(x_i) = \hat{\beta}^{\rm LAD}_{i1}, \tag{11}$$

where

$$(\hat{\beta}^{\rm LAD}_{i0}, \hat{\beta}^{\rm LAD}_{i1}, \hat{\beta}^{\rm LAD}_{i2}, \hat{\beta}^{\rm LAD}_{i3}, \hat{\beta}^{\rm LAD}_{i4}) = \arg\min_{b} \sum_{j=-k}^{k} \left|Y_{i+j} - b_{i0} - b_{i1} d_j^1 - b_{i2} d_j^2 - b_{i3} d_j^3 - b_{i4} d_j^4\right|$$

with $b = (b_{i0}, b_{i1}, b_{i2}, b_{i3}, b_{i4})^T$.

Corollary 5 *Under the assumptions of Theorem 1, the bias and variance of the LAD estimator in* (11) *are, respectively,*

$$\mathrm{Bias}[\hat{\beta}^{\rm LAD}_{i1}] = -\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4} + o\left(\frac{k^4}{n^4}\right), \quad \mathrm{Var}[\hat{\beta}^{\rm LAD}_{i1}] = \frac{75}{32 f(0)^2}\frac{n^2}{k^3} + o\left(\frac{n^2}{k^3}\right).$$

There is one key difference between the LS method and the LAD method. For the LS method, the LS estimator and the LowLSR estimator are asymptotically equivalent; while for the LAD method, the asymptotic variances of the LAD estimator and the LowLAD estimator are very different, although their asymptotic biases are the same. We provide the reasons for this key difference in Appendix F.

3.2. Quantile Regression Estimators

Quantile regression (Koenker and Bassett, 1978; Koenker, 2005) exploits the distribution information of the error term to improve the estimation efficiency. Following the composite quantile regression (CQR) in Zou and Yuan (2008), Kai et al. (2010) proposed the local polynomial CQR estimator. In general, the local polynomial CQR estimator of $m^{(1)}(x_i)$ is defined as

$$\hat{m}^{(1)}_{\rm CQR}(x_i) = \hat{\gamma}^{\rm CQR}_{i1}, \tag{12}$$

where

$$\begin{aligned}&(\left\{\hat{\gamma}^{\rm CQR}_{i0h}\right\}_{h=1}^{q}, \hat{\gamma}^{\rm CQR}_{i1}, \hat{\gamma}^{\rm CQR}_{i2}, \hat{\gamma}^{\rm CQR}_{i3}, \hat{\gamma}^{\rm CQR}_{i4})\\ &= \arg\min_{\gamma} \sum_{h=1}^{q} \left(\sum_{j=-k}^{k} \rho_{\tau_h}(Y_{i+j} - \gamma_{i0h} - \gamma_{i1} d_j^1 - \gamma_{i2} d_j^2 - \gamma_{i3} d_j^3 - \gamma_{i4} d_j^4)\right),\end{aligned}$$

with $\gamma = \left(\{\gamma_{i0h}\}_{h=1}^{q}, \gamma_{i1}, \gamma_{i2}, \gamma_{i3}, \gamma_{i4}\right)^T$, $\rho_\tau(x) = \tau x - xI(x < 0)$ is the check function, and $\tau_h = h/(q+1)$.

Corollary 6 *Under the assumptions of Theorem 1, the bias and variance of the CQR estimator in* (12) *are, respectively,*

$$\mathrm{Bias}[\hat{\gamma}^{\rm CQR}_{i1}] = -\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4} + o\left(\frac{k^4}{n^4}\right), \quad \mathrm{Var}[\hat{\gamma}^{\rm CQR}_{i1}] = \frac{75 R_1(q)}{8}\frac{n^2}{k^3} + o\left(\frac{n^2}{k^3}\right),$$

where $R_1(q) = \sum_{l=1}^{q}\sum_{l'=1}^{q} \tau_{ll'} / \{\sum_{l=1}^{q} f(c_l)\}^2$, $c_l = F^{-1}(\tau_l)$, *and* $\tau_{ll'} = \min\{\tau_l, \tau_{l'}\} - \tau_l \tau_{l'}$. *As* $q \to \infty$,

$$R_1(q) \to \frac{1}{12(E[f(\epsilon)])^2} = \frac{1}{3g(0)^2}, \quad \mathrm{Var}[\hat{\gamma}^{\rm CQR}_{i1}] = \frac{75}{24 g(0)^2}\frac{n^2}{k^3} + o\left(\frac{n^2}{k^3}\right).$$

Zhao and Xiao (2014) proposed the weighted quantile average (WQA) estimator for the regression function in nonparametric regression, an idea originated from Koenker (1984). We now extend the WQA method to estimate $m^{(1)}(x_i)$ using the local polynomial quantile regression. Specifically, we define

$$\hat{m}^{(1)}_{\rm WQA}(x_i) = \sum_{h=1}^{q} w_h \hat{\gamma}^{\rm WQA}_{i1h}, \tag{13}$$

where $\sum_{h=1}^{q} w_h = 1$, and

$$\begin{aligned}&(\hat{\gamma}^{\rm WQA}_{i0h}, \hat{\gamma}^{\rm WQA}_{i1h}, \hat{\gamma}^{\rm WQA}_{i2h}, \hat{\gamma}^{\rm WQA}_{i3h}, \hat{\gamma}^{\rm WQA}_{i4h})\\ &= \arg\min_{\gamma_h} \left(\sum_{j=-k}^{k} \rho_{\tau_h}(Y_{i+j} - \gamma_{i0h} - \gamma_{i1h} d_j^1 - \gamma_{i2h} d_j^2 - \gamma_{i3h} d_j^3 - \gamma_{i4h} d_j^4)\right)\end{aligned}$$

with $\gamma_h = (\gamma_{i0h}, \gamma_{i1h}, \gamma_{i2h}, \gamma_{i3h}, \gamma_{i4h})^T$.

Corollary 7 *Under the assumptions of Theorem 1, the bias and variance of the WQA estimator in* (13) *are, respectively,*

$$\text{Bias}[\hat{m}^{(1)}_{\text{WQA}}(x_i)] = -\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4} + +o\left(\frac{k^4}{n^4}\right), \quad \text{Var}[\hat{m}^{(1)}_{\text{WQA}}(x_i)] = \frac{75R_2(q|\mathbf{w})}{8}\frac{n^2}{k^3} + o\left(\frac{n^2}{k^3}\right),$$

where $R_2(q|\mathbf{w}) = \mathbf{w}^T\mathbf{H}\mathbf{w}$ *with* $\mathbf{w} = (w_1, \cdots, w_q)^T$, *and* $\mathbf{H} = \left\{\frac{\tau_{l,l'}}{f(c_l)f(c_{l'})}\right\}_{1\le l,l'\le q}$. *The optimal weights are given by* $\mathbf{w}^* = \frac{\mathbf{H}^{-1}e_q}{e_q^T\mathbf{H}^{-1}e_q}$, *where* $e_q = (1, \ldots, 1)^T_{q\times 1}$, *and* $R_2(q|\mathbf{w}^*) = (e_q^T\mathbf{H}^{-1}e_q)^{-1}$. *As* $q \to \infty$, *under the regularity assumptions in Theorem 6.2 of Zhao and Xiao (2014), the variance of the optimal CQR is*

$$\text{Var}[\hat{m}^{(1)}_{\text{WQA}}(x_i)] = \frac{75I(f)^{-1}}{8}\frac{n^2}{k^3} + o\left(\frac{n^2}{k^3}\right),$$

where $I(f)$ *is the Fisher information of* f.

	LS	LowLSR	LAD	CQR	WQA	LowLAD	RLowLAD
s^2	σ^2	σ^2	$\frac{1}{4f(0)^2}$	$R_1(q)$	$R_2(q)$	$\frac{1}{2g(0)^2}$	$\frac{1}{3g(0)^2}$
Asymptotic s^2	σ^2	σ^2	$\frac{1}{4f(0)^2}$	$\frac{1}{3g(0)^2}$	$I(f)^{-1}$	$\frac{1}{2g(0)^2}$	$\frac{1}{3g(0)^2}$

Table 1: s^2 in the variances $\frac{75n^2}{8k^3}s^2$ of the existing first-order derivative estimators.

For the LS, LowLSR, LAD, CQR, WQA, LowLAD and RLowLAD estimators with the same k, their asymptotic biases are all the same. In contrast, their asymptotic variances are $\frac{75n^2}{8k^3}s^2$ with s^2 being $\sigma^2, \sigma^2, \frac{1}{4f(0)^2}, R_1(q), R_2(q)$, $\frac{1}{2g(0)^2}$ and $\frac{1}{3g(0)^2}$, as listed in Table 1. From the kernel interpretation of the differenced estimator by Wang and Yu (2017), we expect an equivalence between the LS and LowLSR estimators. As $q \to \infty$, we have $R_1(q) \to 1/\{3g(0)^2\}$ and $R_2(q) \to I(f)^{-1}$, and hence the WQA estimator is the most efficient estimator as q becomes large. Nevertheless, it requires the error density function to be known in advance to carry out the most efficient WQA estimator. Otherwise, a two-step procedure is needed, where the first step estimates the error density. For a fixed q, the three quantile-based estimators (i.e., LAD, CQR, WQA) depend only on the density values of $f(\cdot)$ at finite quantile points, whose behaviors are uncertain and may not be reliable. In contrast, our new estimators rely on $g(0) = 2E[f(x)]$, which includes all information on the density $f(\cdot)$ and hence is more robust.

3.3. Relationship Among the CQR, WQA and RLowLAD Estimators

From the asymptotic variances, we can see that the RLowLAD estimator is asymptotically equivalent to the infinitely CQR estimator. Why can this happen? Intuitively, they use the same information in different ways. First, in infinitely CQR, all local data (i.e., data at all quantiles) are employed to estimate the same parameter $m^{(1)}(x_i)$, which is the same as in RLowLAD. Second, in infinitely CQR, we first use data horizontally (at fixed τ) and then combine data vertically (across τ), while in RLowLAD, we first combine data vertically since

the distribution of the error term of Y_{ijl} is $\int f\left(F^{-1}(\tau)\right)d\tau$, and then run a single regression horizontally (at $\tau = 0.5$). It is interesting (and may be also surprising) to see that these two different ways of using information have the same efficiency. The RLowLAD estimation is more powerful in practice by noticing that a single differencing in Y_{ijl} is equivalent to combining all (infinitely many) quantiles.

It should be emphasized that the ways of combining the infinitely many quantiles in CQR and WQA are different. Suppose we use the equal weights $\mathbf{w}_E = \left(\frac{1}{q}, \cdots, \frac{1}{q}\right)^T$ in WQA. Such a weighting scheme is parallel to CQR where an equal weight is imposed on each check function. Then from Theorem 2 of Kai et al. (2010), $R_2(q|\mathbf{w}_E) \to 1$ as $q \to \infty$; while $R_1(q) \to \frac{1}{3g(0)^2}$. From Table 2 of Kai et al. (2010), $\frac{1}{3g(0)^2} < 1$ for most distributions (except $N(0,1)$). Why can this difference happen? Note that

$$\left\{\hat{\gamma}^{\text{WQA}}_{ih}\right\}_{h=1}^{q} = \arg\min_{\gamma_1,\cdots,\gamma_q}\left(\sum_{h=1}^{q}\sum_{j=-k}^{k}\rho_{\tau_h}(Y_{i+j} - \gamma_{i0h} - \gamma_{i1h}d_j^1 - \gamma_{i2h}d_j^2 - \gamma_{i3h}d_j^3 - \gamma_{i4h}d_j^4)\right)$$

where $\hat{\gamma}^{\text{WQA}}_{ih} = (\hat{\gamma}^{\text{WQA}}_{i0h}, \hat{\gamma}^{\text{WQA}}_{i1h}, \hat{\gamma}^{\text{WQA}}_{i2h}, \hat{\gamma}^{\text{WQA}}_{i3h}, \hat{\gamma}^{\text{WQA}}_{i4h})^T$, so the CQR estimator is a constrained WQA estimator with the constraints being that the slopes at different quantiles must be the same. On the other hand, the WQA estimator can be interpreted as a minimum distance estimator,

$$\arg\min_{\gamma}\left(\hat{\gamma}^{\text{WQA}}_{i1} - \gamma e_q\right)^T\mathbf{W}\left(\hat{\gamma}^{\text{WQA}}_{i1} - \gamma e_q\right) = \frac{e_q^T\mathbf{W}\hat{\gamma}^{\text{WQA}}_{i1}}{e_q^T\mathbf{W}e_q} = \mathbf{w}^T\hat{\gamma}^{\text{WQA}}_{i1},$$

where $\hat{\gamma}^{\text{WQA}}_{i1} = \left(\hat{\gamma}^{\text{WQA}}_{i11}, \cdots, \hat{\gamma}^{\text{WQA}}_{i1q}\right)^T$, $\mathbf{W}$ is a symmetric weight matrix, and $\mathbf{w} = \frac{\mathbf{W}e_q}{e_q^T\mathbf{W}e_q}$. When $\mathbf{W} = \mathbf{I}_q$, the $q \times q$ identity matrix, we get the equally-weighted WQA estimator; when $\mathbf{W} = \mathbf{H}^{-1}$, we get the optimally weighted WQA estimator; and when

$$\mathbf{W} = a\mathbf{I}_q + (1-a)\mathbf{H}^{-1} \neq \mathbf{I}_q,$$

we get an estimator that is asymptotically equivalent to the CQR estimator, where

$$a = \frac{-(q-B)(1-BC) + \sqrt{(q^2-AB)(1-BC)}}{A - (2q-B)(1-BC) - q^2C} \neq 1$$

with $A = e_q^T\mathbf{H}e_q = q^2R_2(q|\mathbf{w}_E)$, $B = R_2(q|\mathbf{w}^*)^{-1}$ and $C = R_1(q)$. For example, if $\varepsilon \sim N(0,1)$, then when $q = 5$, $a = -0.367$; when $q = 9$, $a = -0.165$; when $q = 19$, $a = -0.067$; and when $q = 99$, $a = -0.011$. This is why imposing constraints directly on the objective function (i.e., the CQR estimator) or on the resulting estimators (i.e., the WQA estimator) would generate different estimators. The RLowLAD estimator and the CQR estimator have the same asymptotic variance because both of them impose constraints directly on the objective function.

3.4. Asymptotic Relative Efficiency

In this subsection, we study the Asymptotic Relative Efficiency (ARE) of the RLowLAD estimator with respect to the LowLSR and LAD estimators by comparing their asymptotic variances and AMSEs.

Since all estimators have the same bias, the comparison of their variances becomes important. We define the variance ratios of the LowLSR and LAD estimators relative to the RLowLAD estimator as

$$\mathrm{R}_{\mathrm{LowLSR}} = \frac{\mathrm{Var}(\hat{m}_{\mathrm{LowLSR}^{(1)}})}{\mathrm{Var}(\hat{m}^{(1)}_{\mathrm{RLowLAD}})} = 3\sigma^2 g(0)^2,$$
$$\mathrm{R}_{\mathrm{LAD}} = \frac{\mathrm{Var}(\hat{m}^{(1)}_{\mathrm{LAD}})}{\mathrm{Var}(\hat{m}^{(1)}_{\mathrm{RLowLAD}})} = \frac{3g(0)^2}{4f(0)^2}.$$

In addition, the overall performance of an estimator is usually measured by its AMSE, so we define the AREs based on AMSE as

$$\mathrm{ARE}_{\mathrm{LowLSR}} = \frac{\mathrm{AMSE}(\hat{m}^{(1)}_{\mathrm{LowLSR}})}{\mathrm{AMSE}(\hat{m}^{(1)}_{\mathrm{RLowLAD}})},$$
$$\mathrm{ARE}_{\mathrm{LAD}} = \frac{\mathrm{AMSE}(\hat{m}^{(1)}_{\mathrm{LAD}})}{\mathrm{AMSE}(\hat{m}^{(1)}_{\mathrm{RLowLAD}})}.$$

The LowLSR estimator has the AMSE

$$\mathrm{AMSE}(\hat{m}^{(1)}_{\mathrm{LowLSR}}) = \left\{\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4}\right\}^2 + \frac{75\sigma^2}{8}\frac{n^2}{k^3},$$

and the optimal k minimizing the AMSE is

$$k^{opt}_{\mathrm{LowLSR}} = \left\{\frac{893025\sigma^2}{(m^{(5)}(x_i))^2}\right\}^{1/11} n^{10/11}$$

Similarly, we have

$$k^{opt}_{\mathrm{RLowLAD}} = \left\{\frac{893025}{3g(0)^2(m^{(5)}(x_i))^2}\right\}^{1/11} n^{10/11} = R^{-1/11}_{\mathrm{LowLSR}} k^{opt}_{\mathrm{LowLSR}}.$$

As $n \to \infty$, we can show

$$\mathrm{ARE}_{\mathrm{LowLSR}} = {\mathrm{R}_{\mathrm{LowLSR}}}^{8/11},$$
$$\mathrm{ARE}_{\mathrm{LAD}} = {\mathrm{R}_{\mathrm{LAD}}}^{8/11}.$$

Since the variance ratio and ARE have a close relationship, we only report the variance ratios. We consider eight distributions for the random errors: the normal distribution $N(0, 1^2)$, the Laplace (double exponential) distribution, the Logistic distribution, the t distribution with 3 degrees of freedom, the mixed normal distribution $0.9N(0, 1^2)+0.1N(0, 3^2)$ and $0.9N(0, 1^2)+0.1N(0, 10^2)$, the Cauchy distribution, the mixed double gamma distribution $0.9Gamma(0, 1) + 0.1Gamma(1, 1)$ and $0.9Gamma(0, 1) + 0.1Gamma(3, 1)$, and the bimodal distribution $0.5N(-1, 1) + 0.5N(1, 1)$ and $0.5N(-3, 1) + 0.5N(3, 1)$, which were adopted in the robust location estimation by Koenker and Bassett (1978) and the variable selection by Zou and Yuan (2008).

	$f(0)$	$g(0)$	σ^2	$\mathrm{R}_{\mathrm{LowLSR}}$	$\mathrm{R}_{\mathrm{LAD}}$
$N(0,1)$	0.40	0.56	1	0.95	1.5
$0.9N(0,1)+0.1N(0,3^2)$	0.37	0.50	1.8	1.37	1.38
$0.9N(0,1)+0.1N(0,10^2)$	0.36	0.47	10.9	7.28	1.27
$t(3)$	0.37	0.46	3	1.90	1.17
Laplace	0.50	0.50	2	1.50	0.75
Logistic	0.25	0.33	3.29	1.10	1.33
Cauchy	0.32	0.32	∞	∞	0.75
$0.9Gamma(0,1)+0.1Gamma(1,1)$	0.50	0.45	1.1	1.63	0.68
$0.9Gamma(0,1)+0.1Gamma(3,1)$	0.46	0.42	1.3	2.44	0.68
$0.5N(-1,1)+0.5N(1,1)$	0.15	0.39	2	0.89	5.18
$0.5N(-3,1)+0.5N(3,1)$	4.92×10^{-5}	0.28	10	2.39	2.46×10^{7}

Table 2: Variance ratios for different error distributions.

Table 2 lists the variance ratios for these eight error distributions which are derived in Appendix E. From Table 2, a few interesting results can be drawn. First of all, the variance of the RLowLAD estimator is usually smaller and more robust than that of the LowLSR and LAD estimators in most cases. Secondly, the LAD estimator is not robust since it relies on the density value at one point 0, so when facing the bimodal errors the variance is huge. Thirdly, the minimum value of $\mathrm{R}_{\mathrm{LowLSR}}$ in Table 2 is 0.89. Actually, there is an exact lower bound for $\mathrm{R}_{\mathrm{LowLSR}}$ as stated in Theorem 4 of Kai et al. (2010). For completeness, we repeat their Theorem 4 in the following Lemma 8.

Lemma 8 *Let $\mathcal{F}$ denote the class of error distributions with mean 0 and variance 1. Then*

$$\inf_{f\in\mathcal{F}} \mathrm{R}_{\mathrm{LowLSR}}(f) \approx 0.86.$$

The lower bound is reached if and only if the error follows the rescaled beta$(2, 2)$ distribution. Thus,

$$\inf_{f\in\mathcal{F}} \mathrm{ARE}_{\mathrm{LowLSR}}(f) = \mathrm{R}^{8/11}_{\mathrm{LowLSR}} \approx 0.90.$$

In other words, the potential efficiency loss of the RLowLAD estimator relative to the LowLSR estimator is at most 10%.

In Appendix E, we further illustrate the trade-off between the sharp-peak and heavy-tailed errors using three error distributions.

4. Derivative Estimation at Boundaries

At the left boundary with $2 \le i \le k$, the bias and variance of the LowLAD estimator are $-m^{(5)}(x_i)(i-1)^4/(504n^4)$ and $75n^2/(16g(0)^2(i-1)^3)$, respectively. At the endpoint $i = 1$, the LowLAD estimator is not well defined. Similar results hold for the estimation at the right boundary with $n-k+1 \le i \le n-1$. In this section, we propose an asymmetrical LowLAD method (As-LowLAD) to reduce the estimation variance as well as to improve the finite-sample performance at the boundaries.

Assume that $m(\cdot)$ is twice continuously differentiable on $[0,1]$ and Median$[\epsilon_i]=0$. For $1\le i\le k$, we define the asymmetric lag-j first-order difference quotients as

$$Y_{ij}^{<1>}=\frac{Y_{i+j}-Y_i}{x_{i+j}-x_i},\quad -(i-1)\le j\le k,\quad j\neq 0.$$

By decomposing $Y_{ij}^{<1>}$ into two parts, we have

$$\begin{aligned}Y_{ij}^{<1>}&=\frac{m(x_{i+j})-m(x_i)}{j/n}+\frac{\epsilon_{i+j}-\epsilon_i}{j/n}\\&=m^{(1)}(x_i)+(-\epsilon_i)d_j^{-1}+\frac{\epsilon_{i+j}}{j/n}+\frac{m^{(2)}(x_i)}{2!}\frac{j}{n}+o\left(\frac{j}{n}\right),\end{aligned}$$

where ϵ_i is fixed as j changes. Thus, Median$(Y_{ij}^{<1>}|\epsilon_i)=m^{(1)}(x_i)+(-\epsilon_i)d_j^{-1}+\frac{m^{(2)}(x_i)}{2!}\frac{j}{n}$. Ignoring the last term, we can rewrite the above model as

$$Y_{ij}^{<1>}=\beta_{i1}+\beta_{i0}d_j^{-1}+\delta_{ij}+o(1),\quad -(i-1)\le j\le k,\quad j\neq 0,$$

where $(\beta_{i0},\beta_{i1})^T=(-\epsilon_i,m^{(1)}(x_i))^T$, $\delta_{ij}=\frac{\epsilon_{i+j}}{j/n}$. By the LowLAD method, the regression coefficients can be estimated as

$$\begin{aligned}\left(\hat\beta_{i0},\hat\beta_{i1}\right)&=\arg\min_b\sum_{j=-(i-1)}^{k}\left|Y_{ij}^{<1>}-b_{i1}-b_{i0}d_j^{-1}\right|w_j\\&=\arg\min_b\sum_{j=-(i-1)}^{k}\left|\tilde Y_{ij}^{<1>}-b_{i0}-b_{i1}d_j\right|,\end{aligned}\tag{14}$$

where $w_j=d_j$, $b=(b_{i0},b_{i1})^T$, and $\tilde Y_{ij}^{<1>}=Y_{i+j}-Y_i$. The As-LowLAD estimator of $m^{(1)}(x_i)$ is $\hat\beta_{i1}$.

Similarly to Theorem 1, we can prove the asymptotic normality for $\hat\beta_{i1}$. The following theorem states its asymptotic bias and variance.

Theorem 9 *Assume that ϵ_i are iid random errors with median 0 and a continuous, positive density $f(\cdot)$ in a neighborhood of zero. Furthermore, assume that $m(\cdot)$ is twice continuously differentiable on $[0,1]$. Then for each $1\le i\le k$, the leading terms of the bias and variance of $\hat\beta_{i1}$ in (14) are, respectively,*

$$\begin{aligned}\mathrm{Bias}[\hat\beta_{i1}]&=\frac{m^{(2)}(x_i)}{2}\frac{k^4+2k^3i-2ki^3-i^4}{n(k^3+3k^2i+3ki^2+i^3)},\\\mathrm{Var}[\hat\beta_{i1}]&=\frac{3}{f(0)^2}\frac{n^2}{k^3+3k^2i+3ki^2+i^3}.\end{aligned}$$

For the estimation at the boundaries, Wang and Lin (2015) proposed a one-side LowLSR (OS-LowLSR) estimator of the first-order derivative, with the bias and variance being $m^{(2)}(x_i)k/(2n)$ and $12\sigma^2n^2/k^3$, respectively. In contrast, if we consider the one-side LowLAD (OS-LowLAD) estimator of the first-order derivative, its bias and variance are $m^{(2)}(x_i)k/(2n)$ and $3n^2/(f(0)^2k^3)$, respectively. Note that the two biases are the same, while the variances are different with the former related to σ^2 and the latter related to $f(0)$. Note also that different from the LowLAD estimator at the interior point, the variance of the OS-LowLAD estimator involves $f(0)$ rather than $g(0)$.

Theorem 9 shows that our estimator has a smaller bias than the OS-LowLSR and OS-LowLAD estimators. In the special case $i=k$, the bias disappears (in fact it reduces to the higher-order term $O(k^2/n^2)$). With normal errors, when $1<i<\lfloor 0.163k\rfloor$, the order of variances of the three estimators is $\mathrm{Var}(\hat m^{(1)}_{\mathrm{OS-LAD}}(x_i))>\mathrm{Var}(\hat m^{(1)}_{\mathrm{AS-LAD}}(x_i))>\mathrm{Var}(\hat m^{(1)}_{\mathrm{OS-LSR}}(x_i))$, where for a real number x, $\lfloor x\rfloor$ means the largest integer less than x; when $\lfloor 0.163k\rfloor<i<k$, the order of variances of the three estimators is $\mathrm{Var}(\hat m^{(1)}_{\mathrm{OS-LSR}}(x_i))>\mathrm{Var}(\hat m^{(1)}_{\mathrm{OS-LAD}}(x_i))>\mathrm{Var}(\hat m^{(1)}_{\mathrm{AS-LAD}}(x_i))$. As i approaches k, the variance of the As-LowLAD estimator is reduced to one-eighth of the variance of the OS-LowLAD estimator, and is much smaller than the variance of the OS-LowLSR estimator.

Up to now, we have the first-order derivative estimators $\{\hat m^{(1)}(x_i)\}_{i=1}^n$ on the discrete points $\{x_i\}_{i=1}^n$. To estimate the first-order derivative function, we suggest two strategies for different noise levels of the derivative data: the cubic spline interpolation in Knott (2000) for 'good' derivative estimators, and the local polynomial regression in Brown and Levine (2007) and De Brabanter et al. (2013) for 'bad' derivative estimators. Here, the terms 'good' and 'bad' indicate small and large estimation variances of the derivative estimators, respectively.

5. Second- and Higher-Order Derivative Estimation

In this section, we generalize our robust method for the first-order derivative estimation to the second- and higher-order derivatives estimation.

5.1. Second-Order Derivative Estimation

Define the second-order difference quotients as

$$Y_{ij}^{(2)}=\frac{Y_{i-j}-2Y_i+Y_{i+j}}{j^2/n^2},\quad 1\le j\le k,\tag{15}$$

and assume that $m(\cdot)$ is six times continuously differentiable. Then we can decompose (15) into two parts and simplify it by the Taylor expansion as

$$\begin{aligned}Y_{ij}^{(2)}&=\frac{m(x_{i-j})-2m(x_i)+m(x_{i+j})}{j^2/n^2}+\frac{\epsilon_{i-j}-2\epsilon_i+\epsilon_{i+j}}{j^2/n^2}\\&=m^{(2)}(x_i)+\frac{m^{(4)}(x_i)}{12}\frac{j^2}{n^2}+\frac{m^{(6)}(x_i)}{360}\frac{j^4}{n^4}+o\left(\frac{j^4}{n^4}\right)+\frac{\epsilon_{i-j}-2\epsilon_i+\epsilon_{i+j}}{j^2/n^2}.\end{aligned}$$

Since i is fixed as j varies, the conditional expectation of $Y_{ij}^{(2)}$ given ϵ_i is

$$\mathrm{E}[Y_{ij}^{(2)}|\epsilon_i]=m^{(2)}(x_i)+\frac{m^{(4)}(x_i)}{12}\frac{j^2}{n^2}+\frac{m^{(6)}(x_i)}{360}\frac{j^4}{n^4}+o\left(\frac{j^4}{n^4}\right)+(-2\epsilon_i)\frac{n^2}{j^2}.$$

This results in the true regression model as

$$Y_{ij}^{(2)} = \alpha_{i2} + \alpha_{i4} d_j^2 + \alpha_{i6} d_j^4 + o\left(d_j^4\right) + \alpha_{i0} d_j^{-2} + \delta_{ij}, \quad 1 \le j \le k,$$

where $\alpha_i = (\alpha_{i0}, \alpha_{i2}, \alpha_{i4}, \alpha_{i6})^T = (-2\epsilon_i, m^{(2)}(x_i), m^{(4)}(x_i)/12, m^{(6)}(x_i)/360)^T$, and the errors $\delta_{ij} = \frac{\epsilon_{i+j} + \epsilon_{i-j}}{j^2/n^2}$. If ϵ_i has a symmetric density about zero, then $\tilde{\delta}_{ij} = \frac{j^2}{n^2}\delta_{ij} = \epsilon_{i+j} + \epsilon_{i-j}$ has median 0 (see Appendix D). Following the similar procedure as in the first-order derivative estimation, our LowLAD estimator of $m^{(2)}(x_i)$ is defined as

$$\hat{m}^{(2)}_{\text{LowLAD}}(x_i) = \hat{\alpha}_{i2}, \tag{16}$$

where

$$\begin{aligned}(\hat{\alpha}_{i0}, \hat{\alpha}_{i2}, \hat{\alpha}_{i4})^T &= \arg\min_a \sum_{j=1}^{k} \left|Y_{ij}^{(2)} - (a_{i2} + a_{i4} d_j^2 + a_{i0} d_j^{-2})\right| W_j \\ &= \arg\min_a \sum_{j=1}^{k} \left|\tilde{Y}_{ij}^{(2)} - (a_{i0} + a_{i2} d_j^2 + a_{i4} d_j^4)\right|\end{aligned}$$

with $a = (a_{i0}, a_{i2}, a_{i4})^T$, $W_j = d_j^2$, and $\tilde{Y}_{ij}^{(2)} = Y_{i-j} - 2Y_i + Y_{i+j}$. The following theorem shows that $\hat{m}^{(2)}_{\text{LowLAD}}(x_i)$ behaves similarly as $\hat{m}^{(1)}_{\text{LowLAD}}(x_i)$.

Theorem 10 *Assume that ϵ_i are iid random errors whose density $f(\cdot)$ is continuous and symmetric about zero. Then as $k \to \infty$ and $k/n \to 0$, $\hat{\alpha}_{i2}$ in (16) is asymptotically normally distributed with*

$$\text{Bias}[\hat{\alpha}_{i2}] = -\frac{m^{(6)}(x_i)}{792}\frac{k^4}{n^4} + o\left(\frac{k^4}{n^4}\right), \quad \text{Var}[\hat{\alpha}_{i2}] = \frac{2205}{16h(0)^2}\frac{n^4}{k^5} + o\left(\frac{n^4}{k^5}\right),$$

where $h(0) = \int_{-\infty}^{\infty} f^2(x)\mathrm{d}x$. The optimal k that minimizes the AMSE is

$$k_{opt} \approx 3.93 \left(\frac{1}{h(0)^2 m^{(6)}(x_i)^2}\right)^{1/13} n^{12/13},$$

and, consequently, the minimum AMSE is

$$\text{AMSE}[\hat{\alpha}_{i2}] \approx 0.24 \left(\frac{m^{(6)}(x_i)^{10}}{h(0)^{16}}\right)^{1/13} n^{-8/13}.$$

5.2. Higher-Order Derivative Estimation

We now propose a robust method for estimating the higher-order derivatives $m^{(l)}(x_i)$ with $l > 2$ via a two-step procedure. In the first step, we construct a sequence of symmetric difference quotients in which the higher-order derivative is the intercept of the linear regression derived by the Taylor expansion, and in the second step, we estimate the higher-order derivative using the LowLAD method.

When l is odd, let $d = (l+1)/2$. We linearly combine $m(x_{i\pm j})$ subject to

$$\sum_{h=1}^{d}[a_{jd+h} m(x_{i+jd+h}) + a_{-(jd+h)} m(x_{i-(jd+h)})] = m^{(l)}(x_i) + O\left(\frac{j}{n}\right), \quad 0 \le j \le k,$$

where k is a positive integer. We can derive a total of $2d$ equations through the Taylor expansion to solve out the $2d$ unknown parameters. Define

$$Y_{ij}^{(l)} = \sum_{h=1}^{d}[a_{jd+h} Y_{i+jd+h} + a_{-(jd+h)} Y_{i-(jd+h)}],$$

and consider the linear regression

$$Y_{ij}^{(l)} = m^{(l)}(x_i) + \delta_{ij}, \quad 0 \le j \le k,$$

where $\delta_{ij} = \sum_{h=1}^{d}[a_{i,jd+h}\epsilon_{i+jd+h} + a_{i,-(jd+h)}\epsilon_{i-(jd+h)}] + O(j/n)$.

When l is even, let $d = l/2$. We linearly combine $m(x_{i\pm j})$ subject to

$$b_j m(x_i) + \sum_{h=1}^{d}[a_{jd+h} m(x_{i+jd+h}) + a_{-(jd+h)} m(x_{i-(jd+h)})] = m^{(l)}(x_i) + O\left(\frac{j}{n}\right), \quad 0 \le j \le k,$$

where k is a positive integer. We can derive a total of $2d+1$ equations through the Taylor expansion to solve out the $2d+1$ unknown parameters. Define

$$Y_{ij}^{(l)} = b_j Y_i + \sum_{h=1}^{d}[a_{jd+h} Y_{i+jd+h} + a_{-(jd+h)} Y_{i-(jd+h)}],$$

and consider the linear regression

$$Y_{ij}^{(l)} = m^{(l)}(x_i) + b_j\epsilon_i + \delta_{ij}, \quad 0 \le j \le k,$$

where $\delta_{ij} = \sum_{h=1}^{d}[a_{i,jd+h}\epsilon_{i+jd+h} + a_{i,-(jd+h)}\epsilon_{i-(jd+h)}] + O(j/n)$.

When k is large, we suggest to keep the j^2/n^2 term as in (6) to reduce the estimation bias. If $\sum_{h=1}^{d}[a_{i,jd+h}\epsilon_{i+jd+h} + a_{i,-(jd+h)}\epsilon_{i-(jd+h)}]$ has median zero, then we can obtain the higher-order derivative estimators by the LowLAD method and deduce their asymptotic properties by similar arguments as in the previous sections. To save space, we omit the technical details in this paper.

6. Simulation Studies and Empirical Application

In this section, we conduct simulations to evaluate the finite-sample performance of our first- and second-order derivative estimators and compare them with some existing estimators. We also apply our method to a real data set to illustrate its usefulness in practice.

6.1. First-Order Derivative Estimators

We first consider the following three regression functions:

$$m_0(x) = (x) = \sqrt{x(1-x)}\sin((2.1\pi)/(x+0.05)), \quad x \in [0.25, 1],$$
$$m_1(x) = \sin(2\pi x) + \cos(2\pi x) + \log(4/3 + x), \quad x \in [-1, 1],$$
$$m_2(x) = 32e^{-8(1-2x)^2}(1-2x), \quad x \in [0, 1].$$

These three functions were also considered in Hall (2010) and De Brabanter et al. (2013).

For normal errors, we consider the function $m_0(x)$ and compare the LowLAD, RLowLAD and LowLSR estimators. The data set of size 300 is generated from model (1) with errors $\epsilon_i \overset{iid}{\sim} N(0, 0.1^2)$ and is plotted in Figure 1(a). Figure 1 (2) displays the LowLAD (RLowLAD) estimator (use the R package 'L1pack' in Osorio (2015)) and the LowLSR estimator with $k \in \{6, 12, 25, 30, 50\}$. When k is small (see Figure 1(b) and 1(c)), both estimators are noise-corrupted versions of the true first-order derivatives; as k becomes larger (see Figure 1(d)-(f)), our estimator provides a similar performance as the LowLSR estimator. Furthermore, by combining the left part of Figure 1(d), the middle part of 1(e) and the right part of 1(f), more accurate derivative estimators can be obtained for practical use.

In addition, note that the three estimators have the same variation trend, whereas the LowLAD estimator has a slightly large oscillation and the RLowLAD estimator has a similar performance compared to the LowLSR estimator. These simulation results coincide with the theoretical results: the three estimators have the same bias, which explains the same variation trend; the variance ratios are $\mathrm{Var}(\hat{m}^{(1)}_{\mathrm{LowLSR}})/\mathrm{Var}(\hat{m}^{(1)}_{\mathrm{LowLAD}}) \approx 0.64$ and $\mathrm{Var}(\hat{m}^{(1)}_{\mathrm{LowLSR}})/\mathrm{Var}(\hat{m}^{(1)}_{\mathrm{RLowLAD}}) \approx 0.95$, which explains the oscillation performance.

Next, we consider the non-normal errors: 90% of the errors come from $\epsilon \sim N(0, \sigma^2)$ with $\sigma = 0.1$, and the remaining 10% come from $\epsilon \sim N(0, \sigma_0^2)$ with $\sigma_0 = 1$ or 10 corresponding to the low or high contamination level. Figures 3 and 4 present the finite-sample performance of the first-order derivative estimators for the regression functions m_1 and m_2, respectively. They show that the estimated curves of the first-order derivative based on LowLAD fit the true curves more accurately than LowLSR in the presence of heavy-tailed errors. The heavier the tail, the more significant the improvement.

We also compute the mean absolute errors to further assess the performance of the four methods, i.e., LowLAD, RLowLAD, LowLSR and LAD. Since the oscillation of a periodic function depends on its frequency and amplitude, we consider the sine function in the following form as the regression function,

$$m_3(x) = A\sin(2\pi f x), \quad x \in [0, 1].$$

The errors are generated in the above contaminated way. We consider two sample sizes: $n = 100$ and 500, two standard deviations: $\sigma = 0.1$ and 0.5, two contaminated standard deviations: $\sigma_0 = 1$ and 10, two frequencies: $f = 0.5$ and 1, and two amplitudes: $A = 1$ and 10.

We use the following adjusted mean absolute error (AMAE) as the criterion of performance evalution:

$$\mathrm{AMAE}(k) = \frac{1}{n-2k}\sum_{i=k+1}^{n-k}\left|\hat{m}^{(1)}(x_i) - m^{(1)}(x_i)\right|.$$

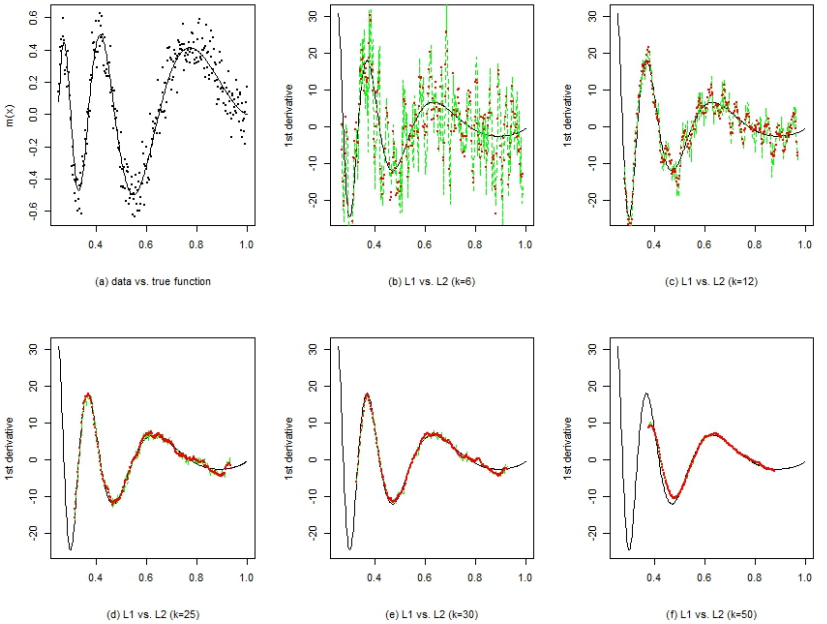


Figure 1: The comparison between the LowLAD and LowLSR estimators. (a) Simulated data set of size 300 from model (1) with equidistant $x_i \in [0.25, 1]$, $\epsilon_i \overset{iid}{\sim} N(0, 0.1^2)$, and the true regression function $m_0(x)$ (bold line). (b)-(f) The first-order LowLAD derivative estimators (green points) and the first-order LowLSR derivative estimators (red dashed line) for $k \in \{6, 9, 12, 25, 30, 50\}$. As a reference, the true first-order derivative fucntion is also plotted (bold line).

Due to the heavy computation (for example, it needs more than 48 hours for the case $n = 500$ and $k = n/5 = 100$ based on 1000 repetitions on our personal computer), we choose $k = n/5$ uniformly.

Table 3 reports the simulation results based on 1000 repetitions. The numbers outside and inside the parentheses represent the mean and standard deviation of the AMAE, respectively. It is evident that RLowLAD performs uniformly better than LowLAD and performs the best for most of cases. In particular for the cases with $\sigma_0 = 2\sigma$, the contamination is very light and thus LowLSR is better than LowLAD; while for the cases with $\sigma_0 = 10\sigma$, the contamination is very heavy and thus LowLSR is worse than LowLAD. These simulation results coincide with the theoretical results.

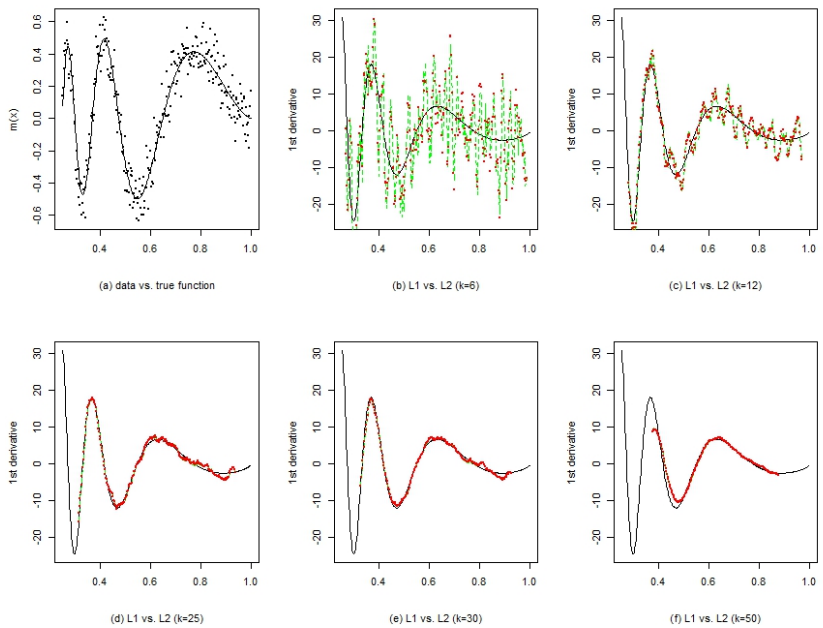


Figure 2: The comparison between the RLowLAD and LowLSR estimators for the same data set as in Figure 1

6.2. Second-Order Derivative Estimators

To assess the finite-sample performance of the second-order derivative estimators, we consider the same regression functions as in Section 6.1. Figures 5 and 6 present the estimated curves of the second-order derivatives of m_1 and m_2, respectively. It shows that our LowLAD estimator fits the true curves more accurately than the LowLSR estimator in all settings.

We further compare LowLAD with two other well-known methods: the local polynomial regression with $p = 5$ (use R package 'locpol' in Cabrera (2012)) and the penalized smoothing splines with $norder = 6$ and $method = 4$ (use R package 'pspline' in Ramsay and Ripley (2013)). For simplicity, we consider the simple version of m_3 with $A = 5$ and $f = 1$:

$$m_4(x) = 5\sin(2\pi x), \quad x \in [0, 1].$$

We let $n = 500$, and generate the errors in the same way as in Section 6.1. With 1000 repetitions, the simulation results are reported in Figures 7 and 8 which indicate that our

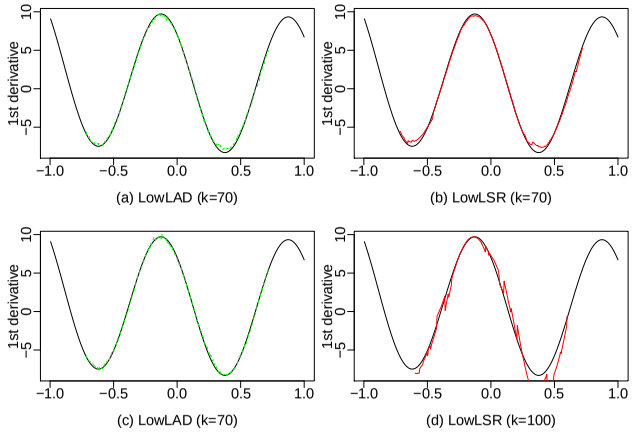


Figure 3: (a-b) The true first-order derivative function (bold line), LowLAD (green line) and LowLSR estimators (red line). Model (1) with equidistant $x_i \in [-1, 1]$, regression function m_1, and $\epsilon \sim 90\%N(0, 0.1^2) + 10\%N(0, 1^2)$. (c-d) The same designs as in (a-b) except $\epsilon \sim 90\%N(0, 0.1^2) + 10\%N(0, 10^2)$.

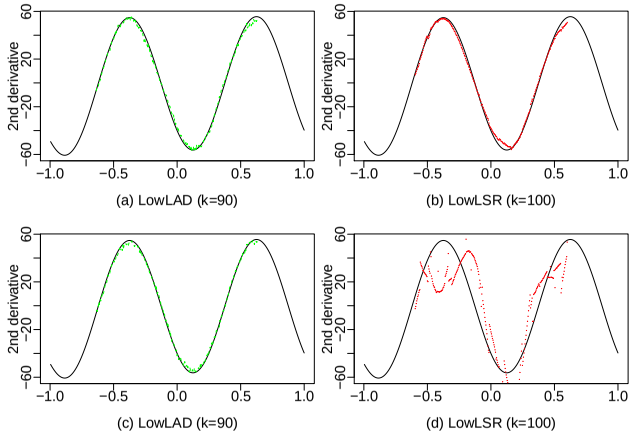


Figure 4: (a-d) The true first-order derivative function (bold line), LowLAD (green line) and LowLSR estimators (red line). The same designs as in Figure 3 except the regression function being m_2.

robust estimator is superior to the existing methods in the presence of sharp-peak and heavy-tailed errors.

6.3. House Price of China in Latest Ten Years

In reality, there are many data sets recorded by year, month, week, day, hour, minute, etc. For example, human growth is usually recorded by year, and temperature is recorded by hour, day or month. In this section, we apply RLowLAD to the data set of house price in two cities of China, i.e., Beijing and Jinan. We collect these monthly data from the web: http://www.creprice.cn/ (see Figure 9), which last from January 2008 to July 2018 and have size 127. We analyze this date set in two steps. Firstly, we apply our method to estimate the first-order derivative with $k = 8$ for RLowLAD and $k = 6$ for lower-order RLowLAD, where the lower-order means that we conduct the Taylor expansion to order 2 instead of order 4. Secondly, we define the relative growth rate as the ratio between the RLowLAD estimator and the house price at the same month, and then plot the relative growth rates in Figures 10 and 11. In the last ten years, the house price goes through tricycle fast increasing, and the monthly growth rate is larger than 0 most of the time with the maximum value at about 0.05.

7. Conclusion and Extensions

In this paper, we propose a robust differenced method for estimating the first- and higher-order derivatives of the regression function in nonparametric models. The new method consists of two main steps: first construct a sequence of symmetric difference quotients, and second estimate the derivatives using the LowLAD regression. The main contributions are as follows:

(1) Unlike LAD, our proposed LowLAD has the unique property of *double robustness* (or *robustness*[2]. Specifically, it is robust not only to heavy-tailed error distributions (like LAD), but also to low density of the error term at a specific quantile (LAD needs a high value of the error density at median; otherwise, the relative efficiency of LAD can be arbitrarily small compared with LowLAD). Following Theorem 1, the asymptotic variance of the LowLAD estimator includes the term $g(0) = 2\int_{-\infty}^{\infty} f^2(x)dx = 2\int_{-\infty}^{\infty} f(F^{-1}(\tau))d\tau$, which implies that we are able to utilize the information of the whole error density. While for the LAD estimator, its variance depends on a single value $f(0)$ only. In this sense, the LowLAD estimator is more robust than the LAD estimator.

(2) Our proposed LowLAD does not require the error distribution to have a zero median, and so is more flexible than LAD. To be more specific, our symmetric differenced errors are guaranteed to have a zero median and a positive symmetric density in a neighborhood of zero, regardless of whether or not the distribution of the original error is symmetric. While for LAD, we must require the error distribution to have a zero median, and consequently, the practical usefulness of LAD will be rather limited.

(3) More surprisingly, as an extension of LowLAD, our proposed RLowLAD based on random difference is asymptotically equivalent to the infinitely composite quantile regression (CQR) estimator. *In other words, running one RLowLAD regression is equivalent to combining infinitely many quantile regressions.*

(4) Lastly, it is also worthwhile to mention that the differences between LowLAD and LAD are strikingly distinct from the differences between LowLSR and LS. For the same data and the same tuning parameter k, we have LS = LowLSR, whereas LAD $\neq$ LowLAD. What is more, RLowLAD is able to further improve the estimation efficiency compared with LowLAD, while RLowLSR, the LS counterpart of RLowLAD, is not able to improve efficiency relative to LowLSR.

LowLAD is a new idea to explore the information of density function by combining first-order difference and LAD. We can adopt the third-order symmetric difference $\{(Y_{i+j} - Y_{i-j}) + (Y_{i+l} - Y_{i-l})\}$ or the third-order random difference $\{(Y_{i+j} + Y_{i+l}) - (Y_{i+u} + Y_{i+v})\}$, even higher-order difference, to explore the information of density function. Whether and how to achieve the Cramer-Rao Lower bound deserves further study. These questions would be investigated in a separate paper.

In this paper, we focus on the derivative estimation with fixed designs and iid errors. With minor technical extensions, the proposed method can be extended to random designs with heteroskedastic errors. Further extensions to linear model, high-dimensional model for variable selection, semiparametric model, and change-point detection are also possible.

Acknowledgements

We would like to thank two anonymous reviewers and action editor for their constructive comments on improving the quality of the paper. Wang's work was supported by Qufu Normal University, University of Hong Kong, Hong Kong Baptist University. Lin's work was supported by NNSF projects of China.

Appendix A. Proof of Theorem 1

Proposition 11 *If ϵ_i are iid with a continuous, positive density $f(\cdot)$ in a neighborhood of the median, then $\tilde{\zeta}_{ij} = (\epsilon_{i+j} - \epsilon_{i-j})/2$ (j=1,..., k) are iid with median 0 and a continuous, positive, symmetric density $g(\cdot)$, where*

$$g(x) = 2\int_{-\infty}^{\infty} f(2x+\epsilon)f(\epsilon)\mathrm{d}\epsilon.$$

Proof of Proposition 11 The distribution of $\tilde{\zeta}_{ij} = (\epsilon_{i+j} - \epsilon_{i-j})/2$ is

$$\begin{aligned} F_{\tilde{\zeta}_{ij}}(x) &= P((\epsilon_{i+j} - \epsilon_{i-j})/2 \leq x) \\ &= \iint_{\epsilon_{i+j} \leq 2x+\epsilon_{i-j}} f(\epsilon_{i+j})f(\epsilon_{i-j})\mathrm{d}\epsilon_{i+j}\mathrm{d}\epsilon_{i-j} \\ &= \int_{-\infty}^{\infty}\{\int_{-\infty}^{2x+\epsilon_{i-j}} f(\epsilon_{i+j})\mathrm{d}\epsilon_{i+j}\}f(\epsilon_{i-j})\mathrm{d}\epsilon_{i-j} \\ &= \int_{-\infty}^{\infty} F(2x+\epsilon_{i-j})f(\epsilon_{i-j})\mathrm{d}\epsilon_{i-j}. \end{aligned}$$

Then the density of $\tilde{\zeta}_{ij}$ is

$$g(x) \triangleq \frac{\mathrm{d}F_{\tilde{\zeta}_{ij}}(x)}{\mathrm{d}x} = 2\int_{-\infty}^{\infty} f(2x+\epsilon_{i-j})f(\epsilon_{i-j})\mathrm{d}\epsilon_{i-j}.$$

The density $g(\cdot)$ is symmetric due to

$$\begin{aligned} g(-x) &= 2\int_{-\infty}^{\infty} f(-2x+\epsilon_{i-j})f(\epsilon_{i-j})\mathrm{d}\epsilon_{i-j} \\ &= 2\int_{-\infty}^{\infty} f(\epsilon_{i-j})f(\epsilon_{i-j}+2x)\mathrm{d}\epsilon_{i-j} \\ &= g(x). \end{aligned}$$

Therefore, we have

$$\begin{aligned} F_{\tilde{\zeta}_{ij}}(0) &= \int_{-\infty}^{\infty} F(\epsilon_{i-j})f(\epsilon_{i-j})\mathrm{d}\epsilon_{i-j} = \frac{1}{2}F^2(\epsilon_{i-j})\,|_{-\infty}^{\infty} = \frac{1}{2}, \\ g(0) &= 2\int_{-\infty}^{\infty} f^2(\epsilon_{i-j})\mathrm{d}\epsilon_{i-j}. \end{aligned}$$

■

Proof of Theorem 1 Rewrite the objective function as

$$S_n(b) = \frac{1}{n}\sum_j f_n\left(\tilde{Y}_{ij}|b\right),$$

where

$$f_n\left(\tilde{Y}_{ij}|b\right) = \left|\tilde{Y}_{ij}^{(1)} - b_{i1}d_j - b_{i3}d_j^3\right|\frac{1}{h}1(0 < d_j \leq h)$$

with $b = (b_{i1}, b_{i3})^T$ and $h = k/n$. Define $X_j = \left(d_j, d_j^3\right)^T$ and $H = \mathrm{diag}\{h, h^3\}$. Note that $\arg\min_b S_n(b) = \arg\min_b [S_n(b) - S_n(\beta)]$, where $\beta = \left(m^{(1)}(x_i), m^{(3)}(x_i)/6\right)^T$.

We first show that $H\left(\widehat{\beta} - \beta\right) = o_p(1)$, where $\widehat{\beta} = (\hat{\beta}_{i1}, \hat{\beta}_{i3})^T$. We use Lemma 4 of Porter and Yu (2015) to show the consistency. Essentially, we need to show that

(i) $\sup_{b\in\mathcal{B}} |S_n(b) - S_n(\beta) - \mathrm{E}[S_n(b) - S_n(\beta)]| \xrightarrow{p} 0$,

(ii) $\inf_{\|H(b-\beta)\|>\delta} \mathrm{E}[S_n(b) - S_n(\beta)] > \varepsilon$ for n large enough, where $\mathcal{B}$ is a compact parameter space for β, and δ and ε are fixed positive small numbers.

We use Lemma 2.8 of Pakes and Pollard (1989) to show (i), where

$$\mathcal{F}_n = \left\{f_n\left(\tilde{Y}|b\right) - f_n\left(\tilde{Y}|\beta\right) : b \in \mathcal{B}\right\}.$$

Note that $\mathcal{F}_n$ is Euclidean (see, e.g., Definition 2.7 of Pakes and Pollard (1989) for the definiton of an Euclidean-class of functions) by applying Lemma 2.13 of Pakes and Pollard (1989), where $\alpha = 1$, $f(\cdot, t_0) = 0$, $\phi(\cdot) = \|X_j\|\frac{1}{h}1(0 < d_j \leq h)$ and the envelope function is $F_n(\cdot) = M\phi(\cdot)$ for some finite constant M. Since $\mathrm{E}[F_n] = \mathrm{E}\left[\|X_j\|\frac{1}{h}1(0 < d_j \leq h)\right] = O(h) < \infty$, Lemma 2.8 of Pakes and Pollard (1989) implies

$$\sup_{b\in\mathcal{B}} |S_n(b) - S_n(\beta) - E[S_n(b) - S_n(\beta)]| \xrightarrow{p} 0.$$

As to $\inf_{\|H(b-\beta)\|>\delta} E[S_n(b) - S_n(\beta)]$, by Proposition 1 of Wang and Scott (1994),

$$\begin{aligned} &\mathrm{E}[S_n(b) - S_n(\beta)] \\ \doteq &\frac{1}{n}\sum_j g(0)\left[X_j^T H^{-1}H(b-\beta)\right]^2\frac{1}{h}1(0 < d_j \leq h) \\ &-\frac{1}{n}\sum_j 2g(0)\left[\frac{m(d_{i+j}) - m(d_{i-j})}{2} - X_j^T\beta\right]\left[X_j^T H^{-1}H(b-\beta)\right]\frac{1}{h}1(0 < d_j \leq h) \\ \gtrsim &\delta^2 - h^5\delta, \end{aligned}$$

where $\doteq$ means that the higher-order terms are omitted, and $\gtrsim$ means the left side is bounded blow by a constant times the right side.

We then derive the asymptotic distribution of $\sqrt{nh}H\left(\widehat{\beta} - \beta\right)$ by applying the empirical process technique. First, the first order conditions can be written as

$$\frac{1}{n}\sum_j \mathrm{sign}\left(\tilde{Y}_{ij}^{(1)} - Z_j^T H\widehat{\beta}\right)Z_j\frac{\sqrt{h}}{h}1(0 < d_j \leq h) = o_p(1),$$

which is denoted as

$$\frac{1}{n}\sum_j f_n'(\tilde{Y}_{ij}^{(1)}|\widehat{\beta}) \triangleq S_n'\left(\widehat{\beta}\right) = o_p\left(1\right),$$

where $Z_j = H^{-1}X_j$. By Example 2.9 of Pakes and Pollard (1989), $\mathcal{F}_n'$ forms an Euclidean-class of functions with envelope $F_n' = \|Z_j\|\frac{\sqrt{h}}{h}1(0 < d_j \le h)$, where $\mathcal{F}_n' = \left\{f_n'(\tilde{Y}_{ij}^{(1)}|b) : b \in \mathcal{B}\right\}$, and $E\left[F_n'^2\right] < \infty$. So by Lemma 2.17 of Pakes and Pollard (1989) and $H\left(\widehat{\beta}-\beta\right) = o_p(1)$,

$$\mathbb{G}_n\left(f_n'(\tilde{Y}_{ij}^{(1)}|\widehat{\beta})\right) = \mathbb{G}_n\left(f_n'(\tilde{Y}_{ij}^{(1)}|\beta)\right) + o_p(1),$$

where $\mathbb{G}_n\left(f\right) = \sqrt{n}\left(P_n - P\right)f$ is the standardized empirical process, and P_n is the empirical measure of the original data. Since

$$\sqrt{n}\sum_j\left(\mathrm{E}\left[f_n'(\tilde{Y}_{ij}^{(1)}|\widehat{\beta})\right] - \mathrm{E}\left[f_n'(\tilde{Y}_{ij}^{(1)}|\beta)\right]\right) \doteq -\sqrt{nh}\frac{2g(0)}{nh}\sum_j Z_j Z_j^T H\left(\widehat{\beta}-\beta\right),$$

and

$$\frac{1}{nh}\sum_j \mathrm{sign}\left(\tilde{Y}_{ij}^{(1)} - Z_j^T H\beta\right)Z_j 1(0 < d_j \le h)$$

$$-\frac{1}{nh}\sum_j \mathrm{sign}\left(\tilde{Y}_{ij}^{(1)} - \frac{m\left(d_{i+j}\right) - m(d_{i-j})}{2}\right)Z_j 1(0 < d_j \le h) \doteq \frac{2g(0)}{nh}\sum_j Z_j d_j^5 \frac{m^{(5)}(x_i)}{5!},$$

we have

$$\sqrt{nh}\left(H\left(\widehat{\beta}-\beta\right) - \left[\frac{1}{nh}\sum_j Z_j Z_j^T\right]^{-1}\frac{1}{nh}\sum_j Z_j d_j^5\frac{m^{(5)}(x_i)}{5!}\right)$$

$$\doteq \frac{1}{2g(0)}\left[\frac{1}{nh}\sum_j Z_j Z_j^T\right]^{-1}\frac{1}{\sqrt{nh}}\sum_j \mathrm{sign}\left(\tilde{\zeta}_{ij}^{(1)}\right)Z_j 1(0 < d_j \le h).$$

In other words,

$$2g(0)V_k\left(\widehat{\beta}-\beta - V_k^{-2}\sum_{j=1}^k X_j d_j^5\frac{m^{(5)}(x_i)}{5!}\right) \doteq V_k^{-1}\sum_{j=1}^k \mathrm{sign}\left(\tilde{\zeta}_{ij}^{(1)}\right)X_j,$$

where $V_k = (\sum_{j=1}^k X_j X_j^T)^{1/2}$ is a symmetric positive definite matrix. By Cramér-Wold device and Lyapunov CLT, we complete the proof of asymptotic normality.

The bias of $\hat{\beta}_{i1}$ is

$$\mathrm{Bias}[\hat{\beta}_{i1}] \doteq [1,0]\left(\sum_j X_j X_j^T\right)^{-1}\sum_{j=1}^k X_j d_j^5\frac{m^{(5)}(x_i)}{5!} = -\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4}.$$

and the variance is

$$\mathrm{Var}[\hat{\beta}_{i1}] \doteq \frac{1}{4g(0)^2}[1,0]\left(\sum_j X_j X_j^T\right)^{-1}\begin{bmatrix}1\\0\end{bmatrix} \approx \frac{75}{16g(0)^2}\frac{n^2}{k^3}.$$

Combining the squared bias and the variance, we obtain the AMSE

$$\mathrm{AMSE}[\hat{\beta}_{i1}] = \frac{m^{(5)}(x_i)^2}{504^2}\frac{k^8}{n^8} + \frac{75}{16g(0)^2}\frac{n^2}{k^3}. \tag{17}$$

To minimize (17) with respect to k, we take the first-order derivative of (17) and yield the gradient as

$$\frac{\mathrm{dAMSE}[\hat{\beta}_{i1}]}{\mathrm{d}k} = \frac{m^{(5)}(x_i)^2}{31752}\frac{k^7}{n^8} - \frac{225}{16g(0)^2}\frac{n^2}{k^4}.$$

Our optimization problem is to solve $\frac{\mathrm{dAMSE}[\hat{\beta}_{i1}]}{\mathrm{d}k} = 0$. So we obtain

$$k_{opt} = \left(\frac{893025}{2g(0)^2 m^{(5)}(x_i)^2}\right)^{1/11} n^{10/11} \approx 3.26\left(\frac{1}{g(0)^2 m^{(5)}(x_i)^2}\right)^{1/11} n^{10/11},$$

and

$$\mathrm{AMSE}[\hat{\beta}_{i1}] \approx 0.19(m^{(5)}(x_i)^6/g(0)^{16})^{1/11}n^{-8/11}.$$

Appendix B. Proof of Theorem 2

Rewrite the objective function as a U-process,

$$S_n\left(b\right) = \sum_{l<j} f_n\left(Y_{i+j}, Y_{i+l}|b\right),$$

where

$$f_n\left(Y_{i+j}, Y_{i+l}|b\right) = \left|Y_{i+j} - Y_{i+l} - b_1(d_j - d_l) - b_{i2}(d_j^2 - d_l^2) - b_3(d_j^3 - d_l^3) - b_4(d_j^4 - d_l^4)\right| \cdot \frac{1}{h^2}1(0 < |d_j| \le h)1(0 < |d_l| \le h)$$

with $b = (b_{i1}, b_{i2}, b_{i3}, b_{i4})^T$ and $h = k/n$. Define $U_n = \frac{2}{n(n-1)}S_n\left(b\right)$, $H = \mathrm{diag}\left\{h, h^2, h^3, h^4\right\}$ and $X_{jl} = \left(d_j - d_l, d_j^2 - d_l^2, d_j^3 - d_l^3, d_j^4 - d_l^4\right)^T$. Note that $\arg\min_b S_n\left(b\right) = \arg\min_b U_n(b) = \arg\min_b\left[U_n(b) - U_n\left(\beta\right)\right]$, where $\beta = \left(m^{(1)}(x_i), m^{(2)}(x_i)/2!, m^{(3)}(x_i)/3!, m^{(4)}(x_i)/4!\right)^T$.

We first show that $H\left(\widehat{\beta}-\beta\right) = o_p(1)$, where $\widehat{\beta} = (\hat{\beta}_{i1}^{\mathrm{RLowLAD}}, \hat{\beta}_{i2}^{\mathrm{RLowLAD}}, \hat{\beta}_{i3}^{\mathrm{RLowLAD}}, \hat{\beta}_{i4}^{\mathrm{RLowLAD}})^T$. We use Lemma 4 of Porter and Yu (2015) to show the consistency. Essentially, we need to show that

(i) $\sup_{b\in\mathcal{B}}\left|U_n\left(b\right) - U_n\left(\beta\right) - \mathrm{E}\left[U_n\left(b\right) - U_n\left(\beta\right)\right]\right| \xrightarrow{p} 0,$

(ii) $\inf_{\|H(b-\beta)\|>\delta} \mathrm{E}\left[U_n(b) - U_n(\beta)\right] > \varepsilon$ for n large enough, where $\mathcal{B}$ is a compact parameter space for β, and δ and ε are fixed positive small numbers.

We use Theorem A.2 of Ghosal et al. (2000) to show (i), where

$$\mathcal{F}_n = \left\{ f_n\left(Y_{i+j}, Y_{i+l}|b\right) - f_n\left(Y_{i+j}, Y_{i+l}|\beta\right) : b \in \mathcal{B} \right\}.$$

Note that $\mathcal{F}_n$ forms an Euclidean-class of functions by applying Lemma 2.13 of Pakes and Pollard (1989), where $\alpha = 1$, $f(\cdot, t_0) = 0$, $\phi(\cdot) = \|X_{jl}\| \frac{1}{h^2} 1(|d_j| \leq h) 1(|d_l| \leq h)$ and the envelope function is $F_n(\cdot) = M\phi(\cdot)$ for some finite constant M. It follows that

$$N\left(\varepsilon \|F_n\|_{Q,2}, \mathcal{F}_n, L_2(Q)\right) \lesssim \varepsilon^{-C}$$

for any probability measure Q and some positive constant C, where $\lesssim$ means the left side is bounded by a constant times the right side. Hence,

$$\frac{1}{n}\mathrm{E}\left[\int_0^\infty \log N\left(\varepsilon, \mathcal{F}_n, L_2\left(U_2^n\right)\right) d\varepsilon\right] \lesssim \frac{1}{n}\sqrt{\mathrm{E}\left[F_n^2\right]} \int_0^\infty \log \frac{1}{\varepsilon} d\varepsilon = O\left(\frac{1}{n}\right),$$

where U_2^n is the random discrete measure putting mass $\frac{1}{n(n-1)}$ on each of the points (Y_{i+j}, Y_{i+l}). Next, by Lemma A.2 of Ghosal et al. (2000), the projections

$$\overline{f}_n\left(Y_{i+j}|b\right) = \int f_n\left(Y_{i+j}, Y_{i+l}|b\right) dF_{Y_{i+l}}(Y_{i+l})$$

satisfy

$$\sup_Q N\left(\varepsilon \left\|\overline{F}_n\right\|_{Q,2}, \overline{\mathcal{F}}_n, L_2(Q)\right) \lesssim \varepsilon^{-2C},$$

where $\overline{\mathcal{F}}_n = \left\{ \overline{f}_n\left(Y_{i+j}|b\right) - \overline{f}_n\left(Y_{i+j}|\beta\right) : b \in \mathcal{B} \right\}$, and $\overline{F}_n$ is an envelope of $\overline{\mathcal{F}}_n$. Thus

$$\frac{1}{\sqrt{n}}\mathrm{E}\left[\int_0^\infty \log N\left(\varepsilon, \overline{\mathcal{F}}_n, L_2\left(P_n\right)\right) d\varepsilon\right] \lesssim \frac{1}{\sqrt{n}}\sqrt{E\left[\overline{F}_n^2\right]} \int_0^\infty \log \frac{1}{\varepsilon} d\varepsilon = O\left(\frac{1}{\sqrt{n}}\right).$$

By Theorem A.2 and Markov's inequality, $\sup_{b\in\mathcal{B}} \left|U_n(b) - U_n(\beta) - \mathrm{E}\left[U_n(b) - U_n(\beta)\right]\right| \xrightarrow{p} 0$.

As to $\inf_{\|H(b-\beta)\|>\delta} E\left[U_n(b) - U_n(\beta)\right]$, by Proposition 1 of Wang and Scott (1994),

$$\begin{aligned}
&\mathrm{E}\left[U_n(b) - U_n(\beta)\right] \\
\doteq &\frac{2}{n(n-1)} \sum_{l<j} \frac{g(0)}{2} \left[X_{jl}^T H^{-1} H\left(b-\beta\right)\right]^2 \frac{1}{h^2} 1(0 < |d_j| \leq h) 1(0 < |d_l| \leq h) \\
&- \frac{2}{n(n-1)} \sum_{l<j} g(0) \left[m\left(d_{i+j}\right) - m(d_{i+l}) - X_{jl}^T\beta\right] \left[X_{jl}^T H^{-1} H\left(b-\beta\right)\right] \\
&\frac{1}{h^2} 1(0 < |d_j| \leq h) 1(0 < |d_l| \leq h) \\
\gtrsim &\delta^2 - h^5\delta.
\end{aligned}$$

We then derive the asymptotic distribution of $\sqrt{nh}H\left(\widehat{\beta} - \beta\right)$. First, by Theorem A.1 of Ghosal et al. (2000), we approximate the first order conditions by an empirical process. Second, we derive the asymptotic distribution of $\sqrt{nh}H\left(\widehat{\beta} - \beta\right)$ by applying the empirical process technique.

First, the first order conditions can be written as

$$\frac{2}{n(n-1)} \sum_{l<j} \mathrm{sign}\left(Y_{i+j} - Y_{i+l} - Z_{jl}^T H\widehat{\beta}\right) Z_{jl} \frac{\sqrt{h}}{h^2} 1(0 < |d_j| \leq h) 1(0 < |d_l| \leq h) = o_p(1),$$

which is denoted as

$$\frac{2}{n(n-1)} \sum_{l<j} f_n'(Y_{i+j}, Y_{i+l}|\widehat{\beta}) \triangleq \frac{2}{n(n-1)} S_n'\left(\widehat{\beta}\right) = o_p(1),$$

where $Z_{jl} = H^{-1}X_{jl}$. By Example 2.9 of Pakes and Pollard (1989), $\mathcal{F}_n'$ forms an Euclidean-class of functions with envelope $F_n' = \|Z_{jl}\| \frac{\sqrt{h}}{h^2} 1(|d_j| \leq h) 1(|d_l| \leq h)$, where

$$\mathcal{F}_n' = \left\{ f_n'(Y_{i+j}, Y_{i+l}|b) : b \in \mathcal{B} \right\},$$

so

$$N\left(\varepsilon \left\|F_n'\right\|_{Q,2}, \mathcal{F}_n', L_2(Q)\right) \lesssim \varepsilon^{-V}$$

for any probability measure Q and some positive constant V. By Theorem A.1 and the discussion following Theorem A.1 and A.2 in Ghosal et al. (2000), it follows that

$$\begin{aligned}
&n\mathrm{E}\left[\sup_{f_n'\in\mathcal{F}_n'} \left|U_2^n f_n' - 2P_n\left[\overline{E}_2\left[f_n'(Y_{i+j}, Y_{i+l}|b)\right]\right] - \overline{E}\left[f_n'(Y_{i+j}, Y_{i+l}|b)\right]\right|\right] \\
\lesssim &E\left[\int_0^\infty \log N\left(\varepsilon, \mathcal{F}_n', L_2\left(U_2^n\right)\right) d\varepsilon\right] \lesssim \int_0^1 \log\left(\varepsilon^{-V}\right) d\varepsilon \sqrt{\mathrm{E}\left[\left(F_n'\right)^2\right]} \lesssim h^{-1/2},
\end{aligned}$$

where $\overline{E}_2[\cdot]$ takes expectation on Y_{i+l} and also averages over d_l, and $\overline{E}[\cdot]$ takes expectation on (Y_{i+j}, Y_{i+l}) and also averages over (d_j, d_l). As a result,

$$\sqrt{n} \sup_{f_n'\in\mathcal{F}_n'} \left|U_2^n f_n' - 2P_n\left[\overline{E}_2[\cdot]\left[f_n'(Y_{i+j}, Y_{i+l}|b)\right]\right] + \overline{E}\left[f_n'(Y_{i+j}, Y_{i+l}|b)\right]\right| = o_p(1),$$

which implies

$$\sqrt{n}\left(2P_n\left[\overline{E}_2\left[f_n'(Y_{i+j}, Y_{i+l}|\widehat{\beta})\right]\right] - \overline{E}\left[f_n'(Y_{i+j}, Y_{i+l}|\widehat{\beta})\right]\right) = o_p(1),$$

where

$$\begin{aligned}
&\overline{E}_2\left[f_n'(Y_{i+j}, Y_{i+l}|b)\right] \\
= &\frac{\sqrt{h}}{nh^2} \sum_l \left[2F_\epsilon\left(Y_{i+j} - m(d_{i+l}) - Z_{jl}^T Hb\right) - 1\right] Z_{jl} 1(0 < |d_j| \leq h) 1(0 < |d_l| \leq h)
\end{aligned}$$

with $F_\epsilon(\cdot)$ being the cumulative distribution function of ϵ. In other words,

$$2\mathbb{G}_n\left(\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l}|\widehat{\beta})\right]\right)+\sqrt{nE}\left[f_n'(Y_{i+j},Y_{i+l}|\widehat{\beta})\right]=o_p(1).$$

By Lemma 2.13 of Pakes and Pollard (1989), $\mathcal{F}_{1n}'=\left\{\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l}|b)\right]:b\in\mathcal{B}\right\}$ is Euclidean with envelope $F_{1n}=\frac{\sqrt{h}}{nh^2}\sum_l\|Z_{jl}\|1(0<|d_j|\le h)1(0<|d_l|\le h)$, so by Lemma 2.17 of Pakes and Pollard (1989) and $H\left(\widehat{\beta}-\beta\right)=o_p(1)$,

$$\mathbb{G}_n\left(\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l}|\widehat{\beta})\right]\right)=\mathbb{G}_n\left(\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\right)+o_p(1).$$

As a result,

$$\begin{aligned}
&2\mathbb{G}_n\left(\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\right)+\sqrt{nE}\left[f_n'(Y_{i+j},Y_{i+l}|\widehat{\beta})\right]\\
=&2\sqrt{n}P_n\left(\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\right)-2\sqrt{nE}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\\
+&\sqrt{n}\left(\overline{E}\left[f_n'(Y_{i+j},Y_{i+l}|\widehat{\beta})\right]-\overline{E}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\right)+\sqrt{nE}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\\
=&2\sqrt{n}P_n\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l})\right]+2\sqrt{n}P_n\left(\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]-\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l})\right]\right)\\
+&\sqrt{n}\left(\overline{E}\left[f_n'(Y_{i+j},Y_{i+l}|\widehat{\beta})\right]-\overline{E}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\right)-\sqrt{nE}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\\
=&2\sqrt{n}P_n\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l})\right]+\sqrt{n}\left(\overline{E}\left[f_n'(Y_{i+j},Y_{i+l}|\widehat{\beta})\right]-\overline{E}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\right)\\
+&\sqrt{nE}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\\
=&o_p(1),
\end{aligned}$$

where

$$\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l})\right]=\frac{\sqrt{h}}{nh^2}\sum_l\left[2F_\epsilon(\epsilon_{i+j})-1\right]Z_{jl}1(0<|d_j|\le h)1(0<|d_l|\le h)$$

satisfies $E\left[\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l})\right]\right]=0$, and the second to last equality is from

$$\sqrt{n}P_n\left(\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]-\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l})\right]\right)\doteq\sqrt{nE}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right].$$

Since

$$\begin{aligned}
\overline{E}\left[f_n'(Y_{i+j},Y_{i+l}|b)\right]=&\frac{\sqrt{h}}{n^2h^2}\sum_{l,j}Z_{jl}1(0<|d_j|\le h)1(0<|d_l|\le h)\\
&\cdot\left[2\int F_\epsilon\left(\epsilon+m(d_{i+j})-m(d_{i+l})-Z_{jl}^THb\right)-1\right]f(\epsilon)\,d\epsilon,
\end{aligned}$$

we have

$$\sqrt{n}\left(\overline{E}\left[f_n'(Y_{i+j},Y_{i+l}|\widehat{\beta})\right]-\overline{E}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\right)\doteq-\sqrt{nh}g(0)\left(\frac{1}{n^2h^2}\sum_{l,j}Z_{jl}Z_{jl}^T\right)H\left(\widehat{\beta}-\beta\right),$$

$$\sqrt{nE}\left[f_n'(Y_{i+j},Y_{i+l}|\beta)\right]\doteq\sqrt{nh}g(0)\frac{1}{n^2h^2}\sum_{l,j}Z_{jl}\left(d_j^5-d_l^5\right)\frac{m^{(5)}(x_i)}{5!}.$$

In summary,

$$\begin{aligned}
&\sqrt{nh}\left(H\left(\widehat{\beta}-\beta\right)-\left(\frac{1}{n^2h^2}\sum_{l,j}Z_{jl}Z_{jl}^T\right)^{-1}\left(\frac{1}{n^2h^2}\sum_{l,j}Z_{jl}\left(d_j^5-d_l^5\right)\right)\frac{m^{(5)}(x_i)}{5!}\right)\\
\doteq&2g(0)^{-1}\left(\frac{1}{n^2h^2}\sum_{l,j}Z_{jl}Z_{jl}^T\right)^{-1}\sqrt{n}P_n\overline{E}_2\left[f_n'(Y_{i+j},Y_{i+l})\right],
\end{aligned}$$

Thus, the asymptotic bias is

$$e^TH^{-1}\left(\frac{1}{n^2h^2}\sum_{l,j}Z_{jl}Z_{jl}^T\right)^{-1}\left(\frac{1}{n^2h^2}\sum_{l,j}Z_{jl}\left(d_j^5-d_l^5\right)\right)\frac{m^{(5)}(x_i)}{5!}=-\frac{m^{(5)}(x_i)}{504}\frac{k^4}{n^4},$$

and the asymptotic variance is

$$\frac{4}{kg(0)^2}e^TH^{-1}G^{-1}VG^{-1}H^{-1}e=\frac{75}{24g(0)^2}\frac{n^2}{k^3},$$

where $e=(1,0,0,0)^T$, $G=\frac{1}{k^2}\sum_{l,j}Z_{jl}Z_{jl}^T$, and $V=\frac{1}{3k}\sum_{j=-k}^k(\frac{1}{k}\sum_{l=-k}^kZ_{jl})(\frac{1}{k}\sum_{l=-k}^kZ_{jl})^T$ with $\mathrm{Var}\left(2F_\epsilon\left(\epsilon_{i+j}\right)-1\right)=1/3$.

Appendix C. Proof of Theorem 9

Proof of Theorem 9 Following the proof of Theorem 1, the leading term of the bias is

$$\mathrm{Bias}[\hat{\beta}_{i11}]=\frac{m^{(2)}(x_i)}{2}\frac{k^4+2k^3i-2ki^3-i^3}{n(k^3+3k^2i+3ki^2+i^3)},$$

and the leading term of the variance is

$$\mathrm{Var}[\hat{\beta}_{i11}]=\frac{3}{f(0)^2}\frac{n^2}{k^3+3k^2i+3ki^2+i^3}.$$

■

Appendix D. Proof of Theorem 10

Proposition 12 *If the errors ϵ_i are iid with a symmetric (about 0), continuous, positive density function $f(\cdot)$, then $\tilde{\delta}_{ij}=\epsilon_{i+j}+\epsilon_{i-j}$ (j=1, ..., k) are iid with $Median[\tilde{\delta}_{ij}]=0$ and a continuous, positive density $h(\cdot)$ in a neighborhood of 0, where*

$$h(x)=\int_{-\infty}^{\infty}f(x-\epsilon)f(\epsilon)\mathrm{d}\epsilon.$$

Proof of Proposition 12 The distribution of $\tilde{\delta}_{ij} = \epsilon_{i+j} + \epsilon_{i-j}$ is

$$\begin{aligned}
F_{\tilde{\delta}_{ij}}(x) =& P(\epsilon_{i+j} + \epsilon_{i-j} \le x) \\
=& \iint_{\epsilon_{i+j} \le x - \epsilon_{i-j}} f(\epsilon_{i+j}) f(\epsilon_{i-j}) \mathrm{d}\epsilon_{i+j} \mathrm{d}\epsilon_{i-j} \\
=& \int_{-\infty}^{\infty} \{ \int_{-\infty}^{x-\epsilon_{i-j}} f(\epsilon_{i+j}) \mathrm{d}\epsilon_{i+j} \} f(\epsilon_{i-j}) \mathrm{d}\epsilon_{i-j} \\
=& \int_{-\infty}^{\infty} F(x - \epsilon_{i-j}) f(\epsilon_{i-j}) \mathrm{d}\epsilon_{i-j}.
\end{aligned}$$

Then the density of $\tilde{\delta}_{ij}$ is

$$h(x) \triangleq \frac{\mathrm{d}F_{\tilde{\delta}_{ij}}(x)}{\mathrm{d}x} = \int_{-\infty}^{\infty} f(x - \epsilon_{i-j}) f(\epsilon_{i-j}) \mathrm{d}\epsilon_{i-j}.$$

By the symmetry of the density function, we have

$$\begin{aligned}
F_{\tilde{\delta}_{ij}}(0) =& \int_{-\infty}^{\infty} F(-\epsilon_{i-j}) f(\epsilon_{i-j}) \mathrm{d}\epsilon_{i-j} \\
=& \int_{-\infty}^{\infty} (1 - F(\epsilon_{i-j})) f(\epsilon_{i-j}) \mathrm{d}\epsilon_{i-j} \\
=& (F - \frac{1}{2} F^2(\epsilon_{i-j})) \mid_{-\infty}^{\infty} \\
=& \frac{1}{2}, \\
h(0) =& \int_{-\infty}^{\infty} f^2(\epsilon_{i-j}) \mathrm{d}\epsilon_{i-j}.
\end{aligned}$$

■

Proof of Theorem 10 Following the proof of Theorem 1, the asymptotic bias is

$$\mathrm{Bias}[\hat{\alpha}_{i2}] = -\frac{m^{(6)}(x_i)}{792} \frac{k^4}{n^4} + o\left(\frac{k^4}{n^4}\right),$$

and the asymptotic variance of $\hat{\alpha}_{i1}$ is

$$\mathrm{Var}[\hat{\alpha}_{i2}] = \frac{2205}{64h(0)^2} \frac{n^4}{k^5} + o\left(\frac{n^4}{k^5}\right).$$

Combining the squared bias and the variance, we obtain the AMSE as

$$\mathrm{AMSE}[\hat{\alpha}_{i2}] = \frac{m^{(6)}(x_i)^2}{792^2} \frac{k^8}{n^8} + \frac{2205}{64h(0)^2} \frac{n^4}{k^5}. \tag{18}$$

To minimize (18) with respect to k, we take the first-order derivative of (18) and yield the gradient as

$$\frac{\mathrm{dAMSE}[\hat{\alpha}_{i2}]}{\mathrm{d}k} = \frac{m^{(6)}(x_i)^2}{78408} \frac{k^7}{n^8} - \frac{11025}{16h(0)^2} \frac{n^4}{k^6}.$$

Now the optimization problem is to solve $\frac{\mathrm{dAMSE}[\hat{\alpha}_{i2}]}{\mathrm{d}k} = 0$. So we obtain

$$k_{opt} = \left(\frac{108056025}{8h(0)^2 m^{(6)}(x_i)^2}\right)^{1/13} n^{12/13} \approx 3.54 \left(\frac{1}{h(0)^2 m^{(6)}(x_i)^2}\right)^{1/13} n^{12/13},$$

and

$$\mathrm{AMSE}[\hat{\alpha}_{i2}] \approx 0.29 (m^{(6)}(x_i)^{10} / h(0)^{16})^{1/13} n^{-8/13}.$$

■

Appendix E. Variance Ratios for Popular Distributions

Variance Ratios for Eight Error Distributions

In this subsection, we investigate the variance ratio of the RLowLAD estimator with respect to the LowLSR and LAD estimators for eight error distributions.

From the main text,

$$\mathrm{R}_{\mathrm{LowLSR}}(f) = 3\sigma^2 g(0)^2, \quad \mathrm{R}_{\mathrm{LAD}}(f) = \frac{3g(0)^2}{4f(0)^2}.$$

Example 1: Normal distribution. The error density function is $f(\epsilon) = \frac{1}{\sqrt{2\pi}} \exp(-\epsilon^2/2)$, which implies

$$f(0) = \frac{1}{\sqrt{2\pi}}, \quad g(0) = 2\int_{-\infty}^{\infty} \frac{1}{2\pi} e^{-\epsilon^2} \mathrm{d}\epsilon = \frac{1}{\sqrt{\pi}}.$$

Due to $\sigma^2 = 1$, we have

$$\mathrm{R}_{\mathrm{LowLSR}}(f) = 3/\pi \approx 0.95, \quad \mathrm{R}_{\mathrm{LAD}}(f) = 1.50.$$

In other words, the RLowLAD estimator is almost as efficient as the LowLSR estimator for the normal distribution.

Example 2: Mixed normal distribution. The error density function is

$$f(\epsilon; \alpha, \sigma_0) = (1 - \alpha) \frac{1}{\sqrt{2\pi}} e^{-\epsilon^2/2} + \alpha \frac{1}{\sqrt{2\pi}\sigma_0} e^{-\epsilon^2/(2\sigma_0^2)}$$

with $0 < \alpha \le 1/2$ and $\sigma_0 > 1$, which implies

$$\begin{aligned}
f(0;\alpha,\sigma_0) =&(1-\alpha)\frac{1}{\sqrt{2\pi}} + \alpha\frac{1}{\sqrt{2\pi}\sigma_0},\\
g(0;\alpha,\sigma_0) =&2\int_{-\infty}^{\infty} f^2(\epsilon)\mathrm{d}\epsilon\\
=&2\left\{(1-\alpha)^2\int_{-\infty}^{\infty}\frac{1}{2\pi}\mathrm{e}^{-\epsilon^2}\mathrm{d}\epsilon + 2\alpha(1-\alpha)\int_{-\infty}^{\infty}\frac{1}{2\pi\sigma_0}\mathrm{e}^{-(\frac{\epsilon^2}{2}+\frac{\epsilon^2}{2\sigma_1^2})}\mathrm{d}\epsilon\right.\\
&\left.+\alpha^2\int_{-\infty}^{\infty}\frac{1}{2\pi\sigma_0^2}\mathrm{e}^{-\frac{x^2}{\sigma_0^2}}\mathrm{d}\epsilon\right\}\\
=&2\left\{(1-\alpha)^2\frac{1}{2\sqrt{\pi}} + 2\alpha(1-\alpha)\frac{1}{\sqrt{2\pi}\sqrt{1+\sigma_0^2}} + \alpha^2\frac{1}{2\sqrt{\pi}\sigma_0}\right\},\\
\mathrm{Var}(\epsilon_i) =&(1-\alpha)+\alpha\sigma_0^2 \triangleq \sigma^2.
\end{aligned}$$

Thus,

$$\begin{aligned}
\mathrm{R}_{\mathrm{LowLSR}}(\alpha,\sigma_0) =&12\left\{(1-\alpha)+\alpha\sigma_0^2\right\}\left\{(1-\alpha)^2\frac{1}{2\sqrt{\pi}} + 2\alpha(1-\alpha)\frac{1}{\sqrt{2\pi}\sqrt{1+\sigma_0^2}} + \alpha^2\frac{1}{2\sqrt{\pi}\sigma_0}\right\}^2,\\
\mathrm{R}_{\mathrm{LAD}}(\alpha,\sigma_0) =&\frac{3\left\{(1-\alpha)^2\frac{1}{2\sqrt{\pi}} + 2\alpha(1-\alpha)\frac{1}{\sqrt{2\pi}\sqrt{1+\sigma_0^2}} + \alpha^2\frac{1}{2\sqrt{\pi}\sigma_0}\right\}^2}{\left\{(1-\alpha)\frac{1}{\sqrt{2\pi}} + \alpha\frac{1}{\sqrt{2\pi}\sigma_0}\right\}^2}.
\end{aligned}$$

In particular,

$$\begin{aligned}
\mathrm{R}_{\mathrm{LowLSR}}(0.1,3) \approx&1.80, \quad \mathrm{R}_{\mathrm{LowLSR}}(0.1,10) \approx 10.90,\\
\mathrm{R}_{\mathrm{LAD}}(0.1,3) \approx&1.38, \quad \mathrm{R}_{\mathrm{LAD}}(0.1,10) \approx 1.27.
\end{aligned}$$

Example 3: t distribution. The error density function is

$$f(\epsilon;\nu) = \frac{\Gamma((\nu+1)/2)}{\sqrt{\nu\pi}\Gamma(\nu/2)}\left(1+\frac{\epsilon^2}{\nu}\right)^{-(\nu+1)/2)}$$

with the degree of freedom $\nu > 2$, which implies

$$\begin{aligned}
f(0) =&\frac{\Gamma((\nu+1)/2)}{\sqrt{\nu\pi}\Gamma(\nu/2)},\\
g(0) =&2\int_{-\infty}^{\infty}\frac{1}{\nu\pi}\left(\frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)}\right)^2\left(1+\frac{\epsilon^2}{\nu}\right)^{-(\nu+1)}\mathrm{d}\epsilon = \frac{2}{\sqrt{\nu\pi}}\left(\frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)}\right)^2\frac{\Gamma(\nu+1/2)}{\Gamma(\nu+1)}.
\end{aligned}$$

Due to $\sigma^2 = \nu/(\nu-2)$, we have

$$\begin{aligned}
\mathrm{R}_{\mathrm{LowLSR}}(\nu) =&\frac{12}{(\nu-2)\pi}\left(\frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)}\right)^4\left(\frac{\Gamma(\nu+1/2)}{\Gamma(\nu+1)}\right)^2,\\
\mathrm{R}_{\mathrm{LAD}}(\nu) =&3\left(\frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)}\right)^2\left(\frac{\Gamma(\nu+1/2)}{\Gamma(\nu+1)}\right)^2.
\end{aligned}$$

For $\nu = 3$,

$$\mathrm{R}_{\mathrm{LowLSR}}(3) = 75/(4\pi^2) \approx 1.90, \quad \mathrm{R}_{\mathrm{LAD}}(3) = 75/64 \approx 1.17.$$

Example 4: Laplace (double exponential) distribution. The error density function is $f(\epsilon) = \frac{1}{2}\exp(-|\epsilon|)$, which implies

$$f(0) = \frac{1}{2}, \quad g(0) = 2\int_{-\infty}^{\infty}\frac{1}{4}e^{-2|\epsilon|}\mathrm{d}\epsilon = \frac{1}{2}.$$

Due to $\sigma^2 = 2$, we have

$$\mathrm{R}_{\mathrm{LowLSR}}(f) = 1.50, \quad \mathrm{R}_{\mathrm{LAD}}(f) = 0.75.$$

Example 5: Logistic distribution. The error density function is $f(\epsilon) = \exp(\epsilon)/(\exp(\epsilon)+1)^2$, which implies

$$f(0) = \frac{1}{4}, \quad g(0) = 2\int_{-\infty}^{\infty}\frac{e^{2\epsilon}}{(\exp(\epsilon)+1)^4}\mathrm{d}\epsilon = \frac{1}{3}.$$

Due to $\sigma^2 = \pi^2/3$, we have

$$\mathrm{R}_{\mathrm{LowLSR}}(f) = \pi^2/9 \approx 1.10, \quad \mathrm{R}_{\mathrm{LAD}}(f) = 4/3 \approx 1.33.$$

Example 6: Cauchy distribution. The error density function is $f(\epsilon) = 1/(\pi(1+\epsilon^2))$, which implies

$$f(0) = \frac{1}{\pi}, \quad g(0) = \frac{1}{\pi}, \quad \mathrm{Var}(\epsilon) = \infty.$$

Thus,

$$\mathrm{R}_{\mathrm{LowLSR}}(f) = \infty, \quad \mathrm{R}_{\mathrm{LAD}}(f) = 0.75.$$

Example 7: Mixed double Gamma distribution. The error density function is

$$f(\epsilon;\alpha,k) = (1-\alpha)\frac{1}{2}e^{-|\epsilon|} + \alpha\frac{1}{2\Gamma(k+1)}|\epsilon|^k e^{-|\epsilon|}$$

with parameter $k > 0$ and the mixed ratio α, which implies

$$\begin{aligned}
f(0;\alpha,k) =&\frac{1-\alpha}{2} + \frac{\alpha}{2\Gamma(k+1)},\\
g(0;\alpha,k) =&2\int_{-\infty}^{\infty} f^2(\epsilon;\alpha,k)\mathrm{d}\epsilon\\
=&\int_{-\infty}^{\infty}\frac{(1-\alpha)^2}{2}e^{-2|\epsilon|} + \frac{(1-\alpha)\alpha}{\Gamma(k+1)}|\epsilon|^k e^{-2|\epsilon|} + \frac{\alpha^2}{2\Gamma(k+1)^2}|\epsilon|^{2k}e^{-2|\epsilon|}\mathrm{d}\epsilon\\
=&\frac{(1-\alpha)^2}{2} + \frac{\alpha(1-\alpha)}{2^k} + \frac{\alpha^2\Gamma(2k+1)}{2^{2k+1}\Gamma^2(k+1)},\\
\mathrm{Var}(\epsilon|\alpha,k) =&1+\alpha k.
\end{aligned}$$

Thus,

$$\mathrm{R}_{\mathrm{LowLSR}}(\alpha,k) =3(1+\alpha k)\left\{\frac{(1-\alpha)^2}{2}+\frac{\alpha(1-\alpha)}{2^k}+\frac{\alpha^2\Gamma(2k+1)}{2^{2k+1}\Gamma^2(k+1)}\right\},$$
$$\mathrm{R}_{\mathrm{LAD}}(\alpha,k) = 3\left\{\frac{(1-\alpha)^2}{2}+\frac{\alpha(1-\alpha)}{2^k}+\frac{\alpha^2\Gamma(2k+1)}{2^{2k+1}\Gamma^2(k+1)}\right\}^2\Bigg/\left\{4\left(\frac{1-\alpha}{2}+\frac{\alpha}{2\Gamma(k+1)}\right)^2\right\}.$$

In particular,

$$\begin{aligned}\mathrm{R}_{\mathrm{LowLSR}}(0.1,3)\approx&1.63,\quad \mathrm{R}_{\mathrm{LowLSR}}(0.1,10)\approx 2.44,\\ \mathrm{R}_{\mathrm{LAD}}(0.1,3)\approx&0.68,\quad \mathrm{R}_{\mathrm{LAD}}(0.1,10)\approx 0.68.\end{aligned}$$

Example 8: Bimodal distribution (mixed normal distribution with different locations). The error density function is

$$f(\epsilon;\mu)=0.5\frac{1}{\sqrt{2\pi}}e^{-(\epsilon-\mu)^2/2}+0.5\frac{1}{\sqrt{2\pi}}e^{-(\epsilon+\mu)^2/2}$$

with $\mu>0$, which implies

$$f(0;\mu)=\frac{e^{-\mu^2}}{\sqrt{2\pi}},\quad g(0;\mu)=\frac{1+e^{-\mu^2}}{2\sqrt{\pi}},\quad \sigma^2=1+\mu^2.$$

Thus,

$$\mathrm{R}_{\mathrm{LowLSR}}(\mu)=3(1+\mu^2)\frac{1+e^{-\mu^2}}{2\sqrt{\pi}},\quad \mathrm{R}_{\mathrm{LAD}}(\mu)=3\left(\frac{1+e^{-\mu^2}}{2\sqrt{\pi}}\right)^2\Bigg/\left(4\left(\frac{e^{-\mu^2}}{\sqrt{2\pi}}\right)^2\right).$$

In particular,

$$\begin{aligned}\mathrm{R}_{\mathrm{LowLSR}}(1)\approx&0.89,\quad \mathrm{R}_{\mathrm{LowLSR}}(3)\approx 2.39,\\ \mathrm{R}_{\mathrm{LAD}}(1)\approx&5.18,\quad \mathrm{R}_{\mathrm{LAD}}(3)\approx 2.46\times 10^7.\end{aligned}$$

Variance Ratio Functions for Three Error Distributions

To further illustrate the trade-off between the sharp-peak and heavy-tailed errors, we consider three of the above examples: Examples 2, 3, and 8.

For the mixed normal distribution, we list in Table 4 the critical value of σ_0 such that the LowLAD (or RLowLAD) and LowLSR estimators have the same variance. When σ_0 is smaller than the critical value, the LowLAD (or RLowLAD) estimator is more efficient than the LowLSR estimator. The overall comparison curve is given in Figure 12. Since LowLAD and RLowLAD have a close relationship, we consider only RLowLAD in the following comparisons. Another critical σ_0 curve comparing RLowLAD and LAD is provided in Figure 13. When σ_0 is larger than the critical value, the RLowLAD estimator is more efficient than the LAD estimator.

For the $t(\nu)$ distribution, the variance ratio function between LowLSR and RLowLAD is shown in Figure 14. We see that the ratio between LowLSR and RLowLAD is greater than 1 for small degrees of freedom (from 3 to 18). As ν increases, the ratio converges to $3/\pi\approx 0.95$. This phenomenon is expected because the T distribution converges to the normal distribution as $\nu\to\infty$, and the variance ratio for the normal distribution is 0.95. Figure 15 shows the ratio between LAD and RLowLAD, where we see that this ratio is greater than 1 for all degrees of freedom. As ν increases, the ratio converges to 1.50. This is expected from the stirling formula $\Gamma(\nu)=\sqrt{2\pi}e^{-\nu}\nu^{\nu-1/2}$, which implies $\mathrm{R}_{\mathrm{LAD}}(f)\to 1.50$ as $\nu\to\infty$.

For the bimodal distribution, the variance ratio function between LowLSR and RLowLAD is given in Figure 16. As μ increases, the ratio becomes smaller and achieves the minimum value 0.89 at point 1.12, and then becomes larger and tends to ∞. The variance ratio function between LAD and RLowLAD is shown in Figure 17. As μ increases, the ratio diverges fast to ∞.

Appendix F. One Key Difference between LS and LAD Methods

For the LS method, the LS estimator and the LowLSR estimator are asymptotically equivalent; while for the LAD method, the asymptotic variances of the LAD estimator and the LowLAD estimator are very different, although their asymptotic biases are the same. To understand this discrepancy, we show that the objective functions of the LS and LowLSR estimation are asymptotically equivalent while those of the LAD and LowLAD estimation are not. Note that the objective function of the LowLSR estimation is equivalent to

$$\begin{aligned}&4\textstyle\sum_{j=1}^k\left(\tilde{Y}_{ij}^{(1)}-\alpha_{i1}d_j-\alpha_{i3}d_j^3\right)^2\\ =&\textstyle\sum_{j=1}^k\left(Y_{i+j}-Y_{i-j}-2\alpha_{i1}d_j-2\alpha_{i3}d_j^3\right)^2\\ =&\textstyle\sum_{j=1}^k\left(Y_{i+j}-Y_{i-j}-(\alpha_{i0}-\alpha_{i0})-\alpha_{i1}\left(d_j-d_{-j}\right)-\alpha_{i2}\left(d_j^2-d_{-j}^2\right)-\alpha_{i3}\left(d_j^3-d_{-j}^3\right)-\alpha_{i4}\left(d_j^4-d_{-j}^4\right)\right)^2\\ =&\textstyle\sum_{j=1}^k\left(Y_{i+j}-\alpha_{i0}-\alpha_{i1}d_j-\alpha_{i2}d_j^2-\alpha_{i3}d_j^3-\alpha_{i4}d_j^4\right)^2\\ +&\textstyle\sum_{j=1}^k\left(Y_{i-j}-\alpha_{i0}-\alpha_{i1}d_{-j}-\alpha_{i2}d_{-j}^2-\alpha_{i3}d_{-j}^3-\alpha_{i4}d_{-j}^4\right)^2\\ -&2\textstyle\sum_{j=1}^k\left(Y_{i+j}-\alpha_{i0}-\alpha_{i1}d_j-\alpha_{i2}d_j^2-\alpha_{i3}d_j^3-\alpha_{i4}d_j^4\right)\left(Y_{i-j}-\alpha_{i0}-\alpha_{i1}d_{-j}-\alpha_{i2}d_{-j}^2-\alpha_{i3}d_{-j}^3-\alpha_{i4}d_{-j}^4\right),\end{aligned}$$

where $d_{-j}=-j/n$. It can be shown that the cross term in the last equality is a higher-order term and is negligible, and the first two terms constitute exactly the objective function of the least squares estimation. More specifically, we can show that

$$\begin{aligned}\hat{m}_{\mathrm{LS}}^{(1)}(x_i)-m^{(1)}(x_i)-\mathrm{Bias}&\doteq(0,1,0,0,0)\left(\textstyle\sum_{j=-k}^k X_jX_j'\right)^{-1}\textstyle\sum_{j=-k}^k X_j\epsilon_{i+j}\\ &=\textstyle\sum_{j=-k}^k D_j\epsilon_{i+j}\end{aligned}$$

with $X_j=\left(1,d_j,d_j^2,d_j^3,d_j^4\right)^T$ and

$$D_j=n\frac{j(75k^4+150k^3-75k+25)-j^3\left(105k^2+105k-35\right)}{k(8k^6+28k^5+14k^4-35k^3-28k^2+7k+6)}=-D_{-j},$$

and

$$\begin{aligned}
&\hat{m}^{(1)}_{\text{LowLSR}}(x_i) - m^{(1)}(x_i) - \text{Bias} \\
&\doteq (1,0)\left[\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}\left(\sum_{j=1}^k X_j X_j'\right)\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}^T\right]^{-1} \\
&\cdot \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}\sum_{j=1}^k X_j \tfrac{\epsilon_{i+j}-\epsilon_{i-j}}{2} \\
&= \sum_{j=1}^k 2D_j \tfrac{\epsilon_{i+j}-\epsilon_{i-j}}{2} = \sum_{j=-k}^k D_j \epsilon_{i+j}.
\end{aligned}$$

The influence functions of $\hat{m}^{(1)}_{\text{LS}}(x_i)$ and $\hat{m}^{(1)}_{\text{LowLSR}}(x_i)$ are exactly the same, so the contribution of the cross term is null.

On the contrary, the objective function of the LowLAD estimator is equivalent to

$$\begin{aligned}
&2\sum_{j=1}^k \left|\tilde{Y}^{(1)}_{ij} - \beta_{i1}d_j - \beta_{i3}d_j^3\right| \\
&= \sum_{j=1}^k \left|Y_{i+j} - Y_{i-j} - (\beta_{i0}-\beta_{i0}) - \beta_{i1}(d_j - d_{-j}) - \beta_{i2}\left(d_j^2 - d_{-j}^2\right) - \beta_{i3}\left(d_j^3 - d_{-j}^3\right) - \beta_{i4}\left(d_j^4 - d_{-j}^4\right)\right| \\
&= \sum_{j=1}^k \left|Y_{i+j} - \beta_{i0} - \beta_{i1}d_j - \beta_{i2}d_j^2 - \beta_{i3}d_j^3 - \beta_{i4}d_j^4\right| + \sum_{j=1}^k \left|Y_{i-j} - \beta_{i0} - \beta_{i1}d_{-j} - \beta_{i2}d_{-j}^2 - \beta_{i3}d_{-j}^3 - \beta_{i4}d_{-j}^4\right| \\
&\quad + \text{extra term},
\end{aligned}$$

where the extra term is equal to

$$\begin{aligned}
&-2\max\left(\text{sign}\left(Y_{i+j} - \beta_{i0} - \beta_{i1}d_j - \beta_{i2}d_j^2 - \beta_{i3}d_j^3 - \beta_{i4}d_j^4\right)\right. \\
&\left.\cdot\text{sign}\left(Y_{i-j} - \beta_{i0} - \beta_{i1}d_{-j} - \beta_{i2}d_{-j}^2 - \beta_{i3}d_{-j}^3 - \beta_{i4}d_{-j}^4\right), 0\right) \\
&\cdot \min\left(\left|Y_{i+j} - \beta_{i0} - \beta_{i1}d_j - \beta_{i2}d_j^2 - \beta_{i3}d_j^3 - \beta_{i4}d_j^4\right|, \left|Y_{i-j} - \beta_{i0} - \beta_{i1}d_{-j} - \beta_{i2}d_{-j}^2 - \beta_{i3}d_{-j}^3 - \beta_{i4}d_{-j}^4\right|\right),
\end{aligned}$$

and cannot be neglected, where $sign(x) = 1$ if $x > 0$ and $sign(x) = -1$ if $x < 0$. More specifically, we can show that

$$\begin{aligned}
&\hat{m}^{(1)}_{\text{LAD}}(x_i) - m^{(1)}(x_i) - \text{Bias} \\
&\doteq (0,1,0,0,0)\left(2f(0)\sum_{j=-k}^k X_j X_j'\right)^{-1}\sum_{j=-k}^k X_j \text{sign}(\epsilon_{i+j}) \\
&= \tfrac{1}{2f(0)}\sum_{j=-k}^k D_j \text{sign}(\epsilon_{i+j}) = \tfrac{1}{f(0)}\sum_{j=1}^k D_j \tfrac{\text{sign}(\epsilon_{i+j})-\text{sign}(\epsilon_{i-j})}{2},
\end{aligned}$$

and

$$\begin{aligned}
&\hat{m}^{(1)}_{\text{LowLAD}}(x_i) - m^{(1)}(x_i) - \text{Bias} \\
&\doteq (1,0)\left[2g(0)\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}\left(\sum_{j=1}^k X_j X_j'\right)\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}^T\right]^{-1} \\
&\cdot \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}\sum_{j=1}^k X_j \text{sign}\left(\tfrac{\epsilon_{i+j}-\epsilon_{i-j}}{2}\right) \\
&= \tfrac{1}{g(0)}\sum_{j=1}^k D_j \text{sign}\left(\tfrac{\epsilon_{i+j}-\epsilon_{i-j}}{2}\right).
\end{aligned}$$

So the contribution of the extra term to the influence function is

$$D_j\left[\frac{\text{sign}\left(\frac{\epsilon_{i+j}-\epsilon_{i-j}}{2}\right)}{g(0)} - \frac{\text{sign}(\epsilon_{i+j})-\text{sign}(\epsilon_{i-j})}{2f(0)}\right] = \frac{D_j}{g(0)}\begin{cases} \text{sign}(\epsilon_{i+j}-\epsilon_{i-j}), & \text{if } \epsilon_{i+j}\epsilon_{i-j} > 0, \\ \left(1 - \frac{g(0)}{f(0)}\right)\text{sign}(\epsilon_{i+j}), & \text{if } \epsilon_{i+j}\epsilon_{i-j} < 0, \end{cases}$$

which is not null, where we neglect the event that $\epsilon_{i+j}\epsilon_{i-j} = 0$ because the probability of such an event is zero.

References

G. Boente and D. Rodriguez. Robust estimators of high order derivatives of regression functions. *Statistics & Probability Letters*, 76(13):1335–1344, 2006.

L.D. Brown and M. Levine. Variance estimation in nonparametric regression via the difference sequence method. *The Annals of Statistics*, 35(5):2219–2232, 2007.

J.L.O. Cabrera. locpol: Kernel local polynomial regression. R package version 0.6-0, 2012. URL `http://mirrors.ustc.edu.cn/CRAN/web/packages/locpol/index.html`.

R. Charnigo, M. Francoeur, M.P. Mengüç, A. Brock, M. Leichter, and C. Srinivasan. Derivatives of scattering profiles: tools for nanoparticle characterization. *Journal of the Optical Society of America*, 24(9):2578–2589, 2007.

R. Charnigo, M. Francoeur, M.P. Mengüç, B. Hall, and C. Srinivasan. Estimating quantitative features of nanoparticles using multiple derivatives of scattering profiles. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 112(8):1369–1382, 2011a.

R. Charnigo, B. Hall, and C. Srinivasan. A generalized C_p criterion for derivative estimation. *Technometrics*, 53(3):238–253, 2011b.

P. Chaudhuri and J.S. Marron. SiZer for exploration of structures in curves. *Journal of the American Statistical Association*, 94(447):807–823, 1999.

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

K. De Brabanter, J. De Brabanter, B. De Moor, and I. Gijbels. Derivative estimation with local polynomial fitting. *Journal of Machine Learning Research*, 14(1):281–301, 2013.

M. Delecroix and A.C. Rosa. Nonparametric estimation of a regression function and its derivatives under an ergodic hypothesis. *Journal of Nonparametric Statistics*, 6(4):367–382, 1996.

N.R. Draper and H. Smith. *Applied Regression Analysis*. Wiley and Sons, New York, 2nd edition, 1981.

J. Fan and I. Gijbels. *Local Polynomial Modelling and Its Applications*. Chapman & Hall, London, 1996.

J. Fan and P. Hall. On curve estimation by minimizing mean absolute deviation and its implications. *The Annals of Statistics*, 22(2):867–885, 1994.

S. Ghosal, A. Sen, and A.W. vander Vaart. Testing monotonicity of regression. *The Annals of Statistics*, 28(4):1054–1082, 2000.

I. Gijbels and A.C. Goderniaux. Data-driven discontinuity detection in derivatives of a regression function. *Communications in Statistics-Theory and Methods*, 33(4):851–871, 2005.

B. Hall. *Nonparametric Estimation of Derivatives with Applications*. PhD thesis, University of Kentucky, Lexington, Kentucky, 2010.

W. Härdle. *Applied Nonparametric Regression*. Cambridge University Press, Cambridge, 1990.

W. Härdle and T. Gasser. Robust non-parametric function fitting. *Journal of the Royal Statistical Society, Series B*, 46(1):42–51, 1984.

W. Härdle and T. Gasser. On robust kernel estimation of derivatives of regression functions. *Scandinavian Journal of Statistics*, 12(3):233–240, 1985.

P.J. Huber and E.M. Ronchetti. *Robust Statistics*. John Wiley & Sons, Inc., 2009.

B. Kai, R. Li, and H. Zou. Local composite quantile regression smoothing: an efficient and sate alternative to local polynomial regression. *Journal of the Royal Statistical Society, Series B*, 72(1):49–69, 2010.

G.D. Knott. *Interpolating Cubic Splines*. Spring, 1st edition, 2000.

R. Koenker. A note on l-estimation for linear models. *Statistics & Probability Letters*, 2(6): 323–325, 1984.

R. Koenker. *Quantile Regression*. Cambridge University Press, New York, 2005.

R. Koenker and G. Bassett. Regression quantiles. *Econometrica*, 46(1):33–50, 1978.

X.R. Li and V.P. Jilkov. Survey of maneuvering target tracking. Part I: Dynamic models. *IEEE Transations on Aerospace and Electronic Systems*, 39(4):1333–1364, 2003.

X.R. Li and V.P. Jilkov. Survey of maneuvering target tracking. Part II: Motion models of ballistic and space targets. *IEEE Transations on Aerospace and Electronic Systems*, 46 (1):96–119, 2010.

Y. Liu and K. De Brabanter. Derivative estimation in random design. 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada., 2018.

I. Matyasovszky. Detecting abrupt climate changes on different time scales. *Theoretical and Applied Climatology*, 105(3-4):445–454, 2011.

H.G. Müller. *Nonparametric Regression Analysis of Longitudinal Data*. Springer, New York, 1988.

H.G. Müller, U. Stadtmüller, and T. Schmitt. Bandwidth choice and confidence intervals for derivatives of noisy data. *Biometrika*, 74(4):743–749, 1987.

J. Newell, J. Einbeck, N. Madden, and K. McMillan. Model free endurance markers based on the second derivative of blood lactate curves. In *Proceedings of the 20th International Workshop on Statistical Modelling*, pages 357–364, Sydney, 2005.

F. Osorio. L1pack: Routines for $L1$ estimation. R package version 0.3, 2015. URL `http://www.ies.ucv.cl/l1pack/`.

A. Pakes and D. Pollard. Simulation and the asymptotics of optimization estimators. *Econometrica*, 57(5):1027–1057, 1989.

C. Park and K.H Kang. SiZer analysis for the comparison of regression curves. *Computational Statistics & Data Analysis*, 52(8):3954–3970, 2008.

J. Porter and P. Yu. Regression discontinuity designs with unknown discontinuity points: Testing and estimation. *Journal of Econometrics*, 189(1):132–147, 2015.

J. Ramsay and B. Ripley. pspline: Penalized smoothing splines. R package version 1.0-16, 2013. URL `http://mirrors.ustc.edu.cn/CRAN/web/packages/pspline/index.html`.

J.O. Ramsay and B.W. Silverman. *Applied Functional Data Analysis: Methods and Case Studies*. Springer, New York, 2002.

D. Ruppert and M.P. Wand. Multivariate locally weighted least squares regression. *The Annals of Statistics*, 22(3):1346–1370, 1994.

C.J. Stone. Additive regression and other nonparametric models. *The Annals of Statistics*, 13(2):689–705, 1985.

P.S. Swain, K. Stevenson, A. Leary, L.F. Montano-Gutierrez, I.B.N. Clark, J. Vogel, and T. Pilizota. Inferring time derivatives including cell growth rates using gaussian process. *Nature Communications*, 7: 13766, 2016.

G. Wahba and Y. Wang. When is the optimal regularization parameter insensitive to the choice of the loss function? *Communications in Statistics-Theory and Methods*, 19(5): 1685–1700, 1990.

F.T. Wang and D.W. Scott. The L_1 method for robust nonparametric regression. *Journal of the American Statistical Association*, 89(425):65–76, 1994.

W.W. Wang and L. Lin. Derivative estimation based on difference sequence via locally weighted least squares regression. *Journal of Machine Learning Research*, 16:2617–2641, 2015.

W.W. Wang and P. Yu. Asymptotically optimal differenced estimators of error variance in nonparametric regression. *Computational Statistics & Data Analysis*, 105:125–143, 2017.

A.H. Welsh. Robust estimation of smooth regression and spread functions and their derivatives. *Statistica Sinica*, 6(2):347–366, 1996.

Z. Zhao and Z. Xiao. Efficient regression via optimally combining quantile information. *Econometric Theory*, 30(6):1272–1314, 2014.

			LowLAD	**RLowLAD**	**LowLSR**	**LAD**	**LowLAD**	**RLowLAD**	**LowLSR**	**LAD**
A	f	σ	$n = 100$	$n = 100$	$n = 100$	$n = 100$	$n = 500$	$n = 500$	$n = 500$	$n = 500$
						$\sigma_0 = 1$				
1	0.5	0.1	0.41(0.10)	0.34(0.10)	0.83(0.32)	0.37(0.10)	0.18(0.04)	0.15(0.04)	0.40(0.12)	0.17(0.05)
		0.5	1.79(0.38)	1.49(0.43)	1.51(0.45)	1.76(0.46)	0.82(0.15)	0.67(0.19)	0.70(0.19)	0.81(0.22)
	1	0.1	0.41(0.10)	0.34(0.10)	0.84(0.32)	0.37(0.10)	0.19(0.04)	0.15(0.04)	0.40(0.12)	0.17(0.05)
		0.5	1.78(0.37)	1.49(0.41)	1.51(0.43)	1.74(0.45)	0.82(0.15)	0.68(0.18)	0.70(0.19)	0.81(0.21)
10	0.5	0.1	0.41(0.09)	0.35(0.10)	0.85(0.33)	0.38(0.10)	0.18(0.03)	0.15(0.04)	0.39(0.12)	0.17(0.04)
		0.5	1.78(0.36)	1.47(0.40)	1.48(0.42)	1.75(0.45)	0.79(0.15)	0.64(0.18)	0.66(0.18)	0.75(0.21)
	1	0.1	0.45(0.09)	0.38(0.10)	0.85(0.31)	0.41(0.10)	0.24(0.04)	0.22(0.04)	0.43(0.12)	0.23(0.05)
		0.5	1.79(0.36)	1.50(0.40)	1.53(0.43)	1.76(0.45)	0.82(0.15)	0.67(0.18)	0.68(0.18)	0.80(0.21)
						$\sigma_0 = 10$				
1	0.5	0.1	0.46(0.16)	0.39(0.14)	7.79(3.15)	0.39(0.11)	0.19(0.04)	0.15(0.04)	3.78(1.17)	0.17(0.04)
		0.5	2.11(0.51)	1.76(0.53)	7.88(3.04)	1.89(0.50)	0.93(0.19)	0.76(0.22)	3.84(1.17)	0.85(0.22)
	1	0.1	0.45(0.14)	0.38(0.14)	7.93(3.33)	0.39(0.11)	0.19(0.04)	0.15(0.04)	3.77(1.13)	0.17(0.04)
		0.5	2.15(0.49)	1.81(0.53)	8.00(3.16)	1.93(0.50)	0.92(0.18)	0.75(0.21)	3.81(1.13)	0.83(0.22)
10	0.5	0.1	0.46(0.14)	0.38(0.13)	8.03(3.38)	0.39(0.11)	0.18(0.04)	0.15(0.05)	3.66(1.26)	0.17(0.04)
		0.5	2,12(0.50)	1.77(0.52)	8.05(3.24)	1.89(0.50)	0.94(0.17)	0.77(0.20)	3.94(1.16)	0.85(0.23)
	1	0.1	0.49(0.13)	0.42(0.13)	7.95(3.09)	0.42(0.10)	0.25(0.04)	0.22(0.04)	3.84(1.13)	0.23(0.04)
		0.5	2.17(0.52)	1.82(0.54)	8.07(3.16)	1.92(0.51)	0.94(0.18)	0.77(0.22)	3.80(1.12)	0.86(0.23)

Table 3: Simulation comparison among LowLAD, RLowLAD, LowLSR and LAD.

S. Zhou and D.A. Wolfe. On derivative estimation in spline regression. *Statistica Sinica*, 10 (1):93–108, 2000.

H. Zou and M. Yuan. Composite quantile regression and the oracle model selection theory. *The Annals of Statistics*, 36(3):1108–1126, 2008.

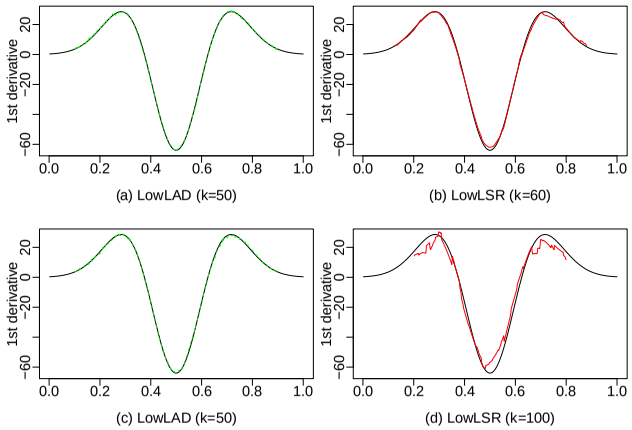


Figure 5: (a)-(d) The true second-order derivative function (bold line), LowLAD (green line) and LowLSR estimators (red line) based on the simulated data set from Figure 3.

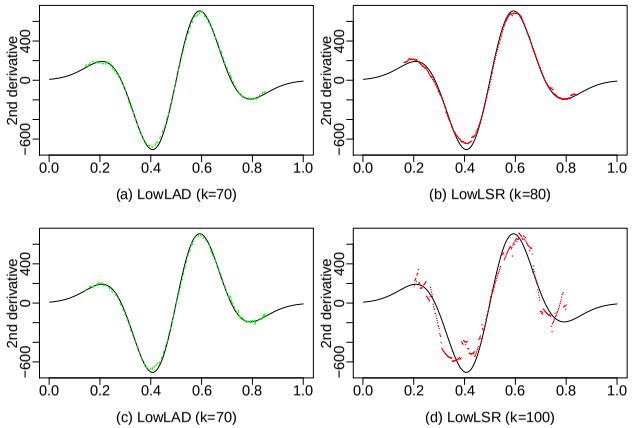


Figure 6: (a)-(d) The true second-order derivative function (bold line), LowLAD (green line) and LowLSR estimators (red line) based on the simulated data set from Figure 4.

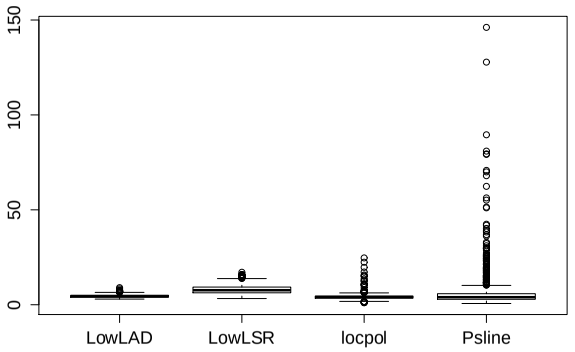


Figure 7: Boxplot of four estimators for the function m_4 with $\epsilon \sim 95\%N(0, 0.1^2) + 5\%N(0, 1^2)$.

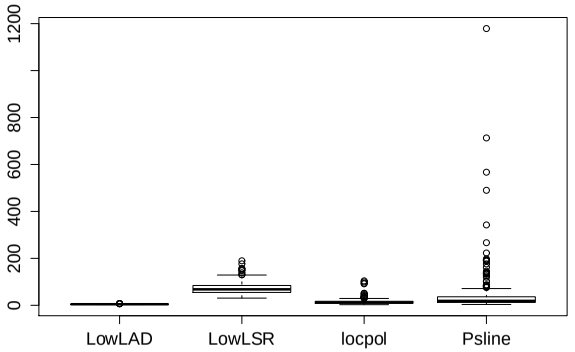


Figure 8: Boxplot of four estimators for the function m_4 with $\epsilon \sim 95\%N(0, 0.1^2) + 5\%N(0, 10^2)$.

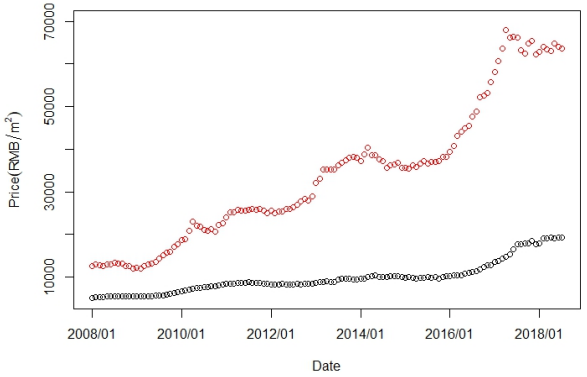


Figure 9: Black and green points denote the house prices in Beijing and Jinan, respectively.

α	0.001	0.002	0.003	0.004	0.005	0.006	0.007	0.008	0.009
σ_0^{LowLAD}	24.04	17.10	14.04	12.22	10.99	10.09	9.38	8.82	8.36
$\sigma_0^{\text{RLowLAD}}$	7.19	5.27	4.44	3.95	3.62	3.38	3.20	3.05	2.93
α	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
σ_0^{LowLAD}	7.96	5.88	4.99	4.47	4.12	3.87	3.69	3.54	3.42
$\sigma_0^{\text{RLowLAD}}$	2.83	2.29	2.06	1.92	1.83	1.76	1.71	1.67	1.64
α	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50
σ_0^{LowLAD}	3.32	3.01	2.86	2.79	2.77	2.78	2.83	2.92	3.05
$\sigma_0^{\text{RLowLAD}}$	1.61	1.52	1.48	1.45	1.43	1.42	1.42	1.42	1.43

Table 4: The critical values of σ_0 that equate the variances of the LowLAD (RLowLAD) and LowLSR derivative estimators with different contaminations.

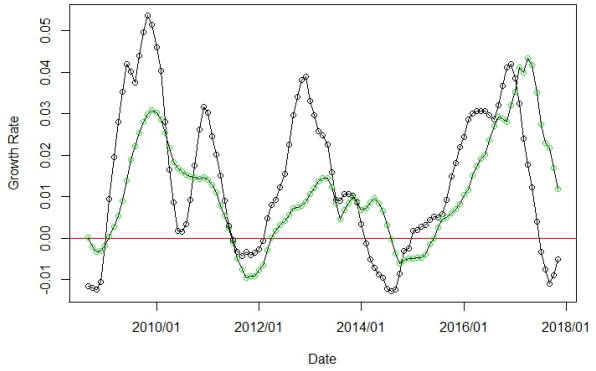


Figure 10: Black and green curves denote the relative growth rates for Beijing and Jinan, respectively. Relative growth rate is defined as $RLowLAD/Price$.

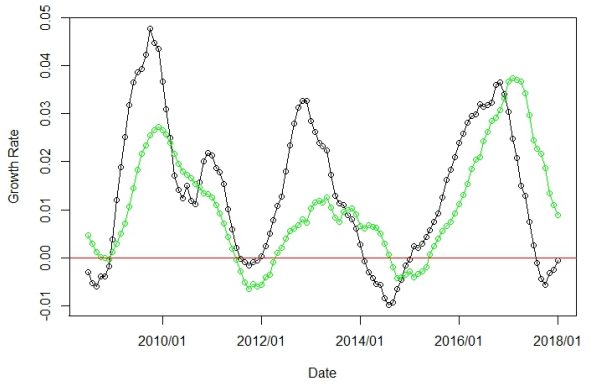


Figure 11: Black and green curves denote the relative growth rates based on the lower-order RLowLAD estimator.

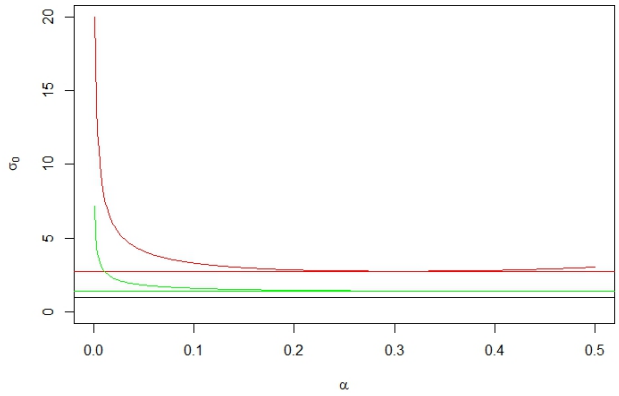


Figure 12: The red line is the critical σ_0 curve between LowLSR and LowLAD with $\epsilon_i \sim (1-\alpha)N(0,1) + \alpha N(0,\sigma_0^2)$, and the red horizontal line is $\sigma_0 = 2.77$; The green line is the critical σ_0 curve between LowLSR and RLowLAD, and the green horizontal line is $\sigma_0 = 1.42$; the black line is $\sigma_0 = 1$.

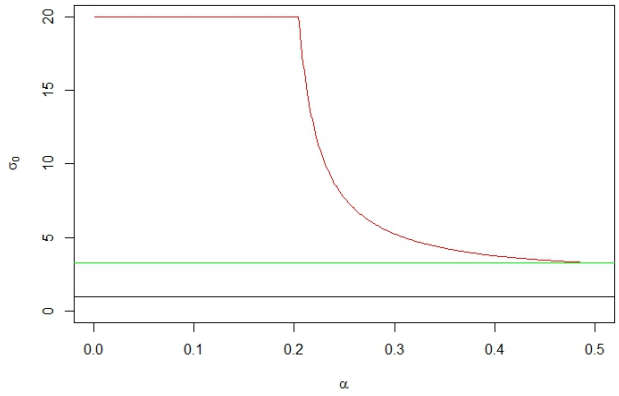


Figure 13: The red line is the critical σ_0 curve between RLowLAD and LAD with the same error distribution as in Figure 12, where the ratio larger than 20 is truncated at 20, the green horizontal line is $\sigma_0 = 3.28$, and the black line is $\sigma_0 = 1$.

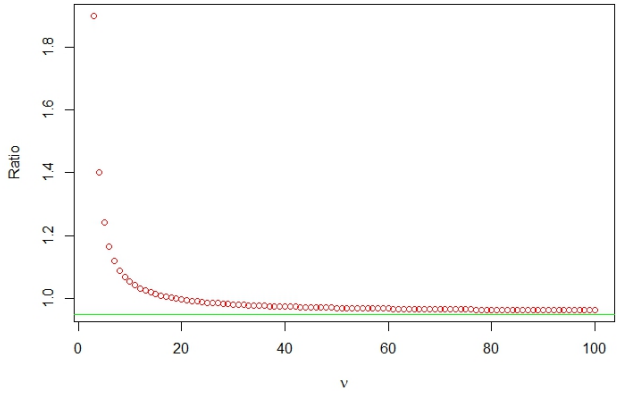


Figure 14: The red point-curve is the variance ratio function between LowLSR and RLowLAD for $t(\nu)$ with different ν's; the green horizontal line is $Ratio = 0.95$.

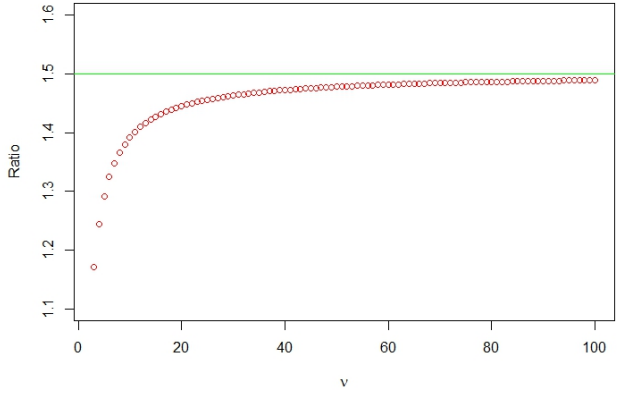


Figure 15: The red point-curve is the variance ratio function between LAD and RLowLAD estimators for (ν) with different ν's; and the green horizontal line is $Ratio = 1.50$.

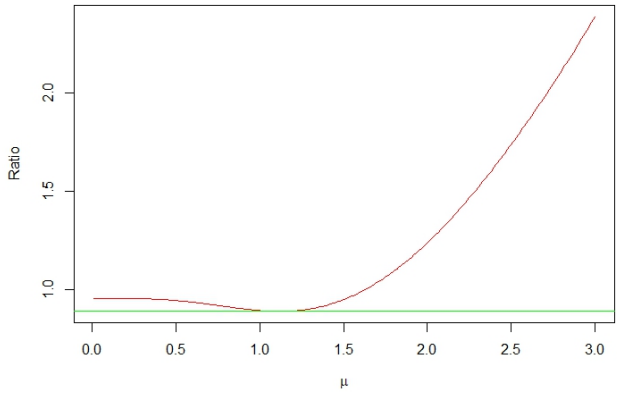


Figure 16: The red curve is the variance ratio function between LowLSR and RLowLAD for $\epsilon_i \sim 0.5N(\mu, 1) + 0.5N(-\mu, 1)$ with different μ's; the green horizontal line is $Ratio = 0.89$.

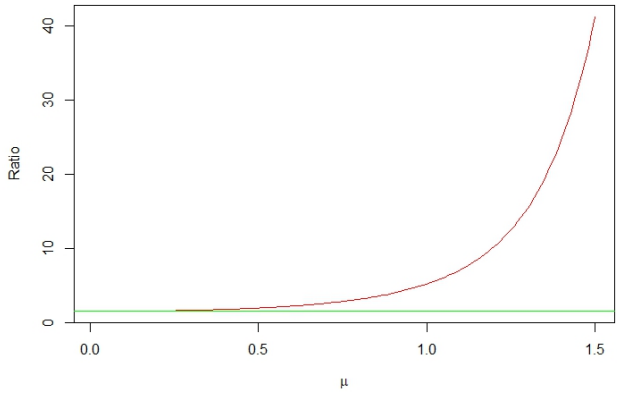


Figure 17: The red curve is the variance ratio function between RLowLAD and LAD for the same error distribution as in Figure 16; the green horizontal line is $Ratio = 1.50$.

Journal of Machine Learning Research 20 (2019) 1-42 Submitted 7/17; Revised 12/18; Published 3/19

The Sup-norm Perturbation of HOSVD and Low Rank Tensor Denoising

Dong Xia * madxia@ust.hk
Department of Mathematics
Hong Kong University of Science and Technology
Hong Kong SAR, China

Fan Zhou † fzhou40@math.gatech.edu
School of Mathematics
Georgia Institute of Technology
Atlanta, GA 30332, USA

Editor: Ryota Tomioka

Abstract

The higher order singular value decomposition (HOSVD) of tensors is a generalization of matrix SVD. The perturbation analysis of HOSVD under random noise is more delicate than its matrix counterpart. Recently, polynomial time algorithms have been proposed where statistically optimal estimates of the singular subspaces and the low rank tensors are attainable in the Euclidean norm. In this article, we analyze the sup-norm perturbation bounds of HOSVD and introduce estimators of the singular subspaces with sharp deviation bounds in the sup-norm. We also investigate a low rank tensor denoising estimator and demonstrate its fast convergence rate with respect to the entry-wise errors. The sup-norm perturbation bounds reveal unconventional phase transitions for statistical learning applications such as the exact clustering in high dimensional Gaussian mixture model and the exact support recovery in sub-tensor localizations. In addition, the bounds established for HOSVD also elaborate the one-sided sup-norm perturbation bounds for the singular subspaces of unbalanced (or fat) matrices.

Keywords: HOSVD, Entry-wise perturbation, Gaussian noise, High dimensional clustering.

1. Introduction

A tensor is a multi-array of more than 2 dimensions, which can be viewed as a higher order generalization of matrices. Data of tensor types has been widely available in many fields, such as image and video processing (see Liu et al. (2013), Westin et al. (2002), Hildebrand and Rüegsegger (1997), Li and Li (2010), Vasilescu and Terzopoulos (2002)); latent variable modelling (see Anandkumar et al. (2014), Cichocki et al. (2015), Chaganty and Liang (2013)); genomic signal processing (Omberg et al. (2007), Muralidhara et al. (2011) and Ponnapalli et al. (2011)) and references therein. It is demanding to handle these datasets in order to take the most advantages of the tensor structures. The task is challenging due to the highly non-convexity of tensor related optimization problems. For instance, computing the tensor operator norm is generally NP-hard (see, e.g., Hillar and Lim (2013)) while it can be implemented fast for matrices.

The higher order singular value decomposition (HOSVD) is one machinery to deal with tensors which generalizes the matrix SVD to higher order tensors, see Zheng and Tomioka (2015), De Lathauwer et al. (2000b), Bergqvist and Larsson (2010), Chen and Saad (2009) and Kolda and Bader (2009). The conceptual simplicity and computational efficiency make HOSVD popular. It has been successfully applied to various statistical learning tasks, for instance, face recognition (see Vasilescu and Terzopoulos (2002)), genomic signal processing (see Muralidhara et al. (2011)) and more examples in a survey paper (Acar and Yener (2009)). Basically, the HOSVD unfolds a higher order tensor into matrices and treat it with standard matrix techniques to obtain the principal singular subspaces in each dimension (see more details in Section 2). Although the HOSVD shows appealing effectiveness, there are several fundamental theoretic mysteries yet to be uncovered.

One particularly important question is related to the perturbation of HOSVD when a low rank tensor is contaminated by stochastic noise. The difficulty comes from both methodological and theoretical aspects. The computation of HOSVD is essentially reduced to matrix SVD which can be implemented efficiently. This naive estimator is actually statistically sub-optimal. It is well-known that further power iterations can ameliorate the naive spectral initializations and thus deliver statistically optimal estimators, see more details in Richard and Montanari (2014), Zhang and Xia (2018), Hopkins et al. (2015), Liu et al. (2017) and references therein. Another intriguing phenomenon is on the phase transitions of the signal-to-noise ratio (SNR). Actually, the SNR exhibits distinct computational and statistical phase transitions, while the differences do not exist for matrix SVD. In particular, there is a gap on SNR between statistical optimality and computational optimality for HOSVD, see Zhang and Xia (2018). For introductory simplicity [1], we focus on the third-order tensors. Suppose that an unknown tensor $\mathbf{A} \in \mathbb{R}^{d\times d\times d}$ with multilinear ranks (r, r, r) is planted in a noisy observation $\mathbf{Y}$ with

$$\mathbf{Y} = \mathbf{A} + \mathbf{Z} \in \mathbb{R}^{d\times d\times d}. \tag{1}$$

The noise tensor $\mathbf{Z}$ has i.i.d. entries with $Z(i,j,k) \sim \mathcal{N}(0,\sigma^2)$ for $i,j,k \in [d]$ and noise variance $\sigma^2 > 0$. Here, we denote by $[d] := \{1,\dots,d\}$. The signal strength $\underline{\Lambda}(\mathbf{A})$ is defined as the smallest nonzero singular values of the matrices unfolded from $\mathbf{A}$ (see definitions in Section 3.3). Let $\mathbf{U},\mathbf{V},\mathbf{W} \in \mathbb{R}^{d\times r}$ denote the singular vectors of $\mathbf{A}$ in the corresponding dimensions. It was proved (see Zheng and Tomioka (2015), Zhang and Xia (2018) and Liu et al. (2017)) that if the signal strength $\underline{\Lambda}(\mathbf{A}) \geq D_1\sigma d^{3/4}$ for a large enough constant

∗. Most of the work was done when the first author was a Visiting Assistant Professor in Department of Statistics at University of Wisconsin-Madison and later a Post-doctoral Research Scientist in Department of Statistics at Columbia University.

†. Supported in part by NSF Grants DMS-1509739 and CCF-1523768.

1. More general results where $\mathbf{A}$ is $d_1 \times d_2 \times d_3$ with multilinear ranks (r_1, r_2, r_3) can be found in Section 3. The results of this article can be easily generalized to higher order tensors.

©2019 Dong Xia and Fan Zhou.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-397.html.

$D_1 > 0$, the following bound holds

$$r^{-1/2}\max\Big\{\big\|\hat{\mathbf{U}}\hat{\mathbf{U}}^\top - \mathbf{U}\mathbf{U}^\top\big\|_{\ell_2}, \big\|\hat{\mathbf{V}}\hat{\mathbf{V}}^\top - \mathbf{V}\mathbf{V}^\top\big\|_{\ell_2}, \big\|\hat{\mathbf{W}}\hat{\mathbf{W}}^\top - \mathbf{W}\mathbf{W}^\top\big\|_{\ell_2}\Big\} = O_p\Big(\frac{\sigma d^{1/2}}{\underline{\Lambda}(\mathbf{A})} + \frac{\sigma d^{3/2}}{\underline{\Lambda}^2(\mathbf{A})}\Big),$$

where $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{W}}$ represent the naive SVD obtained from noisy tensor $\mathbf{Y}$ and $\|\cdot\|_{\ell_2}$ denotes the Euclidean norm. Power iterations (also called higher order orthogonal iterations, see De Lathauwer et al. (2000a)) can improve the estimate (denoted by $\widetilde{\mathbf{U}}, \widetilde{\mathbf{V}}, \widetilde{\mathbf{W}}$) to

$$r^{-1/2}\max\Big\{\big\|\widetilde{\mathbf{U}}\widetilde{\mathbf{U}}^\top - \mathbf{U}\mathbf{U}^\top\big\|_{\ell_2}, \big\|\widetilde{\mathbf{V}}\widetilde{\mathbf{V}}^\top - \mathbf{V}\mathbf{V}^\top\big\|_{\ell_2}, \big\|\widetilde{\mathbf{W}}\widetilde{\mathbf{W}}^\top - \mathbf{W}\mathbf{W}^\top\big\|_{\ell_2}\Big\} = O_p\Big(\frac{\sigma d^{1/2}}{\underline{\Lambda}(\mathbf{A})}\Big), \quad (2)$$

which is statistically optimal (see Zhang and Xia (2018)). Moreover, it is demonstrated in Zhang and Xia (2018), built on a hardness conjecture of the hyper-graphical planted clique detection problem, that if $\underline{\Lambda}(\mathbf{A}) = o\big(\sigma d^{3/4}\big)$, then all polynomial time algorithms deliver trivial estimates of $\mathbf{U}, \mathbf{V}, \mathbf{W}$ in general.

One focus of this article is on estimating the linear forms of tensor singular vectors in model (1). More specifically, let $\mathbf{U} = \big(\mathbf{u}_1, \ldots, \mathbf{u}_r\big) \in \mathbb{R}^{d\times r}$ be $\mathbf{A}$'s singular vectors in certain mode, our goal is to estimate $\langle \mathbf{u}_j, \mathbf{x}\rangle$ for fixed $\mathbf{x} \in \mathbb{R}^d$ and $j = 1, \ldots, r$. Through choosing $\mathbf{x}$ all over the canonical basis vectors in $\mathbb{R}^d$, we end up with an estimate of $\mathbf{u}_j$ whose component-wise perturbation bound can be attained. Unlike the ℓ_2-norm perturbation bound, the ℓ_∞ bound can characterize the entry-wise sign consistency and entry-wise significance (i.e. entry-wise magnitude) of singular vectors. The component-wise signs of singular vectors are critical in numerous applications such as community detection (see Florescu and Perkins (2015), Newman (2004), Mitra (2009) and Jin (2015)). The entry-wise significance is advantageous in sub-matrix localizations, see Cai et al. (2015), Ma and Wu (2015) and references therein. In Section 4, we show that the sup-norm perturbation bounds reveal unconventional phase transitions for the exact clustering in high dimensional Gaussian mixture model. Put it simply, algorithms based on the sup-norm bounds require weaker SNR conditions than algorithms driven by the ℓ_2-norm bounds to guarantee exact clustering. Furthermore, it enables us to construct a low rank denoising estimator of $\mathbf{A}$ so that entry-wise denoising is fulfilled. To the best of our knowledge, ours is the first result concerning the low rank tensor denoising with sharp entry-wise deviation bounds. In Section 4, we show that a simple algorithm based on the ℓ_∞ bounds can exactly recover the supports for sub-tensor localizations (see Remark 13).

To better highlight our contributions, suppose that $\mathbf{A}$ is an orthogonally decomposable third order tensor with (in particular, the CP decomposition of orthogonally decomposable tensors)

$$\mathbf{A} = \sum_{k=1}^{r} \lambda_k\big(\mathbf{u}_k \otimes \mathbf{v}_k \otimes \mathbf{w}_k\big), \quad \lambda_1 \geq \ldots \geq \lambda_r > 0 \quad (3)$$

where $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_r), \mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_r)$ and $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_r)$ are $d \times r$ matrices containing orthonormal columns. The k-th eigengap is written as $\bar{g}_k\big(\mathcal{M}_1(\mathbf{A})\big) = \bar{g}_k\big(\mathcal{M}_2(\mathbf{A})\big) = \bar{g}_k\big(\mathcal{M}_3(\mathbf{A})\big) = \min\big(\lambda_{k-1} - \lambda_k, \lambda_k - \lambda_{k+1}\big)$ where $\mathcal{M}_j(\mathbf{A})$ represents the matrices from unfoldings of $\mathbf{A}$ (see Section 2). We preset $\lambda_0 = +\infty$ and $\lambda_{r+1} = 0$ for notational consistency. We show that, if $\bar{g}_k\big(\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A})\big) \geq D_1\big(\sigma\lambda_1 d^{1/2} + \sigma^2 d^{3/2}\big)$ for a large enough absolute constant $D_1 > 0$, then the following bound holds for any $\mathbf{x} \in \mathbb{R}^d$,

$$\Big|\langle \hat{\mathbf{u}}_k, \mathbf{x}\rangle - (1+b_k)^{1/2}\langle \mathbf{u}_k, \mathbf{x}\rangle\Big| = O_p\Big(\|\mathbf{x}\|_{\ell_2}\frac{\lambda_1\sigma + d\sigma^2}{\bar{g}_k\big(\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A})\big)}\Big) = O_p\Big(\frac{\|\mathbf{x}\|_{\ell_2}}{d^{1/2}}\Big). \quad (4)$$

where $b_k \in [-1/2, 0]$ is a constant which does not depend on $\mathbf{x}$. The $d \times r$ matrix $\hat{\mathbf{U}} = (\hat{\mathbf{u}}_1, \cdots, \hat{\mathbf{u}}_k)$ represent the empirical left singular vectors of mode-1 unfolding of $\mathbf{Y}$ satisfying model (1).

In the special case that $r = 1$ (rank one spiked tensor PCA model, see Richard and Montanari (2014)) such that $\underline{\Lambda}(\mathbf{A}) = \bar{g}_1\big(\mathcal{M}_1(\mathbf{A})\big) = \lambda_1$, we get from (4) that

$$\Big|\langle \hat{\mathbf{u}}_1, \mathbf{x}\rangle - (1+b_1)^{1/2}\langle \mathbf{u}_1, \mathbf{x}\rangle\Big| = O_p\Big(\frac{\sigma}{\underline{\Lambda}(\mathbf{A})} + \frac{\sigma^2 d}{\underline{\Lambda}^2(\mathbf{A})}\Big)\|\mathbf{x}\|_{\ell_2}. \quad (5)$$

By taking $\mathbf{x}$ over the canonical basis vectors in $\mathbb{R}^d$, the bounds in (5) imply that

$$\big\|\hat{\mathbf{u}}_1 - (1+b_1)^{1/2}\mathbf{u}_1\big\|_{\ell_\infty} = O_p\Big(\Big(\frac{\log d}{d}\Big)^{1/2}\Big) \quad (6)$$

under the eigen-gap condition $\underline{\Lambda}(\mathbf{A}) = \lambda_1 \gg \sigma d^{3/4}$. It is the standard requirement in tensor PCA. [2] Based on (6), we propose a low rank tensor estimator (denoted by $\hat{\mathbf{A}}$) under the same SNR requirements such that

$$\|\hat{\mathbf{A}} - \mathbf{A}\|_{\ell_\infty} = O_p\Big(\Big(\frac{\sigma^2 d}{\lambda_1} + \sigma\Big)\big(\|\mathbf{u}_1\|_{\ell_\infty}\|\mathbf{v}_1\|_{\ell_\infty} + \|\mathbf{u}_1\|_{\ell_\infty}\|\mathbf{w}_1\|_{\ell_\infty} + \|\mathbf{v}_1\|_{\ell_\infty}\|\mathbf{w}_1\|_{\ell_\infty}\big)\Big). \quad (7)$$

Equation (7) shows that the entry-wise denoising bound of the novel estimator $\hat{\mathbf{A}}$ is determined by the coherences of the singular vectors $\mathbf{u}_1, \mathbf{v}_1$ and $\mathbf{w}_1$. In particular, if $\mathbf{u}_1, \mathbf{v}_1, \mathbf{w}_1$ are incoherent so that $\max\{\|\mathbf{u}_1\|_{\ell_\infty}, \|\mathbf{v}_1\|_{\ell_\infty}, \|\mathbf{w}_1\|_{\infty}\} = O\big(\frac{1}{\sqrt{d}}\big)$, then equation (7) implies that

$$\|\hat{\mathbf{A}} - \mathbf{A}\|_{\ell_\infty} = O_p\Big(\frac{\sigma^2}{\lambda_1} + \frac{\sigma}{d}\Big).$$

Our main contribution is on the theoretical front. The HOSVD is essentially the standard SVD computed on an unbalanced matrix where the column size is much larger than the row size. The perturbation tools, such as Wedin's $\sin\Theta$ theorem (Wedin (1972)), characterize the ℓ_2 bounds through the larger dimension, even when the left singular space lies in a low dimensional space. At the high level, the HOSVD is connected to the one-sided spectral analysis (see, e.g.,Zheng and Tomioka (2015), Wang (2015), Cai and Zhang (2016) and references therein) which provide sharp perturbation bounds in ℓ_2-norm. There are recent bounds (see Fan et al. (2016) and Cape et al. (2017)) in ℓ_∞-norm developed under

2. We shall point out that a similar result on matrix SVD has appeared in Koltchinskii and Xia (2016) which is sub-optimal for tensors or unbalanced matrices. Indeed, the result in Koltchinskii and Xia (2016) is established under the eigengap condition $\lambda_1 \geq D_1\sigma d$.

additional constraint (incoherent singular subspaces) and structural noise (sparse noise). To obtain a sharp ℓ_∞-norm bound, we borrow the instruments invented by Koltchinskii and Lounici (2016) and extensively applied in Koltchinskii and Xia (2016). Our framework starts from a second order method of estimating the singular subspaces, which improves the eigengap condition than the first order method. Similar techniques have been proposed for tensor completion (Xia and Yuan (2019)) and tensor PCA (Zheng and Tomioka (2015) and Liu et al. (2017)). The success of this seemingly natural treatment hinges upon delicate dealing with the correlations among higher order terms. We benefit from these ℓ_∞-norm perturbation bounds by proposing a low rank estimator for tensor denoising where the entry-wise deviation error is guaranteed by the tensor incoherence conditions.

We organize our paper as follows. Tensor notations and preliminaries on HOSVD are explained in Section 2. Our main theoretical contributions are presented in Section 3 which includes the ℓ_∞-norm bound of the singular subspace perturbation and the entry-wise accuracy of a low rank tensor denoising estimator. In Section 4, we apply our theoretical results on applications including high dimensional clustering and sub-tensor localizations to manifest the advantages of utilizing ℓ_∞ bounds, where algorithms driven by the ℓ_∞-norm bounds are designed. Results of numerical experiments are displayed in Section 4.3. The proofs are provided in Section 6.

2. Preliminaries on Tensor and HOSVD

2.1. Notations

We first review some notations that will be used through the paper. We use boldfaced upper-case letters to denote tensors or matrices, and use the same letter in normal font with indices to denote its entries. We use boldfaced lower-case letters to represent vectors, and the same letter in normal font with indices to represent its entries. For notationally simplicity, our main context is focused on third-order tensors, while our results can be easily generalized to higher order tensors.

Given a third-order tensor $\mathbf{A} \in \mathbb{R}^{d_1\times d_2\times d_3}$, define a linear mapping $\mathcal{M}_1 : \mathbb{R}^{d_1\times d_2\times d_3} \mapsto \mathbb{R}^{d_1\times(d_2d_3)}$ such that

$$\mathcal{M}_1(\mathbf{A})\big(i_1,(i_2-1)d_3+i_3\big) = A(i_1,i_2,i_3), \quad i_1\in[d_1], i_2\in[d_3], i_3\in[d_3]$$

which is conventionally called the unfolding (or matricization) of tensor $\mathbf{A}$. It is also called the mode-1 unfolding of $\mathbf{A}$. The columns of matrix $\mathcal{M}_1(\mathbf{A})$ are called the mode-1 fibers of $\mathbf{A}$. The corresponding matricizations $\mathcal{M}_2(\mathbf{A})$ and $\mathcal{M}_3(\mathbf{A})$ can be defined in a similar fashion. The multilinear ranks of $\mathbf{A}$ are then defined by:

$$r_1(\mathbf{A}) := \mathrm{rank}\big(\mathcal{M}_1(\mathbf{A})\big), \quad r_2(\mathbf{A}) := \mathrm{rank}\big(\mathcal{M}_2(\mathbf{A})\big), \quad r_3(\mathbf{A}) := \mathrm{rank}\big(\mathcal{M}_3(\mathbf{A})\big)$$

Note that $r_1(\mathbf{A}), r_2(\mathbf{A}), r_3(\mathbf{A})$ are unnecessarily equal with each other in general. We write $\mathbf{r}(\mathbf{A}) := \big(r_1(\mathbf{A}), r_2(\mathbf{A}), r_3(\mathbf{A})\big)$ which are also called the Tucker ranks of $\mathbf{A}$.

The marginal product $\times_1 : \mathbb{R}^{r_1\times r_2\times r_3}\times\mathbb{R}^{d_1\times r_1}\mapsto\mathbb{R}^{d_1\times r_2\times r_3}$ is given by

$$\mathbf{C}\times_1\mathbf{U} = \Big(\sum_{j_1=1}^{r_1} C(j_1,j_2,j_3)U(i_1,j_1)\Big)_{i_1\in[d_1],j_2\in[r_2],j_3\in[r_3]},$$

and $\times_2$ and $\times_3$ are defined similarly. Therefore, we write the multilinear product of tensors $\mathbf{C}\in\mathbb{R}^{r_1\times r_2\times r_3}, \mathbf{U}\in\mathbb{R}^{d_1\times r_1}, \mathbf{V}\in\mathbb{R}^{d_2\times r_2}$ and $\mathbf{W}\in\mathbb{R}^{d_3\times r_3}$ as

$$\mathbf{C}\cdot(\mathbf{U},\mathbf{V},\mathbf{W}) = \mathbf{C}\times_1\mathbf{U}\times_2\mathbf{V}\times_3\mathbf{W}\in\mathbb{R}^{d_1\times d_2\times d_3}.$$

We use $\|\cdot\|$ to denote the operator norm of matrices and $\|\cdot\|_{\ell_2}$ and $\|\cdot\|_{\ell_\infty}$ to denote ℓ_2 and ℓ_∞ norms of vectors, or vectorized matrices and tensors.

2.2. HOSVD and Eigengaps

For a tensor $\mathbf{A}\in\mathbb{R}^{d_1\times d_2\times d_3}$ with multilinear ranks $\mathbf{r}(\mathbf{A}) = \big(r_1(\mathbf{A}), r_2(\mathbf{A}), r_3(\mathbf{A})\big)$, let $\mathbf{U}\in\mathbb{R}^{d_1\times r_1(\mathbf{A})}, \mathbf{V}\in\mathbb{R}^{d_2\times r_2(\mathbf{A})}$ and $\mathbf{W}\in\mathbb{R}^{d_3\times r_3(\mathbf{A})}$ be the left singular vectors of $\mathcal{M}_1(\mathbf{A}), \mathcal{M}_2(\mathbf{A})$ and $\mathcal{M}_3(\mathbf{A})$ respectively, which can be computed efficiently via matricization followed by thin singular value decomposition. The higher order singular value decomposition (HOSVD) refers to the decomposition

$$\mathbf{A} = \mathbf{C}\times_1\mathbf{U}\times_2\mathbf{V}\times_3\mathbf{W} \tag{8}$$

where the $r_1(\mathbf{A})\times r_2(\mathbf{A})\times r_3(\mathbf{A})$ core tensor $\mathbf{C}$ is obtained by $\mathbf{C} := \mathbf{A}\times_1\mathbf{U}^\top\times_2\mathbf{V}^\top\times_3\mathbf{W}^\top$.

Suppose that a noisy version of $\mathbf{A}$ is observed as in model (1) so that

$$\mathbf{Y} = \mathbf{A} + \mathbf{Z}$$

where $\mathbf{Z}\in\mathbb{R}^{d_1\times d_2\times d_3}$ is an unknown noise tensor with i.i.d. entries satisfying $Z(i,j,k)\sim\mathcal{N}(0,\sigma^2)$. By observing $\mathbf{Y}$, our goal is to estimate $\mathbf{U},\mathbf{V}$ and $\mathbf{W}$. An immediate approach is to compute HOSVD of $\mathbf{Y}$. To this end, let $\hat{\mathbf{U}}\in\mathbb{R}^{d_1\times r_1}, \hat{\mathbf{V}}\in\mathbb{R}^{d_2\times r_2}, \hat{\mathbf{W}}\in\mathbb{R}^{d_3\times r_3}$ be the corresponding top singular vectors of $\mathcal{M}_1(\mathbf{Y}), \mathcal{M}_2(\mathbf{Y})$ and $\mathcal{M}_3(\mathbf{Y})$. The key factor characterizing the perturbation bounds of $\hat{\mathbf{U}},\hat{\mathbf{V}}$ and $\hat{\mathbf{W}}$ is the so-called eigengap.

Since the computing of $\hat{\mathbf{U}}$ is essentially via the matrix SVD on $\mathcal{M}_1(\mathbf{A})$, it suffices to consider the eigengaps of matrices. Given a rank r matrix $\mathbf{M}\in\mathbb{R}^{m_1\times m_2}$ with SVD:

$$\mathbf{M} = \sum_{k=1}^{r}\lambda_k\big(\mathbf{g}_k\otimes\mathbf{h}_k\big)$$

where $\mathbf{M}$'s singular values are $\lambda_1\ge\lambda_2\ge\ldots\ge\lambda_r>0$ and $\{\mathbf{g}_1,\ldots,\mathbf{g}_r\}$ are the corresponding left singular vectors and $\{\mathbf{h}_1,\ldots,\mathbf{h}_r\}$ are $\mathbf{M}$'s corresponding right singular vectors. We further introduce $\lambda_0=+\infty$ and $\lambda_{r+1}=0$. The k-th eigengap of matrix $\mathbf{M}$ is then defined by

$$\bar{g}_k(\mathbf{M}) := \min\big(\lambda_k-\lambda_{k+1},\lambda_{k-1}-\lambda_k\big), \quad \forall\ 1\le k\le r.$$

Recall that $\mathbf{U},\hat{\mathbf{U}}\in\mathbb{R}^{d_1\times r_1}$ are the top-r_1 left singular vectors of $\mathcal{M}_1(\mathbf{A})$ and $\mathcal{M}_1(\mathbf{Y})$ respectively. By Davis-Kahan Theorem (Davis and Kahan (1970)) or Wedin's $\sin\Theta$ theorem (Wedin (1972)), we get

$$\|\hat{\mathbf{U}}\hat{\mathbf{U}}^\top-\mathbf{U}\mathbf{U}^\top\| = O\Big(\frac{\|\mathcal{M}_1(\mathbf{Z})\|}{\bar{g}_{r_1}\big(\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A})\big)}\Big), \tag{9}$$

which is generally sub-optimal especially when $\mathcal{M}_1(\mathbf{Z})\in\mathbb{R}^{d_1\times(d_2d_3)}$ is unbalanced such that $d_2d_3\gg d_1$. Sharper bounds in ℓ_2-norm concerning one sided perturbation have been derived

in Zheng and Tomioka (2015), Wang (2015) and Cai and Zhang (2016). In this paper, we derive sharp perturbation bounds of $\hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{W}}$ in ℓ_∞-norm which illustrate unconventional phase transitions for various statistical learning applications. More generally, we will investigate the perturbation bounds of linear forms $\langle \hat{\mathbf{u}}_k, \mathbf{x} \rangle$ for any fixed vector $\mathbf{x} \in \mathbb{R}^{d_1}$. Similar results can be obtained for singular vectors $\hat{\mathbf{V}}$ and $\hat{\mathbf{W}}$.

3. Main Results

3.1. Second Order Method for One-sided Spectral Analysis

The ℓ_∞-norm perturbation bounds for singular subspaces of balanced matrices has been developed in Koltchinskii and Xia (2016). Recall that $\mathbf{u}_k$ denotes the k-th left singular vector of $\mathcal{M}_1(\mathbf{A})$ and $\hat{\mathbf{u}}_k$ denotes the k-th left singular vector of $\mathcal{M}_1(\mathbf{Y})$ where $\mathcal{M}_1(\mathbf{A})$ is of size $d_1 \times (d_2 d_3)$. The operator norm $\|\mathcal{M}_1(\mathbf{Z})\|$ is generally determined by the larger dimension $(d_1 \vee d_2 d_3)$, see Section 6. It turns out that the machinery in Koltchinskii and Xia (2016) is sub-optimal concerning the SNR requirement. Indeed, the eigengap requirement in Koltchinskii and Xia (2016) becomes $\bar{g}_k\big(\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A})\big) \gg \sigma\big(d_1 \vee d_2 d_3\big)^{1/2}$, which shall is unnecessarily strong in view of the recent results in Zheng and Tomioka (2015), Cai and Zhang (2016), Zhang and Xia (2018) and Liu et al. (2017).

To bridge such gaps, we conduct a second order spectral analysis for $\hat{\mathbf{U}}$. The key observation is that the top left singular vectors of $\mathcal{M}_1(\mathbf{Y})$ are also the top eigenvectors of $\mathcal{M}_1(\mathbf{Y})\mathcal{M}_1^\top(\mathbf{Y})$. The second order method seeks the eigenspace perturbation on $\mathcal{M}_1(\mathbf{Y})\mathcal{M}_1^\top(\mathbf{Y})$ instead of singular space perturbation on $\mathcal{M}_1(\mathbf{Y})$. Clearly, we have

$$\mathcal{M}_1(\mathbf{Y})\mathcal{M}_1^\top(\mathbf{Y}) = \mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A}) + \mathbf{\Gamma} \in \mathbb{R}^{d_1 \times d_1}$$

where $\mathbf{\Gamma} = \mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{Z}) + \mathcal{M}_1(\mathbf{Z})\mathcal{M}_1^\top(\mathbf{A}) + \mathcal{M}_1(\mathbf{Z})\mathcal{M}_1^\top(\mathbf{Z})$. Note that $\mathbf{U}$ are the leading eigenvectors of $\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A})$ and $\hat{\mathbf{U}}$ are the top-r_1 eigenvectors of $\mathcal{M}_1(\mathbf{Y})\mathcal{M}_1^\top(\mathbf{Y})$. Moreover, the following relation on eigengaps is obvious:

$$\bar{g}_{r_1}\Big(\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A})\Big) \geq \bar{g}_{r_1}^2\big(\mathcal{M}_1(\mathbf{A})\big).$$

The advantage of second order method comes from the observation that even though $\mathbb{E}\big\|\mathcal{M}_1(\mathbf{Z})\mathcal{M}_1^\top(\mathbf{Z})\big\|$ is of the order $\sigma^2(d_1 \vee d_2 d_3)$, the symmetric matrix $\mathcal{M}_1(\mathbf{Z})\mathcal{M}_1^\top(\mathbf{Z})$ is concentrated at $d_2 d_3 \sigma^2 \mathbf{I}_{d_1}$ such that (see more details in Section 6)

$$\big\|\mathcal{M}_1(\mathbf{Z})\mathcal{M}_1^\top(\mathbf{Z}) - \sigma^2 d_2 d_3 \mathbf{I}_{d_1}\big\| = O_p\Big(\sigma^2\big(d_1 d_2 d_3\big)^{1/2}\Big).$$

Note that subtracting by an identity matrix does not affect the eigen-structure. The second order method introduces the additional term $\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{Z})$ whose operator norm is bounded by $O_p\big(\sigma\sqrt{d_1}\big\|\mathcal{M}_1(\mathbf{A})\big\|\big)$, which creates a constraint on the condition number of $\mathcal{M}_1(\mathbf{A})$. However, in order to characterize sharp perturbation bounds of linear forms $\langle \hat{\mathbf{u}}_k, \mathbf{x} \rangle$, we need to pay more attention to dealing with correlations among the higher order terms than the first order method in Koltchinskii and Xia (2016). We note that the idea of second order method is already existing in the literature (see, e.g., Zheng and Tomioka (2015) for the ℓ_2-norm perturbation bounds). The second order moment method is only the starting point of our technical analysis which significantly reduces the SNR requirements.

Our most fundamental contribution is about the sup-norm characterization of the empirical singular vectors. Basically, we observe that the empirical singular vectors are biased and the bias is nicely aligned with the true singular vectors. After subtracting the bias, the empirical singular vectors exhibit the so-called *delocalization* property where all the entry-wise perturbations have comparable magnitudes. Such *delocalization* property is universal meaning that no conditions on the true singular vectors are needed. In Section 4, we show that the sup-norm perturbation bounds indeed reveal unconventional phase transitions in statistical learning applications such as the exact clustering in high dimensional Gaussian mixture models and the exact support recovery in sub-tensor localizations.

3.2. Perturbation of Linear Forms of Singular Vectors

In this section, we present our main theorem characterizing the perturbation of linear forms $\langle \hat{\mathbf{u}}_k, \mathbf{x} \rangle$ for any $\mathbf{x} \in \mathbb{R}^{d_1}$, where $\hat{\mathbf{u}}_k$ is the k-th left singular vector of $\mathcal{M}_1(\mathbf{Y})$. Our results have similar implications as the previous work Koltchinskii and Xia (2016), meaning that the bias $\mathbb{E}\hat{\mathbf{u}}_k\hat{\mathbf{u}}_k^\top - \mathbf{u}_k\mathbf{u}_k^\top$ is well aligned with $\mathbf{u}_k\mathbf{u}_k^\top$. Therefore, after correcting the bias term, we are able to obtain a sharper estimation of linear forms $\langle \mathbf{u}_k, \mathbf{x} \rangle$. To this end, we denote the condition number of the matrix $\mathcal{M}_1(\mathbf{A})$ by

$$\kappa\big(\mathcal{M}_1(\mathbf{A})\big) = \frac{\lambda_{\max}\big(\mathcal{M}_1(\mathbf{A})\big)}{\lambda_{\min}\big(\mathcal{M}_1(\mathbf{A})\big)}$$

where $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ return the largest and smallest nonzero singular values, respectively. Since $\hat{\mathbf{u}}_k$ is up to the switch of signs, we choose $\hat{\mathbf{u}}_k$ in the following theorems, remarks and corollaries so that $\big\langle \hat{\mathbf{u}}_k, \mathbf{u}_k \big\rangle > 0$.

Theorem 1 *Let*[3] $\mathbf{M} := \mathcal{M}_1(\mathbf{A})$ *and* $\delta(d_1, d_2, d_3) := \sigma d_1^{1/2}\|\mathbf{M}\| + \sigma^2(d_1 d_2 d_3)^{1/2}$ *and suppose* $d_2 d_3 e^{-d_1/2} \leq 1$. *There exist absolute constants* $D_1, D_2 > 0$ *such that the following fact holds. Let* $\mathbf{u}_k$ *be* $\mathbf{M}$*'s* k*-th left singular vector with multiplicity* 1. *If* $\bar{g}_k\big(\mathbf{M}\mathbf{M}^\top\big) \geq D_1\delta(d_1, d_2, d_3)$, *there exist a constant* $b_k \in [-1/2, 0]$ *with* $|b_k| \leq \frac{\sqrt{2}\delta(d_1,d_2,d_3)}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)}$ *such that for any* $\mathbf{x}$, *the following bound holds with probability at least* $1 - e^{-t}$,

$$\begin{aligned}&\big|\langle \hat{\mathbf{u}}_k, \mathbf{x} \rangle - (1 + b_k)^{1/2}\langle \mathbf{u}_k, \mathbf{x} \rangle\big| \\ &\quad\leq D_2\bigg(t^{1/2}\frac{\sigma\|\mathbf{M}\| + \sigma^2(d_2 d_3)^{1/2}}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)} + \frac{\sigma^2 d_1}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)}\Big(\frac{\delta(d_1, d_2, d_3)}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)}\Big)\bigg)\|\mathbf{x}\|_{\ell_2} \qquad (10)\end{aligned}$$

for all $\log 8 \leq t \leq d_1$. *In particular, if* $\mathbf{x} = \pm\mathbf{u}_k$, *then with the same probability,*

$$\begin{aligned}\big|\big|\langle \hat{\mathbf{u}}_k, \mathbf{u}_k \rangle\big| - 1\big| &\leq \big|\sqrt{1 + b_k} - 1\big| \\ &\quad + D_2\bigg(t^{1/2}\frac{\sigma\|\mathbf{M}\| + \sigma^2(d_2 d_3)^{1/2}}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)} + \frac{\sigma^2 d_1}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)}\Big(\frac{\delta(d_1, d_2, d_3)}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)}\Big)\bigg).\end{aligned}$$

By Theorem 1, it is easy to check that the condition $\bar{g}_k\big(\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A})\big) \geq D_1\delta(d_1, d_2, d_3)$ holds whenever

$$\bar{g}_k\big(\mathcal{M}_1(\mathbf{A})\big) \geq D_1\Big(\sigma(d_1 d_2 d_3)^{1/4} + \sigma d_1^{1/2}\kappa\big(\mathcal{M}_1(\mathbf{A})\big)\Big).$$

3. Observe that if we set $d_3 = 1$ and consider the case with $d_1 \ll d_2$, then Theorem 1 elaborates the one-sided perturbation bounds in ℓ_∞-norm for singular vectors of unbalanced (or fat) matrices.

If $\kappa\big(\mathcal{M}_1(\mathbf{A})\big) \le \big(\frac{d_2d_3}{d_1}\big)^{1/4}$, the above bound becomes $\bar{g}_k\big(\mathcal{M}_1(\mathbf{A})\big) \gg \sigma(d_1d_2d_3)^{1/4}$ which is a standard requirement in tensor SVD or PCA, see Zheng and Tomioka (2015), Zhang and Xia (2018), Hopkins et al. (2015) and Richard and Montanari (2014). By taking $\mathbf{x}$ over the standard basis vectors in $\mathbb{R}^{d_1}$ and choosing $t \ge D_3 \log d_1$, we end up with a ℓ_∞-norm perturbation bound for empirical singular vector $\hat{\mathbf{u}}_k$.

Corollary 2 *Under the conditions in Theorem 1, there exists a universal constant $D_1 > 0$ such that the following bound holds with probability at least $1 - \frac{1}{d_1}$,*

$$\big\|\hat{\mathbf{u}}_k - (1+b_k)^{1/2}\mathbf{u}_k\big\|_{\ell_\infty} \le D_1\bigg(\Big(\frac{\log d_1}{d_1}\Big)^{1/2} + \Big(\frac{d_1}{d_2d_3}\Big)^{1/2}\bigg).$$

If $d_1 \asymp d_2 \asymp d_3 \asymp d$, we obtain

$$\mathbb{P}\Big(\big\|\hat{\mathbf{u}}_k - (1+b_k)^{1/2}\mathbf{u}_k\big\|_{\ell_\infty} \ge D_1\Big(\frac{\log d}{d}\Big)^{1/2}\Big) \le \frac{1}{d}$$

which has an analogous form to the perturbation bound in Koltchinskii and Xia (2016) implying a famous *delocalization* phenomenon in random matrix theory, see Rudelson and Vershynin (2015) and Vu and Wang (2015) and references therein.

Remark 3 *Let's compare with the ℓ_2-norm bound in Zheng and Tomioka (2015) in the case that rank $r = 1$, $d_1 = d_2 = d_3 = d$ and signal strength $\bar{g}_1(\mathbf{M}\mathbf{M}^\top) = \lambda^2$. By (Zheng and Tomioka, 2015, Theorem 1), if $\lambda \gg \sigma d^{3/4}$, then*

$$\|\hat{\mathbf{u}}_1 - \mathbf{u}_1\|_{\ell_2} = O_p\Big(\frac{d^{1/2}\sigma}{\lambda} + \frac{\sigma^2 d^{3/2}}{\lambda^2}\Big). \tag{11}$$

By Theorem 1, if $\lambda \gg \sigma d^{3/4}$, then we get

$$\big\|\hat{\mathbf{u}}_1 - (1+b_1)^{1/2}\mathbf{u}_1\big\|_{\ell_\infty} = O_p\Big(\frac{\sigma\log^{1/2} d}{\lambda} + \frac{\sigma^2 d\log^{1/2} d}{\lambda^2}\Big) \tag{12}$$

for a constant $b_1 \in [-1/2, 0]$ depending on $\mathbf{u}_1$ and λ only. By (12) and (11), we observe that, after subtracting the bias, the entry-wise deviation of the empirical left singular vector $\hat{\mathbf{u}}_1$ is about $\sqrt{\frac{\log d}{d}}$ of the ℓ_2-norm perturbation bound of $\hat{\mathbf{u}}_1$. It means that, after subtracting the bias, the deviations of all $\hat{\mathbf{u}}_1$'s entries have comparable magnitudes, namely the so-called delocalization property. Interestingly, if $|u_1(j)| \gg \frac{1}{\sqrt{d}}$, then eq. (12) implies that $\hat{u}_1(j)$ has the same sign as $u_1(j)$ as long as $\lambda \gg \sigma d^{3/4}$. This sign consistency is crucial for guaranteeing the exact clustering of high dimensional mixture model, see more details in Section 4.

The bias b_k is usually unknown and we borrow the idea in Koltchinskii and Xia (2016) to estimate b_k based on two independent samples. It happens in the application of tensor decomposition for gene expression data where usually multiple independent copies are available, see more details in Hore et al. (2016).

Suppose that two independent noisy version of $\mathbf{A} \in \mathbb{R}^{d_1\times d_2\times d_3}$ are observed with $\mathbf{Y}^{(1)} = \mathbf{A} + \mathbf{Z}^{(1)}$ and $\mathbf{Y}^{(2)} = \mathbf{A} + \mathbf{Z}^{(2)}$ where $\mathbf{Z}^{(1)}$ and $\mathbf{Z}^{(2)}$ have i.i.d. centered Gaussian entries with

variance σ^2 as in (1). Let $\hat{\mathbf{u}}_k^{(1)}$ and $\hat{\mathbf{u}}_k^{(2)}$ denote the k-th left singular vector of $\mathcal{M}_1\big(\mathbf{Y}^{(1)}\big)$ and $\mathcal{M}_1\big(\mathbf{Y}^{(2)}\big)$, respectively. The signs of $\hat{\mathbf{u}}_k^{(1)}$ and $\hat{\mathbf{u}}_k^{(2)}$ are chosen such that $\langle\hat{\mathbf{u}}_k^{(1)}, \hat{\mathbf{u}}_k^{(2)}\rangle \ge 0$. Define the estimator of b_k by

$$\hat{b}_k := \langle\hat{\mathbf{u}}_k^{(1)}, \hat{\mathbf{u}}_k^{(2)}\rangle - 1.$$

Define the scaled version of empirical singular vector $\widetilde{\mathbf{u}}_k := \frac{\hat{\mathbf{u}}_k}{(1+\hat{b}_k)^{1/2}}$, which is not necessarily a unit vector.

Theorem 4 *Under the assumptions in Theorem 1, there exists an absolute constant $D_1 > 0$ such that for any $\mathbf{x} \in \mathbb{R}^{d_1}$, the follow bound holds with probability at least $1 - e^{-t}$ for all $\log 8 \le t \le d_1$,*

$$\big|\hat{b}_k - b_k\big| \le D_1\bigg(t^{1/2}\frac{\sigma\|\mathbf{M}\| + \sigma^2(d_2d_3)^{1/2}}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)} + \frac{\sigma^2 d_1}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)}\Big(\frac{\delta(d_1,d_2,d_3)}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)}\Big)\bigg)$$

and

$$\big|\langle\widetilde{\mathbf{u}}_k - \mathbf{u}_k, \mathbf{x}\rangle\big| \le D_1\bigg(t^{1/2}\frac{\sigma\|\mathbf{M}\| + \sigma^2(d_2d_3)^{1/2}}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)} + \frac{\sigma^2 d_1}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)}\Big(\frac{\delta(d_1,d_2,d_3)}{\bar{g}_k(\mathbf{M}\mathbf{M}^\top)}\Big)\bigg)\|\mathbf{x}\|_{\ell_2}$$

where $\mathbf{M} = \mathcal{M}_1(\mathbf{A})$.

Remark 5 *By Theorem 4, if $d/2 \le \min_k d_k \le \max_k d_k \le 2d$, we get*

$$\mathbb{P}\Big(\|\widetilde{\mathbf{u}}_k - \mathbf{u}_k\|_{\ell_\infty} \ge D_1\Big(\frac{\log d}{d}\Big)^{1/2}\Big) \le \frac{1}{d}.$$

3.3. Low Rank Tensor Denoising and Entry-wise Deviation Bound

In this section, we study a low rank estimate of $\mathbf{A}$ through the projection of $\mathbf{Y}$. Let $\widetilde{\mathbf{U}} = (\tilde{\mathbf{u}}_1, \ldots, \tilde{\mathbf{u}}_{r_1}) \in \mathbb{R}^{d_1\times r_1}$ be scaled singular vectors each of which is computed as in Theorem 4. Similarly, let $\widetilde{\mathbf{V}} \in \mathbb{R}^{d_2\times r_2}$ and $\widetilde{\mathbf{W}} \in \mathbb{R}^{d_3\times r_3}$ be the corresponding scaled singular vectors computed from $\mathcal{M}_2(\mathbf{Y})$ and $\mathcal{M}_3(\mathbf{Y})$. Define the low rank estimate

$$\widetilde{\mathbf{A}} := \mathbf{Y} \times_1 \mathbf{P}_{\widetilde{\mathbf{U}}} \times_2 \mathbf{P}_{\widetilde{\mathbf{V}}} \times_3 \mathbf{P}_{\widetilde{\mathbf{W}}}$$

where $\mathbf{P}_{\widetilde{\mathbf{U}}}$ represents the scaled projector $\mathbf{P}_{\widetilde{\mathbf{U}}} := \widetilde{\mathbf{U}}\widetilde{\mathbf{U}}^\top$. Clearly, $\mathrm{rank}(\widetilde{\mathbf{A}}) = (r_1, r_2, r_3)$ which serves as a low rank estimate of $\mathbf{A}$. We characterize the entry-wise accuracy of $\widetilde{\mathbf{A}}$, namely, the upper bound of $\|\widetilde{\mathbf{A}} - \mathbf{A}\|_{\ell_\infty}$ in terms of the coherence of $\mathbf{U}, \mathbf{V}$ and $\mathbf{W}$. Our $\|\widetilde{\mathbf{A}} - \mathbf{A}\|_{\ell_\infty}$ bound relies on the simultaneous ℓ_∞-norm perturbation bounds of $\{\tilde{\mathbf{u}}_{k_1}\}_{k_1=1}^{r_1}$, $\{\tilde{\mathbf{v}}_{k_2}\}_{k_2=1}^{r_2}$ and $\{\tilde{\mathbf{w}}_{k_3}\}_{k_3=1}^{r_3}$. We impose the following conditions on the eigengaps: for a large enough constant $D_1 > 0$,

$$\bar{g}_{k_1}\big(\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A})\big) \ge D_1\Big(\sigma d_1^{1/2}\overline{\Lambda}(\mathbf{A}) + \sigma^2(d_1d_2d_3)^{1/2}\Big), \quad 1 \le k_1 \le r_1, \tag{13}$$

$$\bar{g}_{k_2}\big(\mathcal{M}_2(\mathbf{A})\mathcal{M}_2^\top(\mathbf{A})\big) \ge D_1\Big(\sigma d_2^{1/2}\overline{\Lambda}(\mathbf{A}) + \sigma^2(d_1d_2d_3)^{1/2}\Big), \quad 1 \le k_2 \le r_2, \tag{14}$$

$$\bar{g}_{k_3}\big(\mathcal{M}_3(\mathbf{A})\mathcal{M}_3^\top(\mathbf{A})\big) \geq D_1\Big(\sigma d_3^{1/2}\overline{\Lambda}(\mathbf{A}) + \sigma^2(d_1d_2d_3)^{1/2}\Big), \quad 1 \leq k_3 \leq r_3, \tag{15}$$

where we denote by

$$\overline{\Lambda}(\mathbf{A}) := \max\big\{\lambda_{\max}\big(\mathcal{M}_1(\mathbf{A})\big), \lambda_{\max}\big(\mathcal{M}_2(\mathbf{A})\big), \lambda_{\max}\big(\mathcal{M}_3(\mathbf{A})\big)\big\}.$$

Similarly, we define

$$\underline{\Lambda}(\mathbf{A}) := \min\Big\{\lambda_{\min}\big(\mathcal{M}_1(\mathbf{A})\big),\ \lambda_{\min}\big(\mathcal{M}_2(\mathbf{A})\big),\ \lambda_{\min}\big(\mathcal{M}_3(\mathbf{A})\big)\Big\}$$

and the overall eigengap

$$\bar{g}_{\min}\big(\mathbf{A}\big) := \min\Big\{\bar{g}_{k_1}^{1/2}\big(\mathcal{M}_1(\mathbf{A})\mathcal{M}_1^\top(\mathbf{A})\big), \bar{g}_{k_2}^{1/2}\big(\mathcal{M}_2(\mathbf{A})\mathcal{M}_2^\top(\mathbf{A})\big), \bar{g}_{k_3}^{1/2}\big(\mathcal{M}_3(\mathbf{A})\mathcal{M}_3^\top(\mathbf{A})\big) \\ , 1 \leq k_1 \leq r_1, 1 \leq k_2 \leq r_2, 1 \leq k_3 \leq r_3\Big\}.$$

By definition, it is clear that $\underline{\Lambda}(\mathbf{A}) \geq \bar{g}_{\min}(\mathbf{A})$.

Theorem 6 *Suppose conditions (13) (14) (15) hold and assume that for all $i \in [d_1], j \in [d_2], k \in [d_3]$,*

$$\|\mathbf{U}^\top\mathbf{e}_i\|_{\ell_2} \leq \mu_{\mathbf{U}}\sqrt{\frac{r_1}{d_1}}, \quad \|\mathbf{V}^\top\mathbf{e}_j\|_{\ell_2} \leq \mu_{\mathbf{V}}\sqrt{\frac{r_2}{d_2}}, \quad \|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2} \leq \mu_{\mathbf{W}}\sqrt{\frac{r_3}{d_3}}$$

for some constants $\mu_{\mathbf{U}}, \mu_{\mathbf{V}}, \mu_{\mathbf{W}} \geq 0$. Suppose that $\frac{d}{2} \leq \min_{1\leq k\leq 3} d_k \leq \max_{1\leq k\leq 3} d_k \leq 2d$ and $\frac{r}{2} \leq \min_{1\leq k\leq 3} r_k \leq \max_{1\leq k\leq 3} r_k \leq 2r$. Then, there exists an absolute constant $D_2 > 0$ such that, with probability at least $1 - \frac{1}{d}$,

$$\big\|\tilde{\mathbf{A}} - \mathbf{A}\big\|_{\ell_\infty} \\ \leq D_2\sigma r^3\Big(\frac{\widetilde{\kappa}(\mathbf{A})\sigma}{\bar{g}_{\min}(\mathbf{A})} + \frac{\widetilde{\kappa}^2(\mathbf{A})}{d}\Big)\big(\mu_{\mathbf{U}}\mu_{\mathbf{V}} + \mu_{\mathbf{U}}\mu_{\mathbf{W}} + \mu_{\mathbf{V}}\mu_{\mathbf{W}}\big)\log^{3/2} d$$

where $\widetilde{\kappa}(\mathbf{A}) = \overline{\Lambda}(\mathbf{A})/\bar{g}_{\min}(\mathbf{A})$.

Remark 7 *To highlight the contribution of Theorem 6, let $r = O(1)$ and $\widetilde{\kappa}(\mathbf{A}) = O(1)$. Note that if the coherence constants $\mu_{\mathbf{U}}, \mu_{\mathbf{V}}, \mu_{\mathbf{W}} = d^{(\frac{3}{4}-\varepsilon)/2}$ for $\varepsilon \in (0, 3/4)$, i.e., $\mathbf{U}, \mathbf{V}, \mathbf{W}$ can be almost spiked, under the minimal eigengap $\bar{g}_{\min}(\mathbf{A}) \gg \sigma d^{3/4}$, we obtain*

$$\|\tilde{\mathbf{A}} - \mathbf{A}\|_{\ell_\infty} = O_p\Big(\frac{\sigma}{d^\varepsilon}\log^{3/2} d\Big).$$

It worths to point out that the minimax optimal bound of estimating $\mathbf{A}$ in ℓ_2-norm is $O\big(\sigma d^{1/2}\big)$, see Zhang and Xia (2018). Theorem 6 is more interesting when $\mathbf{A}$ is incoherent such that $\mu_{\mathbf{U}}, \mu_{\mathbf{V}}, \mu_{\mathbf{W}} = O(1)$ where we can conclude that

$$\|\tilde{\mathbf{A}} - \mathbf{A}\|_{\ell_\infty} = O_p\Big(\Big(\frac{\sigma^2}{\bar{g}_{\min}(\mathbf{A})} + \frac{\sigma}{d}\Big)\log^{3/2} d\Big) = O_p\Big(\frac{\sigma}{d^{3/4}}\log^{3/2} d\Big). \tag{16}$$

By (16), if the entry $|A(j_1, j_2, j_3)| \gg \frac{\sigma\log^{3/2} d}{d^{3/4}}$, then the entry $\tilde{A}(j_1, j_2, j_3)$ maintains the same sign as $A(j_1, j_2, j_3)$. In Section 4 and Remark 13, we show that the sup-norm bound of $\tilde{\mathbf{A}} - \mathbf{A}$ is useful for the exact support recovery of sub-tensor localizations, under minimal signal strength requirements (that is the support size).

4. Applications

In this section, we review two applications of ℓ_∞-norm perturbation bound. In these applications, we note that it is unnecessary to estimate the bias b_k. We show that the sup-norm perturbation bounds reveal unconventional phase transitions in these statistical learning applications. Meanwhile, novel yet simple statistical algorithms can be designed based on the sup-norm perturbation bounds.

4.1. High Dimensional Clustering

Many statistical and machine learning tasks are associated with clustering high dimensional data, see McCallum et al. (2000), Parsons et al. (2004), Fan and Fan (2008), Hastie et al. (2009), Friedman (1989) and references therein. We consider a two-class Gaussian mixture model such that each data point $\mathbf{y}_i \in \mathbb{R}^p$ can be represented by

$$\mathbf{y}_i = -\ell_i\boldsymbol{\beta} + (1 - \ell_i)\boldsymbol{\beta} + \boldsymbol{\varepsilon}_i \in \mathbb{R}^p \tag{17}$$

where the associated label $\ell_i \in \{0, 1\}$ for $i = 1, 2, \ldots, n$ is unknown and the noise vector $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_p)$. The vector $\boldsymbol{\beta} \in \mathbb{R}^p$ is unknown with $p \gg n$. We denote the true clusters by

$$\mathcal{N}_0 := \{1 \leq i \leq n : \ell_i = 0\} \quad \text{and} \quad \mathcal{N}_1 := \{1 \leq i \leq n : \ell_i = 1\}.$$

Given the data matrix

$$\mathbf{Y} = \big(\mathbf{y}_1, \ldots, \mathbf{y}_n\big)^\top \in \mathbb{R}^{n\times p}$$

, our goal is to cluster the n data points into two disjoint groups. Let $n_{k+1} := \mathrm{Card}\big(\mathcal{N}_k\big)$ for $k = 0, 1$ such that $n_1 + n_2 = n$. Observe that $\mathbb{E}\mathbf{Y}$ has rank 1 and its leading left singular vector $\mathbf{u} \in \mathbb{R}^n$ with

$$u(i) = \frac{1 - \ell_i}{n^{1/2}} - \frac{\ell_i}{n^{1/2}}, \quad 1 \leq i \leq n.$$

The signs of $\mathbf{u}$ immediately suggest the cluster memberships of each data points. Moreover, the leading singular value of $\mathbb{E}\mathbf{Y}$ is $n^{1/2}\|\boldsymbol{\beta}\|_{\ell_2}$. Let $\hat{\mathbf{u}}$ denote the leading left singular vector of $\mathbf{Y}$. By Corollary 2, if $\|\boldsymbol{\beta}\|_{\ell_2} \geq D_1\big(1 \vee (p/n)^{1/4}\big)$ such that $|(1 + b_k)^{-1/2} - 1| \leq 1/2$, then there exists an event $\mathcal{E}$ with $\mathbb{P}(\mathcal{E}) \geq 1 - \frac{1}{n}$ so that on event $\mathcal{E}$,

$$\big\|\hat{\mathbf{u}} - (1 + b_k)^{1/2}\mathbf{u}\big\|_{\ell_\infty} \leq D_2\Big(\frac{1}{\|\boldsymbol{\beta}\|_{\ell_2}} + \frac{(p/n)^{1/2}}{\|\boldsymbol{\beta}\|_{\ell_2}^2}\Big)\Big(\frac{1}{\|\boldsymbol{\beta}\|_{\ell_2}^2} + \sqrt{\frac{\log n}{n}}\Big).$$

On event $\mathcal{E}$, if $\|\boldsymbol{\beta}\|_{\ell_2} \geq D_1\Big(n^{1/6} \vee p^{1/8} \vee \big(p\log(n)/n\big)^{1/4}\Big)$, then we get

$$\|\hat{\mathbf{u}} - \mathbf{u}\|_{\ell_\infty} \leq \|\hat{\mathbf{u}} - (1 + b_k)^{1/2}\mathbf{u}\|_{\ell_\infty} + \big|(1 + b_k)^{-1/2} - 1\big|\|\mathbf{u}\|_{\ell_\infty} \\ \leq \|\hat{\mathbf{u}} - (1 + b_k)^{1/2}\mathbf{u}\|_{\ell_\infty} + \frac{1}{2n^{1/2}} \leq \frac{3}{4n^{1/2}} \tag{18}$$

implying that if $\ell_i = \ell_j$, then $\mathrm{sign}\big(\hat{u}(i)\big) = \mathrm{sign}\big(\hat{u}(j)\big)$ for all $1 \leq i, j \leq n$. Therefore, we propose a simple clustering algorithm by entry-wise signs of $\hat{\mathbf{u}}$ in Algorithm 1.

By the bound (18), Algorithm 1 can guarantee exact clustering as follows.

Algorithm 1 High dimensional bi-clustering by entry-wise signs.

Input: Data matrix $\mathbf{Y} \in \mathbb{R}^{n\times p}$
Calculate the leading left singular vector of $\mathbf{Y}$, denoted by $\hat{\mathbf{u}} \in \mathbb{R}^n$
Initiate $\hat{\mathcal{N}}_0 = \{\}$ and $\hat{\mathcal{N}}_1 = \{\}$
for $i = 1, \cdots, n$ **do**
  if $\hat{u}(i) \geq 0$ **then**
    $\hat{\mathcal{N}}_0 \leftarrow \hat{\mathcal{N}}_0 \cup \{i\}$
  else
    $\hat{\mathcal{N}}_1 \leftarrow \hat{\mathcal{N}}_1 \cup \{i\}$
  end if
end for
Output: $\hat{\mathcal{N}}_0$ and $\hat{\mathcal{N}}_1$.

Theorem 8 *Suppose model (17) holds with noise vector $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \mathbf{I}_p)$. Let $\hat{\mathcal{N}}_0$ and $\hat{\mathcal{N}}_1$ be the output of Algorithm 1. There exists an absolute constant $D_1 > 0$ such that if $\|\boldsymbol{\beta}\|_{\ell_2} \geq D_1\Big(n^{1/6} \vee p^{1/8} \vee \big(p\log(n)/n\big)^{1/4}\Big)$, then with probability at least $1 - \frac{1}{n}$,*

$$\hat{\mathcal{N}}_0 = \mathcal{N}_0 \quad \text{or} \quad \hat{\mathcal{N}}_0 = \mathcal{N}_1.$$

The proof of Theorem 8 is straightforward based on eq. (18). We note that eq. (18) also implies that it is unnecessary to estimate b_k in this application, since scaling switch the entry-wise signs simultaneously and thus maintains the clustering outputs.

Remark 9 *Theorem 8 reveal unconventional phase transition thresholds for the exact clustering of Gaussian mixture model (17). Indeed, by Theorem 8, the sup-norm based clustering algorithm (Algorithm 1) will exactly recover the memberships with high probability when the signal strength satisfies*

$$\|\boldsymbol{\beta}\|_{\ell_2} \gg \Big(n^{1/6} \vee p^{1/8} \vee \big(p\log(n)/n\big)^{1/4}\Big).$$

In comparison, the ℓ_2-norm based clustering algorithm in Cai and Zhang (2016) and Zheng and Tomioka (2015) requires

$$\|\boldsymbol{\beta}\|_{\ell_2} \gg \big(n^{1/2} \vee p^{1/4}\big)$$

for exact clustering. Clearly, with respect to exact recovery, the sup-norm based clustering algorithm requires much weaker SNR conditions.

Remark 10 *The above framework can be directly generalized to Gaussian mixture model with k-clusters. Suppose that the j-th cluster has mean vector $\boldsymbol{\beta}_j$ and size n_j, then without loss of generality, the data matrix $\mathbf{Y} = \mathbf{M} + \mathbf{Z}$*

$$\mathbf{M} = \big(\underbrace{\boldsymbol{\beta}_1, \cdots, \boldsymbol{\beta}_1}_{n_1}, \cdots, \underbrace{\boldsymbol{\beta}_j, \cdots, \boldsymbol{\beta}_j}_{n_j}, \cdots, \underbrace{\boldsymbol{\beta}_k, \cdots, \boldsymbol{\beta}_k}_{n_k}\big)^\top \in \mathbb{R}^{N\times p}$$

with $N = \sum_{j=1}^k n_j$ and $\mathbf{Z} \in \mathbb{R}^{N\times p}$ having i.i.d. standard Gaussian entries. Observe that $\mathrm{rank}(\mathbf{M}) \leq k$, it suffices to consider the top-k left singular vectors of $\mathbf{M}$. However, it

requires nontrivial effort to investigate the eigengaps of $\mathbf{M}$ without further assumptions on $\{\boldsymbol{\beta}_j\}_{j=1}^k$. In the case that $n_j = n$ and $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_k$ are mutually orthogonal such that $\|\boldsymbol{\beta}_1\|_{\ell_2} \geq \ldots \geq \|\boldsymbol{\beta}_k\|_{\ell_2}$, then $\mathbf{M}$'s top-k singular values are $\lambda_j = \sqrt{n_j}\|\boldsymbol{\beta}_j\|_{\ell_2}, 1 \leq j \leq k$. Clearly, the non-zero entries of $\mathbf{M}$'s top-k left singular vectors provide the cluster membership of each data points. By Theorem 1, if $\Delta_j \geq C_1\sqrt{k}\|\boldsymbol{\beta}_1\|_{\ell_2} + C_2(kp/n)^{1/2}$ where $\Delta_j = \min\{\big(\|\boldsymbol{\beta}_j\|_{\ell_2}^2 - \|\boldsymbol{\beta}_{j+1}\|_{\ell_2}^2\big), \big(\|\boldsymbol{\beta}_{j-1}\|_{\ell_2}^2 - \|\boldsymbol{\beta}_j\|_{\ell_2}^2\big)\}$, then

$$\|\hat{\mathbf{u}}_j - \sqrt{1+b_j}\mathbf{u}_j\|_{\ell_\infty} = O_p\Big(\big(\frac{\|\boldsymbol{\beta}_1\|_{\ell_2}}{\Delta_j} + \frac{(p/n)^{1/2}}{\Delta_j}\big)\big(\frac{k^{3/2}}{\Delta_j} + \sqrt{\frac{k\log n}{n}}\big)\Big)$$

for all $1 \leq j \leq k$.

4.2. Sub-tensor Localization

In gene expression association analysis (see Hore et al. (2016), Xiong et al. (2012), Kolar et al. (2011) and Ben-Dor et al. (2003)) and planted clique detection (see Brubaker and Vempala (2009), Anandkumar et al. (2013) and Gauvin et al. (2014)), the goal is equivalent to locating a sub-tensor whose entries are statistically more significant than the others. One simple model characterizing this type of tensor data is as

$$\mathbf{Y} = \lambda \mathbf{1}_{C_1} \otimes \mathbf{1}_{C_2} \otimes \mathbf{1}_{C_3} + \mathbf{Z} \in \mathbb{R}^{d_1\times d_2\times d_3} \tag{19}$$

with $C_k = \cup_{j=1}^{s_k} C_k^{(j)} \subset [d_k]$ where $\big\{C_k^{(1)}, \ldots, C_k^{(s_k)}\big\}$ are disjoint subsets of $[d_k]$ for $k = 1, 2, 3$, i.e., there are $s_k \geq 1$ dense blocks in the k-th direction. Then, in total, there are $s_1s_2s_3$ dense blocks in $\mathbb{E}\mathbf{Y}$. The vector $\mathbf{1}_{C_k} \in \mathbb{R}^{d_k}$ is a zero-or-one vector whose entry equals 1 only when the index belongs to $\mathbf{C}_k$. The noise tensor $\mathbf{Z}$ has i.i.d. entries such that $Z(i,j,k) \sim \mathcal{N}(0,1)$. Given the noisy observation $\mathbf{Y}$, the goal is to locate the unknown subsets $\{C_1^{(j)}\}_{j=1}^{s_1}, \{C_2^{(j)}\}_{j=1}^{s_2}$ and $\{C_3^{(j)}\}_{j=1}^{s_3}$. The appealing scenario is $\lambda = O(1)$, since otherwise the signal is so strong that the problem can be easily solved by just looking at each entry. The tensor $\mathbb{E}\mathbf{Y}$ has rank 1 with leading singular value $\lambda|C_1|^{1/2}|C_2|^{1/2}|C_3|^{1/2}$ and corresponding singular vectors

$$\mathbf{u} = \frac{1}{|C_1|^{1/2}}\mathbf{1}_{C_1}, \quad \mathbf{v} = \frac{1}{|C_2|^{1/2}}\mathbf{1}_{C_2} \quad \text{and} \quad \mathbf{w} = \frac{1}{|C_3|^{1/2}}\mathbf{1}_{C_3},$$

where $|C|$ denotes the cardinality of C. By Theorem 1, if $\lambda \geq D_1\frac{(d_1d_2d_3)^{1/4}}{|C_1|^{1/2}|C_2|^{1/2}|C_3|^{1/2}}$ for a large enough constant $D_1 > 0$ and $d_{\max} \leq (d_1d_2d_3)^{1/2}$ where $d_{\max} := (d_1 \vee d_2 \vee d_3)$, then with probability at least $1 - \frac{1}{d_{\max}}$, we obtain

$$\begin{aligned}&\|\hat{\mathbf{u}} - (1+b_1)^{1/2}\mathbf{u}\|_{\ell_\infty}\\ &\quad\leq \frac{D_1\log^{1/2} d_{\max}}{\lambda|C_1|^{1/2}|C_2|^{1/2}|C_3|^{1/2}} + \frac{D_1(d_2d_3\log d_{\max})^{1/2}}{\lambda^2|C_1||C_2||C_3|} + \frac{D_1d_1}{\lambda^2|C_1||C_2||C_3|}\Big(\frac{(d_1d_2d_3)^{1/2}}{\lambda^2|C_1||C_2||C_3|}\Big),\end{aligned} \tag{20}$$

where $b_1 \in [-0.5, 0]$ is a constant depending on $\mathbf{u}, \mathbf{v}, \mathbf{w}$ and λ only. Similar bounds can be also derived for $\hat{\mathbf{v}}$ and $\hat{\mathbf{w}}$. By eq. (20), we propose a simple algorithm (Algorithm 2) for the

Algorithm 2 Sub-tensor localizations by entry-wise magnitudes.

Input: Data matrix $\mathbf{Y} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$

Calculate the leading left singular vectors of $\{\mathcal{M}_k(\mathbf{Y})\}_{k=1}^3$, denoted by $\hat{\mathbf{u}} \in \mathbb{R}^{d_1}, \hat{\mathbf{v}} \in \mathbb{R}^{d_2}$ and $\hat{\mathbf{w}} \in \mathbb{R}^{d_3}$, respectively.

Take entry-wise magnitudes $\{|\hat{u}(j_1)|\}_{j_1=1}^{d_1}$ and arrange them in a non-increasing order,

Record the top-$|C_1|$ locations and denote them by $\hat{C}_1$;

Take entry-wise magnitudes $\{|\hat{v}(j_2)|\}_{j_2=1}^{d_2}$ and arrange them in a non-increasing order,

Record the top-$|C_2|$ locations and denote them by $\hat{C}_2$;

Take entry-wise magnitudes $\{|\hat{w}(j_3)|\}_{j_3=1}^{d_3}$ and arrange them in a non-increasing order,

Record the top-$|C_3|$ locations and denote them by $\hat{C}_3$;

Output: $\hat{C}_1$, $\hat{C}_2$ and $\hat{C}_3$.

support recovery of sub-tensor model (19). By bound (20), we can immediately guarantee the exact support recovery by Algorithm 2. The proof is straightforward and is omitted here.

Theorem 11 *Suppose model (19) holds and $(d_1 + d_2 + d_3) \leq 2(d_1 d_2 d_3)^{1/2}$. There exist absolute constants $D_1, D_2 > 0$ such that if $\lambda \geq D_1 \frac{(d_1 d_2 d_3)^{1/4}}{(|C_1||C_2||C_3|)^{1/2}}$ and*

$$\max\left\{\sqrt{\frac{|C_1|}{d_1}}, \sqrt{\frac{|C_2|}{d_2}}, \sqrt{\frac{|C_3|}{d_3}}\right\} \cdot \frac{(d_1 d_2 d_3 \log d_{\max})^{1/2}}{\lambda^2 |C_1||C_2||C_3|} \leq \frac{1}{D_2},$$

then, with probability at least $1 - \frac{1}{d_1+d_2+d_3}$, we get

$$\hat{C}_1 = C_1 \quad \text{and} \quad \hat{C}_2 = C_2 \quad \text{and} \quad \hat{C}_3 = C_3$$

where $\{\hat{C}_k\}_{k=1}^3$ are the output of Algorithm 2.

Note that in Algorithm 2 and Theorem 11, it is also unnecessary to estimate the bias b_1 because we are interested in the top-$|C_1|$ largest entries of $|\hat{\mathbf{u}}|$ and scaling does not affect the ordering of the entry-wise magnitudes.

Remark 12 *The phase transition of Algorithm 2 and model (19) is intriguing. Note that the support localizations are trivial when $\lambda \gg 1$. Therefore, we only focus on the case $\lambda = 1$. Now, let $|C_1| \asymp |C_2| \asymp |C_3| = K$ and $d_1 \asymp d_2 \asymp d_3 = d$. By Theorem 11, we conclude that Algorithm 2 can exactly recover the supports C_1, C_2, C_3 with high probability if the support size $K \gg d^{\frac{1}{2}}$. Meanwhile, by the lower bound arguments in Zhang and Xia (2018), we know that if $K \ll d^{\frac{1}{2}}$, then there exist no polynomial time algorithms which can recover C_1 consistently. Put it differently, phase transition occurs at the threshold $O(d^{\frac{1}{2}})$ such that if $K \ll d^{\frac{1}{2}}$, the problem is unsolvable by polynomial time algorithms; if $K \gg d^{\frac{1}{2}}$, the problem can be perfectly solved by Algorithm 2. In comparison, the ℓ_2-norm based algorithms can only guarantee the consistency of support recovery when $K \gg d^{\frac{1}{2}}$, rather than the exact recovery.*

Remark 13 *We could also investigate the entry-wise denoising of model (19). Suppose that $|C_1| \asymp |C_2| \asymp |C_3| = K$ and $d_1 \asymp d_2 \asymp d_3 = d$. We denote by $\mathbf{A} = \mathbf{1}_{C_1} \otimes \mathbf{1}_{C_2} \otimes \mathbf{1}_{C_3}$ where we fix $\lambda = 1$ and we focus only on the support sizes $\{|C_k|\}_{k=1}^3$. Let $\hat{\mathbf{u}}, \hat{\mathbf{v}}$ and $\hat{\mathbf{w}}$ be the empirical singular vectors as in Algorithm 2. Define the projection estimator*

$$\hat{\mathbf{A}} = \mathbf{Y} \times_1 (\hat{\mathbf{u}}\hat{\mathbf{u}}^\top) \times_2 (\hat{\mathbf{v}}\hat{\mathbf{v}}^\top) \times_3 (\hat{\mathbf{w}}\hat{\mathbf{w}}^\top).$$

Similarly as in Theorem 6, we can show that there exists a constant $b \in [\sqrt{2}/4, 1]$ such that with probability at least $1 - \frac{1}{d}$,

$$\|\hat{\mathbf{A}} - b \cdot \mathbf{A}\|_{\ell_\infty} \leq D_1 \cdot \Big(\frac{1}{K} + \frac{d}{K^{5/2}}\Big) \log^{3/2} d \tag{21}$$

for some absolute constant $D_1 > 0$. Recall from model (19) that $A(j_1, j_2, j_3) = 1$ if $(j_1, j_2, j_3) \in C_1 \times C_2 \times C_3$. From eq. (21), we conclude that if $K \geq D_2\big(\sqrt{d} + d^{0.4} \log^{0.6} d\big)$ for a large enough absolute constant $D_2 > 0$ (note that the threshold $\sqrt{d}$ comes from SNR requirement as in eq. (20)), then

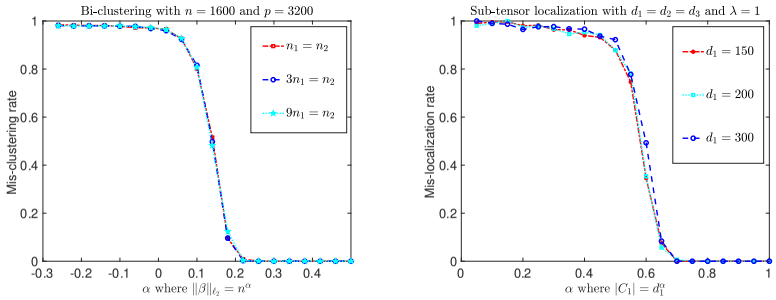
$$\big|\hat{A}(j_1, j_2, j_3)\big| > |\hat{A}(j_1', j_2', j_3')|$$

for all $(j_1, j_2, j_3) \in C_1 \times C_2 \times C_3$ and $(j_1', j_2', j_3') \notin C_1 \times C_2 \times C_3$. As a result, we can choose the locations of $\hat{\mathbf{A}}$'s entries with the largest-$|C_1||C_2||C_3|$ magnitudes and recover $\mathbf{A}$'s supports exactly.

4.3. Numerical Experiments

We present simulation results of experiments for the applications in Section 4. For high dimensional clustering in model (17), we randomly sample a vector $\boldsymbol{\beta} \in \mathbb{R}^p$ with $p = 3200$. For a fixed $\boldsymbol{\beta}$, we sample $n_1 = n/2 = 800$ random vectors from distribution $\mathcal{N}(\boldsymbol{\beta}, \mathbf{I}_p)$ and $n_2 = n/2 = 800$ random vectors from distribution $\mathcal{N}(-\boldsymbol{\beta}, \mathbf{I}_p)$. Then, we calculate the top left singular vector of $\mathbf{Y}$ as in (17) and apply Algorithm 1 to cluster the 1600 points into two disjoint groups. For each $\boldsymbol{\beta}$, we repeat the experiments for 50 times and the average mis-clustering rate is recorded. The signal strengths are chosen so that $\|\boldsymbol{\beta}\|_{\ell_2} = n^\alpha$ with $\alpha = 0.06 * k - 0.5$ for $1 \leq k \leq 20$. The average mis-clustering rates with respect to signal strengths are displayed in Figure (1a). Moreover, in Figure (1a), we also compare the average mis-clustering rates when two clusters have different sizes such as $3n_1 = n_2 = 1200$ and $9n_1 = n_2 = 1440$. As shown in Figure (1a), there exists a threshold around $\alpha = 0.18$ such that the mis-clustering rates by Algorithm 1 decreases extremely fast when the signal strength exceeds the threshold. Meanwhile, Figure (1a) also shows that the size balances of two clusters does not affect the threshold. Both these numerical observations from Figure (1a) are consistent with the theoretic guarantees from Theorem 8.

For sub-tensor localizations in model (19), we fix $\lambda = 1$ because the support localization task is trivial if $\lambda \gg 1$. Similarly as in Remark 12, it then suffices to investigate the efficiency of Algorithm 2 with respect to the support sizes. For simplicity, we choose $d_1 = d_2 = d_3$ and $C_1 = C_2 = C_3 = [|C_1|]$, that is, the sub-tensor is in the bottom-left-front corner of $\mathbb{E}\mathbf{Y}$. For each $d_1 = 150, d_1 = 200$ and $d_1 = 300$, we show the average mis-localization rates by Algorithm 2 with respect to the support size $|C_1|$. The average mis-localization rates are



(a) *Mis-clustering rates of Algorithm 1* **(b)** *Mis-localization rates of Algorithm 2*

Figure 1: *Simulation results for the performances of Algorithm 1 and Algorithm 2. In Figure (1a), the phase transition happens around the signal strength $\|\boldsymbol{\beta}\|_{\ell_2} \approx n^{0.18}$ which coincides with Theorem 8. Figure (1a) shows that Algorithm 1 can exactly recover the true clusters when signal strength exceeds the aforementioned threshold. Figure (1a) also shows that the efficiency of Algorithm 1 is unaffected when two clusters have unbalanced sizes. In Figure (1b), the phase transition happens when the support C_1 has size around $d_1^{0.6}$. It shows that Algorithm 2 can exactly locate the sub-tensor when the support size exceeds the aforementioned threshold.*

calculated from 50 independent experiments. The support sizes are chosen as $|C_1| = \lceil d_1^{\alpha} \rceil$ with $0.06 \le \alpha \le 1$. The results of mis-localization rates are displayed in Figure (1b). Indeed, Figure (1b) shows that the mis-localization rates by Algorithm 2 starts to decrease extremely fast when the support size is around $|C_1| \asymp d_1^{0.6}$. The exponent 0.6 is somewhat larger than the threshold 0.5 claimed in Remark 12. Note that the dimension size d is moderately large (only 300) in our simulations due to the heavy computational cost.

5. Acknowledgements

The authors would like to thank the action editor and three anonymous referees for their insightful suggestions which significantly improve the readability of the article.

6. Proofs

For notational brevity, we write $A \lesssim B$ if there exists an absolute constant D_1 such that $A \le D_1 B$. A similar notation would be $\gtrsim$ and $A \asymp B$ means that $A \lesssim B$ and $A \gtrsim B$ simultaneously. If the constant D_1 depends on some parameter γ, we shall write $\lesssim_\gamma, \gtrsim_\gamma$ and $\asymp_\gamma$.

Recall that the HOSVD is translated directly from SVD on $\mathcal{M}_1(\mathbf{A})$ and the matrix perturbation model $\mathcal{M}_1(\mathbf{Y}) = \mathcal{M}_1(\mathbf{A}) + \mathcal{M}_1(\mathbf{Z})$. Without loss of generality, it suffices to focus on matrices with unbalanced sizes. In the remaining context, we write $\mathbf{A}, \mathbf{Z}, \mathbf{Y} \in \mathbb{R}^{m_1 \times m_2}$ instead of $\mathcal{M}_1(\mathbf{A}), \mathcal{M}_1(\mathbf{Z}), \mathcal{M}_1(\mathbf{Y}) \in \mathbb{R}^{m_1 \times m_2}$, where $m_1 = d_1$ and $m_2 = d_2 d_3$ such

that $m_1 \ll m_2$. The second order spectral analysis begins with

$$\mathbf{Y}\mathbf{Y}^\top = \mathbf{A}\mathbf{A}^\top + \mathbf{\Gamma}, \quad \text{where} \quad \mathbf{\Gamma} = \mathbf{A}\mathbf{Z}^\top + \mathbf{Z}\mathbf{A}^\top + \mathbf{Z}\mathbf{Z}^\top.$$

Suppose that $\mathbf{A}$ has the thin singular value decomposition

$$\mathbf{A} = \sum_{k=1}^{r_1} \lambda_k (\mathbf{u}_k \otimes \mathbf{h}_k) \in \mathbb{R}^{m_1 \times m_2}$$

where $\{\mathbf{h}_1, \ldots, \mathbf{h}_{r_1}\} \subset \mathrm{span}\{\mathbf{v}_j \otimes \mathbf{w}_k^\top : j \in [r_2], k \in [r_3]\}$ are the right singular vectors of $\mathbf{A}$. Moreover, $\mathbf{A}\mathbf{A}^\top$ admits the eigen-decomposition:

$$\mathbf{A}\mathbf{A}^\top = \sum_{k=1}^{r_1} \lambda_k^2 (\mathbf{u}_k \otimes \mathbf{u}_k).$$

In an identical fashion, denote the eigen-decomposition of $\mathbf{Y}\mathbf{Y}^\top$ by

$$\mathbf{Y}\mathbf{Y}^\top = \sum_{k=1}^{m_1} \hat{\lambda}_k^2 (\hat{\mathbf{u}}_k \otimes \hat{\mathbf{u}}_k).$$

Even though Theorem 1 and Theorem 4 are stated when the singular value λ_k has multiplicity 1, we present more general results in this section. Note that when there are repeated singular values, the singular vectors are not uniquely defined. In this case, let $\mu_1 > \mu_2 > \ldots > \mu_s > 0$ be distinct singular values of $\mathbf{A}$ with $s \le r_1$. Denote $\Delta_k := \{j : \lambda_j = \mu_k\}$ for $1 \le k \le s$ and $\nu_k := \mathrm{Card}(\Delta_k)$ the multiplicity of μ_k. Let $\mu_{s+1} = 0$ which is a trivial eigenvalue of $\mathbf{A}\mathbf{A}^\top$ with multiplicity $m_1 - r_1$. Then, the spectral decomposition of $\mathbf{A}\mathbf{A}^\top$ can be represented as

$$\mathbf{A}\mathbf{A}^\top = \sum_{k=1}^{s+1} \mu_k^2 \mathbf{P}_k^{uu}$$

where the spectral projector $\mathbf{P}_k^{uu} := \sum_{j \in \Delta_k} \mathbf{u}_j \otimes \mathbf{u}_j$ which is uniquely defined. Correspondingly, define the empirical spectral projector based on eigen-decomposition of $\mathbf{Y}\mathbf{Y}^\top$,

$$\hat{\mathbf{P}}_k^{uu} := \sum_{j \in \Delta_k} \hat{\mathbf{u}}_j \otimes \hat{\mathbf{u}}_j.$$

We develop a sharp concentration bound for bilinear forms $\langle \hat{\mathbf{P}}_k^{uu} \mathbf{x}, \mathbf{y} \rangle$ for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{m_1}$. Observe that $\mathbf{Y}\mathbf{Y}^\top$ has an identical eigen-space as $\mathbf{Y}\mathbf{Y}^\top - m_2\sigma^2\mathbf{I}_{m_1}$. Let $\hat{\mathbf{\Gamma}} := \mathbf{\Gamma} - m_2\sigma^2\mathbf{I}_{m_1}$ and the spectral analysis shall be realized on $\mathbf{A}\mathbf{A}^\top + \hat{\mathbf{\Gamma}}$.

Several preliminary facts are introduced as follows. It is clear that the k-th eigengap is $\bar{g}_k(\mathbf{A}\mathbf{A}^\top) := \min\left(\mu_{k-1}^2 - \mu_k^2, \mu_k^2 - \mu_{k+1}^2\right)$ for $1 \le k \le s$, where we set $\mu_0 = +\infty$. The proof of Lemma 14 is provided in the Appendix.

Lemma 14 *For any deterministic matrix $\mathbf{B} \in \mathbb{R}^{m_3 \times m_2}$, the following bounds hold*

$$\begin{aligned} \mathbb{E}\|\mathbf{B}\mathbf{Z}^\top\| &\lesssim \sigma\|\mathbf{B}\|\left(m_1^{1/2} + m_3^{1/2} + (m_1 m_3)^{1/4}\right) \\ \left\|\mathbb{E}\mathbf{Z}\mathbf{Z}^\top - m_2\sigma^2\mathbf{I}_{m_1}\right\| &\lesssim \sigma^2 (m_1 m_2)^{1/2}. \end{aligned} \tag{22}$$

For any $t>0$, the following inequalities hold with probability at least $1-e^{-t}$,

$$\|\mathbf{B}\mathbf{Z}^\top\| \lesssim \sigma\|\mathbf{B}\|\Big(m_1^{1/2}+m_3^{1/2}+(m_1m_3)^{1/4}+t^{1/2}+(m_1t)^{1/4}\Big) \tag{23}$$
$$\big\|\mathbf{Z}\mathbf{Z}^\top-m_2\sigma^2\mathbf{I}_{m_1}\big\| \lesssim \sigma^2 m_2^{1/2}\big(m_1^{1/2}+t^{1/2}\big).$$

6.1. Proof of Theorem 1

To this end, define

$$\mathbf{C}_k^{uu} := \sum_{s\neq k}\frac{1}{\mu_s^2-\mu_k^2}\mathbf{P}_s^{uu}$$

and

$$\mathbf{P}_k^{hh} := \sum_{j\in\Delta_k}\mathbf{h}_j\otimes\mathbf{h}_j.$$

Theorem 1 is decomposed into two separate parts. Theorem 15 provides the concentration bound for $\big|\langle\mathbf{P}_k\mathbf{x},\mathbf{y}\rangle-\mathbb{E}\langle\mathbf{P}_k\mathbf{x},\mathbf{y}\rangle\big|$ by Gaussian isoperimetric inequality and the proof is postponed to the Appendix. In Theorem 17, we characterize the bias $\mathbb{E}\hat{\mathbf{P}}_k^{uu}-\mathbf{P}_k^{uu}$.

Theorem 15 *Let $\delta(m_1,m_2) := \mu_1\sigma m_1^{1/2}+\sigma^2(m_1m_2)^{1/2}$ and suppose that $\bar{g}_k(\mathbf{A}\mathbf{A}^\top)\geq D_1\delta(m_1,m_2)$ for a large enough constant $D_1>0$. Then, for any $\mathbf{x},\mathbf{y}\in\mathbb{R}^{m_1}$, there exists an absolute constant $D_2>0$ such that for all $\log 8\leq t\lesssim m_1$, the following bound holds with probability at least $1-e^{-t}$,*

$$\big|\langle\hat{\mathbf{P}}_k^{uu}\mathbf{x},\mathbf{y}\rangle-\mathbb{E}\langle\hat{\mathbf{P}}_k^{uu}\mathbf{x},\mathbf{y}\rangle\big| \leq D_2t^{1/2}\bigg(\frac{\sigma\mu_1+\sigma^2m_2^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\bigg)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.$$

The following spectral representation formula is needed whose proof can be found in Koltchinskii and Lounici (2016).

Lemma 16 *The following bound holds*

$$\|\hat{\mathbf{P}}_k^{uu}-\mathbf{P}_k^{uu}\| \leq \frac{4\|\hat{\mathbf{\Gamma}}\|}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}.$$

Moreover, $\hat{\mathbf{P}}_k^{uu}$ can be represented as

$$\hat{\mathbf{P}}_k^{uu}-\mathbf{P}_k^{uu} = \mathbf{L}_k(\hat{\mathbf{\Gamma}})+\mathbf{S}_k(\hat{\mathbf{\Gamma}})$$

where $\mathbf{L}_k(\hat{\mathbf{\Gamma}})=\mathbf{P}_k^{uu}\hat{\mathbf{\Gamma}}\mathbf{C}_k^{uu}+\mathbf{C}_k^{uu}\hat{\mathbf{\Gamma}}\mathbf{P}_k^{uu}$ and

$$\|\mathbf{S}_k(\hat{\mathbf{\Gamma}})\| \leq 14\bigg(\frac{\|\hat{\mathbf{\Gamma}}\|}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\bigg)^2.$$

Theorem 17 *Let $\delta(m_1,m_2) := \mu_1\sigma m_1^{1/2}+\sigma^2(m_1m_2)^{1/2}$ and suppose that $\bar{g}_k(\mathbf{A}\mathbf{A}^\top)\geq D_1\delta(m_1,m_2)$ for a large enough constant $D_1>0$ and $m_2e^{-m_1/2}\leq 1$. Then there exists an absolute constant $D_2>0$ such that*

$$\big\|\mathbb{E}\hat{\mathbf{P}}_k^{uu}-\mathbf{P}_k^{uu}-\mathbf{P}_k^{uu}\big(\mathbb{E}\hat{\mathbf{P}}_k^{uu}-\mathbf{P}_k^{uu}\big)\mathbf{P}_k^{uu}\big\| \leq D_2\nu_k\frac{\sigma^2m_1+\sigma^2m_2^{1/2}+\sigma\mu_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\bigg(\frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\bigg).$$

Proof [Proof of Theorem 1] Combining Theorem 15 and Theorem 17, we conclude that for any $\mathbf{x},\mathbf{y}\in\mathbb{R}^{m_1}$ with probability at least $1-e^{-t}$ for all $\log 8\leq t\leq m_1$,

$$\begin{aligned}&\big|\langle\hat{\mathbf{P}}_k^{uu}\mathbf{x},\mathbf{y}\rangle-\langle\mathbf{P}_k^{uu}\mathbf{x},\mathbf{y}\rangle-\langle\mathbf{P}_k^{uu}(\mathbb{E}\hat{\mathbf{P}}_k^{uu}-\mathbf{P}_k^{uu})\mathbf{P}_k^{uu}\mathbf{x},\mathbf{y}\rangle\big|\\ &\qquad\lesssim\bigg(t^{1/2}\frac{\sigma\mu_1+\sigma^2m_2^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}+\frac{\sigma^2m_1\delta(m_1,m_2)}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\bigg)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\end{aligned}$$

where we used the fact $\frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\leq 1$ and $\nu_k=1$ (since $\bar{g}_k(\mathbf{M}\mathbf{M}^\top)>0$). Since $\nu_k=1$ such that $\mathbf{P}_k^{uu}=\mathbf{u}_k\otimes\mathbf{u}_k$ and $\hat{\mathbf{P}}_k^{uu}=\hat{\mathbf{u}}_k\otimes\hat{\mathbf{u}}_k$, we can write

$$\mathbf{P}_k^{uu}(\mathbb{E}\hat{\mathbf{P}}_k^{uu}-\mathbf{P}_k^{uu})\mathbf{P}_k^{uu} = b_k\mathbf{P}_k^{uu}$$

where

$$b_k = \mathbb{E}\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle^2-1\in[-1,0].$$

Moreover, a simple fact is $b_k\leq\mathbb{E}\|\hat{\mathbf{P}}_k^{uu}-\mathbf{P}_k^{uu}\|\lesssim\frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}$ by Wedin's sinΘ theorem (Wedin (1972)). If $\bar{g}_k(\mathbf{A}\mathbf{A}^\top)\geq D\delta(m_1,m_2)$ for a large enough constant $D>0$, we can ensure $b_k\in[-1/2,0]$. Then, with probability at least $1-e^{-t}$,

$$\big|\big\langle\big(\hat{\mathbf{P}}_k^{uu}-(1+b_k)\mathbf{P}_k^{uu}\big)\mathbf{x},\mathbf{y}\big\rangle\big|\lesssim\bigg(t^{1/2}\frac{\sigma\mu_1+\sigma^2m_2^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}+\frac{\sigma^2m_1\delta(m_1,m_2)}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\bigg)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.$$

By choosing $\mathbf{x}=\mathbf{y}=\mathbf{u}_k$, we obtain for all $\log 8\leq t\leq m_1$,

$$\mathbb{P}\bigg(\big|\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle^2-(1+b_k)\big|\gtrsim t^{1/2}\frac{\sigma\mu_1+\sigma^2m_2^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}+\frac{\sigma^2m_1\delta(m_1,m_2)}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\bigg)\leq e^{-t}.$$

Denote this event by $\mathcal{E}_1$. Observe that if the constant $D>0$ is large enough and $m_1\ll m_2$, we conclude that on event $\mathcal{E}_1$, $\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle^2\geq\frac{1}{4}$. Then, on event $\mathcal{E}_1$,

$$\begin{aligned}&\big|\langle\hat{\mathbf{u}}_k,\mathbf{x}\rangle-\sqrt{1+b_k}\langle\mathbf{u}_k,\mathbf{x}\rangle\big|\\ &\leq\Big|\frac{1+b_k}{\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle}-\sqrt{1+b_k}\Big|\big|\langle\mathbf{u}_k,\mathbf{x}\rangle\big|\\ &+\frac{1}{\big|\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle\big|}\Big|\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle\langle\hat{\mathbf{u}}_k,\mathbf{x}\rangle-(1+b_k)\langle\mathbf{u}_k,\mathbf{x}\rangle\Big|\\ &=\frac{\sqrt{1+b_k}\big|1+b_k-\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle^2\big|\big|\langle\mathbf{u}_k,\mathbf{x}\rangle\big|}{\big|\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle\big|\big(\sqrt{1+b_k}+\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle\big)}+\frac{1}{\big|\langle\hat{\mathbf{u}}_k,\mathbf{u}_k\rangle\big|}\big|\big\langle\big(\hat{\mathbf{P}}_k^{uu}-(1+b_k)\mathbf{P}_k^{uu}\big)\mathbf{u}_k,\mathbf{x}\big\rangle\big|\\ &\lesssim t^{1/2}\frac{\sigma\mu_1+\sigma^2m_2^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}+\frac{\sigma^2m_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\bigg(\frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\bigg)\|\mathbf{x}\|_{\ell_2},\end{aligned}$$

which concludes the proof after replacing $\mathbf{A}$ with $\mathcal{M}_1(\mathbf{A})$ and μ_1 with $\|\mathcal{M}_1(\mathbf{A})\|$. ■

Proof [Proof of Theorem 17] Recall the representation formula of $\hat{\mathbf{P}}_k^{uu}$ in Lemma 16 that

$$\mathbb{E}\hat{\mathbf{P}}_k^{uu} = \mathbf{P}_k^{uu}+\mathbb{E}\mathbf{S}_k(\hat{\mathbf{\Gamma}})$$

where $\hat{\mathbf{\Gamma}} := \mathbf{A}\mathbf{Z}^\top + \mathbf{Z}\mathbf{A}^\top + \mathbf{Z}\mathbf{Z}^\top - m_2\sigma^2\mathbf{I}_{m_1}$. To this end, define

$$\widetilde{\mathbf{\Gamma}} := \hat{\mathbf{\Gamma}} - \big(\mathbf{Z}\mathbf{P}_k^{hh}\mathbf{Z}^\top - \nu_k\sigma^2\mathbf{I}_{m_1}\big)$$

such that we can write $\mathbb{E}\hat{\mathbf{P}}_k^{uu} = \mathbf{P}_k^{uu} + \mathbb{E}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) + \Big(\mathbb{E}\mathbf{S}_k(\hat{\mathbf{\Gamma}}) - \mathbb{E}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\Big)$. We derive an upper bound on $\big\|\mathbb{E}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbb{E}\mathbf{S}_k(\hat{\mathbf{\Gamma}})\big\|$ and the proof can be found in the Appendix. Lemma 18 implies that our analysis can be proceeded by replacing $\hat{\mathbf{\Gamma}}$ with $\widetilde{\mathbf{\Gamma}}$.

Lemma 18 *There exists a universal constant $D_1 > 0$ such that if $m_2 e^{-m_1/2} \le 1$, then*

$$\big\|\mathbb{E}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbb{E}\mathbf{S}_k(\hat{\mathbf{\Gamma}})\big\| \le D_1\frac{\sigma\mu_1 + \sigma^2 m_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big).$$

Let $\delta_t = \mathbb{E}\|\hat{\mathbf{\Gamma}}\| + D_1\sigma\mu_1 t^{1/2} + D_2\sigma^2 m_2^{1/2}t^{1/2}$ for $0 < t \le m_1$ to be determined later and large enough constants $D_1, D_2 > 0$ such that $\mathbb{P}\big(\|\hat{\mathbf{\Gamma}}\| \ge \delta_t\big) \le e^{-t}$. We write

$$\begin{aligned}
&\mathbb{E}\hat{\mathbf{P}}_k^{uu} - \mathbf{P}_k^{uu} - \mathbf{P}_k^{uu}\mathbb{E}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}\\
&\quad= \mathbb{E}\mathbf{S}_k(\hat{\mathbf{\Gamma}}) - \mathbb{E}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\\
&\quad+ \mathbb{E}\Big(\mathbf{P}_k^{uu}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp + (\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu} + (\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp\Big)\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\big)\\
&\quad+ \mathbb{E}\Big(\mathbf{P}_k^{uu}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp + (\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu} + (\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp\Big)\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\| > \delta_t\big). \quad (24)
\end{aligned}$$

We prove an upper bound for $\mathbb{E}\langle\mathbf{x}, (\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}\mathbf{y}\rangle\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\big)$ for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{m_1}$. Similar to the approach in Koltchinskii and Xia (2016), under the assumption $\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t$, $\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})$ is represented in the following analytic form,

$$\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) = -\frac{1}{2\pi i}\oint_{\gamma_k}\sum_{r\ge 2}(-1)^r\Big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\Big)^r\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)d\eta$$

where γ_k is a circle on the complex plane with center μ_k^2 and radius $\frac{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}{2}$, and $\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)$ is the resolvent of the operator $\mathbf{A}\mathbf{A}^\top$ with $\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta) = (\mathbf{A}\mathbf{A}^\top - \eta\mathbf{I}_{m_1})^{-1}$ which can be explicitly written as

$$\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta) := (\mathbf{A}\mathbf{A}^\top - \eta\mathbf{I}_{m_1})^{-1} = \sum_s\frac{1}{\mu_s^2 - \eta}\mathbf{P}_s^{uu}.$$

We also denote

$$\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta) := \mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta) - \frac{1}{\mu_k^2 - \eta}\mathbf{P}_k^{uu} = \sum_{s\ne k}\frac{1}{\mu_s^2 - \eta}\mathbf{P}_s^{uu}.$$

It is easy to check that

$$\begin{aligned}
&(\mathbf{P}_k^{uu})^\perp\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^r\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{P}_k^{uu}\\
&\quad= (\mathbf{P}_k^{uu})^\perp\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^r\frac{1}{\mu_k^2 - \eta}\mathbf{P}_k^{uu}\\
&\quad= \Big(\frac{1}{(\mu_k^2 - \eta)^2}\sum_{s=2}^r\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-1}\big(\mathbf{P}_k^{uu}\widetilde{\mathbf{\Gamma}}\big)\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\Big)\\
&\quad+ \frac{1}{\mu_k^2 - \eta}\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^r\mathbf{P}_k^{uu},
\end{aligned}$$

where we used the formula $(a+b)^r = b^r + \sum_{s=1}^r b^{s-1}a(a+b)^{r-s}$. As a result,

$$\begin{aligned}
&(\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}\\
&= -\sum_{r\ge 2}(-1)^r\frac{1}{2\pi i}\oint_{\gamma_k}\Big(\frac{1}{(\mu_k^2 - \eta)^2}\sum_{s=2}^r\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-1}\big(\mathbf{P}_k^{uu}\widetilde{\mathbf{\Gamma}}\big)\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\\
&\quad+ \frac{1}{\mu_k^2 - \eta}\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^r\mathbf{P}_k^{uu}\Big)d\eta. \quad (25)
\end{aligned}$$

For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{m_1}$, we shall derive an upper bound for

$$\mathbb{E}\Big\langle\mathbf{x}, \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-1}\big(\mathbf{P}_k^{uu}\widetilde{\mathbf{\Gamma}}\big)\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\mathbf{y}\Big\rangle\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\big), \quad s = 2, \ldots, r.$$

Recall that $\mathrm{rank}(\mathbf{P}_k^{uu}) = \nu_k$ and $\mathbf{P}_k^{uu} = \sum_{j\in\Delta_k}\mathbf{u}_j \otimes \mathbf{u}_j$. Then,

$$\begin{aligned}
&\Big\langle\mathbf{x}, \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-1}\big(\mathbf{P}_k^{uu}\widetilde{\mathbf{\Gamma}}\big)\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\mathbf{y}\Big\rangle\\
&\qquad= \sum_{j\in\Delta_k}\Big\langle\mathbf{x}, \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-1}\big(\mathbf{u}_j \otimes \mathbf{u}_j\widetilde{\mathbf{\Gamma}}\big)\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\mathbf{y}\Big\rangle\\
&\qquad= \sum_{j\in\Delta_k}\big\langle\widetilde{\mathbf{\Gamma}}\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\mathbf{y}, \mathbf{u}_j\big\rangle\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\mathbf{u}_j, \mathbf{x}\big\rangle.
\end{aligned}$$

Observe that

$$\begin{aligned}
\big|\big\langle\widetilde{\mathbf{\Gamma}}\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\mathbf{y}, \mathbf{u}_j\big\rangle\big| \le&\|\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\|^{r-s}\|\widetilde{\mathbf{\Gamma}}\|^{r-s+1}\|\mathbf{y}\|_{\ell_2}\\
\le&\Big(\frac{2}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{(r-s)}\|\widetilde{\mathbf{\Gamma}}\|^{r-s+1}\|\mathbf{y}\|_{\ell_2}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\mathbb{E}\Big\langle\mathbf{x}, \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-1}\big(\mathbf{P}_k^{uu}\widetilde{\mathbf{\Gamma}}\big)\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\mathbf{y}\Big\rangle\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big)\\
&= \sum_{j\in\Delta_k}\mathbb{E}\big\langle\widetilde{\mathbf{\Gamma}}\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\mathbf{y}, \mathbf{u}_j\big\rangle\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-1}\mathbf{u}_j, \mathbf{x}\big\rangle\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big)\\
&\le \sum_{j\in\Delta_k}\mathbb{E}^{1/2}\Big|\big\langle\widetilde{\mathbf{\Gamma}}\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\mathbf{y}, \mathbf{u}_j\big\rangle\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big)\Big|^2\\
&\qquad\qquad\times\mathbb{E}^{1/2}\Big|\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-1}\mathbf{u}_j, \mathbf{x}\big\rangle\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big)\Big|^2\\
&\le \Big(\frac{2\delta_t}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{r-s}\delta_t\|\mathbf{y}\|_{\ell_2}\sum_{j\in\Delta_k}\mathbb{E}^{1/2}\Big|\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\mathbf{u}_j, \mathbf{x}\big\rangle\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big)\Big|^2. \quad (26)
\end{aligned}$$

It then remains to bound, for each $j \in \Delta_k$,

$$\mathbb{E}^{1/2}\Big|\Big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\mathbf{u}_j, \mathbf{x}\Big\rangle\Big|^2\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big).$$

Recall that we can write

$$\widetilde{\mathbf{\Gamma}} = \mathbf{A}\mathbf{Z}^\top + \mathbf{Z}\mathbf{A}^\top + \mathbf{Z}\sum_{k'\neq k}\mathbf{P}^{hh}_{k'}\mathbf{Z}^\top - \sigma^2(m_2-\nu_k)\mathbf{I}_{m_1}$$

and correspondingly

$$\widetilde{\mathbf{\Gamma}}\mathbf{u}_j = \mathbf{A}\mathbf{Z}^\top\mathbf{u}_j + \mathbf{Z}\mathbf{A}^\top\mathbf{u}_j + \mathbf{Z}\sum_{k'\neq k}\mathbf{P}^{hh}_{k'}\mathbf{Z}^\top\mathbf{u}_j - \sigma^2(m_2-\nu_k)\mathbf{u}_j.$$

We write

$$\begin{aligned}
&\Big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\mathbf{u}_j, \mathbf{x}\Big\rangle \\
&\quad = \Big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{Z}\mathbf{A}^\top\mathbf{u}_j, \mathbf{x}\Big\rangle && (27)\\
&\quad + \Big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{A}\mathbf{Z}^\top\mathbf{u}_j, \mathbf{x}\Big\rangle && (28)\\
&\quad + \Big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\Big(\mathbf{Z}\sum_{k'\neq k}\mathbf{P}^{hh}_{k'}\mathbf{Z}^\top\mathbf{u}_j - \sigma^2(m_2-\nu_k)\mathbf{u}_j\Big), \mathbf{x}\Big\rangle. && (29)
\end{aligned}$$

The upper bounds of (27), (28), and (29) shall be obtained separately via different representations.

Bound of $\mathbb{E}^{1/2}\big|\big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{Z}\mathbf{A}^\top\mathbf{u}_j, \mathbf{x}\big\rangle\big|^2\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big)$. Observe that $\mathbf{A}^\top\mathbf{u}_j = \mu_k\mathbf{h}_j \in \mathbb{R}^{m_2}$ for $j \in \Delta_k$ such that

$$\mathbf{Z}\mathbf{A}^\top\mathbf{u}_j = \mu_k\mathbf{Z}\mathbf{h}_j = \mu_k\sum_{i=1}^{m_1}\langle \mathbf{z}_i, \mathbf{h}_j\rangle\mathbf{e}_i$$

where $\{\mathbf{e}_1, \ldots, \mathbf{e}_{m_1}\}$ denote the canonical basis vectors in $\mathbb{R}^{m_1}$ and $\{\mathbf{z}_1^\top, \ldots, \mathbf{z}_{m_1}^\top\}$ denote the rows of $\mathbf{Z}$. Therefore,

$$\begin{aligned}
&\big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{Z}\mathbf{A}^\top\mathbf{u}_j, \mathbf{x}\big\rangle \\
&\qquad = \mu_k\sum_{i=1}^{m_1}\langle \mathbf{z}_i, \mathbf{h}_j\rangle\big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{e}_i, \mathbf{x}\big\rangle.
\end{aligned}$$

It is clear that $\langle \mathbf{z}_i, \mathbf{h}_j\rangle, i = 1, \ldots, m_1$ are i.i.d. and $\langle \mathbf{z}_i, \mathbf{h}_j\rangle \sim \mathcal{N}(0, \sigma^2)$. Recall that $\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta) = \sum_{k'\neq k}\frac{\mathbf{P}^{uu}_{k'}}{\mu_{k'}^2-\eta}$, implying that $\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)$ can be viewed as a linear combination of operators

$$(\mathbf{P}^{uu}_{t_1}\widetilde{\mathbf{\Gamma}}\mathbf{P}^{uu}_{t_2})(\mathbf{P}^{uu}_{t_2}\widetilde{\mathbf{\Gamma}}\mathbf{P}^{uu}_{t_3})\ldots(\mathbf{P}^{uu}_{t_{s-2}}\widetilde{\mathbf{\Gamma}}\mathbf{P}^{uu}_{t_{s-1}})$$

where $t_1, \ldots, t_{s-1} \neq k$. For each $\mathbf{P}^{uu}_{t_1}\widetilde{\mathbf{\Gamma}}\mathbf{P}^{uu}_{t_2}$, we have

$$\mathbf{P}^{uu}_{t_1}\widetilde{\mathbf{\Gamma}}\mathbf{P}^{uu}_{t_2} = \mathbf{P}^{uu}_{t_1}\mathbf{A}\mathbf{Z}^\top\mathbf{P}^{uu}_{t_2} + \mathbf{P}^{uu}_{t_1}\mathbf{Z}\mathbf{A}^\top\mathbf{P}^{uu}_{t_2} + \mathbf{P}^{uu}_{t_1}\big(\mathbf{Z}\sum_{k'\neq k}\mathbf{P}^{hh}_{k'}\mathbf{Z}^\top\big)\mathbf{P}^{uu}_{t_2} - \sigma^2(m_2-\nu_k)\mathbf{P}^{uu}_{t_1}\mathbf{P}^{uu}_{t_2}.$$

Clearly, $\mathbf{P}^{uu}_{t_1}\mathbf{A}\mathbf{Z}^\top$ is a function of random vectors $\mathbf{P}^{uu}_{t_1}\mathbf{A}\mathbf{z}_i, i = 1, \ldots, m_1$; $\mathbf{Z}\mathbf{A}^\top\mathbf{P}^{uu}_{t_2}$ is a function of random vectors $\mathbf{P}^{uu}_{t_2}\mathbf{A}\mathbf{z}_i, i = 1, \ldots, m_1$; $\mathbf{Z}\sum_{k'\neq k}\mathbf{P}^{hh}_{k'}\mathbf{Z}^\top = \mathbf{Z}\sum_{k'\neq k}(\mathbf{P}^{hh}_{k'})^2\mathbf{Z}^\top$ is a function of random vectors $\mathbf{P}^{hh}_{k'}\mathbf{z}_i, i = 1, \ldots, m_1$. The following facts are obvious

$$\mathbb{E}\langle \mathbf{z}_i, \mathbf{h}_j\rangle\mathbf{P}^{uu}_{t_1}\mathbf{A}\mathbf{z}_i = \mathbf{P}^{uu}_{t_1}\mathbf{A}(\mathbb{E}\mathbf{z}_i\otimes\mathbf{z}_i)\mathbf{h}_j = \sigma^2\mathbf{P}^{uu}_{t_1}\mathbf{A}\mathbf{h}_j = \sigma^2\mu_k\mathbf{P}^{uu}_{t_1}\mathbf{u}_j = \mathbf{0}, \quad \forall t_1 \neq k$$

and

$$\mathbb{E}\langle \mathbf{z}_i, \mathbf{h}_j\rangle\mathbf{P}^{hh}_{k'}\mathbf{z}_i = \mathbf{P}^{hh}_{k'}(\mathbb{E}\mathbf{z}_i\otimes\mathbf{z}_i)\mathbf{h}_j = \sigma^2\mathbf{P}^{hh}_{k'}\mathbf{h}_j = \mathbf{0}, \quad \forall k' \neq k.$$

Since $\big\{\langle \mathbf{z}_i, \mathbf{h}_j\rangle, i = 1, \ldots, m_1\big\}$ are Gaussian random variables and $\big\{\mathbf{P}^{uu}_{t_1}\mathbf{A}\mathbf{z}_i, \mathbf{P}^{hh}_{k'}\mathbf{z}_i, i = 1, \ldots, m_1\big\}$ are (complex) Gaussian random vectors, uncorrelations indicate that $\big\{\langle \mathbf{z}_i, \mathbf{h}_j\rangle : i = 1, \ldots, m_1\big\}$ are independent with $\big\{\mathbf{P}^{uu}_{t_1}\mathbf{A}\mathbf{z}_i, \mathbf{P}^{hh}_{k'}\mathbf{z}_i : t_1 \neq k, k' \neq k, i = 1, \ldots, m_1\big\}$. We conclude that $\big\{\langle \mathbf{z}_i, \mathbf{h}_j\rangle : i = 1, \ldots, m_1\big\}$ are independent with

$$\big\{\big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{e}_i, \mathbf{x}\big\rangle, i = 1, \ldots, m_1\big\}.$$

To this end, define the complex random variables

$$\omega_i(\mathbf{x}) = \big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{e}_i, \mathbf{x}\big\rangle = \omega_i^{(1)}(\mathbf{x}) + \omega_i^{(2)}(\mathbf{x})\mathrm{Im} \in \mathbb{C}, \quad i = 1, \ldots, m_1$$

where Im denotes the imaginary number. Then,

$$\begin{aligned}
&\big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{Z}\mathbf{A}^\top\mathbf{u}_j, \mathbf{x}\big\rangle \\
&\qquad = \mu_k\sum_{i=1}^{m_1}\langle \mathbf{z}_i, \mathbf{h}_j\rangle\omega_i^{(1)}(\mathbf{x}) + \Big(\mu_k\sum_{i=1}^{m_1}\langle \mathbf{z}_i, \mathbf{h}_j\rangle\omega_i^{(2)}(\mathbf{x})\Big)\mathrm{Im} \\
&\qquad =: \kappa_1(\mathbf{x}) + \kappa_2(\mathbf{x})\mathrm{Im} \in \mathbb{C}.
\end{aligned}$$

Conditioned on $\big\{\mathbf{P}^{uu}_{t_1}\mathbf{A}\mathbf{z}_i, \mathbf{P}^{hh}_{k'}\mathbf{z}_i : t_1 \neq k, k' \neq k, i = 1, \ldots, m_1\big\}$, we get

$$\mathbb{E}\kappa_1^2(\mathbf{x}) = \mu_k^2\sigma^2\sum_{i=1}^{m_1}\Big(\omega_i^{(1)}(\mathbf{x})\Big)^2$$

and

$$\mathbb{E}\kappa_1(\mathbf{x})\kappa_2(\mathbf{x}) = \mu_k^2\sigma^2\sum_{i=1}^{m_1}\omega_i^{(1)}(\mathbf{x})\omega_i^{(2)}(\mathbf{x})$$

implying that the centered Gaussian random vector $(\kappa_1(\mathbf{x}), \kappa_2(\mathbf{x}))$ has covariance matrix:

$$\Big(\mu_k^2\sigma^2\sum_{i=1}^{m_1}\omega_i^{(k_1)}(\mathbf{x})\omega_i^{(k_2)}(\mathbf{x})\Big)_{k_1,k_2=1,2}.$$

Finally,

$$\begin{aligned}
&\mathbb{E}^{1/2}\big|\big\langle \big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{Z}\mathbf{A}^\top\mathbf{u}_j, \mathbf{x}\big\rangle\big|^2\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big) \\
&\quad = \mathbb{E}^{1/2}\big(\kappa_1^2(\mathbf{x}) + \kappa_2^2(\mathbf{x})\big)\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big) \\
&\quad = \sigma\mu_k\mathbb{E}^{1/2}\Big(\sum_{i=1}^{m_1}\big(\omega_i^{(1)}(\mathbf{x})\big)^2 + \big(\omega_i^{(2)}(\mathbf{x})\big)^2\Big)\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big) \\
&\quad = \sigma\mu_k\mathbb{E}^{1/2}\sum_{i=1}^{m_1}\big|\omega_i(\mathbf{x})\big|^2\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le \delta_t\Big).
\end{aligned}$$

Moreover,

$$\sum_{i=1}^{m_1}\big|\omega_i(\mathbf{x})\big|^2=\sum_{i=1}^{m_1}\big|\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\mathbf{x},\mathbf{e}_j\big\rangle\big|^2\le\big\|\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\mathbf{x}\big\|_{\ell_2}^2$$
$$\le\|\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\|^{2(s-1)}\|\widetilde{\mathbf{\Gamma}}\|^{2(s-2)}\|\mathbf{x}\|_{\ell_2}^2\le\Big(\frac{2}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{2(s-1)}\|\widetilde{\mathbf{\Gamma}}\|^{2(s-2)}\|\mathbf{x}\|_{\ell_2}^2.$$

As a result,

$$\mathbb{E}^{1/2}\big|\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{Z}\mathbf{A}^\top\mathbf{u}_j,\mathbf{x}\big\rangle\big|^2\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\|\le\delta_t\Big)$$
$$\le\sigma\mu_k\mathbb{E}^{1/2}\Big(\frac{2}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{2(s-1)}\|\widetilde{\mathbf{\Gamma}}\|^{2(s-2)}\|\mathbf{x}\|_{\ell_2}^2\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\|\le\delta_t\Big)$$
$$\le\frac{\sigma\mu_k}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta_t}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{s-2}\|\mathbf{x}\|_{\ell_2}.$$

Bound of $\mathbb{E}^{1/2}\big|\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{A}\mathbf{Z}^\top\mathbf{u}_j,\mathbf{x}\big\rangle\big|^2\mathbf{1}(\|\widetilde{\mathbf{\Gamma}}\|\le\delta_t)$. With a little abuse on the notations, we denote by $\mathbf{z}_1,\ldots,\mathbf{z}_{m_2}\in\mathbb{R}^{m_1}$ the corresponding columns of $\mathbf{Z}$ in this paragraph. Then,

$$\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{A}\mathbf{Z}^\top\mathbf{u}_j,\mathbf{x}\big\rangle=\sum_{i=1}^{m_2}\langle\mathbf{z}_i,\mathbf{u}_j\rangle\Big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{A}\mathbf{e}_i,\mathbf{x}\Big\rangle.$$

Similarly, $\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)$ can be represented as linear combination of operators

$$\big(\mathbf{P}_{t_1}^{uu}\widetilde{\mathbf{\Gamma}}\mathbf{P}_{t_2}^{uu}\big)\big(\mathbf{P}_{t_2}^{uu}\widetilde{\mathbf{\Gamma}}\mathbf{P}_{t_3}^{uu}\big)\ldots\big(\mathbf{P}_{t_{s-2}}^{uu}\widetilde{\mathbf{\Gamma}}\mathbf{P}_{t_{s-1}}^{uu}\big),\quad t_1,\ldots,t_{s-1}\ne k.$$

To this end, we write

$$\mathbf{P}_{t_1}^{uu}\widetilde{\mathbf{\Gamma}}\mathbf{P}_{t_2}^{uu}=\mathbf{P}_{t_1}^{uu}\mathbf{A}\mathbf{Z}^\top\mathbf{P}_{t_2}^{uu}+\mathbf{P}_{t_1}^{uu}\mathbf{Z}\mathbf{A}^\top\mathbf{P}_{t_2}^{uu}+\mathbf{P}_{t_1}^{uu}\big(\mathbf{Z}\sum_{k'\ne k}\mathbf{P}_{k'}^{hh}\mathbf{Z}^\top\big)\mathbf{P}_{t_2}^{uu}-\sigma^2(m_2-\nu_k)\mathbf{P}_{t_1}^{uu}\mathbf{P}_{t_2}^{uu}.$$

Observe that $\mathbf{P}_{t_1}^{uu}\mathbf{A}\mathbf{Z}^\top\mathbf{P}_{t_2}^{uu}$, $\mathbf{P}_{t_1}^{uu}\mathbf{Z}\mathbf{A}^\top\mathbf{P}_{t_2}^{uu}$ and $\mathbf{P}_{t_1}^{uu}\big(\mathbf{Z}\sum_{k'\ne k}\mathbf{P}_{k'}^{hh}\mathbf{Z}^\top\big)\mathbf{P}_{t_2}^{uu}$ are functions of random vectors $\{\mathbf{P}_{t_1}^{uu}\mathbf{z}_i,\mathbf{P}_{t_2}^{uu}\mathbf{z}_i:t_1,t_2\ne k,i=1,\ldots,m_2\}$. Moreover,

$$\mathbb{E}\langle\mathbf{z}_i,\mathbf{u}_j\rangle\mathbf{P}_{t_1}^{uu}\mathbf{z}_i=\mathbf{P}_{t_1}^{uu}\big(\mathbb{E}\mathbf{z}_i\otimes\mathbf{z}_i\big)\mathbf{u}_j=\sigma^2\mathbf{P}_{t_1}^{uu}\mathbf{u}_j=\mathbf{0},\quad\forall\ t_1\ne k$$

which implies that $\{\langle\mathbf{z}_i,\mathbf{u}_j\rangle:i=1,\ldots,m_2\}$ and $\Big\{\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{A}\mathbf{e}_i,\mathbf{x}\big\rangle:i=1,\ldots,m_2\Big\}$ are independent. Following an identical analysis as above, we get

$$\mathbb{E}^{1/2}\big|\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{A}\mathbf{Z}^\top\mathbf{u}_j,\mathbf{x}\big\rangle\big|^2\mathbf{1}(\|\widetilde{\mathbf{\Gamma}}\|\le\delta_t)\le\frac{\sigma\mu_1}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta_t}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{s-2}\|\mathbf{x}\|_{\ell_2}.$$

Bound of $\mathbb{E}^{1/2}\Big|\Big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\big(\mathbf{Z}\sum_{k'\ne k}\mathbf{P}_{k'}^{hh}\mathbf{Z}^\top\big)\mathbf{u}_j,\mathbf{x}\Big\rangle\Big|^2\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\|\le\delta_t\big)$. Note that we used the fact $\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{u}_j=\mathbf{0}$ in (29). Again, let $\{\mathbf{z}_1,\ldots,\mathbf{z}_{m_2}\}\subset\mathbb{R}^{m_1}$ denote the corresponding columns of $\mathbf{Z}$. We write

$$\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\big(\mathbf{Z}\sum_{k'\ne k}\mathbf{P}_{k'}^{hh}\mathbf{Z}^\top\big)\mathbf{u}_j,\mathbf{x}\big\rangle$$
$$=\sum_{i=1}^{m_2}\langle\mathbf{z}_i,\mathbf{u}_j\rangle\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{Z}\big(\sum_{k'\ne k}\mathbf{P}_{k'}^{hh}\big)\mathbf{e}_i,\mathbf{x}\big\rangle.$$

In a similar fashion, we show that $\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\mathbf{Z}$ is a function of random vectors $\big\{\mathbf{P}_t^{uu}\mathbf{z}_i:t\ne k,i=1,\ldots,m_2\big\}$ which are independent with $\big\{\langle\mathbf{z}_i,\mathbf{u}_j\rangle:i=1,\ldots,m_2\big\}$. Then,

$$\mathbb{E}^{1/2}\Big|\Big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\big(\mathbf{Z}\sum_{k'\ne k}\mathbf{P}_{k'}^{hh}\mathbf{Z}^\top\big)\mathbf{u}_j,\mathbf{x}\Big\rangle\Big|^2\mathbf{1}(\|\widetilde{\mathbf{\Gamma}}\|\le\delta)$$
$$\le\mathbb{E}^{1/2}\sigma^2\|\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\|^{2(s-1)}\|\widetilde{\mathbf{\Gamma}}\|^{2(s-2)}\|\mathbf{Z}\sum_{k'\ne k}\mathbf{P}_{k'}^{hh}\|^2\|\mathbf{x}\|_{\ell_2}^2\mathbf{1}(\|\widetilde{\mathbf{\Gamma}}\|\le\delta_t)$$
$$\lesssim\frac{\sigma^2m_2^{1/2}}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{\delta_t}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{s-2}\|\mathbf{x}\|_{\ell_2}.$$

where we used the fact $\mathbb{E}^{1/2}\big\|(\sum_{k'\ne k}\mathbf{P}_{k'}^{hh})\mathbf{Z}^\top\big\|^2\lesssim\sigma m_2^{1/2}$ from Lemma 14.

Finalize the proof of Theorem. Combining the above bounds into (28), (27) and (29), we conclude that

$$\mathbb{E}^{1/2}\Big|\big\langle\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-2}\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\mathbf{u}_j,\mathbf{x}\big\rangle\Big|^2\mathbf{1}(\|\widetilde{\mathbf{\Gamma}}\|\le\delta_t)$$
$$\lesssim\frac{\sigma^2m_2^{1/2}+\sigma\mu_1}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta_t}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{s-2}\|\mathbf{x}\|_{\ell_2}.$$

Continue from (26) and we end up with

$$\mathbb{E}\langle\mathbf{x},\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{s-1}(\mathbf{P}_k^{uu}\widetilde{\mathbf{\Gamma}})\big(\mathbf{R}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^{r-s}\mathbf{P}_k^{uu}\mathbf{y}\rangle\mathbf{1}(\|\widetilde{\mathbf{\Gamma}}\|\le\delta_t)$$
$$\lesssim\nu_k\delta_t\frac{\sigma^2m_2^{1/2}+\sigma\mu_1}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta_t}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{r-2}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.$$

Plug the bounds into (25),

$$\big|\mathbb{E}\big\langle(\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}\mathbf{y},\mathbf{x}\big\rangle\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\le\delta_t\|\big)\big|$$
$$\lesssim\sum_{r\ge2}\frac{\pi\bar g_k(\mathbf{A}\mathbf{A}^\top)}{2\pi}\Big(\frac{2}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^2(r-1)\nu_k\delta_t\frac{\sigma^2m_2^{1/2}+\sigma\mu_1}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta_t}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{r-2}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}$$
$$\le D_1\nu_k\frac{\sigma^2m_2^{1/2}+\sigma\mu_1}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\sum_{r\ge2}(r-1)\Big(\frac{2\delta_t}{\bar g_k(\mathbf{A}\mathbf{A}^\top)}\Big)^{r-1}$$

where we used the fact $\oint_{\gamma_k}\big(\widetilde{\mathbf{R}}_{\mathbf{A}\mathbf{A}^\top}(\eta)\widetilde{\mathbf{\Gamma}}\big)^r\mathbf{P}_k^{uu}d\eta=\mathbf{0}$. By the inequality $\sum_{r\geq 1}rq^r=\frac{q}{(1-q)^2},\forall q<1$ and the fact $D_1\delta_t\leq\bar{g}_k(\mathbf{A}\mathbf{A}^\top)$ for some large constant $D_1>0$ and $t\leq m_1$, we conclude with

$$\begin{aligned}&\big|\mathbb{E}\big\langle(\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}\mathbf{y},\mathbf{x}\big\rangle\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\leq\delta_t\|\big)\big|\\&\qquad\lesssim\nu_k\frac{\sigma^2m_2^{1/2}+\sigma\mu_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta_t}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2},\quad\forall\mathbf{x},\mathbf{y}\in\mathbb{R}^{m_1}\end{aligned}$$

implying that

$$\Big\|\mathbb{E}(\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\|\leq\delta_t\big)\Big\|\lesssim\nu_k\frac{\sigma^2m_2^{1/2}+\sigma\mu_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta_t}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big).$$

The same bound holds for

$$\big\|\mathbb{E}\mathbf{P}_k^{uu}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\|\leq\delta_t\big)\big\|\quad\text{and}\quad\big\|\mathbb{E}(\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\|\leq\delta_t\big)\big\|,$$

following the same arguments. As a result,

$$\begin{aligned}&\Big\|\mathbb{E}\Big((\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}+\mathbf{P}_k^{uu}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp+(\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp\Big)\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\|\leq\delta_t\big)\Big\|\\&\qquad\lesssim\nu_k\frac{\sigma^2m_2^{1/2}+\sigma\mu_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta_t}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big).\end{aligned}\tag{30}$$

By choosing $t=m_1$ such that $\mathbb{P}(\|\widetilde{\mathbf{\Gamma}}\|\geq\delta_{m_1})\leq e^{-m_1/2}$, we get

$$\begin{aligned}&\Big\|\mathbb{E}\Big((\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}+\mathbf{P}_k^{uu}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp+(\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp\Big)\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\|>\delta_{m_1}\big)\Big\|\\&\leq\mathbb{E}\Big\|\Big((\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}+\mathbf{P}_k^{uu}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp+(\mathbf{P}_k^{uu})^\perp\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})(\mathbf{P}_k^{uu})^\perp\Big)\Big\|\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\|>\delta_{m_1}\big)\\&\leq\mathbb{E}\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\|\mathbf{1}\big(\|\widetilde{\mathbf{\Gamma}}\|>\delta_{m_1}\big)\leq\mathbb{E}^{1/2}\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\|^2\mathbb{P}^{1/2}\big(\|\widetilde{\mathbf{\Gamma}}\|>\delta_{m_1}\big)\\&\lesssim\Big(\frac{\delta_{m_1}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)^2\mathbb{P}^{1/2}\big(\|\widetilde{\mathbf{\Gamma}}\|>\delta_{m_1}\big)\lesssim\Big(\frac{\delta_{m_1}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)^2e^{-m_1/2},\end{aligned}$$

which is clearly dominated by (30). Substitute the above bounds into (24) and we get

$$\begin{aligned}\Big\|\mathbb{E}\hat{\mathbf{P}}_k^{uu}-\mathbf{P}_k^{uu}-\mathbf{P}_k^{uu}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})\mathbf{P}_k^{uu}\Big\|&\leq\|\mathbb{E}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}})-\mathbf{S}_k(\hat{\mathbf{\Gamma}})\|+D_1\nu_k\frac{\sigma^2m_2^{1/2}+\sigma\mu_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)\\&\leq D_2\nu_k\frac{\sigma^2m_2^{1/2}+\sigma^2m_1+\sigma\mu_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{2\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big).\end{aligned}$$

■

6.2. Proof of Theorem 4

The proof of Theorem 4 is identical to the proof of Corollary 1.5 in Koltchinskii and Xia (2016) and will be skipped here.

6.3. Proof of Theorem 6

It suffices to prove the upper bound of $\big|\widetilde{A}(i,j,k)-A(i,j,k)\big|$ for $i\in[d_1],j\in[d_2],k\in[d_3]$. To this end, denote by $\mathbf{e}_i$ the i-th canonical basis vectors. Observe that

$$\begin{aligned}\big\langle\widetilde{\mathbf{A}}-\mathbf{A},\mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle=&\Big\langle\mathbf{A}\times_1\mathbf{P}_{\widetilde{\mathbf{U}}}\times_2\mathbf{P}_{\widetilde{\mathbf{V}}}\times_3\mathbf{P}_{\widetilde{\mathbf{W}}}-\mathbf{A},\mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\Big\rangle\\&+\Big\langle\mathbf{Z}\times_1\mathbf{P}_{\widetilde{\mathbf{U}}}\times_2\mathbf{P}_{\widetilde{\mathbf{V}}}\times_3\mathbf{P}_{\widetilde{\mathbf{W}}},\mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\Big\rangle.\end{aligned}$$

Some preliminary facts shall be concluded from Theorem 1. By Theorem 4, there exists an event $\mathcal{E}_2$ with $\mathbb{P}\big(\mathcal{E}_2\big)\geq 1-\frac{1}{d^2}$ on which

$$\big\|\mathbf{e}_i^\top\big(\widetilde{\mathbf{U}}-\mathbf{U}\big)\big\|_{\ell_2}\leq r^{1/2}\big\|\mathbf{e}_i^\top\big(\widetilde{\mathbf{U}}-\mathbf{U}\big)\big\|_{\ell_\infty}\lesssim\frac{\sigma\overline{\Lambda}(\mathbf{A})r^{1/2}+\sigma^2dr^{1/2}}{\bar{g}_{\min}^2(\mathbf{A})}\log^{1/2}d$$

and

$$\big\|\widetilde{\mathbf{U}}^\top\mathbf{U}-\mathbf{I}_{r_1}\big\|\leq\|\widetilde{\mathbf{U}}^\top\mathbf{U}-\mathbf{I}_{r_1}\|_{\mathrm{F}}\lesssim r\|\widetilde{\mathbf{U}}^\top\mathbf{U}-\mathbf{I}_{r_1}\|_{\ell_\infty}\lesssim\frac{\sigma\overline{\Lambda}(\mathbf{A})r+\sigma^2dr}{\bar{g}_{\min}^2(\mathbf{A})}\log^{1/2}d.$$

The following decomposition is straightforward,

$$\begin{aligned}&\mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}},\mathbf{P}_{\widetilde{\mathbf{V}}},\mathbf{P}_{\widetilde{\mathbf{W}}}\big)-\mathbf{A}\\&\quad=\mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}}-\mathbf{P}_{\mathbf{U}},\mathbf{P}_{\mathbf{V}},\mathbf{P}_{\mathbf{W}}\big)+\mathbf{A}\cdot\big(\mathbf{P}_{\mathbf{U}},\mathbf{P}_{\widetilde{\mathbf{V}}}-\mathbf{P}_{\mathbf{V}},\mathbf{P}_{\mathbf{W}}\big)\\&\quad+\mathbf{A}\cdot\big(\mathbf{P}_{\mathbf{U}},\mathbf{P}_{\mathbf{V}},\mathbf{P}_{\widetilde{\mathbf{W}}}-\mathbf{P}_{\mathbf{W}}\big)+\mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}}-\mathbf{P}_{\mathbf{U}},\mathbf{P}_{\widetilde{\mathbf{V}}}-\mathbf{P}_{\mathbf{V}},\mathbf{P}_{\mathbf{W}}\big)\\&\quad+\mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}}-\mathbf{P}_{\mathbf{U}},\mathbf{P}_{\mathbf{V}},\mathbf{P}_{\widetilde{\mathbf{W}}}-\mathbf{P}_{\mathbf{W}}\big)+\mathbf{A}\cdot\big(\mathbf{P}_{\mathbf{U}},\mathbf{P}_{\widetilde{\mathbf{V}}}-\mathbf{P}_{\mathbf{V}},\mathbf{P}_{\widetilde{\mathbf{W}}}-\mathbf{P}_{\mathbf{W}}\big)\\&\quad+\mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}}-\mathbf{P}_{\mathbf{U}},\mathbf{P}_{\widetilde{\mathbf{V}}}-\mathbf{P}_{\mathbf{V}},\mathbf{P}_{\widetilde{\mathbf{W}}}-\mathbf{P}_{\mathbf{W}}\big)\end{aligned}$$

Recall that $\mathbf{A}=\mathbf{C}\cdot(\mathbf{U},\mathbf{V},\mathbf{W})$ and we get

$$\begin{aligned}&\Big\langle\mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}}-\mathbf{P}_{\mathbf{U}},\mathbf{P}_{\mathbf{V}},\mathbf{P}_{\mathbf{W}}\big),\mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\Big\rangle\\&\qquad=\mathbf{e}_i^\top\Big(\widetilde{\mathbf{U}}\big(\widetilde{\mathbf{U}}^\top\mathbf{U}\big)-\mathbf{U}\Big)\mathcal{M}_1(\mathbf{C})\big(\mathbf{V}\otimes\mathbf{W}\big)^\top(\mathbf{e}_j\otimes\mathbf{e}_k).\end{aligned}$$

Observe that

$$\mathbf{e}_i^\top\Big(\widetilde{\mathbf{U}}\big(\widetilde{\mathbf{U}}^\top\mathbf{U}\big)-\mathbf{U}\Big)=\mathbf{e}_i^\top\big(\widetilde{\mathbf{U}}-\mathbf{U}\big)\big(\widetilde{\mathbf{U}}^\top\mathbf{U}\big)+\mathbf{e}_i^\top\mathbf{U}\big(\widetilde{\mathbf{U}}^\top\mathbf{U}-\mathbf{I}_{r_1}\big)$$

implying that on event $\mathcal{E}_2$,

$$\begin{aligned}&\Big\|\mathbf{e}_i^\top\Big(\widetilde{\mathbf{U}}\big(\widetilde{\mathbf{U}}^\top\mathbf{U}\big)-\mathbf{U}\Big)\Big\|_{\ell_2}\\&\quad\leq\big\|(\widetilde{\mathbf{U}}-\mathbf{U})^\top\mathbf{e}_i\big\|_{\ell_2}\|\widetilde{\mathbf{U}}^\top\mathbf{U}\|+\big\|\widetilde{\mathbf{U}}^\top\mathbf{U}-\mathbf{I}_{r_1}\big\|\|\mathbf{U}^\top\mathbf{e}_i\|_{\ell_2}\\&\quad\lesssim\frac{\sigma\overline{\Lambda}(\mathbf{A})r^{1/2}+\sigma^2dr^{1/2}}{\bar{g}_{\min}^2(\mathbf{A})}\log^{1/2}d+\|\mathbf{U}^\top\mathbf{e}_i\|_{\ell_2}\frac{\sigma\overline{\Lambda}(\mathbf{A})r+\sigma^2dr}{\bar{g}_{\min}^2(\mathbf{A})}\log^{1/2}d\\&\quad\lesssim\frac{\sigma\overline{\Lambda}(\mathbf{A})r+\sigma^2dr}{\bar{g}_{\min}^2(\mathbf{A})}\log^{1/2}d,\end{aligned}$$

where we used the facts $\|\widetilde{\mathbf{U}}^\top \mathbf{U}\| \le \|\widetilde{\mathbf{U}}\|\|\mathbf{U}\| \le (1+b_k)^{-1/2} = O(1)$ and

$$\|\mathbf{U}^\top \mathbf{e}_i\|_{\ell_2} = \big\langle \mathbf{U}\mathbf{U}^\top, \mathbf{e}_i \otimes \mathbf{e}_i \big\rangle^{1/2} \le 1.$$

Therefore, on event $\mathcal{E}_2$,

$$\begin{aligned}&\big|\big\langle \mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \\ &\qquad \lesssim \overline{\Lambda}(\mathbf{A})\bigg(\frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\log^{1/2} d\bigg)\|\mathbf{V}^\top\mathbf{e}_j\|_{\ell_2}\|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2}.\end{aligned}$$

Similar bounds hold for

$$\big|\big\langle \mathbf{A}\cdot\big(\mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\widetilde{\mathbf{V}}} - \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \quad\text{and}\quad \big|\big\langle \mathbf{A}\cdot\big(\mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\widetilde{\mathbf{W}}} - \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big|.$$

Following the same method, we can show that on event $\mathcal{E}_2$,

$$\begin{aligned}&\big|\big\langle \mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\widetilde{\mathbf{V}}} - \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \\ &\qquad \lesssim \overline{\Lambda}(\mathbf{A})\bigg(\frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\log^{1/2} d\bigg)^2\|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2}\end{aligned}$$

and

$$\begin{aligned}&\big|\big\langle \mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\widetilde{\mathbf{V}}} - \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\widetilde{\mathbf{W}}} - \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \\ &\qquad \lesssim \overline{\Lambda}(\mathbf{A})\bigg(\frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\log^{1/2} d\bigg)^3.\end{aligned}$$

We conclude that on event $\mathcal{E}_2$,

$$\begin{aligned}&\big|\big\langle \mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}}, \mathbf{P}_{\widetilde{\mathbf{V}}}, \mathbf{P}_{\widetilde{\mathbf{W}}}\big) - \mathbf{A}, \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \\ &\lesssim \overline{\Lambda}(\mathbf{A})\bigg(\frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\log^{1/2} d\bigg)\Big(\|\mathbf{V}^\top\mathbf{e}_j\|_{\ell_2}\|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2} \\ &\quad + \|\mathbf{U}^\top\mathbf{e}_i\|_{\ell_2}\|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2} + \|\mathbf{U}^\top\mathbf{e}_i\|_{\ell_2}\|\mathbf{V}^\top\mathbf{e}_j\|_{\ell_2}\Big) \\ &\quad + \overline{\Lambda}(\mathbf{A})\bigg(\frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\log^{1/2} d\bigg)^2\Big(\|\mathbf{V}^\top\mathbf{e}_j\|_{\ell_2} + \|\mathbf{U}^\top\mathbf{e}_i\|_{\ell_2} + \|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2}\Big) \\ &\quad + \overline{\Lambda}(\mathbf{A})\bigg(\frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\log^{1/2} d\bigg)^3.\end{aligned}$$

Recall that for all $i \in [d_1], j \in [d_2], k \in [d_3]$

$$\|\mathbf{U}^\top\mathbf{e}_i\|_{\ell_2} \le \mu_{\mathbf{U}}\sqrt{\frac{r}{d}}, \quad \|\mathbf{V}^\top\mathbf{e}_j\|_{\ell_2} \le \mu_{\mathbf{V}}\sqrt{\frac{r}{d}}, \quad \|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2} \le \mu_{\mathbf{W}}\sqrt{\frac{r}{d}}$$

and conditions (13) (14) (15) imply

$$\frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\log^{1/2} d \lesssim r\Big(\frac{\log d}{d}\Big)^{1/2}.$$

We end up with a simpler bound on event $\mathcal{E}_2$,

$$\begin{aligned}&\big|\big\langle \mathbf{A}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}}, \mathbf{P}_{\widetilde{\mathbf{V}}}, \mathbf{P}_{\widetilde{\mathbf{W}}}\big) - \mathbf{A}, \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \\ &\qquad \lesssim \sigma r^3\bigg(\frac{\sigma\widetilde{\kappa}(\mathbf{A})}{\bar{g}_{\min}(\mathbf{A})} + \frac{\widetilde{\kappa}^2(\mathbf{A})}{d}\bigg)(\mu_{\mathbf{U}}\mu_{\mathbf{V}} + \mu_{\mathbf{U}}\mu_{\mathbf{W}} + \mu_{\mathbf{V}}\mu_{\mathbf{W}})\log^{3/2} d\end{aligned} \tag{31}$$

where $\widetilde{\kappa}(\mathbf{A}) = \overline{\Lambda}(\mathbf{A})/\bar{g}_{\min}(\mathbf{A})$.

Next, we prove the upper bound of $\big|\big\langle \mathbf{Z}\cdot(\mathbf{P}_{\widetilde{\mathbf{U}}}, \mathbf{P}_{\widetilde{\mathbf{V}}}, \mathbf{P}_{\widetilde{\mathbf{W}}}), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big|$ and we proceed with the same decomposition. Observe that

$$\begin{aligned}\big\langle \mathbf{Z}\cdot(\mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\mathbf{W}}), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle =& \big\langle \mathbf{Z}, (\mathbf{P}_{\mathbf{U}}\mathbf{e}_i)\otimes(\mathbf{P}_{\mathbf{V}}\mathbf{e}_j)\otimes(\mathbf{P}_{\mathbf{W}}\mathbf{e}_k)\big\rangle \\ \sim& \mathcal{N}\Big(0, \sigma^2\|\mathbf{P}_{\mathbf{U}}\mathbf{e}_i\|^2_{\ell_2}\|\mathbf{P}_{\mathbf{V}}\mathbf{e}_j\|^2_{\ell_2}\|\mathbf{P}_{\mathbf{W}}\mathbf{e}_k\|^2_{\ell_2}\Big)\end{aligned}$$

The standard concentration inequality of Gaussian random variables yields that with probability at least $1 - \frac{1}{d^2}$,

$$\begin{aligned}\big|\big\langle \mathbf{Z}\cdot(\mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\mathbf{W}}), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \lesssim& \sigma\|\mathbf{U}^\top\mathbf{e}_i\|_{\ell_2}\|\mathbf{V}^\top\mathbf{e}_j\|_{\ell_2}\|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2}\log^{1/2} d \\ \lesssim& \sigma\Big(\frac{r}{d}\Big)^{3/2}\mu_{\mathbf{U}}\mu_{\mathbf{V}}\mu_{\mathbf{W}}\log^{1/2} d.\end{aligned}$$

Similarly, with probability at least $1 - \frac{1}{d^2}$,

$$\begin{aligned}&\big|\big\langle \mathbf{Z}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \\ &\quad = \big|\mathbf{e}_i^\top\big(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}}\big)\mathcal{M}_1(\mathbf{Z})\big(\mathbf{V}\otimes\mathbf{W}\big)\big((\mathbf{V}^\top\mathbf{e}_j)\otimes(\mathbf{W}^\top\mathbf{e}_k)\big)\big| \\ &\quad \le \|(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}})\mathbf{e}_i\|_{\ell_2}\big\|\mathcal{M}_1(\mathbf{Z})(\mathbf{V}\otimes\mathbf{W})\big\|\|\mathbf{V}^\top\mathbf{e}_j\|_{\ell_2}\|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2} \\ &\quad \lesssim \sigma d^{1/2}\|(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}})\mathbf{e}_i\|_{\ell_2}\|\mathbf{V}^\top\mathbf{e}_j\|_{\ell_2}\|\mathbf{W}^\top\mathbf{e}_k\|_{\ell_2}\end{aligned}$$

where we used Lemma 14 for the upper bound of $\big\|\mathcal{M}_1(\mathbf{Z})(\mathbf{V}\otimes\mathbf{W})\big\|$. Moreover, since $\mu_{\mathbf{U}} \ge 1$,

$$\begin{aligned}\big\|\big(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}}\big)\mathbf{e}_i\big\|_{\ell_2} \le& \|(\widetilde{\mathbf{U}} - \mathbf{U})\mathbf{e}_i\|_{\ell_2} + \|\widetilde{\mathbf{U}} - \mathbf{U}\|_{\ell_2}\|\mathbf{U}^\top\mathbf{e}_i\|_{\ell_2} \\ \lesssim& \frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\mu_{\mathbf{U}}\log^{1/2} d.\end{aligned}$$

Denote the above event by $\mathcal{E}_3$. On $\mathcal{E}_2 \cap \mathcal{E}_3$,

$$\big|\big\langle \mathbf{Z}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \lesssim \frac{\sigma r}{d^{1/2}}\bigg(\frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\bigg)\mu_{\mathbf{U}}\mu_{\mathbf{V}}\mu_{\mathbf{W}}\log^{1/2} d.$$

Similar bounds can be attained for

$$\big|\big\langle \mathbf{Z}\cdot\big(\mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\widetilde{\mathbf{V}}} - \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \quad\text{and}\quad \big|\big\langle \mathbf{Z}\cdot\big(\mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\widetilde{\mathbf{W}}} - \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big|.$$

In an identical fashion, on event $\mathcal{E}_2 \cap \mathcal{E}_3$,

$$\begin{aligned}&\big|\big\langle \mathbf{Z}\cdot\big(\mathbf{P}_{\widetilde{\mathbf{U}}} - \mathbf{P}_{\mathbf{U}}, \mathbf{P}_{\widetilde{\mathbf{V}}} - \mathbf{P}_{\mathbf{V}}, \mathbf{P}_{\mathbf{W}}\big), \mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\big\rangle\big| \\ &\qquad \lesssim \sigma r^{1/2}\bigg(\frac{\sigma\overline{\Lambda}(\mathbf{A})r + \sigma^2 dr}{\bar{g}^2_{\min}(\mathbf{A})}\bigg)^2\mu_{\mathbf{U}}\mu_{\mathbf{V}}\mu_{\mathbf{W}}\log d.\end{aligned}$$

and

$$\begin{aligned}&\left|\left\langle \mathbf{Z}\cdot\left(\mathbf{P}_{\widetilde{\mathbf{U}}}-\mathbf{P}_{\mathbf{U}},\mathbf{P}_{\widetilde{\mathbf{V}}}-\mathbf{P}_{\mathbf{V}},\mathbf{P}_{\widetilde{\mathbf{W}}}-\mathbf{P}_{\mathbf{W}}\right),\mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\right\rangle\right|\\&\lesssim\sigma d^{1/2}\left(\frac{\sigma\overline{\Lambda}(\mathbf{A})r+\sigma^2dr}{\bar{g}^2_{\min}(\mathbf{A})}\right)^3\mu_{\mathbf{U}}\mu_{\mathbf{V}}\mu_{\mathbf{W}}\log^{3/2}d.\end{aligned}$$

Observe by conditions (13) (14) (15) that

$$\frac{\sigma\overline{\Lambda}(\mathbf{A})r+\sigma^2dr}{\bar{g}^2_{\min}(\mathbf{A})}\lesssim\frac{r}{d^{1/2}}.$$

We conclude on event $\mathcal{E}_2\cap\mathcal{E}_3$ with

$$\left|\left\langle \mathbf{Z}\cdot\left(\mathbf{P}_{\widetilde{\mathbf{U}}},\mathbf{P}_{\widetilde{\mathbf{V}}},\mathbf{P}_{\widetilde{\mathbf{W}}}\right),\mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\right\rangle\right|\lesssim\frac{\sigma r^2}{d^{1/2}}\left(\frac{\sigma\overline{\Lambda}(\mathbf{A})r+\sigma^2dr}{\bar{g}^2_{\min}(\mathbf{A})}\right)\mu_{\mathbf{U}}\mu_{\mathbf{V}}\mu_{\mathbf{W}}\log^{3/2}d.\tag{32}$$

By combining (31) and (32), we get on event $\mathcal{E}_2\cap\mathcal{E}_3$,

$$\begin{aligned}&\left|\left\langle\widetilde{\mathbf{A}}-\mathbf{A},\mathbf{e}_i\otimes\mathbf{e}_j\otimes\mathbf{e}_k\right\rangle\right|\\&\lesssim\sigma r^3\left(\frac{\sigma\widetilde{\kappa}(\mathbf{A})}{\bar{g}_{\min}(\mathbf{A})}+\frac{\widetilde{\kappa}^2(\mathbf{A})}{d}\right)(\mu_{\mathbf{U}}\mu_{\mathbf{V}}+\mu_{\mathbf{U}}\mu_{\mathbf{W}}+\mu_{\mathbf{V}}\mu_{\mathbf{W}})\log^{3/2}d\\&+\frac{\sigma r^2}{d^{1/2}}\left(\frac{\sigma\overline{\Lambda}(\mathbf{A})r+\sigma^2dr}{\bar{g}^2_{\min}(\mathbf{A})}\right)\mu_{\mathbf{U}}\mu_{\mathbf{V}}\mu_{\mathbf{W}}\log^{3/2}d\\&\lesssim\sigma r^3\left(\frac{\sigma\widetilde{\kappa}(\mathbf{A})}{\bar{g}_{\min}(\mathbf{A})}+\frac{\widetilde{\kappa}^2(\mathbf{A})}{d}\right)(\mu_{\mathbf{U}}\mu_{\mathbf{V}}+\mu_{\mathbf{U}}\mu_{\mathbf{W}}+\mu_{\mathbf{V}}\mu_{\mathbf{W}})\log^{3/2}d,\end{aligned}$$

where the last inequality is due to fact $\bar{g}_{\min}(\mathbf{A})\gtrsim\sigma d^{3/4}$ and $\max\left\{\mu_{\mathbf{U}},\mu_{\mathbf{V}},\mu_{\mathbf{W}}\right\}\lesssim\sqrt{d}$.

References

Evrim Acar and Bülent Yener. Unsupervised multiway data analysis: A literature survey. *IEEE transactions on knowledge and data engineering*, 21(1):6–20, 2009.

Animashree Anandkumar, Rong Ge, Daniel Hsu, and Sham Kakade. A tensor spectral approach to learning mixed membership community models. In *Conference on Learning Theory*, pages 867–881, 2013.

Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research*, 15(1):2773–2832, 2014.

Amir Ben-Dor, Benny Chor, Richard Karp, and Zohar Yakhini. Discovering local structure in gene expression data: the order-preserving submatrix problem. *Journal of computational biology*, 10(3-4):373–384, 2003.

Göran Bergqvist and Erik G Larsson. The higher-order singular value decomposition: Theory and an application [lecture notes]. *IEEE Signal Processing Magazine*, 27(3):151–154, 2010.

S Charles Brubaker and Santosh S Vempala. Random tensors and planted cliques. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 406–419. Springer, 2009.

T Tony Cai and Anru Zhang. Rate-optimal perturbation bounds for singular subspaces with applications to high-dimensional statistics. *arXiv preprint arXiv:1605.00353*, 2016.

T Tony Cai, Tengyuan Liang, and Alexander Rakhlin. Computational and statistical boundaries for submatrix localization in a large noisy matrix. *arXiv preprint arXiv:1502.01988*, 2015.

Joshua Cape, Minh Tang, and Carey E Priebe. The two-to-infinity norm and singular subspace geometry with applications to high-dimensional statistics. *arXiv preprint arXiv:1705.10735*, 2017.

Arun T Chaganty and Percy Liang. Spectral experts for estimating mixtures of linear regressions. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1040–1048, 2013.

Jie Chen and Yousef Saad. On the tensor svd and the optimal low rank orthogonal approximation of tensors. *SIAM Journal on Matrix Analysis and Applications*, 30(4):1709–1734, 2009.

Andrzej Cichocki, Danilo Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar Caiafa, and Huy Anh Phan. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Processing Magazine*, 32 (2):145–163, 2015.

Chandler Davis and William Morton Kahan. The rotation of eigenvectors by a perturbation. iii. *SIAM Journal on Numerical Analysis*, 7(1):1–46, 1970.

Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. On the best rank-1 and rank-(r1, r2,..., rn) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000a.

Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000b.

Jianqing Fan and Yingying Fan. High dimensional classification using features annealed independence rules. *Annals of statistics*, 36(6):2605, 2008.

Jianqing Fan, Weichen Wang, and Yiqiao Zhong. An ℓ_∞ eigenvector perturbation bound and its application to robust covariance estimation. *arXiv preprint arXiv:1603.03516*, 2016.

Laura Florescu and Will Perkins. Spectral thresholds in the bipartite stochastic block model. *arXiv preprint arXiv:1506.06737*, 2015.

Jerome H Friedman. Regularized discriminant analysis. *Journal of the American statistical association*, 84(405):165–175, 1989.

Laetitia Gauvin, André Panisson, and Ciro Cattuto. Detecting the community structure and activity patterns of temporal networks: a non-negative tensor factorization approach. *PloS one*, 9(1):e86028, 2014.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Unsupervised learning. In *The elements of statistical learning*, pages 485–585. Springer, 2009.

T Hildebrand and P Rüegsegger. A new method for the model-independent assessment of thickness in three-dimensional images. *Journal of microscopy*, 185(1):67–75, 1997.

Christopher J Hillar and Lek-Heng Lim. Most tensor problems are NP-hard. *Journal of the ACM (JACM)*, 60(6):45, 2013.

Samuel B Hopkins, Jonathan Shi, and David Steurer. Tensor principal component analysis via sum-of-square proofs. In *COLT*, pages 956–1006, 2015.

Victoria Hore, Ana Viñuela, Alfonso Buil, Julian Knight, Mark I McCarthy, Kerrin Small, and Jonathan Marchini. Tensor decomposition for multiple-tissue gene expression experiments. *Nature Genetics*, 48(9):1094–1100, 2016.

Jiashun Jin. Fast community detection by score. *The Annals of Statistics*, 43(1):57–89, 2015.

Mladen Kolar, Sivaraman Balakrishnan, Alessandro Rinaldo, and Aarti Singh. Minimax localization of structural information in large noisy matrices. In *Advances in Neural Information Processing Systems*, pages 909–917, 2011.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

Vladimir Koltchinskii and Karim Lounici. Asymptotics and concentration bounds for bilinear forms of spectral projectors of sample covariance. In *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, volume 52, pages 1976–2013. Institut Henri Poincaré, 2016.

Vladimir Koltchinskii and Karim Lounici. Concentration inequalities and moment bounds for sample covariance operators. *Bernoulli*, 23(1):110–133, 2017.

Vladimir Koltchinskii and Dong Xia. Perturbation of linear forms of singular vectors under gaussian noise. In *High Dimensional Probability VII*, pages 397–423. Springer, 2016.

Nan Li and Baoxin Li. Tensor completion for on-board compression of hyperspectral images. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 517–520. IEEE, 2010.

Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2013.

Tianqi Liu, Ming Yuan, and Hongyu Zhao. Characterizing spatiotemporal transcriptome of human brain via low rank tensor decomposition. *arXiv preprint arXiv:1702.07449*, 2017.

Zongming Ma and Yihong Wu. Computational barriers in minimax submatrix detection. *The Annals of Statistics*, 43(3):1089–1116, 2015.

Andrew McCallum, Kamal Nigam, and Lyle H Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 169–178. ACM, 2000.

Pradipta Mitra. Entrywise bounds for eigenvectors of random graphs. *Electronic journal of combinatorics*, 16(1):R131, 2009.

Chaitanya Muralidhara, Andrew M Gross, Robin R Gutell, and Orly Alter. Tensor decomposition reveals concurrent evolutionary convergences and divergences and correlations with structural motifs in ribosomal rna. *PloS one*, 6(4):e18768, 2011.

Mark EJ Newman. Detecting community structure in networks. *The European Physical Journal B-Condensed Matter and Complex Systems*, 38(2):321–330, 2004.

Larsson Omberg, Gene H Golub, and Orly Alter. A tensor higher-order singular value decomposition for integrative analysis of DNA microarray data from different studies. *Proceedings of the National Academy of Sciences*, 104(47):18371–18376, 2007.

Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace clustering for high dimensional data: a review. *Acm Sigkdd Explorations Newsletter*, 6(1):90–105, 2004.

Sri Priya Ponnapalli, Michael A Saunders, Charles F Van Loan, and Orly Alter. A higher-order generalized singular value decomposition for comparison of global mrna expression from multiple organisms. *PloS one*, 6(12):e28072, 2011.

Emile Richard and Andrea Montanari. A statistical model for tensor PCA. In *Advances in Neural Information Processing Systems*, pages 2897–2905, 2014.

Mark Rudelson and Roman Vershynin. Delocalization of eigenvectors of random matrices with independent entries. *Duke Mathematical Journal*, 164(13):2507–2538, 2015.

M Vasilescu and Demetri Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. *Computer Vision?ECCV 2002*, pages 447–460, 2002.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Van Vu and Ke Wang. Random weighted projections, random quadratic forms and random eigenvectors. *Random Structures & Algorithms*, 47(4):792–821, 2015.

Rongrong Wang. Singular vector perturbation under gaussian noise. *SIAM Journal on Matrix Analysis and Applications*, 36(1):158–177, 2015.

Perke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

C-F Westin, Stephan E Maier, Hatsuho Mamata, Arya Nabavi, Ferenc A Jolesz, and Ron Kikinis. Processing and visualization for diffusion tensor mri. *Medical image analysis*, 6 (2):93–108, 2002.

Dong Xia and Ming Yuan. On polynomial time methods for exact low rank tensor completion. *Foundations of Computational Mathematics, to appear*, 2019.

Qing Xiong, Nicola Ancona, Elizabeth R Hauser, Sayan Mukherjee, and Terrence S Furey. Integrating genetic and gene expression evidence into genome-wide association analysis of gene sets. *Genome research*, 22(2):386–397, 2012.

Anru Zhang and Dong Xia. Tensor svd: Statistical and computational limits. *IEEE Transactions on Information Theory*, 64(11):7311–7338, 2018.

Qinqing Zheng and Ryota Tomioka. Interpolating convex and non-convex tensor decompositions via the subspace norm. In *Advances in Neural Information Processing Systems*, pages 3106–3113, 2015.

Appendix A. Proof of Lemma 14

Let $\mathbf{z}_i \in \mathbb{R}^{m_1}, i = 1, \ldots, m_2$ denote the columns of $\mathbf{Z}$. Then, we write

$$\mathbf{Z}\mathbf{Z}^\top - \sigma^2 m_2 \mathbf{I}_{m_1} = \sum_{i=1}^{m_2} \big(\mathbf{z}_i \otimes \mathbf{z}_i - \sigma^2 \mathbf{I}_{m_1}\big).$$

Similarly, let $\tilde{\mathbf{z}}_j^\top \in \mathbb{R}^{m_2}, j = 1, \ldots, m_1$ denote the rows of $\mathbf{Z}$ and observe that $\|\mathbf{B}\mathbf{Z}^\top\| = \|\mathbf{B}\mathbf{Z}^\top\mathbf{Z}\mathbf{B}^\top\|^{1/2}$ and

$$\mathbf{B}\mathbf{Z}^\top\mathbf{Z}\mathbf{B}^\top = \sum_{j=1}^{m_1} \Big(\big(\mathbf{B}\tilde{\mathbf{z}}_j\big) \otimes \big(\mathbf{B}\tilde{\mathbf{z}}_j\big) - \sigma^2 \mathbf{B}\mathbf{B}^\top\Big).$$

The inequalities (28) and (23) are on the concentration of sample covariance operator, where a sharp bound has been derived in Koltchinskii and Lounici (2017) and will be skipped here.

Appendix B. Proof of Theorem 15

Since $\mathbb{E}\hat{\mathbf{\Gamma}} = \mathbf{0}$, we immediately get $\mathbb{E}\mathbf{L}_k(\hat{\mathbf{\Gamma}}) = \mathbf{0}$. Then,

$$\big\langle \mathbf{x}, \hat{\mathbf{P}}_k^{uu}\mathbf{y}\big\rangle - \mathbb{E}\big\langle \mathbf{x}, \hat{\mathbf{P}}_k^{uu}\mathbf{y}\big\rangle = \big\langle \mathbf{x}, \mathbf{L}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\big\rangle + \big\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\big\rangle - \mathbb{E}\big\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\big\rangle.$$

Lemma 19 *For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{m_1}$, there exists an absolute constant $D_1 > 0$ such that for all $0 \le t \le m_1$, with probability at least $1 - e^{-t}$,*

$$\big|\big\langle \mathbf{x}, \mathbf{L}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\big\rangle\big| \le D_1 t^{1/2}\bigg(\frac{\sigma\mu_1 + \sigma^2 m_2^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\bigg)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.$$

Proof Recall that

$$\hat{\mathbf{\Gamma}} = \mathbf{A}\mathbf{Z}^\top + \mathbf{Z}\mathbf{A}^\top + \mathbf{Z}\mathbf{Z}^\top - m_2\sigma^2\mathbf{I}_{m_1}.$$

Then, we write $\big\langle \mathbf{x}, \mathbf{L}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\big\rangle$ as

$$\begin{aligned}\big\langle \mathbf{x}, \mathbf{L}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\big\rangle =& \big\langle \hat{\mathbf{\Gamma}}\mathbf{P}_k^{uu}\mathbf{x}, \mathbf{C}_k^{uu}\mathbf{y}\big\rangle + \big\langle \hat{\mathbf{\Gamma}}\mathbf{C}_k^{uu}\mathbf{x}, \mathbf{P}_k^{uu}\mathbf{y}\big\rangle \\ =& \big\langle (\mathbf{A}\mathbf{Z}^\top + \mathbf{Z}\mathbf{A}^\top + \mathbf{Z}\mathbf{Z}^\top - m_2\sigma^2\mathbf{I}_{m_1})\mathbf{P}_k^{uu}\mathbf{x}, \mathbf{C}_k^{uu}\mathbf{y}\big\rangle \\ &+ \big\langle (\mathbf{A}\mathbf{Z}^\top + \mathbf{Z}\mathbf{A}^\top + \mathbf{Z}\mathbf{Z}^\top - m_2\sigma^2\mathbf{I}_{m_1})\mathbf{C}_k^{uu}\mathbf{x}, \mathbf{P}_k^{uu}\mathbf{y}\big\rangle.\end{aligned}$$

It suffices to consider the following terms separately for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{m_1}$:

$$\big\langle \mathbf{Z}\mathbf{A}^\top\mathbf{x}, \mathbf{y}\big\rangle, \quad \big\langle \mathbf{A}\mathbf{Z}^\top\mathbf{x}, \mathbf{y}\big\rangle, \quad \big\langle \big(\mathbf{Z}\mathbf{Z}^\top - m_2\sigma^2\mathbf{I}_{m_1}\big)\mathbf{x}, \mathbf{y}\big\rangle.$$

It is straightforward to check that $\langle \mathbf{Z}\mathbf{A}^\top\mathbf{x}, \mathbf{y}\rangle$ is a normal random variable with zero mean and variance

$$\mathbb{E}\langle \mathbf{Z}\mathbf{A}^\top\mathbf{x}, \mathbf{y}\rangle^2 = \mathbb{E}\langle \mathbf{Z}, \mathbf{y} \otimes (\mathbf{A}^\top\mathbf{x})\rangle^2 = \sigma^2\|\mathbf{y} \otimes (\mathbf{A}^\top\mathbf{x})\|_{\ell_2}^2 = \sigma^2\|\mathbf{y}\|_{\ell_2}^2\|\mathbf{A}^\top\mathbf{x}\|_{\ell_2}^2,$$

where we used the fact that $\mathbf{Z}$ is a $m_1 \times m_2$ matrix with i.i.d. $\mathcal{N}(0, \sigma^2)$ entries. Therefore,

$$\mathbb{E}\langle \mathbf{Z}\mathbf{A}^\top\mathbf{P}_k^{uu}\mathbf{x}, \mathbf{C}_k^{uu}\mathbf{y}\rangle^2 \le \frac{\sigma^2\mu_k^2}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}^2\|\mathbf{y}\|_{\ell_2}^2,$$

where we used the facts $\|\mathbf{C}_k\| \le \frac{1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}$ and $\|\mathbf{A}^\top\mathbf{P}_k^{uu}\| \le \mu_k$. By the standard concentration inequality of Gaussian random variables, we get for all $t \ge 0$,

$$\mathbb{P}\bigg(\big|\langle \mathbf{Z}\mathbf{A}^\top\mathbf{P}_k^{uu}\mathbf{x}, \mathbf{C}_k^{uu}\mathbf{y}\rangle\big| \ge 2t^{1/2}\frac{\sigma\mu_k}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\bigg) \le e^{-t}.$$

Similarly, for all $t \ge 0$,

$$\mathbb{P}\bigg(\big|\langle \mathbf{Z}\mathbf{A}^\top\mathbf{C}_k^{uu}\mathbf{x}, \mathbf{P}_k^{uu}\mathbf{y}\rangle\big| \ge 2t^{1/2}\frac{\sigma\mu_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\bigg) \le e^{-t}.$$

We next turn to the bound of $\big|\big\langle \big(\mathbf{Z}\mathbf{Z}^\top - m_2\sigma^2\mathbf{I}_{m_1}\big)\mathbf{P}_k^{uu}\mathbf{x}, \mathbf{C}_k^{uu}\mathbf{y}\big\rangle\big|$. Recall that $\mathbf{P}_k^{uu}\mathbf{C}_k^{uu} = \mathbf{0}$ implying that it suffices to consider $\big\langle \mathbf{Z}\mathbf{Z}^\top\mathbf{P}_k^{uu}\mathbf{x}, \mathbf{C}_k^{uu}\mathbf{y}\big\rangle$. Let $\mathbf{z}_1, \ldots, \mathbf{z}_{m_2} \in \mathbb{R}^{m_1}$ denote the columns of $\mathbf{Z}$ such that $\mathbf{z}_i \in \mathcal{N}\big(\mathbf{0}, \sigma^2\mathbf{I}_{m_1}\big)$ for $1 \le i \le m_2$. Write

$$\big\langle \mathbf{Z}\mathbf{Z}^\top(\mathbf{P}_k^{uu}\mathbf{x}), \mathbf{C}_k^{uu}\mathbf{y}\big\rangle = \sum_{i=1}^{m_2}\big\langle \mathbf{z}_i, \mathbf{P}_k^{uu}\mathbf{x}\big\rangle\big\langle \mathbf{z}_i, \mathbf{C}_k^{uu}\mathbf{y}\big\rangle.$$

Observe that $\mathbb{E}\big(\mathbf{P}_k^{uu}\mathbf{z}_i\big) \otimes \big(\mathbf{C}_k^{uu}\mathbf{z}_i\big) = \mathbf{0}$ implying that $\big\langle \mathbf{z}_i, \mathbf{P}_k^{uu}\mathbf{x}\big\rangle$ is independent of $\big\langle \mathbf{z}_i, \mathbf{C}_k^{uu}\mathbf{y}\big\rangle$. By concentration inequalities of Gaussian random variables, for all $t \ge 0$,

$$\begin{aligned}\mathbb{P}\bigg(\big|\big\langle \mathbf{Z}\mathbf{Z}^\top(\mathbf{P}_k^{uu}\mathbf{x}), \mathbf{C}_k^{uu}\mathbf{y}\big\rangle\big| \ge 2t^{1/2}\|\mathbf{y}\|_{\ell_2}&\frac{\sigma\big(\sum_{i=1}^{m_2}\langle \mathbf{z}_i, \mathbf{P}_k^{uu}\mathbf{x}\rangle^2\big)^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)} \\ &\Big|\Big\{\langle \mathbf{z}_i, \mathbf{P}_k^{uu}\mathbf{x}\rangle : i = 1, \ldots, m_2\Big\}\bigg) \le e^{-t}.\end{aligned}$$

By (Vershynin, 2010, Prop 5.16), the following bound holds with probability at least $1-e^{-t}$,

$$\Big|\sum_{i=1}^{m_2}\langle \mathbf{z}_i, \mathbf{P}_k^{uu}\mathbf{x}\rangle^2 - \sigma^2 m_2\|\mathbf{x}\|_{\ell_2}^2\Big| \lesssim \sigma\Big(m_2^{1/2}t^{1/2}+t\Big)\|\mathbf{x}\|_{\ell_2}.$$

If $t \lesssim m_1 \le m_2$, we conclude that there exists an absolute constant $D_1>0$ such that

$$\mathbb{P}\bigg(\Big|\big\langle \mathbf{Z}\mathbf{Z}^\top(\mathbf{P}_k^{uu}\mathbf{x}), \mathbf{C}_k^{uu}\mathbf{y}\big\rangle\Big| \ge D_1\frac{\sigma^2 m_2^{1/2}t^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\bigg) \le e^{-t}.$$

To sum up, for all $0\le t\lesssim m_1$, the following bound holds with probability at least $1-e^{-t}$,

$$\big|\big\langle \mathbf{x}, \mathbf{L}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\big\rangle\big| \lesssim t^{1/2}\bigg(\frac{\sigma\mu_1+\sigma^2 m_2^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\bigg)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}$$

which concludes the proof. ■

It remains to derive the upper bound of $\big|\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle - \mathbb{E}\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle\big|$.The following lemma is due to Koltchinskii and Lounici (2016).

Lemma 20 *Let $\delta(m_1,m_2) := \sigma\mu_1 m_1^{1/2} + \sigma^2(m_1m_2)^{1/2}$ and suppose that $\delta(m_1,m_2) \le \frac{1-\gamma}{2(1+\gamma)}\bar{g}_k(\mathbf{A}\mathbf{A}^\top)$ for some $\gamma\in(0,1)$. There exists a constant $D_\gamma>0$ such that, for all symmetric $\widehat{\mathbf{\Gamma}}_1, \hat{\mathbf{\Gamma}}_2 \in \mathbb{R}^{m_1\times m_1}$ satisfying the condition $\max\big\{\|\hat{\mathbf{\Gamma}}_1\|, \|\hat{\mathbf{\Gamma}}_2\|\big\} \le (1+\gamma)\delta(m_1,m_2)$,*

$$\|\mathbf{S}_k(\hat{\mathbf{\Gamma}}_1) - \mathbf{S}_k(\hat{\mathbf{\Gamma}}_2)\| \le D_\gamma\frac{\delta(m_1,m_2)}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\|\hat{\mathbf{\Gamma}}_1-\hat{\mathbf{\Gamma}}_2\|.$$

Define function $\varphi(\cdot):\mathbb{R}_+\mapsto[0,1]$ such that $\varphi(t)=1$ for $0\le t\le 1$ and $\varphi(t)=0$ for $t\ge(1+\gamma)$ and φ is linear in between. Then, function φ is Lipschitz on $\mathbb{R}_+$ with constant $\frac{1}{\gamma}$. To illustrate the dependence of $\hat{\mathbf{\Gamma}}$ on $\mathbf{Z}$, we write $\hat{\mathbf{\Gamma}}(\mathbf{Z})$ instead of $\hat{\mathbf{\Gamma}}$. To this end, fix $\mathbf{x},\mathbf{y}\in\mathbb{R}^{m_1}$ and constants $\delta_1,\delta_2>0$ and define the function

$$F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}) := \Big\langle \mathbf{x}, \mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z})\big)\mathbf{y}\Big\rangle\varphi\Big(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z})\|}{\delta_1}\Big)\varphi\Big(\frac{\|\mathbf{Z}\|}{\delta_2}\Big).$$

where we view $\mathbf{Z}$ as a point in $\mathbb{R}^{m_1\times m_2}$ rather than a random matrix.

Lemma 21 *For any $\delta_1 \le \frac{1-\gamma}{2(1+\gamma)}\bar{g}_k(\mathbf{A}\mathbf{A}^\top)$ for some $\gamma\in(0,1)$ and $\delta_2>0$, there exists an absolute constant $C_\gamma>0$ such that*

$$\big|F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_1) - F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_2)\big| \le C_\gamma\frac{\delta_1}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\Big(\mu_1+\delta_2+\frac{\delta_1}{\delta_2}\Big)\|\mathbf{Z}_1-\mathbf{Z}_2\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}$$

Proof [Proof of Lemma 21] Since $\varphi(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z})\|}{\delta_1})\varphi(\frac{\|\mathbf{Z}\|}{\delta_2}) \ne 0$ only if $\|\hat{\mathbf{\Gamma}}(\mathbf{Z})\| \le (1+\gamma)\delta_1$ and $\|\mathbf{Z}\|\le(1+\gamma)\delta_2$, Lemma 16 implies that

$$\big|F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z})\big| = \Big|\Big\langle \mathbf{x}, \mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z})\big)\mathbf{y}\Big\rangle\varphi\Big(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z})\|}{\delta_1}\Big)\varphi\Big(\frac{\|\mathbf{Z}\|}{\delta_2}\Big)\Big| \le 14(1+\gamma)^2\frac{\delta_1^2}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}.$$

Case 1. If $\max\big\{\big\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big\|, \big\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\big\|\big\} \le (1+\gamma)\delta_1$ and $\max\big\{\|\mathbf{Z}_1\|,\|\mathbf{Z}_2\|\big\}\le(1+\gamma)\delta_2$.

By the Lipschitzity of function φ, Lemma 20 and definition of $\hat{\mathbf{\Gamma}}(\mathbf{Z})$, it is easy to check

$$\begin{aligned}
&|F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_1) - F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_2)|\\
&\quad\le\|\mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big) - \mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\big)\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\\
&\quad+\frac{14(1+\gamma)^2\delta_1}{\gamma\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\big\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)-\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\big\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2} + \frac{14(1+\gamma)^2\delta_1^2}{\delta_2\gamma\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{Z}_1-\mathbf{Z}_2\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\\
&\quad\le D_\gamma\frac{\delta_1}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)-\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2} + \frac{14(1+\gamma)^2\delta_1^2}{\delta_2\gamma\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{Z}_1-\mathbf{Z}_2\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\\
&\quad\le D_\gamma\frac{\delta_1}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\Big(\mu_1+\delta_2+\frac{\delta_1}{\delta_2}\Big)\|\mathbf{Z}_1-\mathbf{Z}_2\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.
\end{aligned}$$

Case 2. If $\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\| \le (1+\gamma)\delta_1$, $\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\| \ge (1+\gamma)\delta_1$ and $\max\big\{\|\mathbf{Z}_1\|,\|\mathbf{Z}_2\|\big\} \le (1+\gamma)\delta_2$. Since $\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\| \ge (1+\gamma)\delta_1$, we have $\varphi\big(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\|}{\delta_1}\big)=0$ and $F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_2)=0$. Then,

$$\begin{aligned}
&\big|F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_1) - F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_2)\big|\\
&\quad=\Big|\Big\langle \mathbf{x}, \mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big)\mathbf{y}\Big\rangle\varphi\Big(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\|}{\delta_1}\Big)\varphi\Big(\frac{\|\mathbf{Z}_1\|}{\delta_2}\Big)\Big|\\
&\quad=\Big|\Big\langle \mathbf{x}, \mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big)\mathbf{y}\Big\rangle\varphi\Big(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\|}{\delta_1}\Big)\varphi\Big(\frac{\|\mathbf{Z}_1\|}{\delta_2}\Big) - \Big\langle \mathbf{x}, \mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big)\mathbf{y}\Big\rangle\varphi\Big(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\|}{\delta_1}\Big)\varphi\Big(\frac{\|\mathbf{Z}_1\|}{\delta_2}\Big)\Big|\\
&\quad\le\big\|\mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big)\big\|\frac{1}{\delta_1\gamma}\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)-\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\\
&\quad\le\frac{(1+\gamma)^2\delta_1^2}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)\delta_1\gamma}\big(2\mu_1+2(1+\gamma)\delta_2\big)\|\mathbf{Z}_1-\mathbf{Z}_2\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\\
&\quad\le D_\gamma\frac{\delta_1}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}(\mu_1+\delta_2)\|\mathbf{Z}_1-\mathbf{Z}_2\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.
\end{aligned}$$

Case 3. If $\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\| \le (1+\gamma)\delta_1$, $\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\| \ge (1+\gamma)\delta_1$, $\|\mathbf{Z}_1\| \le (1+\gamma)\delta_2$, $\|\mathbf{Z}_2\| \ge (1+\gamma)\delta_2$. It can be proved similarly as *Case 2.*

Case 4. If $\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\| \le (1+\gamma)\delta_1$, $\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\| \ge (1+\gamma)\delta_1$, $\|\mathbf{Z}_1\| \ge (1+\gamma)\delta_2$, $\|\mathbf{Z}_2\| \ge (1+\gamma)\delta_2$. It is a trivial case since $F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_1) = F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_2) = 0$.

Case 5. If $\max\big\{\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\|, \|\hat{\mathbf{\Gamma}}(\mathbf{Z}_2)\|\big\} \le (1+\gamma)\delta_1$, $\|\mathbf{Z}_1\| \le (1+\gamma)\delta_2$, $\|\mathbf{Z}_2\| \ge (1+\gamma)\delta_2$. Again, we have $F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_2) = 0$. Then,

$$\begin{aligned}
&\big|F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_1) - F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}_2)\big|\\
&\quad=\Big|\Big\langle \mathbf{x}, \mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big)\mathbf{y}\Big\rangle\varphi\Big(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\|}{\delta_1}\Big)\varphi\Big(\frac{\|\mathbf{Z}_1\|}{\delta_2}\Big)\Big|\\
&\quad=\Big|\Big\langle \mathbf{x}, \mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big)\mathbf{y}\Big\rangle\varphi\Big(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\|}{\delta_1}\Big)\varphi\Big(\frac{\|\mathbf{Z}_1\|}{\delta_2}\Big) - \Big\langle \mathbf{x}, \mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big)\mathbf{y}\Big\rangle\varphi\Big(\frac{\|\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\|}{\delta_1}\Big)\varphi\Big(\frac{\|\mathbf{Z}_2\|}{\delta_2}\Big)\Big|\\
&\quad\le\big\|\mathbf{S}_k\big(\hat{\mathbf{\Gamma}}(\mathbf{Z}_1)\big)\big\|\frac{1}{\delta_2\gamma}\|\mathbf{Z}_1-\mathbf{Z}_2\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2} \le \frac{(1+\gamma)^2\delta_1^2}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)\delta_2\gamma}\|\mathbf{Z}_1-\mathbf{Z}_2\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\\
&\quad\le D_\gamma\frac{\delta_1}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\frac{\delta_1}{\delta_2}\|\mathbf{Z}_1-\mathbf{Z}_2\|\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.
\end{aligned}$$

All the other cases shall be handled similarly and we conclude the proof. ■

Note that $\|\mathbf{Z}_1 - \mathbf{Z}_2\| \le \|\mathbf{Z}_1 - \mathbf{Z}_2\|_{\ell_2}$, Lemma 21 indicates that $F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z})$ is Lipschitz with constant

$$D_\gamma \frac{\delta_1}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\Big(\mu_1 + \delta_2 + \frac{\delta_1}{\delta_2}\Big)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.$$

Lemma 22 *Let $\delta(m_1,m_2) := \sigma\mu_1 m_1^{1/2} + \sigma^2(m_1m_2)^{1/2}$ and suppose that $\mathbb{E}\|\hat{\mathbf{\Gamma}}\| \le \frac{1-\gamma}{2}\bar{g}_k(\mathbf{A}\mathbf{A}^\top)$ for some $\gamma \in (0,1)$. There exists some constant D_γ such that for any $\mathbf{x},\mathbf{y} \in \mathbb{R}^{m_1}$ and all $\log 8 \le t \le m_1$, the following inequality holds with probability at least $1 - e^{-t}$,*

$$\big|\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle - \mathbb{E}\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle\big| \le D_\gamma t^{1/2}\frac{\sigma\mu_1 + \sigma^2 m_2^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.$$

Proof [Proof of Lemma 22] Choose $\delta_1 = \delta_1(m_1,m_2)$ and $\delta_2 = \delta_2(m_1,m_2)$ as follows where $\log 8 \le t \le m_1$ is to be determined:

$$\begin{aligned}\delta_1(m_1,m_2) :&= \delta_1(m_1,m_2,t) := \mathbb{E}\|\widetilde{\mathbf{\Gamma}}\| + D_1 t^{1/2}(\sigma\mu_1 + \sigma^2 m_2^{1/2})\\ \delta_2(m_1,m_2) :&= \delta_2(m_1,m_2,t) := \mathbb{E}\|\mathbf{Z}\| + D_2\sigma t^{1/2}\end{aligned}$$

and the constants $D_1, D_2 > 0$ are chosen such that $\mathbb{P}\big(\|\hat{\mathbf{\Gamma}}\| \ge \delta_1(m_1,m_2,t)\big) \le e^{-t}$ and $\mathbb{P}\big(\|\mathbf{Z}\| \ge \delta_2(m_1,m_2,t)\big) \le e^{-t}$. Let $M := \mathrm{Med}(\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle)$ denote its median.

Case 1. If $D_1 t^{1/2}(\mu_1\sigma + \sigma^2 m_2^{1/2}) \le \frac{\gamma}{4}\bar{g}_k(\mathbf{A}\mathbf{A}^\top)$. Then, $\delta_1 \le (1-\frac{\gamma}{2})\frac{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}{2} = \frac{1-2\gamma'}{1+2\gamma'}\frac{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}{2}$ for some $\gamma' \in (0,1/2)$. By Lemma 21, $F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\cdot)$ satisfies the Lipschitz condition. By definition of $F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z})$, we have $F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}) = \langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle$ on the event $\{\|\hat{\mathbf{\Gamma}}\| \le \delta_1, \|\mathbf{Z}\| \le \delta_2\}$. By Lemma 14 and $t \ge \log 8$,

$$\begin{aligned}&\mathbb{P}\Big\{F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}) \ge M\Big\}\\ &\quad\ge \mathbb{P}\Big\{F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}) \ge M, \quad \|\hat{\mathbf{\Gamma}}\| \le \delta_1, \quad \|\mathbf{Z}\| \le \delta_2\Big\}\\ &\quad\ge \mathbb{P}\Big\{\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle \ge M\Big\} - \mathbb{P}\{\|\hat{\mathbf{\Gamma}}\| \le \delta_1, \|\mathbf{Z}\| \le \delta_2\}\\ &\quad\ge \mathbb{P}\Big\{\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle \ge M\Big\} - \mathbb{P}\Big\{\|\hat{\mathbf{\Gamma}}\| \le \delta_1\Big\} - \mathbb{P}\Big\{\|\mathbf{Z}\| \le \delta_2\Big\}\\ &\quad\ge \frac{1}{2} - \frac{1}{8} - \frac{1}{8} = 1/4,\end{aligned}$$

and similarly,

$$\mathbb{P}\Big\{F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}) \le M)\Big\} \ge 1/4.$$

It follows from Gaussian isoperimetric inequality (see (Koltchinskii and Xia, 2016, Lemma 2.6)) and Lemma 21 that with some constant $D_\gamma > 0$, for all $t \ge \log 8$ with probability at least $1 - e^{-t}$,

$$\big|F_{\delta_1,\delta_2,\mathbf{x},\mathbf{y}}(\mathbf{Z}) - M\big| \le D_\gamma \frac{\sigma\delta_1 t^{1/2}}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\Big(\mu_1 + \delta_2 + \frac{\delta_1}{\delta_2}\Big)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.$$

Since $t \le m_1 \le m_2$, it is easy to check by Lemma 14 that $\delta_1 \asymp \sigma\mu_1 m_1^{1/2} + \sigma^2(m_1m_2)^{1/2}$ and $\delta_2 \asymp \sigma m_2^{1/2}$. Moreover, $\mathbb{P}\big\{\|\hat{\mathbf{\Gamma}}\| \le \delta_1, \|\mathbf{Z}\| \le \delta_2\big\} \ge 1 - 2e^{-t}$. As a result, with probability at least $1 - e^{-3t}$,

$$\big|\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle - M\big| \le D_\gamma \frac{\sigma\mu_1 t^{1/2} + \sigma^2 m_2^{1/2} t^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}. \tag{33}$$

Case 2. If $D_1 t^{1/2}(\sigma\mu_1 + \sigma^2 m_2^{1/2}) > \frac{\gamma}{4}\bar{g}_k(\mathbf{A}\mathbf{A}^\top)$. It implies that

$$\mathbb{E}\|\hat{\mathbf{\Gamma}}\| \le D_1\frac{(1-\gamma)}{\gamma}t^{1/2}(\sigma\mu_1 + \sigma^2 m_2^{1/2}),$$

and $\delta_1 \le D_\gamma t^{1/2}(\sigma\mu_1 + \sigma^2 m_2^{1/2})$. By Lemma 14 and Lemma 16, with probability at least $1 - e^{-t}$,

$$|\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle| \le \|\mathbf{S}_k(\hat{\mathbf{\Gamma}})\| \le D_\gamma t\frac{(\sigma\mu_1 + \sigma^2 m_2^{1/2})^2}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2},$$

which immediately yields that

$$M \le D_\gamma\frac{(\sigma\mu_1 + \sigma^2 m_2^{1/2})^2}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.$$

The above inequalities imply that with probability at least $1 - e^{-t}$ for $\log 8 \le t \le m_1$,

$$\begin{aligned}|\langle \mathbf{x}, \mathbf{S}_k(\hat{\mathbf{\Gamma}})\mathbf{y}\rangle - M| \le& D_\gamma t\frac{(\sigma\mu_1 + \sigma^2 m_2^{1/2})^2}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}\\ \le& D_\gamma\frac{\sigma\mu_1 t^{1/2} + \sigma^2 m_2^{1/2} t^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}.\end{aligned} \tag{34}$$

Therefore, bounds (33) and (34) hold in both cases. The rest of the proof is quite standard by integrating the exponential tails and will be skipped here, see Koltchinskii and Xia (2016). ■

Proof [Proof of Theorem 15] By Lemma 19 and Lemma 22, if $D_1\delta(m_1,m_2) \le \bar{g}_k(\mathbf{A}\mathbf{A}^\top)$ for a large enough constant $D_1 > 0$ such that $\gamma \le 1/2$, we conclude that for all $\log 8 \le t \le m_1$, with probability at least $1 - 2e^{-t}$,

$$\big|\langle \mathbf{x}, \hat{\mathbf{P}}_k\mathbf{y}\rangle\big| \le Dt^{1/2}\frac{\sigma\mu_1 + \sigma^2 m_2^{1/2}}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\|\mathbf{x}\|_{\ell_2}\|\mathbf{y}\|_{\ell_2}$$

which concludes the proof after adjusting the constant D accordingly. ■

Appendix C. Proof of Lemma 18

Observe that for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{m_1}$ with $\|\mathbf{x}\|_{\ell_2} = \|\mathbf{y}\|_{\ell_2} = 1$ and $\delta_t = \mathbb{E}\|\hat{\mathbf{\Gamma}}\| + D_1\sigma\mu_1 t^{1/2} + D_2\sigma^2 m_2^{1/2} t^{1/2}$ with $t \le m_1$ and some $\gamma \in (0, 1/2]$,

$$\begin{aligned}
\Big|\mathbb{E}\big\langle \mathbf{x}, \big(\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbf{S}_k(\hat{\mathbf{\Gamma}})\big)\mathbf{y}\big\rangle\Big| &\le \mathbb{E}\Big\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbf{S}_k(\hat{\mathbf{\Gamma}})\Big\| \\
&= \mathbb{E}\Big\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbf{S}_k(\hat{\mathbf{\Gamma}})\Big\|\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le (1+\gamma)\delta_t\Big)\mathbf{1}\Big(\|\hat{\mathbf{\Gamma}}\| \le (1+\gamma)\delta_t\Big) \\
&+ \mathbb{E}\Big\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbf{S}_k(\hat{\mathbf{\Gamma}})\Big\|\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le (1+\gamma)\delta_t\Big)\mathbf{1}\Big(\|\hat{\mathbf{\Gamma}}\| > (1+\gamma)\delta_t\Big) \\
&+ \mathbb{E}\Big\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbf{S}_k(\hat{\mathbf{\Gamma}})\Big\|\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| > (1+\gamma)\delta_t\Big)\mathbf{1}\Big(\|\hat{\mathbf{\Gamma}}\| \le (1+\gamma)\delta_t\Big) \\
&+ \mathbb{E}\Big\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbf{S}_k(\hat{\mathbf{\Gamma}})\Big\|\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| > (1+\gamma)\delta_t\Big)\mathbf{1}\Big(\|\hat{\mathbf{\Gamma}}\| > (1+\gamma)\delta_t\Big)
\end{aligned}$$

where the constants $D_1, D_2 > 0$ are chosen such that $\max\big\{\mathbb{P}\big(\|\widetilde{\mathbf{\Gamma}}\| \ge \delta_t\big), \mathbb{P}\big(\|\hat{\mathbf{\Gamma}}\| \ge \delta_t\big)\big\} \le e^{-t}$. By Lemma 20,

$$\begin{aligned}
&\mathbb{E}\Big\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbf{S}_k(\hat{\mathbf{\Gamma}})\Big\|\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le (1+\gamma)\delta_t\Big)\mathbf{1}\Big(\|\hat{\mathbf{\Gamma}}\| \le (1+\gamma)\delta_t\Big) \\
&\quad\le D_\gamma \frac{\delta_t}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\mathbb{E}\|\widetilde{\mathbf{\Gamma}} - \hat{\mathbf{\Gamma}}\| \le D_\gamma \frac{\delta_t}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\mathbb{E}\|\mathbf{Z}\mathbf{P}_k^{hh}\mathbf{Z}^\top - \nu_k\sigma^2\mathbf{I}_{m_1}\|.
\end{aligned}$$

By writing $\mathbf{P}_k^{hh} := \sum_{j\in\Delta_k}\mathbf{h}_j \otimes \mathbf{h}_j$, we obtain

$$\begin{aligned}
\mathbf{Z}\mathbf{P}_k^{hh}\mathbf{Z}^\top - \sigma^2\nu_k\mathbf{I}_{m_1} &= \sum_{j\in\Delta_k}(\mathbf{Z}\mathbf{h}_j)\otimes(\mathbf{Z}\mathbf{h}_j) - \sigma^2\nu_k\mathbf{I}_{m_1} \\
&= \nu_k\Big(\frac{1}{\nu_k}\sum_{j\in\Delta_k}(\mathbf{Z}\mathbf{h}_j)\otimes(\mathbf{Z}\mathbf{h}_j) - \sigma^2\mathbf{I}_{m_1}\Big).
\end{aligned}$$

where $\nu_k = \mathrm{Card}(\Delta_k)$. The vectors $\mathbf{Z}\mathbf{h}_j \sim \mathcal{N}(0, \sigma^2\mathbf{I}_{m_1})$ and $\{\mathbf{Z}\mathbf{h}_j : \ldots, j \in \Delta_k\}$ are independent. By Koltchinskii and Lounici (2017),

$$\mathbb{E}\Big\|\frac{1}{\nu_k}\sum_{j\in\Delta_k}(\mathbf{Z}\mathbf{h}_j)\otimes(\mathbf{Z}\mathbf{h}_j) - \sigma^2\mathbf{I}_{m_1}\Big\| \lesssim \sigma^2\Big(\sqrt{\frac{m_1}{\nu_k}} \vee \frac{m_1}{\nu_k}\Big).$$

Since $\nu_k \le m_1$, we conclude with

$$\begin{aligned}
\mathbb{E}\Big\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbf{S}_k(\hat{\mathbf{\Gamma}})\Big\|\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| \le (1+\gamma)\delta_t\Big)\mathbf{1}\Big(\|\hat{\mathbf{\Gamma}}\| \le (1+\gamma)\delta_t\Big) \\
\lesssim_\gamma \frac{\delta_t}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{m_1\sigma^2}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big).
\end{aligned} \tag{35}$$

Choose $t = m_1$, by Lemma 16 and Lemma 14,

$$\begin{aligned}
&\mathbb{E}\Big\|\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbf{S}_k(\hat{\mathbf{\Gamma}})\Big\|\mathbf{1}\Big(\|\hat{\mathbf{\Gamma}}\| \le (1+\gamma)\delta_{m_1}\Big)\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| > (1+\gamma)\delta_{m_1}\Big) \\
&\quad\le D_\gamma\frac{\delta_{m_1}^2}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\mathbb{E}\frac{\|\widetilde{\mathbf{\Gamma}}\|^2}{\bar{g}_k^2(\mathbf{A}\mathbf{A}^\top)}\mathbf{1}\Big(\|\widetilde{\mathbf{\Gamma}}\| > (1+\gamma)\delta_{m_1}\Big) \\
&\quad\lesssim_\gamma \frac{\delta_{m_1}^2}{\bar{g}_k^4(\mathbf{A}\mathbf{A}^\top)}e^{-m_1/2}\mathbb{E}^{1/2}\|\widetilde{\mathbf{\Gamma}}\|^4 \lesssim \frac{\delta_{m_1}^4}{\bar{g}_k^4(\mathbf{A}\mathbf{A}^\top)}e^{-m_1/2} \\
&\quad\lesssim \frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{\sigma\mu_1 + \sigma^2 m_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big)
\end{aligned}$$

which is clearly dominated by (35) for $t = m_1$ and $m_2 e^{-m_1/2} \le 1$. The other terms are bounded in a similar fashion. To sum up, we obtain

$$\|\mathbb{E}\mathbf{S}_k(\widetilde{\mathbf{\Gamma}}) - \mathbb{E}\mathbf{S}_k(\hat{\mathbf{\Gamma}})\| \lesssim \frac{\sigma\mu_1 + \sigma^2 m_1}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big(\frac{\delta(m_1,m_2)}{\bar{g}_k(\mathbf{A}\mathbf{A}^\top)}\Big).$$

Journal of Machine Learning Research 20 (2019) 1-37 Submitted 8/17; Revised 4/19; Published 4/19

Multi-scale Online Learning: Theory and Applications to Online Auctions and Pricing

Sébastien Bubeck SEBUBECK@MICROSOFT.COM
Microsoft Research,
1 Microsoft Way,
Redmond, WA 98052, USA.

Nikhil R. Devanur NIKDEV@MICROSOFT.COM
Microsoft Research,
1 Microsoft Way,
Redmond, WA 98052, USA.

Zhiyi Huang ZHIYI@CS.HKU.HK
Department of Computer Science,
The University of Hong Kong,
Pokfulam, Hong Kong.

Rad Niazadeh* RAD@CS.STANFORD.EDU
Department of Computer Science,
Stanford University,
Stanford, CA 94305, USA.

Editor: Csaba Szepesvari

Abstract

We consider revenue maximization in online auction/pricing problems. A seller sells an identical item in each period to a new buyer, or a new set of buyers. For the online pricing problem, both when the arriving buyer bids or only responds to the posted price, we design algorithms whose regret bounds *scale with the best fixed price in-hindsight*, rather than the range of the values. Under the bidding model, we further show our algorithms achieve a revenue convergence rate that matches the offline sample complexity of the single-item single-buyer auction. We also show regret bounds that are *scale free*, and match the offline sample complexity, when comparing to a benchmark that requires a lower bound on the market share. We further expand our results beyond pricing to multi-buyer auctions, and obtain online learning algorithms for auctions, with convergence rates matching the known sample complexity upper bound of online single-item multi-buyer auctions.

These results are obtained by generalizing the classical learning from experts and multi-armed bandit problems to their *multi-scale* versions. In this version, the reward of each action is in a different range, and the regret with respect to a given action scales with its own range, rather than the maximum range. We obtain almost optimal multi-scale regret bounds by introducing a new Online Mirror Descent (OMD) algorithm whose mirror map is the multi-scale version of the negative entropy function. We further generalize to the bandit setting by introducing the stochastic variant of this OMD algorithm.

Keywords: online learning, multi-scale learning, auction theory, bandit information, sample complexity. †

∗. Zhiyi Huang is supported by an RGC grant HKU17202115E. Rad Niazadeh is supported by the Stanford Motwani Fellowship. Part of this work is done when Rad was interning at Microsoft Research lab.

†. Following the theoretical computer science convention, we used *alphabetical author ordering*.

©2019 Sébastien Bubeck, Nikhil R. Devanur, Zhiyi Huang, and Rad Niazadeh.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-498.html.

1. Introduction

Consider the following revenue maximization problem in a repeated setting, called the *online posted pricing* problem. In each period, the seller has a single item to sell, and a new prospective buyer. The seller offers to sell the item to the buyer at a given price; the buyer buys the item if and only if the price is below his private valuation for the item. The private valuation of the buyer itself is never revealed to the seller. How should a monopolistic seller iteratively set the prices if he wishes to maximize his revenue? What if he also cares about the market share, i.e., the fraction of time periods at which the item is sold?[1]

Estimating price sensitivities and demand models in order to optimize revenue and market share is one of the bedrock of revenue management. The emergence of online marketplaces has enabled sellers to costlessly change prices, as well as collect huge amounts of data. This has renewed the interest in understanding best practices for data driven pricing. The extreme case of this when the price is updated for each buyer is the online pricing problem described above; one can always use this for less frequent price updates. Moreover this problem is intimately related to the classical experimentation and estimation procedures.

This problem has been studied from an *online learning* perspective, as a variant of the *multi-armed bandit* problem. In this variant, there is an arm for each possible price (presumably after an appropriate discretization). The revenue of each arm p is either p or zero, depending on whether the value of the arriving buyer is at least equal to the price p or smaller than the price p, respectively. The total revenue of a pricing algorithm is then compared to the total revenue of the best fixed posted price in hindsight. The difference between the two, called the *regret*, is then bounded from above. No assumption is made on the distribution of values; the regret bounds are required to hold for the *worst case* sequence of values. Blum et al. (2004) assume that the buyer valuations are in $[1, h]$, and show the following multiplicative plus additive bound on the regret: for any $\epsilon \in (0, 1)$, the regret is at most ϵ times the revenue of the optimal price, plus $O(\epsilon^{-2} h \log h \log\log h)$. Blum and Hartline (2005) show that the additive factor can be made to be $O(\epsilon^{-3} h \log\log h)$, trading off a $\log h$ factor for an extra ϵ^{-1} factor.

An undesirable aspect of these bounds is that they scale *linearly with* h; this is particularly problematic when h is an estimate and we might set it to be a generous upper bound on the range of prices we wish to consider. A typical use case is when the same algorithm is used for many different products, with widely varying price ranges. We may not be able to manually tune the range for each product separately.

One might wonder if this dependence on h is unavoidable, as it seems to be reflected by the existing lower bounds for this problem in the literature (lower bounds are discussed later in the introduction with more details). Interestingly, in all of these lower-bound instances the best fixed price is equal to h itself; Therefore, it is not clear whether this dependency on h is required for instances where h is only a pessimistic upper-bound on the best fixed price. We now ask the following question:

> **Question**: *do online learning algorithms exist for the online posted pricing problem, such that their regrets are proportional to the best fixed price instead of the highest value?*

1. As we will see later in the paper, we consider designing pricing algorithms that have vanishing revenue regret with respect to all prices, or only prices with a given minimum market share.

Standard off-the-shelf bounds, e.g., those obtained by using classic results on full information online learning (Arora et al., 2012) or bandit information online learning (Auer et al., 1995; Bubeck, 2011), allow regret to depend on the loss of the best arm instead of the worst case loss. However, even such bounds still depend linearly on the maximum range of all the losses, and thus they would not allow to replace h by the best fixed price.

Fortunately, in the online pricing problem the reward function of the arms is well structured. In particular, as a neat observation, the reward of the arm p is upper-bounded by p (and not only the maximum value). Can we use this structure in our favor to improve the standard regret bounds? We answer this question in the affirmative by the means of reducing the problem to a pure learning problem termed as *mutli-scale online learning*. Notably, this structural observation is the same structural observation that enabled the improved bounds of Blum and Hartline (2005) over the prior work at the time.

1.1. Multi-scale online learning

The main technical ingredients in our results are variants of the classical problems of learning from expert advice and multi-armed bandit. We introduce the multi-scale versions of these problems, where each action has its reward bounded in a different range. We assume these ranges are known in advance by any online algorithm. Here, we seek to design online learning algorithms that guarantee *multi-scale regret* bounds, i.e. their regrets with respect to each certain action scales with the range of that particular action, instead of the maximum possible range. These guarantees are in contrast with the regret bounds of the standard versions, which scale with the maximum range.

> **Main result (informal):** *we give algorithms for the full information and bandit information versions of the multi-scale online learning problem with information theoretically optimal (or almost optimal) multi-scale regret guarantees.*

While we use these bounds mostly for designing online auctions and pricing mechanisms, we expect such bounds to be of independent interest.

The main idea behind our algorithms is to use a tailored variant of *online (stochastic) mirror descent (OSMD)* (Bubeck, 2011). In this tailored version, the algorithm uses a *weighted negative entropy* as the Legendre function (also known as the *mirror map*), where the weight of each term i (corresponding to arm i) is actually equal to the range of that arm. More formally, assuming the range of arm i is equal to c_i (i.e., reward of arm i is in $[0, c_i]$), our mirror descent algorithms (Algorithm 1 for full information, and Algorithm 3 for the bandit information) use the following mirror map:

$$F(x) = \sum_{\text{arms } i} c_i \cdot x_i \ln(x_i)$$

Intuitively speaking, these algorithms take into account different ranges for different arms by first normalizing the reward of each arm by its range (i.e. divide the reward of arm i by its corresponding range c_i), and then projecting the updated weights by performing a *smooth multi-scale projection* into the simplex. This projection is an instance of the more general Bregman projection (Bubeck, 2011) for the special case of weighted negative entropy as the mirror map. The mirror descent framework then gives regret bounds in terms of a

"local norm" as well as an "initial divergence", which we then bound differently for each version of the problem. In the technical sections we highlight how the subtle variations arise as a result of different techniques used to bound these two terms.

While our algorithms have the style of the multiplicative weights update (up to a normalization of the rewards), the smooth projection step at each iteration makes them drastically different. To shed some insight on this projection step, which plays an important role in our analysis, consider a very special case of the problem where the reward of each arm i is deterministically equal to c_i. The multiplicative weights algorithm picks arm i with a probability proportional to $\exp(c_i)$. However, as it is clear from the description of Algorithm 1, our algorithm uniformly scales the weight of each arm first. Then, in the projection step the weight of each arm i is multiplied by $\exp(-\frac{\lambda^*}{c_i})$ for some parameter λ^*. Hence, arm i will be sampled with a probability proportional to $\exp(-\frac{\lambda^*}{c_i})$ (which is a smooth approximation to $i^* = \text{argmax } c_i$, but in a different way compared to the vanilla multiplicative weights).

The multi-scale versions exhibit subtle variations that do not appear in the standard versions. First of all, our applications to auctions and pricing have non-negative rewards, and this actually makes a difference. For both the expert and the bandit versions, the minimax regret bounds for non-negative rewards are *provably better* than those when rewards could be negative. Further, for the bandit version with non-negative rewards, we can prove a better bound (in terms of the dependency on time horizon T) if we only require the bound to hold with respect to the *best* action, rather than *all* actions (which was the requirement of the vanilla version of our definition for a multi-scale regret bound). The various regret bounds and comparison to standard bounds are summarized in Tables 1.

	Standard regret bound $O(\cdot)$	Multi-scale bound (this paper)	
		Upper bound $O(\cdot)$	Lower bound $\Omega(\cdot)$
Experts/non-negative	$c_{\max}\sqrt{T\log(k)}$ *	$c_i\sqrt{T\log(kT)}$	$c_i\sqrt{T\log(k)}$
Bandits/non-negative	$c_{\max}\sqrt{Tk}$ †	$c_i T^{\frac{2}{3}}(k\log(kT))^{\frac{1}{3}}$	$c_i\sqrt{TK}$
		$c_{i^*}\sqrt{Tk\log(k)}$, i^* is the best action	-
Experts/symmetric	$c_{\max}\sqrt{T\log(k)}$ *	$c_i\sqrt{T\log(k\cdot\frac{c_{\max}}{c_{\min}})}$	$c_i\sqrt{T\log(k)}$
Bandits/symmetric	$c_{\max}\sqrt{Tk}$ †	$c_i\sqrt{Tk\cdot\frac{c_{\max}}{c_{\min}}\log(kT\cdot\frac{c_{\max}}{c_{\min}})}$	$c_i\sqrt{Tk\cdot\frac{c_{\max}}{c_{\min}}}$

* Freund and Schapire (1995); † Audibert and Bubeck (2009).

Table 1: Pure-additive regret bounds for non-negative rewards, i.e. when reward of any action i at any time is in $[0, c_i]$, and symmetric range rewards, i.e. when reward of any action i at any time is in $[-c_i, c_i]$ (suppose T is the time horizon, A is the action set, and k is the number of actions).

1.2. The implications for online auctions and pricing

As a direct application of our multi-scale online learning framework, somewhat surprisingly,

Second contribution: *we show that we can get regret proportional to the best fixed price instead of the highest value for the online posted pricing problem.*

(i.e., we can replace h by the best fixed price, which is used in the definition of the benchmark). In particular, we show that the additive bound can be made to be $O(\epsilon^{-2}p^* \log h)$, where p^* is the best fixed price in hindsight. This allows us to use a very generous estimate for h and let the algorithm adapt to the actual range of prices; we only lose a $\log h$ factor. The algorithm balances exploration probabilities of different prices carefully and automatically zooms in on the relevant price range. This does not violate known lower bounds, since in those instances p^* is close to h.

Bar-Yossef et al. (2002), Blum et al. (2004), and Blum and Hartline (2005) also consider the "full information" version of the problem, or what we call the *online (single buyer) auction* problem, where the valuations of the buyers are revealed to the algorithm after the buyer has made a decision. Such information may be available in a context where the buyers have to bid for the items, and are awarded the item if their bid is above a hidden price. In this case, the additive term can be improved to $O(\epsilon^{-1}h \log(\epsilon^{-1}))$, which is tight. Once again, by a reduction to multi-scale online learning, *we show that h can be replaced with p^**; in particular, we show that the additive term can be made to be $O(\epsilon^{-1}p^* \log(h\epsilon^{-1}))$.

1.3. Purely multiplicative bounds and sample complexity

The regret bounds mentioned above can be turned into a purely multiplicative factor in the following way: for any $\epsilon > 0$, the algorithm is guaranteed to get a $1 - O(\epsilon)$ fraction of the best fixed price revenue, provided the number of periods $T \geq E/\epsilon$, where E is the additive term in the regret bounds above. This follows from the observation that a revenue of T is a lower bound on the best fixed price revenue. Define the number of periods required to get a $1 - \epsilon$ multiplicative approximation (as a function of ϵ) to be the *convergence rate* of the algorithm.

A $1 - \epsilon$ multiplicative factor is also the target in the recent line of work, on the *sample complexity* of auctions, started by Balcan et al. (2008); Elkind (2007); Dhangwatnotai et al. (2014); Cole and Roughgarden (2014). (We give a more comprehensive discussion of this line of work in Section 1.4.) Here, *i.i.d.* samples of the valuations are given from a *fixed but unknown distribution*, and the goal is to find a price such that its revenue with respect to the hidden distribution is a $1 - \epsilon$ fraction of the optimum revenue for this distribution. The sample complexity is the minimum number of samples needed to guarantee this (as a function of ϵ).

The sample complexity and the convergence rate (for the full information setting) are closely related to each other. The sample complexity is always smaller than the convergence rate: the problem is easier because of the following.

1. The valuations are i.i.d. in the case of sample complexity, whereas they can be arbitrary (worst case) in the case of convergence rate.

2. Sample complexity corresponds to an offline problem: you get all the samples at once. Convergence rate corresponds to an online problem: you need to decide what to do on a given valuation without knowing what valuations arrive in the future.

This is formalized in terms of an *online to offline reduction* [folklore] which shows that a convergence rate upper bound can be automatically translated to a sample complexity upper bound. This lets us convert sample complexity lower bounds into lower bounds on the convergence rate, and in turn into lower bounds on the additive error E in an additive plus multiplicative regret bound. For example, the additive error for the online auction problem (and hence also for the posted pricing problem[2]) cannot be $o(h\epsilon^{-1})$ (Huang et al., 2015b). Moreover, it is insightful to compare convergence rates we show with *the best known sample complexity upper bound; proving better convergence rates would mean improving these bounds as well.*

A natural target convergence rate for a problem is therefore the corresponding sample complexity, but achieving this is not always trivial. In particular, we consider an interesting version of the sample complexity bound for auctions, for which no analogous convergence rate bound is known in the literature. This version takes into account *both revenue and market share*, and gets sample complexity bounds that are *scale free*; there is no dependence on h, which means it works for unbounded valuations! For any $\delta \in (0, 1)$, the best fixed price benchmark is relaxed to ignore those prices whose market share (which is equivalent to the probability of sale) is below a δ fraction; as δ increases the benchmark is lower. This is a meaningful benchmark since in many cases revenue is not the only goal, even if you are a monopolist. A more reasonable goal is to maximize revenue subject to the constraint that the market share is above a certain threshold. What is more, this gives a sample complexity of $O(\epsilon^{-2}\delta^{-1} \log(\delta^{-1}\epsilon^{-1}))$ (Huang et al., 2015b). In fact δ can be set to h^{-1} without loss of generality, when the values are in $[1, h]$,[3] and the above bound then matches the sample complexity with respect to the best fixed price revenue. In addition, this bound gives a precise interpolation: as the target market share δ increase, the number of samples needed decreases almost linearly.

Third contribution: *we show an average revenue convergence rate that almost matches the above sample complexity, for the full information setting, i.e. number of rounds required for this convergence matches the number of i.i.d. samples required for an offline algorithm to attain the same near-optimality guarantee.*

We have a mild dependence on h; the rate is proportional to $\log \log h$. Further, we also show a near optimal convergence rate for the online posted pricing problem.[4]

Multiple buyers: All of our results in the full information (online auction) setting extend to the multiple buyer model. In this model, in each time period, a new set of n buyers competes for a single item. The seller runs a truthful auction that determines the winning buyer and his payment. The benchmark here is the set of all "Myerson-type" mechanisms. These are mechanisms that are optimal when each period has n buyers of potentially different types, and the value of each buyer is drawn independently from a type dependent

2. We conjecture that the lower bound for the posted pricing problem should be worse by a factor of ϵ^{-1}, since one needs to explore about ϵ^{-1} different prices.
3. When the values are in $[1, h]$, we can guarantee a revenue of T by posting a price of 1, and to beat this, any other price (and in particular a price of h) would have to sell at least T/h times.
4. Unfortunately, we cannot yet guarantee that our online algorithm itself gets a market share of δ, although we strongly believe that it does. Showing such bounds on the market share of the algorithm is an important avenue for future research.

distribution. In fact, our convergence rates also imply new sample complexity bounds for these problems (except that they are not computationally efficient).

The various bounds and comparisons to previous work are summarized in Tables 2 & 3.

	Lower bound	Upper bound		
		Best known (Sample complexity)	Best known (Convergence rate)	This paper (Thm. 16)
Online single buyer auction	$\Omega(\frac{h}{\epsilon^2})$ *	$\tilde{O}(\frac{h}{\epsilon^2})$ †	$\tilde{O}(\frac{h}{\epsilon^2})$ †	$\tilde{O}(\frac{p^*}{\epsilon^2})$
Online posted pricing	$\Omega(\max\{\frac{h}{\epsilon^2}, \frac{1}{\epsilon^3}\})$ *§	-	$\tilde{O}(\frac{h}{\epsilon^3})$ †	$\tilde{O}(\frac{p^*}{\epsilon^3})$
Online multi buyer auction	$\Omega(\frac{h}{\epsilon^2})$ *	$O(\frac{nh}{\epsilon^3})$ ‡	-	$\tilde{O}(\frac{nh}{\epsilon^3})$

* Huang et al. (2015b); † Blum et al. (2004); ‡ Devanur et al. (2016); Gonczarowski and Nisan (2017); Elkind (2007); § Kleinberg and Leighton (2003).

Table 2: Number of rounds/samples needed to get a $1-\epsilon$ approximation to the best offline price/mechanism. Sample complexity is for the offline case with i.i.d. samples from an unknown distribution. Convergence rate is for the online case with a worst case sequence. Sample complexity is always no larger than the convergence rate. Lower bounds hold for sample complexity too, except for the online posted pricing problem for which there is no sample complexity version. The additive plus multiplicative regret bounds are converted to convergence rates by dividing the additive error by ϵ. In the last row, n is the number of buyers. In the last column, p^* denotes the optimal price.

	Lower bound (Sample complexity)	Upper bound	
		Best known (Sample complexity)	This paper (Thm. 17)
Online single buyer auction	$\Omega(\frac{1}{\epsilon^2\delta})$ *	$\tilde{O}(\frac{1}{\epsilon^2\delta})$ *	$\tilde{O}(\frac{1}{\epsilon^2\delta})$
Online posted pricing	$\Omega(\max\{\frac{1}{\epsilon^2\delta}, \frac{1}{\epsilon^3}\})$ *†	-	$\tilde{O}(\frac{1}{\epsilon^4\delta})$
Online multi buyer auction	$\Omega(\frac{1}{\epsilon^2\delta})$ *	-	$\tilde{O}(\frac{n}{\epsilon^3\delta})$

* Huang et al. (2015b); † Kleinberg and Leighton (2003).

Table 3: Sample complexity & convergence rate w.r.t. the opt mechanism/price with market share $\geq \delta$.

1.4. Other related work

The online pricing problem, also called *dynamic pricing*, is a much studied topic, across disciplines such as operations research and management science (Talluri and Van Ryzin, 2006), economics (Segal, 2003), marketing, and of course computer science. The multi-armed bandit approach to pricing is particularly popular. See den Boer (2015) for a recent survey on various approaches to the problem.

Kleinberg and Leighton (2003) consider the online pricing problem, under the assumption that the values are in $[0,1]$, and considered purely additive factors. They showed that the minimax additive regret is $\tilde{\Theta}(T^{2/3})$, where T is the number of periods. This is similar in spirit to regret bounds that scale with h, since one has to normalize the values so that they are in $[0,1]$. The finer distinction about the magnitude of the best fixed price is absent in this work. Recently, Syrgkanis (2017) also consider the online auction problem, with an emphasis on a notion of "oracle based" computational efficiency. He assume the values are all in $[0,1]$ and do not consider the scaling issue that we do; this makes their contribution orthogonal to ours.

Starting with Dhangwatnotai et al. (2014); Elkind (2007), there has been a spate of recent results analyzing the sample complexity of pricing and auction problems. Cole and Roughgarden (2014) and Devanur et al. (2016) consider multiple buyer auctions with regular distributions (with unbounded valuations) and give sample complexity bounds that are polynomial in n and ϵ^{-1}, where n is the number of buyers. Morgenstern and Roughgarden (2015) consider arbitrary distributions with values bounded by h, and gave bounds that are polynomial in n, h, and ϵ^{-1}. Roughgarden and Schrijvers (2016); Huang et al. (2015b) give further improvements on the single- and multi-buyer versions respectively; Tables 2 and 3 give a comparison of these results with our bounds, for the problems we consider. The dynamic pricing problem has also been studied when there are a given number of copies of the item to sell (limited supply) (Agrawal and Devanur, 2014; Babaioff et al., 2015; Badanidiyuru et al., 2013; Besbes and Zeevi, 2009). There are also variants where the seller interacts with the same buyer repeatedly, and the buyer can strategize to influence his utility in the future periods (Amin et al., 2013).

Foster et al. (2017) also consider the multi-scale online learning problem motivated by a model selection problem. They consider additive bounds, for the symmetric case, for full information, but not bandit feedback. Their regret bounds are not comparable to ours in general; our bounds are better for the pricing/auction applications we consider, and their bounds are better for their application.

Organization We start in Section 2 by showing regret upper bounds for the multi-scale experts problem with non-negative rewards (Theorem 1). The corresponding upper bounds for the bandit version are in section 3 (Theorem 12). In Section 4 we show how the multi-scale regret bounds (Theorems 1 and 12) imply the corresponding bounds for the auction/pricing problems (Theorems 16 and 17). Finally, the regret (upper and lower) bounds for the symmetric range are discussed in Section 5 (Theorems 18, 20, 21, and 23).

2. Full Information Multi-scale Online Learning

We consider a variety of online algorithmic problems that are all parts of the *multiscale online learning* framework. We start by defining this framework, in which different actions have different ranges. We exploit this structure and express our results in terms of *action-specific* regret bounds for this general problem. To obtain these results, we use a variant of online mirror descent and propose a multiplicative-weight update style learning algorithm for our problem, termed as *Multi-Scale Multiplicative-Weight (MSMW)* algorithm.

Next, we investigate the single buyer auction problem (or equivalently the full-information single buyer dynamic pricing problem) as a canonical application, and show how to get mul-

tiplicative cum additive approximations here by the help of the multi-scale online learning framework. To show the tightness of our bounds, we compare the convergence rate of our dynamic pricing with the sample complexity of a closely related offline problem, i.e. the near optimal Bayesian revenue maximization from samples (Cole and Roughgarden, 2014).

2.1. The framework

Our full-information multi-scale online learning framework is basically the classical learning from expert advice problem. The main difference is that the *range* of rewards of different experts could be different. More formally, suppose there is a set of actions A.[5] The online problem proceeds in T rounds, where in each round $t \in [T]$:[6]

- The adversary picks a reward function $\mathbf{g}(t)$, where $g_i(t)$ is the reward of action i.
- The algorithm picks an action $i_t \in A$ simultaneously.
- Then the algorithm gets the reward $g_{i_t}(t)$ and observes the entire reward function $\mathbf{g}(t)$.

The total reward of the algorithm is denoted by

$$G_{\text{ALG}} := \sum_{t=1}^{T} g_{i_t}(t).$$

The standard "best fixed action" benchmark is

$$G_{\text{MAX}} := \max_{i \in A} \sum_{t=1}^{T} g_i(t).$$

We further assume that the action set is countable, either finite or infinite (will discuss this later). Without loss of generality, if the action set is of size k, we identify $A = [k]$. The reward $\mathbf{g}(t)$ is such that for all $i \in A$, $g_i(t) \in [0, c_i]$, where $c_i \in \mathbb{R}_+$ is the *range* of action i.

2.2. Multi-scale regret bounds

We prove action-specific regret bounds, which we call also *multi-scale regret guarantees*. Towards this end, we define the following quantities.

$$G_i := \sum_{t \in [T]} g_i(t) \ , \tag{1}$$

$$\text{REGRET}_i := G_i - G_{\text{ALG}} \ . \tag{2}$$

The regret bound w.r.t. action i, i.e., an upper bound on $\mathbb{E}[\text{REGRET}_i]$, depends on the range c_i, as well as any *prior* distribution $\boldsymbol{\pi}$ over the action set A; this way, we can handle countably many actions. Let $c_{\min} = \inf_{i \in A} c_i$ and $c_{\max} = \sup_{i \in A} c_i$ (if applicable) be the minimum and the maximum range. We first state a version of the regret bound which is parameterized by $\epsilon > 0$; such bounds are stronger than $\sqrt{T}$ type bounds which are more standard.

5. We use the terms experts, arms and actions interchangeably in this paper.
6. We use the notation $[n] := \{1, 2, \ldots, n\}$, for any $n \in \mathbb{N}$.

Theorem 1 (Main Result) *There exists an algorithm for the full-information multi-scale online learning problem that takes as input any distribution $\boldsymbol{\pi}$ over A, the ranges c_i, $\forall\, i \in A$ and a parameter $0 < \epsilon \leq 1$, and satisfies:*

$$\forall i \in A: \quad \mathbb{E}[\text{REGRET}_i] \leq \epsilon \cdot G_i + O\left(\frac{1}{\epsilon} \log\left(\frac{1}{\epsilon \pi_i}\right) \cdot c_i\right) \tag{3}$$

Compare this to what you get by using the standard analysis for the experts problem (Arora et al., 2012), where the second term in the regret bound is $O\big(\frac{1}{\epsilon}\log(k) \cdot c_{\max}\big)$. Choosing $\boldsymbol{\pi}$ to be the uniform distribution in the above theorem gives $O\big(\frac{1}{\epsilon}\log\left(\frac{k}{\epsilon}\right) \cdot c_i\big)$. Also, one can compare the pure-additive version of this bound with the classic pure-additive regret bound $O\big(c_{\max} \cdot \sqrt{T\log(k)}\big)$ for the experts problem by setting $\epsilon = \sqrt{\frac{\log(kT)}{T}}$ (Corollary 2).

Corollary 2 *There exists an algorithm for the full-information multi-scale online learning problem that takes as input the ranges c_i, $\forall\ i \in A$, and satisfies:*

$$\forall i \in A: \quad \mathbb{E}[\text{REGRET}_i] \leq O\left(c_i \cdot \sqrt{T \log(kT)}\right) \tag{4}$$

Remark 3 *We should assert that in a multi-scale regret guarantee, we provide a separate regret bound for each action, where the bound on the regret of action i only scales linearly with c_i. This type of guarantee should "not" be mistaken as a bound on the worst action.*

Here is the map of the rest of this section. In Section 2.3 we propose an algorithm that exploits the reward structure, and later in Section 2.4 we show how this algorithm is an online mirror descent with weighted negative entropy as its mirror map. For reward-only instances, we prove the regret bound in Section 2.5. We finally turn our attention to the single buyer online auction problem in Section 2.6.

2.3. Multi-Scale Multiplicative-Weight (MSMW) algorithm

We achieve our regret bound in Theorem 1 by using the MSMW algorithm (Algorithm 1). The main idea behind this algorithm is to take into account different ranges for different experts, and therefore:

1. We normalize the reward of each expert accordingly, i.e. divide the reward of expert i by its corresponding range c_i;
2. We project the updated weights by performing a *smooth multi-scale projection* into the simplex: the algorithm finds a λ^* such that multiplying the current weight of each expert i by $\exp\left(-\frac{\lambda^*}{c_i}\right)$ makes a probability distribution over the experts. It then uses this resulting probability distribution for sampling the next expert.

2.4. Equivalence to online mirror descent with weighted negative entropy

While it is possible to analyze the regret of the MSMW algorithm (Algorithm 1) by using first principles, we take a different approach (the elementary analysis can still be found in the appendix, Section A.2). We show how this algorithm is indeed an instance of the Online Mirror Descent (OMD) algorithm for a particular choice of the *Legendre function* (also known as *the mirror map*).

Algorithm 1 MSMW

input initial distribution $\boldsymbol{\mu}$ over A, learning rate $0 < \eta \leq 1$.
initialize $\mathbf{p}(1)$ such that $p_i(1) = \mu_i$ for all $i \in A$.
for $t = 1, \ldots, T$ **do**
  Randomly pick an action drawn from $\mathbf{p}(t)$, and observe $\mathbf{g}(t)$.
  $\forall i \in A: \quad w_i(t+1) \leftarrow p_i(t) \cdot \exp(\eta \cdot \frac{g_i(t)}{c_i})$.
  Find λ^* (e.g., binary search) s.t. $\sum_{i \in A} w_i(t+1) \cdot \exp(-\frac{\lambda^*}{c_i}) = 1$.
  $\forall i \in A: \quad p_i(t+1) \leftarrow w_i(t+1) \cdot \exp(-\frac{\lambda^*}{c_i})$.
end for

2.4.1. Preliminaries on online mirror descent.

Fix an open convex set $\mathcal{D}$ and its closure $\bar{\mathcal{D}}$, which in our case are $(0, +\infty)^A$ and $[0, +\infty)^A$ respectively, and a closed-convex action set $\mathcal{A} \subset \bar{\mathcal{D}}$, which in our case is Δ_A, i.e. the set of all probability distributions over experts in A. At the heart of an OMD algorithm there is a *Legendre* function $F : \bar{\mathcal{D}} \to \mathbb{R}$, i.e. a strictly convex function that admits continuous first order partial derivatives on $\mathcal{D}$ and $\lim_{x \to \bar{\mathcal{D}} \setminus \mathcal{D}} \|\nabla F(x)\| = +\infty$, where $\nabla F(.)$ denotes the gradient map of F. One can think of OMD as a member of *projected gradient descent* algorithms, where the *gradient update* happens in the *dual space* $\nabla F(\mathcal{D})$ rather than in primal $\mathcal{D}$, and the *projection* is defined by using the *Bregman divergence* associated with F rather than ℓ_2-distance (see Figure 2.4.1).

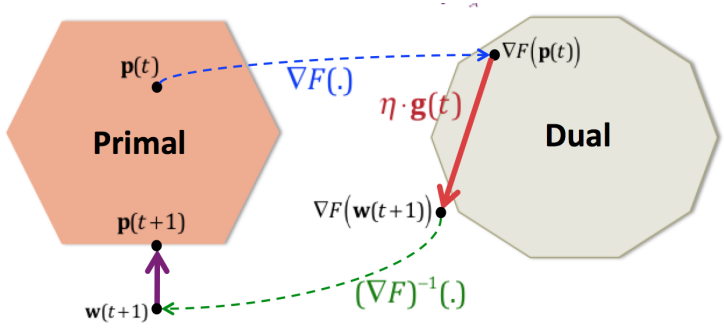


Figure 1: Online Mirror Descent (OMD): moving to the dual space by gradient map *(blue)*, gradient update in the dual space *(red)*, applying the inverse gradient map *(green)*, and finally projecting back to the simplex using Bregman projection *(purple)*.

Definition 4 (Bregman Divergence (Bubeck, 2011)) *Given a Legendre function F over Δ_A, the Bregman divergence associated with F, denoted as $D_F : \Delta_A \times \Delta_A \to \mathbb{R}$, is defined by*

$$D_F(x, y) = F(x) - F(y) - (x - y)^T \nabla F(y)$$

Definition 5 (Online Mirror Descent (Bubeck, 2011)) *Suppose F is a Legendre function. At every time $t \in [T]$, the online mirror descent algorithm with Legendre function F selects an expert drawn from distribution $\mathbf{p}(t)$, and then updates $\mathbf{w}(t)$ and $\mathbf{p}(t)$ given rewards $\mathbf{g}(t)$ by:*

Gradient update:

$$\nabla F(\mathbf{w}(t+1)) = \nabla F(\mathbf{p}(t)) + \eta \cdot \mathbf{g}(t) \Rightarrow \mathbf{w}(t+1) = (\nabla F)^{-1}(\nabla F(\mathbf{p}(t)) + \eta \cdot \mathbf{g}(t)) \quad (5)$$

Bregman projection:

$$\mathbf{p}(t+1) = \underset{\mathbf{p} \in \Delta_A}{argmin}\, (D_F(\mathbf{p}, \mathbf{w}(t+1))) \quad (6)$$

where $\eta > 0$ is called the learning rate of OMD.

We use the following standard regret bound of OMD (Refer to Bubeck (2011) for a thorough discussion on OMD. For completeness, a proof is also provided in the appendix, Section A.3). Roughly speaking, this lemma upper-bounds the regret by the summation of two separate terms: "local norm" (the first term), which captures the total deviation between $\mathbf{p}(t)$ and $\mathbf{w}(t+1)$, and "initial divergence" (the second term), which captures how much the initial distribution is far from the target distribution.

Lemma 6 *For any learning rate parameter $0 < \eta \leq 1$ and any benchmark distribution $\mathbf{q}$ over A, the OMD algorithm with Legendre function $F(.)$ admits the following:*

$$\sum_{t \in [T]} \mathbf{g}(t) \cdot (\mathbf{q} - \mathbf{p}(t)) \leq \frac{1}{\eta} \sum_{t \in [T]} D_F(\mathbf{p}(t), \mathbf{w}(t+1)) + \frac{1}{\eta} D_F(\mathbf{q}, \mathbf{p}(1)) \quad (7)$$

2.4.2. MSMW algorithm as an OMD

For our application, we focus on a particular choice of Legendre function that captures different learning rates proportional to c_i^{-1} for different experts, as we saw earlier in Algorithm 1. We start by defining the *weighted negative entropy* function.

Definition 7 *Given expert-ranges $\{c_i\}_{i \in A}$, the* weighted negative entropy *is defined by*

$$F(x) = \sum_{i \in A} c_i \cdot x_i \ln(x_i) \quad (8)$$

Corollary 8 *It is straightforward to see $F(x) = \sum_{i \in A} c_i \cdot x_i \ln(x_i)$ is a non-negative Legendre function over $\mathbb{R}_+^A$. Moreover, $\nabla F(x)_i = c_i(1 + \ln(x_i))$ and $D_F(x, y) = \sum_{i \in A} c_i \cdot (x_i \ln(\frac{x_i}{y_i}) - x_i + y_i)$.*

We now have the following lemma that shows Algorithm 1 is indeed an OMD algorithm.

Lemma 9 *The MSMW algorithm, i.e. Algorithm 1, is equivalent to an OMD algorithm associated with the weighted negative entropy $F(x) = \sum_{i \in A} c_i \cdot x_i \ln(x_i)$ as its Legendre function.*

Proof Look at the gradient update step of OMD, as in Equation (5), with Legendre function $F(x) = \sum_{i\in A} c_i \cdot x_i \ln(x_i)$. By using Corollary 8 we have

$$\nabla F(\mathbf{w}(t+1)) = \nabla F(\mathbf{p}(t)) + \eta \cdot \mathbf{g}(t) \Rightarrow c_i(1+\ln(w_i(t+1))) = c_i(1+\ln(p_i(t))) + \eta \cdot g_i(t)\ ,$$

and therefore, $w_i(t+1) = p_i(t) \cdot \exp(\eta \cdot \frac{g_i(t)}{c_i})$. Moreover, for the Bregman projection step we have

$$\mathbf{p}(t+1) = \operatorname*{argmin}_{\mathbf{p}\in\Delta_A} \left(D_F(\mathbf{p}, \mathbf{w}(t+1))\right) = \operatorname*{argmin}_{\mathbf{p}\in\Delta_A} \left(\sum_{i\in A} c_i \cdot (p_i \ln(\frac{p_i}{w_i(t+1)}) - p_i + w_i(t+1))\right) \tag{9}$$

This is a convex minimization over a convex set. To find a closed form solution, we look at the Lagrangian dual function $\mathcal{L}(\mathbf{p}, \lambda) \triangleq \sum_{i\in A} c_i \cdot (p_i \ln(\frac{p_i}{w_i(t+1)}) - p_i + w_i(t+1)) + \lambda(\sum_{i\in A} p_i - 1)$ and the Karush-Kuhn-Tucker (KKT) conditions $\nabla\mathcal{L}(\mathbf{p}^*, \lambda^*) = \mathbf{0}$. We have

$$c_i \cdot \ln(\frac{p_i^*}{w_i(t+1)}) + \lambda^* = 0 \Rightarrow p_i^* = w_i(t+1) \cdot \exp(-\frac{\lambda^*}{c_i}) \tag{10}$$

As $\sum_{i\in A} p_i^* = 1$, λ^* should be the unique number s.t. $\sum_{i\in A} w_i(t+1) \cdot \exp(-\frac{\lambda^*}{c_i}) = 1$, and then $p_i(t+1) = w_i(t+1) \cdot \exp(-\frac{\lambda^*}{c_i})$. So, Algorithm 1 is equivalent to OMD with weighted negative entropy as its Legendre function. ■

By combining Lemma 6, Corollary 8 and finally Lemma 9 we prove the following regret bound for the MSMW algorithm. We encourage the reader to also look at the appendix, Section A.2, for an extra proof using first principles.

Proposition 10 *For any initial distribution $\boldsymbol{\mu}$ over A, and any learning rate parameter $0 < \eta \leq 1$, and any benchmark distribution $\mathbf{q}$ over A, the MSMW algorithm satisfies:*

$$\sum_{i\in A} q_i \cdot G_i - \mathbb{E}\left[G_{\text{ALG}}\right] \leq \eta \sum_{t\in[T]} \sum_{i\in A} p_i(t) \frac{(g_i(t))^2}{c_i} + \frac{1}{\eta} \cdot \sum_{i\in A} c_i \left(q_i \ln\left(\frac{q_i}{\mu_i}\right) - q_i + \mu_i\right)\ .$$

Proof We have:

$$\sum_{i\in A} q_i \cdot G_i - \mathbb{E}\left[G_{\text{ALG}}\right] = \sum_{t\in[T]} \mathbf{q}\cdot\mathbf{g}(t) - \sum_{t\in[T]} \mathbf{p}(t)\cdot\mathbf{g}(t) = \sum_{t\in[T]} \mathbf{g}(t)\cdot\left(\mathbf{q}-\mathbf{p}(t)\right) \tag{11}$$

By applying the regret bound of OMD (Lemma 6) to upper-bound the RHS, we have

$$\sum_{i\in A} q_i \cdot G_i - \mathbb{E}\left[G_{\text{ALG}}\right] \leq \frac{1}{\eta} \sum_{t\in[T]} D_F(\mathbf{p}(t), \mathbf{w}(t+1)) + \frac{1}{\eta} D_F(\mathbf{q}, \mathbf{p}(1)) \tag{12}$$

To bound the first term in regret, a.k.a *local norm*, we have:

$$\begin{aligned} D_F(\mathbf{p}(t), \mathbf{w}(t+1)) &= \sum_{i\in A} c_i \cdot (p_i(t) \ln(\frac{p_i(t)}{w_i(t+1)}) - p_i(t) + w_i(t+1)) \\ &= \sum_{i\in A} c_i \cdot p_i(t)(-\eta \cdot \frac{g_i(t)}{c_i} - 1 + exp(\eta \cdot \frac{g_i(t)}{c_i})) \end{aligned} \tag{13}$$

Note that $\eta \cdot \frac{g_i(t)}{c_i} \in [-1, 1]$ because $g_i(t) \in [-c_i, c_i]$ and $0 < \eta \leq 1$. By $\exp(x) - x - 1 \leq x^2$ for $-1 \leq x \leq 1$ and that $\eta g_i(t) \in [-c_i, c_i]$, the above is upper bounded by $\eta^2 \sum_{i\in A} p_i(t) \frac{(g_i(t))^2}{c_i}$. We can also rewrite the second term in regret. In fact, if we set $\mathbf{p}(1) = \boldsymbol{\mu}$, then

$$\frac{1}{\eta} \cdot D_F(\mathbf{q}, \mathbf{p}(1)) = \frac{1}{\eta} \cdot \sum_{i\in A} c_i \left(q_i \ln\left(\frac{q_i}{\mu_i}\right) - q_i + \mu_i\right)$$

By summing the upper-bounds $\eta^2 \sum_{i\in A} p_i(t) \frac{(g_i(t))^2}{c_i}$ on each term of local norm in (13) for $t \in [T]$ and putting all the pieces together, we get the desired bound. ■

2.5. Regret analysis for non-negative rewards

Theorem 1 *There exists an algorithm for the full-information multi-scale online learning problem that takes as input any distribution $\boldsymbol{\pi}$ over A, the ranges c_i, $\forall\, i \in A$ and a parameter $0 < \epsilon \leq 1$, and satisfies:*

$$\forall i \in A: \quad \mathbb{E}\left[\text{REGRET}_i\right] \leq \epsilon \cdot G_i + O\left(\frac{1}{\epsilon} \log\left(\frac{1}{\epsilon\pi_i}\right) \cdot c_i\right) \tag{14}$$

Proof Suppose $i_{\min}$ is an action with the minimum c_i. Let $\boldsymbol{\mu} = (1-\eta)\cdot\mathbf{1}_{i_{\min}} + \eta\cdot\boldsymbol{\pi}$, and let $\mathbf{q} = (1-\eta)\cdot\mathbf{1}_i + \eta\cdot\boldsymbol{\pi}$ in Proposition 10. If $i \neq i_{\min}$, we get that (note that $\mu_j = q_j$ for any $j \neq i, i_{\min}$):

$$\begin{aligned} (1-\eta)\cdot G_i + \eta \cdot \sum_{j\in A} \pi_j \cdot G_j - \mathbb{E}\left[G_{\text{ALG}}\right] \leq \eta \cdot \mathbb{E}\left[G_{\text{ALG}}\right] &+ \frac{1}{\eta} \cdot c_i \cdot \left(q_i \ln\left(\frac{q_i}{\mu_i}\right) - q_i + \mu_i\right) \\ &+ \frac{1}{\eta} \cdot c_{i_{\min}} \cdot \left(q_{i_{\min}} \ln\left(\frac{q_{i_{\min}}}{\mu_{i_{\min}}}\right) - q_{i_{\min}} + \mu_{i_{\min}}\right) \end{aligned}$$

By $1 \geq q_i > \mu_i \geq \eta\pi_i$, the second term on the RHS is upper bounded as:

$$\frac{1}{\eta} \cdot c_i \cdot \left(q_i \ln\left(\frac{q_i}{\mu_i}\right) - q_i + \mu_i\right) \leq \frac{1}{\eta} \cdot c_i \cdot \ln\left(\frac{1}{\eta\pi_i}\right)$$

Similarly, by $1 \geq \mu_{i_{\min}} > q_{i_{\min}} \geq 0$, the third term on the RHS is upper bounded as

$$\frac{1}{\eta} \cdot c_{i_{\min}} \cdot \left(q_{i_{\min}} \ln\left(\frac{q_{i_{\min}}}{\mu_{i_{\min}}}\right) - q_{i_{\min}} + \mu_{i_{\min}}\right) \leq \frac{1}{\eta} \cdot c_{i_{\min}} \leq \frac{1}{\eta} \cdot c_i$$

Finally, note that $G_j \geq 0$ for all $j \in A$ in reward-only instances. So the LHS is lower bounded by

$$(1-\eta)\cdot G_i - \mathbb{E}\left[G_{\text{ALG}}\right] = (1-\eta)\cdot \text{REGRET}_i - \eta\cdot\mathbb{E}\left[G_{\text{ALG}}\right].$$

Putting all this together, we get that

$$\mathbb{E}\left[\text{REGRET}_i\right] \leq \frac{2\eta}{1-\eta} \cdot \mathbb{E}\left[G_{\text{ALG}}\right] + O\left(\frac{1}{\eta} \ln\left(\frac{1}{\eta\pi_i}\right) \cdot c_i\right) \leq 3\eta \cdot \mathbb{E}\left[G_{\text{ALG}}\right] + O\left(\frac{1}{\eta} \ln\left(\frac{1}{\eta\pi_i}\right) \cdot c_i\right)\ .$$

The theorem then follows by choosing $\eta = \frac{\epsilon}{3}$ and rearranging terms. ■

2.6. A canonical application: online single buyer auction

The setup. The simple auction design problem that we consider is as follows. There is a seller with infinite identical copies of an item. Buyers arrive over time. At each round, the seller picks a *price* and the arriving buyer reports her *value*. If the value is no less than the price, the trade happens; money goes to the seller and the copy of the item goes to the arriving buyer. The goal is to maximize the revenue of the seller.

Formally, we look at this problem as an instance of the full information multi-scale online learning framework; The action set is $A = [1, h]$. [7] The reward function is such that at round t the adversary (i.e. the arriving buyer) picks a value $v(t) \in [1, h]$ and for any price $p \in A$ picked by the seller (i.e. the algorithm), the reward is $g_p(t) := p \cdot \mathbf{1}(v(t) \geq p)$. This is a full information setting, because the value $v(t)$ is revealed to the algorithm after each round t.

The additive/multiplicative approximation. In order to obtain a $(1-\epsilon)$-approximation of the optimal revenue, i.e. the revenue of the best fixed price p^* in hindsight, it suffices to consider prices of the form $(1+\epsilon)^j$ for $0 \leq j \leq \lfloor \log_{1+\epsilon} h \rfloor = O(\frac{\log h}{\epsilon})$. As a result, we reduce the online single buyer auction problem to the multi-scale online learning with full information and finite actions. The action set has $k = O(\frac{\log h}{\epsilon})$ actions whose ranges form a geometric sequence $(1+\epsilon)^j$, $0 \leq j < k$.

Recall the definition of G_{MAX} in Section 2.1, and let p^* be the best fixed price in hindsight, which is the price that achieves G_{MAX}. We now show how to get a multiplicative cum additive approximation for this problem with G_{MAX} as the benchmark, à la Blum et al. (2004); Blum and Hartline (2005). The main improvement over these results is that the additive term scales with the best price rather than h.

Theorem 11 *There is an algorithm for the online single buyer auction problem that takes as input a parameter $\epsilon > 0$, and satsify $G_{\text{ALG}} \geq (1-\epsilon)G_{\text{MAX}} - O(E)$, where:*

$$E = \frac{p^* \log(\log h/\epsilon)}{\epsilon} .$$

Also, even if h is not known up front, there is an (slightly modified) algorithm that achieves a similar approximation guarantee for online single buyer auction with:

$$E = \frac{p^* \log(p^*/\epsilon)}{\epsilon} .$$

Proof
[Part 1: known h] Recall the above formulation of the problem as an online learning problem with full information. The proof then follows by Theorem 1, letting $\boldsymbol{\pi}$ to be the uniform distribution over the $k = O(\log h/\epsilon)$ actions, i.e., discretized prices.

[Part 2: unknown h] When h is not known up front, we consider a variant of our algorithm (Algorithm 2) that picks the next price in each round t from the set of *relevant prices* (denoted by $\mathcal{P}$), updates this set if necessary, and then updates the weights of prices in this set as in Algorithm 1. The main new idea here is to update the set of prices $\mathcal{P}$ so that it only includes prices that are at most the highest value we have seen so far (let the highest seen value be 1 at the beginning). Now, for the sake of analysis, consider a hypothetical algorithm (called $\texttt{ALG}^H$) that considers a countably infinite action space comprising all prices of the form $(1+\epsilon)^j$, for $j \geq 0$. We first show this hypothetical algorithm $\texttt{ALG}^H$ satisfies the required approximation guarantee in Theorem 11. We then show the expected revenue of Algorithm 2 is at least the expected revenue of $\texttt{ALG}^H$ (minus a constant that is negligible in our bound), and hence the final proof.

The proof of the regret bound of Theorem 1 works when we have countably many actions (although we cannot implement such algorithms directly). Now, consider simulating $\texttt{ALG}^H$ and let the prior distribution $\boldsymbol{\pi}$ be such that for any price $p = (1+\epsilon)^j$, $\pi_p = \epsilon(\epsilon+2)(1+\epsilon)^{-2(j+1)} = \frac{\epsilon(\epsilon+2)}{(1+\epsilon)^2} \cdot \frac{1}{p^2}$ (this choice will become more clear later in the proof; in short we need π_p to be proportional to $\frac{1}{p^2}$). The approximation guarantee in Theorem 11 then follows by Theorem 1. We now argue the followings:

- For any round t, unless the value in that round is a new highest value, Algorithm 2 gets weakly higher revenue than $\texttt{ALG}^H$. This is because the probability that Algorithm 2 plays any relevant price in $\mathcal{P}$ (that has a non-zero gain in this round) is weakly higher than that in $\texttt{ALG}^H$.

- For any price $p = (1+\epsilon)^j$, consider the first time a value at least p shows up. Algorithm 2 suffers a loss of at most $p \cdot \pi_p$ compared to $\texttt{ALG}^H$, due to $\texttt{ALG}^H$'s probability of playing p in that round, where π_p is the probability of playing p in the initial distribution. This is because the probability that $\texttt{ALG}^H$ plays p in this round is at most π_p as p has not got any positive gains before this round.

- Then, by choosing π_p to be inversely proportional to p^2, we can show that Algorithm 2 has an additive loss of $\sum_p \frac{\beta}{p} = \frac{\epsilon+2}{\epsilon+1} = O(1)$ compared to$\texttt{ALG}^H$, where $\beta = \left(\sum_p \frac{1}{p^2}\right)^{-1} = \frac{\epsilon(2+\epsilon)}{(1+\epsilon)^2}$ is the normalization constant of the initial distribution $\boldsymbol{\pi}$. This finishes the proof.

■

Algorithm 2 Online single buyer auction (for unknown h)

input learning rate $0 < \eta \leq 1$, price discretization parameter $0 < \epsilon \leq 1$.
initialize the set of relevant prices $\mathcal{P} = \{1\}$. Let $\alpha_1(1) = 1$.
for $t = 1, \ldots, T$ **do**
  Randomly pick a price in $\mathcal{P}$ drawn from $\boldsymbol{\alpha}(t)$, and observe $\mathbf{g}(t)$.
  Update $\mathcal{P}$ to be all the prices $(1+\epsilon)^j$ that are at most the highest value until time t.
  $\forall p \in \mathcal{P}: \quad w_p(t+1) \leftarrow \alpha_p(t) \cdot \exp(\eta \cdot \frac{g_p(t)}{p})$.
  Find λ^* (e.g., binary search) s.t. $\sum_{p \in \mathcal{P}} w_p(t+1) \cdot \exp(-\frac{\lambda^*}{p}) = 1$.
  $\forall p \in \mathcal{P}: \quad \alpha_p(t+1) \leftarrow w_p(t+1) \cdot \exp(-\frac{\lambda^*}{p})$.
end for

Bounds on the sample complexity of auctions for single buyer problem (Huang et al., 2015a) imply that the first bound in this theorem is tight up to log factors: the lower bound

7. Here, we allow an infinite action set. Later, we show how to discretize to get around this issue.

is $h\epsilon^{-1}$ in an instance where p^* is actually equal to h. Also, the best upper bound known is by Blum et al. (2004); Blum and Hartline (2005), which is

$$E = \frac{h\log(1/\epsilon)}{\epsilon} .$$

We conclude that Theorem 11 generalizes the known tight sample complexity upper-bound for the offline single buyer Bayesian revenue maximization to the online adversarial setting.

3. Multi-Scale Online Learning with Bandit Feedback

In this section, we look at the bandit feedback version of multi-scale online learning framework proposed in Section 2.1. Essentially, the only difference here is that after the algorithm picks an arm i_t at time t, it only observes the obtained reward, i.e. $g_{i_t}(t)$, and *does not* observe the entire reward function $\mathbf{g}(t)$.

Inspired by the online stochastic mirror descent algorithm (Bubeck, 2011) we introduce *Bandit-MSMW* algorithm. Our algorithm follows the standard bandit route of using unbiased estimators for the rewards in a full information strategy (in this case MSMW). We also mix the MSMW distribution with an extra uniform exploration, and use a tailored initial distribution to obtain the desired mutli-scale regret bounds.

3.1. Bandit multi-scale regret bounds

For the bandit version, we can get similar regret guarantees as in Section 2.2 for the full-information variant, but only for the *best* action. If we require the regret bound to hold for all actions, then we can only get a weaker bound, where the second term has ϵ^{-2} instead of ϵ^{-1}. The difference between the bounds for the bandit and the full information setting is essentially a factor of k, which is unavoidable.

Theorem 12 *There exists an algorithm for the online multi-scale problem with bandit feedback that takes as input the ranges c_i, $\forall\ i \in A$, and a parameter $0 < \epsilon \leq 1$, and satisfies,*

- *for $i^* = \arg\max_{i\in A} G_i$,*

$$\mathbb{E}\left[\text{REGRET}_{i^*}\right] \leq \epsilon \cdot G_{i^*} + O\left(\tfrac{1}{\epsilon} k \log\left(\tfrac{k}{\epsilon}\right) \cdot c_{i^*}\right). \tag{15}$$

- *for all $i \in A$,*

$$\mathbb{E}\left[\text{REGRET}_{i}\right] \leq \epsilon \cdot G_{i} + O\left(\tfrac{1}{\epsilon^2} k \log\left(\tfrac{k}{\epsilon}\right) \cdot c_{i}\right). \tag{16}$$

Also, one can compute the pure-additive versions of the bounds in Theorems 12 by setting $\epsilon = \sqrt{\frac{k\log(kT)}{T}}$ and $\epsilon = (\frac{k\log(kT)}{T})^{\frac{1}{3}}$ resepctively (Corollary 13), and compare with the pure-additive regret bound $O\big(c_{\max}\cdot\sqrt{Tk}\big)$ for the adversarial multi-armed bandit problem (Audibert and Bubeck, 2009; Auer et al., 1995).

Corollary 13 *There exist algorithms for the online multi-scale bandits problem that satisfies,*

- *For $i^* = \arg\max_{i\in A} G_i$,*

$$\mathbb{E}\left[\text{REGRET}_{i^*}\right] \leq O\left(c_{i^*}\cdot\sqrt{Tk\log(kT)}\right) \tag{17}$$

- *For all $i \in A$,*

$$\mathbb{E}\left[\text{REGRET}_{i}\right] \leq O\left(c_i \cdot T^{\frac{2}{3}}(k\log(kT))^{\frac{1}{3}}\right) \tag{18}$$

Here is a map of this section. In Section 3.2 we propose our bandit algorithm and prove its general regret guarantee for non-negative rewards. Then in Section 3.3 we show how to get a multi-scale style regret guarantee for the best arm c_{i^*}, and a weaker guarantee for all arms $\{c_i\}_{i\in A}$.

3.2. Bandit Multi-Scale Multiplicative Weight (Bandit-MSMW) algorithm

We present our Bandit algorithm (Algorithm 3) when the set of actions A is finite (with $|A| = k$). Let η be the learning rate and γ be the exploration probability. We show the following regret bound.

Algorithm 3 Bandit-MSMW

input exploration parameter $\gamma > 0$, learning rate $\eta > 0$.
initialize $\mathbf{p}(1) = (1-\gamma)\mathbf{1}_{i_{\min}} + \frac{\gamma}{k}\mathbf{1}$, where $i_{\min}$ is the arm with minimum range $c_{i_{\min}}$.
for $t = 1, \ldots, T$ **do**
  Let $\tilde{\mathbf{p}}(t) = (1-\gamma)\mathbf{p}(t) + \frac{\gamma}{k}\mathbf{1}$.
  Randomly pick an expert i_t drawn from $\tilde{\mathbf{p}}(t)$, and observe $g_{i_t}(t)$.
  Let $\tilde{\mathbf{g}}(t)$ be such that

$$\tilde{g}_i(t) = \begin{cases} \frac{g_i(t)}{\tilde{p}_i(t)} & \text{if } i = i_t; \\ 0 & \text{otherwise.} \end{cases}$$

  $\forall i \in A: \quad w_i(t+1) \leftarrow p_i(t)\cdot\exp(\frac{\eta}{c_i}\cdot\tilde{g}_i(t))$.
  Find λ^* (e.g., binary search) s.t. $\sum_{i\in A} w_i(t+1)\cdot\exp(-\frac{\lambda^*}{c_i}) = 1$.
  $\forall i \in A: \quad p_i(t+1) \leftarrow w_i(t+1)\cdot\exp(-\frac{\lambda^*}{c_i})$.
end for

Lemma 14 *For any exploration probability $0 < \gamma \leq \frac{1}{2}$ and any learning rate parameter $0 < \eta \leq \frac{\gamma}{k}$, the Bandit-MSMW algorithm achieves the following regret bound when the gains are non-negative :*

$$\forall i \in A : \mathbb{E}\left[\text{REGRET}_i\right] \leq O\left(\tfrac{1}{\eta}\log\left(\tfrac{k}{\gamma}\right)\cdot c_i + \eta\textstyle\sum_{j\in A} G_j + \gamma\cdot G_i\right)$$

Proof We further define:

$$\widetilde{G}_{\text{ALG}} \triangleq \textstyle\sum_{t\in[T]} g_{i_t}(t) = \sum_{t\in[T]} \tilde{\mathbf{p}}(t)\cdot\tilde{\mathbf{g}}(t) ,$$
$$\widetilde{G}_j \triangleq \textstyle\sum_{t\in[T]} \tilde{g}_j(t) .$$

In expectation over the randomness of the algorithm, we have:

1. $\mathbb{E}[G_{\text{ALG}}] = \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right]$; and

2. $G_j = \mathbb{E}\left[\widetilde{G}_j\right]$ for any $j \in A$.

Hence, to upper bound $\mathbb{E}[\text{REGRET}_i] = G_i - \mathbb{E}[G_{\text{ALG}}]$, it suffices to upper bound $\mathbb{E}\left[\widetilde{G}_i - \widetilde{G}_{\text{ALG}}\right]$. By the definition of the probability that the algorithm picks each arm, i.e., $\tilde{\mathbf{p}}(t)$, we have:

$$\mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] \geq (1-\gamma)\textstyle\sum_{t\in[T]} \mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t)\ .$$

Hence, we have for any initial distribution $\mathbf{q}$ over A:

$$\begin{aligned}\sum_{j\in A} q_j\cdot\mathbb{E}\left[\widetilde{G}_j\right] - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] \leq\ & \mathbb{E}\left[\textstyle\sum_{j\in A} q_j\cdot\widetilde{G}_j - \sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t)\right] + \tfrac{\gamma}{1-\gamma}\mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right]\\ \leq\ & \mathbb{E}\left[\textstyle\sum_{j\in A} q_j\cdot\widetilde{G}_j - \sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t)\right] + 2\gamma\mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right]\ . \quad (19)\end{aligned}$$

Next, we upper bound the 1st term on the RHS. Note that $\mathbf{p}(t)$'s are the probabilities of choosing experts by MSMW when the experts have rewards $\tilde{\mathbf{g}}(t)$'s. By Proposition 10, we have for any benchmark distribution $\mathbf{q}$ over S, the Bandit-MSMW algorithm satisfies:

$$\sum_{j\in A} q_j\cdot\widetilde{G}_j - \sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t) \leq \eta\sum_{t\in[T]}\sum_{j\in A}\frac{p_j(t)}{c_j}\cdot\left(\tilde{g}_j(t)\right)^2 + \frac{1}{\eta}\sum_{j\in A} c_j\left(q_j\ln\left(\frac{q_j}{p_j(1)}\right) - q_j + p_j(1)\right)\ . \quad (20)$$

For any $t\in[T]$ and any $j\in A$, by the definition of $\tilde{g}_j(t)$, it equals $\frac{g_j(t)}{\tilde{p}_j(t)}$ with probability $\tilde{p}_j(t)$, and equals 0 otherwise. Thus, if we fix the random coin flips in the first $t-1$ rounds and, thus, fix $\tilde{\mathbf{p}}(t)$, and take expectation over the randomness in round t, we have:

$$\mathbb{E}\left[\frac{p_j(t)}{c_j}\cdot\left(\tilde{g}_j(t)\right)^2\right] = \frac{p_j(t)}{c_j}\cdot\tilde{p}_j(t)\cdot\left(\frac{g_j(t)}{\tilde{p}_j(t)}\right)^2 = \frac{p_j(t)}{\tilde{p}_j(t)}\frac{(g_j(t))^2}{c_j}\ .$$

Further note that since $\tilde{p}_j(t)\geq(1-\gamma)p_j(t)$, and $g_j(t)\leq c_j$, the above is upper bounded by $\frac{1}{1-\gamma}g_j(t)\leq 2g_j(t)$. By putting all together with (20), we have for any $0<\eta\leq\frac{\gamma}{n}$:

$$\begin{aligned}\mathbb{E}\left[\sum_{j\in A} q_j\cdot\widetilde{G}_j - \sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t)\right] \leq\ & \eta\sum_{t\in[T]}\sum_{j\in A} 2g_j(t) + \frac{1}{\eta}\sum_{j\in A} c_j\left(q_j\ln\left(\frac{q_j}{p_j(1)}\right) - q_j + p_j(1)\right)\\ =\ & 2\eta\sum_{j\in A} G_j + \frac{1}{\eta}\sum_{j\in A} c_j\left(q_j\ln\left(\frac{q_j}{p_j(1)}\right) - q_j + p_j(1)\right)\end{aligned}$$

Combining with (19), we have:

$$\sum_{j\in A} q_j\cdot\mathbb{E}\left[\widetilde{G}_j\right] - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] \leq 2\eta\sum_{j\in A} G_j + \frac{1}{\eta}\sum_{j\in A} c_j\left(q_j\ln\left(\frac{q_j}{p_j(1)}\right) - q_j + p_j(1)\right) + 2\gamma\mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right]$$

Let $\mathbf{q} = (1-\gamma)\mathbf{1}_i + \frac{\gamma}{k}\mathbf{1}$. Recall that $\mathbf{p}(1) = (1-\gamma)\mathbf{1}_{i_{\min}} + \frac{\gamma}{k}\mathbf{1}$ (recall $i_{\min}$ is the arm with minimum range $c_{i_{\min}}$). Similar to the discussion for the expert problem in Section 2.5, the 2nd term on the RHS is upper bounded by $O\left(\frac{1}{\eta}\log\left(\frac{k}{\gamma}\right)\cdot c_i\right)$. Hence, we have:

$$\sum_{j\in A} q_j\cdot\mathbb{E}\left[\widetilde{G}_j\right] - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] \leq 2\eta\sum_{j\in A} G_j + O\left(\frac{1}{\eta}\log\left(\frac{k}{\gamma}\right)\cdot c_i\right) + 2\gamma\mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right]\ . \quad (21)$$

Further, the LHS is lower bounded as:

$$(1-\gamma)\mathbb{E}\left[\widetilde{G}_i\right] + \frac{\gamma}{k}\sum_{j\in A}\mathbb{E}\left[\widetilde{G}_j\right] - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] \geq (1-\gamma)\mathbb{E}\left[\widetilde{G}_i\right] - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right]\ .$$

The lemma then follows by putting this lower-bound of LHS of (21) back to the equation and rearranging terms. ■

3.3. Regret bounds for non-negative rewards - proof of Theorem 12

Letting $\gamma = \epsilon$ and $\eta = \frac{\gamma}{k} = \frac{\epsilon}{k}$ in Lemma 14, we get that the expected regret w.r.t. an action $i\in A$ is bounded by:

$$O\left(\epsilon\cdot G_i + \tfrac{\epsilon}{k}\textstyle\sum_{j\in A} G_j + c_i\cdot\frac{k}{\epsilon}\ln\left(\frac{k}{\epsilon}\right)\right)\ .$$

When $i = i^*$ (best arm), regret is bounded by $O\left(\epsilon\cdot G_{i^*} + c_i^*\cdot\frac{k}{\epsilon}\ln\left(\frac{k}{\epsilon}\right)\right)$, as desired.

For the regret w.r.t. an arbitrary action, note that $\mathbb{E}[G_{\text{ALG}}] \geq \frac{\gamma}{k}\sum_{j\in A} G_j$. Thus, the regret bound w.r.t. an action $i\in A$ in Lemma 14 is further upper bounded by:

$$O\left(\tfrac{1}{\eta}\log\left(\tfrac{k}{\gamma}\right)\cdot c_i + \left(\tfrac{\eta k}{\gamma} + \gamma\right)\cdot\mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right]\right)$$

The theorem then follows by letting $\gamma = \epsilon$ and $\eta = \frac{\gamma^2}{k} = \frac{\epsilon^2}{k}$.

4. More Applications of Multi-scale Learning for Auctions and Pricing

In this section, we consider applying the multi-scale online learning framework, developed in Section 2 and Section 3, to design several other online auctions and pricings be the single buyer auction (discussed in Section 2.6). Besides the single buyer auction, the problems that we consider are as follows.

- **Online posted pricing:** The same as the online single buyer auction of Section 2.6, but in the bandit setting. The algorithm only learns the indicator function $\mathbf{1}(v(t)\geq p_t)$ where p_t is the price it picks in round t.

- **Online multi buyer auction:** The action set is the set of all "Myerson-type" mechanisms for n buyers, for some $n\in\mathbb{N}$. (See Definition 15.) The adversary picks a valuation vector $\mathbf{v}(t)\in[1,h]^n$ and the reward of a mechanism M is its revenue when the valuation of the buyers is given by $\mathbf{v}(t)$; this is denoted by $\text{REV}_M(\mathbf{v}(t))$. The algorithm sees the full vector of valuations $\mathbf{v}(t)$.

4.1. Auctions and pricing as multi-scale online learning problems

We now show how to reduce the above problems to special cases of multi-scale online learning.

Online multi buyer auction In multi buyer auctions, we consider the set of all discretized Myerson-type auctions as the action space. We start by defining Myerson-type auctions:

Definition 15 (Myerson-type auctions) *A* Myerson-type auction *is defined by n non-decreasing virtual value mappings $\phi_1, \ldots, \phi_n : [1, h] \mapsto [-\infty, h]$. Given a value profile $v_1, \ldots, v_n$, the item is given to the bidder j with the largest non-negative virtual value $\phi_j(v_j)$. Then, bidder j pays the minimum value that would keep him as the the winner.*

Myerson (1981) shows that when the bidders' values are drawn from independent (but not necessarily identical) distributions, the revenue-optimal auction is a Myerson-type auction. Devanur et al. (2016, Lemma 5) observe that to obtain a $1-\epsilon$ approximation, it suffices to consider the set of discretized Myerson-type auctions that treat each bidder's value as if it is equal to the closest power of $1+\epsilon$ from below. As a result, it suffices to consider the set of discretized Myerson-type auctions, each of which is defined by the virtual values of $(1+\epsilon)^j$'s, i.e., by $O(n \log h/\epsilon)$ real numbers $\phi_\ell((1+\epsilon)^j)$, for $\ell \in [n]$, and $0 \le j \le \lfloor \log_{1+\epsilon} h \rfloor$. Furthermore, first Elkind (2007) and later on Devanur et al. (2016); Gonczarowski and Nisan (2017) note that a discretized Myerson-type auction is in fact completely characterized by the total ordering of $\phi_\ell((1+\epsilon)^j)$'s;[8] their actual values do not matter. Indeed, both the allocation rule and the payment rule are determined by the ordering of virtual values. As a result, our action space is a finite set with at most $O((n \log h/\epsilon)!)$ actions. The range of an action, i.e., a discretized Myerson-type auction, is the largest price ever charged by the auction, i.e., the largest value v of the form $(1+\epsilon)^j$ such that there exists $\ell \in [n]$, $\phi_\ell(v) > \phi_\ell((1+\epsilon)^{-1} v)$.

4.2. Multiplicative/additive approximations

Similar to Section 2.6, we show how to get a multiplicative cum additive approximations for these problems with G_{MAX} as the benchmark. Recall the definition of G_{MAX} in Section 2.1 and let p^* be the best fixed price on hindsight, which is the price that achieves G_{MAX}.

Theorem 16 *There are algorithms for the online posted pricing and the online multi buyer auction problems that take as input a parameter $\epsilon > 0$, and satsify $G_{\text{ALG}} \ge (1-\epsilon) G_{\text{MAX}} - O(E)$, where respectively (for the two problems mentioned above)*

$$E = \frac{p^* \log h \log(\log h/\epsilon)}{\epsilon^2}, \quad \text{and} \quad \frac{hn \log h \log(n \log h/\epsilon)}{\epsilon^2}.$$

Even if h is not known up front, we can still get the similar approximation guarantee for the online multi buyer auction with:

$$E = \frac{hn \log h \log(n \log h/\epsilon)}{\epsilon^2}.$$

8. Cai et al. (2012) also generalizes this observation to multi-dimensional types.

We conjecture that our bound for the online posted pricing problem is tight up to logarithmic factors, and leave resolving this as an open problem. The second bound is not comparable to the best sample complexity for the multi buyer auction problem by Roughgarden and Schrijvers (2016); it is better than theirs for large ϵ (when $1/\epsilon \le o(nh)$), and is worse for smaller ϵ (when $1/\epsilon \ge \omega(nh)$). Also, compare the first bound to the corresponding upper bound for the pricing problem by Blum and Hartline (2005), which is

$$\min\left\{\frac{h \log h \log\log h}{\epsilon^2}, \frac{h \log\log h}{\epsilon^3}\right\}.$$

Essentially, the main improvement over this result is that the additive term scales with the best price rather than h.

Proof [of Theorem 16]

Online posted pricing. Recall the formulation of the problem as an online learning problem with bandit feedback in Section 4.1. This part then follows by Theorem 12 with $k = O(\log h/\epsilon)$ actions.

Online multiple buyer auction. Recall the formulation of the problem as an online learning problem with full information in Section 4.1. The proof then follows by Theorem 1, where we let $\boldsymbol{\pi}$ be the uniform distribution over the $k = O((n \log h/\epsilon)!)$ actions, i.e., Myerson-type auctions.

When h is not known up front, similar to the proof of Theorem 11, we consider a hypothetical algorithm with countably infinite action space A as follows. For any $p = (1+\epsilon)^j$, $j \ge 0$, let the $k_p = O((n \log p/\epsilon)!)$ Myerson-type auctions for values in $[1, p]$ be in A; we assume these auctions treat any values greater than p as if they were p. Further, we choose the prior distribution $\boldsymbol{\pi}$ such that the probability mass of each auction for range $[1, p]$ is equal to $\frac{\epsilon(\epsilon+2)}{(1+\epsilon)^2} \cdot \frac{1}{p^2} \cdot \frac{1}{k_p}$. The approximation guarantee then follows by Theorem 1. To implement this algorithm, we use the same trick as in the proof Theorem 11 by running a modified algorithm that only considers auctions for all ranges $[1, p]$ where p is no larger than the highest value seen so far among all the buyers (i.e. a multi-buyer auction version of Algorithm 2). The rest of the proof that shows the revenue loss of this algorithm compared to the hypothetical algorithm is negligible is similar to the proof of Theorem 11 (and hence omitted for brevity).

■

4.3. Competing with δ-guarded benchmarks

For the single buyer auction/pricing problem, we define a δ-guarded benchmark, for any $\delta \in [0, 1]$. This benchmark is restricted to only those prices that sell the item in at least a δ fraction of the rounds.

$$G_{\text{MAX}}(\delta) := \max\left\{\textstyle\sum_{t=1}^T g_p(t) : p \in A, \sum_{t=1}^T \mathbf{1}(v_t \ge p) \ge \delta T\right\}.$$

As observed in Footnote 3, one can replace δ with $1/h$ and get the corresponding guarantees for G_{MAX} rather than $G_{\text{MAX}}(\delta)$. However, the main point of these results is to show a graceful improvement of the bounds as δ is chosen to be larger.

Multiple buyers: For the multi buyer auction problem, we define the δ-guarded benchmark as follows. For any sequence of value vectors $\mathbf{v}(1), \mathbf{v}(2), \ldots, \mathbf{v}(T)$, let $\bar{V}$ denote the largest value such that there are at least δT distinct $t \in [T]$ with $\max_{i\in[n]} v_i(t) \geq \bar{V}$. Define the δ-guarded benchmark to be

$$G_{\text{MAX}}(\delta) = \max_M \sum_{t=1}^{T} Rev_M \left(\min(\bar{V}\vec{\mathbf{1}}, \mathbf{v}(t))\right),$$

where the "min" is taken coordinate-wise, and the "max" is over all Myerson-type mechanisms. In other words, here is how we can describe the δ-guarded benchmark: for each Myerson-type auction M, after identifying the value cap $\bar{V}$, we cut all the values that are above $\bar{V}$ by this quantity, and then run M. The benchmark is then the revenue of the best Myerson-type auction under these modified values.

We focus on purely multiplicative approximation factors when competing with $G_{\text{MAX}}(\delta)$. In particular, for any given $\epsilon > 0$, we are interested in a $1-\epsilon$ approximation. We state our results in terms of the *convergence rate*. We say that $T(\epsilon, \delta)$ is the convergence rate of an algorithm if for all time horizon $T \geq T(\epsilon, \delta)$, we are guaranteed that $G_{\text{ALG}} \geq (1-\epsilon)G_{\text{MAX}}(\delta)$. Our main results are as follows.

Theorem 17 *There are algorithms for the online single buyer auction, online posted pricing, and the online multi buyer auction problems with convergence rates respectively of*

$$O\left(\frac{\log(\log h/\epsilon)}{\epsilon^2\delta}\right), \quad O\left(\frac{\log h}{\epsilon^4\delta}\right), \quad \text{and } O\left(\frac{n\log(1/\epsilon\delta)\log(n\log(1/\epsilon\delta)/\epsilon)}{\epsilon^3\delta} + \frac{\log(\log h/\epsilon)}{\epsilon^2\delta}\right).$$

Even if h is not known upfront, we can still get the following similar convergence rates for online single buyer auction and online multi buyer auction respectively:

$$O\left(\frac{\log(p^*/\epsilon)}{\epsilon^2\delta}\right), \quad \text{and} \quad O\left(\frac{n\log(1/\epsilon\delta)\log(n\log(1/\epsilon\delta)/\epsilon)}{\epsilon^3\delta} + \frac{\log(h/\epsilon)}{\epsilon^2\delta}\right).$$

Once again, we compare to the sample compexity bounds: our first is within a $\log\log h$ factor of the best sample complexity upper bound in Huang et al. (2015b). The lower bound for the online single buyer auction is $\Omega(\delta^{-1}\epsilon^{-2})$, which is also the best lower bound known for the pricing and the multi-buyer problem.[9]

For the online posted pricing problem, we conjecture that the right dependence on ϵ should be ϵ^{-3}. No sample complexity bounds for the multi-buyer problem were known before; in fact we introduce the definition of a δ-guarded benchmark for this problem.

Proof [of Theorem 17] *Online single buyer auction.* By Theorem 1, letting $\boldsymbol{\pi}$ be the uniform distribution over the $k = O(\log h/\epsilon)$ actions, i.e., discretized prices, we have that for any price p (recall that $c_p = p$):

$$G_{\text{ALG}} \geq (1-\epsilon)\cdot G_p - O\left(\tfrac{\log(\log h/\epsilon)}{\epsilon}\cdot p\right).$$

For the δ-guarded optimal price p^* (i.e., subject to selling in at least δT rounds), we have $G_{p^*} \geq \delta T \cdot p^*$. Therefore, when $T \geq O\left(\log(\log h/\epsilon)/\epsilon^2\delta\right)$, the additive term of the above approximation guarantee is at most $\epsilon \cdot G_{p^*}$. So the theorem holds.

9. Cole and Roughgarden (2014) show that at least a linear dependence on n is necessary when the values are drawn from a regular distribution, but as is, their lower bound needs unbounded valuations. The lower bound probably holds for "large enough h" but it is not clear if it holds for all h.

The treatment for the case when h is not known up front is essentially the same as in Theorem 16 and Theorem 11. As a hypothetical algorithm useful for analysis, we consider an algorithm (similar to Algorithm 1) with a countably infinite action space comprising all prices of the form $(1+\epsilon)^j$, for $j \geq 0$. Then, let the prior distribution $\boldsymbol{\pi}$ be such that for any price $p = (1+\epsilon)^j$, $\pi_p = \epsilon(\epsilon+2)(1+\epsilon)^{-2(j+1)} = \frac{\epsilon(eps+2))}{(1+\epsilon)^2}\cdot\frac{1}{p^2}$. The rest of the proof and how to implement is the same as in the proof of Theorem 11 (i.e. Algorithm 2).

Online posted pricing. Recall the above formulation of the problem as an online learning problem with bandit feedback. By Theorem 12 with $k = O(\log h/\epsilon)$ actions, we have that for any price p:

$$G_{\text{ALG}} \geq (1-\epsilon)\cdot G_p - O\left(\tfrac{\log h\log(\log h/\epsilon)}{\epsilon^3}\cdot p\right).$$

Again, for the δ-guarded optimal price p^* (i.e., subject to selling in at least δT rounds), we have $G_{p^*} \geq \delta T \cdot p^*$. Therefore, when $T \geq O\left(\log h\log\left(\log h/\epsilon\right)/\epsilon^4\delta\right)$, the additive term of the above approximation guarantee is at most $\epsilon \cdot G_{p^*}$. So the theorem holds.

Online multi buyer auction. Suppose i^* is the δ-guarded best Myerson-type auction. Recall that $\bar{V}$ is the largest value such that there are at least δT distinct $v(t)$'s with $\max_{\ell\in[n]} v_\ell(t) \geq \bar{V}$. So we may assume without loss of generality that i^* does not distinguish values greater than $\bar{V}$. Hence:

$$c_{i^*} \leq \bar{V}. \tag{22}$$

Further, note that running a second-price auction with anonymous reserve $\bar{V}$ is a Myerson-type auction (e.g., mapping values less than $\bar{V}$ to virtual value $-\infty$ and values greater than or equal to $\bar{V}$ to virtual value $\bar{V}$), and it gets revenue at least $\delta T\cdot\bar{V}$. So we have that:

$$G_{p^*} \geq \delta T\cdot\bar{V}. \tag{23}$$

Finally, the above implies that to obtain a $1-\epsilon$ approximation, it suffices to consider prices that are at least $\epsilon\delta\bar{V}$. Hence, it suffices to consider Myerson-type auctions that, for a given $\bar{V}$, do not distinguish among values greater than $\bar{V}$, and do not distinguish among values smaller than $\epsilon\delta\bar{V}$. There are $O(\log h/\epsilon)$ different values of $\bar{V}$. Further, given $\bar{V}$, there are only $O(\log(1/\epsilon\delta)/\epsilon)$ distinct values to be considered and, thus, there are at most $O((n\log(1/\epsilon\delta)/\epsilon)!)$ distinct Myerson-type auctions of this kind. Hence, the total number of distinct Myerson-type actions that we need to consider is at most:

$$k = O\left(\frac{\log h}{\epsilon}\cdot\left(\frac{n\log(1/\epsilon\delta)}{\epsilon}\right)!\right).$$

Letting $\boldsymbol{\pi}$ be the uniform distribution over the k actions in Theorem 1, we have that (recall Eqn. (22)):

$$G_{\text{ALG}} \geq (1-\epsilon)\cdot G_{i^*} - O\left(\frac{n\log(1/\epsilon\delta)\log(n\log(1/\epsilon\delta)/\epsilon)}{\epsilon^2} + \frac{\log(\log h/\epsilon)}{\epsilon}\right)\cdot\bar{V}.$$

When $T \geq O\left(\frac{n\log(1/\epsilon\delta)\log(n\log(1/\epsilon\delta)/\epsilon)}{\epsilon^3\delta} + \frac{\log(\log h/\epsilon)}{\epsilon^2\delta}\right)$, the additive term of the above approximation guarantee is at most $\epsilon\cdot G_{i^*}$ due to Eqn. (23). So the theorem holds.

Again, the treatment for the case when h is not known up front is similar to that in Theorem 16. When h is not known up front, we consider a hypothetical algorithm with a countably infinite action space A as follows. For any $\bar{V} = (1+\epsilon)^j$, $j \geq 0$, let the $k' = O((n \log(1/\epsilon\delta)/\epsilon)!)$ Myerson-type auctions that do not distinguish among values greater than $\bar{V}$, and do not distinguish among values smaller than $\epsilon\delta\bar{V}$ be in A. Further, we choose the prior distribution $\boldsymbol{\pi}$ such that the probability mass of each Myerson-type auction for a given $\bar{V}$ is equal to $\frac{\epsilon}{1+\epsilon} \cdot \frac{1}{\bar{V}} \cdot \frac{1}{k'}$. The approximation guarantee then follows by Theorem 1 and essentially the same argument as the known h case. Implementation is similar to the proof of Theorem 16 and Theorem 11 (i.e. a multi-buyer auction version of Algorithm 2). The rest of the proof that shows the revenue loss of this algorithm compared to the hypothetical algorithm is negligible is similar to the proof of Theorem 16 (and hence omitted for brevity). ■

Remark Devanur et al. (2016) show that when the values are drawn from independent regular distributions, the ϵ-guarded optimal price is a $1-\epsilon$ approximation of the unguarded optimal price. So our convergence rate for the online multi buyer auction problem in Theorem 1 implies a $\tilde{O}(n\epsilon^{-4})$ sample complexity modulo a mild $\log\log h$ dependency on the range, almost matching the best known sample complexity upper bound for regular distributions.

5. Multi-scale Online Learning with Symmetric Range

In this section, we consider multi-scale online learning when the rewards are in a symmetric range, i.e. for all $i \in A$ and $t \in [T]$, $g_i(t) \in [-c_i, c_i]$. The standard analysis for the experts and the bandit problems holds even if the range of $g_i(t)$ is $[-c_i, c_i]$, instead of $[0, c_i]$. In contrast, there are subtle differences on the best achievable multi-scale regret bounds between the non-negative and the symmetric range, which we explore in this section. We look at both the full information and bandit setting, and prove action-specific regret upper bounds. We then prove a tight lower-bound in Section 5.3 for the full information case, and an almost tight lower-bound in Section 5.5 for the bandit setting.

5.1. Multi-scale regret bounds for symmetric ranges

We first show the following upper bound for the full information setting when the range is symmetric. This bound follows the same style of action-specific regret bounds as in Theorem 1. More detailed discussion on how the choice of initial distribution $\boldsymbol{\pi}$ affects the bound is deferred to the appendix, Section A.1 (recall that the initial distribution $\boldsymbol{\pi}$ is the distribution over actions that is used in the first round of Algorithm 1).

Theorem 18 *There exists an algorithm for the multi-scale experts problem with symmetric range that takes as input any distribution $\boldsymbol{\pi}$ over A, the ranges c_i, $\forall\, i \in A$, and a parameter $0 < \epsilon \leq 1$, and satisfies:*

$$\forall i \in A: \quad \mathbb{E}\left[\text{REGRET}_i\right] \leq \epsilon \cdot \mathbb{E}\left[\sum_{t\in[T]} \left|g_t(i)\right|\right] + O\left(\frac{1}{\epsilon}\log\left(\frac{1}{\pi_i}\cdot\frac{c_i}{c_{\min}}\right)\cdot c_i\right). \tag{24}$$

Similar to Section 2.1, we can compute the pure-additive version of the bound in Theorem 18 by setting $\epsilon = \sqrt{\frac{\log(k\cdot\frac{c_{\max}}{c_{\min}})}{T}}$, as in Corollary 2.

Corollary 19 *There exists an algorithm for the online multi-scale experts problem with symmetric range that takes as input the ranges c_i, $\forall\, i \in A$, and satisfies:*

$$\forall i \in A: \quad \mathbb{E}\left[\text{REGRET}_i\right] \leq O\left(c_i \cdot \sqrt{T\log(k\cdot\tfrac{c_{\max}}{c_{\min}})}\right) \tag{25}$$

If we compare the above regret bound with the standard $O(c_{\max}\sqrt{T\log k})$ regret bound for the experts problem, we see that we replace the dependency on $c_{\max}$ in the standard bound with $c_i\sqrt{\log(\frac{c_{\max}}{c_{\min}})}$. It is natural to ask whether we could get rid of the dependence on $\log(c_i/c_{\min})$ and show a regret bound of $O(c_i\sqrt{T\log k})$, like we did for non-negative rewards. However, the next theorem shows that this dependence on $\log(c_i/c_{\min})$ in the above bound is necessary, in a weak sense: where the constant in the $O(\cdot)$ is universal and does not depend on the ranges c_i. This is because the lower bound only holds for "small" values of the horizon T, which nonetheless grows with the $\{c_i\}$s.[10]

Theorem 20 *There exists an action set of size k, and ranges $c_i, \forall i \in [k]$, and time horizon T, such that for all algorithms for the online multi-scale experts problem with symmetric range, there is a sequence of T gain vectors such that*

$$\exists i \in A: \quad \mathbb{E}\left[\text{REGRET}_i\right] > \tfrac{c_i}{4} \cdot \sqrt{T\log(k\cdot\tfrac{c_{\max}}{c_{\min}})}$$

We then show the following upper bound for the bandit setting when the range is symmetric. This bound also follows the same style of action-specific regret bounds as in Theorem 12.

Theorem 21 *There exists an algorithm for the multi-scale bandits problem with symmetric range that takes as input the ranges c_i, $\forall\, i \in A$, and a parameter $0 < \epsilon \leq 1/2$, and satisfies:*

$$\forall i \in A: \quad \mathbb{E}\left[\text{REGRET}_i\right] \leq O\big(\epsilon T + \tfrac{k}{\epsilon}\tfrac{c_{\max}}{c_{\min}}\log\left(\tfrac{k}{\epsilon}\tfrac{c_{\max}}{c_{\min}}\right)\big)\cdot c_i. \tag{26}$$

Also, similar to Section 2.1, we can compute the pure-additive version of the bound in Theorem 21 by setting $\epsilon = \sqrt{\frac{k\frac{c_{\max}}{c_{\min}}\log(kT\cdot\frac{c_{\max}}{c_{\min}})}{T}}$, as in Corollary 2. This bound is comparable to the standard regret bound of $O(c_{\max}\sqrt{kT\log k})$ (Auer et al., 1995) for the adversarial multi-armed bandits problem.

Corollary 22 *There exists an algorithm for the online multi-scale bandits problem with symmetric range that satisfies:*

$$\forall i \in A: \quad \mathbb{E}\left[\text{REGRET}_i\right] \leq O\left(c_i \cdot \sqrt{Tk\cdot\tfrac{c_{\max}}{c_{\min}}\log(kT\cdot\tfrac{c_{\max}}{c_{\min}})}\right). \tag{27}$$

Once again, for the bandit problem, the following theorem shows that this bound cannot be improved beyond logarithmic factors (to get a guarantee like that of Theorem 12, for instance).

Theorem 23 *There exists an action set of size k, and ranges $c_i, \forall i \in [k]$, such that for all algorithms for the online multi-scale bandit problem with symmetric range, for all sufficiently large time horizon T, there is a sequence of T gain vectors such that*

$$\exists i \in A: \quad \mathbb{E}\left[\text{REGRET}_i\right] > \frac{c_i}{8\sqrt{2}} \cdot \sqrt{Tk\cdot\frac{c_{\max}}{c_{\min}}}.$$

10. For this reason we chose not to include this bound in Table 1.

5.2. Upper bound for experts with symmetric range - Proof of Theorem 18

Recall the proof of Proposition 10. The proof only requires $g_i(t) \in [-c_i, c_i]$ for all $i \in A, t \in [T]$. Choosing q to be $\mathbf{1}_i$, a vector with a 1-entry in i^{th} coordinate and 0-entries elsewhere for an action $i \in A$, and noting that

$$\sum_{t\in[T]} \sum_{i\in A} p_i(t) \frac{(g_i(t))^2}{c_i} \leq \sum_{t\in[T]} \sum_{i\in A} p_i(t) \cdot \left|g_i(t)\right| ,$$

we get the following regret bound as a corollary of Proposition 10.

Corollary 24 *For any initial distribution μ over A, and any learning rate parameter $0 < \eta \leq 1$, the MSMW algorithm achieves the following regret bound:*

$$\forall i \in A: \quad \mathbb{E}\left[\text{REGRET}_i\right] \leq \eta \cdot \mathbb{E}\left[\sum_{t\in[T]} \left|g_i(t)\right|\right] + \frac{1}{\eta} c_i \cdot \log\left(\frac{1}{\mu_i}\right) + \frac{1}{\eta} \sum_{j\in A} \mu_j c_j \tag{28}$$

Now, we can prove the multi-scale regret upper-bound in Theorem 18 using Corollary 24.

Proof [of Theorem 18] The proof follows by choosing an appropriate initial distribution μ in Corollary 24. By Corollary 24, we have:

$$\mathbb{E}\left[\text{REGRET}_i\right] \leq \eta \cdot \mathbb{E}\left[\sum_{t\in[T]} \left|g_i(t)\right|\right] + \frac{1}{\eta} c_i \cdot \log(\frac{1}{\mu_i}) + \frac{1}{\eta} \sum_{j\in A} \mu_j c_j$$

Let $i_{\min}$ be an action with the minimum range $c_{i_{\min}} = c_{\min}$. Consider an initial distribution $\mu_j = \pi_j \frac{c_{\min}}{c_j}$ for all $j \neq i_{\min}$, and $\mu_{i_{\min}} = 1 - \sum_{j\neq i_{\min}} \mu_j$, i.e., putting all remaining probability mass on action $i_{\min}$. Then, the third term on the RHS is upper bounded by:

$$\sum_{j\in A} \mu_j c_j = \sum_{j\neq i_{\min}} \mu_j c_j + \mu_{i_{\min}} c_{i_{\min}} = \sum_{j\neq i_{\min}} \pi_j c_{\min} + \mu_{i_{\min}} c_{\min} \leq 2c_{\min} \leq 2c_i \ .$$

For $i \neq i_{\min}$, by the definition of μ_i, we have:

$$\begin{aligned}\mathbb{E}\left[\text{REGRET}_i\right] &\leq \eta \cdot \mathbb{E}\left[\sum_{t\in[T]} \left|g_i(t)\right|\right] + \frac{1}{\eta} c_i \cdot \log(\frac{1}{\pi_i} \cdot \frac{c_i}{c_{\min}}) + \frac{1}{\eta} \cdot 2c_{\min} \\ &= \eta \cdot \mathbb{E}\left[\sum_{t\in[T]} \left|g_i(t)\right|\right] + O\left(\frac{1}{\eta} \log\left(\frac{1}{\pi_i} \cdot \frac{c_i}{c_{\min}}\right) \cdot c_i\right) \ .\end{aligned}$$

So the theorem follows by choosing $\eta = \epsilon$. For $i = i_{\min}$, note that $\mu_j \leq \pi_j$ for all $j \neq i_{\min}$ and, thus, $\mu_{i_{\min}} = 1 - \sum_{j\neq i_{\min}} \mu_j \geq 1 - \sum_{j\neq i_{\min}} \pi_j = \pi_{i_{\min}} = \pi_{i_{\min}} \frac{c_{\min}}{c_{i_{\min}}}$. The theorem then holds following the same calculation as in the $j \neq i_{\min}$ case. ■

5.3. Lower bound for experts with symmetric range - proof of Theorem 20

We first show that for any online learning algorithm, and any sufficiently large $h > 1$, there is an instance that has two experts with $c_1 = 1$ and $c_2 = h$ with $T = \Theta(\log h)$ rounds, such that either

$$\mathbb{E}\left[\text{REGRET}_1\right] > \tfrac{1}{2}T + \sqrt{h} \ , \qquad \text{or} \qquad \mathbb{E}\left[\text{REGRET}_2\right] > \tfrac{1}{2}Th + \tfrac{1}{5}h\log_2 h \ .$$

We will construct this instance with $T = \frac{1}{2}\log_2 h - 1$ rounds adaptively that always has gain 0 for action 1 and gain either h or $-h$ for action 2. The proof of the theorem then

follows as $c_{\min} = 1$, $c_{\max} = h$, $T = \frac{1}{2}\log_2 h - 1$, and $k = 2$ in this instance. Let q_t denote the probability that the algorithm picks action 2 in round t after having the same rewards 1 and h for the two actions respectively in the first $t-1$ rounds. We will first show that (1) if the algorithm has small regret with respect to action 1, then q_t must be upper bounded since the adversary may let action 2 have cost $-h$ in any round t in which q_t is too large. Then, we will show that (2) since q_t is upper bounded for any $1 \leq t \leq T$, the algorithm must have large regret with respect to action 2.

We proceed with the upper bounding q_t's. Concretely, we will show the following lemma.

Lemma 25 *Suppose $\mathbb{E}\left[\text{REGRET}_1\right] \leq \frac{1}{2}T + \sqrt{h}$. Then, for any $1 \leq t \leq T$, we have $q_t \leq \frac{2^t}{\sqrt{h}}$.*

Proof We will prove by induction on t. Consider the base case $t = 1$. Suppose for contradiction that $q_1 > \frac{2}{\sqrt{h}}$. Then, consider an instance in which action 2 always has gain. In this case, the expected gain of the algorithm (even if it always correctly picks action 1 in the remaining instance) is at most $q_1 \cdot (-h) < -2\sqrt{h}$. This is a contradiction to the assumption that $\mathbb{E}\left[\text{REGRET}_1\right] \leq \frac{1}{2}T + \sqrt{h} < 2\sqrt{h}$.

Next, suppose the lemma holds for all rounds prior to round t. Then, the expected gain of algorithm in the first $t-1$ rounds if arm 2 has gain H is

$$\sum_{\ell=1}^{t-1} q_\ell \cdot h \leq \sum_{\ell=1}^{t-1} 2^\ell \sqrt{h} = \left(2^t - 2\right)\sqrt{h} \ .$$

Suppose for contradiction that $q_t > \frac{2^t}{\sqrt{h}}$. Then, consider an instance in which action 2 has gain H in the first $t-1$ rounds and $-H$ afterwards. In this case, the expected gain of the algorithm (even if it always correctly picks action 1 after round t) is at most

$$\left(2^t - 2\right)\sqrt{h} + q_t(-h) < \left(2^t - 2\right)\sqrt{h} + 2^t\sqrt{h} < -2\sqrt{h} \ .$$

This is a contradiction to the assumption that $\mathbb{E}\left[\text{REGRET}_1\right] \leq \frac{1}{2}T + \sqrt{h} < 2\sqrt{h}$. ■

Consider an instance in which action 2 always has gain H. Suppose that $\mathbb{E}\left[\text{REGRET}_1\right] \leq \frac{1}{2}T + \sqrt{h}$. As an immediate implication of the above lemma, the algorithm is that the expected gain of the algorithm is upper bounded by:

$$\sum_{t=1}^{T} q_t h \leq \sum_{t=1}^{T} 2^t\sqrt{h} < 2^{T+1}\sqrt{h} = h \ .$$

Note that in this instance $\mathbb{E}\left[G_2\right] = T \cdot h$. Thus, the regret w.r.t. action 2 is at least $(T-1)h$, which is greater than $\frac{1}{2} \cdot \mathbb{E}\left[G_2\right] + \frac{1}{5}h\log_2 h$ for sufficiently large h.

5.4. Upper bound for bandits with symmetric range - Proof of Theorem 21

We start by presenting the following regret bound, whose proof is an alteration of that for Lemma 14 under symmetric range. Next, we prove Theorem 21.

Lemma 26 *For any exploration rate $0 < \gamma \leq \min\{\frac{1}{2}, \frac{c_{\min}}{c_{\max}}\}$ and any learning rate $0 < \eta \leq \frac{\gamma}{k}$, the Bandit-MSMW algorithm (Algorithm 3) achieves the following regret bound:*

$$\forall i \in A : \mathbb{E}\left[\text{REGRET}_i\right] \leq O\left(\frac{1}{\eta}\log\left(\frac{k}{\gamma}\right)\cdot c_i + \gamma T \cdot c_{\max}\right)$$

Proof We further define:

$$\widetilde{G}_{\text{ALG}} \triangleq \sum_{t\in[T]} g_{i_t}(t) = \sum_{t\in[T]} \tilde{\mathbf{p}}(t)\cdot\tilde{\mathbf{g}}(t)\ ,$$
$$\widetilde{G}_j \triangleq \sum_{t\in[T]} \tilde{g}_j(t)\ .$$

In expectation over the randomness of the algorithm, we have:

1. $\mathbb{E}\left[G_{\text{ALG}}\right] = \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right]$; and
2. $G_j = \mathbb{E}\left[\widetilde{G}_j\right]$ for any $j \in A$.

Hence, to upper bound $\mathbb{E}\left[\text{REGRET}_i\right] = G_i - \mathbb{E}\left[G_{\text{ALG}}\right]$, it suffices to upper bound $\mathbb{E}\left[\widetilde{G}_i - \widetilde{G}_{\text{ALG}}\right]$.

By the definition of the probability that the algorithm picks each arm, i.e., $\tilde{\mathbf{p}}(t)$, and that reward of each round is at least $-c_{\max}$, we have that:

$$\mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] \geq (1-\gamma)\sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t) - \gamma T c_{\max}\ .$$

Hence, for any benchmark distribution $\mathbf{q}$ over A, we have that:

$$\begin{aligned}\textstyle\sum_{j\in A} q_j\cdot\mathbb{E}\left[\widetilde{G}_j\right] - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] &\leq \mathbb{E}\left[\textstyle\sum_{j\in A} q_j\cdot\widetilde{G}_j - \sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t)\right] + \tfrac{\gamma}{1-\gamma}\mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] + \tfrac{\gamma}{1-\gamma}Tc_{\max}\\ &\leq \mathbb{E}\left[\textstyle\sum_{j\in A} q_j\cdot\widetilde{G}_j - \sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t)\right] + 2\gamma\mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] + 2\gamma T c_{\max}\\ &\leq \mathbb{E}\left[\textstyle\sum_{j\in A} q_j\cdot\widetilde{G}_j - \sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t)\right] + 4\gamma T c_{\max}\ . \qquad (29)\end{aligned}$$

where the 2nd inequality is due to $\gamma \leq \frac{1}{2}$, and the 3rd inequality follows by that $c_{\max}$ is the largest possible reward per round.

Next, we upper bound the 1st term on the RHS of (29). Note that $\mathbf{p}(t)$'s are the probability of choosing experts by MSMW when the experts have rewards $\tilde{\mathbf{g}}(t)$'s. By Proposition 10, we have that for any benchmark distribution $\mathbf{q}$ over S, the Bandit-MSMW algorithm satisfies that:

$$\sum_{j\in A} q_j\cdot\widetilde{G}_j - \sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t) \leq \eta\sum_{t\in[T]}\sum_{j\in A}\frac{p_j(t)}{c_j}\cdot\left(\tilde{g}_j(t)\right)^2 + \frac{1}{\eta}\sum_{j\in A} c_j\left(q_j\ln\left(\frac{q_j}{p_j(1)}\right) - q_j + p_j(1)\right)\ . \qquad (30)$$

For any $t \in [T]$ and any $j \in A$, by the definition of $\tilde{g}_j(t)$, it equals $\frac{g_j(t)}{\tilde{p}_j(t)}$ with probability $\tilde{p}_j(t)$, and equals 0 otherwise. Thus, if we fix the random coin flips in the first $t-1$ rounds and, thus, fix $\tilde{\mathbf{p}}(t)$, and take expectation over the randomness in round t, we have that:

$$\mathbb{E}\left[\frac{p_j(t)}{c_j}\cdot\left(\tilde{g}_j(t)\right)^2\right] = \frac{p_j(t)}{c_j}\cdot\tilde{p}_j(t)\cdot\left(\frac{g_j(t)}{\tilde{p}_j(t)}\right)^2 = \frac{p_j(t)}{\tilde{p}_j(t)}\frac{(g_j(t))^2}{c_j}\ .$$

Further note that $\tilde{p}_j(t) \geq (1-\gamma)p_j(t)$, and $|g_j(t)| \leq c_j$, the above is upper bounded by $\frac{1}{1-\gamma}|g_j(t)| \leq 2|g_j(t)| \leq 2c_{\max}$. Putting together with (30), we have that for any $0 < \eta \leq \frac{\gamma}{n}$:

$$\begin{aligned}\mathbb{E}\left[\sum_{j\in A} q_j\cdot\widetilde{G}_j - \sum_{t\in[T]}\mathbf{p}(t)\cdot\tilde{\mathbf{g}}(t)\right] &\leq \eta\sum_{t\in[T]}\sum_{j\in A} 2c_{\max} + \frac{1}{\eta}\sum_{j\in A} c_j\left(q_j\ln\left(\frac{q_j}{p_j(1)}\right) - q_j + p_j(1)\right)\\ &= 2\eta T k c_{\max} + \frac{1}{\eta}\sum_{j\in A} c_j\left(q_j\ln\left(\frac{q_j}{p_j(1)}\right) - q_j + p_j(1)\right)\end{aligned}$$

Combining with (29), we have (recall that $\eta \leq \frac{\gamma}{k}$):

$$\begin{aligned}\sum_{j\in A} q_j\cdot\mathbb{E}\left[\widetilde{G}_j\right] - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] &\leq 2\eta T k c_{\max} + \frac{1}{\eta}\sum_{j\in A} c_j\left(q_j\ln\left(\frac{q_j}{p_j(1)}\right) - q_j + p_j(1)\right) + 4\gamma T c_{\max}\\ &\leq \frac{1}{\eta}\sum_{j\in A} c_j\left(q_j\ln\left(\frac{q_j}{p_j(1)}\right) - q_j + p_j(1)\right) + 6\gamma T c_{\max}\end{aligned}$$

Let $\mathbf{q} = (1-\gamma)\mathbf{1}_i + \frac{\gamma}{k}\mathbf{1}$. Recall that $\mathbf{p}(1) = (1-\gamma)\mathbf{1}_{i_{\min}} + \frac{\gamma}{k}\mathbf{1}$ (recall $i_{\min}$ is the arm with minimum range $c_{i_{\min}}$). Similar to the discussion for the expert problem in Section 2.5, the 1st term on the RHS is upper bounded by $O\left(\frac{1}{\eta}\log\left(\frac{k}{\gamma}\right)\cdot c_i\right)$. Hence, we have:

$$\sum_{j\in A} q_j\cdot\mathbb{E}\left[\widetilde{G}_j\right] - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] \leq O\left(\frac{1}{\eta}\log\left(\frac{k}{\gamma}\right)\cdot c_i\right) + 6\gamma T c_{\max}\ . \qquad (31)$$

Further, the LHS is lower bounded as:

$$(1-\gamma)\mathbb{E}\left[\widetilde{G}_i\right] + \frac{\gamma}{k}\sum_{j\in A}\mathbb{E}\left[\widetilde{G}_j\right] - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right] \geq (1-\gamma)\mathbb{E}\left[\widetilde{G}_i\right] - \gamma T c_{\max} - \mathbb{E}\left[\widetilde{G}_{\text{ALG}}\right]\ .$$

The lemma then follows by putting it back to (31) and rearranging terms. ■

Proof [of Theorem 21] Let $\gamma = \epsilon\frac{c_{\min}}{c_{\max}}$ and $\eta = \frac{\gamma}{k}$ in Lemma 26. Theorem follows noting that $\gamma c_{\max} = \epsilon c_{\min} \leq \epsilon c_i$. ■

5.5. Lower-bound for bandits with symmetric range - Proof of Theorem 23

We first show that for any online multi-scale bandits algorithm problem, and there is an instance that has two arms with $c_1 = 1$ and $c_2 = h$ for some sufficiently large h, a sufficiently large T, and $\epsilon = \sqrt{\frac{h}{256T}}$, such that either

$$\mathbb{E}\left[\text{REGRET}_1\right] > \epsilon T + \tfrac{1}{256\epsilon}h\ , \qquad \text{or} \qquad \mathbb{E}\left[\text{REGRET}_2\right] > \epsilon T h + \tfrac{1}{256\epsilon}h^2$$

We will prove the existence of this instance by looking at the stochastic setting, i.e., the gain vectors $\mathbf{g}(t)$'s are i.i.d. for $1 \leq t \leq T$. We consider two instances, both of which admit

a fixed gain of 0 for action 1. In the first instance, the gain of action 2 is h with probability $\frac{1}{2} - 2\epsilon$, and $-h$ otherwise. Hence, the expected gain of playing action 2 is $-4\epsilon h$ per round in instance 1. In the second instance, the gain of action 2 is h with probability $\frac{1}{2} + 2\epsilon$, and $-h$ otherwise. Hence, the expected gain of playing action two is $4\epsilon h$ per round in instance 2. Note this proves the theorem, as $c_{\min} = 1$, $c_{\max} = h$, $k = 2$ and and $T = \frac{h}{256\epsilon^2}$.

Suppose for contradiction that the algorithm satisfies:

$$\mathbb{E}\left[\text{REGRET}_1\right] \leq \epsilon T + \tfrac{1}{256\epsilon} h = \tfrac{1}{128\epsilon} h \quad , \quad \mathbb{E}\left[\text{REGRET}_2\right] \leq \epsilon h T + \tfrac{1}{256\epsilon} h^2 = \tfrac{1}{128\epsilon} h^2 \ .$$

Let N_1 denote the expected number of times that the algorithm plays action 2 in instance 1. Then, the expected regret with respect to action 1 in instance 1 is $N_1 \cdot 4\epsilon h$. By the assumption that $\mathbb{E}\left[\text{REGRET}_1\right] \leq \frac{1}{128\epsilon} h$, we have $N_1 \leq \frac{1}{512\epsilon^2}$.

Next, by standard calculation, we get that the Kullback-Leibler (KL) divergence of the observed rewards in a single round in the two instances is 0 if action 1 is played and is at most $64\epsilon^2$ (for $0 < \epsilon < 0.1$) if action 2 is played. So the KL divergence of the observed reward sequences in the two instances is at most $64\epsilon^2 \cdot N_1 \leq \frac{1}{8}$.

Then, we use a standard inequality about KL divergences. For any measurable function $\psi : X \mapsto \{1, 2\}$, we have $\Pr_{X \sim \rho_1}\big(\psi(X) = 2\big) + \Pr_{X \sim \rho_2}\big(\psi(X) = 1\big) \geq \frac{1}{2}\exp\big(-KL(\rho_1, \rho_2)\big)$. For any $1 \leq t \leq T$, let ρ_1 and ρ_2 be the distribution of observed rewards up to a round t in the two instances, and let $\psi(X)$ be the action played by the algorithm. By this inequality and the above bound on the KL divergence between the observed rewards in the two instances, we get that in each round, the probability that the algorithm plays action 2 in instance 1, plus the probability that the algorithm plays action 1 in instance 2, is at least $\frac{1}{2}\exp\left(-\frac{1}{8}\right) > \frac{2}{5}$ in any round t. Thus, the expected number of times that the algorithm plays action 1 in instance 2 from round 1 to T, denoted as N_2, is at least $N_2 \geq \frac{2}{5} \cdot T - N_1 \geq \frac{1}{3} \cdot T$, where the second inequality holds for sufficiently large h. Therefore, the expected regret w.r.t. action 2 in instance 2 is at least: $4\epsilon h \cdot \frac{1}{3} \cdot T = \frac{4}{3}\epsilon h T > \frac{1}{128\epsilon} h^2$. This is a contradiction to our assumption that $\mathbb{E}\left[\text{REGRET}_2\right] \leq \frac{1}{128\epsilon} h^2$.

6. Conclusion

Revenue management has emerged as a competitive toolbox of strategies for increasing the profit of web-based markets. In particular, dynamic pricing, and dynamic auction design as its less mature relative, have become prevalent market mechanisms in nearly all industries. In this paper, we studied these problems from the perspective of online learning. For the online auction for single buyer, we showed regret bounds that scale with the best fixed price, rather than the range of the values (with a generalization to learning auctions). Moreover, we demonstrated a connection between the optimal regret bounds for this problem and offline sample complexity lower-bounds of approximating optimal revenue, studied in Cole and Roughgarden (2014); Huang et al. (2015a). Using this connection, we showed our regret bounds are almost optimal as they match these information theoretic lower-bounds. We further generalized our result to online pricing (bandit feedback) and online auction with multiple-buyers.

The key to our development and improved regret bounds for online auction design is generalizing the classical learning from experts and multi-armed bandit problems to their "multi-scale versions", where the reward of each action is in a different range. Here the objective is to design online learning algorithms whose regret with respect to a given action scales with its own range, rather than the maximum range. We showed how a variant of online mirror descent solves this learning problem.

Acknowledgments

The authors would like to thank Bobby Kleinberg, Dylan Foster, and Aranyak Mehta for helpful discussions about this work. We would also like to thank anonymous reviewers for their constructive and helpful comments.

References

Shipra Agrawal and Nikhil R Devanur. Bandits with concave rewards and convex knapsacks. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 989–1006. ACM, 2014.

Kareem Amin, Afshin Rostamizadeh, and Umar Syed. Learning prices for repeated auctions with strategic buyers. In *Advances in Neural Information Processing Systems*, pages 1169–1177, 2013.

Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

Jean-Yves Audibert and Sébastien Bubeck. Minimax policies for adversarial and stochastic bandits. In *COLT*, pages 217–226, 2009.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 322–331. IEEE, 1995.

Moshe Babaioff, Shaddin Dughmi, Robert Kleinberg, and Aleksandrs Slivkins. Dynamic pricing with limited supply. *ACM Transactions on Economics and Computation*, 3(1):4, 2015.

Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 207–216. IEEE, 2013.

Maria-Florina Balcan, Avrim Blum, Jason D Hartline, and Yishay Mansour. Reducing mechanism design to algorithm design via machine learning. *Journal of Computer and System Sciences*, 74(8):1245–1270, 2008.

Ziv Bar-Yossef, Kirsten Hildrum, and Felix Wu. Incentive-compatible online auctions for digital goods. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 964–970. Society for Industrial and Applied Mathematics, 2002.

Omar Besbes and Assaf Zeevi. Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research*, 57(6):1407–1420, 2009.

Avrim Blum and Jason D Hartline. Near-optimal online auctions. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1156–1163. Society for Industrial and Applied Mathematics, 2005.

Avrim Blum, Vijay Kumar, Atri Rudra, and Felix Wu. Online learning in online auctions. *Theoretical Computer Science*, 324(2-3):137–146, 2004.

Sébastien Bubeck. Introduction to online optimization. *Lecture Notes*, pages 1–86, 2011.

Yang Cai, Constantinos Daskalakis, and S Matthew Weinberg. Optimal multi-dimensional mechanism design: Reducing revenue to welfare maximization. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 130–139. IEEE, 2012.

Richard Cole and Tim Roughgarden. The sample complexity of revenue maximization. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 243–252, 2014.

Arnoud V den Boer. Dynamic pricing and learning: historical origins, current research, and new directions. *Surveys in operations research and management science*, 20(1):1–18, 2015.

Nikhil R Devanur, Zhiyi Huang, and Christos-Alexandros Psomas. The sample complexity of auctions with side information. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 426–439. ACM, 2016.

Peerapong Dhangwatnotai, Tim Roughgarden, and Qiqi Yan. Revenue maximization with a single sample. *Games and Economic Behavior*, 2014.

Edith Elkind. Designing and learning optimal finite support auctions. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 736–745. Society for Industrial and Applied Mathematics, 2007.

Dylan J Foster, Satyen Kale, Mehryar Mohri, and Karthik Sridharan. Parameter-free online learning via model selection. In *Advances in Neural Information Processing Systems*, pages 6020–6030, 2017.

Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

Yannai A Gonczarowski and Noam Nisan. Efficient empirical revenue maximization in single-parameter auction environments. In *Proceedings of the ACM STOC*, 2017.

Zhiyi Huang, Yishay Mansour, and Tim Roughgarden. Making the most of your samples. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC '15, Portland, OR, USA, June 15-19, 2015*, pages 45–60, 2015a.

Zhiyi Huang, Yishay Mansour, and Tim Roughgarden. Making the most of your samples. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, pages 45–60. ACM, 2015b.

Robert Kleinberg and Tom Leighton. The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 594–605. IEEE, 2003.

Jamie H Morgenstern and Tim Roughgarden. On the pseudo-dimension of nearly optimal auctions. In *Advances in Neural Information Processing Systems*, pages 136–144, 2015.

Roger B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1): 58–73, 1981.

Tim Roughgarden and Okke Schrijvers. Ironing in the dark. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 1–18. ACM, 2016.

Ilya Segal. Optimal pricing mechanisms with unknown demand. *The American economic review*, 93(3):509–529, 2003.

Vasilis Syrgkanis. A sample complexity measure with applications to learning optimal auctions. *arXiv preprint arXiv:1704.02598*, 2017.

Kalyan T Talluri and Garrett J Van Ryzin. *The theory and practice of revenue management*, volume 68. Springer Science & Business Media, 2006.

Appendix A. Other Deferred Proofs and Discussions

A.1. Discussion on choice of $\boldsymbol{\pi}$ for bandit symmetric range

We now describe how the choice of initial distribution $\boldsymbol{\pi}$ affects the bound given in Theorem 18.

- When the action set is finite, we can choose $\boldsymbol{\pi}$ to be the uniform distribution to get the term
$$O\big(\frac{1}{\epsilon}\log(kc_i/c_{\min})\cdot c_i\big)$$
This recovers the standard bound by setting $c_i = c_{\max}$ for all $i \in A$.
- We can choose $\pi_i = \frac{c_i}{\sum_{j\in A} c_j}$ to get $O\big(\frac{1}{\epsilon}\log(\sum_{j\in A} c_j/c_{\min})\cdot c_i\big)$. In particular, if the c_i's form an arithmetic progression with a constant difference then this is just $O\big(\frac{\log k}{\epsilon}\cdot c_i\big)$.

A.2. Elementary proof of Proposition 10 from first principles

Based on the update rule of Algorithm 1, we have $g_i(t) = \frac{c_i}{\eta}\log(\frac{w_i(t+1)}{p_i(t)})$ for any $i \in A$. Therefore:

$$\begin{aligned}
\mathbf{g}(t)\cdot\big(\mathbf{q}-\mathbf{p}(t)\big) &= \sum_{i\in A} g_i(t)\big(q_i - p_i(t)\big)\\
&= \sum_{i\in A}\frac{c_i}{\eta}\cdot\log\big(\frac{w_i(t+1)}{p_i(t)}\big)\cdot\big(q_i-p_i(t)\big)\\
&= \frac{1}{\eta}\left(\sum_{i\in S} c_i\cdot q_i\cdot\log\big(\frac{w_k(t+1)}{p_k(t)}\big) + \sum_{i\in A} c_i\cdot p_i(t)\cdot\log\big(\frac{p_i(t)}{w_i(t+1)}\big)\right)\\
&= \frac{1}{\eta}\Big(\sum_{i\in S} c_i\cdot q_i\cdot\log\big(\frac{w_k(t+1)}{p_k(t+1)}\big) + \sum_{i\in S} c_i\cdot q_i\cdot\log\big(\frac{p_k(t+1)}{p_k(t)}\big)\\
&\qquad + \sum_{i\in A} c_i\cdot p_i(t)\cdot\log\big(\frac{p_i(t)}{w_i(t+1)}\big)\Big) \qquad (32)
\end{aligned}$$

Now, note that due to the normalization step of Algorithm 1, for any $i \in S$ we have:

$$c_i\cdot\log(\frac{w_i(t+1)}{p_i(t+1)}) = \lambda = \sum_{j\in A} c_j\cdot p_j(t+1)\cdot\frac{\lambda}{c_j} = \sum_{j\in A} c_j\cdot p_j(t+1)\cdot\log(\frac{w_j(t+1)}{p_j(t+1)})$$

So the first summation in (32) is equal to:

$$\begin{aligned}
\sum_{i\in S} c_i\cdot q_i\cdot\log\big(\frac{w_k(t+1)}{p_k(t+1)}\big) &= \sum_{i\in S} q_i\cdot\sum_{j\in A} c_j\cdot p_j(t+1)\cdot\log(\frac{w_j(t+1)}{p_j(t+1)})\\
&= \sum_{j\in A} c_j\cdot p_j(t+1)\cdot\log(\frac{w_j(t+1)}{p_j(t+1)})\\
&= \sum_{i\in A} c_i\cdot p_i(t+1)\cdot\log(\frac{w_i(t+1)}{p_i(t+1)}) \qquad (33)
\end{aligned}$$

Combining Eqn. (32) and (33), we have:

$$\begin{aligned}
\mathbf{g}(t)\cdot\big(\mathbf{q}-\mathbf{p}(t)\big) &= \frac{1}{\eta}\sum_{i\in A} c_i\cdot\left(p_i(t)\cdot\log(\frac{p_i(t)}{w_i(t+1)}) + p_i(t+1)\cdot\log(\frac{w_i(t+1)}{p_i(t+1)})\right)\\
&\quad + \frac{1}{\eta}\sum_{i\in S} c_i\cdot q_i\cdot\log\big(\frac{p_i(t+1)}{p_i(t)}\big)
\end{aligned}$$

The 2nd part is a telescopic sum when we sum over t. We will upper bound the 1st part as follows. By $\log(x) \le (x-1)$, we get that:

$$\begin{aligned}
&\sum_{i\in A} c_i\cdot\left(p_i(t)\cdot\log(\frac{p_i(t)}{w_i(t+1)}) + p_i(t+1)\cdot\log(\frac{w_i(t+1)}{p_i(t+1)})\right)\\
&\quad\le \sum_{i\in A} c_i\cdot\left(p_i(t)\cdot\log(\frac{p_i(t)}{w_i(t+1)}) - p_i(t+1) + w_i(t+1)\right)\\
&\quad= \sum_{i\in A} c_i\cdot\big(p_i(t)-p_i(t+1)\big) + \sum_{i\in A} c_i\cdot\left(p_i(t)\cdot\log(\frac{p_i(t)}{w_i(t+1)}) - p_i(t) + w_i(t+1)\right)
\end{aligned}$$

Again, the 1st part is a telescopic sum when we sum over t. We will further work on the 2nd part. By the relation between $w_i(t+1)$ and $p_i(t)$, we get that:

$$\sum_{i\in A} c_i\cdot\left(p_i(t)\cdot\log(\frac{p_i(t)}{w_i(t+1)}) - p_i(t) + w_i(t+1)\right) = \sum_{i\in A} c_i\cdot p_i(t)\left(-\eta\cdot\frac{g_i(t)}{c_i} - 1 + \exp(\eta\cdot\frac{g_i(t)}{c_i})\right)$$

Note that $\eta\cdot\frac{g_i(t)}{c_i} \in [-1,1]$ because $g_i(t) \in [-c_i, c_i]$ and $0 < \eta \le 1$. By $\exp(x) - x - 1 \le x^2$ for $-1 \le x \le 1$ and that $\eta g_i(t) \in [-c_i, c_i]$, the above is upper bounded by $\eta^2\sum_{i\in A} p_i(t)\frac{(g_i(t))^2}{c_i}$. Putting together, we get that:

$$\mathbf{g}(t)\cdot\big(\mathbf{q}-\mathbf{p}(t)\big) \le \frac{1}{\eta}\sum_{i\in S} c_i\cdot\left(q_i\cdot\log\big(\frac{p_i(t+1)}{p_i(t)}\big) + p_i(t) - p_i(t+1)\right) + \eta\sum_{i\in A} p_i(t)\frac{(g_i(t))^2}{c_i}$$

Summing over t, we have:

$$\mathbf{g}(t)\cdot\big(\mathbf{q}-\mathbf{p}(t)\big) \le \frac{1}{\eta}\sum_{i\in S} c_i\cdot\left(q_i\cdot\log\big(\frac{p_i(T+1)}{p_i(1)}\big) + p_i(1) - p_i(T+1)\right) + \eta\sum_{t\in[T]}\sum_{i\in A} p_i(t)\frac{(g_i(t))^2}{c_i}$$

Finally, by $\log(x) \le (x-1)$, we get that $q_i\log\big(\frac{p_i(T+1)}{q_i}\big) \le p_i(T+1) - q_i$. Hence, we have:

$$\mathbf{g}(t)\cdot\big(\mathbf{q}-\mathbf{p}(t)\big) \le \frac{1}{\eta}\sum_{i\in S} c_i\cdot\left(q_i\cdot\log\big(\frac{q_i}{p_i(1)}\big) + p_i(1) - q_i\right) + \eta\sum_{t\in[T]}\sum_{i\in A} p_i(t)\frac{(g_i(t))^2}{c_i}$$

The proposition then follows by our choice of the initial distribution.

A.3. Proof of OMD regret bound

In order to prove the OMD regret bound, we need some properties of Bregman divergence.

Lemma 27 (Properties of Bregman divergence (Bubeck, 2011)) *Suppose $F(\cdot)$ is a Legendre function and $D_F(\cdot,\cdot)$ is its associated Bregman divergence as defined in Definition 4. Then:*

- *$D_F(x,y) > 0$ if $x \neq y$ as F is strictly convex, and $D_F(x,x) = 0$.*
- *$D_F(.,y)$ is a convex function for any choice of y.*
- *(*Pythagorean theorem*) If $\mathcal{A}$ is a convex set, $a \in \mathcal{A}$, $b \notin \mathcal{A}$ and $c = \underset{x\in\mathcal{A}}{argmin}(D_F(x,b))$, then*

$$D_F(a,c) + D_F(c,b) \leq D_F(a,b)$$

Given Lemma 27, we are now ready to prove Lemma 6.

Proof [Proof of Lemma 6] To obtain the OMD regret bound, we have:

$$\begin{aligned}
\mathbf{q}\cdot\mathbf{g}(t) - \mathbf{p}(t)\cdot\mathbf{g}(t) &= \frac{1}{\eta}(\mathbf{q}-\mathbf{p}(t))\cdot(\nabla F(\mathbf{w}(t+1)) - \nabla F(\mathbf{p}(t))) \\
&= \frac{1}{\eta}(D_F(qb,\mathbf{p}(t)) + D_F(\mathbf{p}(t),\mathbf{w}(t+1)) - D_F(qb,\mathbf{w}(t+1))) \\
&\overset{(1)}{\leq} \frac{1}{\eta}D_F(\mathbf{p}(t),\mathbf{w}(t+1)) + \frac{1}{\eta}\left(D_F(\mathbf{q},\mathbf{p}(t)) - D_F(\mathbf{q},\mathbf{p}(t+1))\right) \qquad (34)
\end{aligned}$$

where in (1) we use $D_F(\mathbf{p}(t+1),\mathbf{w}(t+1)) \geq 0$ and $D_F(\mathbf{q},\mathbf{p}(t+1)) + D_F(\mathbf{p}(t+1),\mathbf{w}(t+1)) \leq D_F(\mathbf{q},\mathbf{w}(t+1))$ due to Pythagorean theorem (Lemma 27). By summing up both hand sides of (34) for $t = 1,\cdots,T$ we have:

$$\sum_{t\in[T]} \mathbf{g}(t)\cdot\big(\mathbf{q}-\mathbf{p}(t)\big) \leq \frac{1}{\eta}\sum_{t\in[T]} D_F(\mathbf{p}(t),\mathbf{w}(t+1)) + \frac{1}{\eta}D_F(\mathbf{q},\mathbf{p}(1)) \qquad (35)$$

■

Journal of Machine Learning Research 20 (2019) 1-17 Submitted 10/17; Revised 2/19; Published 4/19

Nearly-tight VC-dimension and Pseudodimension Bounds for Piecewise Linear Neural Networks*

Peter L. Bartlett BARTLETT@CS.BERKELEY.EDU
Department of Statistics and Computer Science Division
University of California
Berkeley, CA 94720-3860, USA

Nick Harvey NICKHAR@CS.UBC.CA
Christopher Liaw CVLIAW@CS.UBC.CA
Abbas Mehrabian† ABBAS.MEHRABIAN@GMAIL.COM
Department of Computer Science
University of British Columbia
Vancouver, BC V6T 1Z4, Canada

Editor: Nicolas Vayatis

Abstract

We prove new upper and lower bounds on the VC-dimension of deep neural networks with the ReLU activation function. These bounds are tight for almost the entire range of parameters. Letting W be the number of weights and L be the number of layers, we prove that the VC-dimension is $O(WL\log(W))$, and provide examples with VC-dimension $\Omega(WL\log(W/L))$. This improves both the previously known upper bounds and lower bounds. In terms of the number U of non-linear units, we prove a tight bound $\Theta(WU)$ on the VC-dimension. All of these bounds generalize to arbitrary piecewise linear activation functions, and also hold for the pseudodimensions of these function classes.

Combined with previous results, this gives an intriguing range of dependencies of the VC-dimension on depth for networks with different non-linearities: there is no dependence for piecewise-constant, linear dependence for piecewise-linear, and no more than quadratic dependence for general piecewise-polynomial.

Keywords: VC-dimension, pseudodimension, neural networks, ReLU activation function, statistical learning theory

1. Introduction

Deep neural networks underlie many of the recent breakthroughs in applied machine learning, particularly in image and speech recognition. These successes motivate a renewed study of these networks' theoretical properties.

Classification is one of the learning tasks in which deep neural networks have been particularly successful, e.g., for image recognition. A natural foundational question that arises is: what are the generalization guarantees of these networks in a statistical learning framework? An established way to address this question is by considering the VC-dimension, which characterizes uniform convergence of misclassification frequencies to probabilities (see Vapnik and Chervonenkis, 1971), and asymptotically determines the sample complexity of PAC learning with such classifiers (see Blumer, Ehrenfeucht, Haussler, and Warmuth, 1989).

Definition 1 (growth function, VC-dimension, shattering) *Let H denote a class of functions from $\mathcal{X}$ to $\{0,1\}$ (the hypotheses, or the classification rules). For any non-negative integer m, we define the* growth function *of H as*

$$\Pi_H(m) \coloneqq \max_{x_1,\ldots,x_m\in\mathcal{X}} |\{(h(x_1),\ldots,h(x_m)) : h\in H\}|\,.$$

If $|\{(h(x_1),\ldots,h(x_m)) : h\in H\}| = 2^m$, we say H shatters *the set $\{x_1,\ldots,x_m\}$. The Vapnik-Chervonenkis dimension of H, denoted* $\mathrm{VCdim}(H)$*, is the size of the largest shattered set, i.e. the largest m such that $\Pi_H(m) = 2^m$. If there is no largest m, we define $\mathrm{VCdim}(H) = \infty$.*

For a class of real-valued functions, such as those generated by neural networks, a natural measure of complexity that implies similar uniform convergence properties is the pseudodimension, introduced by Pollard (1990) (see, e.g., Anthony and Bartlett (1999, Theorem 19.2)).

Definition 2 (pseudodimension) *Let $\mathcal{F}$ be a class of functions from $\mathcal{X}$ to $\Re$. The pseudodimension of $\mathcal{F}$, written* $\mathrm{Pdim}(\mathcal{F})$*, is the largest integer m for which there exists $(x_1,\ldots,x_m,y_1,\ldots,y_m)\in\mathcal{X}^m\times\Re^m$ such that for any $(b_1,\ldots,b_m)\in\{0,1\}^m$ there exists $f\in\mathcal{F}$ such that*

$$\forall i : f(x_i) > y_i \iff b_i = 1$$

For a class $\mathcal{F}$ of real-valued functions, we define $\mathrm{VCdim}(\mathcal{F}) \coloneqq \mathrm{VCdim}(\mathrm{sgn}(\mathcal{F}))$, where

$$\mathrm{sgn}(\mathcal{F}) \coloneqq \{\mathrm{sgn}(f) : f\in\mathcal{F}\}$$

and $\mathrm{sgn}(x) = \mathbf{1}[x > 0]$. For any class $\mathcal{F}$, clearly $\mathrm{VCdim}(\mathcal{F}) \le \mathrm{Pdim}(\mathcal{F})$. If $\mathcal{F}$ is the class of functions generated by a neural network $\mathcal{N}$ with a fixed architecture and fixed activation functions (see Section 1.3 for definitions), then it is not hard to see that indeed $\mathrm{Pdim}(\mathcal{F}) \le \mathrm{VCdim}(\mathcal{F}')$, where $\mathcal{F}'$ is the class of functions generated by a certain neural network with one more parameter and one more layer than $\mathcal{N}$ (see Anthony and Bartlett (1999, Theorem 14.1) for a proof). Therefore, all the results of this paper automatically apply to the pseudodimensions of neural networks as well, after appropriate adjustments.

The main contribution of this paper is to prove nearly-tight bounds on the VC-dimension of deep neural networks in which the non-linear activation function is a piecewise linear function with a constant number of pieces. For simplicity we will henceforth refer to such networks as "piecewise linear networks". The activation function that is the most commonly used in practice is the *rectified linear unit*, also known as *ReLU* (see LeCun, Bengio, and

*. An extended abstract appeared in Proceedings of the Conference on Learning Theory (COLT) 2017: http://proceedings.mlr.press/v65/harvey17a.html; the upper bound was presented at the 2016 ACM Conference on Data Science: http://ikdd.acm.org/Site/CoDS2016/keynotes.html. This version includes all the proofs and a refinement of the upper bound, Theorem 7.

†. corresponding author

©2019 Peter L. Bartlett, Nick Harvey, Christopher Liaw and Abbas Mehrabian.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-612.html.

Hinton, 2015; Goodfellow, Bengio, and Courville, 2016). The ReLU function is defined as $\sigma(x) = \max\{0, x\}$, so it is clearly piecewise linear.

It is particularly interesting to understand how the VC-dimension is affected by the various attributes of the network: the number W of parameters (i.e., weights and biases), the number U of non-linear units (i.e., nodes), and the number L of layers. Among all networks with the same size (number of weights), is it true that those with more layers have larger VC-dimension?

Such a statement is indeed true, and previously known; however, a tight characterization of how depth affects VC-dimension was unknown prior to this work.

1.1. Our results

Our first main result is a new VC-dimension lower bound that holds even for the restricted family of ReLU networks.

Theorem 3 (Main lower bound) *There exists a universal constant C such that the following holds. Given any W, L with $W > CL > C^2$, there exists a ReLU network with $\leq L$ layers and $\leq W$ parameters with VC-dimension $\geq WL\log(W/L)/C$.*

Remark 4 *Our construction can be augmented slightly to give a neural network with linear threshold and identity activation functions with the same guarantees.*

Remark 5 *Our goal in this paper is to give asymptotic lower and upper bounds. However, one may wonder what is the smallest depth for which we can give a positive lower bound for the VC-dimension. Our Theorem 11 gives a positive lower bound as soon as the number of layers is at least 8 (which corresponds to $k = 1$ in that theorem).*

The proof appears in Section 2. Prior to our work, the best known lower bounds were $\Omega(WL)$ (see Bartlett, Maiorov, and Meir, 1998, Theorem 2) and $\Omega(W \log W)$ (see Maass, 1994, Theorem 1). We strictly improve both bounds to $\Omega(WL\log(W/L))$.

Our proof of Theorem 3 uses the "bit extraction" technique, which was also used by Bartlett et al. (1998) to give an $\Omega(WL)$ lower bound. We refine this technique to gain the additional logarithmic factor that appears in Theorem 3.

Unfortunately there is a barrier to refining this technique any further. Our next theorem shows the hardness of computing the mod function, implying that the bit extraction technique cannot yield a stronger lower bound than Theorem 3. Further discussion of this connection may be found in Remark 14.

Theorem 6 *Assume there exists a piecewise linear network with W parameters and L layers that computes a function $f : \Re \to \Re$, with the property that $|f(x) - (x \bmod 2)| < 1/2$ for all $x \in \{0, 1, \ldots, 2^m - 1\}$. Then we have $m = O(L\log(W/L))$.*

The proof of this theorem appears in Section 3. One interesting aspect of the proof is that it does not use Warren's lemma (Warren, 1968), which is a mainstay of VC-dimension upper bounds (see Goldberg and Jerrum, 1995; Bartlett et al., 1998; Anthony and Bartlett, 1999).

Our next main result is an upper bound on the VC-dimension of neural networks with piecewise polynomial activation functions.

Theorem 7 (Main upper bound) *Consider a neural network architecture with W parameters and U computation units arranged in L layers, so that each unit has connections only from units in earlier layers. Let k_i denote the number of units at the ith layer. Suppose that all non-output units have piecewise-polynomial activation functions with $p + 1$ pieces and degree no more than d, and the output unit has the identity function as its activation function.*

If $d = 0$, let W_i denote the number of parameters (weights and biases) at the inputs to units in layer i; if $d > 0$, let W_i denote the total number of parameters (weights and biases) at the inputs to units in all the layers ***up to layer*** *i (i.e., in layers $1, 2, \ldots, i$). Define the* effective depth *as*

$$\bar{L} \coloneqq \frac{1}{W}\sum_{i=1}^{L} W_i,$$

and let

$$R \coloneqq \sum_{i=1}^{L} k_i(1 + (i-1)d^{i-1}) \leq U + U(L-1)d^{L-1}. \tag{1}$$

For the class $\mathcal{F}$ of all (real-valued) functions computed by this network and $m \geq \bar{L}W$, we have

$$\Pi_{\mathrm{sgn}(\mathcal{F})}(m) \leq \prod_{i=1}^{L} 2\left(\frac{2emk_ip(1 + (i-1)d^{i-1})}{W_i}\right)^{W_i},$$

and if $U > 2$ then

$$\mathrm{VCdim}(\mathcal{F}) \leq L + \bar{L}W\log_2(4epR\log_2(2epR)) = O(\bar{L}W\log(pU) + \bar{L}LW\log d).$$

In particular, if $d = 0$, then

$$\mathrm{VCdim}(\mathcal{F}) \leq L + W\log_2(4epU\log_2(2epU)) = O(W\log(pU));$$

and if $d = 1$, then

$$\mathrm{VCdim}(\mathcal{F}) \leq L + \bar{L}W\log_2\left(4ep\sum_{i=1}^{L} ik_i\log_2\left(\sum_{i=1}^{L} 2epik_i\right)\right) = O(\bar{L}W\log(pU)).$$

Remark 8 *The average depth $\bar{L}$ is always between 1 and L, and captures how the parameters are distributed in the network: it is close to 1 if they are concentrated near the output (or if the activation functions are piecewise-constant), while it is of order L if the parameters are concentrated near the input, or are spread out throughout the network. Hence, this suggests that edges and vertices closer to the input have a larger effect in increasing the VC-dimension, a phenomenon not observed before; and indeed our lower bound construction in Theorem 3 (as well as the lower bound construction from Bartlett et al. (1998)) considers a network with most of the parameters near the input.*

Remark 9 *If we have $\sum_{i=1}^{L} k_i \leq \sum_{i=1}^{L} W_i$, which holds for most network architectures used in practice, the upper bound on the growth function can be simplified into*

$$\Pi_{\mathrm{sgn}(\mathcal{F})}(m) \leq \left(4emp(1 + (L-1)d^{L-1})\right)^{\sum W_i}.$$

The proofs of Theorem 7 and Remark 9 appear in Section 4. Prior to our work, the best known upper bounds were $O(W^2)$ (see Goldberg and Jerrum, 1995, Section 3.1) and $O(WL\log W+WL^2)$ (see Bartlett et al., 1998, Theorem 1), both of which hold for piecewise polynomial activation functions with a bounded number of pieces (for the remainder of this section, assume that $p = O(1)$ throughout); we strictly improve the second bound to $O(WL\log W)$ for the special case of piecewise linear functions ($d = 1$). Recall that ReLU is an example of a piecewise linear activation function. For the case $d = 0$, an $O(W\log U)$ bound for the VC-dimension was already proved using different techniques by Cover (1968) and by Baum and Haussler (1989, Corollary 2). Our Theorem 7 implies all of these upper bounds (except the $O(W^2)$ upper bound of Goldberg and Jerrum) using a unified technique, and gives a slightly more refined picture of the dependence of the VC-dimension on the distribution of parameters in a deep network.

To compare our upper and lower bounds, let $d(W, L)$ denote the largest VC-dimension of a piecewise linear network with W parameters and L layers. Theorems 3 and 7 imply there exist constants c, C such that

$$c \cdot WL\log(W/L) \;\le\; d(W, L) \;\le\; C \cdot WL\log W\,. \tag{2}$$

For neural networks arising in practice it would certainly be the case that L is significantly smaller than $W^{0.99}$, in which case our results determine the asymptotic bound $d(W, L) = \Theta(WL\log W)$. On the other hand, in the regime $L = \Theta(W)$, which is merely of theoretical interest, we also now have a tight bound $d(W, L) = \Theta(WL)$, obtained by combining Theorem 3 with results of Goldberg and Jerrum (1995). There is now only a very narrow regime, say $W^{0.99} \ll L \ll W$, in which the bounds of (2) are not asymptotically tight, and they differ only in the logarithmic factor.

Our final result is an upper bound for VC-dimension in terms of W and U (the number of non-linear units, or nodes). This bound is tight in the case $d = 1$ and $p = 2$, as discussed in Remark 12.

Theorem 10 *Consider a neural network with W parameters and U units with activation functions that are piecewise polynomials with at most p pieces and of degree at most d. Let $\mathcal{F}$ be the set of (real-valued) functions computed by this network. Then* $\mathrm{VCDim}(\mathrm{sgn}(\mathcal{F})) = O(WU\log((d+1)p))$.

The proof of this result appears in Section 5. The best known upper bound before our work was $O(W^2)$, implicitly proven for bounded d and p by Goldberg and Jerrum (1995, Section 3.1). Our theorem improves this to the tight result $O(WU)$.

We can summarize the tightest known results on the VC-dimension of neural networks with piecewise polynomial activation functions as follows: for classes $\mathcal{F}$ of functions computed by the class of networks with L layers, W parameters, and U units with the following non-linearities, we have the following bounds on VC-dimension:

Piecewise constant. $\mathrm{VCdim}(\mathcal{F}) = \Theta(W\log W)$ (Cover (1968) and Baum and Haussler (1989) showed the upper bound and Maass (1994) showed the lower bound).

Piecewise linear. $c \cdot WL\log(W/L) \;\le\; \mathrm{VCdim}(\mathcal{F}) \;\le\; C \cdot WL\log W$ (this paper).

Piecewise polynomial. $\mathrm{VCdim}(\mathcal{F}) = O(WL^2 + WL\log W)$ (Bartlett et al., 1998), and $\mathrm{VCdim}(\mathcal{F}) = O(WU)$ (this paper), and $\mathrm{VCdim}(\mathcal{F}) = \Omega(WL\log(W/L))$ (this paper). We note that closing the gap here requires fundamentally new techniques (see Remark 14).

1.2. Related work

For a general overview on theoretical properties of neural networks, we refer the reader to the monograph by Anthony and Bartlett (1999). There are various methods, other than the VC-dimension, for bounding the generalization error of neural networks; see Jakubovitz, Giryes, and Rodrigues (2018) for a recent survey.

In the reset of this section, we review some previous work studying the impact of depth on the representational power of neural networks. It has long been known that two-layer networks with a variety of activation functions can approximate arbitrary continuous functions on compact sets (Hornik, 1991). Sontag (1992) showed that three-layer networks of linear threshold units can approximate inverses of continuous functions, whereas two-layer networks cannot. There are several recent papers that aim to understand which functions can be expressed using a neural network of a given depth and size. There are technical similarities between our work and these. Two striking papers considered the problem of approximating a deep neural network with a shallower network. Telgarsky (2016) shows that there is a ReLU network with L layers and $U = \Theta(L)$ units such that any network approximating it with only $O(L^{1/3})$ layers must have $\Omega(2^{L^{1/3}})$ units; this phenomenon holds even for real-valued functions. Eldan and Shamir (2016) show an analogous result for a high-dimensional 3-layer network that cannot be approximated by a 2-layer network except with an exponential blow-up in the number of nodes.

Very recently, several authors have shown that deep neural networks are capable of approximating broad classes of functions. Safran and Shamir (2017) show that a sufficiently non-linear C^2 function on $[0, 1]^d$ can be approximated with ϵ error in L_2 by a ReLU network with $O(\mathrm{polylog}(1/\epsilon))$ layers and weights, but any such approximation with $O(1)$ layers requires $\Omega(1/\epsilon)$ weights. Yarotsky (2017) shows that any C^n-function on $[0, 1]^d$ can be approximated with ϵ error in L_∞ by a ReLU network with $O(\log(1/\epsilon))$ layers and $O((\frac{1}{\epsilon})^{d/n}\log(1/\epsilon))$ weights. Liang and Srikant (2017) show that a sufficiently smooth univariate function can be approximated with ϵ error in L_∞ by a network with ReLU and threshold gates with $\Theta(\log(1/\epsilon))$ layers and $O(\mathrm{polylog}(1/\epsilon))$ weights, but that $\Omega(\mathrm{poly}(1/\epsilon))$ weights would be required if there were only $o(\log(1/\epsilon))$ layers; they also prove analogous results for multivariate functions. Lastly, Cohen, Sharir, and Shashua (2016) draw a connection to tensor factorizations to show that, for a certain family of arithmetic circuits (in particular, without ReLU non-linearities), the set of functions computable by a shallow network have measure zero among those computable by a deep networks. For more recent results on representation power of neural networks, see Grohs, Perekrestenko, Elbrächter, and Bölcskei (2019) and the references therein.

1.3. Notation

A neural network is defined by an *activation function* $\psi : \Re \to \Re$, a directed acyclic graph, and a set of parameters: a weight for each edge of the graph, and a bias for each node of

the graph. Let W denote the number of parameters (weights and biases) of the network, U denote the number of computation units (nodes), and L denote the length of the longest path in the graph. We will say that the neural network has L layers.

Layer 0 consists of nodes with in-degree 0. We call these nodes input nodes and they simply output the real value given by the corresponding input to the network. We assume that the graph has a single sink node; this is the unique node in layer L, which we call the output layer. This output node can have predecessors in any layer $\ell < L$. For $1 \le \ell < L$, a node is in layer ℓ if it has a predecessor in layer $\ell - 1$ and no predecessor in any layer $\ell' \ge \ell$. (Note that for example there could be an edge connecting a node in layer 1 with a node in layer 3.) In the jargon of neural networks, layers 1 through $L-1$ are called *hidden* layers.

The computation of a neural network proceeds as follows. For $i = 1, \ldots, L$, the input into a computation unit u at layer i is $w^\top x + b$, where x is the (real) vector corresponding to the outputs of the computational units with a directed edge to u, w is the corresponding vector of edge weights, and b is the bias parameter associated with u. For layers $1, \ldots, L-1$, the output of u is $\psi(w^\top x + b)$. For the output layer, we replace ψ with the identity, so the output is simply $w^\top x + b$. Since we consider VC-dimension, we will always take the sign of the output of the network, to make the output lie in $\{0, 1\}$ for binary classification.

A piecewise polynomial function with p pieces is a function f for which there exists a partition of $\Re$ into disjoint intervals (pieces) $I_1, \ldots, I_p$ and corresponding polynomials $f_1, \ldots, f_p$ such that if $x \in I_i$ then $f(x) = f_i(x)$. A piecewise linear function is a piecewise polynomial function in which each f_i is linear. The most common activation function used in practice is the rectified linear unit (ReLU) where $I_1 = (-\infty, 0]$, $I_2 = (0, \infty)$ and $f_1(x) = 0, f_2(x) = x$. We denote this function by $\sigma(x) := \max\{0, x\}$. The set $\{1, 2, \ldots, n\}$ is denoted $[n]$.

2. Proof of Theorem 3

The proof of our main lower bound uses the "bit extraction" technique from Bartlett et al. (1998), who proved an $\Omega(WL)$ lower bound. We refine this technique in a key way — we partition the input bits into blocks and extract multiple bits at a time instead of a single bit at a time. This yields a more efficient bit extraction network, and hence a stronger VC-dimension lower bound.

We show the following result, which immediately implies Theorem 3.

Theorem 11 *Let r, m, n be positive integers, and let $k = \lceil m/r \rceil$. There exists a ReLU network with $3 + 5k$ layers, $2 + n + 4m + k((11 + r)2^r + 2r + 2)$ parameters, $m + n$ input nodes and $m + 2 + k(5 \times 2^r + r + 1)$ computational nodes with VC-dimension $\ge mn$.*

Remark 12 *Choosing $r = 1$ gives a network with $W = O(m + n)$, $U = O(m)$ and VC-dimension $\Omega(mn) = \Omega(WU)$. This implies that the upper bound $O(WU)$ given in Theorem 10 is tight.*

To prove Theorem 3, assume W, L, and W/L are sufficiently large, and set $r = \log_2(W/L)/2$, $m = rL/8$, and $n = W - 5m2^r$ in Theorem 11. The rest of this section is devoted to proving Theorem 11.

Let $S_n \subseteq \Re^n$ denote the standard basis. We shatter the set $S_n \times S_m$. Given an arbitrary function $f : S_n \times S_m \to \{0, 1\}$, we build a ReLU neural network that takes as input $(x_1, x_2) \in S_n \times S_m$ and outputs $f(x_1, x_2)$. Define n numbers $a_1, a_2, \ldots, a_n \in \{\frac{0}{2^m}, \frac{1}{2^m}, \ldots, \frac{2^m-1}{2^m}\}$ so that the ith digit of the binary representation of a_j equals $f(e_j, e_i)$. These numbers will be used as the parameters of the network, as described below.

Given input $(x_1, x_2) \in S_n \times S_m$, assume that $x_1 = e_j$ and $x_2 = e_i$. The network must output the ith bit of a_j. This "bit extraction approach" was used in Bartlett et al. (1998, Theorem 2) to give an $\Omega(WL)$ lower bound for the VC-dimension. We use a similar approach but we introduce a novel idea: we split the bit extraction into blocks and extract r bits at a time instead of a single bit at a time. This allows us to prove a lower bound of $\Omega(WL \log(W/L))$. One can ask, naturally, whether this approach can be pushed further. Our Theorem 6 implies that the bit extraction approach cannot give a lower bound better than $\Omega(WL \log(W/L))$ (see Remark 14).

The first layer of the network "selects" a_j, and the remaining layers "extract" the ith bit of a_j. In the first layer we have a single computational unit that calculates

$$a_j = (a_1, \ldots, a_n)^\top x_1 = \sigma\left((a_1, \ldots, a_n)^\top x_1\right).$$

This part uses 1 layer, 1 computation unit, and $1 + n$ parameters.

The rest of the network extracts all bits of a_j and outputs the ith bit. The extraction is done in k steps, where in each step we extract the r most significant bits and zero them out. We will use the following building block for extracting r bits.

Lemma 13 *Suppose positive integers r and m are given. There exists a ReLU network with 5 layers, $5 \times 2^r + r + 1$ units and $11 \times 2^r + r2^r + 2r + 2$ parameters that given the real number $b = 0.b_1 b_2 \ldots b_m$ (in binary representation) as input, outputs the $(r+1)$-dimensional vector $(b_1, b_2, \ldots, b_r, 0.b_{r+1} b_{r+2} \ldots b_m)$.*

Figure 1 shows a schematic of the ReLU network in the above lemma.

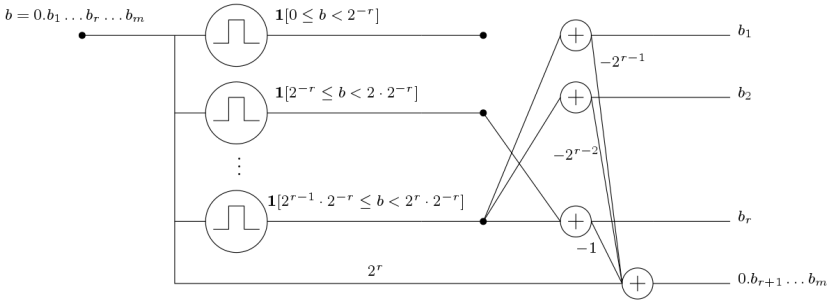


Figure 1: The ReLU network used to extract the most significant r bits of a number. Unlabeled edges indicate a weight of 1 and missing edges indicate a weight of 0.

Proof Partition $[0,1)$ into 2^r even subintervals. Observe that the values of $b_1,\dots,b_r$ are determined by knowing which such subinterval b lies in. We first show how to design a two-layer ReLU network that computes the indicator function for an interval to any arbitrary precision. Using 2^r of these networks in parallel allows us to determine which subinterval b lies in and hence, determine the bits $b_1,\dots,b_r$.

For any $a \le b$ and $\varepsilon > 0$, observe that the function $f(x) \coloneqq \sigma(1-\sigma(a/\varepsilon - x/\varepsilon)) + \sigma(1-\sigma(x/\varepsilon - b/\varepsilon)) - 1$ has the property that, $f(x) = 1$ for $x \in [a,b]$, and $f(x) = 0$ for $x \notin (a-\varepsilon, b+\varepsilon)$, and $f(x) \in [0,1]$ for all x. Thus we can use f to approximate the indicator function for $[a,b]$, to any desired precision. Moreover, this function can be computed with 3 layers, 5 units, and 11 parameters as follows. First, computing $\sigma(a/\varepsilon - x/\varepsilon)$ can be done with 1 unit, 1 layer, and 2 parameters. Computing $\sigma(1-\sigma(a/\varepsilon - x/\varepsilon))$ can be done with 1 additional unit, 1 additional layer, and 2 additional parameters. Similarly, $\sigma(1-\sigma(x/\varepsilon - b/\varepsilon))$ can be computed with 2 units, the same 2 layers, and 4 parameters. Computing the sum can be done with 1 additional layer, 1 additional unit, and 3 additional parameters. In total, computing f can be done with 3 layers, 5 units, and 11 parameters. We will choose $\varepsilon = 2^{-m-2}$ because we are working with m-digit numbers.

Thus, the values $b_1,\dots,b_r$ can be generated by adding the corresponding indicator variables. (For instance, $b_1 = \sum_{k=2^{r-1}}^{2^r-1} \mathbf{1}[b \in [k\cdot 2^{-r}, (k+1)\cdot 2^{-r})]$.) Finally, the remainder $0.b_{r+1}b_{r+2}\dots b_m$ can be computed as $0.b_{r+1}b_{r+2}\dots b_m = 2^r b - \sum_{k=1}^{r} 2^{r-k} b_k$.

Now we count the number of layers and parameters: we use 2^r small networks that work in parallel for producing the indicators, each has 3 layers, 5 units and 11 parameters. To produce $b_1,\dots,b_r$ we need an additional layer, $r\times(2^r+1)$ additional parameters, and r additional units. For producing the remainder we need 1 more layer, 1 more unit, and $r+2$ more parameters. ■

We use $\lceil m/r \rceil$ of these blocks to extract the bits of a_j, denoted by $a_{j,1},\dots,a_{j,m}$. Extracting $a_{j,i}$ is now easy, noting that if $x,y \in \{0,1\}$ then $x \wedge y = \sigma(x+y-1)$. So, since $x_2 = e_i$, we have

$$a_{j,i} = \sum_{t=1}^{m} x_{2,t} \wedge a_{j,t} = \sum_{t=1}^{m} \sigma(x_{2,t} + a_{j,t} - 1) = \sigma\left(\sum_{t=1}^{m} \sigma(x_{2,t} + a_{j,t} - 1)\right).$$

This calculation needs 2 layers, $1+m$ units, and $1+4m$ parameters.

Remark 14 *Theorem 6 implies an inherent barrier to proving lower bounds using the "bit extraction" approach from Bartlett et al. (1998). Recall that this technique uses n binary numbers with m bits to encode a function $f\colon S_n \times S_m \to \{0,1\}$ to show an $\Omega(mn)$ lower bound for VC-dimension, where S_k denotes the set of standard basis vectors in $\Re^k$. The network begins by selecting one of the n binary numbers, and then extracting a particular bit of that number. Bartlett et al. (1998) shows that it is possible to take $m = \Omega(L)$ and $n = \Omega(W)$, thus proving a lower bound of $\Omega(WL)$ for the VC-dimension. In Theorem 3 we showed we can increase m to $\Omega(L\log(W/L))$, improving the lower bound to $\Omega(WL\log(W/L))$. Theorem 6 implies that to extract just the least significant bit, one is forced to have $m = O(L\log(W/L))$; on the other hand, we always have $n \le W$. Hence there is no way to improve the VC-dimension lower bound by more than a constant via the bit extraction technique. In particular, for general piecewise polynomial networks, closing the gap between the $O(WL^2 + WL\log W)$ of Bartlett et al. (1998) and $\Omega(WL\log W/L)$ of this paper will require a different technique.*

3. Proof of Theorem 6

For a piecewise polynomial function $\Re \to \Re$, *breakpoints* are the boundaries between the pieces. So if a function has p pieces, it has $p-1$ breakpoints.

Lemma 15 *Let $f_1,\dots,f_k : \Re \to \Re$ be piecewise polynomials of degree D, and suppose the union of their breakpoints has size B. Let $\psi : \Re \to \Re$ be a piecewise polynomial of degree d with b breakpoints. Let $w_1,\dots,w_k \in \Re$ be arbitrary. The function $g(x) \coloneqq \psi(\sum_i w_i f_i(x))$ is piecewise polynomial of degree Dd with at most $(B+1)(2+bD)-1$ breakpoints.*

Proof Without loss of generality, assume that $w_1 = \cdots = w_k = 1$. The function $\sum_i f_i$ has $B+1$ pieces. Consider one such interval $\mathcal{I}$. We will prove that it will create at most $2+bD$ pieces in g. In fact, if $\sum_i f_i$ is constant on $\mathcal{I}$, g will have 1 piece on $\mathcal{I}$. Otherwise, for any point y, the equation $\sum_i f_i(x) = y$ has at most D solutions on $\mathcal{I}$. Let $y_1,\dots,y_b$ be the breakpoints of ψ. Suppose we move along the curve $(x, \sum_i f_i(x))$ on $\mathcal{I}$. Whenever we hit a point (t, y_i) for some t, one new piece is created in g. So at most bD new pieces are created. In addition, we may have two pieces for the beginning and ending of $\mathcal{I}$. This gives a total of $2+bD$ pieces per interval, as required. Finally, note that the number of breakpoints is one fewer than the number of pieces. ■

Theorem 6 follows immediately from the following theorem.

Theorem 16 *Assume there exists a neural network with W parameters and L layers that computes a function $f : \Re \to \Re$, with the property that $|f(x) - (x \bmod 2)| < 1/2$ for all $x \in \{0,1,\dots,2^m-1\}$. Also suppose the activation functions are piecewise polynomials of degree at most $d \ge 1$, and have at most $p \ge 1$ pieces. Then we have*

$$m \le L\log_2(13pd^{(L+1)/2}\cdot W/L).$$

In the special case of piecewise linear functions, this gives $m = O(L\log(W/L))$.

Proof For a node v of the network, let $\gamma(v)$ count the number of directed paths from the input node to v. Applying Lemma 15 iteratively gives that for a node v at layer $i \ge 1$, the number of breakpoints is bounded by $(6p)^i d^{i(i-1)/2}\gamma(v) - 1$. Let o denote the output node. Hence, o has at most $(6p)^L d^{L(L-1)/2}\gamma(o)$ pieces. The output of node o is piecewise polynomial of degree at most d^L. On the other hand, as we increase x from 0 to 2^m-1, the function $x \bmod 2$ flips 2^m-1 many times, which implies the output of o becomes equal to $1/2$ at least 2^m-1 times, thus we get

$$(6p)^L d^{L(L-1)/2}\gamma(o) \times d^L \ge 2^m - 1. \tag{3}$$

Let us now relate $\gamma(o)$ with W and L. Suppose that, for $i \in [L]$, there are W_i edges between layer i and previous layers. By the AM-GM inequality,

$$\gamma(o) \leq \prod_i (1+W_i) \leq \left(\sum_i \frac{1+W_i}{L}\right)^L \leq (2W/L)^L. \tag{4}$$

Combining Eqs. (3) and (4) gives the theorem. ■

Telgarsky (2016) showed how to construct a function f which satisfies $f(x) = (x \bmod 2)$ for $x \in \{0, 1, \ldots, 2^m - 1\}$ using a neural network with $O(m)$ layers and $O(m)$ parameters. By choosing $m = k^3$, Telgarsky showed that any function g computable by a neural network with $\Theta(k)$ layers and $O(2^k)$ nodes must necessarily have $\|f - g\|_1 > c$ for some constant $c > 0$.

Our theorem above implies a qualitatively similar statement. In particular, if we choose $m = k^{1+\varepsilon}$, then for any function g computable by a neural network with $\Theta(k)$ layers and $O(2^{k^\varepsilon})$ parameters, there must exist $x \in \{0, 1, \ldots, 2^m - 1\}$ such that $|f(x) - g(x)| > 1/2$.

4. Proof of Theorem 7

The proof of this theorem is very similar to the proof of the upper bound for piecewise polynomial networks from Bartlett et al. (1998, Theorem 1) but optimized in a few places. The main technical tool in the proof is a bound on the growth function of a polynomially parametrized function class, due to Goldberg and Jerrum (1995). It uses an argument involving counting the number of connected components of semi-algebraic sets. The form stated here is Bartlett et al. (1998, Lemma 1), which is a slight improvement of a result of Warren (1968) (the proof can be found in Anthony and Bartlett (1999, Theorem 8.3)).

Lemma 17 *Suppose $W \leq M$ and let $P_1, \ldots, P_M$ be polynomials of degree at most D in W variables. Define*

$$K := \left|\left\{(\operatorname{sgn}(P_1(a)), \ldots, \operatorname{sgn}(P_M(a))) : a \in \Re^W\right\}\right|,$$

i.e., K is the number of possible sign vectors attained by the polynomials. Then we have $K \leq 2(2eMD/W)^W$.

Proof [of Theorem 7]. For input $x \in \mathcal{X}$ and parameter vector $a \in \Re^W$, let $f(x, a)$ denote the output of the network. Then $\mathcal{F}$ is simply the class of functions $\{x \mapsto f(x, a) : a \in \Re^W\}$.

Fix $x_1, x_2, \ldots, x_m$ in $\mathcal{X}$. We view the parameters of the network, denoted a, as a collection of W real variables. We wish to bound

$$K := \left|\left\{(\operatorname{sgn}(f(x_1, a)), \ldots, \operatorname{sgn}(f(x_m, a))) : a \in \Re^W\right\}\right|.$$

In other words, K is the number of sign patterns that the neural network can output for the sequence of inputs $(x_1, \ldots, x_m)$. We will prove geometric upper bounds for K, which will imply upper bounds for $\Pi_{\operatorname{sgn}(\mathcal{F})}(m)$.

For any partition $\mathcal{S} = \{P_1, P_2, \ldots, P_N\}$ of the parameter domain $\Re^W$, clearly we have

$$K \leq \sum_{i=1}^N \left|\left\{(\operatorname{sgn}(f(x_1, a)), \ldots, \operatorname{sgn}(f(x_m, a))) : a \in P_i\right\}\right|. \tag{5}$$

We choose the partition in such a way that within each region P_i, the functions $f(x_j, \cdot)$ are all fixed polynomials of bounded degree, so that each term in this sum can be bounded via Lemma 17.

The partition is constructed iteratively layer by layer, through a sequence $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_{L-1}$ of successive refinements, with the following properties:

1. We have $|\mathcal{S}_0| = 1$ and, for each $n \in [L-1]$,
$$\frac{|\mathcal{S}_n|}{|\mathcal{S}_{n-1}|} \leq 2\left(\frac{2emk_np(1+(n-1)d^{n-1})}{W_n}\right)^{W_n}. \tag{6}$$
2. For each $n \in \{1, \ldots, L\}$, each element S of $\mathcal{S}_{n-1}$, each $j \in [m]$, and each unit u in the nth layer, when a varies in S, the net input to u is a fixed polynomial function in W_n variables of a, of total degree no more than $1 + (n-1)d^{n-1}$ (this polynomial may depend on S, j and u).

We may define $\mathcal{S}_0 = \{\Re^W\}$, which satisfies property 2 above, since the input to any node in layer 1 is of the form $w^T x_j + b$, which is an affine function of w, b.

Now suppose that $\mathcal{S}_0, \ldots, \mathcal{S}_{n-1}$ have been defined, and we want to define $\mathcal{S}_n$. For any $h \in [k_n], j \in [m]$, and $S \in \mathcal{S}_{n-1}$, let $p_{h,x_j,S}(a)$ denote the function describing the net input of the hth unit in the nth layer, in response to x_j, when $a \in S$. By the induction hypothesis this is a polynomial with total degree no more than $1 + (n-1)d^{n-1}$, and depends on at most W_n many variables.

Let $\{t_1, \ldots, t_p\}$ denote the set of breakpoints of the activation function. For any fixed $S \in \mathcal{S}_{n-1}$, by Lemma 17, the collection of polynomials

$$\left\{p_{h,x_j,S}(a) - t_i : h \in [k_n], j \in [m], i \in [p]\right\}$$

attains at most

$$\Pi := 2(2e(k_nmp)(1+(n-1)d^{n-1})/W_n)^{W_n}$$

distinct sign patterns when $a \in \Re^W$. Thus, one can partition $\Re^W$ into this many regions, such that all these polynomials have the same signs within each region. We intersect all these regions with S to obtain a partition of S into at most Π subregions. Performing this for all $S \in \mathcal{S}_{n-1}$ gives our desired partition $\mathcal{S}_n$. Thus, the required property 1 (inequality (6)) is clearly satisfied.

Fix some $S' \in \mathcal{S}_n$. Notice that, when a varies in S', all the polynomials

$$\left\{p_{h,x_j,S}(a) - t_i : h \in [k_n], j \in [m], i \in [p]\right\}$$

have the same sign, hence the *input* of each nth layer unit lies between two breakpoints of the activation function, hence the *output* of each nth layer unit in response to an x_j

is a fixed polynomial in W_n variables of degree no more than $d(1+(n-1)d^{n-1}) \le nd^n$. This implies that the *input* of every $(n+1)$th layer unit in response to an x_j is a fixed polynomial function of W_{n+1} variables of degree no more than $1+nd^n$. (When $d=0$, this affine function depends only on the W_{n+1} parameters in layer $n+1$; for $d>0$, it is a polynomial function of all parameters up to layer $n+1$.)

Proceeding in this way we obtain a partition $\mathcal{S}_{L-1}$ of $\Re^W$ such that for each $S \in \mathcal{S}_{L-1}$, the network output in response to any x_j is a fixed polynomial of a of degree no more than $1+(L-1)d^{L-1}$ (recall that the last node just outputs its input), and hence by Lemma 17 again,

$$|\{(\mathrm{sgn}(f(x_1,a)),\ldots,\mathrm{sgn}(f(x_m,a))) : a \in S\}| \le 2\left(\frac{2em(1+(L-1)d^{L-1})}{W_L}\right)^{W_L}.$$

On the other hand, applying (6) iteratively gives

$$|\mathcal{S}_{L-1}| \le \prod_{i=1}^{L-1} 2\left(\frac{2emk_ip(1+(i-1)d^{i-1})}{W_i}\right)^{W_i},$$

and thus using (5), and since the points $x_1,\ldots,x_m$ were chosen arbitrarily, we obtain

$$\begin{aligned}
\Pi_{\mathrm{sgn}(\mathcal{F})}(m) &\le \prod_{i=1}^{L} 2\left(\frac{2emk_ip(1+(i-1)d^{i-1})}{W_i}\right)^{W_i} \\
&\le 2^L\left(\frac{2emp\sum k_i(1+(i-1)d^{i-1})}{\sum W_i}\right)^{\sum W_i} && \text{(weighted AM-GM)} \\
&= 2^L\left(\frac{2empR}{\sum W_i}\right)^{\sum W_i} && \text{(definition of } R \text{ in (1))} \qquad (7)
\end{aligned}$$

For the bound on the VC-dimension, from the definition of VC-dimension we find

$$2^{\mathrm{VCdim}(\mathcal{F})} = \Pi_{\mathrm{sgn}(\mathcal{F})}(\mathrm{VCdim}(\mathcal{F})) \le 2^L\left(\frac{2epR\cdot\mathrm{VCdim}(\mathcal{F})}{\sum W_i}\right)^{\sum W_i}$$

Notice that $U>2$ implies $2eR \ge 16$, hence Lemma 18 below gives

$$\mathrm{VCdim}(\mathcal{F}) \le L + (\sum W_i)\log_2(4epR\log_2(2epR)) = O(\bar{L}W\log(pU) + \bar{L}LW\log d),$$

completing the proof. ■

Lemma 18 *Suppose that $2^m \le 2^t(mr/w)^w$ for some $r \ge 16$ and $m \ge w \ge t \ge 0$. Then, $m \le t + w\log_2(2r\log_2 r)$.*

Proof We would like to show that $2^x > 2^t(xr/w)^w$ for all $x > t + w\log_2(2r\log_2 r) =: m$. Let $f(x) := x - t - w\log_2(xr/w)$. To show that $f(x)>0$ for all $x>m$, we need only show that $f(m)\ge 0$ and $f'(x)>0$ for all $x \ge m$. First, $f(m)\ge 0$ if and only if

$$w\log_2(2r\log_2 r) - w\log_2(mr/w) \ge 0,$$

if and only if

$$(2r\log_2 r) - (mr/w) \ge 0,$$

if and only if

$$2\log_2 r - (t + w\log_2(2r\log_2 r))/w \ge 0,$$

if and only if

$$2\log_2 r - t/w - \log_2(2r\log_2 r) \ge 0,$$

if and only if

$$r^2/2\log_2 r \ge 2^{t/w},$$

which holds since $r \ge 16$ and $t/w \le 1$. Finally, for $x \ge m$, we have $f'(x) \ge 0$ if and only if

$$1 - w/(x\ln(2)) \ge 0$$

if and only if

$$x \ge w/\ln 2,$$

which holds since $r \ge 16$ implies $x \ge m \ge w\log_2(2r\log_2 r) > w/\ln 2$. ■

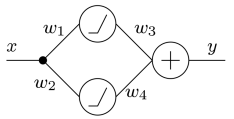
Proof [of Remark 9]. Note that $\sum k_i \le \sum W_i$ implies $L \le \sum W_i$, hence from (7) we have

$$\begin{aligned}
\Pi_{\mathrm{sgn}(\mathcal{F})}(m) &\le 2^L\left(\frac{2empR}{\sum W_i}\right)^{\sum W_i} \\
&\le \left(\frac{4emp(1+(L-1)d^{L-1})\sum k_i}{\sum W_i}\right)^{\sum W_i} && (L \le \textstyle\sum W_i) \\
&\le \left(4emp(1+(L-1)d^{L-1})\right)^{\sum W_i} && (\textstyle\sum k_i \le \sum W_i).
\end{aligned}$$

■

5. Proof of Theorem 10

The idea of the proof is that the sign of the output of a neural network can be expressed as a Boolean formula where each predicate is a polynomial inequality. For example, consider the following toy network, where the activation function of the hidden units is a ReLU.



The sign of the output of the network is $\mathrm{sgn}(y) = \mathrm{sgn}(w_3\sigma(w_1x) + w_4\sigma(w_2x))$. Define the following Boolean predicates: $p_1 = (w_1x > 0)$, $p_2 = (w_2x > 0)$, $q_1 = (w_3w_1x > 0)$, $q_2 = (w_4w_2x > 0)$, and $q_3 = (w_3w_1x + w_4w_2x > 0)$. Then, we can write

$$\mathrm{sgn}(y) = (\neg p_1 \wedge \neg p_2 \wedge 0) \vee (p_1 \wedge \neg p_2 \wedge q_1) \vee (\neg p_1 \wedge p_2 \wedge q_2) \vee (p_1 \wedge p_2 \wedge q_3).$$

A theorem of Goldberg and Jerrum states that any class of functions that can be expressed using a relatively small number of distinct polynomial inequalities has small VC-dimension.

Theorem 19 (Theorem 2.2 of Goldberg and Jerrum (1995)) *Let k, n be positive integers and $f : \Re^n \times \Re^k \to \{0,1\}$ be a function that can be expressed as a Boolean formula containing s distinct atomic predicates where each atomic predicate is a polynomial inequality or equality in $k+n$ variables of degree at most d. Let $\mathcal{F} = \{f(\cdot, w) : w \in \Re^k\}$. Then* $\mathrm{VCDim}(\mathcal{F}) \leq 2k \log_2(8eds)$.

Proof [of Theorem 10]. Consider a neural network with W weights and U computation units, and assume that the activation function ψ is piecewise polynomial of degree at most d with p pieces. To apply Theorem 19, we will express the sign of the output of the network as a Boolean function consisting of less than $2(1+p)^U$ atomic predicates, each being a polynomial inequality of degree at most $\max\{U+1, 2d^U\}$.

Since the neural network graph is acyclic, it can be topologically sorted. For $i \in [U]$, let u_i denote the ith computation unit in the topological ordering. The input to each computation unit u lies in one of the p pieces of ψ. For $i \in [U]$ and $j \in [p]$, we say "u_i is in state j" if the input to u_i lies in the jth piece.

For u_1 and any j, the predicate "u_1 is in state j" is a single atomic predicate which is the quadratic inequality indicating whether its input lies in the corresponding interval. So, the state of u_1 can be expressed as a function of p atomic predicates. Conditioned on u_1 being in a certain state, the state of u_2 can be determined using p atomic predicates, which are polynomial inequalities of degree at most $2d+1$. Consequently, the state of u_2 can be determined using $p + p^2$ atomic predicates, each of which is a polynomial of degree at most $2d+1$. Continuing similarly, we obtain that for each i, the state of u_i can be determined using $p(1+p)^{i-1}$ atomic predicates, each of which is a polynomial of degree at most $d^{i-1} + \sum_{j=0}^{i-1} d^j$. Consequently, the state of all nodes can be determined using less than $(1+p)^U$ atomic predicates, each of which is a polynomial of degree at most $d^{U-1} + \sum_{j=0}^{U-1} d^j \leq \max\{U+1, 2d^U\}$ (the output unit is linear). Conditioned on all nodes being in certain states, the sign of the output can be determined using one more atomic predicate, which is a polynomial inequality of degree at most $\max\{U+1, 2d^U\}$.

In total, we have less than $2(1+p)^U$ atomic polynomial-inequality predicates and each polynomial has degree at most $\max\{U+1, 2d^U\}$. Thus, by Theorem 19, we get an upper bound of $2W \log_2(16e \cdot \max\{U+1, 2d^U\} \cdot (1+p)^U) = O(WU \log((1+d)p))$ for the VC-dimension. ■

Acknowledgments

Christopher Liaw is supported by an NSERC graduate scholarship. Abbas Mehrabian is supported by an NSERC Postdoctoral Fellowship and a Simons-Berkeley Research Fellowship. Peter Bartlett gratefully acknowledges the support of the NSF through grant IIS-1619362 and of the Australian Research Council through an Australian Laureate Fellowship (FL110100281) and through the Australian Research Council Centre of Excellence for Mathematical and Statistical Frontiers (ACEMS). Part of this work was done while Peter Bartlett and Abbas Mehrabian were visiting the Simons Institute for the Theory of Computing at UC Berkeley.

References

Martin Anthony and Peter Bartlett. *Neural network learning: theoretical foundations*. Cambridge University Press, 1999.

Peter Bartlett, Vitaly Maiorov, and Ron Meir. Almost linear VC-dimension bounds for piecewise polynomial networks. *Neural Computation*, 10(8):2159–2173, Nov 1998.

Eric B. Baum and David Haussler. What size net gives valid generalization? *Neural Computation*, 1(1):151–160, 1989.

Anselm Blumer, A. Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, 36(4):929–965, October 1989. ISSN 0004-5411. doi: 10.1145/76359.76371. URL `http://doi.acm.org/10.1145/76359.76371`. (Conference version in STOC'86).

Nadav Cohen, Or Sharir, and Amnon Shashua. On the expressive power of deep learning: A tensor analysis. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 698–728, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR. URL `http://proceedings.mlr.press/v49/cohen16.html`.

Thomas M. Cover. Capacity problems for linear machines. In L. Kanal, editor, *Pattern Recognition*, pages 283–289. Thompson Book Co., 1968.

Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 907–940, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR. URL `http://proceedings.mlr.press/v49/eldan16.html`.

Paul W. Goldberg and Mark R. Jerrum. Bounding the Vapnik-Chervonenkis dimension of concept classes parameterized by real numbers. *Machine Learning*, 18(2):131–148, 1995. (Conference version in COLT'93).

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Philipp Grohs, Dmytro Perekrestenko, Dennis Elbrächter, and Helmut Bölcskei. Deep neural network approximation theory. *arXiv preprint arXiv:1901.02220*, 2019.

Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

Daniel Jakubovitz, Raja Giryes, and Miguel RD Rodrigues. Generalization error in deep learning. *arXiv preprint arXiv:1808.01174*, 2018.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015.

Shyu Liang and R. Srikant. Why deep neural networks for function approximation?, 2017. arXiv:1610.04161.

Wolfgang Maass. Neural nets with superlinear VC-dimension. *Neural Computation*, 6(5): 877–884, Sept 1994.

David Pollard. *Empirical Processes: Theory and Applications*, volume 2. Institute of Mathematical Statistics, 1990.

Itay Safran and Ohad Shamir. Depth-width tradeoffs in approximating natural functions with neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2979–2987, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL `http://proceedings.mlr.press/v70/safran17a.html`.

Eduardo D. Sontag. Feedback stabilization using two-hidden-layer nets. *IEEE Transactions on Neural Networks*, 3(6):981–990, 1992.

Matus Telgarsky. benefits of depth in neural networks. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 1517–1539, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR. URL `http://proceedings.mlr.press/v49/telgarsky16.html`.

V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971. doi: 10.1137/1116025.

Hugh E. Warren. Lower bounds for approximation by nonlinear manifolds. *Transactions of the American Mathematical Society*, 133(1):167–178, 1968.

Dmitry Yarotsky. Error bounds for approximations with deep relu networks. *Neural Networks*, 94:103–114, 2017.

Journal of Machine Learning Research 20 (2019) 1-32 Submitted 10/17; Revised 6/18; Published 4/19

A Representer Theorem for Deep Kernel Learning

Bastian Bohn bohn@ins.uni-bonn.de
Christian Rieger rieger@ins.uni-bonn.de
Institute for Numerical Simulation, University of Bonn
Endenicher Allee 19b, 53115 Bonn, Germany
Michael Griebel griebel@ins.uni-bonn.de
Institute for Numerical Simulation, University of Bonn
Endenicher Allee 19b, 53115 Bonn, Germany,
Fraunhofer Center for Machine Learning
Fraunhofer Institute for Algorithms and Scientific Computing SCAI
Schloss Birlinghoven, 53754 Sankt Augustin, Germany

Editor: Massimiliano Pontil

Abstract

In this paper we provide a finite-sample and an infinite-sample representer theorem for the concatenation of (linear combinations of) kernel functions of reproducing kernel Hilbert spaces. These results serve as mathematical foundation for the analysis of machine learning algorithms based on compositions of functions. As a direct consequence in the finite-sample case, the corresponding infinite-dimensional minimization problems can be recast into (nonlinear) finite-dimensional minimization problems, which can be tackled with nonlinear optimization algorithms. Moreover, we show how concatenated machine learning problems can be reformulated as neural networks and how our representer theorem applies to a broad class of state-of-the-art deep learning methods.

Keywords: deep kernel learning, representer theorem, artificial neural networks, multi-layer kernel, regularized least-squares regression

1. Introduction

The interpolation or regression of given function values is one of the main tasks in modern data mining and machine learning applications. Due to the famous representer theorem for empirical risk minimization in reproducing kernel Hilbert spaces (RKHS), see e.g. Schölkopf and Smola (2002); Steinwart and Christmann (2008); Kimeldorf and Wahba (1970), various algorithms based on finite linear combinations of kernel translates have gained much popularity in the last decade, like, for example, support vector machines (SVMs) and Tikhonov-regularized least-squares in RKHS. In general, these methods work very well if the underlying problem fits the chosen reproducing kernel space H, e.g. if the given input values stem from a function $g \in H$. However, if H contains for instance only smooth functions but g has a kink or a jump, the interpolant or regressor, respectively, in H might not represent a good approximation to the true function g anymore. Then, if it is not known how to choose an appropriate kernel K of H a priorily, one usually relies on so-called multiple kernel learning (MKL) algorithms, which try to determine the optimal kernel adaptively, see e.g. Bach et al. (2004). But while most of these methods allow to learn a suitable kernel by simply constructing a linear or convex combination of a given set of input kernels, they still do not achieve considerably better results than standard a-priori kernel choices for many applications, see Gönen and Alpaydin (2011).

In recent years, promising new variants of kernel learning methods, namely deep kernel learning and multi-layer-MKL (MLMKL) algorithms have been developed. They have proven to be very successful in regression and classification tasks. Here, motivated by multi-layer feed-forward neural networks, a kernel function is concatenated with one or more nonlinear functions in order to achieve a highly flexible new kernel function, see e.g. Cho and Saul (2009); Damianou and Lawrence (2013); Rebai et al. (2016); Strobl and Visweswaran (2013); Wilson et al. (2016); Zhuang et al. (2011). The main idea behind this approach is to combine the flexibility of deep neural networks, in which the feature detection in the data set is done completely automatically, with the approximation power of kernel methods, in which a feature map is determined by the chosen kernel. This way, the neural network architecture learns the optimal kernel that best represents important features of the data for the task at hand. While first steps towards creating a mathematical framework to analyze deep neural networks—especially for image classification tasks—have been made in e.g. Mallat (2016); Mhaskar et al. (2017); Montavon et al. (2017), deep approximation theory for kernel based approaches is still missing at large. Moreover, the underlying nonlinear minimization problem is usually tackled by simple gradient descent and heuristic backpropagation algorithms without a thorough theoretical analysis of its properties. An initial cornerstone for the analysis of chained kernel approximations has been provided by Dinuzzo (2011), where two-layer kernel networks were considered and their relation to MKL was established. However, an analysis of deeper kernel networks and their connection to MLMKL has not been considered so far.

In this paper, we consider the problem of optimal concatenated approximation in reproducing kernel Hilbert spaces, which will directly lead to a variant of multi-layer kernel learning problems and will extend the results achieved in Dinuzzo (2011). For this class, we will prove a representer theorem, which allows us to reduce the nonlinear, potentially infinite-dimensional optimization problem to a finite-dimensional one. Consequently, standard nonlinear optimization techniques can be used to tackle this problem. At least to our knowledge, this is the first derivation of a representer theorem for concatenated function approximation in the literature. It is also valid for certain types of hidden layer neural networks and deep SVMs.

The remainder of this paper is organized as follows: In Section 2, we briefly review the interpolation and the regression problem in an (possibly infinite-dimensional) RKHS and discuss how the classical representer theorem allows to recast these problems into finite-dimensional linear equation systems. In Section 3, we introduce the optimal *concatenated* approximation problem for arbitrary loss functions and regularizers. We derive a representer theorem for this problem in the multi-layer case and discuss its relation to deep learning and multi-layer multiple kernel learning methods. Furthermore, we exemplarily derive algorithms for interpolation and least-squares regression in the two-layer case from it. The latter will be a natural generalization of the RLS2 method developed in Dinuzzo (2011), which only deals with a linear outer kernel. Section 4 illustrates the application of our concatenated interpolation and regression algorithms to two simple examples and serves as a proof of concept. Finally, we conclude with a summary and an outlook in Section 5.

©2019 Bastian Bohn, Michael Griebel and Christian Rieger.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-621.html.

2. Interpolation and regression in reproducing kernel Hilbert spaces

In this section we shortly review interpolation and least-squares regression problems, respectively, in an RKHS. To this end, we consider the standard representer theorem and show how it helps to find an interpolant/regressor.

2.1. Interpolation

Let $\Omega \subset \mathbb{R}^d$ be an open domain and let the pairwise disjoint points $X := \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\} \subset \Omega$ and the values $Y := \{y_1, \dots, y_N\} \subset \mathbb{R}$ be given. Let furthermore $H := H(\Omega, \mathbb{R})$ be a reproducing kernel Hilbert space of real-valued functions on Ω. The minimal norm interpolant is

$$f^*_{X,Y} := \arg\min_{f \in H} \|f\|_H \quad \text{such that} \quad f(\boldsymbol{x}_i) = y_i \quad \forall\, i = 1, \dots, N. \tag{1}$$

The classical representer theorem, see e.g. Schölkopf and Smola (2002); Steinwart and Christmann (2008) for scalar-valued functions and Micchelli and Pontil (2005) for vector-valued functions, now states that $f^*_{X,Y}$ can be written as a *finite* linear combination of kernel evaluations in the data, namely

$$f^*_{X,Y}(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i^* K(\boldsymbol{x}_i, \boldsymbol{x}), \tag{2}$$

where $K : \Omega \times \Omega \to \mathbb{R}$ denotes the reproducing kernel of H and $\alpha_i^* \in \mathbb{R}, i = 1, \dots, N$, are the corresponding coefficients. For details on RKHS, see Aronszajn (1950). Therefore, the solution to the possibly infinite-dimensional optimization problem (1) resides in the N-dimensional span of the functions $K(\boldsymbol{x}_i, \cdot), i = 1, \dots, N$. To compute the coefficients, we simply have to solve the system

$$\boldsymbol{M}_{X,X} \boldsymbol{\alpha}^* = \boldsymbol{y} \tag{3}$$

of linear equations with

$$\boldsymbol{M}_{X,X} := \begin{pmatrix} K(\boldsymbol{x}_1, \boldsymbol{x}_1) & \dots & K(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ K(\boldsymbol{x}_N, \boldsymbol{x}_1) & \dots & K(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{pmatrix}, \quad \boldsymbol{\alpha}^* := \begin{pmatrix} \alpha_1^* \\ \vdots \\ \alpha_N^* \end{pmatrix} \quad \text{and } \boldsymbol{y} := \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}. \tag{4}$$

Note that this $N \times N$ system admits a unique solution if the kernel K is strictly positive definite. For example, for Sobolev kernels it can be shown that the condition number of the system matrix $\boldsymbol{M}_{X,X}$ only grows moderately with the size N provided that the data points are quasi-uniformly distributed, see de Marchi and Schaback (2010). Moreover, for infinitely smooth kernel functions (e.g. Gaussian kernels or multiquadrics) it can be necessary to perform an appropriate basis change before solving the above equation system, see e.g. Wendland (2005).

2.2. Least-squares regression

In real-world applications, the values $y_i, i = 1, \dots, N$ are usually not exactly given, but are perturbed by some noise term. Therefore, a direct interpolation might no longer be appropriate. In this case, one considers the corresponding regularized least-squares regression

problem

$$f^\lambda_{X,Y} := \arg\min_{f \in H} \lambda \|f\|_H^2 + \sum_{j=1}^{N} |f(\boldsymbol{x}_i) - y_i|^2, \tag{5}$$

where the side condition in (1) is substituted by a penalty term. Here, the Lagrange multiplier λ weights the importance of the norm minimization against the function evaluation error. Again, the representer theorem Micchelli and Pontil (2005); Schölkopf and Smola (2002); Steinwart and Christmann (2008) tells us that $f^\lambda_{X,Y}$ is of the form (2), i.e.

$$f^\lambda_{X,Y}(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i^\lambda K(\boldsymbol{x}_i, \boldsymbol{x}).$$

This time the coefficients $\alpha_i^\lambda, i = 1, \dots, N$, are determined by

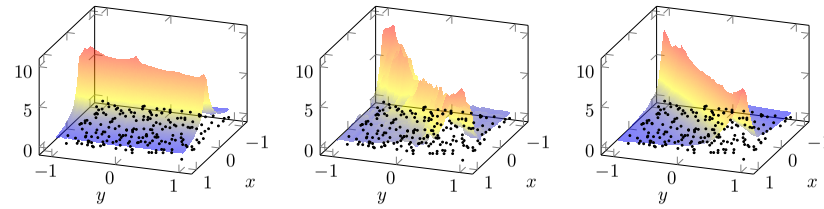
$$(\boldsymbol{M}_{X,X} + \lambda \boldsymbol{I})\, \boldsymbol{\alpha}^\lambda = \boldsymbol{y}, \tag{6}$$

where $\boldsymbol{I}$ denotes the $N \times N$ identity matrix. The size of the Lagrange parameter $\lambda > 0$ now also influences the condition number of the system matrix, i.e. the larger λ is, the smaller the condition number becomes.

3. Interpolation and regression with compositions of reproducing kernel Hilbert spaces

As already mentioned in the introduction, the standard interpolation and regression algorithms in RKHS work well if the samples y_i are (perturbed) evaluations of a function $g \in H$, where the reproducing kernel space H is known in the first place. However, if the appropriate RKHS H is unknown, it is advisable to resort to multiple kernel learning methods or multi-layer multiple kernel learning methods.

We now explain this aspect in more detail and, to this end, motivate a first two-dimensional, two-layer approach with an example: Let the kernel K of H be a tensor-product of two univariate Matérn Sobolev kernels of order one on $\mathbb{R}$, see Section 4 for a definition of this kernel. The corresponding function space H is often also called Sobolev space of "mixed smoothness" of order one and it is of special importance for e.g. sparse grid discretizations, see Bungartz and Griebel (2004), and quasi Monte Carlo quadrature, see Hinrichs et al. (2016). Now, let us consider the continuous function $g_1(x, y) := (0.1 + |x|)^{-1}$, which has a kink that is perpendicular to the x-axis. It can easily be shown that $g_1 \in H$ and, therefore, the interpolant of g_1 by a function from H resembles a good approximation to g_1, see Figure 1(a). If we now look at $g_2(x, y) := (0.1 + |x - y|)^{-1}$, which has a kink along the diagonal with $x = y$, then $g_2 \notin H$. Therefore, the interpolant of g_2 by a function in H is a rather bad approximation to g_2. This can be seen in Figure 1(b). However, if we let R^{-1} be a rotation by 45°, then $g_2 \circ R^{-1} \in H$ would have an axis-aligned kink like g_1. To use this fact when interpolating g_2, we can simply look for the best interpolant in $\{f \circ R \mid f \in H\}$ in (1) instead of $f \in H$. This example is illustrated in Figure 1(c). As we can see, the interpolant in Figure 1(c) is a much better representative for g_2 than the one in Figure 1(b). This example illustrates that, already in the very simple case of employing a concatenation with a rotation, a two-layer approach can be a good choice to overcome



(a) $y_i = g_1(\boldsymbol{x}_i), f \in H$ (b) $y_i = g_2(\boldsymbol{x}_i), f \in H$ (c) $y_i = g_2(\boldsymbol{x}_i), f \in \{h \circ R \mid h \in H\}$

Figure 1: Solutions to (1) in the two-variate tensor-product Matérn-kernel Sobolev space H of order one, see also Fasshauer and Ye (2011), with 200 uniform samples $\boldsymbol{x}_i, i = 1, \ldots, 200$ (marked in black), shown in the domain $[-1, 1]^2$. (a) depicts the solution $f \in H$ for values y_i sampled from g_1, whereas (b) shows the optimal solution for y_i sampled from g_2. (c) presents the best interpolant of type $f \circ R$, where $f \in H$ and R is a rotation by 45° for y_i sampled from g_2. For reasons of comparability, we restricted our representation to $[-1, 1]^2$ here, although some data points were mapped outside of this domain by applying the rotation R and the kernel was defined on the whole $\mathbb{R}^2$.

the restrictions of a standard kernel learning algorithm. Let us remark that already this motivating example exhibits a fundamentally different setting from the one considered in Dinuzzo (2011) because of the nonlinearity of the outer kernel. While the RLS2 algorithm introduced there can be interpreted as an MKL variant, where a convex combination of given kernel functions is computed, we are looking for an inner function, which transforms the domain in such a way that it is optimal for the (possibly nonlinear) outer kernel.

Now, instead of just considering one layer of simple rotations as in the above example, we allow for a fully flexible multi-layer kernel learning approach, where we employ arbitrary functions from reproducing kernel Hilbert spaces in each layer. This approach can successfully deal with a much broader class of interpolation and regression problems, see also Rebai et al. (2016); Zhuang et al. (2011). To this end, we consider *concatenated* machine learning problems. We introduce a new representer theorem for the case of multiple concatenations of functions from RKHS, which allows us to derive the related, finite-dimensional, nonlinear optimization problem.

3.1. A representer theorem for concatenated kernel learning

In this section, we show how a *concatenated representer theorem* can be derived for a very general class of problem types and an arbitrary number $L \in \mathbb{N}$ of concatenations. For more details on vector-valued reproducing kernel Hilbert spaces, we refer the reader to Micchelli and Pontil (2005). For a two-layer variant of this theorem, we refer to Dinuzzo (2011).

Theorem 1 *Let $\mathcal{H}_1, \ldots, \mathcal{H}_L$ be reproducing kernel Hilbert spaces of functions with finite-dimensional domains D_l and ranges $R_l \subseteq \mathbb{R}^{d_l}$ with $d_l \in \mathbb{N}$ for $l = 1, \ldots, L$ such that $R_l \subseteq D_{l-1}$ for $l = 2, \ldots, L$, $D_L = \Omega$ and $R_1 \subseteq \mathbb{R}$. Let furthermore $\mathcal{L} : \mathbb{R}^2 \to [0, \infty]$ be an arbitrary loss function and let $\Theta_1, \ldots, \Theta_L : [0, \infty) \to [0, \infty)$ be strictly monotonically increasing functions. Then, a set of minimizers $(f_l)_{l=1}^L$ with $f_l \in \mathcal{H}_l$ of*

$$J(f_1, \ldots, f_L) \coloneqq \sum_{i=1}^{N} \mathcal{L}\left(y_i, f_1 \circ \ldots \circ f_L(\boldsymbol{x}_i)\right) + \sum_{l=1}^{L} \Theta_l\left(\|f_l\|_{\mathcal{H}_l}^2\right) \tag{7}$$

fulfills $f_l \in \tilde{V}_l \subset \mathcal{H}_l$ for all $l = 1, \ldots, L$ with

$$\tilde{V}_l = \operatorname{span}\left\{K_l\left(f_{l+1} \circ \ldots \circ f_L\left(\boldsymbol{x}_i\right), \cdot\right) \boldsymbol{e}_{k_l} \mid i = 1, \ldots, N \text{ and } k_l = 1, \ldots, d_l\right\},$$

where K_l denotes the reproducing kernel of $\mathcal{H}_l$ and $\boldsymbol{e}_{k_l} \in \mathbb{R}^{d_l}$ is the k_l-th unit vector.

Proof We denote by $\Pi_{\tilde{V}_l}$ and $\Pi_{\tilde{V}_l}^{\perp}$ the projector onto $\tilde{V}_l$ and its orthogonal complement in $\mathcal{H}_l$, respectively, for $l = 1, \ldots, L$. First, we note that

$$\begin{aligned}
f_l \circ f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i) &= \sum_{k=1}^{d_l} \left(\Pi_{\tilde{V}_l}\left(f_l\right) + \Pi_{\tilde{V}_l^{\perp}}\left(f_l\right), K_l\left(f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i), \cdot\right) \boldsymbol{e}_k\right)_{\mathcal{H}_l} \cdot \boldsymbol{e}_k \\
&= \sum_{k=1}^{d_l} \left(\Pi_{\tilde{V}_l}\left(f_l\right), K_l\left(f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i), \cdot\right) \boldsymbol{e}_k\right)_{\mathcal{H}_l} \cdot \boldsymbol{e}_k \\
&= \sum_{k=1}^{d_l} \left(\boldsymbol{e}_k^T \Pi_{\tilde{V}_l}\left(f_l\right)\left(f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i)\right)\right) \cdot \boldsymbol{e}_k \\
&= \Pi_{\tilde{V}_l}\left(f_l\right)\left(f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i)\right)
\end{aligned}$$

for all $i = 1, \ldots, N$ and $l = 1, \ldots, L$. Since this holds for each function in the chain, we can iterate this process to obtain

$$f_l \circ f_{l+1} \circ \ldots \circ f_L(\boldsymbol{x}_i) = \Pi_{\tilde{V}_l}\left(f_l\right) \circ \Pi_{\tilde{V}_{l+1}}\left(f_{l+1}\right) \circ \ldots \circ \Pi_{\tilde{V}_L}\left(f_L\right)\left(\boldsymbol{x}_i\right) \tag{8}$$

for each $l = 1, \ldots, L$. Therefore, we have

$$\begin{aligned}
J(f_1, \ldots, f_L) = &\sum_{i=1}^{N} \mathcal{L}\left(y_i, \Pi_{\tilde{V}_1}(f_1) \circ \ldots \circ \Pi_{\tilde{V}_L}(f_L)(\boldsymbol{x}_i)\right) \\
&+ \sum_{l=1}^{L} \Theta_l\left(\|\Pi_{\tilde{V}_l}(f_l)\|_{\mathcal{H}_l}^2 + \|\Pi_{\tilde{V}_l^{\perp}}(f_l)\|_{\mathcal{H}_l}^2\right) \geq J(\Pi_{\tilde{V}_1}(f_1), \ldots, \Pi_{\tilde{V}_L}(f_L))
\end{aligned}$$

and equality only holds if $f_l \in \tilde{V}_l$ for each $l = 1, \ldots, L$ because of the strict monotonicity of each Θ_l. This completes the proof. ■

Note that theorem 1 also holds for

$$J(f_1, \ldots, f_L) \coloneqq \mathcal{L}\left(y_1, f_1 \circ \ldots \circ f_L(\boldsymbol{x}_1), \ldots, y_N, f_1 \circ \ldots f_L(\boldsymbol{x}_N)\right) + \sum_{l=1}^{L} \Theta_l\left(\|f_l\|_{\mathcal{H}_l}^2\right)$$

with arbitrary loss $\mathcal{L} : \left(\mathbb{R}^2\right)^N \to [0, \infty]$. However, the version we proved above is more consistent with the remainder of this paper. Furthermore, because of (8), we could also

state an even more general version of Theorem 1 where the loss function $\mathcal{L}$ not only depends on the point evaluations $f_1 \circ \ldots \circ f_L(\boldsymbol{x}_i)$ for $i = 1, \ldots, N$, but also on the intermediate values $f_l \circ \ldots \circ f_L(\boldsymbol{x}_i)$ for any $l = 2, \ldots, L$. However, for the sake of readability, we proceed with (7). Theorem 1 now states that

$$(f_1, \ldots, f_L) = \underset{\substack{\bar{f}_l \in \mathcal{H}_l \\ l=1,\ldots,L}}{\arg\min}\, J(\bar{f}_1, \ldots, \bar{f}_L) = \underset{\substack{\bar{f}_l \in \tilde{V}_l \\ l=1,\ldots,L}}{\arg\min}\, J(\bar{f}_1, \ldots, \bar{f}_L) \tag{9}$$

with J from (7). This means that the (possibly) infinite-dimensional optimization problem

$$\underset{\substack{\bar{f}_l \in \mathcal{H}_l \\ l=1,\ldots,L}}{\arg\min}\, J(\bar{f}_1, \ldots, \bar{f}_L)$$

can be recast into the finite-dimensional optimization problem

$$\underset{\substack{\bar{f}_l \in \tilde{V}_l \\ l=1,\ldots,L}}{\arg\min}\, J(\bar{f}_1, \ldots, \bar{f}_L).$$

In this way, our representer theorem is a direct extension of the classical representer theorem, see Section 2 and Schölkopf and Smola (2002), to concatenated functions. We obtain that the solution to (9) is given by a linear combination of at most N basis functions in each layer. Therefore, the overall number of degrees of freedom in the underlying optimization problem (9) is given by

$$\#\text{dof} = \sum_{l=1}^{L} \dim\left(\tilde{V}_l\right) = \sum_{l=1}^{L} N \cdot d_l = N \cdot \left(1 + \sum_{l=2}^{L} d_l\right).$$

According to Theorem 1, we can write f_1 as

$$f_1(\cdot) = \sum_{j=1}^{N} \alpha_j K_1\left(f_2 \circ \ldots \circ f_L(\boldsymbol{x}_j), \cdot\right)$$

for some coefficients $\alpha_j \in \mathbb{R}$. Therefore, the concatenated function $h(\cdot) = f_1 \circ \ldots \circ f_L(\cdot)$, which we are interested in, can be expressed as

$$h(\cdot) = \sum_{j=1}^{N} \alpha_j \mathcal{K}^L(\boldsymbol{x}_j, \cdot)$$

with the deep kernel

$$\mathcal{K}^L(\boldsymbol{x}, \boldsymbol{y}) = K_1\left(f_2 \circ \ldots \circ f_L(\boldsymbol{x}), f_2 \circ \ldots \circ f_L(\boldsymbol{y})\right). \tag{10}$$

Due to the definition of $\tilde{V}_l$ for $l = 1, \ldots, L$, the corresponding f_l is defined recursively. In general, it is thus not possible to simply write down a closed formula for $\mathcal{K}^L$ for arbitrary

L. To illustrate the structure of the kernel $\mathcal{K}^L$, we therefore consider a two-layer example with $L = 2$ in the following. In this case, we obtain $\tilde{V}_2 = \text{span}\{K_2(\boldsymbol{x}_i, \cdot)\boldsymbol{e}_{k_2} \mid i = 1, \ldots, N \text{ and } k_2 = 1, \ldots, d_2\}$. From Theorem 1, we know that

$$f_2(\cdot) = \sum_{i=1}^{N} \sum_{k_2=1}^{d_2} c_{i,k_2} K_2(\boldsymbol{x}_i, \cdot)\boldsymbol{e}_{k_2}$$

for certain coefficients $c_{i,k_2} \in \mathbb{R}$. Furthermore, we have that $f_1 \in \tilde{V}_1 = \text{span}\{K_1(f_2(\boldsymbol{x}_i), \cdot) \mid i = 1, \ldots, N\}$ and thus

$$f_1(\cdot) = \sum_{j=1}^{N} \alpha_j K_1\left(\sum_{i=1}^{N} \sum_{k_2=1}^{d_2} c_{i,k_2} K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)\boldsymbol{e}_{k_2}, \cdot\right).$$

The concatenated function is then given by $h(\cdot) := f_1 \circ f_2(\cdot) = \sum_{j=1}^{N} \alpha_j \mathcal{K}^2(\boldsymbol{x}_j, \cdot)$ with the composition kernel

$$\mathcal{K}^2(\boldsymbol{x}, \boldsymbol{y}) = K_1\left(\sum_{i=1}^{N} \sum_{k_2=1}^{d_2} c_{i,k_2} K_2(\boldsymbol{x}_i, \boldsymbol{x})\boldsymbol{e}_{k_2}, \sum_{i=1}^{N} \sum_{k_2=1}^{d_2} c_{i,k_2} K_2(\boldsymbol{x}_i, \boldsymbol{y})\boldsymbol{e}_{k_2}\right). \tag{11}$$

Therefore, instead of considering the infinite-dimensional optimization problem of finding $f_1 \in \mathcal{H}_1$ and $f_2 \in \mathcal{H}_2$ that minimize

$$J(f_1, f_2) = \sum_{i=1}^{N} \mathcal{L}\left(y_i, f_1(f_2(\boldsymbol{x}_i))\right) + \Theta_1\left(\|f_1\|_{\mathcal{H}_1}^2\right) + \Theta_2\left(\|f_2\|_{\mathcal{H}_2}^2\right),$$

we can restrict ourselves to finding the $N + N \cdot d_2$ coefficients α_j, c_{i,k_2} for $i, j = 1, \ldots, N$ and $k_2 = 1, \ldots, d_2$.

Note at this point that the problem of finding these coefficients is highly nonlinear and becomes more complicated for a larger number of layers L. While the corresponding problem of optimizing the outermost coefficients, i.e. α_j for $j = 1, \ldots, N$ in our example, is still convex if the loss $\mathcal{L}$ and the penalty terms Θ_1, Θ_2 are convex, the optimization of the inner coefficients, i.e. c_{i,k_2} for $i = 1, \ldots, N$ and $k_2 = 1, \ldots, d_2$, is usually not convex anymore and can have many local minima. Here, finding a global minimum is an issue because standard (iterative) optimization methods strongly depend on the chosen initial value and usually just deliver some local minimum.

If the optimization functional J is smooth, one can rely on a Newton-type minimizer such as BFGS to solve the underlying optimization problem. However, if one deals with nonsmooth loss functionals or penalty terms, one should resort to specifically designed stochastic gradient algorithms which fit the problem at hand, see e.g. Reddi et al. (2016).

It remains to note that our representer theorem covers much more than just interpolation or least-squares regression algorithms. In the same fashion as the standard representer theorem in Schölkopf and Smola (2002), it can directly be applied to more involved settings such as regression with a concatenation of support vector machines for instance. To this end, just choose $\mathcal{L}$ to be the ε-insensitive loss function and $\Theta_1(x) = \ldots = \Theta_L(x) = x$. Furthermore, the choice of the additive penalties $\Theta_1, \ldots, \Theta_L$ in (7) is rather arbitrary and one could think of more complex interactions between the penalties for each function $f_l, l = 1, \ldots, L$, as long as the arguments in the proof of Theorem 1 remain valid.

3.2. An infinite-sample representer theorem for concatenated kernel learning

After deriving the representer theorem 1 for the case of multi-layer kernel approximations, we now extend our results to the case of infinitely many samples. This has to be understood in analogy to the results in chapter 5 of Steinwart and Christmann (2008), where such an infinite-sample representer theorem is provided for the single-layer case. Although such a result can usually not directly be applied to a practical problem unless the distribution of the data points is known, it can serve as a cornerstone for the analysis of robustness with respect to a measure change and can lead to a-priori convergence results, see Steinwart and Christmann (2008). We will restrict the loss function to be an L-times differentiable Nemitski loss for the following theorem. For a definition, we refer to Steinwart and Christmann (2008) or our appendix, where we define an even more general type of *Nemitski vector loss*. Note that, when we refer to convexity or differentiability of Nemitski losses or reproducing kernels, this should always be understood with respect to the second argument, i.e. $\mathrm{d}K(\boldsymbol{x},\boldsymbol{z})$ should be understood as $\frac{\partial}{\partial \boldsymbol{z}}K(\boldsymbol{x},\boldsymbol{z})$. In the following, we denote by $\mathcal{B}(X,Y)$ the space of bounded linear operators from X to Y, endowed with the standard operator norm.

Theorem 2 *Let $\mathcal{H}_1,\dots,\mathcal{H}_L$ and the domains and ranges of their elements be as in theorem 1 and let $\lambda_1,\dots,\lambda_L>0$. Let, furthermore, the kernel K_l of $\mathcal{H}_l$ fulfill $K_l\in C^1(D_l\times D_l)$ together with*

$$\sup_{\boldsymbol{x}\in D_l}\|K_l(\boldsymbol{x},\boldsymbol{x})\|_2\leq c_l \quad \text{and} \quad \sup_{\boldsymbol{x},\boldsymbol{z}\in D_l}\|\mathrm{d}K_l(\boldsymbol{x},\boldsymbol{z})\|_{\mathcal{B}\left(D_l,\mathbb{R}^{d_l\times d_l}\right)}\leq c_l \tag{12}$$

for some $c_l<\infty$ and all $l=1,\dots,L$. Let $\mathbb{P}$ be a distribution on $\Omega\times R_1$ and let $\mathcal{L}:R_1\times\mathbb{R}\to[0,\infty)$ be a convex, $\mathbb{P}$-integrable and 1-times differentiable (w.r.t. the second variable) Nemitski loss such that the absolute value of the derivative is also a $\mathbb{P}$-integrable Nemitski loss, which fulfills

$$\left|\mathcal{L}^{(k)}(y,z)\right|\leq b_k(y)+h_k(|z|) \quad \text{for all } (y,z)\in R_1\times\mathbb{R}$$

for some $L_{1,\mathbb{P}_{R_1}}$-integrable[1] $b_k:R_1\to[0,\infty)$ and some increasing $h_k:[0,\infty)\to[0,\infty)$ for $k=0,1$. Then, if we assume that a set of minimizers $(f_l)_{l=1}^L$ with $f_l\in\mathcal{H}_l$ of

$$J(f_1,\dots,f_L):=\int_{\Omega\times R_1}\mathcal{L}\left(y,f_1\circ\ldots\circ f_L(\boldsymbol{x})\right)\,\mathrm{d}\mathbb{P}(\boldsymbol{x},y)+\sum_{l=1}^{L}\lambda_l\|f_l\|_{\mathcal{H}_l}^2 \tag{13}$$

exists, it fulfills the Bochner-type integral equation

$$f_l(\cdot)=-\frac{1}{2\lambda_i}\int_{\Omega\times R_1}K_l\left(\cdot,f_{l+1}\circ\ldots\circ f_L(\boldsymbol{x})\right)A_{f_l,f_{l+1},\dots,f_L}(\boldsymbol{x},y)\,\mathrm{d}\mathbb{P}(\boldsymbol{x},y) \tag{14}$$

for some $A_{f_l,f_{l+1},\dots,f_L}\in L_{1,\mathbb{P}}(\Omega\times R_1;R_l)$ for all $l=1,\dots,L$.

Proof The proof works layer-wise and it is an extension of the proof of theorem 5.8 of Steinwart and Christmann (2008) to the multi-layer case and to Nemitski vector loss functions, see also definition 5. Let $g_i\in\mathcal{H}_i$ be arbitrary for all $i=1,\dots,L$. Let $G_1:\Omega\times R_1\to R_2\times R_1$ be defined by $G_1(\boldsymbol{x},y)=(g_2\circ\ldots\circ g_L(\boldsymbol{x}),y)$. Obviously, G_1 is a measurable map and we can define the pushforward $G_{1,\star}(\mathbb{P})$ of $\mathbb{P}$ onto $R_2\times R_1$. With this we obtain

$$\int_{\Omega\times R_1}\mathcal{L}\left(y,g_1\circ\ldots\circ g_L(\boldsymbol{x})\right)\,\mathrm{d}\mathbb{P}(\boldsymbol{x},y)=\int_{R_2\times R_1}\mathcal{L}\left(y,g_1(\boldsymbol{\xi})\right)\,\mathrm{d}G_{1,\star}(\mathbb{P})(\boldsymbol{\xi},y).$$

Now, with the functional $J_{g_2,\dots,g_L}:\mathcal{H}_1\to[0,\infty)$ defined by

$$J_{g_2,\dots,g_L}(g_1):=\int_{R_2\times R_1}\mathcal{L}\left(y,g_1(\boldsymbol{\xi})\right)\,\mathrm{d}G_{1,\star}(\mathbb{P})(\boldsymbol{\xi},y)+\lambda_1\|g_1\|_{\mathcal{H}_1}^2,$$

we can reformulate the minimization problem as

$$\min_{g_1\in\mathcal{H}_1,\dots,g_L\in\mathcal{H}_L}J(g_1,\dots,g_L)=\min_{g_2\in\mathcal{H}_2,\dots,g_L\in\mathcal{H}_L}\left(\min_{g_1\in\mathcal{H}_1}J_{g_2,\dots,g_L}(g_1)\right)+\sum_{l=2}^{L}\lambda_l\|g_l\|_{\mathcal{H}_l}^2.$$

Since G_1 leaves the second argument unchanged, it directly follows from the $\mathbb{P}$-integrability that $\mathcal{L}$ is also a $G_{1,\star}(\mathbb{P})$-integrable Nemitski loss. Therefore, the application of the infinite-sample representer theorem 5.8 in Steinwart and Christmann (2008) states that the minimizer $g_1^\star$ of $J_{g_2,\dots,g_L}$ can be written as

$$\begin{aligned}g_1^\star(\cdot)&=-\frac{1}{2\lambda_1}\int_{R_2\times R_1}\mathcal{L}^{(1)}\left(y,g_1^\star(\boldsymbol{\xi})\right)K_1(\cdot,\boldsymbol{\xi})\,\mathrm{d}G_{1,\star}(\mathbb{P})(\boldsymbol{\xi},y)\\&=-\frac{1}{2\lambda_1}\int_{\Omega\times R_1}\mathcal{L}^{(1)}\left(y,g_1^\star\circ g_2\circ\ldots\circ g_L(\boldsymbol{x})\right)K_1(\cdot,g_2\circ\ldots\circ g_L(\boldsymbol{x}))\,\mathrm{d}\mathbb{P}(\boldsymbol{x},y),\end{aligned}$$

where $\mathcal{L}^{(1)}$ denotes the first derivative of $\mathcal{L}$ w.r.t. the second argument. For the choice $g_i=f_i$ for $i=2,\dots,L$, we obtain the minimizer $f_1=g_1^\star$. Note that f_1 is continuous and $\|f_1\|_\infty:=\sup_{\boldsymbol{x}\in D_1}|f_1(\boldsymbol{x})|<\infty$ since $\mathcal{H}_1\hookrightarrow C(D_1)$ follows directly by (12). Therefore, (14) is true for $l=1$ since

$$\begin{aligned}|A_{f_1,\dots,f_L}(\cdot)|:=&\left|\mathcal{L}^{(1)}\left(y,f_1\circ f_2\circ\ldots\circ f_L(\cdot)\right)\right|\leq b_1(y)+h_1\left(|f_1\circ f_2\circ\ldots\circ f_L(\cdot)|\right)\\&\leq b_1(y)+h_1\left(\|f_1\|_\infty\right)\end{aligned}$$

is in $L_{1,\mathbb{P}}$ since $b_1\in L_{1,\mathbb{P}_{R_1}}(R_1)$.

To tackle the next layer, we define $\tilde{\mathcal{L}}:R_1\times R_2\to[0,\infty)$ by

$$\tilde{\mathcal{L}}(y,\boldsymbol{z}):=\mathcal{L}(y,f_1(\boldsymbol{z})).$$

We proceed by showing that $\tilde{\mathcal{L}}$ is a $\mathbb{P}$-integrable and 1-times differentiable Nemitski vector loss. Then we show that we can use analogous techniques as in Steinwart and Christmann (2008)—but for vector-valued functions—to ensure the representation (14) for $l=2$. These arguments can then be iterated until we reach the innermost layer and the proof is completed. Since the details are quite technical, we outsourced them into appendix A. ■

Theorem 2 states that the solution f_l in the l-th layer of (13) is an element of the range of

1. Here, $\mathbb{P}_{R_1}$ denotes the marginal distribution of $\mathbb{P}$ w.r.t. the second variable.

the integral operator defined by the kernel $K_l\left(\cdot, f_{l+1} \circ \ldots \circ f_L(\cdot)\right) : D_l \times \Omega \to \mathbb{R}^{d_l \times d_l}$. Note that the statement of theorem 1 can be derived by choosing a sum of finitely many Dirac measures $\delta_{\boldsymbol{x}_i, y_i}$ as $\mathbb{P}$ in theorem 2. In this special case, the result boils down to f_l being in the span of the kernel evaluations in the data points.

Note furthermore that—in contrast to the finite sample case—f_l is defined as a convolution with the asymmetric kernel in (14). This can be interpreted as a smoothing step for many kernel choices. In this sense, we can expect the solutions f_l of (13) to employ a higher degree of smoothness than in the case of (7), where the solutions are only finite linear combinations of kernels. However, this of course comes at the cost of the regularity condition on the kernels in the requirements of theorem 2.

3.3. Relation to neural networks and deep learning

We now come back to the finite sample case in this section and discuss the relation of our representer theorem 1 to two of the most common approaches in deep learning with kernels, namely multi-layer multiple kernel learning (MLMKL) and deep kernel networks (DKN), see e.g. Cho and Saul (2009); Damianou and Lawrence (2013); Rebai et al. (2016); Strobl and Visweswaran (2013); Wilson et al. (2016); Zhuang et al. (2011). For reasons of simplicity, we restrict ourselves to the two-layer case $L = 2$ here.

3.3.1. Relation to hidden layer neural networks

Let us first illustrate how our approach can be encoded as a hidden layer feed-forward neural network. The idea behind artificial neural networks is the same as for multi-layer kernel learning, namely using concatenations of functions to compute good approximations. More precisely, the so-called universal approximation theorem states that already a two-layer neural network can approximate any continuous function arbitrarily well, see Cybenko (1989); Hornik (1991). For more details on artificial neural networks and deep learning, we refer the reader to Goodfellow et al. (2016).

As mentioned in the two-layer case above, we are aiming to find a function $h(\cdot) = f_1 \circ f_2(\cdot) = \sum_{j=1}^{N} \alpha_j \mathcal{K}^2(\boldsymbol{x}_j, \cdot)$ with $f_1 \in \mathcal{H}_1$ and $f_2 \in \mathcal{H}_2$ and associated K_1 and K_2, respectively, where the kernel $\mathcal{K}^2$ is given by (11). The construction of h can be easily encoded as a feed-forward neural network with one hidden layer if K_1 is a radial basis function (RBF) kernel for instance[2]. We illustrate[3] the case $d_2 = 1$ with an RBF kernel $K_1(z_1, z_2) = a(|z_1 - z_2|)$ for some function $a : \mathbb{R} \to \mathbb{R}$ in Figure 2. The first layer is split into the input layer with values $K_2(\boldsymbol{x}_i, \boldsymbol{x})$ for $i = 1, \ldots, N$ and an artificial "always on" layer with neuron-clusters that supply the constant values $K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ with weights $-c_j$ for $i, j = 1, \ldots, N$. Note that the i-th cluster $K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ of the "always on" layer is only connected to the i-th neuron of the hidden layer. Note furthermore that the inputs $K_2(\boldsymbol{x}_i, \boldsymbol{x})$ can also easily be computed by a neural network with fixed weights if K_2 is a radial basis

2. For many other types of kernels, e.g. tensor products of RBF kernels, one can still construct a more complex Sigma-Pi neural network for the computation of the output values.
3. Note that we only choose $d_2 = 1$ for illustrative reasons. For $d_2 > 1$, a neural network can be built analogously with an additional hidden layer to compute the norm of the difference of d_2-dimensional vectors. However, this additional layer, which just computes $\|\boldsymbol{x} - \boldsymbol{y}\|_2$ for given $\boldsymbol{x}$ and $\boldsymbol{y}$, has fixed weights and does not play any role for the optimization of the neural network.

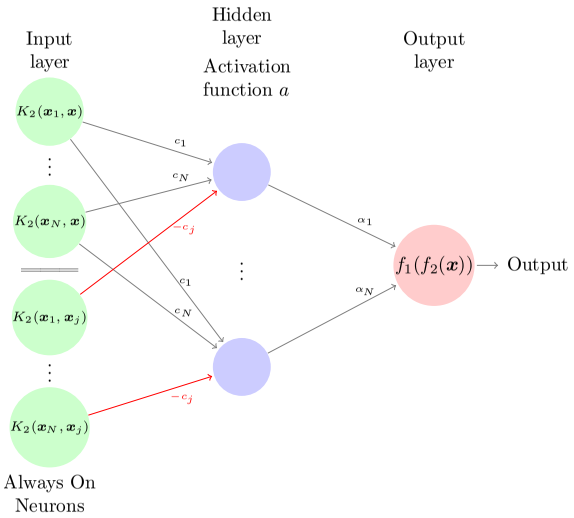


Figure 2: A hidden layer, feed-forward neural network to simulate the concatenation of two functions f_1 and f_2 from reproducing kernel Hilbert spaces. For reasons of readability, we choose $d_2 = 1$ and write $c_i := c_{i,1}$. The outer kernel is $K_1(z_1, z_2) = a(|z_1 - z_2|)$. Note that the i-th artificial "always on" neuron-cluster in the lower half of the first layer is written as $K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$, which stands for N single neurons with values $K_2(\boldsymbol{x}_i, \boldsymbol{x}_1), \ldots, K_2(\boldsymbol{x}_i, \boldsymbol{x}_N)$. The cluster $K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is only connected to the i-th neuron of the hidden layer with weights $-c_j$ (red lines). This means that the value $\sum_{j=1}^{N} -c_j K_2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is forwarded to the i-th neuron of the hidden layer.

kernel. If we consider a "deeper" concatenation, we would need a deeper neural network with additional layers, i.e. for $f_1 \circ \ldots \circ f_L$, we need $L - 1$ hidden layers.

3.3.2. Relation to multi-layer multiple kernel learning

The common idea in MLMKL methods is to learn a kernel $\tilde{K}$, which consists of a chain of linear combinations of functions and an inner kernel, e.g.

$$\tilde{K}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\ell=1}^{n_1} \nu_{1,\ell} k_{1,\ell} \left(\sum_{i=1}^{n_2} \nu_{2,i} K_{2,i}(\boldsymbol{x}, \boldsymbol{y}) \right)$$

in the two-layer case, where $k_{1,\ell}$ are real-valued functions for $\ell = 1, \ldots, n_1$ and $K_{2,i}$ are different scalar-valued kernels for $i = 1, \ldots, n_2$. Note that the functions $k_{1,\ell}$ are chosen such that $\tilde{K}$ is still a kernel. In the case of linear $k_{1,\ell}$, Dinuzzo (2011) has shown that the resulting algorithm becomes a standard MKL procedure and can be interpreted as a two-layer kernel network with a linear outer kernel. However, for arbitrary $k_{1,\ell}$ this is not

the case and we are dealing with a true MLMKL approach. The specific MLMKL algorithm then aims to find the optimal values for the coefficients $\nu_{1,\ell}, \nu_{2,i}$ in order to determine the best $\tilde{K}$ for a regression of the given data X and Y with e.g. a support vector regression algorithm. Note that the kernels and the k-functions are usually chosen heuristically, e.g. as polynomials, Gaussians, sigmoidals, etc., see Rebai et al. (2016); Zhuang et al. (2011).

To apply our result to the two-layer MKL method above, let us consider the case $n_1 = 1$ and $n_2 = N$. We set $\nu_{1,1} = 1$ without loss of generality. We let the outer function $k_{1,1}(z) = a(|z|)$ be the radial basis function used for the outer kernel (i.e. middle layer) in Figure 2. Furthermore, we set

$$K_{2,i}(\boldsymbol{x}, \boldsymbol{y}) := K_2(\boldsymbol{x}_i, \boldsymbol{x}) - K_2(\boldsymbol{x}_i, \boldsymbol{y}).$$

Note that the $K_{2,i}$ are no longer kernels anymore in this setting. However, they are now directly connected to our concatenated function learning approach since

$$\begin{aligned}\tilde{K}(\boldsymbol{x}, \boldsymbol{y}) &= k_{1,1}\left(\sum_{i=1}^{N} \nu_{2,i} K_{2,i}(\boldsymbol{x}, \boldsymbol{y})\right) = a\left(\left|\sum_{i=1}^{N} \nu_{2,i} K_2(\boldsymbol{x}_i, \boldsymbol{x}) - \nu_{2,i} K_2(\boldsymbol{x}_i, \boldsymbol{y})\right|\right) \\ &= K_1\left(\sum_{i=1}^{N} \nu_{2,i} K_2(\boldsymbol{x}_i, \boldsymbol{x}), \sum_{i=1}^{N} \nu_{2,i} K_2(\boldsymbol{x}_i, \boldsymbol{y})\right) = \mathcal{K}^2(\boldsymbol{x}, \boldsymbol{y})\end{aligned}$$

from (11) with $c_i = \nu_{2,i}$ and the kernels K_1 and K_2 used in Figure 2. Altogether, we thus see that an MLMKL algorithm with these parameters already determines the optimal solution (provided that the right hand side of (9) is solved exactly) among all functions of type $h = f_1 \circ f_2$ with $f_1 \in \mathcal{H}_1$ and $f_2 \in \mathcal{H}_2$ according to Theorem 1. This way, our representer theorem for concatenated functions directly applies to a special case of MLMKL networks. Note however that a generalization of our arguments to more layers, i.e. $L > 2$, is not straightforward for MLMKL.

3.3.3. Relation to deep kernel learning approaches

The class of DKN methods consists of algorithms which build a kernel by nonlinearly transforming the input vectors before applying an outer kernel function. This is in contrast to the MLMKL approach, where only the innermost function is a two-variate kernel and its evaluations are modified by some nonlinear outer functions. The models in this class range from simple feature map powers for some function Ψ, i.e.

$$\tilde{K}(x, y) = \underbrace{\Psi \circ \ldots \circ \Psi}_{L-1 \text{ times}}(\boldsymbol{x}) \cdot \underbrace{\Psi \circ \ldots \circ \Psi}_{L-1 \text{ times}}(\boldsymbol{y}),$$

see Cho and Saul (2009), to more general variants like

$$\tilde{K}(x, y) = K\left(f_2 \circ \ldots \circ f_L(\boldsymbol{x}), f_2 \circ \ldots \circ f_L(\boldsymbol{y})\right)$$

with nonlinear functions $f_2, \ldots, f_L$, see Wilson et al. (2016). If we assume that $f_l \in \mathcal{H}_l$ for $l = 2, \ldots, L$ stem from reproducing kernel Hilbert spaces with associated kernels K_l, we can apply Theorem 1 to this approach and obtain that each f_l can be written as a finite linear combination of evaluations of the kernel K_l. Thus, we can directly apply our representer theorem for L-layer DKN algorithms.

3.4. The two-layer interpolation problem

After analyzing the general multi-layer kernel concatenation problem in Theorem 1, we now have a closer, more detailed look at the main component of it, namely the *concatenation of two functions*. To this end, we specifically consider the interpolation problem for $L = 2$. This simple, illustrative setting gives further insights into the way concatenation works in machine learning problems.

3.4.1. Definition of the problem

We slightly adapt our notation to this special case to obtain a direct relation to the single-layer interpolation problem from Section 2. To this end, let $D := d_2$ and consider the domain $\Phi := D_1 \subseteq \mathbb{R}^D$ together with the two function spaces

$$\begin{aligned} H(\Phi, \mathbb{R}) &:= \mathcal{H}_1 \subset C(\Phi) := \{f : \Phi \to \mathbb{R} \mid f \text{ continuous}\} && \text{– “outer” space,} \\ \boldsymbol{H}(\Omega, \Phi) &:= \mathcal{H}_2 \subset \left\{\boldsymbol{g} = (g_1, \ldots, g_D)^T : \Omega \to \Phi \mid \boldsymbol{g} \text{ continuous}\right\} && \text{– “inner” space.}\end{aligned}$$

Both spaces are supposed to be reproducing kernel Hilbert spaces, i.e. there is an (outer) kernel $K := K_1 : \Phi \times \Phi \to \mathbb{R}$ for $H(\Phi, \mathbb{R})$ such that

$$\begin{aligned} K(\boldsymbol{x}, \cdot) &\in H(\Phi, \mathbb{R}) && \text{for all } \boldsymbol{x} \in \Omega, \\ f(\boldsymbol{x}) &= (f, K(\boldsymbol{x}, \cdot))_{H(\Phi,\mathbb{R})} && \text{for all } \boldsymbol{x} \in \Omega \text{ and all } f \in H(\Phi, \mathbb{R}).\end{aligned}$$

The function space $\boldsymbol{H}(\Omega, \Phi)$ is assumed to be a vector-valued RKHS, i.e. there is an (inner) kernel $\boldsymbol{K} : \Omega \times \Omega \to \mathbb{R}^{D \times D}$ such that

$$\begin{aligned} \boldsymbol{K}(\boldsymbol{x}, \cdot)\, \boldsymbol{c} &\in \boldsymbol{H}(\Omega, \Phi) && \text{for all } \boldsymbol{x} \in \Omega \text{ and all } \boldsymbol{c} \in \mathbb{R}^D, \\ \boldsymbol{c}^T \boldsymbol{g}(\boldsymbol{x}) &= (\boldsymbol{g}, \boldsymbol{K}(\boldsymbol{x}, \cdot)\, \boldsymbol{c})_{\boldsymbol{H}(\Omega,\Phi)} && \text{for all } \boldsymbol{x} \in \Omega, \text{ all } \boldsymbol{c} \in \mathbb{R}^D \text{ and all } \boldsymbol{g} \in \boldsymbol{H}(\Omega, \Phi).\end{aligned}$$

To formulate the concatenated interpolation problem in the spirit of (1), we have to define an appropriate functional and propose an appropriate search set for the minimization task. To this end, we consider the functional $J : H(\Phi, \mathbb{R}) \times \boldsymbol{H}(\Omega, \Phi) \to \mathbb{R}$ given by

$$J(f, \boldsymbol{g}) := \|f\|^2_{H(\Phi,\mathbb{R})} + \|\boldsymbol{g}\|^2_{\boldsymbol{H}(\Omega,\Phi)},$$

which penalizes the norms of both the outer and the inner function, and the admissible set

$$\mathcal{A}_{X,Y} := \{(f, \boldsymbol{g}) \in H(\Phi, \mathbb{R}) \times \boldsymbol{H}(\Omega, \Phi) \mid f \circ \boldsymbol{g}(\boldsymbol{x}_j) = y_j \ 1 \le j \le N\},$$

i.e. the set of all concatenations of functions from $H(\Phi, \mathbb{R})$ and $\boldsymbol{H}(\Omega, \Phi)$ which interpolate the data. With this notation, we can define the following variational optimization problem

$$J(f, \boldsymbol{g}) \to \min \quad \text{for } (f, \boldsymbol{g}) \in \mathcal{A}_{X,Y} \tag{P}$$

As explained in Section 2, the solution $f^*_{X,Y}$ to the standard interpolation problem (1) can be computed by solving the system (3) of linear equations for a given set of fixed and pairwise disjoint input data points $X := \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$. Therefore, if we assume for a moment the inner function $\boldsymbol{g}$ in (P) to be fixed and $Z := \boldsymbol{g}(X) = \{\boldsymbol{z}_i = \boldsymbol{g}(\boldsymbol{x}_i) \mid i = 1, \ldots, N\}$,

then we obtain that the solution $f^*_{Z,Y}$ to (1) with data points Z is the only admissible minimizer of the concatenated interpolation problem (P), i.e.

$$f^*_{Z,Y} = \underset{f\in\{h\in H(\Phi,\mathbb{R})\mid(h,\boldsymbol{g})\in\mathcal{A}_{X,Y}\}}{\arg\min} \|f\|^2_{H(\Phi,\mathbb{R})}.$$

Note that the coefficients $\boldsymbol{\alpha}^* \in \mathbb{R}^N$ of $f^*_{Z,Y} = \sum_{i=1}^N \alpha_i^* K(\boldsymbol{z}_i,\cdot)$ can be computed by solving the system

$$\boldsymbol{M}_{Z,Z}\boldsymbol{\alpha}^* = \boldsymbol{y}$$

and the value of the optimal energy, i.e. the squared norm, is given by

$$\left\|f^*_{Z,Y}\right\|^2_{H(\Phi,\mathbb{R})} = \boldsymbol{\alpha}^{*T}\boldsymbol{M}_{Z,Z}\boldsymbol{\alpha}^* = \boldsymbol{y}^T\boldsymbol{M}^{-1}_{Z,Z}\boldsymbol{y}.$$

3.4.2. Application of the representer theorem

In order to rewrite the concatenated interpolation problem (P) into an unconstrained minimization problem by applying the above result, we first have to discuss what happens if $\boldsymbol{g}(\boldsymbol{x}_j) = \boldsymbol{g}(\boldsymbol{x}_k)$ for two indices $j \neq k$. If equality holds also for the corresponding values from Y, i.e. $y_j = y_k$, we can simply remove the pair $(x_j, y_j) \in X \times Y$ from the input data and with it also the corresponding condition from the admissible set. However, if $y_j \neq y_k$, there cannot be an $f \in H(\Phi,\mathbb{R})$ such that $(f,\boldsymbol{g}) \in \mathcal{A}_{X,Y}$. In this case, we simply set $J(f,\boldsymbol{g}) = \infty$. Using this convention, we can recast (P) into the unrestricted optimization problem

$$J\left(f^*_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) = \boldsymbol{y}^T\boldsymbol{M}^{-1}_{\boldsymbol{g}(X),\boldsymbol{g}(X)}\boldsymbol{y} + \|\boldsymbol{g}\|^2_{\boldsymbol{H}(\Omega,\Phi)} \to \min \quad \text{for } \boldsymbol{g}\in\boldsymbol{H}(\Omega,\Phi). \tag{uP}$$

Therefore, we only have to consider the minimization with respect to $\boldsymbol{g} \in \boldsymbol{H}(\Omega,\Phi)$ since the optimal outer function $f^*_{\boldsymbol{g}(X),Y}$ is completely determined by the inner function values $\boldsymbol{g}(X)$ and Y.

Note that the side condition $\boldsymbol{g}(\boldsymbol{x}_j) \neq \boldsymbol{g}(\boldsymbol{x}_k)$ for $j \neq k$ can also be enforced by adding a penalty term of type $\sum_{i<j} W\left(\|\boldsymbol{g}(\boldsymbol{x}_i) - \boldsymbol{g}(\boldsymbol{x}_j)\|_2^2\right)$ to J, where W is a smooth function with $W(0) = \infty$, e.g. $W(x) = \coth(x)$. This can also remedy the problem of small condition numbers of $\boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)}$ for large sample sizes since it maximizes distances between the point evaluations of $\boldsymbol{g}$. Adding this to (uP), we obtain

$$J_\gamma\left(f^*_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) := J\left(f^*_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) + \gamma \sum_{1\le i<j\le N} \coth\left(\|\boldsymbol{g}(\boldsymbol{x}_i) - \boldsymbol{g}(\boldsymbol{x}_j)\|_2^2\right) \tag{15}$$
$$\to \min \quad \text{for } \boldsymbol{g}\in\boldsymbol{H}(\Omega,\Phi).$$

However, since using $J_0 = J$ in our experiments in Section 4 works out already well and the side condition does not seem to affect the results for moderate sample sizes, we restrict ourselves to the problem (uP) in the following.

Although the above considerations seem to simplify the concatenated interpolation problem, we still have to solve a highly nonlinear optimization problem over the (possibly) infinite-dimensional RKHS $\boldsymbol{H}(\Omega,\Phi)$. Nonetheless, by applying Theorem 1 to the unrestricted concatenated interpolation problem (uP), we can restrict the search space $\boldsymbol{H}(\Omega,\Phi)$ to the span of the kernel translates in the input data.

Corollary 3 *Let $V_X := \operatorname{span}\{\boldsymbol{K}(\boldsymbol{x}_i,\cdot)\boldsymbol{e}_j \mid i = 1,\dots,N \text{ and } j = 1,\dots,D\}$, where $\boldsymbol{e}_j$ denotes the j-th unit vector in $\mathbb{R}^D$. Then, the solution $\boldsymbol{g}^*$ to the unconstrained concatenated interpolation problem* (uP) *fulfills $\boldsymbol{g}^* \in V_X$.*

Proof We apply Theorem 1 with $L = 2$, $\Theta_1(x) = \Theta_2(x) = x$ and

$$\mathcal{L}(y_i, f\circ\boldsymbol{g}(\boldsymbol{x}_i)) = \begin{cases} 0 & \text{if } f\circ\boldsymbol{g}(\boldsymbol{x}_i) = y_i \\ \infty & \text{else,}\end{cases}$$

which exactly resembles the interpolation problem (uP). ■

Due to Corollary 3, we can recast the unrestricted concatenated interpolation problem (uP) into

$$J\left(f^*_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) = \boldsymbol{y}^T\boldsymbol{M}^{-1}_{\boldsymbol{g}(X),\boldsymbol{g}(X)}\boldsymbol{y} + \|\boldsymbol{g}\|^2_{\boldsymbol{H}(\Omega,\Phi)} \to \min \text{ for } \boldsymbol{g}\in V_X \subset \boldsymbol{H}(\Omega,\Phi). \tag{uP-X}$$

This is a nonlinear, finite-dimensional and unrestricted optimization problem. We fix the basis $\{\boldsymbol{K}(\boldsymbol{x}_j,\cdot)\boldsymbol{e}_\ell \mid (j,\ell)\in\mathcal{I}\}$ with $\mathcal{I} := \{(j,\ell) \in \mathbb{N}^2 \mid 1\le j\le N, 1\le\ell\le D\}$ to solve (uP-X). Then, the optimal solution can be written as

$$\boldsymbol{g}^*(\cdot) = \sum_{(j,\ell)\in\mathcal{I}} c^*_{j,\ell}\boldsymbol{K}(\boldsymbol{x}_j,\cdot)\boldsymbol{e}_\ell. \tag{16}$$

In order to express the minimization problem (uP-X) with respect to the coefficients $\boldsymbol{c}^* = \left(c^*_{1,1},\dots,c^*_{N,D}\right)^T$, we introduce $\boldsymbol{Q}_{X,X}(\boldsymbol{c}) := \boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)}$, i.e.

$$\boldsymbol{Q}_{X,X}(\boldsymbol{c}) = \left(K\left(\sum_{(j,\ell)\in I} c_{j,\ell}\boldsymbol{K}(\boldsymbol{x}_j,\boldsymbol{x}_n)\boldsymbol{e}_\ell, \sum_{(j,\ell)\in I} c_{j,\ell}\boldsymbol{K}(\boldsymbol{x}_j,\boldsymbol{x}_m)\boldsymbol{e}_\ell\right)\right)_{1\le n,m\le N} \tag{17}$$

and the corresponding quadratic form

$$\mathcal{Q}: \mathbb{R}^{ND}\to\mathbb{R}, \quad \boldsymbol{c}\mapsto \boldsymbol{y}^T\boldsymbol{Q}_{X,X}(\boldsymbol{c})^{-1}\boldsymbol{y}.$$

Furthermore, to express $\|\boldsymbol{g}^*\|^2_{\boldsymbol{H}(\Omega,\Phi)}$ with respect to $\boldsymbol{c}^*$, we need

$$\mathcal{N}: \mathbb{R}^{ND}\to\mathbb{R}, \quad \boldsymbol{c}\mapsto \sum_{j,k=1}^N \begin{pmatrix} c_{j,1}\\ \vdots \\ c_{j,D}\end{pmatrix}^T \boldsymbol{K}(\boldsymbol{x}_j,\boldsymbol{x}_k) \begin{pmatrix} c_{k,1}\\ \vdots \\ c_{k,D}\end{pmatrix}. \tag{18}$$

Finally, we obtain the finite-dimensional optimization problem

$$\boldsymbol{c}^* = \underset{\boldsymbol{c}\in\mathbb{R}^{ND}}{\arg\min} \underbrace{\mathcal{Q}(\boldsymbol{c})}_{\|f^*_{\boldsymbol{g}(X),Y}\|^2_{H(\Phi,\mathbb{R})}} + \underbrace{\mathcal{N}(\boldsymbol{c})}_{\|\boldsymbol{g}\|^2_{\boldsymbol{H}(\Omega,\Phi)}}. \tag{Int}$$

3.4.3. Solving the minimization problem

The unconstrained problem (Int) is highly nonlinear because the coefficients $c_{j,\ell}$ are transformed by the outer kernel function K. It can be tackled by any suitable iterative optimization algorithm. If the kernels K and $\boldsymbol{K}$ are differentiable, a quasi-Newton approach is appropriate. If this is not the case, a derivative-free optimizer should be chosen.

Note that we can restrict the minimization in (Int) to a compact subset of $\mathbb{R}^{ND}$ without loss of generality. To this end, let $\underline{\boldsymbol{K}} \in \mathbb{R}^{ND\times ND}$ be the $N \times N$ matrix of matrices $\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathbb{R}^{D\times D}$ and note that

$$\mathcal{Q}(\boldsymbol{c}) + \mathcal{N}(\boldsymbol{c}) \geq 0 + \lambda_{\min}\left(\underline{\boldsymbol{K}}\right) \cdot \|\boldsymbol{c}\|_2^2 \overset{\|\boldsymbol{c}\|_2\to\infty}{\longrightarrow} \infty,$$

where $\lambda_{\min}\left(\underline{\boldsymbol{K}}\right) > 0$ denotes the smallest eigenvalue of $\underline{\boldsymbol{K}}$. Therefore, we can restrict our search to the compact set $A := \left\{\boldsymbol{c} \in \mathbb{R}^{ND} \mid \|\boldsymbol{c}\|_2 \leq C\right\}$ for a large enough $C > 0$. Unfortunately, we cannot directly obtain the existence of a minimizer from this since (Int) is not continuous. However, if we add a smooth term

$$\begin{aligned}\mathcal{P}^\gamma(\boldsymbol{c}) &= \gamma \sum_{1\leq m<n\leq N} \coth\left(\|\boldsymbol{g}(\boldsymbol{x}_m) - \boldsymbol{g}(\boldsymbol{x}_n)\|_2^2\right) \\ &= \gamma \sum_{1\leq m<n\leq N} \coth\left(\left\|\sum_{(j,\ell)\in\mathcal{I}} c_{j,\ell}\left(\boldsymbol{K}\left(\boldsymbol{x}_j, \boldsymbol{x}_m\right) - \boldsymbol{K}\left(\boldsymbol{x}_j, \boldsymbol{x}_n\right)\right)\boldsymbol{e}_\ell\right\|_2^2\right),\end{aligned}$$

which is equivalent to (15), for $\gamma > 0$, we can deduce the existence of a minimizer with the direct method from the calculus of variations. To this end, note that for a minimizing sequence $(\boldsymbol{c}_i)_{i=1}^\infty$ of $\mathcal{Q} + \mathcal{N} + \mathcal{P}^\gamma$, there necessarily exist $i_0 \in \mathbb{N}$ and $C_0 > 0$ such that all mutual squared distances $\|\boldsymbol{g}(\boldsymbol{x}_m) - \boldsymbol{g}(\boldsymbol{x}_n)\|_2^2$ with $1 \leq m < n \leq N$ are larger than C_0 for all $\boldsymbol{c}_i$ with $i > i_0$. Therefore, we can restrict the minimization to the compact subdomain

$$A \cap \left\{\boldsymbol{c} \in \mathbb{R}^{ND} \mid \|\boldsymbol{g}(\boldsymbol{x}_m) - \boldsymbol{g}(\boldsymbol{x}_n)\|_2^2 \geq C_0 \ \text{ for } 1 \leq m < n \leq N\right\},$$

on which $\mathcal{Q}+\mathcal{N}+\mathcal{P}^\gamma$ is continuous, and the existence of a minimizer follows. Nevertheless, as we explained above, the critical condition $\mathcal{P}^\gamma(\boldsymbol{c}) = \infty$ is practically never met for moderate data set sizes and, therefore, it is safe to assume that there also exists a minimizer for (Int). Note however that, depending on the kernels and the data at hand, there usually might exist many minimizers and the solution to (Int) might not be unique. To reduce the chance of getting stuck in a local minimum, we propose to restart the minimization procedure several times with different starting values for $\boldsymbol{c}^*$.

Since we will be dealing with differentiable kernel functions in Section 4 and since the derivatives of these kernels can be computed explicitly, we propose a BFGS minimization algorithm to solve (Int). To this end, note that the only derivatives we need are essentially the derivative of the inverse of $\boldsymbol{Q}_{X,X}(\boldsymbol{c})$, i.e.

$$\frac{\partial}{\partial c_{m,n}}\boldsymbol{Q}_{X,X}^{-1}(\boldsymbol{c}) = -\boldsymbol{Q}_{X,X}^{-1}(\boldsymbol{c})\frac{\partial}{\partial c_{m,n}}\boldsymbol{Q}_{X,X}(\boldsymbol{c})\boldsymbol{Q}_{X,X}^{-1}(\boldsymbol{c}),$$

and the derivative of $\boldsymbol{Q}_{X,X}(\boldsymbol{c})$. The latter consists of the derivative of the outer kernel K, which is known analytically for all kernel choices that we discuss in Section 4, and

$$\frac{\partial}{\partial c_{m,n}}\boldsymbol{g}(\boldsymbol{x}) = \frac{\partial}{\partial c_{m,n}}\sum_{(j,\ell)\in\mathcal{I}} c_{j,\ell}\boldsymbol{K}\left(\boldsymbol{x}_j, \boldsymbol{x}\right)\boldsymbol{e}_\ell = \boldsymbol{K}\left(\boldsymbol{x}_m, \boldsymbol{x}\right)\boldsymbol{e}_n$$

for each $(m, n) \in \mathcal{I}$. The overall computational cost complexity for one BFGS step, i.e. the evaluation of $\mathcal{Q}, \mathcal{N}$ and their derivatives, is bounded by $\mathcal{O}\left(N^3D + (ND)^2\right)$.

3.5. Two-layer Least-squares regression

After the discussion of the two-layer interpolation problem in the last section, we now consider the regularized two-layer least-squares problem in more detail. This is a natural extension of the two-layer least-squares problem RLS2 considered in Dinuzzo (2011) to the case of nonlinear outer kernels.

3.5.1. Definition of the problem

For concatenated, regularized least-squares regression, the minimization task changes to

$$J_{\lambda,\mu}\left(f, \boldsymbol{g}\right) := \sum_{j=1}^N \left|f \circ \boldsymbol{g}\left(\boldsymbol{x}_j\right) - y_j\right|^2 + \lambda\left\|f\right\|_{H(\Phi,\mathbb{R})}^2 + \mu\left\|\boldsymbol{g}\right\|_{\boldsymbol{H}(\Omega,\Phi)}^2 \qquad \text{(R)}$$
$$\to \min \ \text{ for } f \in H\left(\Phi, \mathbb{R}\right), g \in \boldsymbol{H}\left(\Omega, \Phi\right)$$

with $\lambda, \mu > 0$, which is in the same fashion as the standard least-squares regression problem (5).

Analogously to our considerations in Section 3.4, we find that, for fixed inner points $Z = \boldsymbol{g}(X) \subset \Phi$, the function $f_{Z,Y}^\lambda$, see (5), is the solution of the problem

$$\sum_{j=1}^N \left|f(\boldsymbol{z}_j) - y_j\right|^2 + \lambda\left\|f\right\|_{H(\Phi,\mathbb{R})}^2 \to \min \ \text{ for } f \in H\left(\Phi, \mathbb{R}\right).$$

The corresponding coefficients $\boldsymbol{\alpha}^\lambda \in \mathbb{R}^N$ with respect to the basis $\{K(\boldsymbol{z}_j, \cdot) \mid j = 1, \ldots, N\}$ are computed by solving

$$\left(\boldsymbol{M}_{Z,Z} + \lambda\boldsymbol{I}\right)\boldsymbol{\alpha}^\lambda = \boldsymbol{y}.$$

Therefore, each of the terms of the optimal energy can be expressed as

$$\begin{aligned}\left\|f_{Z,Y}^\lambda\right\|_{H(\Phi,\mathbb{R})}^2 &= {\boldsymbol{\alpha}^\lambda}^T\boldsymbol{M}_{Z,Z}\boldsymbol{\alpha}^\lambda = \boldsymbol{y}^T\left(\boldsymbol{M}_{Z,Z} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{M}_{Z,Z}\left(\boldsymbol{M}_{Z,Z} + \lambda\boldsymbol{I}\right)^{-1}\boldsymbol{y}, \\ \sum_{j=1}^N\left|f_{Z,Y}^\lambda\left(\boldsymbol{z}_j\right) - y_j\right|^2 &= \left\|\boldsymbol{M}_{Z,Z}\boldsymbol{\alpha}^\lambda - \boldsymbol{y}\right\|_2^2 = \left\|\left(\boldsymbol{I} - \boldsymbol{M}_{Z,Z}\left(\boldsymbol{M}_{Z,Z} + \lambda\boldsymbol{I}\right)^{-1}\right)\boldsymbol{y}\right\|_2^2.\end{aligned}$$

3.5.2. Application of the representer theorem

Analogously to (uP), we can use

$$J_{\lambda,\mu}\left(f^{\lambda}_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) = \lambda \boldsymbol{y}^T \left(\boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} \left(\boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{y} + \mu \left\|\boldsymbol{g}\right\|^2_{\boldsymbol{H}(\Omega,\Phi)} + \left\| \left(\boldsymbol{I} - \boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} \left(\boldsymbol{M}_{\boldsymbol{g}(X),\boldsymbol{g}(X)} + \lambda \boldsymbol{I}\right)^{-1}\right) \boldsymbol{y} \right\|^2_2 \tag{19}$$

to reformulate (R) as

$$J_{\lambda,\mu}\left(f^{\lambda}_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) \to \min \text{ for } \boldsymbol{g} \in \boldsymbol{H}(\Omega,\Phi). \tag{uR}$$

Corollary 4 *The solution $\boldsymbol{g}^{\lambda,\mu}$ to the unconstrained concatenated regression problem* (uR) *fulfills $\boldsymbol{g}^{\lambda,\mu} \in V_X$.*

Proof We apply Theorem 1 with $L = 2$, $\Theta_1(x) = \lambda \cdot x$, $\Theta_2(x) = \mu \cdot x$ and

$$\mathcal{L}\left(y_i, f \circ \boldsymbol{g}(\boldsymbol{x}_i)\right) = \left|f \circ \boldsymbol{g}\left(\boldsymbol{x}_i\right) - y_i\right|^2,$$

which resembles the regression problem (R). ■

Hence, as for interpolation, we obtain a representer theorem for concatenated least-squares regression, which allows us to replace the infinite-dimensional optimization problem (R) with the finite-dimensional problem

$$J_{\lambda,\mu}\left(f^{\lambda}_{\boldsymbol{g}(X),Y},\boldsymbol{g}\right) \to \min \text{ for } \boldsymbol{g} \in V_X \subset \boldsymbol{H}(\Omega,\Phi). \tag{uR-X}$$

Finally, we want to express (uR-X) in terms of the coefficients $\boldsymbol{c}^{\lambda,\mu} = \left(c^{\lambda,\mu}_{1,1}, \ldots, c^{\lambda,\mu}_{N,D}\right)^T$ of $\boldsymbol{g}^{\lambda,\mu}$ with respect to the basis $\{\boldsymbol{K}(\boldsymbol{x}_j,\cdot)\,\boldsymbol{e}_\ell \mid (j,\ell) \in \mathcal{I}\}$. To this end, we set $\boldsymbol{A} := \left(\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}\right)^{-1}$ and define the quadratic forms

$$\begin{aligned}
\mathcal{Q}^{\lambda} &: \mathbb{R}^{ND} \to \mathbb{R}, \quad \boldsymbol{c} \mapsto \lambda \cdot \boldsymbol{y}^T \boldsymbol{A} \boldsymbol{Q}_{X,X}(\boldsymbol{c})\, \boldsymbol{A} \boldsymbol{y}, \\
\mathcal{N}^{\mu} &: \mathbb{R}^{ND} \to \mathbb{R}, \quad \boldsymbol{c} \mapsto \mu \cdot \mathcal{N}(\boldsymbol{c}) \quad \text{and} \\
\mathcal{C}^{\lambda} &: \mathbb{R}^{ND} \to \mathbb{R}, \quad \boldsymbol{c} \mapsto \boldsymbol{y}^T \left(\boldsymbol{I} - \boldsymbol{Q}_{X,X}(\boldsymbol{c})\, \boldsymbol{A}\right)^T \left(\boldsymbol{I} - \boldsymbol{Q}_{X,X}(\boldsymbol{c})\, \boldsymbol{A}\right) \boldsymbol{y}
\end{aligned}$$

with the help of (17) and (18). Subsequently, we arrive at the optimization problem

$$\boldsymbol{c}^{\lambda,\mu} = \underset{\boldsymbol{c} \in \mathbb{R}^{ND}}{\arg\min}\, \mathcal{Q}^{\lambda}(\boldsymbol{c}) + \mathcal{N}^{\mu}(\boldsymbol{c}) + \mathcal{C}^{\lambda}(\boldsymbol{c}), \tag{Reg}$$

which is the equivalent to (uR-X).

3.5.3. Solving the minimization problem

Note that the existence of a minimizer follows by similar arguments as in the previous section for the interpolation problem, i.e.

$$\mathcal{Q}^{\lambda}(\boldsymbol{c}) + \mathcal{N}^{\mu}(\boldsymbol{c}) + \mathcal{C}^{\lambda}(\boldsymbol{c}) \geq \mu \cdot \lambda_{\min}\left(\underline{\boldsymbol{K}}\right) \cdot \left\|\boldsymbol{c}\right\|^2_2 \overset{\|\boldsymbol{c}\|_2 \to \infty}{\longrightarrow} \infty$$

and we can thus restrict the search for a minimizer to a compact subset of $\mathbb{R}^{ND}$. For regression we need the inverse of $\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}$ to compute $\mathcal{Q}^{\lambda}$, which is positive definite for every $\lambda > 0$ and, therefore, there are no pathological cases as in the interpolation setting. Thus, the functions $\mathcal{Q}^{\lambda}, \mathcal{N}^{\mu}, \mathcal{C}^{\lambda}$ are continuous and the minimization of (Reg) over a compact subset of $\mathbb{R}^{ND}$ has a minimizer. Nevertheless, also in this case the minimizer is not necessarily unique.

While the optimization for the coefficients in the RLS2 algorithm proposed in Dinuzzo (2011) boils down to a simplex-constrained linear least-squares problem, we have to deal with a high degree of nonlinearity here. Nevertheless, if the kernel functions are differentiable, we can again—as in the interpolation case—employ a BFGS algorithm with several restarts to approximately find the optimal coefficients $\boldsymbol{c}^{\lambda,\mu}$. To this end, note that $\mathcal{Q}^{\lambda}$ and $\mathcal{N}^{\mu}$ can be computed similarly as $\mathcal{Q}$ and $\mathcal{N}$ in the interpolation case. Furthermore, also the derivative of $\mathcal{C}^{\lambda}$ can be computed with the same techniques since we essentially only need the derivatives of $\boldsymbol{Q}_{X,X}(\boldsymbol{c})$ and $\left(\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}\right)^{-1}$. While the number of terms is larger than in the interpolation case, the asymptotic computational runtime is still bounded by $\mathcal{O}\left(N^3 D + (ND)^2\right)$. Furthermore, the condition number of the matrix $\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}$ is smaller than the one of $\boldsymbol{Q}_{X,X}$, which had to be inverted for interpolation. Therefore, computing $\mathcal{Q}^{\lambda}(\boldsymbol{c})$ with an iterative solver for the application of $\left(\boldsymbol{Q}_{X,X}(\boldsymbol{c}) + \lambda \boldsymbol{I}\right)^{-1}$ needs fewer computational steps than computing $\mathcal{Q}(\boldsymbol{c})$ in the interpolation case.

Finally, let us remark that for both interpolation and least-squares regression there exists another possibility to obtain a finite-dimensional optimization problem from (P) and (R), respectively, without using the representer theorem. We could discretize the functions $f_1 \in \mathcal{H}_1$ and $f_2 \in \mathcal{H}_2$ by $\tilde{f}_1 \in V_1$ and $\tilde{f}_2 \in V_2$ with finite-dimensional spaces V_1, V_2, see e.g. Bohn and Griebel (2017) for an error analysis of this scenario for single-layer regression. However, when following this approach, the choice of the specific discretization can severely influence the results of the minimization. Furthermore, we are limited by the size of the dimensions of the discretization spaces V_1, V_2, which influences the computational costs for solving the underlying optimization problem.

4. The effects of concatenated learning

This section serves to illustrate the main operating principle behind the concatenated interpolation and regression algorithms presented in the previous section. Note that our brief considerations in this section are not meant to provide a thorough numerical analysis of the performance of the algorithms but are rather thought to aid the understanding of their internal mechanisms. For benchmarks of highly performant variants of our basic algorithms on real-world data we refer the interested reader to Damianou and Lawrence (2013); Rebai et al. (2016); Zhuang et al. (2011).

4.1. Kernel choice

For reasons of simplicity, we will stick to the two-layer case and to outer function spaces $H(\Phi, \mathbb{R})$ with associated kernel K which are defined on the whole space $\mathbb{R}^D$. This way, the image Φ of the inner function space is automatically contained in the domain of the outer function space. Furthermore, if not stated otherwise, we assume that the matrix-valued kernel $\boldsymbol{K} : \Omega \times \Omega \to \mathbb{R}^{D\times D}$ of the inner RKHS can be written as

$$\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{y}) = K_{\mathcal{I}}(\boldsymbol{x}, \boldsymbol{y}) \cdot \operatorname{diag}(\boldsymbol{a}) \tag{20}$$

for some weight vector $\boldsymbol{a} \in \mathbb{R}^D_+$. Here, $\operatorname{diag}(\boldsymbol{a})$ denotes the diagonal matrix $\boldsymbol{A}$ with $\boldsymbol{A}_{ii} = \boldsymbol{a}_i$ and $K_{\mathcal{I}} : \Omega \times \Omega \to \mathbb{R}$ is a scalar-valued kernel function.

Possible outer and inner kernel functions K and $K_{\mathcal{I}}$ are the polynomial kernel

$$K_{\mathrm{Poly},p}(\boldsymbol{x}, \boldsymbol{y}) := \left(\boldsymbol{x}^T \boldsymbol{y} + 1\right)^p,$$

the Gaussian kernel

$$K_{\mathrm{Gauss},\sigma}(\boldsymbol{x}, \boldsymbol{y}) := \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{y}\|^2}{2\sigma^2}\right)$$

and the tensor-product Matérn kernel

$$K_{\mathrm{TensorMatérn},s}(\boldsymbol{x}, \boldsymbol{y}) := \prod_{i=1}^{d} \kappa_{\frac{2s-1}{2}}\left(|x_i - y_i|\right) \cdot |x_i - y_i|^{\frac{2s-1}{2}}.$$

where κ_α denotes the modified (hyperbolic) Bessel function of the second kind with parameter α. Note that the latter characterizes the Sobolev space of dominating mixed smoothness of order $s \in \mathbb{N}$, see e.g. Fasshauer and Ye (2011); Griebel and Harbrecht (2014) for a bivariate version. These Sobolev spaces play an important role for hyperbolic cross or sparse grid approximations for instance, see e.g. Bungartz and Griebel (2004). Note that the Gaussian kernel is already a tensor product kernel by nature.

4.2. Experiment design

Let us choose $\Omega = [-1, 1]^2$. We will evaluate our method for the two test functions

$$\begin{aligned} h_1 : \Omega \to \mathbb{R} \qquad & h_1(x, y) := (0.1 + |x - y|)^{-1} \\ h_2 : \Omega \to \mathbb{R} \qquad & h_2(x, y) := \begin{cases} 1 & \text{if } x \cdot y > \frac{3}{20} \\ 0 & \text{else} \end{cases}. \end{aligned}$$

The function h_1 employs a kink-like structure along the diagonal of the domain, while h_2 represents an indicator function with a jump. Neither of these two functions is an element of a reproducing kernel space spanned by any of the above kernel functions for arbitrary parameters $p, s \in \mathbb{N}, \sigma \in (0, \infty)$. Therefore they cannot be approximated too well by a single-layer method. The approximation of such functions with kinks or jumps by (a composition of) smooth functions plays an important role in applications from econometrics, finance or two-phase flow problems for example.

We choose $D = d = 2$, i.e. $\Omega, \Phi \subset \mathbb{R}^2$, and $\boldsymbol{a} = (1\ 1)^T$. Then, we independently draw $N = 100$ random equidistributed points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \subset \Omega$ and set $y_i := h_*(\boldsymbol{x}_i) + \varepsilon_i$ for all $i = 1, \ldots, N$ for the function $h_* \in \{h_1, h_2\}$. Here, ε_i are additive noise perturbations which are drawn i.i.d. according to a centered Gaussian distribution with standard deviation 0.01. To solve (Int) or (Reg), respectively, we use a BFGS algorithm with random initialization of the coefficient vector $\boldsymbol{c}$ of the inner function, see also (16). As the goal functions employ many local minima, we run the algorithm sufficiently many times to achieve a good approximation to the global minimum. It turned out that 64 runs were sufficient for our case of 100 data points in 2 dimensions. From the 64 runs we pick the vector $\boldsymbol{c}$ (and with this the functions f and $\boldsymbol{g}$) for which the smallest goal function value in (Int) or (Reg), respectively, is achieved.

To be able to compare our computed $f(g(\cdot))$, which approximates the true solution $f^*_{\boldsymbol{g}(X),Y}\left(\boldsymbol{g}^*(\cdot)\right)$ or $f^{\lambda}_{\boldsymbol{g}(X),Y}\left(\boldsymbol{g}^{\lambda,\mu}(\cdot)\right)$, respectively, to the result of a standard kernel interpolation/regression, we also calculate the interpolant/regressor $w \in \left\{f^*_{X,Y}, f^{\lambda}_{X,Y}\right\}$. This resembles the solution to (1) or (5), respectively, for the reproducing kernel Hilbert space $H(\Omega, \mathbb{R})$ which employs the same kernel type and parameters as $H(\Phi, \mathbb{R})$ but on the domain Ω instead of Φ. We then define $\boldsymbol{t}_i, i = 1, \ldots, n_t$, as the points of a uniform grid of meshwidth $\frac{1}{50}$ over $\Omega = [-1, 1]^2$, i.e. $n_t = 101^2$, and consider the *pointwise error*

$$\left|\left(f \circ \boldsymbol{g} - h_*\right)(\boldsymbol{t}_i)\right| \quad \text{and} \quad \left|\left(w - h_*\right)(\boldsymbol{t}_i)\right|,$$

which we visualize in a two-dimensional contour plot.

4.2.1. Interpolation

We first compare the results for two-layer interpolation, see (Int), with the results for single-layer interpolation, see (1). To this end, we choose an outer Matérn kernel $K = K_{\mathrm{TensorMatérn},s}$ with $s = 1$ and an inner polynomial kernel $K_{\mathcal{I}} = K_{\mathrm{Poly},p}$ with $p = 1$ or $p = 2$. In Figure 3 we display the pointwise errors. We observe that there is a visible improvement in the error when dealing with two-layer interpolation instead of single-layer interpolation. While the benefits of two-layer interpolation are already observable for the test function h_2, they become even more obvious for h_1. As explained in the beginning of Section 3, the fact that the kink of h_1 is not parallel to a coordinate axis poses a problem when dealing with the tensor-product kernel. Since a linear transformation (rotation) would suffice to remedy this problem, the polynomial kernel of degree $p = 1$ already suffices to obtain a better error behavior. Therefore, $p = 2$ can already lead to a small overfitting effect as we observe in Figure 3. Nevertheless, the error is still significantly better than in the single-layer case. In the case of h_2, however, we have a jump along two nonlinear curves. Here, $p = 2$ seems to be more appropriate to deal with this problem. Overall, we come to the conclusion that interpolation in reproducing kernel Hilbert spaces can significantly benefit from a two-layer approach if the reproducing kernel at hand does not suit the underlying function.

4.2.2. Regression

Now we have a look at solving the least-squares regression problem (Reg). To determine the optimal parameters λ and μ, we run a 5-fold cross-validation on the input data for all possible choices $\lambda, \mu \in \{2^{-2t+1} \mid t = 1, \ldots, 10\}$. Subsequently, we use the parameter pair (λ, μ) for which the smallest function value of (Reg) is achieved and run the regression

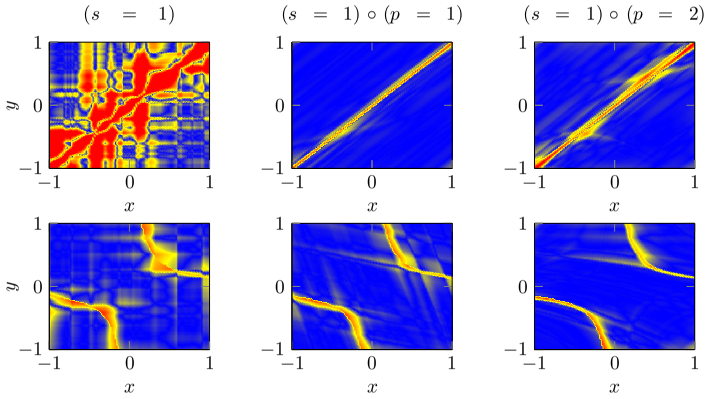


Figure 3: The pointwise error for standard interpolation with $w = f^*_{X,Y}$ (left) and for concatenated interpolation with outer kernel $K_{\text{TensorMatérn},1}$ and inner kernel $K_{\text{Poly},1}$ (mid) or $K_{\text{Poly},2}$ (right), respectively. We plotted both, the error for h_1 (top) and h_2 (bottom). The color scale ranges from blue (0% error) to red (more than 10% error), where the percentage has to be understood with respect to the $\|\cdot\|_{L_\infty}$ norm of h_1 or h_2, respectively.

algorithm on the whole input data set to obtain our final results. We compare the two-layer case with the single-layer regression, see also (5), with the parameter λ, which achieves the smallest error, i.e. we compare to the best possible single-layer solution.

Since the results for interpolation and least-squares regression with the same kernel choices as above happen to be similar, we employ an outer kernel of Gaussian type $K = K_{\text{Gauss},\sigma}$ with $\sigma = 0.1$ instead of Matérn type here. For the inner kernel we again choose $K_{\mathcal{I}} = K_{\text{Poly},p}$ with $p = 1, 2$. As we observe in Figure 4, there is a significant improvement of the two-layer approach over the single-layer one. Note that we deliberately employ the kernel width $\sigma = 0.1$, which appears to be too small for single-layer regression. However, the two-layer approach seems to remedy this bad choice automatically by adjusting the inner transformation accordingly. In this regard, the algorithm can also be understood as an implicit hyperparameter tuner.

4.2.3. Linear outer kernel

In this section, we again want to emphasize the difference of our approach, which allows for nonlinear outer kernels, to the MKL-type RLS2 algorithm of Dinuzzo (2011), where only a linear outer kernel is considered and the inner kernel is given by a diagonal matrix with its entries being different scalar-valued (nonlinear) kernels. To this end, we run our two-layer least-squares regression approach for the following two settings:

(1) Outer polynomial kernel $K = K_{\text{Poly},1}$ of order 1, inner mixture kernel $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{y})$,

(2) Outer Matérn kernel $K = K_{\text{TensorMatérn},1}$ of order 1, inner mixture kernel $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{y})$.

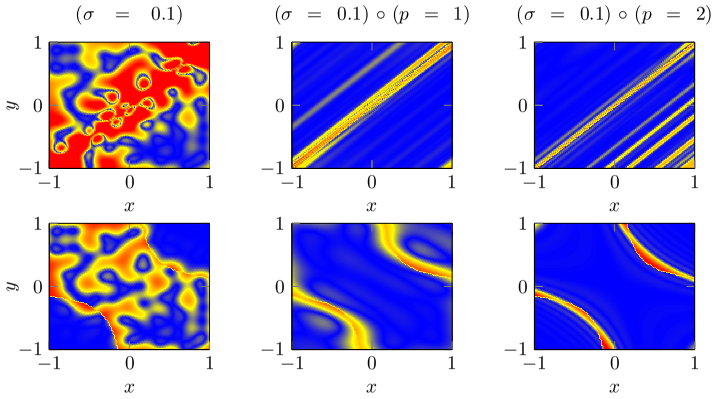


Figure 4: The pointwise error for standard least-squares with $w = f^{\lambda}_{X,Y}$ (left) and for concatenated least-squares with outer kernel $K_{\text{Gauss},0.1}$ and inner kernel $K_{\text{Poly},1}$ (mid) or $K_{\text{Poly},2}$ (right), respectively. We plotted both, the error for h_1 (top) and h_2 (bottom). The color scale ranges from blue (0% error) to red (more than 10% error), where the percentage has to be understood with respect to the $\|\cdot\|_{L_\infty}$ norm of h_1 or h_2, respectively.

For the inner mixture kernel, we deviate from (20) and from $D = 2$ here. To this end, we set $D = 5$ and use a diagonal kernel $\boldsymbol{K}$ with different scalar-valued kernels as entries. For the five scalar-valued kernels we choose three Gaussian kernels $K_{\text{Gauss},\sigma}$ with $\sigma = 0.1, 1, 10$ and two polynomial kernels $K_{\text{Poly},p}$ with $p = 1, 2$. Setting (1) serves to represent the RLS2 algorithm[4], where similar choices for the inner kernel have been made, see Dinuzzo (2011). To determine the optimal parameters $\lambda, \mu \in \{10^{-2t+1} \mid t = 1, \ldots, 6\}$, we again run a 5-fold crossvalidation[5]. The results can be found in figure 5. As we have already seen for interpolation, the structure of the function h_1 admits a good representation by a two-layer kernel discretization of type (2). However, despite the quite generic choice of the inner kernel in setting (1), the two-layer kernel approach with a linear outer kernel is not able to find a good representation of the function. This shows that a nonlinear choice for the outer kernel can be necessary to find suitable approximations by the two-layer algorithm. Although the results do not differ that much for h_2, we again see that there is a slight advantage in approximating with a nonlinear outer kernel.

4. Note however that we did not use a diagonal scaling of the linear kernel and our optimization algorithm is different from the one used in Dinuzzo (2011), which is adjusted to the problem with a linear outer kernel.

5. Note that we scan a coarser (but wider) range than in the previous section, which seemed to be appropriate here.

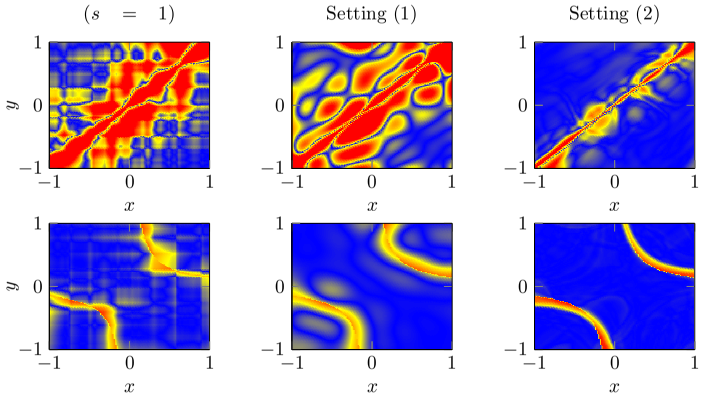


Figure 5: The pointwise error for standard least-squares with Matérn kernel and $w = f^{\lambda}_{X,Y}$ (left) and for concatenated least-squares with setting (1) (mid) and setting (2) (right) from section 4.2.3. We plotted both, the error for h_1 (top) and h_2 (bottom). The color scale ranges from blue (0% error) to red (more than 10% error), where the percentage has to be understood with respect to the $\|\cdot\|_{L_\infty}$ norm of h_1 or h_2, respectively.

4.3. Transformation by the inner function

To get a better impression on how the two-layer algorithms work, we exemplarily inspect the inner function $\boldsymbol{g}$ in the case of interpolation with $K = K_{\text{TensorMatérn},s}$ for $s = 1$ and $K_{\mathcal{I}} = K_{\text{Poly},p}$ for $p = 1$ or $p = 2$, i.e. for the setting from Section 4.2.1. To this end, we depict isotropic grid points in $\Omega = [-1, 1]^2$ and have a look at how these points are transformed by $\boldsymbol{g}$ in Figure 6.

We observe that for h_1 in both cases $p = 1$ and $p = 2$, the inner function aligns the kink almost perpendicular to the y-axis. Therefore, one can easily characterize the kink by the y-coordinate after the inner transformation. This reduces the original two-dimensional kink description $x - y = 0$ to just the one-dimensional description $y = 0$. While the function with the kink along the diagonal does not reside in the tensor-product Matérn space of order 1, which corresponds to the outer kernel in this example, a function with a kink parallel to one of the coordinate axes does. Therefore, the inner function $\boldsymbol{g}$ transforms the domain in such a way that the result resides in the RKHS to which the outer function belongs.

Considering the test function h_2, we see that a linear inner transformation, i.e. $p = 1$, essentially just rotates and shears the domain and does not change the alignment of the jump very much. However, in the case $p = 2$, the inner function $\boldsymbol{g}$ manages to transform the domain in such a way that the jump is now almost parallel to the y-axis. We observe that the pointwise errors in Figure 3 really benefit from this transformation and the jump is resolved almost perfectly. Overall, we see that the inner function $\boldsymbol{g}$ tries to align the features of the original test function in such a way that they can be easily resolved by the outer function f.

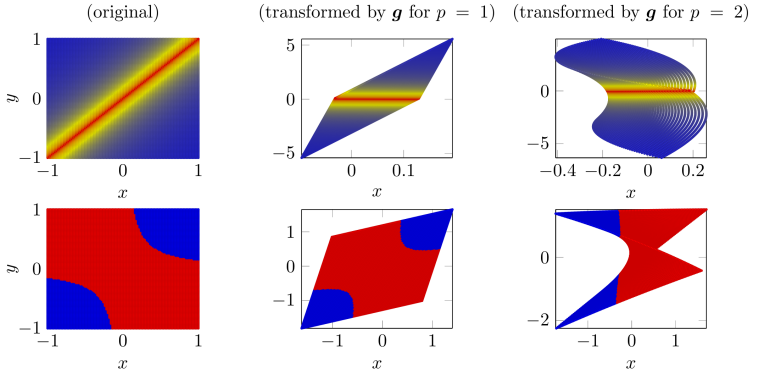


Figure 6: The transformation of the isotropic grid points (left) by the inner function with $p = 1$ (mid) and $p = 2$ (right). The underlying problem is interpolation of h_1 (top) and h_2 (bottom) for the outer kernel $K_{\text{TensorMatérn},1}$ and the inner kernel $K_{\text{Poly},p}$. The color scale represents the values of h_1 or h_2, respectively.

5. Conclusion

In this paper, we presented both a finite- and an infinite-sample representer theorem for concatenated machine learning problems. In the finite-sample case, the statement essentially boils down to the fact that the a priori infinite-dimensional optimization problem, which appears when dealing with function compositions from reproducing kernel Hilbert spaces, can be recast into a finite-dimensional optimization problem, where we only have to deal with at most N kernel translates in each layer of the composition. Here, N denotes the number of input data points. In the infinite-sample case, we derived an analogous result stating that the solution in each layer is an element of the image space of the integral operator defined by the corresponding kernel evaluated at the innermost functions. We introduced a simple neural network architecture, which represents the concatenated functions we are dealing with. Furthermore, we established a connection between our representer theorem and two types of state-of-the-art deep learning algorithms, namely multi-layer multiple kernel learning and deep kernel networks. Finally, we presented a detailed analysis on a two-layer interpolation and a two-layer least-squares regression algorithm, which can directly be derived from our representer theorem. We illustrated the operating principles of these algorithms with the help of two artificial test functions and explained why the two-layer approach is able to remedy the shortcomings of a single-layer variant. Furthermore, we highlighted that the use of a nonlinear outer kernel, instead of a linear one as in Dinuzzo (2011), can be inevitable to obtain good two-layer approximations. Nevertheless, the nonlinearity of the outer layer makes the numerical treatment of the underlying optimization problem more difficult.

While we presented specific two-layer ($L = 2$) algorithms and applied them to two-dimensional ($d = 2$) toy problems for illustrative reasons, our representer theorems can also be applied in the high-dimensional case with an arbitrary number of layers. Note

furthermore that, apart from interpolation and least-squares regression, also more general choices of the loss function $\mathcal{L}$ and the regularizers Θ_l are allowed in (7). Therefore, one can also think of multi-layer support vector machines for instance. The construction of such efficient deep kernel learning algorithms for high-dimensional problems and a thorough analysis of the interplay between the number of layers L and the dimension d will be future work.

Acknowledgments

The authors want to thank the anonymous referees for their suggestions and remarks and especially for pointing out the relation to Dinuzzo (2011). The authors were partially supported by the Sonderforschungsbereich 1060 *The Mathematics of Emergent Effects* funded by the Deutsche Forschungsgemeinschaft.

Appendix A. Remainder of the proof of theorem 2

To continue the proof of theorem 2, we note that we already showed that f_1 has the desired structure (14). Let us assume we have shown (14) for all $f_1, \dots, f_{l-1}$ for an $l \in \{2, \dots, L\}$. To obtain (14) for f_l, we proceed in the same fashion as in the first part of the proof in section 3.2. To this end, we now define $J_{g_{l+1},\dots,g_L} : \mathcal{H}_l \to [0,\infty)$ by

$$J_{g_{l+1},\dots,g_L}(g_l) := \int_{R_{l+1}\times R_1} \tilde{\mathcal{L}}_l\left(y, g_l(\boldsymbol{\xi})\right) \, \mathrm{d}G_{l,\star}(\mathbb{P})(\boldsymbol{\xi}, y) + \lambda_l \|g_l\|^2_{\mathcal{H}_l},$$

where $\tilde{\mathcal{L}}_l(y, \boldsymbol{z}) := \mathcal{L}(y, f_1 \circ \ldots \circ f_{l-1}(\boldsymbol{z}))$ and $G_{l,\star}(\mathbb{P})$ is the pushforward of $\mathbb{P}$ onto $R_{l+1} \times R_1$ defined by $G_l(\boldsymbol{x}, y) = (g_{l+1} \circ \ldots \circ g_L(\boldsymbol{x}), y)$. Then it holds

$$\begin{aligned}
&\min_{g_l \in \mathcal{H}_l, \dots, g_L \in \mathcal{H}_L} J(f_1, f_2, \dots, f_{l-1}, g_l, g_{l+1}, \dots, g_L) \\
&= \min_{g_{l+1} \in \mathcal{H}_{l+1}, \dots, g_L \in \mathcal{H}_L} \left(\min_{g_l \in \mathcal{H}_l} J_{g_{l+1},\dots,g_{l+1}}(g_l) \right) + \sum_{i=l+1}^{L} \lambda_i \|g_i\|^2_{\mathcal{H}_i}
\end{aligned}$$

and we need to show that a minimizer of $J_{g_{l+1},\dots,g_L}$ admits a representation of type (14). To this end, we begin by defining a Nemitski vector loss function and we subsequently prove that these loss functions admit the representation we need.

Definition 5 *Let $\mathcal{L} : R_1 \times D \to [0,\infty)$ for some domain $D \subset \mathbb{R}^d$. Let $\mathbb{P}_{R_1}$ denote the marginal distribution of $\mathbb{P}$ w.r.t. the second variable. We call $\mathcal{L}$ a $\mathbb{P}$-integrable Nemitski vector loss, if there exist $b : R_1 \to [0,\infty)$ with $b \in L_{1,\mathbb{P}_{R_1}}(R_1)$ and a measurable, increasing $h : [0,\infty) \to [0,\infty)$ such that*

$$\mathcal{L}(y, \boldsymbol{z}) \le b(y) + h(\|\boldsymbol{z}\|) \quad \text{for all } (y, \boldsymbol{z}) \in R_1 \times D.$$

If $\mathcal{L}$ is k-times differentiable w.r.t. the second variable for all $y \in R_1$, we call it a k-times differentiable Nemitski vector loss.

Lemma 6 *Let $l \in \{2, \dots, L\}$ and let $\mathbb{P}^l$ be a distribution[6] on $R_{l+1} \times R_1$ and let $\mathcal{L}^\star$ be a $\mathbb{P}^l$-integrable and 1-differentiable Nemitski vector loss on $R_1 \times R_l$ such that the derivative w.r.t. the second argument $\nabla_2 \mathcal{L}$ fulfills*

$$\|\nabla_2 \mathcal{L}^\star(y, \boldsymbol{z})\| \le b^\star(y) + h^\star(\|\boldsymbol{z}\|) \quad \text{for all } (y, \boldsymbol{z}) \in R_1 \times R_l$$

for some $b^\star \in L_{1,\mathbb{P}^l_{R_1}}(R_1)$ and a measurable, increasing $h^\star : [0,\infty) \to [0,\infty)$. Then, the functional $\mathcal{R}_{l,\mathbb{P}^l} : \mathcal{H}_l \to [0,\infty)$ defined by

$$\mathcal{R}_{l,\mathbb{P}^l}(f) := \int_{R_{l+1}\times R_1} \mathcal{L}^\star(y, f(\boldsymbol{z})) \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y)$$

is Frechet differentiable and the derivative $\mathrm{d}\mathcal{R}_{l,\mathbb{P}^l} : \mathcal{H}_l \to \mathcal{B}(\mathcal{H}_l, \mathbb{R})$ is given by

$$\mathrm{d}\mathcal{R}_{l,\mathbb{P}^l}(f)(g) = \int_{R_{l+1}\times R_1} \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z}) \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y). \tag{21}$$

Furthermore, a critical point of $J^\star : \mathcal{H}_l \to [0,\infty)$ defined by

$$J^\star(f) := \mathcal{R}_{l,\mathbb{P}^l}(f) + \lambda_l \|f\|^2_{\mathcal{H}_l}$$

is given by

$$f(\cdot) = -\frac{1}{2\lambda_l} \int_{R_{l+1}\times R_1} K_l(\cdot, \boldsymbol{z}) \cdot \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z})) \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y). \tag{22}$$

Proof We have

$$\begin{aligned}
&\lim_{\|g\|_{\mathcal{H}_l} \to 0} \frac{\mathcal{R}_{l,\mathbb{P}^l}(f+g) - \mathcal{R}_{l,\mathbb{P}^l}(f) - \int_{R_{l+1}\times R_1} \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z}) \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y)}{\|g\|_{\mathcal{H}_l}} \\
&= \lim_{\|g\|_{\mathcal{H}_l} \to 0} \int_{R_{l+1}\times R_1} \frac{\mathcal{L}^\star(y, f(\boldsymbol{z}) + g(\boldsymbol{z})) - \mathcal{L}^\star(y, f(\boldsymbol{z})) - \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})}{\|g\|_{\mathcal{H}_l}} \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y) \\
&\overset{(*)}{=} \int_{R_{l+1}\times R_1} \lim_{\|g\|_{\mathcal{H}_l} \to 0} \frac{\mathcal{L}^\star(y, f(\boldsymbol{z}) + g(\boldsymbol{z})) - \mathcal{L}^\star(y, f(\boldsymbol{z})) - \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})}{\|g\|_{\mathcal{H}_l}} \, \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y) \\
&= 0,
\end{aligned}$$

where the last equation follows from the differentiability of $\mathcal{L}^\star$ and $(*)$ follows by the dominated convergence theorem since the integrand is bounded by

$$\begin{aligned}
&\left| \frac{\mathcal{L}^\star(y, f(\boldsymbol{z}) + g(\boldsymbol{z})) - \mathcal{L}^\star(y, f(\boldsymbol{z})) - \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})}{\|g\|_{\mathcal{H}_l}} \right| \\
&= \left| \frac{\nabla_2 \mathcal{L}^\star(y, cf(\boldsymbol{z}) + (1-c)g(\boldsymbol{z}))^T \cdot g(\boldsymbol{z}) - \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})}{\|g\|_{\mathcal{H}_l}} \right| \\
&\le 2b^\star(y) + h^\star(\|cf(\boldsymbol{z}) + (1-c)g(\boldsymbol{z})\|) + h^\star(\|f(\boldsymbol{z})\|)
\end{aligned}$$

6. Note that we set $R_{L+1} := D_L = \Omega$.

for some $c \in [0,1]$ due to the mean value theorem. Since the last line is bounded by $2b^\star(y) + 2h^\star(\|f(\boldsymbol{z})\| + 1)$ independently of g for any g with $\|g\|_{\mathcal{H}_l} \leq 1$, the dominated convergence theorem can be applied, which proves (21).

Since a critical point f of $J^\star$ fulfills

$$0 = \mathrm{d}J^\star(f)(g) = \mathrm{d}\mathcal{R}_{l,\mathbb{P}^l}(f)(g) + 2\lambda_l \langle f, g\rangle_{\mathcal{H}_l},$$

for all $g \in \mathcal{H}_l$, we obtain

$$\begin{aligned}
\langle f, g\rangle_{\mathcal{H}_l} =& -\frac{1}{2\lambda_l}\mathrm{d}\mathcal{R}_{l,\mathbb{P}^l}(f)(g) \\
=& -\frac{1}{2\lambda_l}\int_{R_{l+1}\times R_1} \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot g(\boldsymbol{z})\ \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y) \\
=& -\frac{1}{2\lambda_l}\int_{R_{l+1}\times R_1} \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z}))^T \cdot \sum_{i=1}^{d_l} \langle g, K_l(\cdot, \boldsymbol{z})\boldsymbol{e}_i\rangle_{\mathcal{H}_l} \cdot \boldsymbol{e}_i\ \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y) \\
=& -\frac{1}{2\lambda_l}\sum_{i=1}^{d_l}\left\langle \int_{R_{l+1}\times R_1} \nabla_2 \mathcal{L}^\star(y, f(\boldsymbol{z}))^T K_l(\cdot, \boldsymbol{z})\boldsymbol{e}_i\ \mathrm{d}\mathbb{P}^l(\boldsymbol{z}, y), g\right\rangle_{\mathcal{H}_l} \cdot \boldsymbol{e}_i
\end{aligned}$$

with the reproducing property of K_l, which is equivalent to the Bochner-type integral formulation (22). This finishes the proof. ■

Now, we can apply lemma 6 with $\mathbb{P}^l = G_{l,\star}(\mathbb{P})$ and $\mathcal{L}^\star = \tilde{\mathcal{L}}_l$, which shows that a critical point $g_l^\star$ of $J_{g_{l+1},\ldots,g_L}$ can be written as

$$\begin{aligned}
g_l^\star(\cdot) =& -\frac{1}{2\lambda_l}\int_{R_{l+1}\times R_1} K_l(\cdot, \boldsymbol{\xi}) \cdot \nabla_2 \tilde{\mathcal{L}}_l(y, g_l^\star(\boldsymbol{\xi}))\ \mathrm{d}G_{l,\star}(\mathbb{P})(\boldsymbol{\xi}, y) \qquad (23) \\
=& -\frac{1}{2\lambda_l}\int_{\Omega\times R_1} K_l(\cdot, g_{l+1}\circ\ldots\circ g_L(\boldsymbol{x})) \cdot \nabla_2 \tilde{\mathcal{L}}_l(y, g_l^\star \circ g_{l+1}\circ\ldots\circ g_L(\boldsymbol{x}))\ \mathrm{d}\mathbb{P}(\boldsymbol{x}, y),
\end{aligned}$$

which is of type (14). Therefore, it just remains to show that $\tilde{\mathcal{L}}_l$ fulfills the prerequisites of lemma 6 and that $A_{f_l, f_{l+1},\ldots,f_L}(\boldsymbol{x}, y) := \nabla_2 \tilde{\mathcal{L}}_l(y, f_l \circ f_{l+1}\circ\ldots\circ f_L(\boldsymbol{x})) \in L_{1,\mathbb{P}}$.

Lemma 7 *$\tilde{\mathcal{L}}_l$ is a $G_{l,\star}(\mathbb{P})$-integrable and 1-differentiable Nemitski loss and the derivative w.r.t. the second argument fulfills*

$$\left\|\nabla_2 \tilde{\mathcal{L}}_l(y, \boldsymbol{z})\right\| \leq \tilde{b}(y) + \tilde{h}(\|\boldsymbol{z}\|) \quad \text{for all } (y, \boldsymbol{z}) \in R_1 \times R_l \qquad (24)$$

for a $\tilde{b} \in L_{1,G_{l,\star}(\mathbb{P})_{R_1}}(R_1)$ and a measurable, increasing $\tilde{h} : [0,\infty) \to [0,\infty)$.

Proof Since

$$|\tilde{\mathcal{L}}_l(y, \boldsymbol{z})| = |\mathcal{L}(y, f_1\circ\ldots\circ f_{l-1}(\boldsymbol{z}))| \leq b_0(y) + h_0(f_1\circ\ldots\circ f_{l-1}(\boldsymbol{z})) \leq b_0(y) + h_0(\|f_1\|_\infty),$$

$\tilde{\mathcal{L}}_l$ is a $G_{l,\star}(\mathbb{P})$-integrable Nemitski-loss. Here, we again used that $f_1 \in \mathcal{H}_1 \hookrightarrow C(D_1)$ because of (12). Since $f_1, \ldots, f_{l-1}$ are differentiable because the respective kernels are in C^1, $\tilde{\mathcal{L}}_l$ is

also 1-differentiable by the chain rule. It remains to show (24). To this end, note that the chain rule gives us

$$\begin{aligned}
&\nabla_2 \tilde{\mathcal{L}}_l(y, \boldsymbol{z})(\cdot) = \frac{\partial}{\partial \boldsymbol{z}}\left(\mathcal{L}\left(y, f_1\circ\ldots\circ f_{l-1}(\boldsymbol{z})\right)\right) \\
&= \mathcal{L}^{(1)}(y, f_1\circ\ldots\circ f_{l-1}(\boldsymbol{z})) \cdot \mathrm{d}f_1(f_2\circ\ldots\circ f_{l-1}(\boldsymbol{z}))\left(\mathrm{d}f_2(f_3\circ\ldots\circ f_{l-1}(\boldsymbol{z}))\left(\ldots \mathrm{d}f_{l-1}(\boldsymbol{z})(\cdot)\right)\right),
\end{aligned}$$

which leads to

$$\begin{aligned}
\|\nabla_2 \tilde{\mathcal{L}}_l(y, \boldsymbol{z})\| \leq& \left(b_1(y) + h_1(|f_1\circ\ldots\circ f_{l-1}(\boldsymbol{z})|)\right) \cdot \prod_{i=1}^{l-1}\sup_{\boldsymbol{x}_i \in D_i} \|\mathrm{d}f_i(\boldsymbol{x}_i)\|_{\mathcal{B}(D_i,R_i)} \\
\leq& \left(b_1(y) + h_1(\|f_1\|_\infty)\right) \cdot \prod_{i=1}^{l-1}\sup_{\boldsymbol{x}_i \in D_i} \|\mathrm{d}f_i(\boldsymbol{x}_i)\|_{\mathcal{B}(D_i,R_i)}. \qquad (25)
\end{aligned}$$

Because of our assumption that we already showed (14) for $f_1, \ldots, f_{l-1}$ and because of (12), we get by the dominated convergence theorem that

$$\sup_{\boldsymbol{x}_i \in D_i} \|\mathrm{d}f_i(\boldsymbol{x}_i)\|_{\mathcal{B}(D_i,R_i)} \leq \frac{1}{2\lambda_i} c_i \|A_{f_i,\ldots,f_L}\|_{L_{1,\mathbb{P}}}$$

for all $i = 1, \ldots, l-1$. Therefore, by setting $\tilde{b}(y) := c \cdot b_1(y)$ and choosing a constant $\tilde{h} := c \cdot h_1(\|f_1\|_\infty)$ with $c := \prod_{i=1}^{l-1} \frac{1}{2\lambda_i} c_i \|A_{f_i,\ldots,f_L}\|_{L_{1,\mathbb{P}}} < \infty$, we obtain (24). ■

Applying lemma 6 and lemma 7 shows us that f_l fulfills the integral equation (23). To conclude the proof of theorem 2, we note that

$$A_{f_l, f_{l+1},\ldots,f_L}(\boldsymbol{x}, y) := \nabla_2 \tilde{\mathcal{L}}_l(y, f_l \circ f_{l+1}\circ\ldots\circ f_L(\boldsymbol{x})) \in L_{1,\mathbb{P}},$$

which directly follows from (25) and the fact that $b_1 \in L_{1,\mathbb{P}_{R_1}}$. This finally shows that f_l admits a representation of type (14). Since the argument is valid for each $l = 2, \ldots, L$ and we already proved (14) for $l = 1$ in section 3.2, this finishes the proof of theorem 2.

References

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the 21st International Conference on Machine Learning*, pages 1–9, 2004.

B. Bohn and M. Griebel. Error estimates for multivariate regression on discretized function spaces. *SIAM Journal on Numerical Analysis*, 55(4):1843–1866, 2017.

H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numerica*, 13:147–269, 2004.

Y. Cho and L. Saul. Kernel methods for deep learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22, pages 342–350. Curran Associates, Inc., 2009.

G. Cybenko. Approximations by superpositions of sigmoidal functions. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, 1989.

A. Damianou and N. Lawrence. Deep Gaussian processes. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, pages 207–215, 2013.

S. de Marchi and R. Schaback. Stability of kernel-based interpolation. *Adv. Comput. Math.*, 32:155–161, 2010.

F. Dinuzzo. *Learning functions with kernel methods*. PhD thesis, University of Pavia, Pavia, Italy, 2011.

G. Fasshauer and Q. Ye. Reproducing kernels of generalized Sobolev spaces via a Green function approach with distributional operators. *Numerische Mathematik*, 119(3):585–611, 2011.

M. Gönen and E. Alpaydin. Multiple kernel learning algorithms. *JMLR*, 12:2211–2268, 2011.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

M. Griebel and H. Harbrecht. Approximation of bi-variate functions: singular value decomposition versus sparse grids. *IMA J. Numer. Anal.*, 34(1):28–54, 2014.

A. Hinrichs, L. Markhasin, J. Oettershagen, and T. Ullrich. Optimal quasi-Monte Carlo rules on higher order digital nets for the numerical integration of multivariate periodic functions. *Numerische Mathematik*, 134(1):163–196, 2016.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

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

S. Mallat. Understanding deep convolutional networks. *Phil. Trans. R. Soc. A*, 374(2065), 2016.

H. Mhaskar, Q. Liao, and T. Poggio. When and why are deep networks better than shallow ones? In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 2343–2349, 2017.

C. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17: 177–204, 2005.

G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller. Explaining nonlinear classification decisions with deep Taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

I. Rebai, Y. Benayed, and W. Mahdi. Deep multilayer multiple kernel learning. *Neural Computing and Applications*, 27(8):2305–2314, 2016.

S. Reddi, S. Sra, B. Poczos, and A. Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1145–1153. Curran Associates, Inc., 2016.

B. Schölkopf and A. Smola. *Learning with Kernels – Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press – Cambridge, Massachusetts, 2002.

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

E. Strobl and S. Visweswaran. Deep multiple kernel learning. In *Proceedings of the 12th International Conference on Machine Learning and Applications*, pages 414–417, 2013.

H. Wendland. *Scattered Data Approximation*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2005.

A. Wilson, Z. Hu, R. Salakhutdinov, and E. Xing. Deep kernel learning. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 370–378, 2016.

J. Zhuang, I. Tsang, and S. Hoi. Two-layer multiple kernel learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pages 909–917, 2011.

Journal of Machine Learning Research 20 (2019) 1-50 Submitted 11/17; Revised 3/19; Published 4/19

Active Learning for Cost-Sensitive Classification

Akshay Krishnamurthy AKSHAY@CS.UMASS.EDU
Microsoft Research
New York, NY 10011

Alekh Agarwal ALEKHA@MICROSOFT.COM
Microsoft Research
Redmond, WA 98052

Tzu-Kuo Huang TKHUANG@PROTONMAIL.COM
Uber Advanced Technology Center
Pittsburgh, PA 15201

Hal Daumé III HAL@UMIACS.UMD.EDU
Microsoft Research
New York, NY 10011

John Langford JCL@MICROSOFT.COM
Microsoft Research
New York, NY 10011

Editor: Sanjoy Dasgupta

Abstract

We design an active learning algorithm for cost-sensitive multiclass classification: problems where different errors have different costs. Our algorithm, COAL, makes predictions by regressing to each label's cost and predicting the smallest. On a new example, it uses a set of regressors that perform well on past data to estimate possible costs for each label. It queries only the labels that *could be* the best, ignoring the sure losers. We prove COAL can be efficiently implemented for any regression family that admits squared loss optimization; it also enjoys strong guarantees with respect to predictive performance and labeling effort. We empirically compare COAL to passive learning and several active learning baselines, showing significant improvements in labeling effort and test cost on real-world datasets.

Keywords: Active Learning, Cost-sensitive Learning, Structured Prediction, Statistical Learning Theory, Oracle-based Algorithms.

1. Introduction

The field of active learning studies how to efficiently elicit relevant information so learning algorithms can make good decisions. Almost all active learning algorithms are designed for binary classification problems, leading to the natural question: How can active learning address more complex prediction problems? Multiclass and importance-weighted classification require only minor modifications but we know of no active learning algorithms that enjoy theoretical guarantees for more complex problems.

One such problem is cost-sensitive multiclass classification (CSMC). In CSMC with K classes, passive learners receive input examples x and cost vectors $c \in \mathbb{R}^K$, where $c(y)$ is the cost of predicting label y on x.[1] A natural design for an *active* CSMC learner then is to adaptively query the costs of only a (possibly empty) subset of labels on each x. Since measuring label complexity is more nuanced in CSMC (e.g., is it more expensive to query three costs on a single example or one cost on three examples?), we track both the number of examples for which at least one cost is queried, along with the total number of cost queries issued. The first corresponds to a fixed human effort for inspecting the example. The second captures the additional effort for judging the cost of each prediction, which depends on the number of labels queried. (By querying a label, we mean querying the cost of predicting that label given an example.)

In this setup, we develop a new active learning algorithm for CSMC called Cost Overlapped Active Learning (COAL). COAL assumes access to a set of regression functions, and, when processing an example x, it uses the functions with good past performance to compute the range of possible costs that each label might take. Naturally, COAL only queries labels with large cost range, akin to uncertainty-based approaches in active regression (Castro et al., 2005), but furthermore, it only queries labels that could possibly have the smallest cost, avoiding the uncertain, but surely suboptimal labels. The key algorithmic innovation is an efficient way to compute the cost range realized by good regressors. This computation, and COAL as a whole, only requires that the regression functions admit efficient squared loss optimization, in contrast with prior algorithms that require 0/1 loss optimization (Beygelzimer et al., 2009; Hanneke, 2014).

Among our results, we prove that when processing n (unlabeled) examples with K classes and a regression class with pseudo-dimension d (See Definition 1),

1. The algorithm needs to solve $\mathcal{O}(Kn^5)$ regression problems over the function class (Corollary 4). Thus COAL runs in polynomial time for convex regression sets.

2. With no assumptions on the noise in the problem, the algorithm achieves generalization error $\tilde{\mathcal{O}}(\sqrt{Kd/n})$ and requests $\tilde{\mathcal{O}}(n\theta_2\sqrt{Kd})$ costs from $\tilde{\mathcal{O}}(n\theta_1\sqrt{Kd})$ examples (Theorems 5 and 9) where θ_1, θ_2 are the disagreement coefficients (Definition 8)[2]. The worst case offers minimal improvement over passive learning, akin to active learning for binary classification.

3. With a Massart-type noise assumption (Assumption 3), the algorithm has generalization error $\tilde{\mathcal{O}}(Kd/n)$ while requesting $\tilde{\mathcal{O}}(Kd(\theta_2 + K\theta_1)\log n)$ labels from $\tilde{\mathcal{O}}(Kd\theta_1 \log n)$ examples (Corollary 6, Theorem 10). Thus under favorable conditions, COAL requests *exponentially fewer* labels than passive learning.

We also derive generalization and label complexity bounds under a milder Tsybakov-type noise condition (Assumption 4). Existing lower bounds from binary classification (Hanneke, 2014) suggest that our results are optimal in their dependence on n, although these lower bounds do not directly apply to our setting. We also discuss some intuitive examples highlighting the benefits of using COAL.

CSMC provides a more expressive language for success and failure than multiclass classification, which allows learning algorithms to make the trade-offs necessary for good performance and broadens potential applications. For example, CSMC can naturally express

1. Cost here refers to prediction cost and not labeling effort or the cost of acquiring different labels.
2. $\tilde{\mathcal{O}}(\cdot)$ suppresses logarithmic dependence on n, K, and d.

©2019 Akshay Krishnamurthy, Alekh Agarwal, Tzu-Kuo Huang, Hal Daumé III, John Langford.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-681.html.

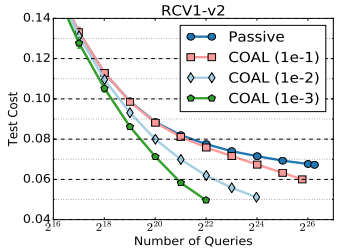


Figure 1: Empirical evaluation of COAL on Reuters text categorization dataset. Active learning achieves *better* test cost than passive, with a factor of 16 fewer queries. See Section 7 for details.

partial failure in hierarchical classification (Silla Jr. and Freitas, 2011). Experimentally, we show that COAL substantially outperforms the passive learning baseline with orders of magnitude savings in the labeling effort on a number of hierarchical classification datasets (see Figure 1 for comparison between passive learning and COAL on Reuters text categorization).

CSMC also forms the basis of learning to avoid cascading failures in joint prediction tasks like structured prediction and reinforcement learning (Daumé III et al., 2009; Ross and Bagnell, 2014; Chang et al., 2015). As our second application, we consider learning to search algorithms for joint or structured prediction, which operate by a reduction to CSMC. In this reduction, evaluating the cost of a class often involves a computationally expensive "roll-out," so using an active learning algorithm inside such a passive joint prediction method can lead to significant computational savings. We show that using COAL within the Aggravate algorithm (Ross and Bagnell, 2014; Chang et al., 2015) reduces the number of roll-outs by a factor of $\frac{1}{4}$ to $\frac{3}{4}$ on several joint prediction tasks.

Our code is publicly available as part of the Vowpal Wabbit machine learning library.[3]

2. Related Work

Active learning is a thriving research area with many theoretical and empirical studies. We recommend the survey of Settles (2012) for an overview of more empirical research. We focus here on theoretical results.

Our work falls into the framework of *disagreement-based active learning*, which studies general hypothesis spaces typically in an agnostic setup (see Hanneke (2014) for an excellent survey). Existing results study binary classification, while our work generalizes to CSMC, assuming that we can accurately predict costs using regression functions from our class. One difference that is natural for CSMC is that our query rule checks the range of predicted costs for a label.

The other main difference is that we use a square loss oracle to search the version space. In contrast, prior work either explicitly enumerates the version space (Balcan et al., 2006; Zhang and Chaudhuri, 2014) or uses a 0/1 loss *classification* oracle for the search (Dasgupta et al., 2007; Beygelzimer et al., 2009, 2010; Huang et al., 2015). In most instantiations, the oracle solves an NP-hard problem and so does not directly lead to an efficient algorithm, although practical implementations using heuristics are still quite effective. Our approach instead uses a squared-loss *regression* oracle, which can be implemented efficiently via convex optimization and leads to a polynomial time algorithm.

In addition to disagreement-based approaches, much research has focused on plug-in rules for active learning in binary classification, where one estimates the class-conditional regression function (Castro and Nowak, 2008; Minsker, 2012; Hanneke and Yang, 2012; Carpentier et al., 2017). Apart from Hanneke and Yang (2012), these works make smoothness assumptions and have a nonparametric flavor. Instead, Hanneke and Yang (2012) assume a calibrated surrogate loss and abstract realizable function class, which is more similar to our setting. While the details vary, our work and these prior results employ the same algorithmic recipe of maintaining an implicit version space and querying in a suitably-defined disagreement region. Our work has two notable differences: (1) our algorithm operates in an oracle computational model, only accessing the function class through square loss minimization problems, (2) our results apply to general CSMC, which exhibit significant differences from binary classification. See Subsection 6.1 for further discussion.

Focusing on linear representations, Balcan et al. (2007); Balcan and Long (2013) study active learning with distributional assumptions, while the selective sampling framework from the online learning community considers adversarial assumptions (Cavallanti et al., 2011; Dekel et al., 2010; Orabona and Cesa-Bianchi, 2011; Agarwal, 2013). These methods use query strategies that are specialized to linear representations and do not naturally generalize to other hypothesis classes.

Supervised learning oracles that solve NP-hard optimization problems in the worst case have been used in other problems including contextual bandits (Agarwal et al., 2014; Syrgkanis et al., 2016) and structured prediction (Daumé III et al., 2009). Thus we hope that our work can inspire new algorithms for these settings as well.

Lastly, we mention that square loss regression has been used to estimate costs for passive CSMC (Langford and Beygelzimer, 2005), but, to our knowledge, using a square loss oracle for active CSMC is new.

Advances over Krishnamurthy et al. (2017). Active learning for CSMC was introduced recently in Krishnamurthy et al. (2017) with an algorithm that also uses cost ranges to decide where to query. They compute cost ranges by using the regression oracle to perform a binary search for the maximum and minimum costs, but this computation results in a sub-optimal label complexity bound. We resolve this sub-optimality with a novel cost range computation that is inspired by the multiplicative weights technique for solving linear programs. This algorithmic improvement also requires a significantly more sophisticated statistical analysis for which we derive a novel uniform Freedman-type inequality for classes with bounded pseudo-dimension. This result may be of independent interest.

Krishnamurthy et al. (2017) also introduce an online approximation for additional scalability and use this algorithm for their experiments. Our empirical results use this same online approximation and are slightly more comprehensive. Finally, we also derive general-

3. `http://hunch.net/~vw`

ization and label complexity bounds for our algorithm in a setting inspired by Tsybakov's low noise condition (Mammen and Tsybakov, 1999; Tsybakov, 2004).

Comparison with Foster et al. (2018). In a follow-up to the present paper, Foster et al. (2018) build on our work with a regression-based approach for contextual bandit learning, a problem that bears some similarities to active learning for CSMC. The results are incomparable due to the differences in setting, but it is worth discussing their techniques. As in our paper, Foster et al. (2018) maintain an implicit version space and compute maximum and minimum costs for each label, which they use to make predictions. They resolve the sub-optimality in Krishnamurthy et al. (2017) with epoching, which enables a simpler cost range computation than our multiplicative weights approach. However, epoching incurs an additional $\log(n)$ factor in the label complexity, and under low-noise conditions where the overall bound is $\mathcal{O}(\text{polylog}(n))$, this yields a polynomially worse guarantee than ours.

3. Problem Setting and Notation

We study cost-sensitive multiclass classification (CSMC) problems with K classes, where there is an instance space $\mathcal{X}$, a label space $\mathcal{Y} = \{1, \ldots, K\}$, and a distribution $\mathcal{D}$ supported on $\mathcal{X} \times [0,1]^K$.[4] If $(x, c) \sim \mathcal{D}$, we refer to c as the *cost-vector*, where $c(y)$ is the cost of predicting $y \in \mathcal{Y}$. A classifier $h : \mathcal{X} \to \mathcal{Y}$ has expected cost $\mathbb{E}_{(x,c)\sim\mathcal{D}}[c(h(x))]$ and we aim to find a classifier with minimal expected cost.

Let $\mathcal{G} \triangleq \{g : \mathcal{X} \mapsto [0,1]\}$ denote a set of base regressors and let $\mathcal{F} \triangleq \mathcal{G}^K$ denote a set of vector regressors where the y^{th} coordinate of $f \in \mathcal{F}$ is written as $f(\cdot; y)$. The set of classifiers under consideration is $\mathcal{H} \triangleq \{h_f \mid f \in \mathcal{F}\}$ where each f defines a classifier $h_f : \mathcal{X} \mapsto \mathcal{Y}$ by

$$h_f(x) \triangleq \operatorname*{argmin}_y f(x; y). \tag{1}$$

When using a set of regression functions for a classification task, it is natural to assume that the expected costs under $\mathcal{D}$ can be predicted by some function in the set. This motivates the following realizability assumption.

Assumption 1 (Realizability) *Define the Bayes-optimal regressor $f^\star$, which has $f^\star(x; y) \triangleq \mathbb{E}_c[c(y)|x], \forall x \in \mathcal{X}$ (with $\mathcal{D}(x) > 0$), $y \in \mathcal{Y}$. We assume that $f^\star \in \mathcal{F}$.*

While $f^\star$ is always well defined, note that the cost itself may be noisy. In comparison with our assumption, the existence of a zero-cost classifier in $\mathcal{H}$ (which is often assumed in active learning) is stronger, while the existence of $h_{f^\star}$ in $\mathcal{H}$ is weaker but has not been leveraged in active learning.

We also require assumptions on the complexity of the class $\mathcal{G}$ for our statistical analysis. To this end, we assume that $\mathcal{G}$ is a compact convex subset of $L_\infty(\mathcal{X})$ with finite pseudo-dimension, which is a natural extension of VC-dimension for real-valued predictors.

Definition 1 (Pseudo-dimension) *The pseudo-dimension* $\text{Pdim}(\mathcal{F})$ *of a function class $\mathcal{F} : \mathcal{X} \to \mathbb{R}$ is defined as the VC-dimension of the set of threshold functions $\mathcal{H}^+ \triangleq \{(x, \xi) \mapsto \mathbf{1}\{f(x) > \xi\} : f \in \mathcal{F}\} \subset \mathcal{X} \times \mathbb{R} \to \{0, 1\}$.*

4. In general, labels just serve as indices for the cost vector in CSMC, and the data distribution is over (x, c) pairs instead of (x, y) pairs as in binary and multiclass classification.

Assumption 2 *We assume that $\mathcal{G}$ is a compact convex set with* $\text{Pdim}(\mathcal{G}) = d < \infty$.

As an example, linear functions in some basis representation, e.g., $g(x) = \sum_{i=1}^d w_i \phi_i(x)$, where weights w_i are bounded in some norm, have pseudodimension d. In fact, our result can be stated entirely in terms of covering numbers, and we translate to pseudo-dimension using the fact that such classes have "parametric" covering numbers of the form $(1/\varepsilon)^d$. Thus, our results extend to classes with "nonparametric" growth rates as well (e.g., Holder-smooth functions), although we focus on the parametric case for simplicity. Note that this is a significant departure from Krishnamurthy et al. (2017), which assumed that $\mathcal{G}$ was finite.

Our assumption that $\mathcal{G}$ is a compact convex set introduces a computational challenging of managing this infinitely large set. To address this challenge, we follow the trend in active learning of leveraging existing algorithmic research on supervised learning (Dasgupta et al., 2007; Beygelzimer et al., 2010, 2009) and access $\mathcal{G}$ exclusively through a *regression oracle*. Given an importance-weighted dataset $D = \{x_i, c_i, w_i\}_{i=1}^n$ where $x_i \in \mathcal{X}, c_i \in \mathbb{R}, w_i \in \mathbb{R}_+$, the regression oracle computes

$$\textsc{Oracle}(D) \in \operatorname*{argmin}_{g\in\mathcal{G}} \sum_{i=1}^n w_i (g(x_i) - c_i)^2. \tag{2}$$

Since we assume that $\mathcal{G}$ is a compact convex set it is amenable to standard convex optimization techniques, so this imposes no additional restriction. However, in the special case of linear functions, this optimization is just least squares and can be computed in closed form. Note that this is fundamentally different from prior works that use a 0/1-loss minimization oracle (Dasgupta et al., 2007; Beygelzimer et al., 2010, 2009), which involves an NP-hard optimization in most cases of interest.

Remark 2 *Our assumption that $\mathcal{G}$ is convex is only for computational tractability, as it is crucial in the efficient implementation of our query strategy, but is not required for our generalization and label complexity bounds. Unfortunately recent guarantees for learning with non-convex classes (Liang et al., 2015; Rakhlin et al., 2017) do not immediately yield efficient active learning strategies. Note also that Krishnamurthy et al. (2017) obtain an efficient algorithm without convexity, but this yields a suboptimal label complexity guarantee.*

Given a set of examples and queried costs, we often restrict attention to regression functions that predict these costs well and assess the uncertainty in their predictions given a new example x. For a subset of regressors $G \subset \mathcal{G}$, we measure uncertainty over possible cost values for x with

$$\gamma(x, G) \triangleq c_+(x, G) - c_-(x, G), \qquad c_+(x, G) \triangleq \max_{g\in G} g(x), \qquad c_-(x, G) \triangleq \min_{g\in G} g(x). \tag{3}$$

For vector regressors $F \subset \mathcal{F}$, we define the *cost range* for a label y given x as $\gamma(x, y, F) \triangleq \gamma(x, G_F(y))$ where $G_F(y) \triangleq \{f(\cdot, y) \mid f \in F\}$ are the base regressors induced by F for y. Note that since we are assuming realizability, whenever $f^\star \in F$, the quantities $c_+(x, G_F(y))$ and $c_-(x, G_F(y))$ provide valid upper and lower bounds on $\mathbb{E}[c(y)|x]$.

To measure the labeling effort, we track the number of examples for which even a single cost is queried as well as the total number of queries. This bookkeeping captures settings

Algorithm 1 Cost Overlapped Active Learning (COAL)

1: Input: Regressors $\mathcal{G}$, failure probability $\delta \leq 1/e$.
2: Set $\psi_i = 1/\sqrt{i}$, $\kappa = 3$, $\nu_n = 324(d\log(n) + \log(8Ke(d+1)n^2/\delta))$.
3: Set $\Delta_i = \kappa \min\{\frac{\nu_n}{i-1}, 1\}$.
4: **for** $i = 1, 2, \ldots, n$ **do**
5: $g_{i,y} \leftarrow \arg\min_{g\in\mathcal{G}} \widehat{R}_i(g; y)$. (See (5)).
6: Define $f_i \leftarrow \{g_{i,y}\}_{y=1}^K$.
7: (Implicitly define) $\mathcal{G}_i(y) \leftarrow \{g \in \mathcal{G}_{i-1}(y) \mid \widehat{R}_i(g; y) \leq \widehat{R}_i(g_{i,y}; y) + \Delta_i\}$.
8: Receive new example x. $Q_i(y) \leftarrow 0, \forall y \in \mathcal{Y}$.
9: **for** every $y \in \mathcal{Y}$ **do**
10: $\widehat{c_+}(y) \leftarrow \textsc{MaxCost}((x, y), \psi_i/4)$ and $\widehat{c_-}(y) \leftarrow \textsc{MinCost}((x, y), \psi_i/4)$.
11: **end for**
12: $Y' \leftarrow \{y \in \mathcal{Y} \mid \widehat{c_-}(y) \leq \min_{y'} \widehat{c_+}(y')\}$.
13: **if** $|Y'| > 1$ **then**
14: $Q_i(y) \leftarrow 1$ if $y \in Y'$ and $\widehat{c_+}(y) - \widehat{c_-}(y) > \psi_i$.
15: **end if**
16: Query costs of each y with $Q_i(y) = 1$.
17: **end for**

where the editorial effort for inspecting an example is high but each cost requires minimal further effort, as well as those where each cost requires substantial effort. Formally, we define $Q_i(y) \in \{0, 1\}$ to be the indicator that the algorithm queries label y on the i^{th} example and measure

$$L_1 \triangleq \sum_{i=1}^{n} \bigvee_{y} Q_i(y), \text{ and } L_2 \triangleq \sum_{i=1}^{n} \sum_{y} Q_i(y). \tag{4}$$

4. Cost Overlapped Active Learning

The pseudocode for our algorithm, Cost Overlapped Active Learning (COAL), is given in Algorithm 1. Given an example x, COAL queries the costs of some of the labels y for x. These costs are chosen by (1) computing a set of good regression functions based on the past data (i.e., the version space), (2) computing the range of predictions achievable by these functions for each y, and (3) querying each y that could be the best label *and* has substantial uncertainty. We now detail each step.

To compute an approximate version space we first find the regression function that minimizes the empirical risk for each label y, which at round i is:

$$\widehat{R}_i(g; y) \triangleq \frac{1}{i-1} \sum_{j=1}^{i-1} (g(x_j) - c_j(y))^2 Q_j(y). \tag{5}$$

Recall that $Q_j(y)$ is the indicator that we query label y on the j^{th} example. Computing the minimizer requires one oracle call. We implicitly construct the version space $\mathcal{G}_i(y)$ in Line 7 as the surviving regressors with low square loss regret to the empirical risk minimizer. The

tolerance on this regret is Δ_i at round i, which scales like $\tilde{\mathcal{O}}(d/i)$, where recall that d is the pseudo-dimension of the class $\mathcal{G}$.

COAL then computes the maximum and minimum costs predicted by the version space $\mathcal{G}_i(y)$ on the new example x. Since the true expected cost is $f^\star(x; y)$ and, as we will see, $f^\star(\cdot; y) \in \mathcal{G}_i(y)$, these quantities serve as a confidence bound for this value. The computation is done by the MaxCost and MinCost subroutines which produce approximations to $c_+(x, \mathcal{G}_i(y))$ and $c_-(x, \mathcal{G}_i(y))$ respectively (See (3)).

Finally, using the predicted costs, COAL issues (possibly zero) queries. The algorithm queries any *non-dominated* label that has a large *cost range*, where a label is non-dominated if its estimated minimum cost is smaller than the smallest maximum cost (among all other labels) and the cost range is the difference between the label's estimated maximum and minimum costs.

Intuitively, COAL queries the cost of every label which cannot be ruled out as having the smallest cost on x, but only if there is sufficient ambiguity about the actual value of the cost. The idea is that labels with little disagreement do not provide much information for further reducing the version space, since by construction all regressors would suffer similar square loss. Moreover, only the labels that could be the best need to be queried at all, since the cost-sensitive performance of a hypothesis h_f depends only on the label that it predicts. Hence, labels that are dominated or have small cost range need not be queried.

Similar query strategies have been used in prior works on binary and multiclass classification (Orabona and Cesa-Bianchi, 2011; Dekel et al., 2010; Agarwal, 2013), but specialized to linear representations. The key advantage of the linear case is that the set $\mathcal{G}_i(y)$ (formally, a different set with similar properties) along with the maximum and minimum costs have closed form expressions, so that the algorithms are easily implemented. However, with a general set $\mathcal{G}$ and a regression oracle, computing these confidence intervals is less straightforward. We use the MaxCost and MinCost subroutines, and discuss this aspect of our algorithm next.

4.1. Efficient Computation of Cost Range

In this section, we describe the MaxCost subroutine which uses the regression oracle to approximate the maximum cost on label y realized by $\mathcal{G}_i(y)$, as defined in (3). The minimum cost computation requires only minor modifications that we discuss at the end of the section.

Describing the algorithm requires some additional notation. Let $\tilde{\Delta}_j \triangleq \Delta_j + \widehat{R}_j(g_{j,y}; y)$ be the right hand side of the constraint defining the version space at round j, where $g_{j,y}$ is the ERM at round j for label y, $\widehat{R}_j(\cdot; y)$ is the risk functional, and Δ_j is the radius used in COAL. Note that this quantity can be efficiently computed since $g_{j,y}$ can be found with a single oracle call. Due to the requirement that $g \in \mathcal{G}_{i-1}(y)$ in the definition of $\mathcal{G}_i(y)$, an equivalent representation is $\mathcal{G}_i(y) = \bigcap_{j=1}^{i} \{g : \widehat{R}_j(g; y) \leq \tilde{\Delta}_j\}$. Our approach is based on the observation that given an example x and a label y at round i, finding a function $g \in \mathcal{G}_i(y)$ which predicts the maximum cost for the label y on x is equivalent to solving the minimization problem:

$$\text{minimize}_{g\in\mathcal{G}} (g(x) - 1)^2 \quad \text{such that} \quad \forall 1 \leq j \leq i, \widehat{R}_j(g; y) \leq \tilde{\Delta}_j. \tag{7}$$

Algorithm 2 MaxCost

Input: (x, y), tolerance TOL, (implicitly) risk functionals $\{\widehat{R}_j(\cdot; y)\}_{j=1}^i$.
Compute $g_{j,y} = \operatorname{argmin}_{g\in\mathcal{G}} \widehat{R}_j(g; y)$ for each j.
Let $\Delta_j = \kappa \min\{\frac{\nu_n}{j-1}, 1\}$, $\tilde{\Delta}_j = \widehat{R}_j(g_{j,y}; y) + \Delta_j$ for each j.
Initialize parameters: $c_\ell \leftarrow 0, c_h \leftarrow 1, T \leftarrow \frac{\log(i+1)(12/\Delta_i)^2}{\text{TOL}^4}, \eta \leftarrow \sqrt{\log(i+1)/T}$.
while $|c_\ell - c_h| > \text{TOL}^2/2$ **do**
  $c \leftarrow \frac{c_h - c_\ell}{2}$
  $\mu^{(1)} \leftarrow \mathbf{1} \in \mathbb{R}^{i+1}$. ▷ Use MW to check feasibility of Program (9).
  for $t = 1, \ldots, T$ **do**
    Use the regression oracle to find

$$g_t \leftarrow \operatorname*{argmin}_{g\in\mathcal{G}} \mu_0^{(t)}(g(x)-1)^2 + \sum_{j=1}^{i} \mu_j^{(t)} \widehat{R}_j(g; y) \tag{6}$$

    If the objective in (6) for g_t is at least $\mu_0^{(t)} c + \sum_{j=1}^i \mu_j^{(t)} \tilde{\Delta}_j$, $c_\ell \leftarrow c$, go to 5.
    Update

$$\mu_j^{(t+1)} \leftarrow \mu_j^{(t)} \left(1 - \eta \frac{\tilde{\Delta}_j - \widehat{R}_j(g_t; y)}{\Delta_j + 1}\right), \quad \mu_0^{(t+1)} \leftarrow \mu_0^{(t)} \left(1 - \eta \frac{c - (g_t(x)-1)^2}{2}\right).$$

  end for
  $c_h \leftarrow c$.
end while
Return $\widehat{c_+}(y) = 1 - \sqrt{c_\ell}$.

Given this observation, our strategy will be to find an approximate solution to the problem (7) and it is not difficult to see that this also yields an approximate value for the maximum predicted cost on x for the label y.

In Algorithm 2, we show how to efficiently solve this program using the regression oracle. We begin by exploiting the convexity of the set $\mathcal{G}$, meaning that we can further rewrite the optimization problem (7) as

$$\text{minimize}_{P\in\Delta(\mathcal{G})} \mathbb{E}_{g\sim P}\left[(g(x)-1)^2\right] \quad \text{such that} \quad \forall 1 \le j \le i, \mathbb{E}_{g\sim P}\left[\widehat{R}_j(g; y)\right] \le \tilde{\Delta}_j. \tag{8}$$

The above rewriting is effectively cosmetic as $\mathcal{G} = \Delta(\mathcal{G})$ by the definition of convexity, but the upshot is that our rewriting results in both the objective and constraints being linear in the optimization variable P. Thus, we effectively wish to solve a linear program in P, with our computational tool being a regression oracle over the set $\mathcal{G}$. To do this, we create a series of feasibility problems, where we repeatedly guess the optimal objective value for the problem (8) and then check whether there is indeed a distribution P which satisfies all the constraints and gives the posited objective value. That is, we check

$$?\exists P \in \Delta(\mathcal{G}) \text{ such that } \mathbb{E}_{g\sim P}(g(x)-1)^2 \le c \text{ and } \forall 1 \le j \le i, \mathbb{E}_{g\sim P}\widehat{R}_j(g; y) \le \tilde{\Delta}_j. \tag{9}$$

If we find such a solution, we increase our guess, and otherwise we reduce the guess and proceed until we localize the optimal value to a small enough interval.

It remains to specify how to solve the feasibility problem (9). Noting that this is a linear feasibility problem, we jointly invoke the Multiplicative Weights (MW) algorithm and the regression oracle in order to either find an approximately feasible solution or certify the problem as infeasible. MW is an iterative algorithm that maintains weights μ over the constraints. At each iteration it (1) collapses the constraints into one, by taking a linear combination weighted by μ, (2) checks feasibility of the simpler problem with a single constraint, and (3) if the simpler problem is feasible, it updates the weights using the slack of the proposed solution. Details of steps (1) and (3) are described in Algorithm 2.

For step (2), the simpler problem that we must solve takes the form

$$?\exists P \in \Delta(\mathcal{G}) \text{ such that } \mu_0 \mathbb{E}_{g\sim P}(g(x)-1)^2 + \sum_{j=1}^{i} \mu_j \mathbb{E}_{g\sim P}\widehat{R}_j(g; y) \le \mu_0 c + \sum_{j=1}^{i} \mu_j \tilde{\Delta}_j.$$

This program can be solved by a single call to the regression oracle, since all terms on the left-hand-side involve square losses while the right hand side is a constant. Thus we can efficiently implement the MW algorithm using the regression oracle. Finally, recalling that the above description is for a fixed value of objective c, and recalling that the maximum can be approximated by a binary search over c leads to an oracle-based algorithm for computing the maximum cost. For this procedure, we have the following computational guarantee.

Theorem 3 *Algorithm 2 returns an estimate $\widehat{c_+}(x; y)$ such that $c_+(x; y) \le \widehat{c_+}(x; y) \le c_+(x; y) + \text{TOL}$ and runs in polynomial time with $\mathcal{O}(\max\{1, i^2/\nu_n^2\}\log(i)\log(1/\text{TOL})/\text{TOL}^4)$ calls to the regression oracle.*

The minimum cost can be estimated in exactly the same way, replacing the objective $(g(x)-1)^2$ with $(g(x)-0)^2$ in Program (7). In COAL, we set $\text{TOL} = 1/\sqrt{i}$ at iteration i and have $\nu_n = \tilde{\mathcal{O}}(d)$. As a consequence, we can bound the total oracle complexity after processing n examples.

Corollary 4 *After processing n examples,* COAL *makes $\tilde{\mathcal{O}}(K(d^3 + n^5/d^2))$ calls to the square loss oracle.*

Thus COAL can be implemented in polynomial time for any set $\mathcal{G}$ that admits efficient square loss optimization. Compared to Krishnamurthy et al. (2017) which required $\mathcal{O}(n^2)$ oracle calls, the guarantee here is, at face value, worse, since the algorithm is slower. However, the algorithm enforces a much stronger constraint on the version space which leads to a much better statistical analysis, as we will discuss next. Nevertheless, these algorithms that use batch square loss optimization in an iterative or sequential fashion are too computational demanding to scale to larger problems. Our implementation alleviates this with an alternative heuristic approximation based on a sensitivity analysis of the oracle, which we detail in Section 7.

5. Generalization Analysis

In this section, we derive generalization guarantees for COAL. We study three settings: one with minimal assumptions and two low-noise settings.

Our first low-noise assumption is related to the Massart noise condition (Massart and Nédélec, 2006), which in binary classification posits that the Bayes optimal predictor is bounded away from 1/2 for all x. Our condition generalizes this to CSMC and posits that the expected cost of the best label is separated from the expected cost of all other labels.

Assumption 3 *A distribution $\mathcal{D}$ supported over (x, c) pairs satisfies the* Massart noise condition *with parameter $\tau > 0$, if for all x (with $\mathcal{D}(x) > 0$),*

$$f^\star(x; y^\star(x)) \leq \min_{y \neq y^\star(x)} f^\star(x; y) - \tau,$$

where $y^\star(x) \triangleq \mathit{argmin}_y f^\star(x; y)$ is the true best label for x.

The Massart noise condition describes favorable prediction problems that lead to sharper generalization and label complexity bounds for COAL. We also study a milder noise assumption, inspired by the Tsybakov condition (Mammen and Tsybakov, 1999; Tsybakov, 2004), again generalized to CSMC. See also Agarwal (2013).

Assumption 4 *A distribution $\mathcal{D}$ supported over (x, c) pairs satisfies the* Tsbyakov noise condition *with parameters (τ_0, α, β) if for all $0 \leq \tau \leq \tau_0$,*

$$\mathbb{P}_{x \sim \mathcal{D}}\left[\min_{y \neq y^\star(x)} f^\star(x; y) - f^\star(x; y^\star(x)) \leq \tau\right] \leq \beta \tau^\alpha,$$

where $y^\star(x) \triangleq \mathit{argmin}_y f^\star(x; y)$.

Observe that the Massart noise condition in Assumption 3 is a limiting case of the Tsybakov condition, with $\tau = \tau_0$ and $\alpha \to \infty$. The Tsybakov condition states that it is polynomially unlikely for the cost of the best label to be close to the cost of the other labels. This condition has been used in previous work on cost-sensitive active learning (Agarwal, 2013) and is also related to the condition studied by Castro and Nowak (2008) with the translation that $\alpha = \frac{1}{\kappa - 1}$, where $\kappa \in [0, 1]$ is their noise level.

Our generalization bound is stated in terms of the noise level in the problem so that they can be readily adapted to the favorable assumptions. We define the noise level using the following quantity, given any $\zeta > 0$.

$$P_\zeta \triangleq \mathbb{P}_{x \sim \mathcal{D}}\left[\min_{y \neq y^\star(x)} f^\star(x; y) - f^\star(x; y^\star(x)) \leq \zeta\right]. \tag{10}$$

P_ζ describes the probability that the expected cost of the best label is close to the expected cost of the second best label. When P_ζ is small for large ζ the labels are well-separated so learning is easier. For instance, under a Massart condition $P_\zeta = 0$ for all $\zeta \leq \tau$.

We now state our generalization guarantee.

Theorem 5 *For any $\delta < 1/e$, for all $i \in [n]$, with probability at least $1 - \delta$, we have*

$$\mathbb{E}_{x,c}[c(h_{f_{i+1}}(x)) - c(h_{f^\star}(x))] \leq \min_{\zeta > 0}\left\{\zeta P_\zeta + \frac{32 K \nu_n}{\zeta i}\right\},$$

where ν_n, f_i are defined in Algorithm 1, and h_{f_i} is defined in (1).

In the worst case, we bound P_ζ by 1 and optimize for ζ to obtain an $\tilde{\mathcal{O}}(\sqrt{Kd\log(1/\delta)/i})$ bound after i samples, where recall that d is the pseudo-dimension of $\mathcal{G}$. This agrees with the standard generalization bound of $O(\sqrt{\mathrm{Pdim}(\mathcal{F})\log(1/\delta)/i})$ for VC-type classes because $\mathcal{F} = \mathcal{G}^K$ has $O(Kd)$ statistical complexity. However, since the bound captures the difficulty of the CSMC problem as measured by P_ζ, we can obtain sharper results under Assumptions 3 and 4 by appropriately setting ζ.

Corollary 6 *Under Assumption 3, for any $\delta < 1/e$, with probability at least $1 - \delta$, for all $i \in [n]$, we have*

$$\mathbb{E}_{x,c}[c(h_{f_{i+1}}(x)) - c(h_{f^\star}(x))] \leq \frac{32 K \nu_n}{i\tau}.$$

Corollary 7 *Under Assumption 4, for any $\delta < 1/e$, with probability at least $1 - \delta$, for all $\frac{32K\nu_n}{\beta\tau_0^{\alpha+2}} \leq i \leq n$, we have*

$$\mathbb{E}_{x,c}[c(h_{f_{i+1}}(x)) - c(h_{f^\star}(x))] \leq 2\beta^{\frac{1}{\alpha+2}}\left(\frac{32K\nu_n}{i}\right)^{\frac{\alpha+1}{\alpha+2}}.$$

Thus, Massart and Tsybakov-type conditions lead to a faster convergence rate of $\tilde{\mathcal{O}}(1/n)$ and $\tilde{\mathcal{O}}(n^{-\frac{\alpha+1}{\alpha+2}})$. This agrees with the literature on active learning for classification (Massart and Nédélec, 2006) and can be viewed as a generalization to CSMC. Both generalization bounds match the optimal rates for binary classification under the analogous low-noise assumptions (Massart and Nédélec, 2006; Tsybakov, 2004). We emphasize that COAL obtains these bounds as is, without changing any parameters, and hence COAL is *adaptive* to favorable noise conditions.

6. Label Complexity Analysis

Without distributional assumptions, the label complexity of COAL can be $\mathcal{O}(n)$, just as in the binary classification case, since there may always be confusing labels that force querying. In line with prior work, we introduce two *disagreement coefficients* that characterize favorable distributional properties. We first define a set of good classifiers, the *cost-sensitive regret* ball:

$$\mathcal{F}_{\mathrm{csr}}(r) \triangleq \left\{f \in \mathcal{F} \;\middle|\; \mathbb{E}\left[c(h_f(x)) - c(h_{f^\star}(x))\right] \leq r\right\}.$$

We also recall our earlier notation $\gamma(x, y, F)$ (see (3) and the subsequent discussion) for a subset $F \subseteq \mathcal{F}$ which indicates the range of expected costs for (x, y) as predicted by the regressors corresponding to the classifiers in F. We now define the disagreement coefficients.

Definition 8 (Disagreement coefficients) *Define*

$$\mathit{DIS}(r, y) \triangleq \left\{x \mid \exists f, f' \in \mathcal{F}_{csr}(r), h_f(x) = y \neq h_{f'}(x)\right\}.$$

Then the disagreement coefficients are defined as:

$$\theta_1 \triangleq \sup_{\psi, r>0} \frac{\psi}{r} \mathbb{P}\left(\exists y \mid \gamma(x, y, \mathcal{F}_{csr}(r)) > \psi \wedge x \in DIS(r, y)\right),$$

$$\theta_2 \triangleq \sup_{\psi, r>0} \frac{\psi}{r} \sum_y \mathbb{P}\left(\gamma(x, y, \mathcal{F}_{csr}(r)) > \psi \wedge x \in DIS(r, y)\right).$$

Intuitively, the conditions in both coefficients correspond to the checks on the *domination* and *cost range* of a label in Lines 12 and 14 of Algorithm 1. Specifically, when $x \in \mathrm{DIS}(r, y)$, there is confusion about whether y is the optimal label or not, and hence y is not dominated. The condition on $\gamma(x, y, \mathcal{F}_{\mathrm{csr}}(r))$ additionally captures the fact that a small cost range provides little information, even when y is non-dominated. Collectively, the coefficients capture the probability of an example x where the good classifiers disagree on x in both predicted costs and labels. Importantly, the notion of good classifiers is via the algorithm-independent set $\mathcal{F}_{\mathrm{csr}}(r)$, and is only a property of $\mathcal{F}$ and the data distribution.

The definitions are a natural adaptation from binary classification (Hanneke, 2014), where a similar disagreement region to $\mathrm{DIS}(r, y)$ is used. Our definition asks for confusion about the optimality of a specific label y, which provides more detailed information about the cost-structure than simply asking for any confusion among the good classifiers. The $1/r$ scaling is in agreement with previous related definitions (Hanneke, 2014), and we also scale by the cost range parameter ψ, so that the favorable settings for active learning can be concisely expressed as having θ_1, θ_2 bounded, as opposed to a complex function of ψ.

The next three results bound the labeling effort (4), in the high noise and low noise cases respectively. The low noise assumptions enable significantly sharper bounds. Before stating the bounds, we recall that L_1 corresponds to the number of examples where at least one cost is queried, while L_2 is the total number of costs queried across all examples.

Theorem 9 *With probability at least $1 - \delta$, the label complexity of the algorithm over n examples is at most*

$$L_1 = \mathcal{O}\left(n\theta_1 \sqrt{K\nu_n} + \log(1/\delta)\right),$$
$$L_2 = \mathcal{O}\left(n\theta_2 \sqrt{K\nu_n} + K\log(1/\delta)\right).$$

Theorem 10 *Assume the Massart noise condition holds. With probability at least $1 - \delta$ the label complexity of the algorithm over n examples is at most*

$$L_1 = \mathcal{O}\left(\frac{K\log(n)\nu_n}{\tau^2}\theta_1 + \log(1/\delta)\right), \qquad L_2 = \mathcal{O}\left(KL_1\right)$$

Theorem 11 *Assume the Tsybakov noise condition holds. With probability at least $1 - \delta$ the label complexity of the algorithm over n examples is at most*

$$L_1 = \mathcal{O}\left(\theta_1^{\frac{\alpha}{\alpha+1}}(K\nu_n)^{\frac{\alpha}{\alpha+2}} n^{\frac{2}{\alpha+2}} + \log(1/\delta)\right), \qquad L_2 = \mathcal{O}\left(KL_1\right)$$

In the high-noise case, the bounds scales with $n\theta$ for the respective coefficients. In comparison, for binary classification the leading term is $\tilde{\mathcal{O}}\left(n\theta \mathrm{error}(h_{f^\star})\right)$ which involves a

different disagreement coefficient and which scales with the error of the optimal classifier $h_{f^\star}$ (Hanneke, 2014; Huang et al., 2015). Qualitatively the bounds have similar worst-case behavior, demonstrating minimal improvement over passive learning, but by scaling with $\mathrm{error}(h_{f^\star})$ the binary classification bound reflects improvements on benign instances. For the special case of multiclass classification, we are able to recover the dependence on $\mathrm{error}(h_{f^\star})$ and the standard disagreement coefficient with a simple modification to our proof, which we discuss in detail in the next subsection.

On the other hand, in both low noise cases the label complexity scales sublinearly with n. With bounded disagreement coefficients, this improves over the standard passive learning analysis where all labels are queried on n examples to achieve the generalization guarantees in Theorem 5, Corollary 6, and Corollary 7 respectively. In particular, under the Massart condition, both L_1 and L_2 bounds scale with $\theta \log(n)$ for the respective disagreement coefficients, which is an *exponential improvement* over the passive learning analysis. Under the milder Tsybakov condition, the bounds scale with $\theta^{\frac{\alpha}{\alpha+1}} n^{\frac{2}{\alpha+2}}$, which improves polynomially over passive learning. These label complexity bounds agree with analogous results from binary classification (Castro and Nowak, 2008; Hanneke, 2014; Hanneke and Yang, 2015) in their dependence on n.

Note that $\theta_2 \leq K\theta_1$ always and it can be much smaller, as demonstrated through an example in the next section. In such cases, only a few labels are ever queried and the L_2 bound in the high noise case reflects this additional savings over passive learning. Unfortunately, in low noise conditions, we do not benefit when $\theta_2 \ll K\theta_1$. This can be resolved by letting ψ_i in the algorithm depend on the noise level τ, but we prefer to use the more robust choice $\psi_i = 1/\sqrt{i}$ which still allows COAL to partially adapt to low noise and achieve low label complexity.

The main improvement over Krishnamurthy et al. (2017) is demonstrated in the label complexity bounds under low noise assumptions. For example, under Massart noise, our bound has the optimal $\log(n)/\tau^2$ rate, while the bound in Krishnamurthy et al. (2017) is exponentially worse, scaling with n^β/τ^2 for $\beta \in (0, 1)$. This improvement comes from explicitly enforcing monotonicity of the version space, so that once a regressor is eliminated it can never force COAL to query again. Algorithmically, computing the maximum and minimum costs with the monotonicity constraint is much more challenging and requires the new subroutine using MW.

6.1. Recovering Hanneke's Disagreement Coefficient

In this subsection we show that in many cases we can obtain guarantees in terms of Hanneke's disagreement coefficient (Hanneke, 2014), which has been used extensively in active learning for binary classification. We also show that, for multiclass classification, the label complexity scales with the error of the optimal classifier $h^\star$, a refinement on Theorem 9. The guarantees require no modifications to the algorithm and enable a precise comparison with prior results. Unfortunately, they do not apply to the general CSMC setting, so they have not been incorporated into our main theorems.

We start with defining Hanneke's disagreement coefficient (Hanneke, 2014). Define the *disagreement ball* $\widetilde{\mathcal{F}}(r) \triangleq \{f \in \mathcal{F} : \mathbb{P}[h_f(x) \neq h_{f^\star}(x)] \leq r\}$ and the disagreement region

$\widetilde{\mathrm{DIS}}(r) \triangleq \{x \mid \exists f, f' \in \widetilde{\mathcal{F}}(r), h_f(x) \neq h_{f'}(x)\}$. The coefficient is defined as

$$\theta_0 \triangleq \sup_{r>0} \frac{1}{r} \mathbb{P}\left[x \in \widetilde{\mathrm{DIS}}(r)\right]. \tag{11}$$

This coefficient is known to be $O(1)$ in many cases, for example when the hypothesis class consists of linear separators and the marginal distribution is uniform over the unit sphere (Hanneke, 2014, Chapter 7). In comparison with Definition 8, the two differences are that θ_1, θ_2 include the cost-range condition and involve the cost-sensitive regret ball $\mathcal{F}_{\mathrm{csr}}(r)$ rather than $\widetilde{\mathcal{F}}(r)$. As $\widetilde{\mathcal{F}}(r) \subset \mathcal{F}_{\mathrm{csr}}(r)$, we expect that θ_1 and θ_2 are typically larger than θ_0, so bounds in terms of θ_0 are more desirable. We now show that such guarantees are possible in many cases.

The low noise case. For general CSMC, low noise conditions admit the following:

Proposition 12 *Under Massart noise, with probability at least $1-\delta$ the label complexity of the algorithm over n examples is at most $L_1 = \mathcal{O}\left(\frac{\log(n)\nu_n}{\tau^2}\theta_0 + \log(1/\delta)\right)$. Under Tsybakov noise, the label complexity is at most $L_1 = \mathcal{O}\left(\theta_0 n^{\frac{2}{\alpha+2}} (\log(n)\nu_n)^{\frac{\alpha}{\alpha+2}} + \log(1/\delta)\right)$. In both cases we have $L_2 = O(KL_1)$.*

That is, for any low noise CSMC problem, COAL obtains a label complexity bound in terms of Hanneke's disagreement coefficient θ_0 directly. Note that this adaptivity requires no change to the algorithm. Proposition 12 enables a precise comparison with disagreement-based active learning for binary classification. In particular, this bound matches the guarantee for CAL (Hanneke, 2014, Theorem 5.4) with the caveat that our measure of statistical complexity is the pseudodimension of the $\mathcal{F}$ instead of the VC-dimension of the hypothesis class. As a consequence, under low noise assumptions, COAL has favorable label complexity in all examples where θ_0 is small.

The high noise case. Outside of the low noise setting, we can introduce θ_0 into our bounds, but only for multiclass classification, where we always have $c \triangleq \mathbf{1} - e_y$ for some $y \in [K]$. Note that $f(x; y)$ is now interpreted as a prediction for $1 - P(y|x)$, so that the least cost prediction $y^\star(x)$ corresponds to the most likely label. We also obtain a further refinement by introducing $\mathrm{error}(h_{f^\star}) \triangleq \mathbb{E}_{(x,c)}[c(h_{f^\star}(x))]$.

Proposition 13 *For multiclass classification, with probability at least $1-\delta$, the label complexity of the algorithm over n examples is at most*

$$L_1 = 4\theta_0 n \cdot \mathit{error}(h_{f^\star}) + \mathcal{O}\left(\theta_0\left(\sqrt{Kn\nu_n \cdot \mathit{error}(h_{f^\star})} + K\kappa\nu_n \log(n)\right) + \log(1/\delta)\right).$$

This result exploits two properties of the multiclass cost structure. First we can relate $\mathcal{F}_{\mathrm{csr}}(r)$ to the disagreement ball $\widetilde{\mathcal{F}}(r)$, which lets us introduce Hanneke's disagreement coefficient θ_0. Second, we can bound P_ζ in Theorem 5 in terms of $\mathrm{error}(h_{f^\star})$. Together the bound is comparable to prior results for active learning in binary classification (Hsu, 2010; Hanneke and Yang, 2012; Hanneke, 2014), with a slight generalization to the multiclass setting. Unfortunately, both of these refinements do not apply for general CSMC.

Summary. In important special cases, COAL achieves label complexity bounds directly comparable with results for active learning in binary classification, scaling with θ_0 and $\mathrm{error}(h_{f^\star})$. In such cases, whenever θ_0 is bounded — for which many examples are known — COAL has favorable label complexity. However, in general CSMC without low-noise assumptions, we are not able to obtain a bound in terms of these quantities, and we believe a bound involving θ_0 does not hold for COAL. We leave understanding natural settings where θ_1 and θ_2 are small, or obtaining sharper guarantees as intriguing future directions.

6.2. Three Examples

We now describe three examples to give more intuition for COAL and our label complexity bounds. Even in the low noise case, our label complexity analysis does not demonstrate all of the potential benefits of our query rule. In this section we give three examples to further demonstrate these advantages.

Our first example shows the benefits of using the domination criterion in querying, in addition to the cost range condition. Consider a problem under Assumption 3, where the optimal cost is predicted perfectly, the second best cost is τ worse and all the other costs are substantially worse, but with variability in the predictions. Since all classifiers predict the correct label, we get $\theta_1 = \theta_2 = 0$, so our label complexity bound is $\mathcal{O}(1)$. Intuitively, since every regressor is certain of the optimal label and its cost, we actually make zero queries. On the other hand, all of the suboptimal labels have large cost ranges, so querying based solely on a cost range criteria, as would happen with an active regression algorithm (Castro et al., 2005), leads to a large label complexity.

A related example demonstrates the improvement in our query rule over more naïve approaches where we query either no label or all labels, which is the natural generalization of query rules from multiclass classification (Agarwal, 2013). In the above example, if the best and second best labels are confused occasionally θ_1 may be large, but we expect $\theta_2 \ll K\theta_1$ since no other label can be confused with the best. Thus, the L_2 bound in Theorem 9 is a factor of K smaller than with a naïve query rule since COAL only queries the best and second best labels. Unfortunately, without setting ψ_i as a function of the noise parameters, the bounds in the low noise cases do not reflect this behavior.

The third example shows that both θ_0 and θ_1 yield pessimistic bounds on the label complexity of COAL in some cases. The example is more involved, so we describe it in detail. We focus on statistical issues, using a finite regressor class $\mathcal{F}$. Note that our results on generalization and label complexity hold in this setting, replacing $d\log(n)$ with $\log|\mathcal{F}|$, and the algorithm can be implemented by enumerating $\mathcal{F}$. Throughout this example, we use $\tilde{O}(\cdot)$ to further suppress logarithmic dependence on n.

Let $\mathcal{X} \triangleq \{x_1, \ldots, x_M\}$, $\mathcal{Y} \triangleq \{0, 1\}$, and consider functions $\mathcal{F} \triangleq \{f^\star, f_1, \ldots, f_M\}$. We have $f^\star(x) \triangleq (1/4, 1/2), \forall x \in \mathcal{X}$ and $f_i(x_i) \triangleq (1/4, 0)$ and $f_i(x_j) \triangleq (1/4, 1)$ for $i \neq j$. The marginal distribution is uniform and the true expected costs are given by $f^\star$ so that the problem satisfies the Massart noise condition with $\tau = 1/4$. The key to the construction is that f_is have high square loss on labels that they do not predict.

Observe that as $\mathbb{P}[h_{f_i}(x) \neq h_{f^\star}(x)] = 1/M$ and $h_{f_i}(x_i) \neq h_{f^\star}(x_i)$ for all i, the probability of disagreement is 1 until all f_i are eliminated. As such, we have $\theta_0 = M$. Similarly, we have $\mathbb{E}[c(h_{f_i}(x)) - c(h_{f^\star}(x))] = \frac{1}{4M}$ and $\gamma(x, 1, \mathcal{F}_{\mathrm{csr}}(\frac{1}{4M})) = 1$, so $\theta_1 = 4M$. Therefore, the

bounds in Theorem 10 and Proposition 12 are both $\tilde{O}(M \log |\mathcal{F}|) = \tilde{O}(|\mathcal{F}|)$. On the other hand, since $(f_i(x_j, 1) - f^\star(x_j, 1))^2 = 1/4$ for all $i, j \in [M]$, COAL eliminates every f_i once it has made a total of $\tilde{O}(\log |\mathcal{F}|)$ queries to label $y = 1$. Thus the label complexity is actually just $\tilde{O}(\log |\mathcal{F}|)$, which is exponentially better than the disagreement-based analyses. Thus, COAL can perform much better than suggested by the disagreement-based analyses, and an interesting future direction is to obtain refined guarantees for cost-sensitive active learning.

7. Experiments

We now turn to an empirical evaluation of COAL. For further computational efficiency, we implemented an approximate version of COAL using: 1) a relaxed version space $\mathcal{G}_i(y) \leftarrow \{g \in \mathcal{G} \mid \widehat{R}_i(g; y) \le \widehat{R}_i(g_{i,y}; y) + \Delta_i\}$, which does not enforce monotonicity, and 2) *online optimization*, based on online linear least-squares regression. The algorithm processes the data in one pass, and the idea is to (1) replace $g_{i,y}$, the ERM, with an approximation $g^o_{i,y}$ obtained by online updates, and (2) compute the minimum and maximum costs via a sensitivity analysis of the online update. We describe this algorithm in detail in Subsection 7.1. Then, we present our experimental results, first for simulated active learning (Subsection 7.2) and then for learning to search for joint prediction (Subsection 7.3).

7.1. Finding Cost Ranges with Online Approximation

Consider the maximum and minimum costs for a fixed example x and label y at round i, all of which may be suppressed. We ignore all the constraints on the empirical square losses for the past rounds. First, define $\widehat{R}(g, w, c; y) \triangleq \widehat{R}(g; y) + w(g(x) - c)^2$, which is the risk functional augmented with a fake example with weight w and cost c. Also define

$$\underline{g}_w \triangleq \arg\min_{g \in \mathcal{G}} \widehat{R}(g, w, 0; y), \qquad \overline{g}_w \triangleq \arg\min_{g \in \mathcal{G}} \widehat{R}(g, w, 1; y),$$

and recall that $g_{i,y}$ is the ERM given in Algorithm 1. The functional $\widehat{R}(g, w, c; y)$ has a monotonicity property that we exploit here, proved in Appendix C.

Lemma 14 *For any c and for $w' \ge w \ge 0$, define $g = \operatorname{argmin}_g \widehat{R}(g, w, c)$ and $g' = \operatorname{argmin}_g \widehat{R}(g, w', c)$. Then*

$$\widehat{R}(g') \ge \widehat{R}(g) \text{ and } (g'(x) - c)^2 \le (g(x) - c)^2.$$

As a result, an alternative to MinCost and MaxCost is to find

$$\underline{w} \triangleq \max\{w \mid \widehat{R}(\underline{g}_w) - \widehat{R}(g_{i,y}) \le \Delta_i\}, \tag{12}$$

$$\overline{w} \triangleq \max\{w \mid \widehat{R}(\overline{g}_w) - \widehat{R}(g_{i,y}) \le \Delta_i\}, \tag{13}$$

and return $\underline{g}_{\underline{w}}(x)$ and $\overline{g}_{\overline{w}}(x)$ as the minimum and maximum costs. We use two steps of approximation here. Using the definition of $\overline{g}_w$ and $\underline{g}_w$ as the minimizers of $\widehat{R}(g, w, 1; y)$ and $\widehat{R}(g, w, 0; y)$ respectively, we have

$$\widehat{R}(\underline{g}_w) - \widehat{R}(g_{i,y}) \le w \cdot g_{i,y}(x)^2 - w \cdot \underline{g}_w(x)^2,$$
$$\widehat{R}(\overline{g}_w) - \widehat{R}(g_{i,y}) \le w \cdot (g_{i,y}(x) - 1)^2 - w \cdot (\overline{g}_w(x) - 1)^2.$$

We use this upper bound in place of $\widehat{R}(g_w) - \widehat{R}(g_{i,y})$ in (12) and (13). Second, we replace $g_{i,y}$, $\underline{g}_w$, and $\overline{g}_w$ with approximations obtained by online updates. More specifically, we replace $g_{i,y}$ with $g^o_{i,y}$, the current regressor produced by all online linear least squares updates so far, and approximate the others by

$$\underline{g}_w(x) \approx g^o_{i,y}(x) - w \cdot s(x, 0, g^o_{i,y}), \qquad \overline{g}_w(x) \approx g^o_{i,y}(x) + w \cdot s(x, 1, g^o_{i,y}),$$

where $s(x, y, g^o_{i,y}) \ge 0$ is a *sensitivity* value that approximates the change in prediction on x resulting from an online update to $g^o_{i,y}$ with features x and label y. The computation of this sensitivity value is governed by the actual online update where we compute the derivative of the change in the prediction as a function of the importance weight w for a hypothetical example with cost 0 or cost 1 and the same features. This is possible for essentially all online update rules on importance weighted examples, and it corresponds to taking the limit as $w \to 0$ of the change in prediction due to an update, divided by w. Since we are using linear representations, this requires only $\mathcal{O}(s)$ time per example, where s is the average number of non-zero features. With these two steps, we obtain approximate minimum and maximum costs using

$$g^o_{i,y}(x) - \underline{w}^o \cdot s(x, 0, g^o_{i,y}), \qquad g^o_{i,y}(x) + \overline{w}^o \cdot s(x, 1, g^o_{i,y}),$$

where

$$\underline{w}^o \triangleq \max\{w \mid w\left(g^o_{i,y,}(x)^2 - (g^o_{i,y}(x) - w \cdot s(x, 0, g^o_{i,y}))^2\right) \le \Delta_i\}$$
$$\overline{w}^o \triangleq \max\{w \mid w\left((g^o_{i,y,}(x) - 1)^2 - (g^o_{i,y}(x) + w \cdot s(x, 1, g^o_{i,y}) - 1)^2\right) \le \Delta_i\}.$$

The online update guarantees that $g^o_{i,y}(x) \in [0, 1]$. Since the minimum cost is lower bounded by 0, we have $\underline{w}^o \in \left(0, \frac{g^o_{i,y}(x)}{s(x,0,g^o_{i,y})}\right]$. Finally, because the objective $w(g^o_{i,y}(x))^2 - w(g^o_{i,y}(x) - w \cdot s(x, 0, g^o_{i,y}))^2$ is increasing in w within this range (which can be seen by inspecting the derivative), we can find $\underline{w}^o$ with binary search. Using the same techniques, we also obtain an approximate maximum cost.

7.2. Simulated Active Learning

We performed simulated active learning experiments with three datasets. ImageNet 20 and 40 are sub-trees of the ImageNet hierarchy covering the 20 and 40 most frequent classes, where each example has a single zero-cost label, and the cost for an incorrect label is the tree-distance to the correct one. The feature vectors are the top layer of the Inception neural network (Szegedy et al., 2015). The third, RCV1-v2 (Lewis et al., 2004), is a multilabel text-categorization dataset, which has 103 labels, organized as a tree with a similar tree-distance cost structure as the ImageNet data. Some dataset statistics are in Table 1.

We compare our online version of COAL to passive online learning. We use the cost-sensitive one-against-all (csoaa) implementation in Vowpal Wabbit[5], which performs online linear regression for each label separately. There are two tuning parameters in our implementation. First, instead of Δ_i, we set the radius of the version space to $\Delta'_i = \frac{\kappa \nu_{i-1}}{i-1}$ (i.e. the $\log(n)$ term in the definition of ν_n is replaced with $\log(i)$) and instead tune the constant

5. `http://hunch.net/~vw`

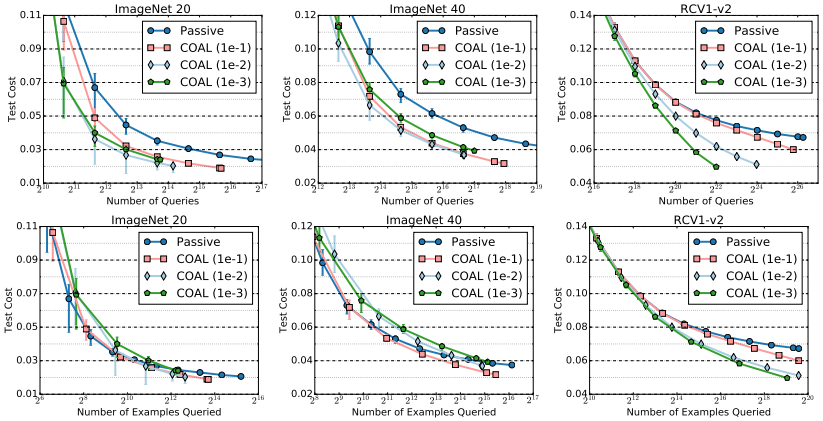


Figure 2: Experiments with COAL. Top row shows test cost vs. number of queries for simulated active learning experiments. Bottom row shows test cost vs. number of examples with even a single label query for simulated active learning experiments.

	K	n	feat	density		K	n	feat	len
ImageNet 20	20	38k	6k	21.1%	POS	45	38k	40k	24
ImageNet 40	40	71k	6k	21.0%	NER	9	15k	15k	14
RCV1-v2	103	781k	47k	0.16%	Wiki	9	132k	89k	25

Table 1: Dataset statistics. *len* is the average sequence length and density is the percentage of non-zero features.

κ. This alternate "mellowness" parameter controls how aggressive the query strategy is. The second parameter is the learning rate used by online linear regression[6].

For all experiments, we show the results obtained by the best learning rate for each mellowness on each dataset, which is tuned as follows. We randomly permute the training data 100 times and make one pass through the training set with each parameter setting. For each dataset let $\text{perf}(mel, l, q, t)$ denote the test performance of the algorithm using mellowness mel and learning rate l on the t^{th} permutation of the training data under a query budget of $2^{(q-1)} \cdot 10 \cdot K, q \geq 1$. Let $\text{query}(mel, l, q, t)$ denote the number of queries actually made. Note that $\text{query}(mel, l, q, t) < 2^{(q-1)} \cdot 10 \cdot K$ if the algorithm runs out of the

6. We use the default online learning algorithm in Vowpal Wabbit, which is a scale-free (Ross et al., 2013) importance weight invariant (Karampatziakis and Langford, 2011) form of AdaGrad (Duchi et al., 2010).

	ImageNet 20	ImageNet 40	RCV1-v2	POS	NER	NER-wiki
passive	1	1	0.5	1.0	0.5	0.5
active (10^{-1})	0.05	0.1	0.5	1.0	0.1	0.5
active (10^{-2})	0.05	0.5	0.5	1.0	0.5	0.5
active (10^{-3})	1	10	0.5	10	0.5	0.5

Table 2: Best learning rates for each learning algorithm and each dataset.

training data before reaching the q^{th} query budget[7]. To evaluate the trade-off between test performance and number of queries, we define the following performance measure:

$$\text{AUC}(mel, l, t) = \frac{1}{2} \sum_{q=1}^{q_{\max}} \Big(\text{perf}(mel, l, q+1, t) + \text{perf}(mel, l, q, t) \Big) \cdot \log_2 \frac{\text{query}(mel, l, q+1, t)}{\text{query}(mel, l, q, t)}, \tag{14}$$

where $q_{\max}$ is the minimum q such that $2^{(q-1)} \cdot 10$ is larger than the size of the training data. This performance measure is the area under the curve of test performance against number of queries in $\log_2$ scale. A large value means the test performance quickly improves with the number of queries. The best learning rate for mellowness mel is then chosen as

$$l^\star(mel) \triangleq \arg\max_l \text{median}_{1 \leq t \leq 100} \quad \text{AUC}(mel, l, t).$$

The best learning rates for different datasets and mellowness settings are in Table 2.

In the top row of Figure 2, we plot, for each dataset and mellowness, the number of queries against the median test cost along with bars extending from the 15^{th} to 85^{th} quantile. Overall, COAL achieves a better trade-off between performance and queries. With proper mellowness parameter, active learning achieves similar test cost as passive learning with a factor of 8 to 32 fewer queries. On ImageNet 40 and RCV1-v2 (reproduced in Figure 1), active learning achieves *better* test cost with a factor of 16 fewer queries. On RCV1-v2, COAL queries like passive up to around $256k$ queries, since the data is very sparse, and linear regression has the property that the cost range is maximal when an example has a new unseen feature. Once COAL sees all features a few times, it queries much more efficiently than passive. These plots correspond to the label complexity L_2.

In the bottom row, we plot the test error as a function of the number of examples for which at least one query was requested, for each dataset and mellowness, which experimentally corresponds to the L_1 label complexity. In comparison to the top row, the improvements offered by active learning are slightly less dramatic here. This suggests that our algorithm queries just a few labels for each example, but does end up issuing at least one query on most of the examples. Nevertheless, one can still achieve test cost competitive with passive learning using a factor of 2-16 less labeling effort, as measured by L_1.

We also compare COAL with two active learning baselines. Both algorithms differ from COAL only in their query rule. AllOrNone queries either all labels or no labels using both domination and cost-range conditions and is an adaptation of existing multiclass active learners (Agarwal, 2013). NoDom just uses the cost-range condition, inspired by active

7. In fact, we check the test performance only in between examples, so $\text{query}(mel, l, q, t)$ may be larger than $2^{(q-1)} \cdot 10 \cdot K$ by an additive factor of K, which is negligibly small.

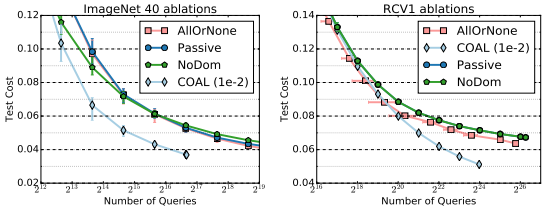


Figure 3: Test cost versus number of queries for COAL, in comparison with active and passive baselines on the ImageNet40 and RCV1-v2 dataset. On RCV1-v2, passive learning and NoDom are nearly identical.

regression (Castro et al., 2005). The results for ImageNet 40 and RCV1-v2 are displayed in Figure 3, where we use the AUC strategy to choose the learning rate. We choose the mellowness by visual inspection for the baselines and use 0.01 for COAL[8]. On ImageNet 40, the ablations provide minimal improvement over passive learning, while on RCV1-v2, AllOrNone does provide marginal improvement. However, on both datasets, COAL substantially outperforms both baselines and passive learning.

While not always the best, we recommend a mellowness setting of 0.01 as it achieves reasonable performance on all three datasets. This is also confirmed by the learning-to-search experiments, which we discuss next.

7.3. Learning to Search

We also experiment with COAL as the base leaner in *learning-to-search* (Daumé III et al., 2009; Chang et al., 2015), which reduces joint prediction problems to CSMC. A joint prediction example defines a search space, where a sequence of decisions are made to generate the structured label. We focus here on sequence labeling tasks, where the input is a sentence and the output is a sequence of labels, specifically, parts of speech or named entities.

Learning-to-search solves such problems by generating the output one label at a time, conditioning on all past decisions. Since mistakes may lead to compounding errors, it is natural to represent the decision space as a CSMC problem, where the classes are the "actions" available (e.g., possible labels for a word) and the costs reflect the long term loss of each choice. Intuitively, we should be able to avoid expensive computation of long term loss on decisions like "is 'the' a DETERMINER?" once we are quite sure of the answer. Similar ideas motivate adaptive sampling for structured prediction (Shi et al., 2015).

We specifically use Aggravate (Ross and Bagnell, 2014; Chang et al., 2015; Sun et al., 2017), which runs a learned policy to produce a backbone sequence of labels. For each position in the input, it then considers all possible deviation actions and executes an oracle for the rest of the sequence. The loss on this complete output is used as the cost for the

8. We use 0.01 for AllOrNone and 10^{-3} for NoDom.

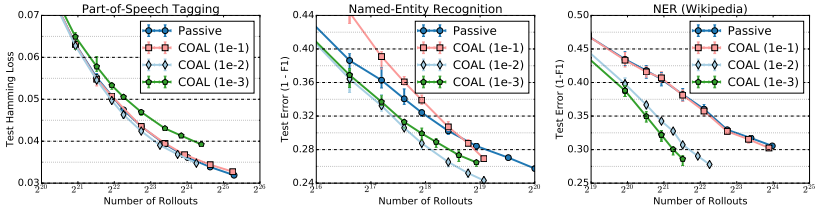


Figure 4: Learning to search experiments with COAL. Accuracy vs. number of rollouts for active and passive learning as the CSMC algorithm in learning-to-search.

deviating action. Run in this way, Aggravate requires $len \times K$ roll-outs when the input sentence has len words and each word can take one of K possible labels.

Since each roll-out takes $\mathcal{O}(len)$ time, this can be computationally prohibitive, so we use active learning to reduce the number of roll-outs. We use COAL and a passive learning baseline inside Aggravate on three joint prediction datasets (statistics are in Table 1). As above, we use several mellowness values and the same AUC criteria to select the best learning rate (see Table 2). The results are in Figure 4, and again our recommended mellowness is 0.01.

Overall, active learning reduces the number of roll-outs required, but the improvements vary on the three datasets. On the Wikipedia data, COAL performs a factor of 4 fewer rollouts to achieve similar performance to passive learning and achieves substantially better test performance. A similar, but less dramatic, behavior arises on the NER task. On the other hand, COAL offers minimal improvement over passive learning on the POS-tagging task. This agrees with our theory and prior empirical results (Hsu, 2010), which show that active learning may not always improve upon passive learning.

8. Proofs

In this section we provide proofs for the main results, the oracle-complexity guarantee and the generalization and label complexity bounds. We start with some supporting results, including a new uniform freedman-type inequality that may be of independent interest. The proof of this inequality, and the proofs for several other supporting lemmata are deferred to the appendices.

8.1. Supporting Results

A deviation bound. For both the computational and statistical analysis of COAL, we require concentration of the square loss functional $\widehat{R}_j(\cdot; y)$, uniformly over the class $\mathcal{G}$. To describe the result, we introduce the central random variable in the analysis:

$$M_j(g; y) \triangleq Q_j(y) \left[(g(x_j) - c_j(y))^2 - (f^\star(x_j; y) - c_j(y))^2 \right], \tag{15}$$

where (x_j, c_j) is the j^{th} example and cost presented to the algorithm and $Q_j(y) \in \{0,1\}$ is the query indicator. For simplicity we often write M_j when the dependence on g and y is clear from context. Let $\mathbb{E}_j[\cdot]$ and $\mathrm{Var}_j[\cdot]$ denote the expectation and variance conditioned on all randomness up to and including round $j-1$.

Theorem 15 *Let $\mathcal{G}$ be a function class with $\mathrm{Pdim}(\mathcal{G}) = d$, let $\delta \in (0,1)$ and define $\nu_n \triangleq 324(d\log(n) + \log(8Ke(d+1)n^2/\delta))$. Then with probability at least $1-\delta$, the following inequalities hold simultaneously for all $g \in \mathcal{G}$, $y \in [K]$, and $i < i' \in [n]$.*

$$\sum_{j=i}^{i'} M_j(g;y) \le \frac{3}{2}\sum_{j=i}^{i'} \mathbb{E}_j M_j(g;y) + \nu_n, \tag{16}$$

$$\frac{1}{2}\sum_{j=i}^{i'} \mathbb{E}_j M_j(g;y) \le \sum_{j=i}^{i'} M_j(g;y) + \nu_n. \tag{17}$$

This result is a uniform Freedman-type inequality for the martingale difference sequence $\sum_i M_i - \mathbb{E}_i M_i$. In general, such bounds require much stronger assumptions (e.g., sequential complexity measures (Rakhlin and Sridharan, 2017)) on $\mathcal{G}$ than the finite pseudo-dimension assumption that we make. However, by exploiting the structure of our particular martingale, specifically that the dependencies arise *only* from the query indicator, we are able to establish this type of inequality under weaker assumptions. The result may be of independent interest, but the proof, which is based on arguments from Liang et al. (2015), is quite technical and deferred to Appendix A. Note that we did not optimize the constants.

The Multiplicative Weights Algorithm. We also use the standard analysis of multiplicative weights for solving linear feasibility problems. We state the result here and, for completeness, provide a proof in Appendix B. See also Arora et al. (2012); Plotkin et al. (1995) for more details.

Consider a linear feasibility problem with decision variable $v \in \mathbb{R}^d$, explicit constraints $\langle a_i, v\rangle \le b_i$ for $i \in [m]$ and some implicit constraints $v \in S$ (e.g., v is non-negative or other simple constraints). The MW algorithm either finds an approximately feasible point or certifies that the program is infeasible assuming access to an oracle that can solve a simpler feasibility problem with just one explicit constraint $\sum_i \mu_i \langle a_i, v\rangle \le \sum_i \mu_i b_i$ for any non-negative weights $\mu \in \mathbb{R}^m_+$ and the implicit constraint $v \in S$. Specifically, given weights μ, the oracle either reports that the simpler problem is infeasible, or returns any feasible point v that further satisfies $\langle a_i, v\rangle - b_i \in [-\rho_i, \rho_i]$ for parameters ρ_i that are known to the MW algorithm.

The MW algorithm proceeds iteratively, maintaining a weight vector $\mu^{(t)} \in \mathbb{R}^m_+$ over the constraints. Starting with $\mu_i^{(1)} = 1$ for all i, at each iteration, we query the oracle with the weights $\mu^{(t)}$ and the oracle either returns a point v_t or detects infeasibility. In the latter case, we simply report infeasibility and in the former, we update the weights using the rule

$$\mu_i^{(t+1)} \leftarrow \mu_i^{(t)} \times \left(1 - \eta\frac{b_i - \langle a_i, v_t\rangle}{\rho_i}\right).$$

Here η is a parameter of the algorithm. The intuition is that if v_t satisfies the i^{th} constraint, then we down-weight the constraint, and conversely, we up-weight every constraint that is

violated. Running the algorithm with appropriate choice of η and for enough iterations is guaranteed to approximately solve the feasibility problem.

Theorem 16 (Arora et al. (2012); Plotkin et al. (1995)) *Consider running the MW algorithm with parameter $\eta = \sqrt{\log(m)/T}$ for T iterations on a linear feasibility problem where oracle responses satisfy $\langle a_i, v\rangle - b_i \in [-\rho_i, \rho_i]$. If the oracle fails to find a feasible point in some iteration, then the linear program is infeasible. Otherwise the point $\bar{v} \triangleq \frac{1}{T}\sum_{t=1}^T v_t$ satisfies $\langle a_i, \bar{v}\rangle \le b_i + 2\rho_i\sqrt{\log(m)/T}$ for all $i \in [m]$.*

Other Lemmata. Our first lemma evaluates the conditional expectation and variance of M_j, defined in (15), which we will use heavily in the proofs. Proofs of the results stated here are deferred to Appendix C.

Lemma 17 (Bounding variance of regression regret) *We have for all $(g,y) \in \mathcal{G}\times\mathcal{Y}$,*

$$\mathbb{E}_j[M_j] = \mathbb{E}_j\left[Q_j(y)(g(x_j) - f^\star(x_j;y))^2\right], \qquad \operatorname*{Var}_j[M_j] \le 4\mathbb{E}_i[M_j].$$

The next lemma relates the cost-sensitive error to the random variables M_j. Define

$$\mathcal{F}_i = \left\{f \in \mathcal{G}^K \mid \forall y, f(\cdot;y) \in \mathcal{G}_i(y)\right\},$$

which is the version space of vector regressors at round i. Additionally, recall that P_ζ captures the noise level in the problem, defined in (10) and that $\psi_i = 1/\sqrt{i}$ is defined in the algorithm pseudocode.

Lemma 18 *For all $i > 0$, if $f^\star \in \mathcal{F}_i$, then for all $f \in \mathcal{F}_i$*

$$\mathbb{E}_{x,c}[c(h_f(x)) - c(h_{f^\star}(x))] \le \min_{\zeta>0}\left\{\zeta P_\zeta + \mathbb{1}\left(\zeta \le 2\psi_i\right)2\psi_i + \frac{4\psi_i^2}{\zeta} + \frac{6}{\zeta}\sum_y \mathbb{E}_i\left[M_i\right]\right\}.$$

Note that the lemma requires that both $f^\star$ and f belong to the version space $\mathcal{F}_i$.

For the label complexity analysis, we will need to understand the cost-sensitive performance of all $f \in \mathcal{F}_i$, which requires a different generalization bound. Since the proof is similar to that of Theorem 5, we defer the argument to appendix.

Lemma 19 *Assuming the bounds in Theorem 15 hold, then for all i, $\mathcal{F}_i \subset \mathcal{F}_{csr}(r_i)$ where $r_i \triangleq \min_{\zeta>0}\left\{\zeta P_\zeta + \frac{44K\Delta_i}{\zeta}\right\}$.*

The final lemma relates the query rule of COAL to a hypothetical query strategy driven by $\mathcal{F}_{\text{csr}}(r_i)$, which we will subsequently bound by the disagreement coefficients. Let us fix the round i and introduce the shorthand $\widehat{\gamma}(x_i,y) = \widehat{c}_+(x_i,y) - \widehat{c}_-(x_i,y)$, where $\widehat{c}_+(x_i,y)$ and $\widehat{c}_-(x_i,y)$ are the approximate maximum and minimum costs computed in Algorithm 1 on the i^{th} example, which we now call x_i. Moreover, let Y_i be the set of non-dominated labels at round i of the algorithm, which in the pseudocode we call Y'. Formally, $Y_i = \{y \mid \widehat{c}_-(x_i,y) \le \min_{y'}\widehat{c}_+(x_i,y')\}$. Finally recall that for a set of vector regressors $F \subset \mathcal{F}$, we use $\gamma(x,y,F)$ to denote the cost range for label y on example x witnessed by the regressors in F.

Lemma 20 *Suppose that the conclusion of Lemma 19 holds. Then for any example x_i and any label y at round i, we have*

$$\widehat{\gamma}(x_i, y) \leq \gamma(x_i, y, \mathcal{F}_{csr}(r_i)) + \psi_i.$$

Further, with $y_i^\star = \operatorname{argmin}_y f^\star(x_i; y), \bar{y}_i = \operatorname{argmin}_y \widehat{c}_+(x_i, y)$, and $\tilde{y}_i = \operatorname{argmin}_{y \neq y_i^\star} \widehat{c}_-(x_i, y)$,

$$y \neq y_i^\star \wedge y \in Y_i \Rightarrow f^\star(x_i; y) - f^\star(x_i; y_i^\star) \leq \gamma(x_i, y, \mathcal{F}_{csr}(r_i)) + \gamma(x_i, y_i^\star, \mathcal{F}_{csr}(r_i)) + \psi_i/2,$$
$$|Y_i| > 1 \wedge y_i^\star \in Y_i \Rightarrow f^\star(x_i; \tilde{y}_i) - f^\star(x_i; y_i^\star) \leq \gamma(x_i, \tilde{y}_i, \mathcal{F}_{csr}(r_i)) + \gamma(x_i, y_i^\star, \mathcal{F}_{csr}(r_i)) + \psi_i/2.$$

8.2. Proof of Theorem 3

The proof is based on expressing the optimization problem (7) as a linear optimization in the space of distributions over $\mathcal{G}$. Then, we use binary search to re-formulate this as a series of feasibility problems and apply Theorem 16 to each of these.

Recall that the problem of finding the maximum cost for an (x, y) pair is equivalent to solving the program (7) in terms of the optimal g. For the problem (7), we further notice that since $\mathcal{G}$ is a convex set, we can instead write the minimization over g as a minimization over $P \in \Delta(\mathcal{G})$ without changing the optimum, leading to the modified problem (8).

Thus we have a linear program in variable P, and Algorithm 2 turns this into a feasibility problem by guessing the optimal objective value and refining the guess using binary search. For each induced feasibility problem, we use MW to certify feasibility. Let $c \in [0, 1]$ be some guessed upper bound on the objective, and let us first turn to the MW component of the algorithm. The program in consideration is

$$?\exists P \in \Delta(G) \text{ s.t. } \mathbb{E}_{g\sim P}(g(x_i) - 1)^2 \leq c \text{ and } \forall j \in [i], \mathbb{E}_{g\sim P}\widehat{R}_j(g; y) \leq \tilde{\Delta}_j. \tag{18}$$

This is a linear feasibility problem in the infinite dimensional variable P, with $i + 1$ constraints. Given a particular set of weights μ over the constraints, it is clear that we can use the regression oracle over g to compute

$$g_\mu = \arg\min_{g\in\mathcal{G}} \mu_0(g(x_i) - 1)^2 + \sum_{j\in[i]} \mu_j \mathbb{E}_{g\sim P}\widehat{R}_j(g; y). \tag{19}$$

Observe that solving this simpler program provides one-sided errors. Specifically, if the objective of (19) evaluated at g_μ is larger than $\mu_0 c + \sum_{j\in[i]} \mu_j \tilde{\Delta}_j$ then there cannot be a feasible solution to problem (18), since the weights μ are all non-negative. On the other hand if g_μ has small objective value it does not imply that g_μ is feasible for the original constraints in (18).

At this point, we would like to invoke the MW algorithm, and specifically Theorem 16, in order to find a feasible solution to (18) or to certify infeasibility. Invoking the theorem requires the ρ_j parameters which specify how badly g_μ might violate the j^{th} constraint. For us, $\rho_j \triangleq \kappa$ suffices since $\hat{R}_j(g; y) - \hat{R}_j(g_{j,y}; y) \in [0, 1]$ (since $g_{j,y}$ is the ERM) and $\Delta_j \leq \kappa$. Since $\kappa \geq 2$ this also suffices for the cost constraint.

If at any iteration, MW detects infeasibility, then our guessed value c for the objective is too small since no function satisfies both $(g(x_i) - 1)^2 \leq c$ and the empirical risk constraints in (18) simultaneously. In this case, in Line 10 of Algorithm 2, our binary search procedure

increases our guess for c. On the other hand, if we apply MW for T iterations and find a feasible point in every round, then, while we do not have a point that is feasible for the original constraints in (18), we will have a distribution P_T such that

$$\mathbb{E}_{P_T}(g(x_i) - 1)^2 \leq c + 2\kappa\sqrt{\frac{\log(i+1)}{T}} \text{ and } \forall j \in [i], \mathbb{E}_{P_T}\widehat{R}_j(g; y) \leq \tilde{\Delta}_j + 2\kappa\sqrt{\frac{\log(i+1)}{T}}.$$

We will set T toward the end of the proof.

If we do find an approximately feasible solution, then we reduce c and proceed with the binary search. We terminate when $c_h - c_\ell \leq \tau^2/2$ and we know that problem (18) is approximately feasible with c_h and infeasible with c_ℓ. From c_h we will construct a strictly feasible point, and this will lead to a bound on the true maximum $c_+(x, y, \mathcal{G}_i)$.

Let $\bar{P}$ be the approximately feasible point found when running MW with the final value of c_h. By Jensen's inequality and convexity of $\mathcal{G}$, there exists a single regressor that is also approximately feasible, which we denote $\bar{g}$. Observe that $g^\star$ satisfies all constraints with strict inequality, since by (20) we know that $\widehat{R}_j(g^\star; y) - \widehat{R}_j(g_{j,y}; y) \leq \Delta_j/\kappa < \Delta_j$. We create a strictly feasible point g_ζ by mixing $\bar{g}$ with $g^\star$ with proportion $1 - \zeta$ and ζ for

$$\zeta = \frac{4\kappa}{\Delta_i}\sqrt{\frac{\log(i+1)}{T}},$$

which will be in $[0, 1]$ when we set T. Combining inequalities, we get that for any $j \in [i]$

$$\begin{aligned}(1-\zeta)\widehat{R}_j(\bar{g}; y) + \zeta\widehat{R}_j(g^\star; y) &\leq \widehat{R}_j(g_{j,y}; y) + (1-\zeta)\left(\Delta_j + 2\kappa\sqrt{\frac{\log(i+1)}{T}}\right) + \zeta\left(\frac{\Delta_j}{\kappa}\right) \\ &\leq \widehat{R}_j(g_{j,y}; y) + \Delta_j - \left(\frac{\zeta\Delta_j(\kappa-1)}{\kappa} - 2\kappa\sqrt{\frac{\log(i+1)}{T}}\right) \\ &\leq \widehat{R}_j(g_{j,y}; y) + \Delta_j,\end{aligned}$$

and hence this mixture regressor g_ζ is exactly feasible. Here we use that $\kappa \geq 2$ and that Δ_i is monotonically decreasing. With the pessimistic choice $g^\star(x_i) = 0$, the objective value for g_ζ is at most

$$\begin{aligned}(g_\zeta(x_i) - 1)^2 &\leq (1-\zeta)(\bar{g}(x_i) - 1)^2 + \zeta(g^\star(x_i) - 1)^2 \leq (1-\zeta)\left(c_h + 2\kappa\sqrt{\frac{\log(i+1)}{T}}\right) + \zeta \\ &\leq c_\ell + \tau^2/2 + \left(2\kappa + \frac{4\kappa}{\Delta_i}\right)\sqrt{\frac{\log(i+1)}{T}}.\end{aligned}$$

Thus g_ζ is *exactly* feasible and achieves the objective value above, which provides an upper bound on the maximum cost. On the other hand c_ℓ provides a lower bound. Our setting of $T = \frac{\log(i+1)(8\kappa^2/\Delta_i)^2}{\tau^4}$ ensures that that this excess term is at most τ^2, since $\Delta_i \leq 1$. Note that since $\tau \leq [0, 1]$, this also ensures that $\zeta \in [0, 1]$. With this choice of T, we know that $c_\ell \leq (c_+(y) - 1)^2 \leq c_\ell + \tau^2$, which implies that $c_+(y) \in [1 - \sqrt{c_\ell + \tau^2}, 1 - \sqrt{c_\ell}]$. Since $\sqrt{c_\ell + \tau^2} \leq \sqrt{c_\ell} + \tau$, we obtain the guarantee.

As for the oracle complexity, since we start with $c_\ell = 0$ and $c_h = 1$ and terminate when $c_h - c_\ell \le \tau^2/2$, we perform $\mathcal{O}(\log(1/\tau^2))$ iterations of binary search. Each iteration requires $T = \mathcal{O}\left(\max\{1, i^2/\nu_n^2\}\frac{\log(i)}{\tau^4}\right)$ rounds of MW, each of which requires exactly one oracle call. Hence the oracle complexity is $\mathcal{O}\left(\max\{1, i^2/\nu_n^2\}\frac{\log(i)\log(1/\tau)}{\tau^4}\right)$. ■

8.3. Proof of the Generalization Bound

Recall the central random variable $M_j(g; y)$, defined in (15), which is the excess square loss for function g on label y for the j^{th} example, if we issued a query. The idea behind the proof is to first apply Theorem 15 to argue that all the random variables $M_j(g; y)$ concentrate uniformly over the function class $\mathcal{G}$. Next for a vector regressor f, we relate the cost-sensitive risk to the excess square loss via Lemma 18. Finally, using the fact that $g_{i,y}$ minimizes the empirical square loss at round i, this implies a cost-sensitive risk bound for the vector regressor $f_i = (g_{i,y})$ at round i.

First, condition on the high probability event in Theorem 15, which ensures that the empirical square losses concentrate. We first prove that $f^\star \in \mathcal{F}_i$ for all $i \in [n]$. At round i, by (17), for each y and for any g we have

$$0 \le \frac{1}{2}\sum_{j=1}^{i} \mathbb{E}_j M_j(g; y) \le \sum_{j=1}^{i} M_j(g; y) + \nu_n.$$

The first inequality here follows from the fact that $\mathbb{E}_j M_j(g; y)$ is a quadratic form by Lemma 17. Expanding $M_j(g; y)$, this implies that

$$\widehat{R}_{i+1}(f^\star(\cdot; y); y) \le \widehat{R}_{i+1}(g; y) + \frac{\nu_n}{i}.$$

Since this bound applies to all $g \in \mathcal{G}$ it proves that $f^\star \in \mathcal{F}_{i+1}$ for all i, using the definition of Δ_i and κ. Trivially, we know that $f^\star \in \mathcal{F}_1$. Together with the fact that the losses are in $[0, 1]$ and the definition of Δ_i, the above analysis yields

$$\widehat{R}_i(f^\star(\cdot; y); y) \le \widehat{R}_i(g; y) + \frac{\Delta_i}{\kappa}. \tag{20}$$

This implies that $f^\star(\cdot; y)$ strictly satisfies the inequalities defining the version space, which we used in the MW proof.

We next prove that $f_{i+1} \in \mathcal{F}_j$ for all $j \in [i]$. Fix some label y and to simplify notation, we drop dependence on y. If $g_{i+1} \notin \mathcal{G}_{t+1}$ for some $t \in \{0, \dots, i\}$ then, first observe that we must have t large enough so that $\nu_n/t \le 1$. In particular, since $\Delta_{t+1} = \kappa \min\{1, \nu_n/t\}$ and we always have $\hat{R}_{t+1}(g_{i+1}; y) \le \hat{R}_{t+1}(g_{t+1}; y) + 1$ due to boundedness, we do not evict any functions until $\nu_n/t \le 1$. For $t \ge \nu_n$, we get

$$\begin{aligned}
\sum_{j=1}^{t} M_j &= t\left(\widehat{R}_{t+1}(g_{i+1}) - \widehat{R}_{t+1}(g^\star)\right) \\
&= t\left(\widehat{R}_{t+1}(g_{i+1}) - \widehat{R}_{t+1}(g_{t+1}) + \widehat{R}_{t+1}(g_{t+1}) - \widehat{R}_{t+1}(g^\star)\right) \\
&\ge \kappa\nu_n - \nu_n = (\kappa - 1)\nu_n.
\end{aligned}$$

The inequality uses the radius of the version space and the fact that by assumption $g_{i+1} \notin \mathcal{G}_{t+1}$, so the excess empirical risk is at least $\Delta_{t+1} = \kappa\nu_n/t$ since we are considering large t. We also use (20) on the second term. Moreover, we know that since g_{i+1} is the empirical square loss minimizer for label y after round i, we have $\sum_{j=1}^{i} M_j(g_{i+1}; y) \le 0$. These two facts together establish that

$$\sum_{j=t+1}^{i} M_j(g_{i+1}) \le (1 - \kappa)\nu_n.$$

However, by Theorem 15 on this intermediary sum, we know that

$$0 \le \frac{1}{2}\sum_{j=t+1}^{i} \mathbb{E}_j M_j(g_{i+1}) \le \sum_{j=t+1}^{i} M_j(g_{i+1}) + \nu_n < (2 - \kappa)\nu_n < 0$$

using the definition of κ. This is a contradiction, so we must have that $g_{i+1} \in \mathcal{G}_j$ for all $j \in \{1, \dots, i\}$. The same argument applies for all y and hence we can apply Lemma 18 on all rounds to obtain

$$\begin{aligned}
&i\big(\mathbb{E}_{x,c}[c(h_{f_{i+1}}(x)) - c(h_{f^\star}(x))]\big) \\
&\le \min_{\zeta > 0}\left\{ i\zeta P_\zeta + \sum_{j=1}^{i}\left(\mathbb{1}\left(\zeta \le 2\psi_j\right) 2\psi_j + \frac{4\psi_j^2}{\zeta} + \frac{6}{\zeta}\sum_y \mathbb{E}_j\left[M_j(f_{i+1}; y)\right]\right)\right\}.
\end{aligned}$$

We study the four terms separately. The first one is straightforward and contributes ζP_ζ to the instantaneous cost sensitive regret. Using our definition of $\psi_j = 1/\sqrt{j}$ the second term can be bounded as

$$\sum_{j=1}^{i} \mathbb{1}\left(\zeta < 2\psi_j\right) 2\psi_j = \sum_{j=1}^{\lceil 4/\zeta^2 \rceil} \frac{2}{\sqrt{j}} \le 4\sqrt{\lceil 4/\zeta^2 \rceil} \le \frac{12}{\zeta}.$$

The inequality above, $\sum_{i=1}^{n} \frac{1}{\sqrt{i}} \le 2\sqrt{n}$, is well known. For the third term, using our definition of ψ_j gives

$$\sum_{j=1}^{i} \frac{4\psi_j^2}{\zeta} = \frac{4}{\zeta}\sum_{j=1}^{i} \frac{1}{j} \le \frac{4}{\zeta}(1 + \log(i)).$$

Finally, the fourth term can be bounded using (17), which reveals

$$\sum_{j=1}^{i} \mathbb{E}_j[M_j] \le 2\sum_{j=1}^{i} M_j + 2\nu_n$$

Since for each y, $\sum_{j=1}^{i} M_j(f_{i+1}; y) \le 0$ for the empirical square loss minimizer (which is what we are considering now), we get

$$\frac{6}{\zeta}\sum_y \sum_{j=1}^{i} \mathbb{E}_j[M_j(f_{i+1}; y)] \le \frac{12}{\zeta} K\nu_n.$$

And hence, we obtain the generalization bound

$$\mathbb{E}_{x,c}[c(x;h_{f_{i+1}}(x)) - c(x;h_{f^\star}(x))] \le \min_{\zeta>0}\left\{\zeta P_\zeta + \frac{1}{\zeta i}\left(4\log(i) + 16 + 12K\nu_n\right)\right\}$$
$$\le \min_{\zeta>0}\left\{\zeta P_\zeta + \frac{32K\nu_n}{\zeta i}\right\}. \qquad \blacksquare$$

Proof of Corollary 6. Under the Massart noise condition, set $\zeta = \tau$ so that $P_\zeta = 0$ and we immediately get the result. $\blacksquare$

Proof of Corollary 7. Set $\zeta = \min\left\{\tau_0, \left(\frac{32K\nu_n}{i\beta}\right)^{\frac{1}{\alpha+2}}\right\}$, so that for i sufficiently large the second term is selected and we obtain the bound. $\blacksquare$

8.4. Proof of the Label Complexity bounds

The proof for the label complexity bounds is based on first relating the version space $\mathcal{F}_i$ at round i to the cost-sensitive regret ball $\mathcal{F}_{\text{csr}}$ with radius r_i. In particular, the containment $\mathcal{F}_i \subset \mathcal{F}_{\text{csr}}(r_i)$ in Lemma 19 implies that our query strategy is more aggressive than the query strategy induced by $\mathcal{F}_{\text{csr}}(r_i)$, except for a small error introduced when computing the maximum and minimum costs. This error is accounted for by Lemma 20. Since the probability that $\mathcal{F}_{\text{csr}}$ will issue a query is intimately related to the disagreement coefficient, this argument leads to the label complexity bounds for our algorithm.

Proof of Theorem 9. Fix some round i with example x_i, let $\mathcal{F}_i$ be the vector regressors used at round i and let $\mathcal{G}_i(y)$ be the corresponding regressors for label y. Let $\bar{y}_i = \operatorname{argmin}_y \widehat{c_+}(x_i, y), y_i^\star = \operatorname{argmin}_y f^\star(x_i; y)$, and $\tilde{y}_i = \operatorname{argmin}_{y\neq y_i^\star} \widehat{c_-}(x_i, y)$. Assume that Lemma 19 holds. The L_2 label complexity is

$$\sum_y Q_i(y) = \sum_y \mathbf{1}\{|Y_i| > 1 \wedge y \in Y_i\}\mathbf{1}\{\widehat{\gamma}(x_i, y) > \psi_i\}$$
$$\le \sum_y \mathbf{1}\{|Y_i| > 1 \wedge y \in Y_i\}\mathbf{1}\{\gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) > \psi_i/2\}.$$

For the former indicator, observe that $y \in Y_i$ implies that there exists a vector regressor $f \in \mathcal{F}_i \subset \mathcal{F}_{\text{csr}}(r_i)$ such that $h_f(x_i) = y$. This follows since the domination condition means that there exists $g \in \mathcal{G}_i(y)$ such that $g(x_i) \le \min_{y'} \max_{g'\in\mathcal{G}_i(y')} g'(x_i)$. Since we are using a factored representation, we can take f to use g on the y^{th} coordinate and use the maximizers for all the other coordinates. Similarly, there exists another regressor $f' \in \mathcal{F}_i$ such that $h_{f'}(x_i) \neq y$. Thus this indicator can be bounded by the disagreement coefficient

$$\sum_y Q_i(y) \le \sum_y \mathbf{1}\{\exists f, f' \in \mathcal{F}_{\text{csr}}(r_i) \mid h_f(x_i) = y \neq h_{f'}(x_i)\}\mathbf{1}\{\gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) > \psi_i/2\}$$
$$= \sum_y \mathbf{1}\{x \in \text{DIS}(r_i, y) \wedge \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) \ge \psi_i/2\}.$$

We will now apply Freedman's inequality on the sequence $\{\sum_y Q_i(y)\}_{i=1}^n$, which is a martingale with range K. Moreover, due to non-negativity, the conditional variance is at most

K times the conditional mean, and in such cases, Freedman's inequality reveals that with probability at least $1-\delta$

$$X \le \mathbb{E}X + 2\sqrt{R\mathbb{E}X\log(1/\delta)} + 2R\log(1/\delta) \le 2\mathbb{E}X + 3R\log(1/\delta),$$

where X is the non-negative martingale with range R and expectation $\mathbb{E}X$. The last step is by the fact that $2\sqrt{ab} \le a + b$.

For us, Freedman's inequality implies that with probability at least $1-\delta/2$

$$\sum_{i=1}^n \sum_y Q_i(y) \le 2\sum_i \mathbb{E}_i \sum_y Q_i(y) + 3K\log(2/\delta)$$
$$\le 2\sum_i \mathbb{E}_i \sum_y \mathbf{1}\{x \in \text{DIS}(r_i, y) \wedge \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) \ge \psi_i/2\} + 3K\log(2/\delta)$$
$$\le 4\sum_i \frac{r_i}{\psi_i}\theta_2 + 3K\log(2/\delta).$$

The last step here uses the definition of the disagreement coefficient θ_2. To wrap up the proof we just need to upper bound the sequence, using our choices of $\psi_i = 1/\sqrt{i}$, $r_i = 2\sqrt{44K\Delta_i}$, and $\Delta_i = \kappa\min\{1, \frac{\nu_n}{i-1}\}$. With simple calculations this is easily seen to be at most

$$8n\theta_2\sqrt{88K\kappa\nu_n} + 3K\log(2/\delta).$$

Similarly for L_1 we can derive the bound

$$L_1 \le \sum_i \mathbb{1}\left(\exists y \mid \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) \ge \psi_i/2 \wedge x \in \text{DIS}(r_i, y)\right),$$

and then apply Freedman's inequality to obtain that with probability at least $1-\delta/2$

$$L_1 \le 2\sum_{i=1}^n \frac{2r_i}{\psi_i}\theta_1 + 3\log(2/\delta) \le 8\theta_1 n\sqrt{88K\kappa\nu_n} + 3\log(2/\delta). \qquad \blacksquare$$

Proof of Theorem 10. Using the same notations as in the bound for the high noise case we first express the L_2 label complexity as

$$\sum_y Q_i(y) = \sum_y \mathbf{1}\{|Y_i| > 1, y \in Y_i\}Q_i(y).$$

We need to do two things with the first part of the query indicator, so we have duplicated it here. For the second, we will use the derivation above to relate the query rule to the disagreement region. For the first, by Lemma 20, for $y \neq y_i^\star$, we can derive the bound

$$\mathbf{1}\{f^\star(x_i; y) - f^\star(x_i; y_i^\star) \le \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) + \gamma(x_i, y_i^\star, \mathcal{F}_{\text{csr}}(r_i)) + \psi_i/2\}$$
$$\le \mathbf{1}\{\tau - \psi_i/2 \le \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) + \gamma(x_i, y_i^\star, \mathcal{F}_{\text{csr}}(r_i))\}.$$

For $y = y_i^\star$, we get the same bound but with $\tilde{y}_i$, also by Lemma 20. Focusing on just one of these terms, say where $y \neq y_i^\star$ and any round where $\tau \geq \psi_i$, we get

$$\begin{aligned}
&\mathbf{1}\{\tau - \psi_i/2 \leq \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) + \gamma(x_i, y_i^\star, \mathcal{F}_{\text{csr}}(r_i))\}Q_i(y) \\
&\leq \mathbf{1}\{\tau/2 \leq \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) + \gamma(x_i, y_i^\star, \mathcal{F}_{\text{csr}}(r_i))\}Q_i(y) \\
&\leq \mathbf{1}\{\tau/4 \leq \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i))\}Q_i(y) + \mathbf{1}\{\tau/4 \leq \gamma(x_i, y_i^\star, \mathcal{F}_{\text{csr}}(r_i))\}Q_i(y) \\
&\leq D_i(y) + D_i(y_i^\star),
\end{aligned}$$

where for shorthand we have defined $D_i(y) \triangleq \mathbf{1}\{\tau/4 \leq \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) \wedge x_i \in \text{DIS}(r_i, y)\}$. The derivation for the first term is straightforward. We obtain the disagreement region for $y_i^\star$ since the fact that we query y (i.e. $Q_i(y)$) implies there is f such that $h_f(x_i) = y$, so this function witnesses disagreement to $y_i^\star$.

The term involving $y_i^\star$ and $\tilde{y}_i$ is bounded in essentially the same way, since we know that when $|Y_i| > 1$, there exists two function $f, f' \in \mathcal{F}_i$ such that $h_f(x_i) = \tilde{y}_i$ and $h_{f'}(x_i) = y_i^\star$. In total, we can bound the L_2 label complexity at any round i such that $\tau \geq \psi_i$ by

$$L_2 \leq D_i(\tilde{y}_i) + D_i(y_i^\star) + \sum_{y \neq y_i^\star} (D_i(y) + D_i(y_i^\star)) \leq K D_i(y_i^\star) + 2\sum_y D_i(y)$$

For the earlier rounds, we simply upper bound the label complexity by K. Since the range of this random variable is at most $3K$, using Freedman's inequality just as in the high noise case, we get that with probability at least $1 - \delta/2$

$$\begin{aligned}
L_2 &\leq \sum_{i=1}^n K\mathbf{1}\{\tau \leq \psi_i\} + 2\sum_{i=2}^n \mathbb{E}_i\left[K D_i(y_i^\star) + 2\sum_y D_i(y)\right] + 9K\log(2/\delta) \\
&\leq K\lceil 1/\tau^2 \rceil + \frac{8}{\tau}(K\theta_1 + \theta_2)\sum_{i=2}^n r_i + 9K\log(2/\delta).
\end{aligned}$$

The first line here is the application of Freedman's inequality. In the second, we evaluate the expectation, which we can relate to the disagreement coefficients θ_1, θ_2. Moreover, we use the setting $\psi_i = 1/\sqrt{i}$ to evaluate the first term. As a technicality, we remove the index $i = 1$ from the second summation, since we are already accounting for queries on the first round in the first term. The last step is to evaluate the series, for which we use the definition of $r_i = \min_{\zeta > 0}\{\zeta P_\zeta + 44K\Delta_i/\zeta\}$ and set $\zeta = \tau$, the Massart noise level. This gives $r_i = 44K\Delta_i/\tau$. In total, we get

$$L_2 \leq K\lceil 1/\tau^2 \rceil + \frac{352K\kappa\nu_n}{\tau^2}(K\theta_1 + \theta_2)\log(n) + 9K\log(2/\delta).$$

As $\theta_2 \leq K\theta_1$ always, we drop θ_2 from the above expression to obtain the stated bound.

For L_1 we use a very similar argument. First, by Lemmas 19 and 20

$$L_1 \leq \sum_{i=1}^n \mathbf{1}\{|Y_i| > 1 \wedge \exists y \in Y_i, \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) \geq \psi_i/2\}.$$

Again by Lemma 20, we know that

$$|Y_i| > 1 \wedge y \in Y_i \Rightarrow \exists f, f' \in \mathcal{F}_{\text{csr}}(r_i), h_f(x_i) = y \neq h_{f'}(x_i).$$

Moreover, one of the two classifiers can be $f^\star$, and so, when $\tau \geq \psi_i$, we can deduce

$$\begin{aligned}
&\Rightarrow f^\star(x_i, y) - f^\star(x_i, y_i^\star) \leq \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) + \gamma(x_i, y_i^\star, \mathcal{F}_{\text{csr}}(r_i)) + \psi_i/2 \\
&\Rightarrow \tau/4 \leq \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) \wedge \tau/4 \leq \gamma(x_i, y_i^\star, \mathcal{F}_{\text{csr}}(r_i)).
\end{aligned}$$

Combining this argument, we bound the L_1 label complexity as

$$L_1 \leq \sum_{i=1}^n \mathbf{1}\{\tau \leq \psi_i\} + \sum_{i=2}^n \mathbf{1}\{\exists y \mid x_i \in \text{DIS}(\mathcal{F}_{\text{csr}}(r_i), y) \wedge \gamma(x_i, y, \mathcal{F}_{\text{csr}}(r_i)) \geq \tau/4\}. \quad (21)$$

Applying Freedman's inequality just as before gives

$$L_1 \leq \lceil 1/\tau^2 \rceil + 2\sum_{i=2}^n \frac{4r_i}{\tau}\theta_1 + 2\log(2/\delta) \leq \lceil 1/\tau^2 \rceil + \frac{352K\kappa\nu_n}{\tau^2}\theta_1\log(n) + 2\log(2/\delta),$$

with probability at least $1 - \delta/2$. ■

Proof of Theorem 11. For the Tsybakov case, the same argument as in the Massart case gives that with probability at least $1 - \delta/2$

$$L_2 \leq K\lceil 1/\tau^2 \rceil + \frac{8}{\tau}(K\theta_1 + \theta_2)\sum_{i=2}^n r_i + 2n\beta\tau^\alpha + 9K\log(2/\delta).$$

The main difference here is the term scaling with $n\tau^\alpha$ which arises since we do not have the deterministic bound $f^\star(x_i, y) - f^\star(x_i, y_i^\star) \geq \tau$ as we used in the Massart case, but rather this happens except with probability $\beta\tau^\alpha$ (provided $\tau \leq \tau_0$). Now we must optimize ζ in the definition of r_i and then τ.

For ζ the optimal setting is $(44K\Delta_i/\beta)^{\frac{1}{\alpha+2}}$ which gives $r_i \leq 2\beta^{\frac{1}{\alpha+2}}(44K\Delta_i)^{\frac{\alpha+1}{\alpha+2}}$. Since we want to set $\zeta \leq \tau_0$, this requires $i \geq 1 + \frac{44K\kappa\nu_n}{\beta\tau_0^{\alpha+2}}$. For these early rounds we will simply pay K in the label complexity, but this will be dominated by other higher order terms. For the later rounds, we get

$$\sum_{i=2}^n r_i \leq 2\beta^{\frac{1}{\alpha+2}}(44K\kappa\nu_n)^{\frac{\alpha+1}{\alpha+2}}\sum_{i=2}^n\left(\frac{1}{i-1}\right)^{\frac{\alpha+1}{\alpha+2}} \leq 2(\alpha+2)(44K\kappa\nu_n)^{\frac{\alpha+1}{\alpha+2}}(\beta n)^{\frac{1}{\alpha+2}}.$$

This bound uses the integral approximation $\sum_{i=2}^n (i-1)^{-\frac{\alpha+1}{\alpha+2}} \leq 1 + \int_1^{n-1} x^{-\frac{\alpha+1}{\alpha+2}}dx \leq (\alpha+2)n^{\frac{1}{\alpha+2}}$. At this point, the terms involving τ in our bound are

$$K\lceil 1/\tau^2 \rceil + \frac{16}{\tau}(K\theta_1 + \theta_2)(\alpha+2)(44K\kappa\nu_n)^{\frac{\alpha+1}{\alpha+2}}(\beta n)^{\frac{1}{\alpha+2}} + 2n\beta\tau^\alpha.$$

We set $\tau = (8(\alpha+2)(K\theta_1+\theta_2))^{\frac{1}{\alpha+1}}(44K\kappa\nu_n)^{\frac{1}{\alpha+2}}(\beta n)^{\frac{-1}{\alpha+2}}$ by optimizing the second two terms which gives a final bound of

$$L_2 \leq \mathcal{O}\left((K\theta_1+\theta_2)^{\frac{\alpha}{\alpha+1}}(K\nu_n)^{\frac{\alpha}{\alpha+2}}n^{\frac{2}{\alpha+2}} + K\log(1/\delta)\right).$$

This follows since the $1/\tau^2$ term agrees in the n dependence and is lower order in other parameters, while the unaccounted for querying in the early rounds is independent of n. The bound of course requires that $\tau \leq \tau_0$, which again requires n large enough. Note we are treating α and β as constants and we drop θ_2 from the final statement.

The L_1 bound requires only slightly different calculations. Following the derivation for the Massart case, we get

$$\begin{aligned} L_1 &\leq \lceil 1/\tau^2 \rceil + \frac{8\theta_1}{\tau}\sum_{i=2}^{n} r_i + 2n\beta\tau^\alpha + 2\log(2/\delta) \\ &\leq \lceil 1/\tau^2 \rceil + \frac{16\theta_1(\alpha+2)}{\tau}\left(44K\kappa\nu_n\right)^{\frac{\alpha+1}{\alpha+2}}(\beta n)^{\frac{1}{\alpha+2}} + 2n\beta\tau^\alpha + 2\log(2/\delta), \end{aligned}$$

not counting the lower order term for the querying in the early rounds. Here we set $\tau = (8(\alpha+2)\theta_1)^{\frac{1}{\alpha+1}}\left(44K\kappa\nu_n\right)^{\frac{1}{\alpha+2}}(\beta n)^{\frac{-1}{\alpha+2}}$ to obtain

$$L_1 \leq \mathcal{O}\left(\theta_1^{\frac{\alpha}{\alpha+1}}(K\nu_n)^{\frac{\alpha}{\alpha+2}}n^{\frac{2}{\alpha+2}} + \log(1/\delta)\right). \qquad \blacksquare$$

Proof of Proposition 12: Massart Case. Observe that with Massart noise, we have $\mathcal{F}_{\mathrm{csr}}(r) \subset \widetilde{\mathcal{F}}(r/\tau)$, which implies that

$$\begin{aligned} \mathbb{E}\mathbf{1}\left\{\exists y \mid x \in \mathrm{DIS}(r,y) \wedge \gamma(x,y,\mathcal{F}_{\mathrm{csr}}(r)) \geq \tau/4\right\} &\leq \mathbb{E}\mathbf{1}\left\{x \in \widetilde{\mathrm{DIS}}(r/\tau)\right\} \\ &\leq \frac{r}{\tau}\sup_{r>0}\frac{1}{r}\mathbb{E}\mathbf{1}\left\{x \in \widetilde{\mathrm{DIS}}(r)\right\}. \end{aligned}$$

Thus we may replace θ_1 with θ_0 in the proof of the L_1 label complexity bound above. $\blacksquare$

Proof of Proposition 12: Tsybakov Case. The proof is identical to Theorem 11 except we must introduce the alternative coefficient θ_0. To do so, define $\mathcal{X}(\tau) \triangleq \{x \in \mathcal{X} : \min_{y \neq y^\star(x)} f^\star(x,y) - f^\star(x,y^\star(x)) \geq \tau\}$, and note that under the Tsybakov noise condition, we have $\mathbb{P}[x \notin \mathcal{X}(\tau)] \leq \beta\tau^\alpha$ for all $0 \leq \tau \leq \tau_0$. For such a value of τ we have

$$\begin{aligned} \mathbb{P}[h_f(x) \neq h_{f^\star}(x)] &\leq \mathbb{E}\mathbf{1}\{x \in \mathcal{X}(\tau) \wedge h_f(x) \neq h_{f^\star}(x)\} + \mathbb{P}[x \notin \mathcal{X}(\tau)] \\ &\leq \frac{1}{\tau}\mathbb{E}c(h_f(x)) - c(h_{f^\star}(x)) + \beta\tau^\alpha. \end{aligned}$$

We use this fact to prove that $\mathcal{F}_{\mathrm{csr}}(r) \subset \widetilde{\mathcal{F}}(2r/\tau)$ for τ sufficiently small. This can be seen from above by noting that if $f \in \mathcal{F}_{\mathrm{csr}}(r)$ then the right hand side is $\frac{r}{\tau} + \beta\tau^\alpha$, and if $\tau \leq (r/\beta)^{\frac{1}{1+\alpha}}$ the containment holds. Therefore, we have

$$\begin{aligned} \mathbb{E}\mathbf{1}\left\{\exists y \mid x \in \mathrm{DIS}(r,y) \wedge \gamma(x,y,\mathcal{F}_{\mathrm{csr}}(r)) \geq \tau/4\right\} &\leq \mathbb{E}\mathbf{1}\left\{\exists y \mid x \in \mathrm{DIS}(r,y)\right\} \\ &\leq \mathbb{E}\mathbf{1}\left\{x \in \widetilde{\mathrm{DIS}}(2r/\tau)\right\} \\ &\leq \frac{2r}{\tau}\cdot\sup_{r>0}\mathbb{E}\mathbf{1}\left\{x \in \widetilde{\mathrm{DIS}}(r)\right\}. \end{aligned}$$

Thus, provided $\tau \leq (r_n/\beta)^{\frac{1}{\alpha+1}}$, we can replace θ_1 with θ_0 in the above argument. This gives

$$\begin{aligned} L_1 &\leq \lceil 1/\tau^2 \rceil + \frac{4\theta_0}{\tau}\sum_{i=2}^{n} r_i + 2n\beta\tau^\alpha + 2\log(2/\delta) \\ &\leq \lceil 1/\tau^2 \rceil + \frac{8\theta_0(\alpha+2)}{\tau}\left(44K\kappa\nu_n\right)^{\frac{\alpha+1}{\alpha+2}}(\beta n)^{\frac{1}{\alpha+2}} + 2n\beta\tau^\alpha + 2\log(2/\delta). \end{aligned}$$

As above, we have taken $r_i = 2\beta^{\frac{1}{\alpha+2}}(44K\Delta_i)^{\frac{\alpha+1}{\alpha+2}}$ and approximated the sum by an integral. Since $\Delta_n = \kappa\nu_n/(n-1)$, we can set $\tau = 2^{\frac{\alpha+1}{\alpha+2}}(44K\kappa\nu_n)^{\frac{1}{\alpha+2}}(\beta n)^{-\frac{1}{\alpha+2}}$. This is a similar choice to what we used in the proof of Theorem 11 except that we are not incorporating θ_0 into the choice of τ, and it yields a final bound of $\mathcal{O}\left(\theta_0 n^{\frac{2}{\alpha+2}}\nu_n^{\frac{\alpha}{\alpha+2}} + \log(1/\delta)\right)$. $\blacksquare$

Proof of Proposition 13. First we relate θ_1 to θ_0 in the multiclass case. For $f \in \mathcal{F}_{\mathrm{csr}}(r)$, we have

$$\begin{aligned} \mathbb{P}[h_f(x) \neq h_{f^\star}(x)] &\leq \mathbb{P}[h_f(x) \neq y] + \mathbb{P}[h_{f^\star}(x) \neq y] = \mathbb{E}c(h_f(x)) - c(h_{f^\star}(x)) \\ &\leq 2\mathrm{error}(h_{f^\star}) + r \end{aligned}$$

Therefore for any $r > 0$ and any x,

$$\mathbf{1}\{\exists y \mid x \in \mathrm{DIS}(r,y) \wedge \gamma(x,y,\mathcal{F}_{\mathrm{csr}}(r)) \geq \psi/2\} \leq \mathbf{1}\{x \in \widetilde{\mathrm{DIS}}(r_i + 2\mathrm{error}(h_{f^\star}))\}.$$

Applying this argument in the L_1 derivation above, we obtain

$$\begin{aligned} L_1 &\leq 2\sum_i \mathbb{E}_i\mathbf{1}\{x \in \widetilde{\mathrm{DIS}}(r_i + 2\mathrm{error}(h_{f^\star}))\} + 3\log(2/\delta) \\ &\leq 2\sum_i (r_i + 2\mathrm{error}(h_{f^\star}))\theta_0 + 3\log(2/\delta). \end{aligned}$$

We now bound r_i via Lemma 19. In multiclass classification, the fact that $c = \mathbf{1} - e_y$ for some y implies that $f^\star(x,y)$ is one minus the probability that the true label is y. Thus $f^\star(x,y) \in [0,1]$, $\sum_y f^\star(x,y) = K-1$, and for any x we always have

$$\min_{y \neq y^\star(x)} f^\star(x,y) - f^\star(x,y^\star(x)) \geq 1 - 2f^\star(x,y^\star(x)).$$

Hence, we may bound P_ζ, for $\zeta \leq 1/2$, as follows

$$\begin{aligned} P_\zeta &= \mathbb{P}_{x\sim\mathcal{D}}\left[\min_{y \neq y^\star(x)} f^\star(x,y) - f^\star(x,y^\star(x)) \leq \zeta\right] \leq \mathbb{P}_{x\sim\mathcal{D}}\left[1 - 2f^\star(x,y^\star(x)) \leq \zeta\right] \\ &\leq \mathbb{E}_x f^\star(x,y^\star(x)) \cdot \frac{2}{1-\zeta} \leq 4\mathrm{error}(h_{f^\star}). \end{aligned}$$

Now apply Lemma 19 with $\zeta = \min\{1/2, \sqrt{44K\Delta_i/\mathrm{error}(h_{f^\star})}\}$, and we obtain

$$r_i \leq \sqrt{44K\Delta_i\mathrm{error}(h_{f^\star})} + 264K\Delta_i.$$

Using the definition of Δ_i the final L_1 label complexity bound is

$$L_1 \leq 4\theta_0 n\cdot\mathrm{error}(h_{f^\star}) + \mathcal{O}\left(\theta_0\left(\sqrt{Kn\kappa\nu_n\cdot\mathrm{error}(h_{f^\star})} + K\kappa\nu_n\log(n)\right) + \log(1/\delta)\right). \qquad \blacksquare$$

9. Discussion

This paper presents a new active learning algorithm for cost-sensitive multiclass classification. The algorithm enjoys strong theoretical guarantees on running time, generalization error, and label complexity. The main algorithmic innovation is a new way to compute the maximum and minimum costs predicted by a regression function in the version space. We also design an online algorithm inspired by our theoretical analysis that outperforms passive baselines both in CSMC and structured prediction.

On a technical level, our algorithm uses a square loss oracle to search the version space and drive the query strategy. This contrasts with many recent results using argmax or 0/1-loss minimization oracles for information acquisition problems like contextual bandits (Agarwal et al., 2014). As these involve NP-hard optimizations in general, an intriguing question is whether we can use a square loss oracle for other information acquisition problems. We hope to answer this question in future work.

Acknowledgments

Part of this research was completed while TKH was at Microsoft Research and AK was at University of Massachusetts, Amherst. AK thanks Chicheng Zhang for insightful conversations. AK is supported in part by NSF Award IIS-1763618.

Appendix A. Proof of Theorem 15

In the proof, we mostly work with the empirical ℓ_1 covering number for $\mathcal{G}$. At the end, we translate to pseudo-dimension using a lemma of Haussler.

Definition 21 (Covering numbers) *Given class $\mathcal{G} \subset \mathcal{X} \to \mathbb{R}$, $\alpha > 0$, and sample $X = (x_1, \ldots, x_n) \in \mathcal{X}^n$, the covering number $\mathcal{N}_1(\alpha, \mathcal{H}, X)$ is the minimum cardinality of a set $V \subset \mathbb{R}^n$ such that for any $g \in \mathcal{G}$, there exists a $(v_1, \ldots, v_n) \in V$ with $\frac{1}{n}\sum_{i=1}^n |g(x_i) - v_i| \leq \alpha$.*

Lemma 22 (Covering number and Pseudo-dimension (Haussler, 1995)) *Given a hypothesis class $\mathcal{G} \subset \mathcal{X} \to \mathbb{R}$ with $\mathrm{Pdim}(\mathcal{G}) \leq d$, for any $X \in \mathcal{X}^n$ we have*

$$\mathcal{N}_1(\alpha, \mathcal{G}, X) \leq e(d+1)\left(\frac{2e}{\alpha}\right)^d.$$

Fixing i and y, and working toward (16) we seek to bound

$$\mathbb{P}\left(\sup_{g\in\mathcal{G}} \sum_{j=1}^{i} M_j(g;y) - \frac{3}{2}\mathbb{E}_j[M_j(g;y)] \geq \tau\right). \tag{22}$$

The bound on the other tail is similar. In this section, we sometimes treat the query rule as a function which maps an x to a query decision. We use the notation $\mathbf{Q}_j : \mathcal{X} \to \{0,1\}$ to denote the query function used after seeing the first $j-1$ examples. Thus, our query indicator Q_j is simply the instantiation $\mathbf{Q}_j(x_j)$. In this section, we work with an individual label y and omit the explicit dependence in all our arguments and notation. For notational convenience, we use $z_j = (x_j, c_j, Q_j)$ and with $g^\star(\cdot) = f^\star(\cdot; y)$, we define

$$\xi_j = g^\star(x_j) - c_j \quad \text{and} \quad \ell(g, x) = (g(x) - g^\star(x)). \tag{23}$$

Note that ξ_j is a centered random variable, independent of everything else. We now introduce some standard concepts from martingale theory for the proof of Theorem 15.

Definition 23 (Tangent sequence) *For a dependent sequence $z_1, \ldots, z_i$ we use $z'_1, \ldots, z'_i$ to denote a* tangent sequence, *where $z'_j|z_{1:j-1} \stackrel{d}{=} z_j|z_{1:j-1}$, and, conditioned on $z_1, \ldots, z_i$, the random variables $z'_1, \ldots, z'_i$ are independent.*

In fact, in our case we have $z'_j = (x'_j, c'_j, Q'_j)$ where $(x'_j, c'_j) \sim \mathcal{D}$ and $Q'_j : \mathcal{X} \to \{0,1\}$ is identical to Q_j. We use $M'_j(g)$ to denote the empirical excess square loss on sample z'_j. Note that we will continue to use our previous notation of $\mathbb{E}_j$ to denote conditioning on $z_1, \ldots, z_{j-1}$. We next introduce one more random process, and then proceed with the proof.

Definition 24 (Tree process) *A* tree process *$\mathcal{Q}$ is a binary tree of depth i where each node is decorated with a value from $\{0,1\}$. For a Rademacher sequence $\epsilon \in \{-1,1\}^i$ we use $\mathcal{Q}_i(\epsilon)$ to denote the value at the node reached when applying the actions $\epsilon_1, \ldots, \epsilon_{i-1}$ from the root, where $+1$ denotes left and -1 denotes right.*

The proof follows a fairly standard recipe for proving uniform convergence bounds, but has many steps that all require minor modifications from standard arguments. We compartmentalize each step in various lemmata:

1. In Lemma 25, we introduce a ghost sample and replace the conditional expectation $\mathbb{E}_j[\cdot]$ in (22) with an empirical term evaluated on an *tangent sequence*.
2. In Lemma 26, we perform symmetrization and introduce Rademacher random variables and the associated tree process.
3. In Lemma 27 we control the symmetrized process for finite $\mathcal{G}$.
4. In Lemma 28, we use the covering number to discretize $\mathcal{G}$.

We now state and prove the intermediate results.

Lemma 25 (Ghost sample) *Let $Z = (z_1, \dots, z_i)$ be the sequence of (x, c, Q) triples and let $Z' = (z'_1, \dots, z'_i)$ be a tangent sequence. Then for $\beta_0 \geq \beta_1 > 0$ if $\tau \geq \frac{4(1+\beta_1)^2}{(\beta_0-\beta_1)}$, then*

$$\mathbb{P}_Z\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i} M_j(g) - (1+2\beta_0)\mathbb{E}_j M_j(g) \geq \tau\right]$$
$$\leq 2\mathbb{P}_{Z,Z'}\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i} M_j(g) - M'_j(g) - 2\beta_1 Q_j(x'_j)\ell^2(g,x'_j) \geq \frac{\tau}{2}\right],$$
$$\mathbb{P}_Z\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i}(1-2\beta_0)\mathbb{E}_j M_j(g) - M_j(g) \geq \tau\right]$$
$$\leq 2\mathbb{P}_{Z,Z'}\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i} M'_j(g) - M_j(g) - 2\beta_1 Q_j(x'_j)\ell^2(g,x'_j) \geq \frac{\tau}{2}\right].$$

Proof We derive the first inequality, beginning with the right hand side and working toward a lower bound. The main idea is to condition on Z and just work with the randomness in Z'. To this end, let $\hat{g}$ achieve the supremum on the left hand side, and define the events

$$E = \left\{\sum_{j=1}^{i} M_j(\hat{g}) - (1+2\beta_0)\mathbb{E}_j M_j(\hat{g}) \geq \tau\right\}$$
$$E' = \left\{\sum_{i=1}^{n} M'_j(\hat{g}) + 2\beta_1 Q_j(x'_j)\ell^2(\hat{g},x'_j) - (1+2\beta_0)\mathbb{E}_j M'_j(\hat{g}) \leq \tau/2\right\}.$$

Starting from the right hand side, by adding and subtracting $(1+2\beta_0)\mathbb{E}_j M_j(\hat{g})$ we get

$$\mathbb{P}_{Z,Z'}\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i} M_j(g) - M'_j(g) - 2\beta_1 Q_j(x'_j)\ell^2(g,x'_j) \geq \tau/2\right]$$
$$\geq \mathbb{P}_{Z,Z'}\left[\sum_{j=1}^{i} M_j(\hat{g}) - M'_j(\hat{g}) - 2\beta_1 Q_j(x'_j)\ell^2(\hat{g},x'_j) \geq \tau/2\right]$$
$$\geq \mathbb{P}_{Z,Z'}\left[E\bigcap E'\right] = \mathbb{E}_Z\left[\mathbf{1}\{E\}\times\mathbb{P}_{Z'}[E'|Z]\right].$$

Since we have defined $\hat{g}$ to achieve the supremum, we know that

$$\mathbb{E}_Z\mathbf{1}\{E\} = \mathbb{P}_Z\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i} M_j(g) - (1+2\beta_0)\mathbb{E}_j M_j(g) \geq \tau\right].$$

which is precisely the left hand side of the desired inequality. Hence we need to bound the $\mathbb{P}_{Z'}[E'|Z]$ term. For this term, we note that

$$\mathbb{P}_{Z'}[E'|Z] = \mathbb{P}_{Z'}\left[\sum_{i=1}^{n} M'_j(\hat{g}) + 2\beta_1 Q_j(x'_j)\ell^2(\hat{g},x'_j) - (1+2\beta_0)\mathbb{E}_j M'_j(\hat{g}) \leq \tau/2 \mid Z\right]$$
$$\overset{(a)}{=} \mathbb{P}_{Z'}\left[\sum_{i=1}^{n} M'_j(\hat{g}) + 2\beta_1\mathbb{E}_j M'_j(\hat{g}) - (1+2\beta_0)\mathbb{E}_j M'_j(\hat{g}) \leq \tau/2 \mid Z\right]$$
$$= \mathbb{P}_{Z'}\left[\sum_{i=1}^{n}\left(M'_j(\hat{g}) - \mathbb{E}_j M'_j(\hat{g})\right) + 2\left(\beta_1-\beta_0\right)\mathbb{E}_j M'_j(\hat{g})\right) \leq \tau/2 \mid Z\right].$$

Here, (a) follows since $\mathbb{E}_j[M'_j(g) \mid Z] = Q_j(x'_j)\ell^2(g,x'_j)$ for any g by Lemma 17. Since we are conditioning on Z, $\hat{g}$ is also not a random function and the same equality holds when we take expectation over Z', even for $\hat{g}$. With this, we can now invoke Chebyshev's inequality:

$$\mathbb{P}_{Z'}[E'|Z] = 1 - \mathbb{P}_{Z'}\left[\sum_{j=1}^{i} M'_j(\hat{g}) + 2\beta_1 Q_j(x'_j)\ell^2(\hat{g},x'_j) - (1+2\beta_0)\mathbb{E}_j M'_j(\hat{g}) \geq \tau/2 \,\middle|\, Z\right]$$
$$\geq 1 - \frac{\mathrm{Var}\left[\sum_{j=1}^{i} M'_j(\hat{g}) + 2\beta_1 Q_j(x'_j)\ell(\hat{g},x'_j)\middle| Z\right]}{\left(\tau/2 + 2(\beta_0-\beta_1)\sum_i \mathbb{E}_j[Q_j(x'_j)\ell^2(\hat{g},x'_j)\middle| Z]\right)^2}.$$

Since we are working conditional on Z, we can leverage the independence of Z' (recall Definition 23) to bound the variance term.

$$\mathrm{Var}\left[\sum_{j=1}^{i} M'_j(g) + 2\beta_1 Q_j(x'_j)\ell^2(\hat{g},x'_j)\middle| Z\right]$$
$$\leq \sum_{j=1}^{i}\mathbb{E}_j\left[M'_j(\hat{g})^2 + (2\beta_1)^2 Q_j(x'_j)\ell^4(\hat{g},x'_j) + 4\beta_1 M'_j(\hat{g})Q_j(x'_j)\ell^2(\hat{g},x'_j)\right]$$
$$\leq 4(1+\beta_1)^2\sum_{j=1}^{i}\mathbb{E}_j Q_j(x'_j)\ell^2(\hat{g},x'_j).$$

Here we use that $\mathrm{Var}[X] \leq \mathbb{E}[X^2]$ and then we use that $\ell^2 \leq \ell$ since the loss is bounded in $[0,1]$ along with Lemma 17. Returning to the application of Chebyshev's inequality, if we expand the quadratic in the denominator and drop all but the cross term, we get the bound

$$\mathbb{P}'_Z[E'|Z] \geq 1 - \frac{2(1+\beta_1)^2}{\tau(\beta_0-\beta_1)} \geq 1/2,$$

where the last step uses the requirement on τ. This establishes the first inequality.

For the second inequality the steps are nearly identical. Let $\hat{g}$ achieve the supremum on the left hand side and define

$$E = \left\{ \sum_{j=1}^{i} (1-2\beta_0)\mathbb{E}_j M_j(\hat{g}) - M_j(\hat{g}) \geq \tau \right\}$$
$$E' = \left\{ \sum_{j=1}^{i} (1-2\beta_0)\mathbb{E}_j M'_j(\hat{g}) - M'_j(\hat{g}) + 2\beta_1 Q_j(x'_j)\ell^2(\hat{g}, x'_j) \leq \tau/2 \right\}.$$

Using the same argument, we can lower bound the right hand side by

$$\mathbb{P}\left[\sup_{g\in\mathcal{G}} \sum_{j=1}^{i} M'_j(g) - M_j(g) - 2\beta_1 Q_j(x'_j)\ell^2(g, x'_j) \geq \tau/2 \right] \geq \mathbb{E}_Z\left[\mathbf{1}\{E\} \times \mathbb{P}_{Z'}\left[E' \middle| Z\right]\right]$$

Applying Chebyshev's inequality yields the same expression as for the other tail. ■

Lemma 26 (Symmetrization) *Using the same notation as in Lemma 25, we have*

$$\mathbb{P}_{Z,Z'}\left[\sup_{g\in\mathcal{G}} \sum_{j=1}^{i} M_j(g) - M'_j(g) - 2\beta_1 Q_j(x'_j)\ell^2(g, x'_j) \geq \tau \right]$$
$$\leq 2\mathbb{E} \sup_{\mathcal{Q}} \mathbb{P}_\epsilon \left[\sup_{g\in\mathcal{G}} \sum_{j=1}^{i} \epsilon_j \mathcal{Q}_j(\epsilon)((1+\beta_1)\ell(g, x_j)^2 + 2\xi_j \ell(g, x_j)) - \beta_1 \mathcal{Q}_j(\epsilon)\ell(g, x_j)^2 \geq \tau/2 \right].$$

the same bound holds on the lower tail with $(1-\beta_1)$ *replacing* $(1+\beta_1)$.

Proof For this proof, we think of Q_j as a binary variable that is dependent on $z_1, \ldots, z_{j-1}$ and x_j. Similarly Q'_j depends on $z_1, \ldots, z_{j-1}$ and x'_j. Using this notation, and decomposing the square loss, we get

$$M_j(g) = Q_j\left[(g(x_j) - c_j)^2 - (g^\star(x_j) - c_j)^2\right] = Q_j\ell^2(g, x_j) + 2Q_j\xi_j\ell(g, x_j).$$

As such, we can write

$$M_j(g) - M'_j(g) - 2\beta_1 Q'_j\ell^2(g, x'_j) = \underbrace{(1+\beta_1)Q_j\ell^2(g, x_j) + 2Q_j\xi_j\ell(g, x_j)}_{\triangleq T_{1,j}} - \underbrace{\beta_1 Q_j\ell^2(g, x_j)}_{\triangleq T_{2,j}}$$
$$- \underbrace{(1+\beta_1)Q'_j\ell^2(g, x'_j) - 2Q'_j\xi'_j\ell(g, x'_j)}_{\triangleq T'_{1,j}} - \underbrace{\beta_1 Q'_j\ell^2(g, x'_j)}_{\triangleq T'_{2,j}}.$$

Here we have introduce the short forms $T_{1,j}, T_{2,j}$ and the primed version just to condense the derivations. Overall we must bound

$$\mathbb{P}\left[\sup_{g\in\mathcal{G}} \sum_{j=1}^{i} T_{1,j} - T_{2,j} - T'_{1,j} - T'_{2,j} \geq \tau \right] = \mathbb{E}\mathbf{1}\left\{ \sup_{g\in\mathcal{G}} \sum_{j=1}^{i} T_{1,j} - T_{2,j} - T'_{1,j} - T'_{2,j} \geq \tau \right\}.$$

Observe that in the final term $T_{1,i}, T'_{1,i}$ are random variables with identical conditional distribution, since there are no further dependencies and (x_i, ξ_i, Q_i) are identically distributed to (x'_i, ξ'_i, Q'_i). As such, we can symmetrize the i^{th} term by introducing the Rademacher random variable $\epsilon_i \in \{-1, +1\}$ to obtain

$$\mathbb{E}_{Z,Z'}\mathbb{E}_{\epsilon_i}\mathbf{1}\left\{ \sup_{g\in\mathcal{G}} \sum_{j=1}^{i-1} T_{1,j} - T_{2,j} + \epsilon_i(T_{1,i} - T'_{1,i}) - \sum_{j=1}^{i}(T_{2,j} + T'_{2,j}) \geq \tau \right\}$$
$$\leq \mathbb{E}_{Z,Z'} \sup_{Q_i, Q'_i} \mathbb{E}_{\epsilon_i}\mathbf{1}\left\{ \sup_{g\in\mathcal{G}} \sum_{j=1}^{i-1} T_{1,j} - T_{2,j} + \epsilon_i(T_{1,i} - T'_{1,i}) - \sum_{j=1}^{i}(T_{2,j} + T'_{2,j}) \geq \tau \right\}.$$

Here in the second step, we have replace the expectation over Q_i, Q'_i with supremum, which breaks the future dependencies for the $(i-1)^{\text{st}}$ term. Note that while we still write $\mathbb{E}_{Z,Z'}$, we are no longer taking expectation over Q_i, Q'_i here. The important point is that since x_j, ξ_j are all i.i.d., the only dependencies in the martingale are through Q_js and by taking supremum over Q_i, Q'_i, swapping the role of $T_{1,i-1}$ and $T'_{1,i-1}$ no longer has any future effects. Thus we can symmetrize the $(i-1)^{\text{st}}$ term. Continuing in this way, we get

$$\mathbb{E}\mathbb{E}_{\epsilon_{i-1}} \sup_{Q_i, Q'_i} \mathbb{E}_{\epsilon_i}\mathbf{1}\left\{ \sup_{g\in\mathcal{G}} \sum_{j=1}^{i-2} T_{1,j} - T_{2,j} + \sum_{j=i-1}^{i} \epsilon_j(T_{1,j} - T'_{1,j}) - \sum_{j=1}^{i}(T_{2,j} + T'_{2,j}) \geq \tau \right\}$$
$$\leq \mathbb{E}\left\langle \sup_{Q_j, Q'_j} \mathbb{E}_{\epsilon_j} \right\rangle_{j=1}^{i} \mathbf{1}\left\{ \sup_{g\in\mathcal{G}} \sum_{j=1}^{i} \epsilon_j(T_{1,j} - T'_{1,j}) - \sum_{j=1}^{i}(T_{2,j} + T'_{2,j}) \geq \tau \right\}.$$

Here in the final expression the outer expectation is just over the variables x_j, x'_j, ξ_j, ξ'_j and the bracket notation denotes interleaved supremum and expectation. Expanding the definitions of $T_{1,i}, T_{2,i}$, we currently have

$$\mathbb{E}\left\langle \sup_{Q_j, Q'_j} \mathbb{E}_{\epsilon_j} \right\rangle_{j=1}^{i} \mathbf{1}\left\{ \sup_{g\in\mathcal{G}} \sum_{j=1}^{i} \epsilon_j\left[(1+\beta_1)Q_j\ell^2(g, x_j) + 2Q_j\xi_j\ell(g, x_j)\right] - \sum_{j=1}^{i} \beta_1 Q_j\ell^2(g, x_j) \right.$$
$$\left. - \sum_{j=1}^{i} \epsilon_j\left[(1+\beta_1)Q'_j\ell^2(g, x'_j) + 2Q'_j\xi'_j\ell(g, x'_j)\right] - \sum_{j=1}^{i} \beta_1 Q'_j\ell^2(g, x'_j) \geq \tau \right\}.$$

Next we use the standard trick of splitting the supremum over g into a supremum over two functions g, g', where g' optimizes the primed terms. This provides an upper bound, but moreover if we replace τ with $\tau/2$ we can split the indicator into two and this becomes

$$2\mathbb{E}\left\langle \sup_{Q_j} \mathbb{E}_{\epsilon_j} \right\rangle_{j=1}^{i} \mathbf{1}\left\{ \sup_{g\in\mathcal{G}} \sum_{j=1}^{i} \epsilon_j Q_j\left[(1+\beta_1)\ell^2(g, x_j) + 2\xi_j\ell(g, x_j)\right] - \beta_1 Q_j\ell^2(g, x_j) \geq \tau/2 \right\}$$
$$= 2\mathbb{E} \sup_{\mathcal{Q}} \mathbb{P}_\epsilon\left[\sup_{g\in\mathcal{G}} \sum_{j=1}^{i} \epsilon_j \mathcal{Q}_j(\epsilon)\left[(1+\beta_1)\ell^2(g, x_j) + 2\xi_j\ell(g, x_j)\right] - \beta_1 \mathcal{Q}_j(\epsilon)\ell^2(g, x_j) \geq \tau/2 \right].$$

The tree process $\mathcal{Q}$ arises here because the interleaved supremum and expectation is equivalent to choosing a binary tree decorated with values from $\{0,1\}$ and then navigating the tree using the Rademacher random variables ϵ. The bound for the other tail is proved in the same way, except $(1+\beta_1)$ is replaced by $(1-\beta_1)$. ■

The next lemma is more standard, and follows from the union bound and the bound on the Rademacher moment generating function.

Lemma 27 (Finite class bound) *For any $x_{1:i}, \xi_{1:i}, \mathcal{Q}$, and for finite $\mathcal{G}$, we have*

$$\mathbb{P}_\epsilon\left[\max_{g\in\mathcal{G}}\sum_{j=1}^{i}\epsilon_j\mathcal{Q}_j(\epsilon)\left[(1+\beta_1)\ell^2(g,x_j)+2\xi_j\ell(g,x_j)\right]-\beta_1\mathcal{Q}_j(\epsilon)\ell^2(g,x_j)\geq\tau\right]$$
$$\leq|\mathcal{G}|\exp\left(\frac{-2\beta_1\tau}{(3+\beta_1)^2}\right).$$

The same bound applies for the lower tail.

Proof Applying the union bound and the Chernoff trick, we get that for any $\lambda>0$ the LHS is bounded by

$$\sum_g\exp(-\tau\lambda)\mathbb{E}_\epsilon\exp\left\{\lambda\left[\sum_{j=1}^{i}\underbrace{\epsilon_j\mathcal{Q}_j(\epsilon)\left[(1+\beta_1)\ell^2(g,x_j)+2\xi_j\ell(g,x_j)\right]-\beta_1\mathcal{Q}_i(\epsilon)\ell^2(g,x_j)}_{\triangleq T_{3,j}}\right]\right\}$$
$$=\sum_g\exp(-\tau\lambda)\mathbb{E}_{\epsilon_{1:i-1}}\exp\left\{\lambda\sum_{j=1}^{i-1}T_{3,j}\right\}\times\mathbb{E}_{\epsilon_i|\epsilon_{1:i-1}}\exp(\lambda T_{3,i}).$$

Let us examine the i^{th} term conditional on $\epsilon_{1:i-1}$. Conditionally on $\epsilon_{1:i-1}$, $\mathcal{Q}_i(\epsilon)$ is no longer random, so we can apply the MGF bound for Rademacher random variables to get

$$\mathbb{E}_{\epsilon_i|\epsilon_{1:i-1}}\exp\left\{\lambda\epsilon_i\mathcal{Q}_i(\epsilon)\left[(1+\beta_1)\ell^2(g,x_i)+2\xi_i\ell(g,x_i)\right]-\lambda\beta_1\mathcal{Q}_i(\epsilon)\ell^2(g,x_i)\right\}$$
$$\leq\exp\left\{\frac{\lambda^2}{2}\mathcal{Q}_i(\epsilon)\left[(1+\beta_1)\ell^2(g,x_i)+2\xi_i\ell(g,x_i)\right]^2-\lambda\beta_1\mathcal{Q}_i(\epsilon)\ell^2(g,x_i)\right\}$$
$$\leq\exp\left\{\lambda^2\mathcal{Q}_i(\epsilon)\frac{(3+\beta_1)^2}{2}\ell^2(g,x_i)-\lambda\beta_1\mathcal{Q}_i(\epsilon)\ell^2(g,x_i)\right\}\leq 1.$$

Here the first inequality is the standard MGF bound on Rademacher random variables $\mathbb{E}\exp(a\epsilon)\leq\exp(a^2/2)$. In the second line we expand the square and use that $\ell,\xi\in[-1,1]$ to upper bound all the terms. Finally, we use the choice of $\lambda=\frac{2\beta_1}{(3+\beta_1)^2}$. Repeating this argument from i down to 1, finishes the proof of the upper tail. The same argument applies for the lower tail, but we actually get $(3-\beta_1)^2$ in the denominator, which is of course upper bounded by $(3+\beta_1)^2$, since $\beta_1>0$. ■

Lemma 28 (Discretization) *Fix $x_1,\ldots,x_i$ and let $V\subset\mathbb{R}^i$ be a cover of $\mathcal{G}$ at scale α on points $x_1,\ldots,x_i$. Then for any $\mathcal{Q}$*

$$\mathbb{P}_\epsilon\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i}\epsilon_j\mathcal{Q}_j(\epsilon)\left[(1+\beta_1)\ell^2(g,x_j)+2\xi_j\ell(g,x_j)\right]-\beta_1\mathcal{Q}_j(\epsilon)\ell^2(g,x_j)\geq\tau\right]\leq$$
$$\mathbb{P}_\epsilon\left[\sup_{v\in V}\sum_{j=1}^{i}\epsilon_j\mathcal{Q}_j(\epsilon)\left[(1+\beta_1)\ell^2(v,x_j)+2\xi_j\ell(v,x_j)\right]-\beta_1\mathcal{Q}_j(\epsilon)\ell^2(v,x_j)\geq\tau-4i(1+\beta)\alpha\right].$$

The same bound holds for the lower tail with $1-\beta_1$. Here $\ell(v,x_j)=(v_j-g^\star(x_j))$.

Proof Observe first that if v is the covering element for g, then we are guaranteed that

$$\frac{1}{i}\sum_{j=1}^{i}|\ell(g,x_j)-\ell(v,x_j)|=\frac{1}{i}\sum_{j=1}^{i}|g(x_j)-v_j|\leq\alpha,$$
$$\frac{1}{i}\sum_{j=1}^{i}|\ell^2(g,x_j)-\ell^2(v,x_j)|=\frac{1}{i}\sum_{j=1}^{i}|(g(x_j)-v_j)(g(x_j)+v_j-2g^\star(x_j)|\leq 2\alpha,$$

since $g,v,g^\star\in[0,1]$. Thus, adding and subtracting the corresponding terms for v, and applying these bounds, we get a residual term of $i\alpha(2(1+\beta_1)+2+2\beta_1)=4i\alpha(1+\beta_1)$. ■

Proof of Theorem 15. Finally we can derive the deviation bound. We first do the upper tail, $M_j-\mathbb{E}_jM_j$. Set $\beta_0=1/4,\beta_1=1/8$ and apply Lemmas 25 and 26 to (22).

$$\mathbb{P}\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i}M_j(g)-\frac{3}{2}\mathbb{E}_jM_j(g)\geq\tau\right]$$
$$\leq 2\mathbb{P}\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i}M_j(g)-M_j'(g)-\frac{1}{4}Q_j(x_j')\ell^2(g,x_j')\geq\tau/2\right]$$
$$\leq 4\mathbb{E}\sup_{\mathcal{Q}}\mathbb{P}_\epsilon\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i}\epsilon_j\mathcal{Q}_j(\epsilon)\left(\frac{9}{8}\ell^2(g,x_j)+2\xi_j\ell(g,x_j)\right)-\frac{1}{8}\mathcal{Q}_j(\epsilon_i)\ell^2(g,x_j)\geq\frac{\tau}{4}\right].$$

Now let $V(X)$ be the cover for $\mathcal{G}$ at scale $\alpha=\frac{\tau}{32i(9/8)}=\frac{\tau}{36i}$, which makes $\tau/4-4i(1+\beta_1)\alpha=\frac{\tau}{8}$. Thus we get the bound

$$\leq 4\mathbb{E}_{X,\xi}\sup_{\mathcal{Q}}\mathbb{P}_\epsilon\left[\sup_{h\in\mathcal{H}(X)}\sum_{j=1}^{i}\epsilon_j\mathcal{Q}_j(\epsilon)\left(\frac{9}{8}\ell^2(h,x_j)+2\xi_j\ell(h,x_j)\right)-\frac{1}{8}\mathcal{Q}_j(\epsilon_i)\ell^2(h,x_j)\geq\frac{\tau}{8}\right]$$
$$\leq 4\mathbb{E}_X|V(X)|\exp\left(\frac{-2(1/8)(\tau/8)}{(3+1/8)^2}\right)=2\exp(-2\tau/625)\mathbb{E}_X\mathcal{N}\left(\frac{\tau}{36i},\mathcal{G},X\right).$$

This entire derivation requires that $\tau\geq\frac{4(9/8)^2}{(1/8)^2}=324$.

The lower tail bound is similar. By Lemmas 25 and 26, with $\beta_0 = 1/4$ and $\beta_1 = 1/8$,

$$\begin{aligned}
&\mathbb{P}\left[\sup_{g\in\mathcal{G}}\sum_{i=1}^{n}\frac{1}{2}\mathbb{E}_j M_j(g) - M_j(g) \geq \tau\right] \\
&\leq 2\mathbb{P}\left[\sup_{g\in\mathcal{G}}\sum_{i=1}^{n}\frac{1}{2}\mathbb{E}_j M_j(g) - M_j(g) - 2(1/8)Q_j(x'_j)\ell^2(g,x'_j) \geq \frac{\tau}{2}\right] \\
&\leq 4\mathbb{E}\sup_{\mathcal{Q}}\mathbb{P}_\epsilon\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i}\epsilon_j\mathcal{Q}_j(\epsilon)\left[\frac{7}{8}\ell^2(g,x_j) + 2\xi_j\ell(g,x_j)\right] - \frac{1}{8}\sum_{j=1}^{i}\mathcal{Q}_i(\epsilon)\ell^2(g,x_j) \geq \frac{\tau}{4}\right] \\
&\leq 4\mathbb{E}\sup_{\mathcal{Q}}\mathbb{P}_\epsilon\left[\sup_{g\in\mathcal{G}}\sum_{j=1}^{i}\epsilon_j\mathcal{Q}_j(\epsilon)\left[\frac{9}{8}\ell^2(g,x_j) + 2\xi_j\ell(g,x_j)\right] - \frac{1}{8}\sum_{j=1}^{i}\mathcal{Q}_i(\epsilon)\ell^2(g,x_j) \geq \frac{\tau}{4}\right].
\end{aligned}$$

This is the intermediate term we had for the upper tail, so we obtain the same bound.

To wrap up the proof, apply Haussler's Lemma 22, to bound the covering number

$$\mathbb{E}_X\mathcal{N}\left(\frac{\tau}{36i}, \mathcal{G}, X\right) \leq e(d+1)\left(\frac{72ie}{\tau}\right)^d.$$

Finally take a union bound over all pairs of starting and ending indices $i < i'$, all labels y, and both tails to get that the total failure probability is at most

$$8Ke(d+1)\exp\left(-2\tau/625\right)\sum_{i<i'\in[n]}\left(\frac{72e(i'-i)}{\tau}\right)^d.$$

The result now follows from standard approximations. Specifically we use the fact that we anyway require $\tau \geq 324$ to upper bound that $1/\tau^d$ term, use $(i'-i)^d \leq n^d$ and set the whole expression to be at most δ.

Appendix B. Multiplicative Weights

For completeness we prove Theorem 16 here. For this section only let $q^{(t)} \propto p^{(t)}$ be the distribution used by the algorithm at round t. If the program is feasible, then there exists a point that is also feasible against every distribution q. By contraposition, if on iteration t, the oracle reports infeasibility against $q^{(t)}$, then the original program must be infeasible.

Now suppose the oracle always finds v_t that is feasible against $q^{(t)}$. This implies

$$0 \leq \sum_{t=1}^{T}\sum_{i=1}^{n} q_i^{(t)}(b_i - \langle a_i, v_t\rangle).$$

Define $\ell_t(i) = \frac{b_i - \langle a_i, v_t\rangle}{\rho_i}$ which is in $[-1,1]$ by assumption. We compare this term to the corresponding term for a single constraint i. Using the standard potential-based analysis,

with $\Phi^{(t)} = \sum_i p_i^{(t)}$, we get

$$\begin{aligned}
\Phi^{(T+1)} = \sum_{i=1}^{m} p_i^{(T)}\left(1 - \eta\ell_T(i)\right) &\leq \Phi^{(T)}\exp\left(-\eta\sum_{i=1}^{m} q_i^{(T)}\ell_T(i)\right) \\
&\leq m\exp\left(-\eta\sum_{t=1}^{T}\sum_{i=1}^{m} q_i^{(t)}\ell_t(i)\right).
\end{aligned}$$

For any i, we also have

$$\Phi^{(T+1)} \geq \prod_{t=1}^{T}(1 - \eta\ell_t(i)).$$

Thus, taking taking logarithms and re-arranging we get

$$0 \leq \sum_{t=1}^{T}\sum_{i=1}^{m} q_i^{(t)}\ell_t(i) \leq \frac{\log(m)}{\eta} + \frac{1}{\eta}\sum_{t=1}^{T}\log\left(\frac{1}{1-\eta\ell_t(i)}\right) \leq \frac{\log m}{\eta} + \sum_{t=1}^{T}\ell_t(i) + \eta T.$$

Here we use standard approximations $\log(1/(1-x)) \leq x + x^2$ (which holds for $x \leq 1/2$) along with the fact that $|\ell_t(i)| \leq 1$. Using the definition of $\ell_t(i)$ we have hence proved that

$$\sum_{t=1}^{T}\langle a_i, v_t\rangle \leq Tb_i + \frac{\rho_i\log m}{\eta} + \rho_i\eta T.$$

Now with our choice of $\eta = \sqrt{\log(m)/T}$ we get the desired bound. If $\eta \geq 1/2$ then the result is trivial by the boundedness guarantee on the oracle.

Appendix C. Proofs of Lemmata

Proof [Proof of Lemma 14] By the definitions,

$$\begin{aligned}
\widehat{R}(g') + w'(g'(x)-c)^2 = \widehat{R}(g', w', c) &\leq \widehat{R}(g) + w'(g(x)-c)^2 \\
&= \widehat{R}(g) + w(g(x)-c)^2 + (w'-w)(g(x)-c)^2 \\
&\leq \widehat{R}(g') + w(g'(x)-c)^2 + (w'-w)(g(x)-c)^2.
\end{aligned}$$

Rearranging shows that $(w'-w)(g'(x)-c)^2 \leq (w'-w)(g(x)-c)^2$. Since $w' \geq w$, we have $(g'(x)-c)^2 \leq (g(x)-c)^2$, which is the second claim. For the first, the definition of g gives

$$\widehat{R}(g) + w(g(x)-c)^2 \leq \widehat{R}(g') + w(g'(x)-c)^2.$$

Rearranging this inequality gives, $\widehat{R}(g') - \widehat{R}(g) \geq w((g(x)-c)^2 - (g'(x)-c)^2) \geq 0$, which yields the result. ∎

Proof [Proof of Lemma 17] We take expectation of M_j over the cost conditioned on a fixed example $x_j = x$ and a fixed query outcome $Q_j(y)$:

$$\begin{aligned}\mathbb{E}[M_j \mid x_j = x, Q_j(y)] &= Q_j(y) \times \mathbb{E}_c[g(x)^2 - f^\star(x;y)^2 - 2c(y)(g(x) - f^\star(x;y)) \mid x_j = x] \\ &= Q_j(y)\big(g(x)^2 - f^\star(x;y)^2 - 2f^\star(x;y)(g(x) - f^\star(x;y))\big) \\ &= Q_j(y)(g(x) - f^\star(x;y))^2.\end{aligned}$$

The second equality is by Assumption 1, which implies $\mathbb{E}[c(y) \mid x_j = x] = f^\star(x;y)$. Taking expectation over x_j and $Q_j(y)$, we have

$$\mathbb{E}_j[M_j] \;=\; \mathbb{E}_j\left[Q_j(y)(g(x_j) - f^\star(x_j;y))^2\right].$$

For the variance:

$$\begin{aligned}\operatorname*{Var}_j[M_j] \le \mathbb{E}_j[M_j^2] &= \mathbb{E}_j\left[Q_j(y)(g(x_j) - f^\star(x_j;y))^2(g(x_j) + f^\star(x_j;y) - 2c(y))^2\right] \\ &\le 4 \cdot \mathbb{E}_j\left[Q_j(y)(g(x_j) - f^\star(x_j;y))^2\right] = 4\mathbb{E}_j[M_j].\end{aligned}$$

This concludes the proof. ■

Proof [Proof of Lemma 18] Fix some $f \in \mathcal{F}_i$, and let $\hat{y} = h_f(x)$ and $y^\star = h_{f^\star}(x)$ for shorthand, but note that both depend on x. Define

$$S_\zeta(x) = \mathbb{1}\left(f^\star(x,\hat{y}) \le f^\star(x,y^\star) + \zeta\right), \qquad S'_\zeta(x) = \mathbb{1}\left(\min_{y \neq y^\star} f^\star(x,y) \le f^\star(x,y^\star) + \zeta\right).$$

Observe that for fixed ζ, $S_\zeta(x)\mathbb{1}(\hat{y} \neq y^\star) \le S'_\zeta(x)$ for all x. We can also majorize the complementary indicator to obtain the inequality

$$S^C_\zeta(x) \le \frac{f^\star(x,\hat{y}) - f^\star(x,y^\star)}{\zeta}.$$

We begin with the definition of realizability, which gives

$$\begin{aligned}\mathbb{E}_{x,c}[c(h_f(x)) - c(h_{f^\star}(x)] &= \mathbb{E}_x\left[(f^\star(x,\hat{y}) - f^\star(x,y^\star))\,\mathbb{1}(\hat{y} \neq y^\star)\right] \\ &= \mathbb{E}_x\left[\left(S_\zeta(x) + S^C_\zeta(x)\right)(f^\star(x,\hat{y}) - f^\star(x,y^\star))\,\mathbb{1}(\hat{y} \neq y^\star)\right] \\ &\le \zeta\mathbb{E}_x S'_\zeta(x) + \mathbb{E}_i\left[S^C_\zeta(x)\mathbb{1}(\hat{y} \neq y^\star)(f^\star(x,\hat{y}) - f^\star(x,y^\star))\right].\end{aligned}$$

The first term here is exactly the ζP_ζ term in the bound. We now focus on the second term, which depends on our query rule. For this we must consider three cases.

Case 1. If both $\hat{y}$ and $y^\star$ are not queried, then it must be the case that both have small cost ranges. This follows since $f \in \mathcal{F}_i$ and $h_f(x) = \hat{y}$ so $y^\star$ does not dominate $\hat{y}$. Moreover, since the cost ranges are small on both $\hat{y}$ and $y^\star$ and since we know that $f^\star$ is well separated under event $S^C_\zeta(x)$, the relationship between ζ and ψ_i governs whether we make a mistake or not. Specifically, we get that $S^C_\zeta(x)\mathbb{1}(\hat{y} \neq y^\star)\,\mathbb{1}(\text{no query}) \le \mathbb{1}(\zeta \le 2\psi_i)$ at round i. In other words, if we do not query and the separation is big but we make a mistake, then it must mean that the cost range threshold ψ_i is also big.

Using this argument, we can bound the second term as,

$$\begin{aligned}&\mathbb{E}_i\left[S^C_\zeta(x)\mathbb{1}(\hat{y} \neq y^\star)(1 - Q_i(\hat{y}))(1 - Q_i(y^\star))(f^\star(x,\hat{y}) - f^\star(x,y^\star))\right] \\ &\le \mathbb{E}_i\left[S^C_\zeta(x)\mathbb{1}(\hat{y} \neq y^\star)(1 - Q_i(\hat{y}))(1 - Q_i(y^\star))2\psi_i\right] \\ &\le \mathbb{E}_i\left[\mathbb{1}(\zeta \le 2\psi_i)\,2\psi_i\right] = \mathbb{1}(\zeta \le 2\psi_i)\,2\psi_i.\end{aligned}$$

Case 2. If both $\hat{y}$ and $y^\star$ are queried, we can relate the second term to the square loss,

$$\begin{aligned}&\mathbb{E}_i\left[S^C_\zeta(x)Q_i(\hat{y})Q_i(y^\star)(f^\star(x,\hat{y}) - f^\star(x,y^\star))\right] \\ &\le \frac{1}{\zeta}\mathbb{E}_i\left[Q_i(\hat{y})Q_i(y^\star)(f^\star(x,\hat{y}) - f^\star(x,y^\star))^2\right] \\ &\le \frac{1}{\zeta}\mathbb{E}_i\left[Q_i(\hat{y})Q_i(y^\star)(f^\star(x,\hat{y}) - f(x,\hat{y}) + f(x,y^\star) - f^\star(x,y^\star))^2\right] \\ &\le \frac{2}{\zeta}\mathbb{E}_i\left[Q_i(\hat{y})(f^\star(x,\hat{y}) - f(x,\hat{y}))^2 + Q_i(y^\star)(f(x,y^\star) - f^\star(x,y^\star))^2\right] \\ &\le \frac{2}{\zeta}\sum_y \mathbb{E}_i\left[Q_i(y)(f^\star(x,y) - f(x,y))^2\right] = \frac{2}{\zeta}\sum_y \mathbb{E}_i\left[M_i(f;y)\right].\end{aligned}$$

Passing from the second to third line here is justified by the fact that $f^\star(x,\hat{y}) \ge f^\star(x,y^\star)$ and $f(x,\hat{y}) \le f(x,y^\star)$ so we added two non-negative quantities together. The last step uses Lemma 17. While not written, we also use the event $\mathbb{1}(\hat{y} \neq y^\star)$ to save a factor of 2.

Case 3. The last case is if one label is queried and the other is not. Both cases here are analogous, so we do the derivation for when $y(x)$ is queried but $y^\star(x)$ is not. Since in this case, $y^\star(x)$ is not dominated ($h_f(x)$ is never dominated provided $f \in \mathcal{F}_i$), we know that the cost range for $y^\star(x)$ must be small. Using this fact, and essentially the same argument as in case 2, we get

$$\begin{aligned}&\mathbb{E}_i\left[S^C_\zeta(x)Q_i(\hat{y})(1 - Q_i(y^\star))(f^\star(x,\hat{y}) - f^\star(x,y^\star))\right] \\ &\le \frac{1}{\zeta}\mathbb{E}_i\left[Q_i(\hat{y})(1 - Q_i(y^\star))(f^\star(x,\hat{y}) - f^\star(x,y^\star))^2\right] \\ &\le \frac{2}{\zeta}\mathbb{E}_i\left[Q_i(\hat{y})(f^\star(x,\hat{y}) - f(x,\hat{y}))^2 + (1 - Q_i(y^\star))(f(x,y^\star) - f^\star(x,y^\star))^2\right] \\ &\le \frac{2\psi_i^2}{\zeta} + \frac{2}{\zeta}\mathbb{E}_i\left[Q_i(\hat{y})(f^\star(x,\hat{y}) - f(x,\hat{y}))^2\right] \le \frac{2\psi_i^2}{\zeta} + \frac{2}{\zeta}\sum_y \mathbb{E}_i\left[M_i(f;y)\right].\end{aligned}$$

We also obtain this term for the other case where $y^\star$ is queried but $\hat{y}$ is not.

To summarize, adding up the contributions from these cases (which is an over-estimate since at most one case can occur and all are non-negative), we get

$$\mathbb{E}_{x,c}[c(h_f(x)) - c(h_{f^\star}(x)] \le \zeta P_\zeta + \mathbb{1}(\zeta \le 2\psi_i)\,2\psi_i + \frac{4\psi_i^2}{\zeta} + \frac{6}{\zeta}\sum_y \mathbb{E}_i\left[M_i(f;y)\right].$$

This bound holds for any ζ, so it holds for the minimum. ■

Proof [Proof of Lemma 19] The proof here is an easier version of the generalization bound proof for f_i. First, condition on the high probability event in Theorem 15, under which we already showed that $f^\star \in \mathcal{F}_i$ for all $i \in [n]$. Now fix some $f \in \mathcal{F}_i$. Since by the monotonicity property defining the sets $\mathcal{G}_i$, we must have $f \in \mathcal{F}_j$ for all $1 \le j \le i$, we can immediately apply Lemma 18 on all rounds to bound the cost sensitive regret by

$$\frac{1}{i-1}\sum_{j=1}^{i-1}\min_{\zeta>0}\left\{\zeta P_\zeta + \mathbf{1}\{\zeta \le 2\psi_j\}2\psi_j + \frac{4\psi_j^2}{\zeta} + \frac{6}{\zeta}\sum_y \mathbb{E}_j[M_j(f;y)]\right\}.$$

As in the generalization bound, the first term contributes ζP_ζ, the second is at most $\frac{12}{\zeta}$ and the third is at most $4/\zeta(1+\log(i-1))$. The fourth term is slightly different. We still apply (17) to obtain

$$\begin{aligned}\frac{1}{i-1}\sum_{j=1}^{i-1}\mathbb{E}_j[M_j(f;y)] &\le \frac{2}{i-1}\sum_{j=1}^{i-1}M_j(f;y) + \frac{2\nu_n}{i-1}\\ &= 2\left(\widehat{R}_i(f;y) - \widehat{R}_i(g_{i,y};y) + \widehat{R}_i(g_{i,y};y) - \widehat{R}_i(f^\star;y)\right) + \frac{2\nu_n}{i-1}\\ &\le 2\Delta_i + \frac{2\nu_n}{i-1} = \frac{2(\kappa+1)\nu_n}{i-1}.\end{aligned}$$

We use this bound for each label. Putting terms together, the cost sensitive regret is

$$\min_{\zeta>0}\left\{\zeta P_\zeta + \frac{12}{\zeta} + \frac{4(1+\log(i-1))}{\zeta} + \frac{12K(\kappa+1)\nu_n}{\zeta(i-1)}\right\} \le \min_{\zeta>0}\left\{\zeta P_\zeta + \frac{44\kappa K\nu_n}{\zeta(i-1)}\right\}.$$

This proves containment, since this upper bounds the cost sensitive regret of every $f \in \mathcal{F}_i$. ■

Proof [Proof of Lemma 20] The first claim is straightforward, since $\mathcal{F}_i \subset \mathcal{F}_{\text{csr}}(r_i)$ and since we set the tolerance parameter in the calls to MaxCost and MinCost to $\psi_i/4$. Specifically,

$$\widehat{\gamma}(x_i,y) \le \gamma(x_i,y,\mathcal{F}_i) + \psi_i/2 \le \gamma(x_i,y,\mathcal{F}_{\text{csr}}(r_i)) + \psi_i/2.$$

For the second claim, suppose $y \neq y_i^\star$. Then

$$\begin{aligned}y \in Y_i &\Rightarrow \widehat{c_-}(x_i,y) \le \widehat{c_+}(x_i,\bar{y}_i)\\ &\Rightarrow \widehat{c_-}(x_i,y) \le \widehat{c_+}(x_i,y_i^\star)\\ &\Rightarrow c_-(x_i,y,\mathcal{F}_{\text{csr}}(r_i)) \le c_+(x_i,y_i^\star,\mathcal{F}_{\text{csr}}(r_i)) + \psi_i/2\\ &\Rightarrow f^\star(x_i;y) - \gamma(x_i,y,\mathcal{F}_{\text{csr}}(r_i)) \le f^\star(x_i;y_i^\star) + \gamma(x_i,y_i^\star,\mathcal{F}_{\text{csr}}(r_i)) + \psi_i/2.\end{aligned}$$

This argument uses the tolerance setting $\psi_i/4$, Lemma 19 to translate between the version space and the cost-sensitive regret ball, and finally the fact that $f^\star \in \mathcal{F}_{\text{csr}}(r_i)$ since it has zero cost-sensitive regret. This latter fact lets us lower (upper) bound the minimum (maximum) cost by $f^\star$ prediction minus (plus) the cost range.

For $y_i^\star$ we need to consider two cases. First assume $y_i^\star = \bar{y}_i$. Since by assumption $|Y_i| > 1$, it must be the case that $\widehat{c_-}(x_i,\tilde{y}_i) \le \widehat{c_+}(x_i,y_i^\star)$ at which point the above derivation produces the desired implication. On the other hand, if $y_i^\star \neq \bar{y}_i$ then $\widehat{c_+}(x_i,\bar{y}_i) \le \widehat{c_+}(x_i,y_i^\star)$, but this also implies that $\widehat{c_-}(x_i,\tilde{y}_i) \le \widehat{c_+}(x_i,y_i^\star)$, since minimum costs are always smaller than maximum costs, and $\bar{y}_i$ is included in the search defining $\tilde{y}_i$. ■

References

Alekh Agarwal. Selective sampling algorithms for cost-sensitive multiclass prediction. In *International Conference on Machine Learning*, 2013.

Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert E. Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In *International Conference on Machine Learning*, 2014.

Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 2012.

Maria Florina Balcan and Philip M. Long. Active and passive learning of linear separators under log-concave distributions. In *Conference on Learning Theory*, 2013.

Maria Florina Balcan, Alina Beygelzimer, and John Langford. Agnostic active learning. In *International Conference on Machine Learning*, 2006.

Maria Florina Balcan, Andrei Broder, and Tong Zhang. Margin based active learning. In *Conference on Learning Theory*, 2007.

Alina Beygelzimer, Sanjoy Dasgupta, and John Langford. Importance weighted active learning. In *International Conference on Machine Learning*, 2009.

Alina Beygelzimer, Daniel Hsu, John Langford, and Tong Zhang. Agnostic active learning without constraints. In *Advances in Neural Information Processing Systems*, 2010.

Alexandra Carpentier, Andrea Locatelli, and Samory Kpotufe. Adaptivity to noise parameters in nonparametric active learning. In *Conference on Learning Theory*, 2017.

R.M. Castro and R.D. Nowak. Minimax bounds for active learning. *Transaction on Information Theory*, 2008.

Rui Castro, Rebecca Willett, and Robert D. Nowak. Faster rates in regression via active learning. In *Advances in Neural Information Processing Systems*, 2005.

Giovanni Cavallanti, Nicolo Cesa-Bianchi, and Claudio Gentile. Learning noisy linear classifiers via adaptive and selective sampling. *Machine Learning*, 2011.

Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. Learning to search better than your teacher. In *International Conference on Machine Learning*, 2015.

Sanjoy Dasgupta, Daniel Hsu, and Claire Monteleoni. A general agnostic active learning algorithm. In *Advances in Neural Information Processing Systems*, 2007.

Hal Daumé III, John Langford, and Daniel Marcu. Search-based structured prediction. *Machine Learning*, 2009.

Ofer Dekel, Claudio Gentile, and Karthik Sridharan. Robust selective sampling from single and multiple teachers. In *Conference on Learning Theory*, 2010.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *Conference on Learning Theory*, 2010.

Dylan Foster, Alekh Agarwal, Miroslav Dudik, Haipeng Luo, and Robert Schapire. Practical contextual bandits with regression oracles. In *International Conference on Machine Learning*, 2018.

Steve Hanneke. Theory of disagreement-based active learning. *Foundations and Trends in Machine Learning*, 2014.

Steve Hanneke and Liu Yang. Surrogate losses in passive and active learning. *arXiv:1207.3772*, 2012.

Steve Hanneke and Liu Yang. Minimax analysis of active learning. *Journal of Machine Learning Research*, 2015.

David Haussler. Sphere packing numbers for subsets of the Boolean n-cube with bounded Vapnik-Chervonenkis dimension. *Journal of Combinatorial Theory, Series A*, 1995.

Daniel Hsu. *Algorithms for Active Learning*. PhD thesis, University of California at San Diego, 2010.

Tzu-Kuo Huang, Alekh Agarwal, Daniel Hsu, John Langford, and Robert E. Schapire. Efficient and parsimonious agnostic active learning. In *Advances in Neural Information Processing Systems*, 2015.

Nikos Karampatziakis and John Langford. Online importance weight aware updates. In *Uncertainty in Artificial Intelligence*, 2011.

Akshay Krishnamurthy, Alekh Agarwal, Tzu-Kuo Huang, Hal Daumé, III, and John Langford. Active learning for cost-sensitive classification. In *International Conference on Machine Learning*, 2017.

John Langford and Alina Beygelzimer. Sensitive error correcting output codes. In *Conference on Learning Theory*, 2005.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 2004. Data available at `http://www.jmlr.org/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm`.

Tengyuan Liang, Alexander Rakhlin, and Karthik Sridharan. Learning with square loss: Localization through offset rademacher complexity. In *Conference on Learning Theory*, 2015.

Enno Mammen and Alexandre B. Tsybakov. Smooth discrimination analysis. *The Annals of Statistics*, 1999.

Pascal Massart and Élodie Nédélec. Risk bounds for statistical learning. *The Annals of Statistics*, 2006.

Stanislav Minsker. Plug-in approach to active learning. *Journal of Machine Learning Research*, 2012.

Francesco Orabona and Nicolo Cesa-Bianchi. Better algorithms for selective sampling. In *International Conference on Machine Learning*, 2011.

Serge A. Plotkin, David B. Shmoys, and Éva Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 1995.

Alexander Rakhlin and Karthik Sridharan. On equivalence of martingale tail bounds and deterministic regret inequalities. In *Proceedings of the 2017 Conference on Learning Theory*, 2017.

Alexander Rakhlin, Karthik Sridharan, Alexandre B Tsybakov, et al. Empirical entropy, minimax regret and minimax risk. *Bernoulli*, 2017.

Stephane Ross and J. Andrew Bagnell. Reinforcement and imitation learning via interactive no-regret learning. *arXiv:1406.5979*, 2014.

Stephane Ross, Paul Mineiro, and John Langford. Normalized online learning. In *Uncertainty in Artificial Intelligence*, 2013.

Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2012.

Tianlian Shi, Jacob Steinhardt, and Percy Liang. Learning where to sample in structured prediction. In *Artificial Intelligence and Statistics*, 2015.

Carlos N. Silla Jr. and Alex A. Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 2011.

Wen Sun, Arun Venkatraman, Geoffrey J. Gordon, Byron Boots, and J. Andrew Bagnell. Deeply AggreVaTeD: Differentiable imitation learning for sequential prediction. In *International Conference on Machine Learning*, 2017.

Vasilis Syrgkanis, Akshay Krishnamurthy, and Robert E. Schapire. Efficient algorithms for adversarial contextual learning. In *International Conference on Machine Learning*, 2016.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Computer Vision and Pattern Recognition*, 2015.

Alexandre B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics*, 2004.

Chicheng Zhang and Kamalika Chaudhuri. Beyond disagreement-based agnostic active learning. In *Advances in Neural Information Processing Systems*, 2014.

Journal of Machine Learning Research 20 (2019) 1-38 Submitted 11/17; Revised 3/19; Published 4/19

Proximal Distance Algorithms: Theory and Practice

Kevin L. Keys KEVIN.KEYS@UCSF.EDU
Department of Medicine
University of California, San Francisco
CA 94158, USA

Hua Zhou HUAZHOU@UCLA.EDU
Department of Biostatistics
University of California, Los Angeles
CA 90095-1772, USA

Kenneth Lange KLANGE@UCLA.EDU
Departments of Biomathematics, Human Genetics, and Statistics
University of California, Los Angeles
CA 90095-1766, USA

Editor: Edo Airoldi

Abstract

Proximal distance algorithms combine the classical penalty method of constrained minimization with distance majorization. If $f(\boldsymbol{x})$ is the loss function, and C is the constraint set in a constrained minimization problem, then the proximal distance principle mandates minimizing the penalized loss $f(\boldsymbol{x}) + \frac{\rho}{2}\operatorname{dist}(\boldsymbol{x}, C)^2$ and following the solution $\boldsymbol{x}_\rho$ to its limit as ρ tends to ∞. At each iteration the squared Euclidean distance $\operatorname{dist}(\boldsymbol{x}, C)^2$ is majorized by the spherical quadratic $\|\boldsymbol{x} - P_C(\boldsymbol{x}_k)\|^2$, where $P_C(\boldsymbol{x}_k)$ denotes the projection of the current iterate $\boldsymbol{x}_k$ onto C. The minimum of the surrogate function $f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{x} - P_C(\boldsymbol{x}_k)\|^2$ is given by the proximal map $\operatorname{prox}_{\rho^{-1}f}[P_C(\boldsymbol{x}_k)]$. The next iterate $\boldsymbol{x}_{k+1}$ automatically decreases the original penalized loss for fixed ρ. Since many explicit projections and proximal maps are known, it is straightforward to derive and implement novel optimization algorithms in this setting. These algorithms can take hundreds if not thousands of iterations to converge, but the simple nature of each iteration makes proximal distance algorithms competitive with traditional algorithms. For convex problems, proximal distance algorithms reduce to proximal gradient algorithms and therefore enjoy well understood convergence properties. For nonconvex problems, one can attack convergence by invoking Zangwill's theorem. Our numerical examples demonstrate the utility of proximal distance algorithms in various high-dimensional settings, including a) linear programming, b) constrained least squares, c) projection to the closest kinship matrix, d) projection onto a second-order cone constraint, e) calculation of Horn's copositive matrix index, f) linear complementarity programming, and g) sparse principal components analysis. The proximal distance algorithm in each case is competitive or superior in speed to traditional methods such as the interior point method and the alternating direction method of multipliers (ADMM). Source code for the numerical examples can be found at `https://github.com/klkeys/proxdist`.

Keywords: constrained optimization, EM algorithm, majorization, projection, proximal operator

©2019 Kevin L. Keys, Hua Zhou, and Kenneth Lange.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v20/17-687.html`.

1. Introduction

The solution of constrained optimization problems is part science and part art. As mathematical scientists explore the largely uncharted territory of high-dimensional nonconvex problems, it is imperative to consider new methods. The current paper studies a class of optimization algorithms that combine Courant's penalty method of optimization (Beltrami, 1970; Courant, 1943) with the notion of a proximal operator (Bauschke and Combettes, 2011; Moreau, 1962; Parikh and Boyd, 2013). The classical penalty method turns constrained minimization of a function $f(\boldsymbol{x})$ over a closed set C into unconstrained minimization. The general idea is to seek the minimum point of a penalized version $f(\boldsymbol{x}) + \rho q(\boldsymbol{x})$ of $f(\boldsymbol{x})$, where the penalty $q(\boldsymbol{x})$ is nonnegative and vanishes precisely on C. If one follows the solution vector $\boldsymbol{x}_\rho$ as ρ tends to ∞, then in the limit one recovers the constrained solution. The penalties of choice in the current paper are squared Euclidean distances $\operatorname{dist}(\boldsymbol{x}, C)^2 = \inf_{\boldsymbol{y} \in C} \|\boldsymbol{x} - \boldsymbol{y}\|^2$.

The formula

$$\operatorname{prox}_f(\boldsymbol{y}) \;=\; \operatorname{argmin}_{\boldsymbol{x}} \left[f(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2 \right] \tag{1}$$

defines the proximal map of a function $f(\boldsymbol{x})$. Here $\|\cdot\|$ is again the standard Euclidean norm, and $f(\boldsymbol{x})$ is typically assumed to be closed and convex. Projection onto a closed convex set C is realized by choosing $f(\boldsymbol{x})$ to be the $0/\infty$ indicator $\delta_C(\boldsymbol{x})$ of C. It is possible to drop the convexity assumption if $f(\boldsymbol{x})$ is nonnegative or coercive. In so doing, $\operatorname{prox}_f(\boldsymbol{y})$ may become multi-valued. For example, the minimum distance from a nonconvex set to an exterior point may be attained at multiple boundary points. The point $\boldsymbol{x}$ in the definition (1) can be restricted to a subset S of Euclidean space by replacing $f(\boldsymbol{x})$ by $f(\boldsymbol{x}) + \delta_S(\boldsymbol{x})$, where $\delta_S(\boldsymbol{x})$ is the indicator of S.

One of the virtues of exploiting proximal operators is that they have been thoroughly investigated. For a large number of functions $f(\boldsymbol{x})$, the map $\operatorname{prox}_{cf}(\boldsymbol{y})$ for $c > 0$ is either given by an exact formula or calculable by an efficient algorithm. The known formulas tend to be highly accurate. This is a plus because the classical penalty method suffers from ill conditioning for large values of the penalty constant. Although the penalty method seldom delivers exquisitely accurate solutions, moderate accuracy suffices for many problems.

There are ample precedents in the optimization literature for the proximal distance principle. Proximal gradient algorithms have been employed for many years in many contexts, including projected Landweber, alternating projection onto the intersection of two or more closed convex sets, the alternating-direction method of multipliers (ADMM), and fast iterative shrinkage thresholding algorithms (FISTA) (Beck and Teboulle, 2009; Combettes and Pesquet, 2011; Landweber, 1951). Applications of distance majorization are more recent (Chi et al., 2014; Lange and Keys, 2014; Xu et al., 2017). The overall strategy consists of replacing the distance penalty $\operatorname{dist}(\boldsymbol{x}, C)^2$ by the spherical quadratic $\|\boldsymbol{x} - \boldsymbol{y}_k\|^2$, where $\boldsymbol{y}_k$ is the projection of the kth iterate $\boldsymbol{x}_k$ onto C. To form the next iterate, one then sets

$$\boldsymbol{x}_{k+1} \;=\; \operatorname{prox}_{\rho^{-1}f}(\boldsymbol{y}_k) \quad \text{with} \quad \boldsymbol{y}_k \;=\; P_C(\boldsymbol{x}_k).$$

The MM (majorization-minimization) principle guarantees that $\boldsymbol{x}_{k+1}$ decreases the penalized loss. We call the combination of Courant's penalty method with distance majorization the *proximal distance principle*. Algorithms constructed according to the principle are *proximal distance algorithms*.

The current paper extends and deepens our previous preliminary treatments of the proximal distance principle. Details of implementation such as Nesterov acceleration matter in performance. We have found that squared distance penalties tend to work better than exact penalties. In the presence of convexity, it is now clear that every proximal distance algorithm reduces to a proximal gradient algorithm. Hence, convergence analysis can appeal to a venerable body of convex theory. This does not imply that the proximal distance algorithm is limited to convex problems. In fact, its most important applications may well be to nonconvex problems. A major focus of this paper is on practical exploration of the proximal distance algorithm.

In addition to reviewing the literature, the current paper presents some fresh ideas. Among the innovations are: a) recasting proximal distance algorithms with convex losses as concave-convex programs, b) providing new perspectives on convergence for both convex and nonconvex proximal distance algorithms, c) demonstrating the virtue of folding constraints into the domain of the loss, and d) treating in detail seven interesting examples. It is noteworthy that some our new convergence theory is pertinent to more general MM algorithms.

It is our sincere hope to enlist other mathematical scientists in expanding and clarifying this promising line of research. The reviewers of the current paper have correctly pointed out that we do not rigorously justify our choices of the penalty constant sequence ρ_k. The recent paper by Li et al. (2017) may be a logical place to start in filling this theoretical gap. They deal with the problem of minimizing $f(\boldsymbol{x})$ subject to $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ through the quadratic penalized objective $f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2$. For the right choices of the penalty sequence ρ_k, their proximal gradient algorithm achieves a $O(k^{-1})$ rate of convergence for $f(\boldsymbol{x})$ strongly convex. As a substitute, we explore the classical problem of determining how accurately the solution $\boldsymbol{y}_\rho$ of the problem $\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \frac{\rho}{2} q(\boldsymbol{x})^2$ approximates the solution $\boldsymbol{y}$ of the constrained problem $\min_{\boldsymbol{x} \in C} f(\boldsymbol{x})$. Polyak (1971) demonstrates that $f(\boldsymbol{y}) - f(\boldsymbol{y}_\rho) = O(\rho^{-1})$ for a penalty function $q(\boldsymbol{x})$ that vanishes precisely on C. Polyak's proof relies on strong differentiability assumptions. Our proof for the case $q(\boldsymbol{x}) = \operatorname{dist}(\boldsymbol{x}, C)$ relies on convexity and is much simpler.

As a preview, let us outline the remainder of our paper. Section 2 briefly sketches the underlying MM principle. We then show how to construct proximal distance algorithms from the MM principle and distance majorization. The section concludes with the derivation of a few broad categories proximal distance algorithms. Section 3 covers convergence theory for convex problems, while Section 4 provides a more general treatment of convergence for nonconvex problems. To avoid breaking the flow of our exposition, all proofs are relegated to the Appendix. Section 5 discusses our numerical experiments on various convex and nonconvex problems. Section 6 closes by indicating some future research directions.

2. Derivation

The derivation of our proximal distance algorithms exploits the majorization-minimization (MM) principle (Hunter and Lange, 2004; Lange, 2010). In minimizing a function $f(\boldsymbol{x})$, the MM principle exploits a surrogate function $g(\boldsymbol{x} \mid \boldsymbol{x}_k)$ that majorizes $f(\boldsymbol{x})$ around the current iterate $\boldsymbol{x}_k$. Majorization mandates both domination $g(\boldsymbol{x} \mid \boldsymbol{x}_k) \geq f(\boldsymbol{x})$ for all feasible $\boldsymbol{x}$ and tangency $g(\boldsymbol{x}_k \mid \boldsymbol{x}_k) = f(\boldsymbol{x}_k)$ at the anchor $\boldsymbol{x}_k$. If $\boldsymbol{x}_{k+1}$ minimizes $g(\boldsymbol{x} \mid \boldsymbol{x}_k)$, then the

descent property $f(\boldsymbol{x}_{k+1}) \leq f(\boldsymbol{x}_k)$ follows from the string of inequalities and equalities

$$f(\boldsymbol{x}_{k+1}) \;\leq\; g(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k) \;\leq\; g(\boldsymbol{x}_k \mid \boldsymbol{x}_k) \;=\; f(\boldsymbol{x}_k).$$

Clever selection of the surrogate $g(\boldsymbol{x} \mid \boldsymbol{x}_{k+1})$ can lead to a simple algorithm with an explicit update that requires little computation per iterate. The number of iterations until convergence of an MM algorithm depends on how tightly $g(\boldsymbol{x} \mid \boldsymbol{x}_k)$ hugs $f(\boldsymbol{x})$. Constraint satisfaction is built into any MM algorithm. If maximization of $f(\boldsymbol{x})$ is desired, then the objective $f(\boldsymbol{x})$ should dominate the surrogate $g(\boldsymbol{x} \mid \boldsymbol{x}_k)$ subject to the tangency condition. The next iterate $\boldsymbol{x}_{k+1}$ is then chosen to maximize $g(\boldsymbol{x} \mid \boldsymbol{x}_k)$. The minorization-maximization version of the MM principle guarantees the ascent property.

The constraint set C over which the loss $f(\boldsymbol{x})$ is minimized can usually be expressed as an intersection $\cap_{i=1}^m C_i$ of closed sets. It is natural to define the penalty

$$q(\boldsymbol{x}) \;=\; \frac{1}{2}\sum_{i=1}^m \alpha_i \operatorname{dist}(\boldsymbol{x}, C_i)^2$$

using a convex combination of the squared distances. The neutral choice $\alpha_i = \frac{1}{m}$ is one we prefer in practice. Distance majorization gives the surrogate function

$$\begin{aligned} g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k) &= f(\boldsymbol{x}) + \frac{\rho}{2}\sum_{i=1}^m \alpha_i \|\boldsymbol{x} - P_{C_i}(\boldsymbol{x}_k)\|^2 \\ &= f(\boldsymbol{x}) + \frac{\rho}{2}\Big\|\boldsymbol{x} - \sum_{i=1}^m \alpha_i P_{C_i}(\boldsymbol{x}_k)\Big\|^2 + c_k \end{aligned}$$

for an irrelevant constant c_k. If we put $\boldsymbol{y}_k = \sum_{i=1}^m \alpha_i P_{C_i}(\boldsymbol{x}_k)$, then by definition the minimum of the surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k)$ occurs at the proximal point

$$\boldsymbol{x}_{k+1} \;=\; \operatorname{prox}_{\rho^{-1} f}(\boldsymbol{y}_k). \tag{2}$$

We call this MM algorithm the proximal distance algorithm. The penalty $q(\boldsymbol{x})$ is generally smooth because

$$\nabla \frac{1}{2}\operatorname{dist}(\boldsymbol{x}, C)^2 \;=\; \boldsymbol{x} - P_C(\boldsymbol{x})$$

at any point $\boldsymbol{x}$ where the projection $P_C(\boldsymbol{x})$ is single valued (Borwein and Lewis, 2006; Lange, 2016). This is always true for convex sets and almost always true for nonconvex sets. For the moment, we will ignore the possibility that $P_C(\boldsymbol{x})$ is multi-valued.

For the special case of projection of an external point $\boldsymbol{z}$ onto the intersection C of the closed sets C_i, one should take $f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{z} - \boldsymbol{x}\|^2$. The proximal distance iterates then obey the explicit formula

$$\boldsymbol{x}_{k+1} \;=\; \frac{1}{1+\rho}(\boldsymbol{z} + \rho \boldsymbol{y}_k).$$

Linear programming with arbitrary convex constraints is another example. Here the loss is $f(\boldsymbol{x}) = \boldsymbol{v}^t \boldsymbol{x}$, and the update reduces to

$$\boldsymbol{x}_{k+1} \;=\; \boldsymbol{y}_k - \frac{1}{\rho}\boldsymbol{v}.$$

If the proximal map is impossible to calculate, but $f(\boldsymbol{x})$ is L-smooth ($\nabla f(\boldsymbol{x})$ is Lipschitz with constant L), then one can substitute the standard majorization

$$f(\boldsymbol{x}) \leq f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^t(\boldsymbol{x}-\boldsymbol{x}_k) + \frac{L}{2}\|\boldsymbol{x}-\boldsymbol{x}_k\|^2$$

for $f(\boldsymbol{x})$. Minimizing the sum of the loss majorization plus the penalty majorization leads to the MM update

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \frac{1}{L+\rho}[-\nabla f(\boldsymbol{x}_k) + L\boldsymbol{x}_k + \rho\boldsymbol{y}_k] \\ &= \boldsymbol{x}_k - \frac{1}{L+\rho}[\nabla f(\boldsymbol{x}_k) + \rho\nabla q(\boldsymbol{x}_k)]. \end{aligned} \tag{3}$$

This is a gradient descent algorithm without an intervening proximal map.

In moderate-dimensional problems, local quadratic approximation of $f(\boldsymbol{x})$ can lead to a viable algorithm. For instance, in generalized linear statistical models, Xu et al. (2017) suggest replacing the observed information matrix by the expected information matrix. The latter matrix has the advantage of being positive semidefinite. In our notation, if $\boldsymbol{A}_k \approx d^2 f(\boldsymbol{x}_k)$, then an approximate quadratic surrogate is

$$f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^t(\boldsymbol{x}-\boldsymbol{x}_k) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_k)^t\boldsymbol{A}_k(\boldsymbol{x}-\boldsymbol{x}_k) + \frac{\rho}{2}\|\boldsymbol{x}-\boldsymbol{y}_k\|^2.$$

The natural impulse is to update $\boldsymbol{x}$ by the Newton step

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - (\boldsymbol{A}_k + \rho\boldsymbol{I})^{-1}[\nabla f(\boldsymbol{x}_k) - \rho\boldsymbol{y}_k]. \tag{4}$$

This choice does not necessarily decrease $f(\boldsymbol{x})$. Step halving or another form of backtracking restores the descent property.

A more valid concern is the effort expended in matrix inversion. If $\boldsymbol{A}_k$ is dense and constant, then extracting the spectral decomposition $\boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^t$ of $\boldsymbol{A}$ reduces formula (4) to

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \boldsymbol{V}(\boldsymbol{D}+\rho\boldsymbol{I})^{-1}\boldsymbol{V}^t[\nabla f(\boldsymbol{x}_k) - \rho\boldsymbol{y}_k],$$

which can be implemented as a sequence of matrix-vector multiplications. Alternatively, one can take just a few terms of the series

$$(\boldsymbol{A}_k + \rho\boldsymbol{I})^{-1} = \rho^{-1}\sum_{j=0}^{\infty}(-\rho^{-1}\boldsymbol{A}_k)^j$$

when ρ is sufficiently large. For a generalized linear model, parameter updating involves solving the linear system

$$(\boldsymbol{Z}^t\boldsymbol{W}_k\boldsymbol{Z} + \rho\boldsymbol{I})\boldsymbol{x} = \boldsymbol{Z}^t\boldsymbol{W}_k^{1/2}\boldsymbol{v}_k + \rho\boldsymbol{y}_k \tag{5}$$

for $\boldsymbol{W}_k$ a diagonal matrix with positive diagonal entries. This task is equivalent to minimizing the least squares criterion

$$\left\|\begin{pmatrix}\boldsymbol{W}_k^{1/2}\boldsymbol{Z} \\ \sqrt{\rho}\boldsymbol{I}\end{pmatrix}\boldsymbol{x} - \begin{pmatrix}\boldsymbol{v}_k \\ \sqrt{\rho}\boldsymbol{y}_k\end{pmatrix}\right\|^2. \tag{6}$$

In the unweighted case, extracting the singular value decomposition $\boldsymbol{Z} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$ facilitates solving the system of equations (5). The svd decomposition is especially cheap if there is a substantial mismatch between the number rows and columns of $\boldsymbol{Z}$. For sparse $\boldsymbol{Z}$, the conjugate gradient algorithm adapted to least squares (Paige and Saunders, 1982b) is subject to much less ill conditioning than the standard conjugate gradient algorithm. Indeed, the algorithm LSQR and its sparse version LSMR (Fong and Saunders, 2011) perform well even when the matrix $(\boldsymbol{Z}^t\boldsymbol{W}_k^{1/2}, \sqrt{\rho}\boldsymbol{I})^t$ is ill conditioned.

The proximal distance principle also applies to unconstrained problems. For example, consider the problem of minimizing a penalized loss $\ell(\boldsymbol{x}) + p(\boldsymbol{A}\boldsymbol{x})$. The presence of the linear transformation $\boldsymbol{A}\boldsymbol{x}$ in the penalty complicates optimization. The strategy of parameter splitting introduces a new variable $\boldsymbol{y}$ and minimizes $\ell(\boldsymbol{x}) + p(\boldsymbol{y})$ subject to the constraint $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$. If $P_M(\boldsymbol{z})$ denotes projection onto the manifold

$$M = \{\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y}) : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}\},$$

then the constrained problem can be solved approximately by minimizing the function

$$\ell(\boldsymbol{x}) + p(\boldsymbol{y}) + \frac{\rho}{2}\operatorname{dist}(\boldsymbol{z}, M)^2$$

for large ρ. If $P_M(\boldsymbol{z}_k)$ consists of two subvectors $\boldsymbol{u}_k$ and $\boldsymbol{v}_k$ corresponding to $\boldsymbol{x}_k$ and $\boldsymbol{y}_k$, then the proximal distance updates are

$$\boldsymbol{x}_{k+1} = \operatorname{prox}_{\rho^{-1}\ell}(\boldsymbol{u}_k) \quad \text{and} \quad \boldsymbol{y}_{k+1} = \operatorname{prox}_{\rho^{-1}p}(\boldsymbol{v}_k).$$

Given the matrix $\boldsymbol{A}$ is $n \times p$, one can attack the projection by minimizing the function

$$q(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{u}\|^2 + \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{v}\|^2.$$

This leads to the solution

$$\boldsymbol{x} = (\boldsymbol{I}_p + \boldsymbol{A}^t\boldsymbol{A})^{-1}(\boldsymbol{A}^t\boldsymbol{v} + \boldsymbol{u}) \quad \text{and} \quad \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}.$$

If $n < p$, then the Woodbury formula

$$(\boldsymbol{I}_p + \boldsymbol{A}^t\boldsymbol{A})^{-1} = \boldsymbol{I}_p - \boldsymbol{A}^t(\boldsymbol{I}_n + \boldsymbol{A}\boldsymbol{A}^t)^{-1}\boldsymbol{A}$$

reduces the expense of matrix inversion.

Traditionally, convex constraints have been posed as inequalities $C = \{\boldsymbol{x} : a(\boldsymbol{x}) \leq t\}$. Parikh and Boyd (2013) point out how to project onto such sets. The relevant Lagrangian for projecting an external point $\boldsymbol{y}$ amounts to

$$\mathcal{L}(\boldsymbol{x}, \lambda) = \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{x}\|^2 + \lambda[a(\boldsymbol{x}) - t]$$

with $\lambda \geq 0$. The corresponding stationarity condition

$$\boldsymbol{0} = \boldsymbol{x} - \boldsymbol{y} + \lambda\nabla a(\boldsymbol{x}), \tag{7}$$

can be interpreted as $a[\text{prox}_{\lambda a}(\boldsymbol{y})] = t$. One can solve this one-dimensional equation for λ by bisection. Once λ is available, $\boldsymbol{x} = \text{prox}_{\lambda a}(\boldsymbol{y})$ is available as well. Parikh and Boyd (2013) note that the value $a[\text{prox}_{\lambda a}(\boldsymbol{y})]$ is decreasing in λ. One can verify their claim by implicit differentiation of equation (7). This gives

$$\frac{d}{d\lambda}\boldsymbol{x} \;=\; -[\boldsymbol{I} + \lambda d^2 a(\boldsymbol{x})]^{-1}\nabla a(\boldsymbol{x})$$

and consequently the chain rule inequality

$$\frac{d}{d\lambda}a[\text{prox}_{\lambda a}(\boldsymbol{y})] \;=\; -da(\boldsymbol{x})[\boldsymbol{I} + \lambda d^2 a(\boldsymbol{x})]^{-1}\nabla a(\boldsymbol{x}) \;\le\; 0.$$

3. Convergence: Convex Case

In the presence of convexity, the proximal distance algorithm reduces to a proximal gradient algorithm. This follows from the representation

$$\boldsymbol{y} \;=\; \sum_{i=1}^m \alpha_i P_{C_i}(\boldsymbol{x}) \;=\; \boldsymbol{x} - \sum_{i=1}^m \alpha_i \Big[\boldsymbol{x} - P_{C_i}(\boldsymbol{x})\Big] \;=\; \boldsymbol{x} - \nabla q(\boldsymbol{x})$$

involving the penalty $q(\boldsymbol{x})$. Thus, the proximal distance algorithm can be expressed as

$$\boldsymbol{x}_{k+1} \;=\; \text{prox}_{\rho^{-1}f}[\boldsymbol{x}_k - \nabla q(\boldsymbol{x}_k)].$$

In this regard, there is the implicit assumption that $q(\boldsymbol{x})$ is 1-smooth. This is indeed the case. According to the Moreau decomposition (Bauschke and Combettes, 2011), for a single closed convex set C

$$\nabla q(\boldsymbol{x}) \;=\; \boldsymbol{x} - P_C(\boldsymbol{x}) \;=\; \text{prox}_{\delta_C^\star}(\boldsymbol{x}),$$

where $\delta_C^\star(\boldsymbol{x})$ is the Fenchel conjugate of the indicator function

$$\delta_C(\boldsymbol{x}) \;=\; \begin{cases} 0 & \boldsymbol{x} \in C \\ \infty & \boldsymbol{x} \notin C. \end{cases}$$

Because proximal operators of closed convex functions are nonexpansive (Bauschke and Combettes, 2011), the result follows for a single set. For the general penalty $q(\boldsymbol{x})$ with m sets, the Lipschitz constants are scaled by the convex coefficients α_i and added to produce an overall Lipschitz constant of 1.

It is enlightening to view the proximal distance algorithm through the lens of concave-convex programming. Recall that the function

$$s(\boldsymbol{x}) \;=\; \sup_{\boldsymbol{y}\in C}\left[\boldsymbol{y}^t\boldsymbol{x} - \frac{1}{2}\|\boldsymbol{y}\|^2\right] \;=\; \frac{1}{2}\|\boldsymbol{x}\|^2 - \frac{1}{2}\,\text{dist}(\boldsymbol{x}, C)^2 \tag{8}$$

is closed and convex for any nonempty closed set C. Danskin's theorem (Lange, 2016) justifies the directional derivative expression

$$d_{\boldsymbol{v}}s(\boldsymbol{x}) \;=\; \sup_{\boldsymbol{y}\in P_C(\boldsymbol{x})} \boldsymbol{y}^t\boldsymbol{v} \;=\; \sup_{\boldsymbol{y}\in \text{conv}\, P_C(\boldsymbol{x})} \boldsymbol{y}^t\boldsymbol{v}.$$

This equality allows us to identify the subdifferential $\partial s(\boldsymbol{x})$ as the convex hull $\text{conv}\, P_C(\boldsymbol{x})$. For any $\boldsymbol{y} \in \partial s(\boldsymbol{x}_k)$, the supporting hyperplane inequality entails

$$\begin{aligned} \frac{1}{2}\,\text{dist}(\boldsymbol{x}, C)^2 &= \frac{1}{2}\|\boldsymbol{x}\|^2 - s(\boldsymbol{x}) \\ &\le \frac{1}{2}\|\boldsymbol{x}\|^2 - s(\boldsymbol{x}_k) - \boldsymbol{y}^t(\boldsymbol{x} - \boldsymbol{x}_k) \\ &= \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2 + d, \end{aligned}$$

where d is a constant not depending on $\boldsymbol{x}$. The same majorization can be generated by rearranging the majorization

$$\frac{1}{2}\,\text{dist}(\boldsymbol{x}, C)^2 \;\le\; \frac{1}{2}\sum_i \beta_i\|\boldsymbol{x} - \boldsymbol{p}_i\|^2$$

when $\boldsymbol{y}$ is the convex combination $\sum_i \beta_i\boldsymbol{p}_i$ of vectors $\boldsymbol{p}_i$ from $P_C(\boldsymbol{x}_k)$. These facts demonstrate that the proximal distance algorithm minimizing

$$f(\boldsymbol{x}) + \frac{\rho}{2}\,\text{dist}(\boldsymbol{x}, C)^2 \;=\; f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{x}\|^2 - \rho s(\boldsymbol{x})$$

is a special case of concave-convex programming when $f(\boldsymbol{x})$ is convex. It is worth emphasizing that $f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{x}\|^2$ is often strongly convex regardless of whether $f(\boldsymbol{x})$ itself is convex. If we replace the penalty $\text{dist}(\boldsymbol{x}, C)^2$ by the penalty $\text{dist}(\boldsymbol{D}\boldsymbol{x}, C)^2$ for a matrix $\boldsymbol{D}$, then the function $s(\boldsymbol{D}\boldsymbol{x})$ is still closed and convex, and minimization of $f(\boldsymbol{x}) + \frac{\rho}{2}\,\text{dist}(\boldsymbol{D}\boldsymbol{x}, C)^2$ can also be viewed as an exercise in concave-convex programming.

In the presence of convexity, the proximal distance algorithm is guaranteed to converge. Our exposition relies on well-known operator results (Bauschke and Combettes, 2011). Proximal operators in general and projection operators in particular are nonexpansive and averaged. By definition an averaged operator

$$M(\boldsymbol{x}) \;=\; \alpha\boldsymbol{x} + (1-\alpha)N(\boldsymbol{x})$$

is a convex combination of a nonexpansive operator $N(\boldsymbol{x})$ and the identity operator $\boldsymbol{I}$. The averaged operators on $\mathbb{R}^p$ with $\alpha \in (0, 1)$ form a convex set closed under functional composition. Furthermore, $M(\boldsymbol{x})$ and the base operator $N(\boldsymbol{x})$ share their fixed points. The celebrated theorem of Krasnosel'skii (1955) and Mann (1953) says that if an averaged operator $M(\boldsymbol{x}) = \alpha\boldsymbol{x} + (1-\alpha)N(\boldsymbol{x})$ possesses one or more fixed points, then the iteration scheme $\boldsymbol{x}_{k+1} = M(\boldsymbol{x}_k)$ converges to a fixed point.

These results immediately apply to minimization of the penalized loss

$$h_\rho(\boldsymbol{x}) \;=\; f(\boldsymbol{x}) + \frac{\rho}{2}\sum_{i=1}^m \alpha_i\,\text{dist}(\boldsymbol{x}, C_i)^2. \tag{9}$$

Given the choice $\boldsymbol{y}_k = \sum_{i=1}^m \alpha_i P_{C_i}(\boldsymbol{x}_k)$, the algorithm map $\boldsymbol{x}_{k+1} = \text{prox}_{\rho^{-1}f}(\boldsymbol{y}_k)$ is an averaged operator, being the composition of two averaged operators. Hence, the Krasnosel'skii-Mann theorem guarantees convergence to a fixed point if one or more exist. Now $\boldsymbol{z}$ is a fixed point if and only if

$$h_\rho(\boldsymbol{z}) \;\le\; f(\boldsymbol{x}) + \frac{\rho}{2}\sum_{i=1}^m \alpha_i\|\boldsymbol{x} - P_{C_i}(\boldsymbol{z})\|^2$$

for all $\boldsymbol{x}$. In the presence of convexity, this is equivalent to the directional derivative inequality

$$0 \;\leq\; d_{\boldsymbol{v}} f(\boldsymbol{z}) + \rho \sum_{i=1}^{m} \alpha_i [\boldsymbol{z} - P_{C_i}(\boldsymbol{z})]^t \boldsymbol{v} \;=\; d_{\boldsymbol{v}} h_\rho(\boldsymbol{z})$$

for all $\boldsymbol{v}$, which is in turn equivalent to $\boldsymbol{z}$ minimizing $h_\rho(\boldsymbol{x})$. Hence, if $h_\rho(\boldsymbol{x})$ attains its minimum value, then the proximal distance iterates converge to a minimum point.

Convergence of the overall proximal distance algorithm is tied to the convergence of the classical penalty method (Beltrami, 1970). In our setting, the loss is $f(\boldsymbol{x})$, and the penalty is $q(\boldsymbol{x}) = \frac{1}{2}\sum_{i=1}^m \alpha_i \operatorname{dist}(\boldsymbol{x}, C_i)^2$. Assuming the objective $f(\boldsymbol{x}) + \rho q(\boldsymbol{x})$ is coercive for all $\rho \geq 0$, the theory mandates that the solution path $\boldsymbol{x}_\rho$ is bounded and any limit point of the path attains the minimum value of $f(\boldsymbol{x})$ subject to the constraints. Furthermore, if $f(\boldsymbol{x})$ is coercive and possesses a unique minimum point in the constraint set C, then the path $\boldsymbol{x}_\rho$ converges to that point.

Proximal distance algorithms often converge at a painfully slow rate. Following Mairal (2013), one can readily exhibit a precise bound.

Proposition 1 *Suppose C is closed and convex and $f(\boldsymbol{x})$ is convex. If the point $\boldsymbol{z}$ minimizes $h_\rho(\boldsymbol{x}) = f(\boldsymbol{x}) + \frac{\rho}{2}\operatorname{dist}(\boldsymbol{x}, C)^2$, then the proximal distance iterates satisfy*

$$0 \;\leq\; h_\rho(\boldsymbol{x}_{k+1}) - h_\rho(\boldsymbol{z}) \;\leq\; \frac{\rho}{2(k+1)}\|\boldsymbol{z} - \boldsymbol{x}_0\|^2.$$

The $O(\rho k^{-1})$ convergence rate of the proximal distance algorithm suggests that one should slowly send ρ to ∞ and refuse to wait until convergence occurs for any given ρ. It also suggests that Nesterov acceleration may vastly improve the chances for convergence. Nesterov acceleration for the general proximal gradient algorithm with loss $\ell(\boldsymbol{x})$ and penalty $p(\boldsymbol{x})$ takes the form

$$\begin{aligned} \boldsymbol{z}_k &= \boldsymbol{x}_k + \frac{k-1}{k+d-1}(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) \\ \boldsymbol{x}_{k+1} &= \operatorname{prox}_{L^{-1}\ell}[\boldsymbol{z}_k - L^{-1}\nabla p(\boldsymbol{z}_k)], \end{aligned} \tag{10}$$

where L is the Lipschitz constant for $\nabla p(\boldsymbol{x})$ and d is typically chosen to be 3. Nesterov acceleration achieves an $O(k^{-2})$ convergence rate (Su et al., 2014), which is vastly superior to the $O(k^{-1})$ rate achieved by proximal gradient descent. The Nesterov update possesses the further desirable property of preserving affine constraints. In other words, if $\boldsymbol{A}\boldsymbol{x}_{k-1} = \boldsymbol{b}$ and $\boldsymbol{A}\boldsymbol{x}_k = \boldsymbol{b}$, then $\boldsymbol{A}\boldsymbol{z}_k = \boldsymbol{b}$ as well. In subsequent examples, we will accelerate our proximal distance algorithms by applying the algorithm map $M(\boldsymbol{x})$ given by equation (2) to the shifted point $\boldsymbol{z}_k$ of equation (10), yielding the accelerated update $\boldsymbol{x}_{k+1} = M(\boldsymbol{z}_k)$. Algorithm 1 provides a schematic of a proximal distance algorithm with Nesterov acceleration. The recent paper of Ghadimi and Lan (2015) extends Nestorov acceleration to nonconvex settings.

In ideal circumstances, one can prove linear convergence of function values in the framework of Karimi et al. (2016).

Proposition 2 *Suppose C is closed and convex and $f(\boldsymbol{x})$ is L-smooth and μ-strongly convex. Then $h_\rho(\boldsymbol{x}) = f(\boldsymbol{x}) + \frac{\rho}{2}\operatorname{dist}(\boldsymbol{x}, C)^2$ possesses a unique minimum point $\boldsymbol{y}$, and the proximal distance iterates $\boldsymbol{x}_k$ satisfy*

$$h_\rho(\boldsymbol{x}_k) - h_\rho(\boldsymbol{y}) \;\leq\; \Big(1 - \frac{\mu^2}{2(L+\rho)^2}\Big)^k [h_\rho(\boldsymbol{x}_0) - h_\rho(\boldsymbol{y})].$$

We now turn to convergence of the penalty function iterates as the penalty constants ρ_k tends to ∞. To simplify notation, we restrict attention to a single closed constraint set S. Let us start with a proposition requiring no convexity assumptions.

Proposition 3 *If $f(\boldsymbol{x})$ is continuous and coercive and S is compact, then the proximal distance iterates $\boldsymbol{x}_k$ are bounded and the distance to the constraint set satisfies*

$$\operatorname{dist}(\boldsymbol{x}_k, S)^2 \;\leq\; \frac{c}{\rho_k}$$

for some constant c. If in addition $f(\boldsymbol{x})$ is continuously differentiable, then

$$\operatorname{dist}(\boldsymbol{x}_k, S)^2 \;\leq\; \frac{d}{\rho_k^2}$$

for some further constant d. Similar claims hold for the solutions $\boldsymbol{y}_k$ of the penalty problem $\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \frac{\rho_k}{2}\operatorname{dist}(\boldsymbol{x}, S)^2$ except that the assumption that S is compact can be dropped.

As a corollary, if the penalty sequence ρ_k tends to ∞, then all limit points of $\boldsymbol{x}_k$ must obey the constraint. Proposition 3 puts us into position to prove the next important result.

Proposition 4 *If $f(\boldsymbol{x})$ is continuously differentiable and coercive and S is convex, then the penalty function iterates defined by $\boldsymbol{y}_k \in \operatorname{argmin}_{\boldsymbol{x}}[f(\boldsymbol{x}) + \frac{\rho_k}{2}\operatorname{dist}(\boldsymbol{x}, S)^2]$ satisfy*

$$0 \;\leq f(\boldsymbol{y}) - f(\boldsymbol{y}_k) \;\leq\; \tfrac{d+2\sqrt{d}\|\nabla f(\boldsymbol{y})\|}{2\rho_k},$$

where $\boldsymbol{y}$ attains the constrained minimum and d is the constant identified in Proposition 3.

4. Convergence: General Case

Our strategy for addressing convergence in nonconvex problems fixes ρ and relies on Zangwill's global convergence theorem (Luenberger and Ye, 1984). This result depends in turn on the notion of a closed multi-valued map $\boldsymbol{N}(\boldsymbol{x})$. If $\boldsymbol{x}_k$ converges to $\boldsymbol{x}_\infty$ and $\boldsymbol{y}_k \in N(\boldsymbol{x}_k)$ converges to $\boldsymbol{y}_\infty$, then for $\boldsymbol{N}(\boldsymbol{x})$ to be closed, we must have $\boldsymbol{y}_\infty \in N(\boldsymbol{x}_\infty)$. The next proposition furnishes a prominent example.

Proposition 5 *If S is a closed nonempty set in $\mathbb{R}^p$, then the projection operator $P_S(\boldsymbol{x})$ is closed. Furthermore, if the sequence $\boldsymbol{x}_k$ is bounded, then the set $\cup_k P_S(\boldsymbol{x}_k)$ is bounded as well.*

ALGORITHM 1: A typical proximal distance algorithm

input :

$\rho_{\text{initial}} > 0$ # an initial penalty value
$\rho_{\text{inc}} > 1$ # the penalty increment
$\rho_{\max}$ # a maximum penalty value
$K_{\max}$ # the maximum number of iterations
k_ρ # the increment frequency
f # the function to optimize
P_C # the projection onto the constraint set
$\epsilon_{\text{loss}} > 0$ # convergence tolerance for the loss function
$\epsilon_{\text{dist}} > 0$ # convergence tolerance for constraint feasibility

output: A vector $\boldsymbol{x}^+ \approx \operatorname{argmin}_{\boldsymbol{x}} f(\boldsymbol{x})$ # subject to the constraint $\boldsymbol{x} \in C$

$\rho \leftarrow \rho_{\text{initial}}$ # Set initial penalty value
$q_0 = q_1 = \infty$ # Track convergence of f
$d_0 = d_1 = \infty$ # Track distance to constraint
$\boldsymbol{x}_0 = \boldsymbol{x}_1 = \boldsymbol{0}$ # Set initial iterates to origin

# Main algorithm loop
for $k = 2, \ldots, K_{\max}$ **do**
  $\boldsymbol{z}_k \leftarrow \boldsymbol{x}_{k-1} + \frac{k-1}{k+2}(\boldsymbol{x}_{k-1} - \boldsymbol{x}_{k-2})$ # Apply Nesterov acceleration
  $\boldsymbol{x}_{k-2} \leftarrow \boldsymbol{x}_{k-1}$ # Save penultimate iterate
  $\boldsymbol{z}_{\text{proj},k} \leftarrow P_C(\boldsymbol{z}_k)$ # Project onto constraints
  $\boldsymbol{x}_k \leftarrow \text{prox}_{\rho f}(\boldsymbol{z}_{\text{proj},k})$ # Apply proximal distance update
  $q_k \leftarrow f(\boldsymbol{x}_k)$ # Compute new loss
  $d_k \leftarrow \|\boldsymbol{x}_k - \boldsymbol{z}_{\text{proj},k}\|_2$ # Compute new distance to C
  # Exit if converged
  if $|q_k - q_{k-1}| < \epsilon_{\text{loss}}$ and $|d_k - d_{k-1}| < \epsilon_{\text{dist}}$ **then**
    return $\boldsymbol{x}^+ \leftarrow \boldsymbol{x}_{k+1}$
  else
    $q_{k-1} \leftarrow q_k$ # Save current loss
    $d_{k-1} \leftarrow d_k$ # Save current distance to C
    # Update penalty ρ every k_ρ iterations
    if $k = k_\rho$ **then**
      $\rho \leftarrow \min(\rho_{\max}, \rho \times \rho_{\text{inc}})$
      $\boldsymbol{x}_{k-1} \leftarrow \boldsymbol{x}_k$ # Save previous iterate

Zangwill's global convergence theorem is phrased in terms of an algorithm map $M(\boldsymbol{x})$ and a real-valued objective $h(\boldsymbol{x})$. The theorem requires a critical set Γ outside which $M(\boldsymbol{x})$ is closed. Furthermore, all iterates $\boldsymbol{x}_{k+1} \in M(\boldsymbol{x}_k)$ must fall within a compact set. Finally, the descent condition $h(\boldsymbol{y}) \le h(\boldsymbol{x})$ should hold for all $\boldsymbol{y} \in M(\boldsymbol{x})$, with strict inequality when $\boldsymbol{x} \notin \Gamma$. If these conditions are valid, then every convergent subsequence of $\boldsymbol{x}_k$ tends to a point in Γ. In the proximal distance context, we define the complement of Γ to consist of the points $\boldsymbol{x}$ with

$$f(\boldsymbol{y}) + \frac{\rho}{2}\operatorname{dist}(\boldsymbol{y}, S)^2 \quad < \quad f(\boldsymbol{x}) + \frac{\rho}{2}\operatorname{dist}(\boldsymbol{x}, S)^2$$

for all $\boldsymbol{y} \in M(\boldsymbol{x})$. This definition plus the monotonic nature of the proximal distance algorithm

$$\boldsymbol{x}_{k+1} \quad \in \quad M(\boldsymbol{x}_k) \quad = \quad \bigcup_{\boldsymbol{z}_k \in P_S(\boldsymbol{x}_k)} \operatorname{argmin}_{\boldsymbol{x}} \left[f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{z}_k\|^2 \right]$$

force the satisfaction of Zangwill's final requirement. Note that if $f(\boldsymbol{x})$ is differentiable, then a point $\boldsymbol{x}$ belongs to Γ whenever $\boldsymbol{0} \in \nabla f(\boldsymbol{x}) + \rho\boldsymbol{x} - \rho P_S(\boldsymbol{x})$.

In general, the algorithm map $M(\boldsymbol{x})$ is multi-valued in two senses. First, for a given $\boldsymbol{z}_k \in P_S(\boldsymbol{x}_k)$, the minimum may be achieved at multiple points. This contingency is ruled out if the proximal map of $f(\boldsymbol{x})$ is unique. Second, because S may be nonconvex, the projection may be multi-valued. This sounds distressing, but the points $\boldsymbol{x}_k$ where this occurs are exceptionally rare. Accordingly, it makes no practical difference that we restrict the anchor points $\boldsymbol{z}_k$ to lie in $P_S(\boldsymbol{x}_k)$ rather than in $\operatorname{conv} P_S(\boldsymbol{x}_k)$.

Proposition 6 *If S is a closed nonempty set in $\mathbb{R}^p$, then the projection operator $P_S(\boldsymbol{x})$ is single valued except on a set of Lebesgue measure 0.*

In view of the preceding results, one can easily verify the next proposition.

Proposition 7 *The algorithm map $M(\boldsymbol{x})$ is everywhere closed.*

To apply Zangwill's global convergence theory, we must in addition prove that the iterates $\boldsymbol{x}_{k+1} = M(\boldsymbol{x}_k)$ remain within a compact set. This is true whenever the objective is coercive since the algorithm is a descent algorithm. As noted earlier, the coercivity of $f(\boldsymbol{x})$ is a sufficient condition. One can readily concoct other sufficient conditions. For example, if $f(\boldsymbol{x})$ is bounded below, say nonnegative, and S is compact, then the objective is also coercive. Indeed, if S is contained in the ball of radius r about the origin, then

$$\|\boldsymbol{x}\| \quad \le \quad \|\boldsymbol{x} - P_S(\boldsymbol{x})\| + \|P_S(\boldsymbol{x})\| \quad \le \quad \operatorname{dist}(\boldsymbol{x}, S) + r,$$

which proves that $\operatorname{dist}(\boldsymbol{x}, S)$ is coercive. The next proposition summarizes these findings.

Proposition 8 *If S is closed and nonempty, the objective $f(\boldsymbol{x}) + \frac{1}{2}\operatorname{dist}(\boldsymbol{x}, S)^2$ is coercive, and the proximal operator $\operatorname{prox}_{\rho^{-1}f}(\boldsymbol{x})$ is everywhere nonempty, then all limit points of the iterates $\boldsymbol{x}_{k+1} \in M(\boldsymbol{x}_k)$ of the proximal distance algorithm occur in the critical set Γ.*

This result is slightly disappointing. A limit point $\boldsymbol{x}$ could potentially exist with improvement in the objective for some but not all $\boldsymbol{y} \in \operatorname{conv} P_S(\boldsymbol{x})$. This fault is mitigated by the fact that $P_S(\boldsymbol{x})$ is almost always single valued. In common with other algorithms in nonconvex optimization, we also cannot rule out convergence to a local minimum or a saddlepoint. One can improve on Proposition 8 by assuming that the surrogates $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k)$ are all μ-strongly convex. This is a small concession to make because ρ is typically large. If $f(\boldsymbol{x})$ is convex, then $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k)$ is ρ-strongly convex by definition. It is also worth noting that any convex MM surrogate $g(\boldsymbol{x} \mid \boldsymbol{x}_k)$ can be made μ-strongly convex by adding the viscosity penalty $\frac{\mu}{2}\|\boldsymbol{x} - \boldsymbol{x}_k\|^2$ majorizing 0. The addition of a viscosity penalty seldom complicates finding the next iterate $\boldsymbol{x}_{n+1}$ and has little impact on the rate of convergence when $\mu > 0$ is small.

Proposition 9 *Under the μ-strongly convexity assumption on the surrogates $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k)$, the proximal distance iterates satisfy $\lim_{k\to\infty} \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\| = 0$. As a consequence, the set of limit points is connected as well as closed. Furthermore, if each limit point is isolated, then the iterates converge to a critical point.*

Further progress requires even more structure. Fortunately, what we now pursue applies to generic MM algorithms. We start with the concept of a Fréchet subdifferential (Kruger, 2003). If $h(\boldsymbol{x})$ is a function mapping $\mathbb{R}^p$ into $\mathbb{R} \cup \{+\infty\}$, then its Fréchet subdifferential at $\boldsymbol{x} \in \operatorname{dom} f$ is the set

$$\partial^F h(\boldsymbol{x}) \quad = \quad \Big\{\boldsymbol{v} : \liminf_{\boldsymbol{y}\to\boldsymbol{x}} \frac{h(\boldsymbol{y}) - h(\boldsymbol{x}) - \boldsymbol{v}^t(\boldsymbol{y} - \boldsymbol{x})}{\|\boldsymbol{y} - \boldsymbol{x}\|} \geq 0\Big\}.$$

The set $\partial^F h(\boldsymbol{x})$ is closed, convex, and possibly empty. If $h(\boldsymbol{x})$ is convex, then $\partial^F h(\boldsymbol{x})$ reduces to its convex subdifferential. If $h(\boldsymbol{x})$ is differentiable, then $\partial^F h(\boldsymbol{x})$ reduces to its ordinary differential. At a local minimum $\boldsymbol{x}$, Fermat's rule $\boldsymbol{0} \in \partial^F h(\boldsymbol{x})$ holds.

Proposition 10 *In an MM algorithm, suppose that $h(\boldsymbol{x})$ is coercive, that the surrogates $g(\boldsymbol{x} \mid \boldsymbol{x}_k)$ are differentiable, and that the algorithm map $M(\boldsymbol{x})$ is closed. Then every limit point $\boldsymbol{z}$ of the MM sequence $\boldsymbol{x}_k$ is critical in the sense that $\boldsymbol{0} \in \partial^F(-h)(\boldsymbol{z})$.*

We will also need to invoke Łojasiewicz's inequality. This deep result depends on some rather arcane algebraic geometry (Bierstone and Milman, 1988; Bochnak et al., 2013). It applies to semialgebraic functions and their more inclusive cousins semianalytic functions and subanalytic functions. For simplicity we focus on semialgebraic functions. The class of semialgebraic subsets of $\mathbb{R}^p$ is the smallest class that:

a) contains all sets of the form $\{\boldsymbol{x} : q(\boldsymbol{x}) > 0\}$ for a polynomial $q(\boldsymbol{x})$ in p variables,

b) is closed under the formation of finite unions, finite intersections, and set complementation.

A function $a : \mathbb{R}^p \mapsto \mathbb{R}^r$ is said to be semialgebraic if its graph is a semialgebraic set of $\mathbb{R}^{p+r}$. The class of real-valued semialgebraic functions contains all polynomials $p(\boldsymbol{x})$ and all 0/1 indicators of algebraic sets. It is closed under the formation of sums, products, absolute

values, reciprocals when $a(\boldsymbol{x}) \neq 0$, nth roots when $a(\boldsymbol{x}) \geq 0$, and maxima $\max\{a(\boldsymbol{x}), b(\boldsymbol{x})\}$ and minima $\min\{a(\boldsymbol{x}), b(\boldsymbol{x})\}$. For our purposes, it is important to note that $\operatorname{dist}(\boldsymbol{x}, S)$ is a semialgebraic function whenever S is a semialgebraic set.

Łojasiewicz's inequality in its modern form (Bolte et al., 2007) requires a function $h(\boldsymbol{x})$ to be closed (lower semicontinuous) and subanalytic with a closed domain. If $\boldsymbol{z}$ is a critical point of $h(\boldsymbol{x})$, then

$$|h(\boldsymbol{x}) - h(\boldsymbol{z})|^\theta \quad \leq \quad c\|\boldsymbol{v}\|$$

for all $\boldsymbol{x} \in B_r(\boldsymbol{z}) \cap \operatorname{dom} \partial^F h$ satisfying $h(\boldsymbol{x}) > h(\boldsymbol{z})$ and all $\boldsymbol{v}$ in $\partial^F h(\boldsymbol{x})$. Here the exponent $\theta \in [0, 1)$, the radius r, and the constant c depend on $\boldsymbol{z}$. This inequality is valid for semialgebraic functions since they are automatically subanalytic. We will apply Łojasiewicz's inequality to the limit points of an MM algorithm. The next proposition is an elaboration and expansion of known results (Attouch et al., 2010; Bolte et al., 2007; Cui et al., 2018; Kang et al., 2015; Le Thi et al., 2009).

Proposition 11 *In an MM algorithm suppose the objective $h(\boldsymbol{x})$ is coercive, continuous, and subanalytic and all surrogates $g(\boldsymbol{x} \mid \boldsymbol{x}_k)$ are continuous, μ-strongly convex, and satisfy the L-smoothness condition*

$$\|\nabla g(\boldsymbol{a} \mid \boldsymbol{x}_k) - \nabla g(\boldsymbol{b} \mid \boldsymbol{x}_k)\| \quad \leq \quad L\|\boldsymbol{a} - \boldsymbol{b}\|$$

on the compact set $\{\boldsymbol{x} : h(\boldsymbol{x}) \leq h(\boldsymbol{x}_0)\}$. Then the MM iterates $\boldsymbol{x}_{k+1} = \operatorname{argmin}_{\boldsymbol{x}} g(\boldsymbol{x} \mid \boldsymbol{x}_k)$ converge to a critical point.

The last proposition applies to proximal distance algorithms. The loss $f(\boldsymbol{x})$ must be subanalytic and differentiable with a locally Lipschitz gradient. Furthermore, all surrogates $g(\boldsymbol{x} \mid \boldsymbol{x}_k) = f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{y}_k\|^2$ should be coercive and μ-strongly convex. Finally, the constraints sets S_i should be subanalytic. Semialgebraic sets and functions will do. Under these conditions and regardless of how the projected points $P_{S_i}(\boldsymbol{x})$ are chosen, the MM iterates are guaranteed to converge to a critical point.

5. Examples

The following examples highlight the versatility of proximal distance algorithms in a variety of convex and nonconvex settings. Programming details matter in solving these problems. Individual programs are not necessarily long, but care must be exercised in projecting onto constraints, choosing tuning schedules, folding constraints into the domain of the loss, implementing acceleration, and declaring convergence. All of our examples are coded in the Julia programming language. Whenever possible, competing software was run in the Julia environment via the Julia module `MathProgBase` (Dunning et al., 2017; Lubin and Dunning, 2015). The sparse PCA problem relies on the software of Witten et al. (Witten et al., 2009), which is coded in R. Convergence is tested at iteration k by the two criteria

$$|f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k-1})| \quad \leq \quad \epsilon_1[|f(\boldsymbol{x}_{k-1})| + 1] \quad \text{and} \quad \operatorname{dist}(\boldsymbol{x}_k, C) \quad \leq \quad \epsilon_2,$$

where $\epsilon_1 = 10^{-6}$ and $\epsilon_2 = 10^{-4}$ are typical values. The number of iterations until convergence is about 1000 in most examples. This handicap is offset by the simplicity of each stereotyped update. Our code is available as supplementary material to this paper. Readers are encouraged to try the code and adapt it to their own examples.

5.1. Linear Programming

Two different tactics suggest themselves for constructing a proximal distance algorithm. The first tactic rolls the standard affine constraints $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ into the domain of the loss function $\boldsymbol{v}^t\boldsymbol{x}$. The standard nonnegativity requirement $\boldsymbol{x} \geq \boldsymbol{0}$ is achieved by penalization. Let $\boldsymbol{x}_k$ be the current iterate and $\boldsymbol{y}_k = (\boldsymbol{x}_k)_+$ be its projection onto $\mathbb{R}^n_+$. Derivation of the proximal distance algorithm relies on the Lagrangian

$$\boldsymbol{v}^t\boldsymbol{x} + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{y}_k\|^2 + \boldsymbol{\lambda}^t(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}).$$

One can multiply the corresponding stationarity equation

$$\boldsymbol{0} \;=\; \boldsymbol{v} + \rho(\boldsymbol{x} - \boldsymbol{y}_k) + \boldsymbol{A}^t\boldsymbol{\lambda}$$

by $\boldsymbol{A}$ and solve for the Lagrange multiplier $\boldsymbol{\lambda}$ in the form

$$\boldsymbol{\lambda} \;=\; (\boldsymbol{A}\boldsymbol{A}^t)^{-1}(\rho\boldsymbol{A}\boldsymbol{y}_k - \rho\boldsymbol{b} - \boldsymbol{A}\boldsymbol{v}), \tag{11}$$

assuming $\boldsymbol{A}$ has full row rank. Inserting this value into the stationarity equation gives the MM update

$$\boldsymbol{x}_{k+1} \;=\; \boldsymbol{y}_k - \frac{1}{\rho}\boldsymbol{v} - \boldsymbol{A}^-\left(\boldsymbol{A}\boldsymbol{y}_k - \boldsymbol{b} - \frac{1}{\rho}\boldsymbol{A}\boldsymbol{v}\right), \tag{12}$$

where $\boldsymbol{A}^- = \boldsymbol{A}^t(\boldsymbol{A}\boldsymbol{A}^t)^{-1}$ is the pseudo-inverse of $\boldsymbol{A}$.

The second tactic folds the nonnegativity constraints into the domain of the loss. Let $\boldsymbol{p}_k$ denote the projection of $\boldsymbol{x}_k$ onto the affine constraint set $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$. Fortunately, the surrogate function $\boldsymbol{v}^t\boldsymbol{x} + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{p}_k\|^2$ splits the parameters. Minimizing one component at a time gives the update $\boldsymbol{x}_{k+1}$ with components

$$x_{k+1,j} \;=\; \max\left\{p_{kj} - \frac{v_j}{\rho}, 0\right\}. \tag{13}$$

The projection $\boldsymbol{p}_k$ can be computed via

$$\boldsymbol{p}_k \;=\; \boldsymbol{x}_k - \boldsymbol{A}^-(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}), \tag{14}$$

where $\boldsymbol{A}^-$ is again the pseudo-inverse of $\boldsymbol{A}$.

Table 1 compares the accelerated versions of these two proximal distance algorithms to two efficient solvers. The first is the open-source Splitting Cone Solver (SCS) (O'Donoghue et al., 2016), which relies on a fast implementation of ADMM. The second is the commercial Gurobi solver, which ships with implementations of both the simplex method and a barrier (interior point) method; in this example, we use its barrier algorithm. The first seven rows of the table summarize linear programs with dense data $\boldsymbol{A}$, $\boldsymbol{b}$, and $\boldsymbol{v}$. The bottom six rows rely on random sparse matrices $\boldsymbol{A}$ with sparsity level 0.01. For dense problems, the proximal distance algorithms start the penalty constant ρ at 1 and double it every 100 iterations. Because we precompute and cache the pseudoinverse $\boldsymbol{A}^-$ of $\boldsymbol{A}$, the updates (12) and (13) reduce to vector additions and matrix-vector multiplications.

Table 1: CPU times and optima for linear programming. Here m is the number of constraints, n is the number of variables, PD1 is the proximal distance algorithm over an affine domain, PD2 is the proximal distance algorithm over a nonnegative domain, SCS is the Splitting Cone Solver, and Gurobi is the Gurobi solver. After $m = 512$ the constraint matrix $\boldsymbol{A}$ is initialized to be sparse with sparsity level $s = 0.01$.

Dimensions		Optima				CPU Times (secs)			
m	n	PD1	PD2	SCS	Gurobi	PD1	PD2	SCS	Gurobi
2	4	0.2629	0.2629	0.2629	0.2629	0.0142	0.0010	0.0034	0.0038
4	8	1.0455	1.0457	1.0456	1.0455	0.0212	0.0021	0.0009	0.0011
8	16	2.4513	2.4515	2.4514	2.4513	0.0361	0.0048	0.0018	0.0029
16	32	3.4226	3.4231	3.4225	3.4223	0.0847	0.0104	0.0090	0.0036
32	64	6.2398	6.2407	6.2397	6.2398	0.1428	0.0151	0.0140	0.0055
64	128	14.671	14.674	14.671	14.671	0.2117	0.0282	0.0587	0.0088
128	256	27.116	27.125	27.116	27.116	0.3993	0.0728	0.8436	0.0335
256	512	58.501	58.512	58.494	58.494	0.7426	0.1538	2.5409	0.1954
512	1024	135.35	135.37	135.34	135.34	1.6413	0.5799	5.0648	1.7179
1024	2048	254.50	254.55	254.47	254.48	2.9541	3.2127	3.9433	0.6787
2048	4096	533.29	533.35	533.23	533.23	7.3669	17.318	25.614	5.2475
4096	8192	991.78	991.88	991.67	991.67	30.799	95.974	98.347	46.957
8192	16384	2058.8	2059.1	2058.5	2058.5	316.44	623.42	454.23	400.59

For sparse problems the proximal distance algorithms update ρ by a factor of 1.5 every 50 iterations. To avoid computing large pseudoinverses, we appeal to the LSQR variant of the conjugate gradient method (Paige and Saunders, 1982b,a) to solve the linear systems (11) and (14). The optima of all four methods agree to about 4 digits of accuracy. It is hard to declare an absolute winner in these comparisons. Gurobi and SCS clearly perform better on low-dimensional problems, but the proximal distance algorithms are competitive as dimensions increase. PD1, the proximal distance algorithm over an affine domain, tends to be more accurate than PD2. If high accuracy is not a concern, then the proximal distance algorithms are easily accelerated with a more aggressive update schedule for ρ.

5.2. Constrained Least Squares

Constrained least squares programming subsumes constrained quadratic programming. A typical quadratic program involves minimizing the quadratic $\frac{1}{2}\boldsymbol{x}^t\boldsymbol{Q}\boldsymbol{x} - \boldsymbol{p}^t\boldsymbol{x}$ subject to $\boldsymbol{x} \in C$ for a positive definite matrix $\boldsymbol{Q}$. Quadratic programming can be reformulated as least squares by taking the Cholesky decomposition $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^t$ of $\boldsymbol{Q}$ and noting that

$$\frac{1}{2}\boldsymbol{x}^t\boldsymbol{Q}\boldsymbol{x} - \boldsymbol{p}^t\boldsymbol{x} \;=\; \frac{1}{2}\|\boldsymbol{L}^{-1}\boldsymbol{p} - \boldsymbol{L}^t\boldsymbol{x}\|^2 - \frac{1}{2}\|\boldsymbol{L}^{-1}\boldsymbol{p}\|^2.$$

The constraint $\boldsymbol{x} \in C$ applies in both settings. It is particularly advantageous to reframe a quadratic program as a least squares problem when $\boldsymbol{Q}$ is already presented in factored

Dimensions		Optima			CPU Times		
n	p	PD	IPOPT	Gurobi	PD	IPOPT	Gurobi
16	8	4.1515	4.1515	4.1515	0.0038	0.0044	0.0010
32	16	10.8225	10.8225	10.8225	0.0036	0.0039	0.0010
64	32	29.6218	29.6218	29.6218	0.0079	0.0079	0.0019
128	64	43.2626	43.2626	43.2626	0.0101	0.0078	0.0033
256	128	111.7642	111.7642	111.7642	0.0872	0.0151	0.0136
512	256	231.6455	231.6454	231.6454	0.1119	0.0710	0.0619
1024	512	502.1276	502.1276	502.1276	0.2278	0.4013	0.2415
2048	1024	994.2447	994.2447	994.2447	1.2575	2.3346	1.1682
4096	2048	2056.8381	2056.8381	2056.8381	1.3253	15.2214	7.4971
8192	4096	4103.4611	4103.4611	4103.4611	3.0289	146.1604	49.7411
16384	8192	8295.2136	8295.2136	8295.2136	6.8739	732.1039	412.3612

Table 2: CPU times and optima for simplex-constrained least squares. Here $\boldsymbol{A} \in \mathbb{R}^{n \times p}$, PD is the proximal distance algorithm, IPOPT is the Ipopt solver, and Gurobi is the Gurobi solver. After $n = 1024$, the predictor matrix $\boldsymbol{A}$ is sparse.

form or when it is nearly singular (Bemporad, 2018). To simplify subsequent notation, we replace $\boldsymbol{L}^t$ by the rectangular matrix $\boldsymbol{A}$ and $\boldsymbol{L}^{-1}\boldsymbol{p}$ by $\boldsymbol{y}$. The key to solving constrained least squares is to express the proximal distance surrogate as

$$\frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \frac{\rho}{2}\|\boldsymbol{x} - P_C(\boldsymbol{x}_k)\|^2 \;=\; \frac{1}{2}\left\|\begin{pmatrix}\boldsymbol{y}\\ \sqrt{\rho}P_C(\boldsymbol{x}_k)\end{pmatrix} - \begin{pmatrix}\boldsymbol{A}\\ \sqrt{\rho}\boldsymbol{I}\end{pmatrix}\boldsymbol{x}\right\|^2$$

as in equation (6). As noted earlier, in sparse problems the update $\boldsymbol{x}_{k+1}$ can be found by a fast stable conjugate gradient solver.

Table 2 compares the performance of the proximal distance algorithm for least squares estimation with probability-simplex constraints to the open source nonlinear interior point solver Ipopt (Wächter and Biegler, 2005, 2006) and the interior point method of Gurobi. Simplex constrained problems arise in hyperspectral imaging (Heylen et al., 2011; Keshava, 2003), portfolio optimization (Markowitz, 1952), and density estimation (Bunea et al., 2010). Test problems were generated by filling an $n \times p$ matrix $\boldsymbol{A}$ and an n-vector $\boldsymbol{y}$ with standard normal deviates. For sparse problems we set the sparsity level of $\boldsymbol{A}$ to be $10/p$. Our setup ensures that $\boldsymbol{A}$ has full rank and that the quadratic program has a solution. For the proximal distance algorithm, we start ρ at 1 and multiply it by 1.5 every 200 iterations. Table 2 suggests that the proximal distance algorithm and the interior point solvers perform equally well on small dense problems. However, in high-dimensional and low-accuracy environments, the proximal distance algorithm provides much better scalability.

5.3. Closest Kinship Matrix

In genetics studies, kinship is measured by the fraction of genes two individuals share identical by descent. For a given pedigree, the kinship coefficients for all pairs of individuals

appear as entries in a symmetric kinship matrix $\boldsymbol{Y}$. This matrix possesses three crucial properties: a) it is positive semidefinite, b) its entries are nonnegative, and c) its diagonal entries are $\frac{1}{2}$ unless some pedigree members are inbred. Inbreeding is the exception rather than the rule. Kinship matrices can be estimated empirically from single nucleotide polymorphism (SNP) data, but there is no guarantee that the three highlighted properties are satisfied. Hence, it helpful to project $\boldsymbol{Y}$ to the nearest qualifying matrix.

This projection problem is best solved by folding the positive semidefinite constraint into the domain of the Frobenius loss function $\frac{1}{2}\|\boldsymbol{X} - \boldsymbol{Y}\|_F^2$. As we shall see, the alternative of imposing two penalties rather than one is slower and less accurate. Projection onto the constraints implied by conditions b) and c) is trivial. All diagonal entries x_{ii} of $\boldsymbol{X}$ are reset to $\frac{1}{2}$, and all off-diagonal entries x_{ij} are reset to $\max\{x_{ij}, 0\}$. If $P(\boldsymbol{X}_k)$ denotes the current projection, then the proximal distance algorithm minimizes the surrogate

$$\begin{aligned} g(\boldsymbol{X} \mid \boldsymbol{X}_k) &= \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{Y}\|_F^2 + \frac{\rho}{2}\|\boldsymbol{X} - P(\boldsymbol{X}_k)\|_F^2 \\ &= \frac{1+\rho}{2}\left\|\boldsymbol{X} - \frac{1}{1+\rho}\boldsymbol{Y} - \frac{\rho}{1+\rho}P(\boldsymbol{X}_k)\right\|_F^2 + c_k, \end{aligned}$$

where c_k is an irrelevant constant. The minimum is found by extracting the spectral decomposition $\boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^t$ of $\frac{1}{1+\rho}\boldsymbol{Y} + \frac{\rho}{1+\rho}P(\boldsymbol{X}_k)$ and truncating the negative eigenvalues. This gives the update $\boldsymbol{X}_{k+1} = \boldsymbol{U}\boldsymbol{D}_+\boldsymbol{U}^t$ in obvious notation. This proximal distance algorithm and its Nesterov acceleration are simple to implement in a numerically oriented language such as Julia. The most onerous part of the calculation is clearly the repeated eigen-decompositions.

Table 3 compares three versions of the proximal distance algorithm to Dykstra's algorithm (Boyle and Dykstra, 1986). Higham proposed Dykstra's algorithm for the related problem of finding the closest correlation matrix Higham (2002). In Table 3 algorithm PD1 is the unadorned proximal distance algorithm, PD2 is the accelerated proximal distance, and PD3 is the accelerated proximal distance algorithm with the positive semidefinite constraints folded into the domain of the loss. On this demanding problem, these algorithms are comparable to Dykstra's algorithm in speed but slightly less accurate. Acceleration of the proximal distance algorithm is effective in reducing both execution time and error. Folding the positive semidefinite constraint into the domain of the loss function leads to further improvements. The data matrices $\boldsymbol{M}$ in these trials were populated by standard normal deviates and then symmetrized by averaging opposing off-diagonal entries. In algorithm PD1 we set $\rho_k = \max\{1.2^k, 2^{22}\}$. In the accelerated versions PD2 and PD3 we started ρ at 1 and multiplied it by 5 every 100 iterations. At the expense of longer compute times, better accuracy can be achieved by all three proximal distance algorithms with a less aggressive update schedule.

5.4. Projection onto a Second-Order Cone Constraint

Second-order cone programming is one of the unifying themes of convex analysis (Alizadeh and Goldfarb, 2003; Lobo et al., 1998). It revolves around conic constraints of the form $\{\boldsymbol{u} : \|\boldsymbol{A}\boldsymbol{u} + \boldsymbol{b}\| \le \boldsymbol{c}^t\boldsymbol{u} + d\}$. Projection of a vector $\boldsymbol{x}$ onto such a constraint is facilitated by parameter splitting. In this setting parameter splitting introduces a vector $\boldsymbol{w}$, a scalar r, and the two affine constraints $\boldsymbol{w} = \boldsymbol{A}\boldsymbol{u} + \boldsymbol{b}$ and $r = \boldsymbol{c}^t\boldsymbol{u} + d$. The conic constraint then

Table 3: CPU times and optima for the closest kinship matrix problem. Here the kinship matrix is $n \times n$, PD1 is the proximal distance algorithm, PD2 is the accelerated proximal distance, PD3 is the accelerated proximal distance algorithm with the positive semidefinite constraints folded into the domain of the loss, and Dykstra is Dykstra's adaptation of alternating projections. All times are in seconds.

Size	PD1		PD2		PD3		Dykstra	
n	Loss	Time	Loss	Time	Loss	Time	Loss	Time
2	1.64	0.36	1.64	0.01	1.64	0.01	1.64	0.00
4	2.86	0.10	2.86	0.01	2.86	0.01	2.86	0.00
8	18.77	0.21	18.78	0.03	18.78	0.03	18.78	0.00
16	45.10	0.84	45.12	0.18	45.12	0.12	45.12	0.02
32	169.58	4.36	169.70	0.61	169.70	0.52	169.70	0.37
64	837.85	16.77	838.44	2.90	838.43	2.63	838.42	4.32
128	3276.41	91.94	3279.44	18.00	3279.25	14.83	3279.23	19.73
256	14029.07	403.59	14045.30	89.58	14043.59	64.89	14043.46	72.79

reduces to the Lorentz cone constraint $\|\boldsymbol{w}\| \leq r$, for which projection is straightforward (Boyd and Vandenberghe, 2009). If we concatenate the parameters into the single vector

$$\boldsymbol{y} \;=\; \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{w} \\ r \end{pmatrix}$$

and define $L = \{\boldsymbol{y} : \|\boldsymbol{w}\| \leq r\}$ and $M = \{\boldsymbol{y} : \boldsymbol{w} = \boldsymbol{A}\boldsymbol{u} + \boldsymbol{b} \text{ and } r = \boldsymbol{c}^t\boldsymbol{u} + d\}$, then we can rephrase the problem as minimizing $\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{u}\|^2$ subject to $\boldsymbol{y} \in L \cap M$. This is a fairly typical set projection problem except that the $\boldsymbol{w}$ and r components of $\boldsymbol{y}$ are missing in the loss function.

Taking a cue from Example 5.1, we incorporate the affine constraints in the domain of the objective function. If we represent projection onto L by

$$P\begin{pmatrix} \boldsymbol{w}_k \\ r_k \end{pmatrix} \;=\; \begin{pmatrix} \tilde{\boldsymbol{w}}_k \\ \tilde{r}_k \end{pmatrix},$$

then the Lagrangian generated by the proximal distance algorithm amounts to

$$\mathcal{L} \;=\; \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{u}\|^2 + \frac{\rho}{2}\left\|\begin{pmatrix} \boldsymbol{w} - \tilde{\boldsymbol{w}}_k \\ r - \tilde{r}_k \end{pmatrix}\right\|^2 + \boldsymbol{\lambda}^t(\boldsymbol{A}\boldsymbol{u} + \boldsymbol{b} - \boldsymbol{w}) + \theta(\boldsymbol{c}^t\boldsymbol{u} + d - r).$$

This gives rise to a system of three stationarity equations

$$\begin{aligned} \boldsymbol{0} &= \boldsymbol{u} - \boldsymbol{x} + \boldsymbol{A}^t\boldsymbol{\lambda} + \theta\boldsymbol{c} && (15) \\ \boldsymbol{0} &= \rho(\boldsymbol{w} - \tilde{\boldsymbol{w}}_k) - \boldsymbol{\lambda} && (16) \\ 0 &= \rho(r - \tilde{r}_k) - \theta. && (17) \end{aligned}$$

Solving for the multipliers $\boldsymbol{\lambda}$ and θ in equations (16) and (17) and substituting their values in equation (15) yield

$$\begin{aligned} \boldsymbol{0} &= \boldsymbol{u} - \boldsymbol{x} + \rho\boldsymbol{A}^t(\boldsymbol{w} - \tilde{\boldsymbol{w}}_k) + \rho(r - \tilde{r}_k)\boldsymbol{c} \\ &= \boldsymbol{u} - \boldsymbol{x} + \rho\boldsymbol{A}^t(\boldsymbol{A}\boldsymbol{u} + \boldsymbol{b} - \tilde{\boldsymbol{w}}_k) + \rho(\boldsymbol{c}^t\boldsymbol{u} + d - \tilde{r}_k)\boldsymbol{c}. \end{aligned}$$

This leads to the MM update

$$\boldsymbol{u}_{k+1} \;=\; (\rho^{-1}\boldsymbol{I} + \boldsymbol{A}^t\boldsymbol{A} + \boldsymbol{c}\boldsymbol{c}^t)^{-1}[\rho^{-1}\boldsymbol{x} + \boldsymbol{A}^t(\tilde{\boldsymbol{w}}_k - \boldsymbol{b}) + (\tilde{r}_k - d)\boldsymbol{c}]. \qquad (18)$$

The updates $\boldsymbol{w}_{k+1} = \boldsymbol{A}\boldsymbol{u}_{k+1} + \boldsymbol{b}$ and $r_{k+1} = \boldsymbol{c}^t\boldsymbol{u}_{k+1} + d$ follow from the constraints.

Table 4 compares the proximal distance algorithm to SCS and Gurobi. Echoing previous examples, we tailor the update schedule for ρ differently for dense and sparse problems. Dense problems converge quickly and accurately when we set $\rho_0 = 1$ and double ρ every 100 iterations. Sparse problems require a greater range and faster updates of ρ, so we set $\rho_0 = 0.01$ and then multiply ρ by 2.5 every 10 iterations. For dense problems, it is clearly advantageous to cache the spectral decomposition of $\boldsymbol{A}^t\boldsymbol{A} + \boldsymbol{c}\boldsymbol{c}^t$ as suggested in Example 5.2. In this regime, the proximal distance algorithm is as accurate as Gurobi and nearly as fast. SCS is comparable to Gurobi in speed but notably less accurate.

With a large sparse constraint matrix $\boldsymbol{A}$, extraction of its spectral decomposition becomes prohibitive. If we let $\boldsymbol{E} = (\rho^{-1/2}\boldsymbol{I} \;\; \boldsymbol{A}^t \;\; \boldsymbol{c})$, then we must solve a linear system of equations defined by the Gramian matrix $\boldsymbol{G} = \boldsymbol{E}\boldsymbol{E}^t$. There are three reasonable options for solving this system. The first relies on computing and caching a sparse Cholesky decomposition of $\boldsymbol{G}$. The second computes the QR decomposition of the sparse matrix $\boldsymbol{E}$. The R part of the QR decomposition coincides with the Cholesky factor. Unfortunately, every time ρ changes, the Cholesky or QR decomposition must be redone. The third option is the conjugate gradient algorithm. In our experience the QR decomposition offers superior stability and accuracy. When $\boldsymbol{E}$ is very sparse, the QR decomposition is often much faster than the Cholesky decomposition because it avoids forming the dense matrix $\boldsymbol{A}^t\boldsymbol{A}$. Even when only 5% of the entries of $\boldsymbol{A}$ are nonzero, 90% of the entries of $\boldsymbol{A}^t\boldsymbol{A}$ can be nonzero. If exquisite accuracy is not a concern, then the conjugate gradient method provides the fastest update. Table 4 reflects this choice.

5.5. Copositive Matrices

A symmetric matrix $\boldsymbol{M}$ is copositive if its associated quadratic form $\boldsymbol{x}^t\boldsymbol{M}\boldsymbol{x}$ is nonnegative for all $\boldsymbol{x} \geq \boldsymbol{0}$. Copositive matrices find applications in numerous branches of the mathematical sciences (Berman and Plemmons, 1994). All positive semidefinite matrices and all matrices with nonnegative entries are copositive. The variational index

$$\mu(\boldsymbol{M}) \;=\; \min_{\|\boldsymbol{x}\|=1,\, \boldsymbol{x}\geq\boldsymbol{0}} \boldsymbol{x}^t\boldsymbol{M}\boldsymbol{x}$$

is one key to understanding copositive matrices (Hiriart-Urruty and Seeger, 2010). The constraint set S is the intersection of the unit sphere and the nonnegative cone $\mathbb{R}^n_+$. Projection of an external point $\boldsymbol{y}$ onto S splits into three cases. When all components of $\boldsymbol{y}$ are negative, then $P_S(\boldsymbol{y}) = \boldsymbol{e}_i$, where y_i is the least negative component of $\boldsymbol{y}$, and $\boldsymbol{e}_i$ is the standard unit

Table 4: CPU times and optima for the second-order cone projection. Here m is the number of constraints, n is the number of variables, PD is the accelerated proximal distance algorithm, SCS is the Splitting Cone Solver, and Gurobi is the Gurobi solver. After $m = 512$ the constraint matrix $\boldsymbol{A}$ is initialized with sparsity level 0.01.

Dimensions		Optima			CPU Seconds		
m	n	PD	SCS	Gurobi	PD	SCS	Gurobi
2	4	0.10598	0.10607	0.10598	0.0043	0.0103	0.0026
4	8	0.00000	0.00000	0.00000	0.0003	0.0009	0.0022
8	16	0.88988	0.88991	0.88988	0.0557	0.0011	0.0027
16	32	2.16514	2.16520	2.16514	0.0725	0.0012	0.0040
32	64	3.03855	3.03864	3.03853	0.0952	0.0019	0.0094
64	128	4.86894	4.86962	4.86895	0.1225	0.0065	0.0403
128	256	10.5863	10.5843	10.5863	0.1975	0.0810	0.0868
256	512	31.1039	31.0965	31.1039	0.5463	0.3995	0.3405
512	1024	27.0483	27.0475	27.0483	3.7667	1.6692	2.0189
1024	2048	1.45578	1.45569	1.45569	0.5352	0.3691	1.5489
2048	4096	2.22936	2.22930	2.22921	1.0845	2.4531	5.5521
4096	8192	1.72306	1.72202	1.72209	3.1404	17.272	15.204
8192	16384	5.36191	5.36116	5.36144	13.979	133.25	88.024

vector along coordinate direction i. The origin $\mathbf{0}$ is equidistant from all points of S. If any component of $\boldsymbol{y}$ is positive, then the projection is constructed by setting the negative components of $\boldsymbol{y}$ equal to 0, and standardizing the truncated version of $\boldsymbol{y}$ to have Euclidean norm 1.

As a test case for the proximal distance algorithm, consider the Horn matrix (Hall and Newman, 1963)

$$\boldsymbol{M} = \begin{bmatrix} 1 & -1 & 1 & 1 & -1 \\ -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 \end{bmatrix}.$$

The value $\mu(\boldsymbol{M}) = 0$ is attained for the vectors $\frac{1}{\sqrt{2}}(1,1,0,0,0)^t$, $\frac{1}{\sqrt{6}}(1,2,1,0,0)^t$, and equivalent vectors with their entries permuted. Matrices in higher dimensions with the same Horn pattern of 1's and -1's are copositive as well (Johnson and Reams, 2008). A Horn matrix of odd dimension cannot be written as a positive semidefinite matrix, a nonnegative matrix, or a sum of two such matrices.

The proximal distance algorithm minimizes the criterion

$$g(\boldsymbol{x} \mid \boldsymbol{x}_k) = \frac{1}{2}\boldsymbol{x}^t\boldsymbol{M}\boldsymbol{x} + \frac{\rho}{2}\|\boldsymbol{x} - P_S(\boldsymbol{x}_k)\|^2$$

Table 5: CPU times (seconds) and optima for approximating the Horn variational index of a Horn matrix. Here n is the size of Horn matrix, PD is the proximal distance algorithm, aPD is the accelerated proximal distance algorithm, and Mosek is the Mosek solver.

Dimension	Optima			CPU Seconds		
n	PD	aPD	Mosek	PD	aPD	Mosek
4	0.000000	0.000000	feasible	0.5555	0.0124	2.7744
5	0.000000	0.000000	infeasible	0.0039	0.0086	0.0276
8	0.000021	0.000000	feasible	0.0059	0.0083	0.0050
9	0.000045	0.000000	infeasible	0.0055	0.0072	0.0082
16	0.000377	0.000001	feasible	0.0204	0.0237	0.0185
17	0.000441	0.000001	infeasible	0.0204	0.0378	0.0175
32	0.001610	0.000007	feasible	0.0288	0.0288	0.1211
33	0.002357	0.000009	infeasible	0.0242	0.0346	0.1294
64	0.054195	0.000026	feasible	0.0415	0.0494	3.6284
65	0.006985	0.000026	infeasible	0.0431	0.0551	2.7862

and generates the updates

$$\boldsymbol{x}_{k+1} = (\boldsymbol{M} + \rho\boldsymbol{I})^{-1}\rho P_S(\boldsymbol{x}_k).$$

It takes a gentle tuning schedule to get decent results. The choice $\rho_k = 1.2^k$ converges in 600 to 700 iterations from random starting points and reliably yields objective values below 10^{-5} for Horn matrices. The computational burden per iteration is significantly eased by exploiting the cached spectral decomposition of $\boldsymbol{M}$. Table 5 compares the performance of the proximal distance algorithm to the Mosek solver on a range of Horn matrices. Mosek uses semidefinite programming to decide whether $\boldsymbol{M}$ can be decomposed into a sum of a positive semidefinite matrix and a nonnegative matrix. If not, Mosek declares the problem infeasible. Nesterov acceleration improves the final loss for the proximal distance algorithm, but it does not decrease overall computing time.

Testing for copositivity is challenging because neither the loss function nor the constraint set is convex. The proximal distance algorithm offers a fast screening device for checking whether a matrix is copositive. On random 1000×1000 symmetric matrices $\boldsymbol{M}$, the method invariably returns a negative index in less than two seconds of computing time. Because the vast majority of symmetric matrices are not copositive, accurate estimation of the minimum is not required. Table 6 summarizes a few random trials with lower-dimensional symmetric matrices. In higher dimensions, Mosek becomes non-competitive, and Nesterov acceleration is of dubious value.

5.6. Linear Complementarity Problem

The linear complementarity problem (Murty and Yu, 1988) consists of finding vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ with nonnegative components such that $\boldsymbol{x}^t\boldsymbol{y} = 0$ and $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$ for a given square

Table 6: CPU times and optima for testing the copositivity of random symmetric matrices. Here n is the size of matrix, PD is the proximal distance algorithm, aPD is the accelerated proximal distance algorithm, and Mosek is the Mosek solver.

Dimension	Optima			CPU Seconds		
n	PD	aPD	Mosek	PD	aPD	Mosek
4	-0.391552	-0.391561	infeasible	0.0029	0.0031	0.0024
8	-0.911140	-2.050316	infeasible	0.0037	0.0044	0.0045
16	-1.680697	-1.680930	infeasible	0.0199	0.0272	0.0062
32	-2.334520	-2.510781	infeasible	0.0261	0.0242	0.0441
64	-3.821927	-3.628060	infeasible	0.0393	0.0437	0.6559
128	-5.473609	-5.475879	infeasible	0.0792	0.0798	38.3919
256	-7.956365	-7.551814	infeasible	0.1632	0.1797	456.1500

matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$. The natural loss function is $\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2$. To project a vector pair $(\boldsymbol{u},\boldsymbol{v})$ onto the nonconvex constraint set, one considers each component pair (u_i,v_i) in turn. If $u_i \geq \max\{v_i,0\}$, then the nearest pair $(\boldsymbol{x},\boldsymbol{y})$ has components $(x_i,y_i)=(u_i,0)$. If $v_i \geq \max\{u_i,0\}$, then the nearest pair has components $(x_i,y_i)=(0,v_i)$. Otherwise, $(x_i,y_i)=(0,0)$. At each iteration the proximal distance algorithm minimizes the criterion

$$\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2+\frac{\rho}{2}\|\boldsymbol{x}-\tilde{\boldsymbol{x}}_k\|^2+\frac{\rho}{2}\|\boldsymbol{y}-\tilde{\boldsymbol{y}}_k\|^2,$$

where $(\tilde{\boldsymbol{x}}_k,\tilde{\boldsymbol{y}}_k)$ is the projection of $(\boldsymbol{x}_k,\boldsymbol{y}_k)$ onto the constraint set. The stationarity equations become

$$\begin{aligned}\boldsymbol{0} &= -\boldsymbol{A}^t(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})+\rho(\boldsymbol{x}-\tilde{\boldsymbol{x}}_k)\\ \boldsymbol{0} &= \boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}+\rho(\boldsymbol{y}-\tilde{\boldsymbol{y}}_k).\end{aligned}$$

Substituting the value of $\boldsymbol{y}$ from the second equation into the first equation leads to the updates

$$\begin{aligned}\boldsymbol{x}_{k+1} &= [(1+\rho)\boldsymbol{I}+\boldsymbol{A}^t\boldsymbol{A}]^{-1}[\boldsymbol{A}^t(\tilde{\boldsymbol{y}}_k-\boldsymbol{b})+(1+\rho)\tilde{\boldsymbol{x}}_k] \qquad (19)\\ \boldsymbol{y}_{k+1} &= \frac{1}{1+\rho}(\boldsymbol{A}\boldsymbol{x}_{k+1}+\boldsymbol{b})+\frac{\rho}{1+\rho}\tilde{\boldsymbol{y}}_k.\end{aligned}$$

The linear system (19) can be solved in low to moderate dimensions by computing and caching the spectral decomposition of $\boldsymbol{A}^t\boldsymbol{A}$ and in high dimensions by the conjugate gradient method. Table 7 compares the performance of the proximal gradient algorithm to the Gurobi solver on some randomly generated problems.

5.7. Sparse Principal Components Analysis

Let $\boldsymbol{X}$ be an $n\times p$ data matrix gathered on n cases and p predictors. Assume the columns of $\boldsymbol{X}$ are centered to have mean 0. Principal component analysis (PCA) (Hotelling, 1933;

Table 7: CPU times (seconds) and optima for the linear complementarity problem with randomly generated data. Here n is the size of matrix, PD is the accelerated proximal distance algorithm, and Gurobi is the Gurobi solver.

Dimension	Optima		CPU Seconds	
n	PD	Mosek	PD	Mosek
4	0.000000	0.000000	0.0230	0.0266
8	0.000000	0.000000	0.0062	0.0079
16	0.000000	0.000000	0.0269	0.0052
32	0.000000	0.000000	0.0996	0.4303
64	0.000074	0.000000	2.6846	360.5183

Pearson, 1901) operates on the sample covariance matrix $\boldsymbol{S}=\frac{1}{n}\boldsymbol{X}^t\boldsymbol{X}$. Here we formulate a proximal distance algorithm for sparse PCA (SPCA), which has attracted substantial interest in the machine learning community (Berthet and Rigollet, 2013b,a; D'Aspremont et al., 2007; Johnstone and Lu, 2009; Journée et al., 2010; Witten et al., 2009; Zou et al., 2006). According to a result of Ky Fan (Fan, 1949), the first q principal components (PCs) $\boldsymbol{u}_1,\ldots,\boldsymbol{u}_q$ can be extracted by maximizing the function $\mathrm{tr}(\boldsymbol{U}^t\boldsymbol{S}\boldsymbol{U})$ subject to the matrix constraint $\boldsymbol{U}^t\boldsymbol{U}=\boldsymbol{I}_q$, where $\boldsymbol{u}_i$ is the ith column of the $p\times q$ matrix $\boldsymbol{U}$. This constraint set is called a Stiefel manifold. One can impose sparsity by insisting that any given column $\boldsymbol{u}_i$ have at most r nonzero entries. Alternatively, one can require the entire matrix $\boldsymbol{U}$ to have at most r nonzero entries. The latter choice permits sparsity to be distributed non-uniformly across columns.

Extraction of sparse PCs is difficult for three reasons. First, the Stiefel manifold M_q and both sparsity sets are nonconvex. Second, the objective function is concave rather than convex. Third, there is no simple formula or algorithm for projecting onto the intersection of the two constraint sets. Fortunately, it is straightforward to project onto each separately. Let $P_{M_q}(\boldsymbol{U})$ denote the projection of $\boldsymbol{U}$ onto the Stiefel manifold. It is well known that $P_{M_q}(\boldsymbol{U})$ can be calculated by extracting a partial singular value decomposition $\boldsymbol{U}=\boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{W}^t$ of $\boldsymbol{U}$ and setting $P_{M_q}(\boldsymbol{U})=\boldsymbol{V}\boldsymbol{W}^t$ (Golub and Van Loan, 2012). Here $\boldsymbol{V}$ and $\boldsymbol{W}$ are orthogonal matrices of dimension $p\times q$ and $q\times q$, respectively, and $\boldsymbol{\Sigma}$ is a diagonal matrix of dimension $q\times q$. Let $P_{S_r}(\boldsymbol{U})$ denote the projection of $\boldsymbol{U}$ onto the sparsity set

$$S_r = \{\boldsymbol{V}: v_{ij}\neq 0 \text{ for at most } r \text{ entries of each column } \boldsymbol{v}_i\}.$$

Because $P_{S_r}(\boldsymbol{U})$ operates column by column, it suffices to project each column vector $\boldsymbol{u}_i$ to sparsity. This entails nothing more than sorting the entries of $\boldsymbol{u}_i$ by magnitude, saving the r largest, and sending the remaining $p-r$ entries to 0. If the entire matrix $\boldsymbol{U}$ must have at most r nonzero entries, then $\boldsymbol{U}$ can be treated as a concatenated vector during projection.

The key to a good algorithm is to incorporate the Stiefel constraints into the domain of the objective function (Kiers, 1990; Kiers and ten Berge, 1992) and the sparsity constraints into the distance penalty. Thus, we propose decreasing the criterion

$$f(\boldsymbol{U}) = -\frac{1}{2}\mathrm{tr}(\boldsymbol{U}^t\boldsymbol{S}\boldsymbol{U})+\frac{\rho}{2}\mathrm{dist}(\boldsymbol{U},S_r)^2.$$

at each iteration subject to the Stiefel constraints. The loss can be majorized via

$$\begin{aligned} -\frac{1}{2}\operatorname{tr}(\boldsymbol{U}^t\boldsymbol{S}\boldsymbol{U}) &= -\frac{1}{2}\operatorname{tr}[(\boldsymbol{U}-\boldsymbol{U}_k)^t\boldsymbol{S}(\boldsymbol{U}-\boldsymbol{U}_k)] - \operatorname{tr}(\boldsymbol{U}^t\boldsymbol{S}\boldsymbol{U}_k) + \frac{1}{2}\operatorname{tr}(\boldsymbol{U}_k^t\boldsymbol{S}\boldsymbol{U}_k) \\ &\le -\operatorname{tr}(\boldsymbol{U}^t\boldsymbol{S}\boldsymbol{U}_k) + \frac{1}{2}\operatorname{tr}(\boldsymbol{U}_k^t\boldsymbol{S}\boldsymbol{U}_k) \end{aligned}$$

because $\boldsymbol{S}$ is positive semidefinite. The penalty is majorized by

$$\frac{\rho}{2}\operatorname{dist}(\boldsymbol{U}, S_r)^2 \le -\rho\operatorname{tr}[\boldsymbol{U}^t P_{S_r}(\boldsymbol{U}_k)] + c_k$$

up to an irrelevant constant c_k since the squared Frobenius norm satisfies the relation $\|\boldsymbol{U}^t\boldsymbol{U}\|_F^2 = q$ on the Stiefel manifold. It now follows that $f(\boldsymbol{U})$ is majorized by

$$\frac{1}{2}\|\boldsymbol{U} - \boldsymbol{S}\boldsymbol{U}_k - \rho P_{S_r}(\boldsymbol{U}_k)\|_F^2$$

up to an irrelevant constant. Accordingly, the Stiefel projection

$$\boldsymbol{U}_{k+1} = P_{M_q}[\boldsymbol{S}\boldsymbol{U}_k + \rho P_{S_r}(\boldsymbol{U}_k)]$$

provides the next MM iterate.

Figures 1 and 2 compare the proximal distance algorithm to the SPC function from the R package `PMA` (Witten et al., 2009). The breast cancer data from `PMA` provide the data matrix $\boldsymbol{X}$. The data consist of $p = 19,672$ RNA measurements on $n = 89$ patients. The two figures show computation times and the proportion of variance explained (PVE) by the $p \times q$ loading matrix $\boldsymbol{U}$. For sparse PCA, PVE is defined as $\operatorname{tr}(\boldsymbol{X}_q^t\boldsymbol{X}_q)/\operatorname{tr}(\boldsymbol{X}^t\boldsymbol{X})$, where $\boldsymbol{X}_q = \boldsymbol{X}\boldsymbol{U}(\boldsymbol{U}^t\boldsymbol{U})^{-1}\boldsymbol{U}^t$ (Shen and Huang, 2008). When the loading vectors of $\boldsymbol{U}$ are orthogonal, this criterion reduces to the familiar definition $\operatorname{tr}(\boldsymbol{U}^t\boldsymbol{X}^t\boldsymbol{X}\boldsymbol{U})/\operatorname{tr}(\boldsymbol{X}^t\boldsymbol{X})$ of PVE for ordinary PCA. The proximal distance algorithm enforces either matrix-wise or column-wise sparsity. In contrast, SPC enforces only column-wise sparsity via the constraint $\|\boldsymbol{u}_i\|_1 \le c$ for each column $\boldsymbol{u}_i$ of $\boldsymbol{U}$. We take $c = 8$. The number of nonzeroes per loading vector output by SPC dictates the sparsity level for the column-wise version of the proximal distance algorithm. Summing these counts across all columns dictates the sparsity level for the matrix version of the proximal distance algorithm.

Figures 1 and 2 demonstrate the superior PVE and computational speed of both proximal distance algorithms versus SPC. The type of projection does not appear to affect the computational performance of the proximal distance algorithm, as both versions scale equally well with q. However, the matrix projection, which permits the algorithm to more freely assign nonzeroes to the loadings, attains better PVE than the more restrictive column-wise projection. For both variants of the proximal distance algorithm, Nesterov acceleration improves both fitting accuracy and computational speed, especially as the number of PCs q increases.

6. Discussion

The proximal distance algorithm applies to a host of problems. In addition to the linear and quadratic programming examples considered here, our previous paper (Lange and Keys,

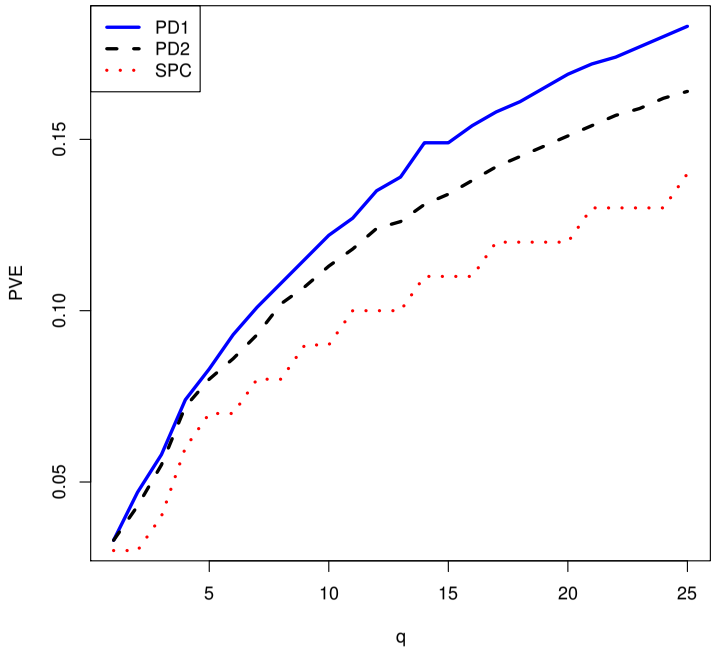


Figure 1: Proportion of variance explained by q PCs for each algorithm. Here PD1 is the accelerated proximal distance algorithm enforcing matrix sparsity, PD2 is the accelerated proximal distance algorithm enforcing column-wise sparsity, and SPC is the orthogonal sparse PCA method from `PMA`.

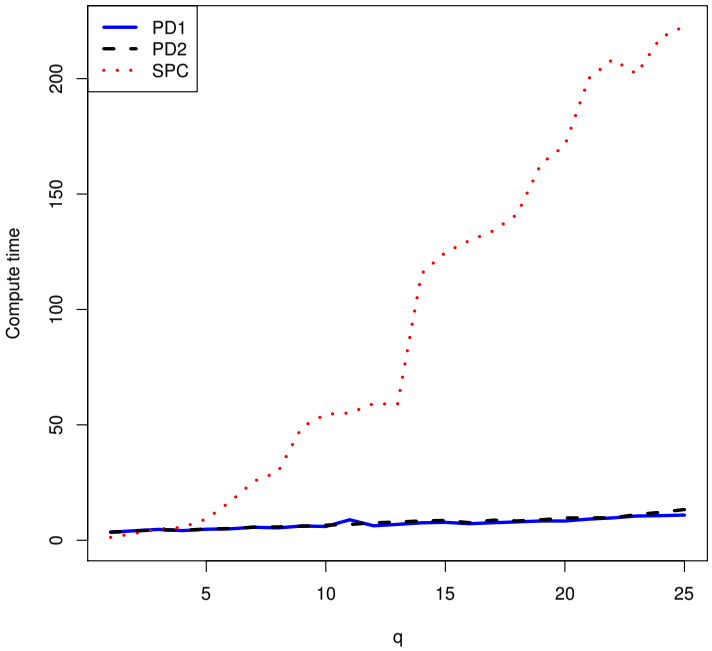


Figure 2: Computation times for q PCs for each algorithm. Here PD1 is the accelerated proximal distance algorithm enforcing matrix sparsity, PD2 is the accelerated proximal distance algorithm enforcing column-wise sparsity, and SPC is the orthogonal sparse PCA method from `PMA`.

2014) derives and tests algorithms for binary piecewise-linear programming, ℓ_0 regression, matrix completion (Cai et al., 2010; Candès and Tao, 2010; Chen et al., 2012; Mazumder et al., 2010), and sparse precision matrix estimation (Friedman et al., 2008). Other potential applications immediately come to mind. An integer linear program in standard form can be expressed as minimizing $\boldsymbol{c}^t\boldsymbol{x}$ subject to $\boldsymbol{A}\boldsymbol{x} + \boldsymbol{s} = \boldsymbol{b}$, $\boldsymbol{s} \geq \boldsymbol{0}$, and $\boldsymbol{x} \in \mathbb{Z}^p$. The latter two constraints can be combined in a single constraint for which projection is trivial. The affine constraints should be folded into the domain of the objective. Integer programming is NP hard, so that the proximal distance algorithm just sketched is merely heuristic. Integer linear programming includes traditional NP hard problems such as the traveling salesman problem, the vertex cover problem, set packing, and Boolean satisfiability. It will be interesting to see if the proximal distance principle is competitive in meeting these challenges. Our experience with the closest lattice point problem (Agrell et al., 2002) and the eight queens problem suggests that the proximal distance algorithm can be too greedy for hard combinatorial problems. The nonconvex problems solved in this paper are in some vague sense easy combinatorial problems.

The behavior of a proximal distance algorithm depends critically on a sensible tuning schedule for increasing ρ. Starting ρ too high puts too much stress on satisfying the constraints. Incrementing ρ too quickly causes the algorithm to veer off the solution path guaranteed by the penalty method. Given the chance of roundoff error even with double precision arithmetic, it is unwise to take ρ all the way to ∞. Trial and error can help in deciding whether a given class of problems will benefit from an aggressive update schedule and strict or loose convergence criteria. In problems with little curvature such as linear programming, more conservative updates are probably prudent. The linear programming, closest kinship matrix, and SPCA problems document the value of folding constraints into the domain of the loss. In the same spirit it is wise to minimize the number of constraints. A single penalty for projecting onto the intersection of two constraint sets is almost always preferable to the sum of two penalties for their separate projections. Exceptions to this rule obviously occur when projection onto the intersection is intractable. The integer linear programming problem mentioned previously illustrates these ideas.

Our earlier proximal distance algorithms ignored acceleration. In many cases the solutions produced had very low accuracy. The realization that convex proximal distance algorithms can be phrased as proximal gradient algorithms convinced us to try Nesterov acceleration. We now do this routinely on the subproblems with ρ fixed. This typically forces tighter path following and a reduction in overall computing times. Our examples generally bear out the contention that Nesterov acceleration is useful in nonconvex problems (Ghadimi and Lan, 2015). It is noteworthy that the value of acceleration often lies in improving the quality of a solution as much as it does in increasing the rate of convergence. Of course, acceleration cannot prevent convergence to an inferior local minimum.

On both convex and nonconvex problems, proximal distance algorithms enjoy global convergence guarantees. On nonconvex problems, one must confine attention to subanalytic sets and subanalytic functions. This minor restriction is not a handicap in practice. Determining local convergence rates is a more vexing issue. For convex problems, we review existing theory for a fixed penalty constant ρ. The classical results buttress an $O(\rho k^{-1})$ sublinear rate for general convex problems. Better results require restrictive smoothness assumptions on both the objective function and the constraint sets. For instance, when $f(\boldsymbol{x})$ is L-smooth

and strongly convex, linear convergence can be demonstrated. When $f(\boldsymbol{x})$ equals a difference of convex functions, proximal distance algorithms reduce to concave-convex programming. Le Thi et al. (2009) attack convergence in this setting.

We hope readers will sense the potential of the proximal distance principle. This simple idea offers insight into many existing algorithms and a straightforward path in devising new ones. Effective proximal and projection operators usually spell the difference between success and failure. The number and variety of such operators is expanding quickly as the field of optimization relinquishes it fixation on convexity. The current paper research leaves many open questions about tuning schedules, rates of convergence, and acceleration in the face of nonconvexity. We welcome the contributions of other mathematical scientists in unraveling these mysteries and in inventing new proximal distance algorithms.

Acknowledgments

We thank Joong-Ho Won for many insightful discussions. In partiular, he pointed out the utility of the least squares criterion (6). Hua Zhou and Kenneth Lange were supported by grants from the National Human Genome Research Institute (HG006139) and the National Institute of General Medical Sciences (GM053275). Kevin Keys was supported by a National Science Foundation Graduate Research Fellowship (DGE-0707424), a Predoctoral Training Grant (HG002536) from the National Human Genome Research Institute, a National Heart, Lung, Blood Institute grant (R01HL135156), the UCSF Bakar Computational Health Sciences Institute, the Gordon and Betty Moore Foundation grant GBMF3834, and the Alfred P. Sloan Foundation grant 2013-10-27 to UC Berkeley through the Moore-Sloan Data Sciences Environment initiative at the Berkeley Institute for Data Science (BIDS).

Appendix A. Proofs of the Stated Propositions

A.1. Proposition 1

We first observe that the surrogate function $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k)$ is ρ-strongly convex. Consequently, the stationarity condition $\mathbf{0} \in \partial g_\rho(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k)$ implies

$$g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k) \;\geq\; g_\rho(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{x}_{k+1}\|^2 \tag{20}$$

for all $\boldsymbol{x}$. In the notation (9), the difference

$$d_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k) \;=\; g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k) - h_\rho(\boldsymbol{x}) \;=\; \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{y}_k\|^2 - \frac{\rho}{2}\sum_{i=1}^m \alpha_i \operatorname{dist}(\boldsymbol{x}, C_i)^2$$

is ρ-smooth because

$$\nabla d_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k) \;=\; \rho(\boldsymbol{x} - \boldsymbol{y}_k) - \rho\sum_{i=1}^m \alpha_i[\boldsymbol{x} - P_{C_i}(\boldsymbol{x})] \;=\; \rho\sum_{i=1}^m \alpha_i P_{C_i}(\boldsymbol{x}) - \rho\boldsymbol{y}_k.$$

The tangency conditions $d_\rho(\boldsymbol{x}_k \mid \boldsymbol{x}_k) = 0$ and $\nabla d_\rho(\boldsymbol{x}_k \mid \boldsymbol{x}_k) = \mathbf{0}$ therefore yield

$$d_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k) \;\leq\; d_\rho(\boldsymbol{x}_k \mid \boldsymbol{x}_k) + \nabla d_\rho(\boldsymbol{x}_k)^t(\boldsymbol{x} - \boldsymbol{x}_k) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{x}_k\|^2 \;=\; \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{x}_k\|^2 \tag{21}$$

for all $\boldsymbol{x}$. Combining inequalities (20) and (21) gives

$$\begin{aligned} h_\rho(\boldsymbol{x}_{k+1}) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{x}_{k+1}\|^2 &\leq g_\rho(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{x}_{k+1}\|^2 \\ &\leq g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k) \\ &= h_\rho(\boldsymbol{x}) + d_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k) \\ &\leq h_\rho(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{x}_k\|^2. \end{aligned}$$

Adding the result

$$h_\rho(\boldsymbol{x}_{k+1}) - h_\rho(\boldsymbol{x}) \;\leq\; \frac{\rho}{2}\Big(\|\boldsymbol{x} - \boldsymbol{x}_k\|^2 - \|\boldsymbol{x} - \boldsymbol{x}_{k+1}\|^2\Big)$$

over k and invoking the descent property $h_\rho(\boldsymbol{x}_{k+1}) \leq h_\rho(\boldsymbol{x}_k)$ produce the error bound

$$h_\rho(\boldsymbol{x}_{k+1}) - h_\rho(\boldsymbol{x}) \;\leq\; \frac{\rho}{2(k+1)}\Big(\|\boldsymbol{x} - \boldsymbol{x}_0\|^2 - \|\boldsymbol{x} - \boldsymbol{x}_{k+1}\|^2\Big) \;\leq\; \frac{\rho}{2(k+1)}\|\boldsymbol{x} - \boldsymbol{x}_0\|^2.$$

Setting $\boldsymbol{x}$ equal to a minimal point $\boldsymbol{z}$ gives the stated result. ∎

A.2. Proposition 2

The existence and uniqueness of $\boldsymbol{y}$ are obvious. The remainder of the proof hinges on the assumptions that $h_\rho(\boldsymbol{x})$ is μ-strongly convex and the surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k)$ is $L + \rho$ smooth. The latter assumption yields

$$\begin{aligned} h_\rho(\boldsymbol{x}) - h_\rho(\boldsymbol{y}) &\leq g_\rho(\boldsymbol{x} \mid \boldsymbol{y}) - g_\rho(\boldsymbol{y} \mid \boldsymbol{y}) \\ &\leq \nabla g_\rho(\boldsymbol{y} \mid \boldsymbol{y})^t(\boldsymbol{x} - \boldsymbol{y}) + \frac{L+\rho}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2 \\ &= \frac{L+\rho}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2. \end{aligned} \tag{22}$$

The strong convexity condition $h_\rho(\boldsymbol{y}) - h_\rho(\boldsymbol{x}) \geq \nabla h_\rho(\boldsymbol{x})^t(\boldsymbol{y} - \boldsymbol{x}) + \frac{\mu}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2$ implies

$$\|\nabla h_\rho(\boldsymbol{x})\| \cdot \|\boldsymbol{y} - \boldsymbol{x}\| \;\geq\; -\nabla h_\rho(\boldsymbol{x})^t(\boldsymbol{y} - \boldsymbol{x}) \;\geq\; \frac{\mu}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2.$$

It follows that $\|\nabla h_\rho(\boldsymbol{x})\| \geq \frac{\mu}{2}\|\boldsymbol{x} - \boldsymbol{y}\|$. This last inequality and inequality (22) produce the Polyak-Łojasiewicz bound

$$\frac{1}{2}\|\nabla h_\rho(\boldsymbol{x})\|^2 \;\geq\; \frac{\mu^2}{2(L+\rho)}[h_\rho(\boldsymbol{x}) - h_\rho(\boldsymbol{y})].$$

Taking $\boldsymbol{x} = \boldsymbol{x}_k - \frac{1}{L+\rho}\nabla g_\rho(\boldsymbol{x}_k \mid \boldsymbol{x}_k) = \boldsymbol{x}_k - \frac{1}{L+\rho}\nabla h_\rho(\boldsymbol{x}_k)$, the Polyak-Łojasiewicz bound gives

$$\begin{aligned} h_\rho(\boldsymbol{x}_{k+1}) - h_\rho(\boldsymbol{x}_k) &\leq g_\rho(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k) - g_\rho(\boldsymbol{x}_k \mid \boldsymbol{x}_k) \\ &\leq g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_k) - g_\rho(\boldsymbol{x}_k \mid \boldsymbol{x}_k) \\ &\leq -\frac{1}{L+\rho}\nabla g_\rho(\boldsymbol{x}_k \mid \boldsymbol{x}_k)^t\nabla h_\rho(\boldsymbol{x}_k) + \frac{1}{2(L+\rho)}\|\nabla h_\rho(\boldsymbol{x}_k)\|^2 \\ &= -\frac{1}{2(L+\rho)}\|\nabla h_\rho(\boldsymbol{x}_k)\|^2 \\ &\leq -\frac{\mu^2}{2(L+\rho)^2}[h_\rho(\boldsymbol{x}_k) - h_\rho(\boldsymbol{y})]. \end{aligned}$$

Rearranging this inequality yields

$$h_\rho(\boldsymbol{x}_{k+1}) - h_\rho(\boldsymbol{y}) \;\le\; \Big[1 - \frac{\mu^2}{2(L+\rho)^2}\Big][h_\rho(\boldsymbol{x}_k) - h_\rho(\boldsymbol{y})],$$

which can be iterated to give the stated bound. ∎

A.3. Proposition 3

Consider first the proximal distance iterates. The inequality

$$f(\boldsymbol{x}_k) + \frac{\rho_k}{2}\operatorname{dist}(\boldsymbol{x}_k, S)^2 \;\le\; f(\boldsymbol{x}_k) + \frac{\rho_k}{2}\|\boldsymbol{x}_k - P_S(\boldsymbol{x}_{k-1})\|^2 \;\le\; f[P_S(\boldsymbol{x}_{k-1})] \;\le\; \sup_{\boldsymbol{x}\in S} f(\boldsymbol{x})$$

plus the coerciveness of $f(\boldsymbol{x})$ imply that $\boldsymbol{x}_k$ is a bounded sequence. The claimed bound now holds for c equal to the finite supremum of the sequence $2[\sup_{\boldsymbol{x}\in S} f(\boldsymbol{x}) - f(\boldsymbol{x}_k)]$. If in addition, $f(\boldsymbol{x})$ is continuously differentiable, then the stationarity equation

$$\boldsymbol{0} \;=\; \nabla f(\boldsymbol{x}_k) + \rho_k[\boldsymbol{x}_k - P_S(\boldsymbol{x}_{k-1})]$$

and the Cauchy-Schwarz inequality give

$$\rho_k\|\boldsymbol{x}_k - P_S(\boldsymbol{x}_{k-1})\|^2 \;=\; -\nabla f(\boldsymbol{x}_k)^t[\boldsymbol{x}_k - P_S(\boldsymbol{x}_{k-1})] \;\le\; \|\nabla f(\boldsymbol{x}_k)\| \cdot \|\boldsymbol{x}_k - P_S(\boldsymbol{x}_{k-1})\|.$$

Dividing this by $\|\boldsymbol{x}_k - P_S(\boldsymbol{x}_{k-1})\|$ and squaring further yield

$$\rho_k^2 \operatorname{dist}(\boldsymbol{x}_k, S)^2 \;\le\; \rho_k^2\|\boldsymbol{x}_k - P_S(\boldsymbol{x}_{k-1})\|^2 \;\le\; \|\nabla f(\boldsymbol{x}_k)\|^2.$$

Taking $d = \sup_k \|\nabla f(\boldsymbol{x}_k)\|^2$ over the bounded sequence $\boldsymbol{x}_k$ completes the proof.

For the penalty method iterates, the bound

$$f(\boldsymbol{y}_k) + \frac{\rho_k}{2}\operatorname{dist}(\boldsymbol{y}_k, S)^2 \;\le\; f(\boldsymbol{y})$$

is valid by definition, where $\boldsymbol{y}$ attains the constrained minimum. Thus, coerciveness implies that the sequence $\boldsymbol{y}_k$ is bounded. When $f(\boldsymbol{x})$ is continuously differentiable, the proof of the second claim also applies if we substitute $\boldsymbol{y}_k$ for $\boldsymbol{x}_k$ and $P_S(\boldsymbol{y}_k)$ for $P_S(\boldsymbol{x}_{k-1})$. ∎

A.4. Proposition 4

Because the function $f(\boldsymbol{x}) + \frac{\rho_k}{2}\operatorname{dist}(\boldsymbol{x}, S)^2$ is convex and has the value $f(\boldsymbol{y})$ and gradient $\nabla f(\boldsymbol{y})$ at a constrained minimum $\boldsymbol{y}$, the supporting hyperplane principle says

$$\begin{aligned} f(\boldsymbol{y}_k) + \frac{\rho_k}{2}\operatorname{dist}(\boldsymbol{y}_k, S)^2 &\ge f(\boldsymbol{y}) + \nabla f(\boldsymbol{y})^t(\boldsymbol{y}_k - \boldsymbol{y}) \\ &= f(\boldsymbol{y}) + \nabla f(\boldsymbol{y})^t[P_S(\boldsymbol{y}_k) - \boldsymbol{y}] + \nabla f(\boldsymbol{y})^t[\boldsymbol{y}_k - P_S(\boldsymbol{y}_k)]. \end{aligned}$$

The first-order optimality condition $\nabla f(\boldsymbol{y})^t[P_S(\boldsymbol{y}_k) - \boldsymbol{y}] \ge 0$ holds given $\boldsymbol{y}$ is a constrained minimum. Hence, the Cauchy-Schwarz inequality and Proposition 3 imply

$$\begin{aligned} f(\boldsymbol{y}) - f(\boldsymbol{y}_k) &\le \frac{\rho_k}{2}\operatorname{dist}(\boldsymbol{y}_k, S)^2 - \nabla f(\boldsymbol{y})^t[\boldsymbol{y}_k - P_S(\boldsymbol{y}_k)] \\ &\le \frac{\rho_k}{2}\operatorname{dist}(\boldsymbol{y}_k, S)^2 + \|\nabla f(\boldsymbol{y})\| \cdot \operatorname{dist}(\boldsymbol{y}_k, S) \\ &\le \frac{d + 2\sqrt{d}\|\nabla f(\boldsymbol{y})\|}{2\rho_k}. \end{aligned}$$

∎

A.5. Proposition 5

Let $\boldsymbol{x}_k$ converge to $\boldsymbol{x}_\infty$ and $\boldsymbol{y}_k \in P_S(\boldsymbol{x}_k)$ converge to $\boldsymbol{y}_\infty$. For an arbitrary $\boldsymbol{y} \in S$, taking limits in the inequality $\|\boldsymbol{x}_k - \boldsymbol{y}_k\| \le \|\boldsymbol{x}_k - \boldsymbol{y}\|$ yields $\|\boldsymbol{x}_\infty - \boldsymbol{y}_\infty\| \le \|\boldsymbol{x}_\infty - \boldsymbol{y}\|$; consequently, $\boldsymbol{y}_\infty \in P_S(\boldsymbol{x}_\infty)$. To prove the second assertion, take $\boldsymbol{y}_k \in P_S(\boldsymbol{x}_k)$ and observe that

$$\begin{aligned} \|\boldsymbol{y}_k\| &\le \|\boldsymbol{x}_k - \boldsymbol{y}_k\| + \|\boldsymbol{x}_k\| \\ &\le \|\boldsymbol{x}_k - \boldsymbol{y}_1\| + \|\boldsymbol{x}_k\| \\ &\le \|\boldsymbol{x}_k - \boldsymbol{x}_1\| + \|\boldsymbol{x}_1 - \boldsymbol{y}_1\| + \|\boldsymbol{x}_k\| \\ &\le \|\boldsymbol{x}_k\| + \|\boldsymbol{x}_1\| + \operatorname{dist}(\boldsymbol{x}_1, S) + \|\boldsymbol{x}_k\|, \end{aligned}$$

which is bounded above by the constant $\operatorname{dist}(\boldsymbol{x}_1, S) + 3\sup_{m\ge 1}\|\boldsymbol{x}_m\|$. ∎

A.6. Proposition 6

In fact, a much stronger result holds. Since the function $s(\boldsymbol{x})$ of equation (8) is convex and finite, Alexandrov's theorem (Niculescu and Persson, 2006) implies that it is almost everywhere twice differentiable. In view of the identities $\frac{1}{2}\operatorname{dist}(\boldsymbol{x}, S)^2 = \frac{1}{2}\|\boldsymbol{x}\|^2 - s(\boldsymbol{x})$ and $\boldsymbol{x} - P_S(\boldsymbol{x}) = \nabla\frac{1}{2}\operatorname{dist}(\boldsymbol{x}, S)^2$ where $P_S(\boldsymbol{x})$ is single valued, it follows that $P_S(\boldsymbol{x}) = \nabla s(\boldsymbol{x})$ is almost everywhere differentiable. ∎

A.7. Proposition 8

See the discussion just prior to the statement of the proposition. ∎

A.8. Proposition 9

The strong-convexity inequality

$$g_\rho(\boldsymbol{x}_k \mid \boldsymbol{x}_k) \;\ge\; g_\rho(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k) + \frac{\mu}{2}\|\boldsymbol{x}_k - \boldsymbol{x}_{k+1}\|^2$$

and the tangency and domination properties of the algorithm imply

$$h_\rho(\boldsymbol{x}_k) - h_\rho(\boldsymbol{x}_{k+1}) \;\ge\; \frac{\mu}{2}\|\boldsymbol{x}_k - \boldsymbol{x}_{k+1}\|^2. \tag{23}$$

Since the difference in function values tends to 0, this validates the stated limit. The remaining assertions follow from Propositions 7.3.3 and 7.3.5 of (Lange, 2016). ∎

A.9. Proposition 10

Let the subsequence x_{k_m} of the MM sequence $\boldsymbol{x}_{k+1} \in M(\boldsymbol{x}_k)$ converge to $\boldsymbol{z}$. By passing to a subsubsequence if necessary, we may suppose that $\boldsymbol{x}_{k_m+1}$ converges to $\boldsymbol{y}$. Owing to our closedness assumption, $\boldsymbol{y} \in M(\boldsymbol{z})$. Given that $h(\boldsymbol{y}) = h(\boldsymbol{z})$, it is obvious that $\boldsymbol{z}$ also minimizes $g(\boldsymbol{x} \mid \boldsymbol{z})$ and that $\boldsymbol{0} = \nabla g(\boldsymbol{z} \mid \boldsymbol{z})$. Since the difference $\Delta(\boldsymbol{x} \mid \boldsymbol{z}) = g(\boldsymbol{x} \mid \boldsymbol{z}) - h(\boldsymbol{x})$ achieves its minimum at $\boldsymbol{x} = \boldsymbol{z}$, the Fréchet subdifferential $\partial^F\Delta(\boldsymbol{x} \mid \boldsymbol{z})$ satisfies

$$\boldsymbol{0} \;\in\; \partial^F\Delta(\boldsymbol{z} \mid \boldsymbol{z}) \;=\; \nabla g(\boldsymbol{z} \mid \boldsymbol{z}) + \partial^F(-h)(\boldsymbol{z}).$$

It follows that $\boldsymbol{0} \in \partial^F(-h)(\boldsymbol{z})$. ∎

A.10. Proposition 11

Because $\Delta(\boldsymbol{x} \mid \boldsymbol{y}) = g(\boldsymbol{x} \mid \boldsymbol{y}) - h(\boldsymbol{x})$ achieves its minimum at $\boldsymbol{x} = \boldsymbol{y}$, the Fréchet subdifferential $\partial^F \Delta(\boldsymbol{x} \mid \boldsymbol{y})$ satisfies

$$\mathbf{0} \;\in\; \partial^F \Delta(\boldsymbol{y} \mid \boldsymbol{y}) \;=\; \nabla g(\boldsymbol{y} \mid \boldsymbol{y}) + \partial^F(-h)(\boldsymbol{y}).$$

It follows that $-\nabla g(\boldsymbol{y} \mid \boldsymbol{y}) \in \partial^F(-h)(\boldsymbol{y})$. Furthermore, by assumption

$$\|\nabla g(\boldsymbol{a} \mid \boldsymbol{x}_k) - \nabla g(\boldsymbol{b} \mid \boldsymbol{x}_k)\| \;\leq\; L\|\boldsymbol{a} - \boldsymbol{b}\|$$

for all relevant $\boldsymbol{a}$ and $\boldsymbol{b}$ and $\boldsymbol{x}_k$. In particular, because $\nabla g(\boldsymbol{x}_{k+1} \mid \boldsymbol{x}_k) = \mathbf{0}$, we have

$$\|\nabla g(\boldsymbol{x}_k \mid \boldsymbol{x}_k)\| \;\leq\; L\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|. \tag{24}$$

Let W denote the set of limit points. The objective $h(\boldsymbol{x})$ is constant on W with value $\bar{h} = \lim_{k\to\infty} h(\boldsymbol{x}_k)$. According to the Łojasiewicz inequality applied for the subanalytic function $\bar{h} - h(\boldsymbol{x})$, for each $\boldsymbol{z} \in W$ there exists an open ball $B_{r(\boldsymbol{z})}(\boldsymbol{z})$ of radius $r(\boldsymbol{z})$ around $\boldsymbol{z}$ and an exponent $\theta(\boldsymbol{z}) \in [0, 1)$ such that

$$|h(\boldsymbol{u}) - h(\boldsymbol{z})|^{\theta(\boldsymbol{z})} \;=\; |\bar{h} - h(\boldsymbol{u}) - \bar{h} + \bar{h}|^{\theta(\boldsymbol{z})} \;\leq\; c(\boldsymbol{z})\|\boldsymbol{v}\|$$

for all $\boldsymbol{u} \in B_{r(\boldsymbol{z})}(\boldsymbol{z})$ and all $\boldsymbol{v} \in \partial^F(\bar{h} - h)(\boldsymbol{u}) = \partial^F(-h)(\boldsymbol{u})$. We will apply this inequality to $\boldsymbol{u} = \boldsymbol{x}_k$ and $\boldsymbol{v} = -\nabla g(\boldsymbol{x}_k \mid \boldsymbol{x}_k)$. In so doing, we would like to assume that the exponent $\theta(\boldsymbol{z})$ and constant $c(\boldsymbol{z})$ do not depend on $\boldsymbol{z}$. With this end in mind, cover the compact set W by a finite number of balls $B_{r(\boldsymbol{z}_i)}(\boldsymbol{z}_i)$ and take $\theta = \max_i \theta(\boldsymbol{z}_i) < 1$ and $c = \max_i c(\boldsymbol{z}_i)$. For a sufficiently large K, every $\boldsymbol{x}_k$ with $k \geq K$ falls within one of these balls and satisfies $|\bar{h} - h(\boldsymbol{x}_k)| < 1$. Without loss of generality assume $K = 0$. The Łojasiewicz inequality reads

$$|\bar{h} - h(\boldsymbol{x}_k)|^\theta \;\leq\; c\|\nabla g(\boldsymbol{x}_k \mid \boldsymbol{x}_k)\|. \tag{25}$$

In combination with the concavity of the function $t^{1-\theta}$ on $[0, \infty)$, inequalities (23), (24), and (25) imply

$$\begin{aligned}
[h(\boldsymbol{x}_k) - \bar{h}]^{1-\theta} - [h(\boldsymbol{x}_{k+1}) - \bar{h}]^{1-\theta} &\geq \frac{1-\theta}{[h(\boldsymbol{x}_k) - \bar{h}]^\theta}[h(\boldsymbol{x}_k) - h(\boldsymbol{x}_{k+1})] \\
&\geq \frac{1-\theta}{c\|\nabla g(\boldsymbol{x}_k \mid \boldsymbol{x}_k)\|}\frac{\mu}{2}\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|^2 \\
&\geq \frac{(1-\theta)\mu}{2cL}\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|.
\end{aligned}$$

Rearranging this inequality and summing over k yield

$$\sum_{n=0}^{\infty} \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\| \;\leq\; \frac{2cL}{(1-\theta)\mu}[h(\boldsymbol{x}_0) - \bar{h}]^{1-\theta}$$

Thus, the sequence $\boldsymbol{x}_k$ is a fast Cauchy sequence and converges to a unique limit in W. ■

References

Erik Agrell, Thomas Eriksson, Alexander Vardy, and Kenneth Zeger. Closest point search in lattices. *IEEE Transactions on Information Theory*, 48(8):2201–2214, 2002.

Farid Alizadeh and Donald Goldfarb. Second-order cone programming. *Mathematical Programming*, 95:3–51, 2003.

Hédy Attouch, Jérôme Bolte, Patrick Redont, and Antoine Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Łojasiewicz inequality. *Mathematics of Operations Research*, 35(2):438–457, 2010.

Heinz H Bauschke and Patrick L Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2011.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Edward J Beltrami. *An Algorithmic Approach to Nonlinear Analysis and Optimization*. Academic Press, 1970.

Alberto Bemporad. A numerically stable solver for positive semidefinite quadratic programs based on nonnegative least squares. *IEEE Transactions on Automatic Control*, 63(2): 525–531, 2018.

Abraham Berman and Robert J Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. Classics in Applied Mathematics. SIAM, 1994.

Quentin Berthet and Philippe Rigollet. Complexity theoretic lower bounds for sparse principal component detection. In *Conference on Learning Theory*, pages 1046–1066, 2013a.

Quentin Berthet and Philippe Rigollet. Optimal detection of sparse principal components in high dimension. *The Annals of Statistics*, 41(4):1780–1815, 2013b.

Edward Bierstone and Pierre D Milman. Semianalytic and subanalytic sets. *Publications Mathématiques de l'Institut des Hautes Études Scientifiques*, 67(1):5–42, 1988.

Jacek Bochnak, Michel Coste, and Marie-Françoise Roy. *Real Algebraic Geometry*, volume 36. Springer Science & Business Media, 2013.

Jérôme Bolte, Aris Daniilidis, and Adrian Lewis. The Łojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems. *SIAM Journal on Optimization*, 17(4):1205–1223, 2007.

Jonathan M Borwein and Adrian S Lewis. *Convex Analysis and Nonlinear Optimization: Theory and Examples*. CMS Books in Mathematics. Springer, New York, 2nd edition, 2006.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2009.

James P Boyle and Richard L Dykstra. A method for finding projections onto the intersection of convex sets in Hilbert spaces. In *Advances in Order Restricted Statistical Inference*, pages 28–47. Springer, 1986.

Florentina Bunea, Alexandre B Tsybakov, Marten H Wegkamp, and Adrian Barbu. Spades and mixture models. *The Annals of Statistics*, 38(4):2525–2558, 2010.

Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20:1956–1982, 2010.

Emmanuel J Candès and Terence Tao. The power of convex relaxation: near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56:2053–2080, 2010.

Caihua Chen, Bingsheng He, and Xiaoming Yuan. Matrix completion via an alternating direction method. *IMA Journal of Numerical Analysis*, 32:227–245, 2012.

Eric C Chi, Hua Zhou, and Kenneth Lange. Distance majorization and its applications. *Mathematical Programming Series A*, 146:409–436, 2014.

Patrick L Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer, 2011.

Richard Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bulletin of the American Mathematical Society*, 49:1–23, 1943.

Ying Cui, Jong-Shi Pang, and Bodhisattva Sen. Composite difference-max programs for modern statistical estimation problems. *SIAM Journal on Optimization*, 28(4):3344–3374, 2018.

Alexandre D'Aspremont, Laurent El Ghaoui, Michael I Jordan, and Gert R G Lanckriet. A direct formulation for sparse PCA using semidefinite programming. *SIAM Review*, 49(3): 434–448, 2007.

Iain Dunning, Joey Huchette, and Miles Lubin. JuMP: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

Ky Fan. On a theorem of Weyl concerning eigenvalues of linear transformations I. *Proceedings of the National Academy of Sciences of the United States of America*, 35:652–655, 1949.

David Chin-Lung Fong and Michael Saunders. Lsmr: An iterative algorithm for sparse least-squares problems. *SIAM Journal on Scientific Computing*, 33(5):2950–2971, 2011.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9:432–441, 2008. ISSN 1468-4357.

Saeed Ghadimi and Guanghui Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1):59–99, 2015.

Gene H Golub and Charles F Van Loan. *Matrix Computations*. JHU Press, 3rd edition, 2012.

Marshall Hall and Morris Newman. Copositive and completely positive quadratic forms. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 59, pages 329–339. Cambridge Univ Press, 1963.

Rob Heylen, Dževdet Burazerović, and Paul Scheunders. Fully constrained least squares spectral unmixing by simplex projection. *IEEE Transactions on Geoscience and Remote Sensing*, 49(11):4112–4122, Nov 2011.

Nicholas J Higham. Computing the nearest correlation matrix - a problem from finance. *IMA Journal of Numerical Analysis*, 22(3):329–343, 2002.

Jean-Baptiste Hiriart-Urruty and Alberto Seeger. A variational approach to copositive matrices. *SIAM Review*, 52:593–629, 2010.

Harold Hotelling. Analysis of a complex of statistical variables into principle components. *Journal of Educational Psychology*, 24:417–441, 1933.

David R Hunter and Kenneth Lange. A tutorial on MM algorithms. *American Statistician*, 58:30–37, 2004.

Charles R Johnson and Robert Reams. Constructing copositive matrices from interior matrices. *Electronic Journal of Linear Algebra*, 17:9–20, 2008.

Iain M Johnstone and Arthur Yu Lu. On consistency and sparsity for principal components analysis in high dimensions. *Journal of the American Statistical Association*, 104(486): 682–693, 2009.

Michel Journée, Yurii Nesterov, Peter Richtárik, and Rodolphe Sepulchre. Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research*, 11:517–553, 2010.

Yangyang Kang, Zhihua Zhang, and Wu-Jun Li. On the global convergence of majorization minimization algorithms for nonconvex optimization problems. *arXiv preprint arXiv:1504.07791*, 2015.

Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 795–811. Springer, 2016.

Nirmal Keshava. A survey of spectral unmixing algorithms. *Lincoln Laboratory Journal*, 14 (1):55–78, 2003.

Henk AL Kiers. Majorization as a tool for optimizing a class of matrix functions. *Psychometrika*, 55:417–428, 1990.

Henk AL Kiers and Jos MF ten Berge. Minimization of a class of matrix trace functions by means of refined majorization. *Psychometrika*, 57:371–382, 1992.

Mark Aleksandrovich Krasnosel'skii. Two remarks on the method of successive approximations. *Uspekhi Matematicheskikh Nauk*, 10(1):123–127, 1955.

Alexander Ya Kruger. On Fréchet subdifferentials. *Journal of Mathematical Sciences*, 116 (3):3325–3358, 2003.

Louis Landweber. An iteration formula for Fredholm integral equations of the first kind. *American Journal of Mathematics*, 73(3):615–624, 1951.

Kenneth Lange. *Optimization*. Springer, New York, 2nd edition, 2010.

Kenneth Lange. *MM Optimization Algorithms*. SIAM, 2016.

Kenneth Lange and Kevin L Keys. The proximal distance algorithm. In *Proceedings of the 2014 International Congress of Mathematicians*, pages 95–116. Kyung Moon, August 2014.

Hoai An Le Thi, Van Ngai Huynh, and Tao Pham Dinh. Convergence analysis of DC algorithm for DC programming with subanalytic data. *Annals of Operation Research Technical Report, LMI, INSA-Rouen*, 2009.

Huan Li, Cong Fang, and Zhouchen Lin. Convergence rates analysis of the quadratic penalty method and its applications to decentralized distributed optimization. *arXiv preprint arXiv:1711.10802*, 2017.

Miguel Sousa Lobo, Lieven Vandenberghe, Stephen Boyd, and Hervé Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

Miles Lubin and Iain Dunning. Computing in operations research using Julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.

David G Luenberger and Yinyu Ye. *Linear and Nonlinear Programming*, volume 2. Springer, 1984.

Julien Mairal. Optimization with first-order surrogate functions. In *International Conference on Machine Learning*, pages 783–791, 2013.

W Robert Mann. Mean value methods in iteration. *Proceedings of the American Mathematical Society*, 4(3):506–510, 1953.

Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research*, 11: 2287–2322, 2010.

Jean-Jacques Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *Comptes Rendus de l'Académie des Sciences de Paris A*, 255:2897–2899, 1962.

Katta G Murty and Feng-Tien Yu. *Linear Complementarity, Linear and Nonlinear Programming*. Heldermann Verlag, West Berlin, 1988.

Constantin Niculescu and Lars-Erik Persson. *Convex Functions and their Applications: A Contemporary Approach*. Springer, 2006.

Brendan O'Donoghue, Eric Chu, Neal Parikh, and Stephen Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journel of Optimization Theory and Applications*, pages 1–27, 2016.

Christopher C Paige and Michael A Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software (TOMS)*, 8 (1):43–71, 1982a.

Christopher C Paige and Michael A Saunders. Algorithm 583: LSQR: Sparse linear equations and least squares problems. *ACM Transactions on Mathematical Software (TOMS)*, 8(2): 195–209, 1982b.

Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2013.

Karl Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11):559–572, 1901.

Boris T Polyak. The convergence rate of the penalty function method. *USSR Computational Mathematics and Mathematical Physics*, 11(1):1–12, 1971.

Haipeng Shen and Jianhua Z Huang. Sparse principal component analysis via regularized low rank matrix approximation. *Journal of Multivariate Analysis*, 99:1015–1034, 2008.

Weijie Su, Stephen Boyd, and Emmanuel Candès. A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. In *Advances in Neural Information Processing Systems*, pages 2510–2518, 2014.

Andreas Wächter and Lorenz T Biegler. Line search filter methods for nonlinear programming: Motivation and global convergence. *SIAM Journal on Optimization*, 16(1):1–31, 2005.

Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

Daniela M Witten, Robert Tibshirani, and Trevor Hastie. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, 10(3):515–534, 2009.

Jason Xu, Eric Chi, and Kenneth Lange. Generalized linear model regression under distance-to-set penalties. In *Advances in Neural Information Processing Systems*, pages 1385–1394, 2017.

Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal components analysis. *Journal of Computational and Graphical Statistics*, 15(2):262–282, 2006.

Journal of Machine Learning Research 20 (2019) 1-28 Submitted 12/17; Revised 1/19; Published 3/19

Learnability of Solutions to Conjunctive Queries

Hubie Chen h.chen@dcs.bbk.ac.uk
Birkbeck, University of London,
London WC1E 7HX, United Kingdom

Matthew Valeriote matt@math.mcmaster.ca
Department of Mathematics & Statistics,
McMaster University,
Hamilton, Canada

Editor: Mehryar Mohri

Abstract

The problem of learning the solution space of an unknown formula has been studied in multiple embodiments in computational learning theory. In this article, we study a family of such learning problems; this family contains, for each relational structure, the problem of learning the solution space of an unknown conjunctive query evaluated on the structure. A progression of results aimed to classify the learnability of each of the problems in this family, and thus far a culmination thereof was a positive learnability result generalizing all previous ones. This article completes the classification program towards which this progression of results strived, by presenting a negative learnability result that complements the mentioned positive learnability result. In addition, a further negative learnability result is exhibited, which indicates a dichotomy within the problems to which the first negative result applies. In order to obtain our negative results, we make use of universal-algebraic concepts.

Keywords: concept learning, computational learning theory, dichotomy theorems, reductions, universal algebra

1. Introduction

In this section we provide an overview of the question that we address in this paper and describe some related work as well as our contributions to its resolution.

1.1. Overview

The problem of learning the solution space of an unknown formula has long been of interest in computational learning theory. While the general problem of learning the solution space of even a propositional formula is known to be hard (Kearns and Valiant, 1994; Angluin and Kharitonov, 1995), researchers have considered many restricted versions of formula learning over the years, and have obtained a variety of learnability and non-learnability results (see for example Angluin, 1988; Angluin et al., 1992; Arias and Khardon, 2002; Bshouty et al., 2005; Jackson and Servedio, 2006; Bulatov et al., 2007; Idziak et al., 2010; Bshouty, 2013).

Conjunctive queries are formulas which are considered heavily in database theory and in the theory of constraint satisfaction. They can be defined logically as formulas built from predicate applications, equality of variables, conjunction, and existential quantification. For example, if $\mathbf{G} = (V, E)$ is an irreflexive graph (with vertex set V and edge set E, considered as a binary relation on V), then the conjunctive query

$$\exists y \exists z \left(E(x, y) \wedge E(y, z) \wedge E(z, x) \right)$$

is true for a vertex $x \in V$ if and only if x is part of a 3-cycle of the graph.

The problem of deciding, given a conjunctive query and a *relational structure* (which defines the predicates of the query), whether or not the solution space of the query is non-empty, is a formulation of the *constraint satisfaction problem*, a very general NP-complete problem. One obtains a rich framework of problems, by considering, for each relational structure $\mathbf{B}$, the constraint satisfaction problem, denoted $\mathsf{CSP}(\mathbf{B})$, where the relational structure is fixed as $\mathbf{B}$; the computational aspects of this problem framework are of interest and have been explored in numerous contexts (see for example Creignou et al., 2001; Raghavendra, 2008; Allender et al., 2009; Chen, 2012; Bhattacharyya and Yoshida, 2013; Bulatov, 2013; Chen et al., 2016; Chen and Larose, 2017). Schaefer's celebrated dichotomy theorem (Schaefer, 1978) provides that, for each relational structure $\mathbf{B}$ with a two-element universe, $\mathsf{CSP}(\mathbf{B})$ is either polynomial-time decidable or is NP-complete. A line of research strives to obtain a complexity classification of the constraint satisfaction problem over all relational structures with finite universe; results here include sufficient conditions for tractability (Idziak et al., 2010; Barto and Kozik, 2014), a unifying explanation for known intractability proofs (Bulatov et al., 2005), and—as a culmination thus far—the classification of all such structures whose constraint satisfaction problem is polynomial-time tractable (Bulatov, 2017; Zhuk, 2017).

As a means of systematically exploring the boundary between learnability and non-learnability, an analogous framework has been considered in learning theory: for each relational structure $\mathbf{B}$, we may define a problem $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ wherein the aim is to learn the solution space of an unknown conjunctive query evaluated on $\mathbf{B}$ (refer to Section 2 for formal definitions). As three particular examples, consider the following. Other examples can be found in Barto et al. (2017).

Example 1 *Let $\mathbf{B}_{3\mathrm{SAT}}$ be the relational structure with domain $\{0,1\}$ that has, for each triple $(a,b,c) \in \{0,1\}^3$, the ternary relation $R_{(a,b,c)} = \{0,1\}^3 \setminus \{(a,b,c)\}$. Then the set of solution spaces of the conjunctive queries on $\mathbf{B}_{3\mathrm{SAT}}$ is known to be the set of all Boolean relations.*

Example 2 *Let $\mathbf{B}_{\mathrm{HORN-3SAT}}$ be the relational structure with universe $\{0,1\}$ whose relations consists of the four relations $\{0\}$, $\{1\}$, $R_{(1,1,0)}$ and $R_{(1,1,1)}$. It is readily verified that the solution spaces of conjunctive queries on $\mathbf{B}_{\mathrm{HORN-3SAT}}$ are exactly the solution spaces of conjunctions of propositional* Horn clauses*; these solution spaces can be equivalently characterized as those closed under the pointwise application of the Boolean AND ($\wedge$) operation (Creignou et al., 2001, Lemma 4.8).*

Example 3 *For a finite field $\mathbb{F} = (F; +, \cdot, -, 0, 1)$, let $\mathbf{V}_{\mathbb{F}}$ be the relational structure with universe F and whose relations are the singleton unary relations $\{f\}$, for $f \in F$; the graph of the function $x + y$; and, the graph of $\lambda_f(x) = f \cdot x$, for each $f \in F$. Then the solution spaces of conjunctive queries on $\mathbf{V}_{\mathbb{F}}$ are exactly the affine subspaces of the vector spaces $(\langle F, +, -, 0, \lambda_f \rangle_{f \in F})^n$, for $n \geq 1$.*

©2019 Hubie Chen and Matthew Valeriote.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-734.html.

A primary research goal of this line of inquiry is to completely understand, over all finite structures $\mathbf{B}$, which problems of the form $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ are learnable and which are not.

1.2. Related Previous Work

Let us survey the main known results about the framework of learning problems $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$.[1] Dalmau (1999) presented an analog of Schaefer's theorem, namely, a dichotomy theorem indicating, for each relational structure $\mathbf{B}$, which of the problems $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ are learnable. Precisely, this dichotomy theorem implies that each such problem is either polynomially learnable with equivalence queries, or is not polynomially predictable with membership queries. This dichotomy is sharp in that each problem that is polynomially learnable with equivalence queries is also polynomially predictable with membership queries (Littlestone, 1988, Section 4). The negative result, and all others under discussion, are proved under established cryptographic assumptions which are invoked in the present article (see Section 2.2), and the positive and negative results in the following discussion are proved in the two mentioned models, respectively.

Dalmau and Jeavons (2003) established a link between this framework and universal algebra; gave a general strategy for presenting positive results; and provided dichotomy theorems for two restricted classes of structures. Bulatov et al. (2007) gave a positive learnability result which applies to each relational structure having a so-called *generalized majority-minority polymorphism*. Later, Idziak et al. (2010) gave a positive learnability result generalizing all previous positive results; their result applies to any structure $\mathbf{B}$ for which all solution spaces have *small* (polynomial-size) generating sets, in a precise sense (see the discussion after Definition 9). They point out that all previous positive results were based on small generating sets, and hence that their result is a natural culmination of the progression of positive results.

1.3. Contributions

In this article, we complete the classification program towards which all of these previous works strive, by presenting a negative learnability result that complements the positive learnability result of Idziak et al. and hence that encompasses all previous negative learnability results in the framework at hand. We prove that for any structure $\mathbf{B}$ for which the small generating sets condition of Idziak et al. fails, it holds that $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ is not polynomially predictable with membership queries. We accomplish this by reducing the problem of learning the solution space of an unknown propositional formula, denoted by $\mathcal{C}_{\mathrm{PF}}$, to the problem $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$. We then refine this result by showing that if $\mathbf{B}$ fails to satisfy a weaker condition, that of having a Taylor polymorphism, then the problem $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ is as hard to learn as a prediction problem concerning partial assignments to circuits.

In order to establish our negative results, we make significant use of universal-algebraic notions and results, which we now turn to elaborate on. Each structure $\mathbf{B}$ can be passed to an algebra, its so-called algebra of polymorphisms, and it is known that the complexity of learning $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ is an invariant of this passage (that is, two structures that are passed to the same algebra have the same complexity of learning; see Proposition 4). We consider the variety generated by the algebra of a structure, which we show is justified (Proposition 5). If this variety is *congruence modular*, then we invoke a theorem, due to Barto (2018), which shows that the algebra of $\mathbf{B}$ has a property called *few subpowers*, and thus that the Idziak et al. positive result can be applied. (Barto's theorem resolved in the positive a conjecture known as the *Edinburgh Conjecture*, see Bova et al., 2013).

A main result of this article is the negative learnability result that, if the mentioned variety is *not* congruence modular, then the problem $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ is hard to learn. In order to prove this, we make use of concepts developed in a previous work which also studied non-congruence modularity (Bova et al., 2013). In particular, we make use of a structural result established there (Lemma 21) which essentially shows that, to prove hardness, one can work with a relational structure which can be localized to behave as a set of *pentagons*, which are a certain type of relational structure. Exploiting this structural result in the context of learning, however, is far from obvious, and involves developing significantly more detailed reductions than those used in the previous work (Bova et al., 2013), which dealt with comparing the solution spaces of two given conjunctive queries. The reason the reductions need to be more detailed here is that, when reducing one problem to another, one needs to translate from one concept to a second in a way that closely preserves structure (in our case of studying learning of unknown formulas, reductions need to preserve structure of the solution spaces); this contrasts sharply with the earlier work (Bova et al., 2013), where reductions needed only preserve a single bit, namely, the answer to a decision problem. Indeed, as an intermediate step, we show the hardness of a natural term-learning problem on lattices, which may be of independent interest (Section 5).

To obtain our second negative result, we make use of another part of Bova et al. (2013) that deals with finite structures that fail to have a special type of polymorphism called a Taylor polymorphism. There it is shown that in the absence of a Taylor polymorphism, a finite structure $\mathbf{B}$ interprets, in a certain sense, a structure representing the 3-SAT problem. From this, we prove our hardness result.

2. Preliminaries

When P is a condition (such as a containment $x \in c$), we use $[P]$ to denote the value equal to 1 if P is true, and 0 if P is false. When $f : A \to B$ and $g : B \to C$ are functions, we sometimes use $g(f)$ to denote their composition.

2.1. Concept Learning

Our terminology and notation is based on those employed by Pitt and Warmuth (1990) and by Angluin and Kharitonov (1995).

1. Let us mention that, in the existing literature, (in for example in Bulatov et al., 2007) some positive results are stated for queries where universal quantification is permitted in addition to predicate applications, equality of variables, conjunction, and existential quantification; let us refer to such queries as *quantified conjunctive queries*. Let $\mathbf{B}^*$ be the structure obtained from $\mathbf{B}$ by adding each element of the universe of $\mathbf{B}$ as a relation. These positive results typically apply to the structure $\mathbf{B}^*$ whenever they apply to a structure $\mathbf{B}$. For a structure of the form $\mathbf{B}^*$, it can be readily verified that the solution space of any quantified conjunctive query is also the solution space of a conjunctive query; this can be shown by transforming a quantified conjunctive query into a conjunctive query by using the additional relations to eliminate each instance of universal quantification.

   As the main contribution of the present article is to present a negative result, we focus the present discussion on conjunctive queries.

We assume that objects are encoded over the binary alphabet $\{0,1\}$, and use X to denote $\{0,1\}^*$. When x is a string, we use $|x|$ to denote its length, and for each $n \in \mathbb{N}$, we use $X^{[n]}$ to denote $\{x \in X \ : \ |x| \leq n\}$. A *prediction problem* $\mathcal{C}$ is a subset of $X \times X$; when $(u,x) \in \mathcal{C}$, we refer to u as a *concept name* or *concept representation* (of $\mathcal{C}$). Relative to a prediction problem $\mathcal{C}$, the *concept represented by* u is defined as $\kappa_{\mathcal{C}}(u) = \{x \mid (u,x) \in \mathcal{C}\}$.

A *pwm-algorithm* (short for *prediction with membership queries algorithm*) is an algorithm A with the following properties. The algorithm A takes as input a bound $s \in \mathbb{N}$ on the size of the target concept representation, a bound $n \in \mathbb{N}$ on the length of examples, and an *accuracy bound* ϵ, a positive rational number. It may make three types of oracle calls, the responses to which are determined by an unknown target concept c and an unknown distribution D on $X^{[n]}$: (1) A *membership query* takes a string $x \in X$ as input and returns $[x \in c]$; (2) A request for a random classified example takes no input and returns a pair (x,b), where x is a string chosen independently according to D, and $b = [x \in c]$; (3) A request for an element to predict takes no input and returns a string x chosen independently according to D. The algorithm A may make any number of oracle calls of types 1 and 2; however, in any run, it must make exactly one oracle call of type 3 and then eventually halt with an output of 1 or 0 without making any further oracle calls.

A pwm-algorithm is said to run in polynomial time if its running time is bounded by a polynomial in s, n, and $1/\epsilon$. A pwm-algorithm A is said to *successfully* predict a prediction problem $\mathcal{C}$ if for each input (s,n,ϵ), each concept name $u \in X^{[s]}$ of $\mathcal{C}$, and for each probability distribution D on $X^{[n]}$, when A is run on (s,n,ϵ) and the oracle calls of type 1 and 2 are answered according to $c = \kappa_{\mathcal{C}}(u)$ and D, the probability that the output of A is not equal to $[x \in c]$ is bounded above by ϵ. A prediction problem is *polynomially predictable with membership queries* if there exists a pwm-algorithm that runs in polynomial time and successfully predicts $\mathcal{C}$.

2.2. Problems

We introduce the problems that will be of concern.

A *relational signature* is a finite set of *relation symbols*; each relation symbol has an arity $k \geq 0$ associated with it. Note that we assume that all relational signatures under discussion are finite. A *relational structure* $\mathbf{B}$ over a relational signature σ consists of a finite set B called its *universe* and, for each relation symbol $R \in \sigma$, a relation $R^{\mathbf{B}} \subseteq B^k$, where k is the arity of R. We generally use the letters $\mathbf{A}$, $\mathbf{B}$, ... to denote relational structures, and the corresponding letters A, B, ... to denote their respective universes. Note that we assume that all relational structures under discussion are *finite* in that each has a finite universe; nonetheless, we sometimes state this explicitly for emphasis. A *conjunctive query* on a relational signature σ is a first-order formula built from predicate applications $R(v_1,\ldots,v_k)$ (where $R \in \sigma$ and $v_1,\ldots,v_k$ are variables, with k equal to the arity of R), equality of variables $v = v'$, conjunction, and existential quantification. When $\mathbf{B}$ is a relational structure and $Q \subseteq B^k$ is a relation, we say that Q is *cq-definable* over $\mathbf{B}$ if there exists a conjunctive query $\phi(v_1,\ldots,v_k)$ such that $(b_1,\ldots,b_k)$ satisfies ϕ on $\mathbf{B}$ if and only if $(b_1,\ldots,b_k) \in Q$.

The prediction problems that we study are as follows. There is a problem for each relational structure $\mathbf{B}$. Each conjunctive query $\phi(V)$ over the signature of $\mathbf{B}$ is a concept

representation, and its concept is the set that contains an assignment $f : V \to B$ if it holds that $\mathbf{B}, f \models \phi$, that is, if it satisfies $\phi(V)$ over $\mathbf{B}$. Formally, for each relational structure $\mathbf{B}$, we define $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ to be the prediction problem

$$\{(\phi(V), f) \mid \phi \text{ is a conjunctive query and } f : V \to B \text{ is a mapping such that } \mathbf{B}, f \models \phi\}.$$

Our hardness results for prediction problems are based on the hardness of predicting propositional formulas and of predicting Boolean circuits. By a propositional formula, we understand a formula built from propositional variables and the basis consisting of AND ($\wedge$), OR ($\vee$), and NOT ($\neg$), where the fan-in of AND and OR is assumed to be two. We define $\mathcal{C}_{\mathrm{PF}}$ as the prediction problem containing those pairs (θ, f) where θ is a propositional formula, and f is a propositional assignment to the variables of θ that satisfies θ. (Note that the existence of a pwm-algorithm for $\mathcal{C}_{\mathrm{PF}}$ is readily verified to be insensitive to our assumption of fan-in two for AND and OR gates.) The following cryptographic evidence is known for the hardness of learning $\mathcal{C}_{\mathrm{PF}}$. Let us refer to the following three hypotheses, studied in Kearns and Valiant (1994), as the *Kearns-Valiant hypotheses*: testing quadratic residues is intractable; inverting RSA encryption is intractable; factoring Blum integers is intractable.

Theorem 1 *(Angluin and Kharitonov, 1995, Corollary 3) Under the assumption that one of the Kearns-Valiant hypotheses holds, the prediction problem $\mathcal{C}_{\mathrm{PF}}$ is not polynomially predictable with membership queries.*

By a *Boolean circuit*, we understand a circuit built from input variables and the standard basis consisting of AND ($\wedge$), OR ($\vee$), and NOT ($\neg$). We define $\mathcal{C}_{\mathrm{CIRC}}$ as the prediction problem containing those pairs (T, f) where T is a Boolean circuit, and f is an assignment to the variables of T that satisfies T. We define $\mathcal{C}_{\exists\mathrm{CIRC}}$ as the prediction problem containing those pairs $((T, U), f)$ where T is a Boolean circuit, U is a subset of the variables on which T is defined, and f is an assignment to U that can be extended to an assignment that satisfies T.

3. Reducibility and Hardness

In this section, we describe the notion of reduction that will be used throughout the paper (Section 3.1); we demonstrate how certain standard algebraic constructions are relevant in our learning context, and also present notions of algebra to be used (Section 3.2); and, we provide a certain learning problem on propositional formulas that will be wieldy (Section 3.3).

3.1. Oracular pwm-reducibility

We define an extension of the notion of *pwm-reduction* due to Angluin and Kharitonov (1995); we refer to our notion of reduction as *oracular pwm-reduction*.

An *oracular pwm-reduction* from a prediction problem $\mathcal{C}$ to a second prediction problem $\mathcal{C}'$ is a triple (f, g, H) where f and g are mappings and H is an algorithm with the following properties:

1. There exists a polynomial q such that for each $s, n \in \mathbb{N}$ and for each $u \in X^{[s]}$, it holds that $g(s, n, u)$ is a string with $|g(s, n, u)| \leq q(s, n, |u|)$.

2. For each $s, n \in \mathbb{N}$, for each $u \in X^{[s]}$, and for each $x \in X^{[n]}$, it holds that $x' = f(s, n, x)$ is a string such that $x \in \kappa_{\mathcal{C}}(u)$ if and only if $x' \in \kappa_{\mathcal{C}'}(g(s, n, u))$. Also, there exists a polynomial t such that f is computable in time $t(s, n, |x|)$.

3. For each $s, n \in \mathbb{N}$, for each $u \in X^{[s]}$, and for each $x' \in X^{[n]}$, the algorithm H, on input (s, n, x'), may submit strings $x \in X$ as queries to an oracle, which responds $[x \in \kappa_{\mathcal{C}}(u)]$; the algorithm's output must be $[x' \in \kappa_{\mathcal{C}'}(g(s, n, u))]$. The algorithm H is required to run in polynomial time (in s, n, and $|x'|$).

Let us remark that the existence of a pwm-reduction between two prediction problems immediately implies the existence of an oracular pwm-reduction: pwm-reducibility can be viewed as the special case of oracular pwm-reducibility where the algorithm H can make at most one oracle query and, in the case that this query is made, the result must be the output of H.

Proposition 2 *Let $\mathcal{C}$ and $\mathcal{C}'$ be prediction problems. If there exists an oracular pwm-reduction from $\mathcal{C}$ to C' and it holds that $\mathcal{C}'$ is polynomially predictable with membership queries, then $\mathcal{C}$ is also polynomially predictable with membership queries.*

The proof of Proposition 2 is extremely similar to that of Angluin and Kharitonov (1995, Lemma 2), so we only give the idea of the proof. Let A' be a pwm-algorithm that witnesses that $\mathcal{C}'$ is polynomially predictable with membership queries. We describe a pwm-algorithm A that witnesses that $\mathcal{C}$ is polynomially predictable with membership queries, as follows. When A is run on input (s, n, ϵ), it computes $s' = q(s, n, s)$ and $n' = t(s, n, n)$. It then performs a simulation of A' on input (s', n', ϵ). Oracle calls made by the simulation of A' are answered by A as follows.

1. When A' makes a membership query on string $x' \in X$, the algorithm A runs $H(s, n, x')$ using its own membership queries to respond to the oracle calls of H, and then returns the result to A'.

2. When A' requests a random classified example, the algorithm A makes a request for a random classified example to obtain (x, b), and then returns the pair $(f(s, n, x), b)$ to A'.

3. When A' requests an element to predict, the algorithm A requests an element x, and returns the string $f(s, n, x)$ to A'.

When the simulated algorithm A' halts with an output b, the algorithm A halts with the output b.

For each input (s, n, ϵ), for each concept name $u \in X^{[s]}$ of $\mathcal{C}$, and for each probability distribution D on $X^{[n]}$, set $u' = g(s, n, u)$ and set D' to be the induced distribution $f(s, n, D)$ on $X^{[n']}$. When the algorithm A is invoked on (s, n, ϵ) with u and D, in its simulation of A', membership queries are answered according to the concept $\kappa_{\mathcal{C}'}(u')$ and random classified examples are generated according to D'. The assumption that A' predicts correctly within

an error bound of ϵ can be verified to imply that A will predict within an error bound of ϵ. This concludes our description of the proof of Proposition 2.

The following property, which is straightforward to verify, will be used tacitly.[2]

Proposition 3 *Oracular pwm-reducibility is transitive.*

We leave, to future research, a study of whether and how oracular pwm-reducibility is more general than pwm-reducibility.

3.2. Algebras and Varieties

We make use of basic notions from universal algebra, and suggest Burris and Sankappanavar (1981) and McKenzie et al. (1987) as references. For our purposes in this article, an *algebra* is a pair $(A; F)$ consisting of a nonempty set A, the *universe* of the algebra, and a set F of finitary operations on A, the set of *basic operations* of the algebra. An algebra is *finite* if its universe is finite; we deal here mainly with finite algebras. A *term operation* (or sometimes just term) of an algebra $\mathbb{A}$ is any operation on A that can be obtained from the basic operations of $\mathbb{A}$ and the projection maps on A by composition. The *variety generated by an algebra* $\mathbb{A}$, denoted by $\mathcal{V}(\mathbb{A})$, is the smallest class of algebras containing $\mathbb{A}$ that is closed under taking homomorphic images, subalgebras, and products. An operation $f : B^m \to B$ is a *polymorphism* of a relation $Q \subseteq B^k$ if for any m tuples $(b_1^1, \ldots, b_k^1), \ldots, (b_1^m, \ldots, b_k^m)$ in Q, the tuple $(f(b_1^1, \ldots, b_1^m), \ldots, f(b_k^1, \ldots, b_k^m))$ is in Q. A relational structure $\mathbf{B}$ is *compatible* with an algebra having the same universe B if for each operation $f : B^m \to B$ of the algebra, it holds that f *is a polymorphism of* $\mathbf{B}$, by which is meant, f is a polymorphism of each relation of $\mathbf{B}$. We similarly speak of a single relation or a set of relations being *compatible* with an algebra. For a relational structure $\mathbf{B}$, we define $\mathbb{A}(\mathbf{B})$ to be the algebra with universe B and whose operations are the polymorphisms of $\mathbf{B}$.

We will make use of the following two facts. The first was established in previous work, and shows the relevance of the algebra of a structure to the problem framework at hand. The second shows the relevance of the variety of the algebra of a structure thereto.

Proposition 4 *(follows from Dalmau and Jeavons, 2003, Proof of Lemma 9) Suppose that $\mathbf{B}$ and $\mathbf{B}'$ are relational structures with the same universe and such that $\mathbf{B}$ is compatible with $\mathbb{A}(\mathbf{B}')$. Then there exists an oracular pwm-reduction from $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ to $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B}')$.*

Proposition 5 *Suppose that $\mathbb{B}$ is a finite algebra, and that $\mathbf{A}$ is a finite structure which is compatible with an algebra in $\mathcal{V}(\mathbb{B})$. Then, there exists a relational structure $\mathbf{B}$ which is compatible with $\mathbb{B}$ such that there exists an oracular pwm-reduction from $\mathcal{C}_{\mathrm{CQ}}(\mathbf{A})$ to $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$.*

It is well-known that a finite algebra is in $\mathcal{V}(\mathbb{B})$ if and only if it is a homomorphic image of a subalgebra of a finite power of $\mathbb{B}$. Proposition 5 thus follows immediately from the three following lemmas.

2. We remark that, strictly speaking, transitivity of oracular pwm-reducibility is not needed to derive the main result of the paper. Our main result shows that for certain relational structures $\mathbf{B}$, the prediction problem $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ is not polynomially predictable with membership queries unless $\mathcal{C}_{\mathrm{PF}}$ is as well. To establish this, it suffices to give a sequence of pwm-reductions from $\mathcal{C}_{\mathrm{PF}}$ to $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$ (which is what we do) and then invoke Proposition 2.

Lemma 6 *Suppose that $\mathbb{B}$ is a finite algebra and that $\mathbf{A}$ is a finite structure which is compatible with a finite power $\mathbb{A} = \mathbb{B}^n$ of $\mathbb{B}$. Then there exists a relational structure $\mathbf{B}$ compatible with $\mathbb{B}$ such that there exists an oracular pwm-reduction from $\mathcal{C}_{\mathrm{CQ}}(\mathbf{A})$ to $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$.*

Proof Let $\mathbf{A}$ be a structure on signature $\sigma_{\mathbf{A}}$ which is compatible with $\mathbb{A}$. Define $\sigma_{\mathbf{B}}$ to be the signature having the same symbols as $\sigma_{\mathbf{A}}$, but where the arity of a symbol $R \in \sigma_{\mathbf{B}}$ is nk, where k is the arity of R in $\sigma_{\mathbf{A}}$. Define $\mathbf{B}$ so that a relation $R^{\mathbf{B}}$ contains a tuple $(b_1^1, \ldots, b_1^n, \ldots, b_k^1, \ldots, b_k^n)$ if and only if $R^{\mathbf{A}}$ contains $((b_1^1, \ldots, b_1^n), \ldots, (b_k^1, \ldots, b_k^n))$.

The reduction (f, g, H) is as follows. For a conjunctive query $\phi_{\mathbf{A}}$ on $\sigma_{\mathbf{A}}$, the function g is defined so that $g(s, m, \phi_{\mathbf{A}})$ is equal to $\phi_{\mathbf{B}}$, where $\phi_{\mathbf{B}}$ is derived from $\phi_{\mathbf{A}}$ by replacing each variable v by a tuple $(v^1, \ldots, v^n)$ of variables. The mapping f is defined so that, when h is an assignment from V to $A = B^n$, $f(s, m, h)$ is the map $h' : \{v^1, \ldots, v^n \mid v \in V\} \to B$ such that the following condition holds: for each $v \in V$, it holds that $h(v) = (h'(v^1), \ldots, h'(v^n))$. The algorithm $H(s, m, h' : \{v^1, \ldots, v^n \mid v \in V\} \to B)$ calculates the mapping $h : V \to A$ defined according to the just-stated condition, submits h to its oracle, and outputs the result. This reduction is correct, as for a pair of assignments h, h' satisfying the condition, it holds that h satisfies $\phi_{\mathbf{A}}$ on $\mathbf{A}$ if and only if h' satisfies $\phi_{\mathbf{B}}$ on $\mathbf{B}$. ■

Lemma 7 *Suppose that $\mathbb{B}$ is a finite algebra and that $\mathbf{A}$ is a finite structure which is compatible with a subalgebra $\mathbb{A}$ of $\mathbb{B}$. Then there exists a relational structure $\mathbf{B}$ compatible with $\mathbb{B}$ such that there exists an oracular pwm-reduction from $\mathcal{C}_{\mathrm{CQ}}(\mathbf{A})$ to $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$.*

Proof Suppose that $\mathbf{A}$ is a structure on signature $\sigma_{\mathbf{A}}$ compatible with $\mathbb{A}$. Define $\sigma_{\mathbf{B}}$ to be a signature equal to $\sigma_{\mathbf{A}}$ but expanded by a relation symbol U of arity 1. Let $\mathbf{B}$ be the structure over $\sigma_{\mathbf{B}}$ with universe B where $R^{\mathbf{B}} = R^{\mathbf{A}}$ for each $R \in \sigma_{\mathbf{A}}$ and where $U^{\mathbf{B}} = A$.

The reduction (f, g, H) is as follows. For a conjunctive query $\phi_{\mathbf{A}}$ on $\sigma_{\mathbf{A}}$, define $g(s, n, \phi_{\mathbf{A}})$ to be the formula $\phi_{\mathbf{B}}$ which is obtained from $\phi_{\mathbf{A}}(V)$ by replacing each predicate application $R(v_1, \ldots, v_k)$ by $R(v_1, \ldots, v_k) \wedge U(v_1) \wedge \cdots \wedge U(v_k)$ to obtain $\psi(V)$, and then defining $\phi_{\mathbf{B}} = \psi(V) \wedge \bigwedge_{v \in V} U(v)$. Essentially, $\phi_{\mathbf{B}}$ is obtained from $\phi_{\mathbf{A}}$ by restricting all free and used variables to take on values in $U^{\mathbf{B}} = A$. It is straightforward to verify that $\phi_{\mathbf{A}}$ and $\phi_{\mathbf{B}}$ have the same solutions (with respect to the structures $\mathbf{A}$ and $\mathbf{B}$, respectively). Hence, f may be defined by $f(s, n, h) = h$, and $H(s, n, h')$ may be defined as the algorithm that passes h' to its oracle and outputs the result. ■

A *congruence* of an algebra $\mathbb{A} = (A; F)$ is an equivalence relation on A that is compatible with $\mathbb{A}$. Suppose that θ is a congruence of $\mathbb{A}$. We use a^θ to denote the equivalence class of θ containing $a \in A$. For each operation $f \in F$, the operation f^θ defined by $f^\theta(a_1^\theta, \ldots, a_k^\theta) = (f(a_1, \ldots, a_k))^\theta$ is well-defined. An algebra is a *homomorphic image* of $\mathbb{A}$ if it is isomorphic to an algebra of the form $(A^\theta; F^\theta)$ where $A^\theta = \{a^\theta \mid a \in A\}$ and $F^\theta = \{f^\theta \mid f \in F\}$.

Lemma 8 *Suppose that $\mathbb{B}$ is a finite algebra and that $\mathbf{A}$ is a finite structure which is compatible with a homomorphic image $\mathbb{A}$ of $\mathbb{B}$. Then there exists a relational structure $\mathbf{B}$ compatible with $\mathbb{B}$ such that there exists an oracular pwm-reduction from $\mathcal{C}_{\mathrm{CQ}}(\mathbf{A})$ to $\mathcal{C}_{\mathrm{CQ}}(\mathbf{B})$.*

Proof We assume that $\mathbb{A}$ is equal to (A^θ, F^θ) where θ is a congruence of $\mathbb{B}$. Let $\mathbf{A}$ be a structure over signature $\sigma_{\mathbf{A}}$ which is compatible with $\mathbb{A}$. Define $\mathbf{B}$ to be the structure over signature $\sigma_{\mathbf{A}}$ defined by $R^{\mathbf{B}} = \{(b_1, \ldots, b_k) \mid (b_1^\theta, \ldots, b_k^\theta) \in R^{\mathbf{A}}\}$. For any conjunctive query $\phi(V)$ over $\sigma_{\mathbf{A}}$, it is straightforward to verify that an assignment $h : V \to A$ satisfies ϕ on $\mathbf{A}$ if and only if one (equivalently, every) assignment $h' : V \to B$ with $(h'(v))^\theta = h(v)$ satisfies ϕ on $\mathbf{B}$.

The following is thus a reduction. Define $g(s, n, \phi) = \phi$, define $f(s, n, h)$ to be an assignment h' defined as above, and define $H(s, n, h')$ to be the algorithm that passes the assignment h defined by $h(v) = (h'(v))^\theta$ to its oracle and returns the result. ■

A *lattice* is an algebra $(L; \wedge, \vee)$ where each of the operations $\wedge$ and $\vee$ is binary, idempotent, commutative, and associative; and, the absorption law $a \wedge (a \vee b) = a \vee (a \wedge b) = a$ holds. A lattice naturally induces a partial order $\leq$ defined by $a \leq b$ if and only if $a \wedge b = a$. A lattice is *distributive* if it satisfies the identity $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$. We say that a lattice is *non-trivial* if its universe has size strictly greater than 1. By a *lattice term*, we refer to a term built from variables and the two operation symbols $\wedge$ and $\vee$. The congruences of an algebra naturally form a lattice. An algebra $\mathbb{A}$ is *congruence modular* if its lattice of congruences satisfies the modular law: $x \leq y \to x \vee (y \wedge z) = y \wedge (x \vee z)$. A class of algebras is *congruence modular* if each algebra therein is congruence modular.

Varieties whose members are all congruence modular have special properties and have been extensively studied in the universal algebra literature. For example, there is a useful notion of a commutator for such varieties and it has been used to establish strong structural results (Freese and McKenzie, 1987). Congruence modular varieties can be characterized in terms of the existence of special terms, called *Day terms* or *Gumm terms*, that satisfy particular equations (see Freese and Valeriote, 2009, Section 8). In the special case where the variety is of the form $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ for some finite relational structure $\mathbf{B}$, Barto (2018) has established a deep result that shows that this variety will be congruence modular if and only if $\mathbf{B}$ has a special polymorphism called an *edge term*.

Definition 9 *Let A be a nonempty set and $k > 1$. A* k-edge operation *on A is a $k+1$-ary operation $t(x_1, x_2, \ldots, x_{k+1})$ that satisfies the following equations:*

$$\begin{aligned}
t(y, y, x, x, x, \ldots, x) &= x \\
t(y, x, y, x, x, \ldots, x) &= x \\
t(x, x, x, y, x, \ldots, x) &= x \\
t(x, x, x, x, y, \ldots, x) &= x \\
&\vdots \\
t(x, x, x, x, x, \ldots, y) &= x.
\end{aligned}$$

A structure $\mathbf{B}$ has an edge term polymorphism *if for some $k > 1$ it has a polymorphism that is a k-edge operation.*

Example 4 *1. If $\mathbb{G} = (G, \cdot, {}^{-1}, e)$ is a group then the term operation $t(x_1, x_2, x_3) = x_2 \cdot x_1^{-1} \cdot x_3$ is a 2-edge operation on G.*

2. *If* $\mathbf{L} = (L; \wedge, \vee)$ *is a lattice, then the term operation*

$$m(x_1, x_2, x_3, x_4) = (x_2 \wedge x_3) \vee (x_2 \wedge x_4) \vee (x_3 \wedge x_4)$$

is a 3-edge operation on L.

The notion of an edge term was introduced and investigated in Berman et al. (2010). It is shown in that paper that for a finite algebra $\mathbb{A}$, having a k-edge term for some $k > 1$ is equivalent to the *few subpowers property* for $\mathbb{A}$: there is some polynomial $p(n)$ such that for each $n > 0$, the number of subalgebras of $\mathbb{A}^n$ is less than $2^{p(n)}$. In general, the number of subalgebras of $\mathbb{A}^n$ can grow doubly exponentially in n, so this is a special property of an algebra. It is also shown that having a k-edge term for some $k > 1$ is equivalent to the subalgebras of finite powers of $\mathbb{A}$ having "small generating sets", which means that there is some polynomial $g(n)$ such that for $n > 0$, each subalgebra of $\mathbb{A}^n$ has a generating set of size at most $g(n)$.

In the companion paper (Idziak et al., 2010) it is shown that if a finite relational structure $\mathbf{B}$ has an edge term polymorphism then the prediction problem $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ is polynomially exactly learnable with improper equivalence queries, using a concept representation that is polynomially evaluable. The key property of $\mathbf{B}$ used in the proof of this fact is that the solution space of a conjunctive query over $\mathbf{B}$ is a subalgebra of a power of the algebra $\mathbb{A}(\mathbf{B})$ and so has a small generating set. This allows for a nice compact representation of each concept that is represented by a conjunctive query over $\mathbf{B}$.

A weaker example of a term condition, along the lines of having an edge term, is that of having a *Taylor term* or *Taylor operation*. A *Taylor operation* on a set A is an n-ary function $t : A^n \to A$ that is *idempotent* in that it satisfies the equation $t(x, x, \ldots, x) = x$ and, for each $1 \leq i \leq n$, an equation of the form $t(v_1, v_2, \ldots, v_n) = t(w_1, w_2, \ldots, w_n)$, where the v_j and w_j are variables from the set $\{x, y\}$ and $v_i \neq w_i$. A Taylor term of an algebra $\mathbb{A}$ is a term operation of $\mathbb{A}$ that is a Taylor operation. It can be readily verified that any k-edge operation on a set is also a Taylor operation on it. In Taylor (1977) it is shown that, by some measure, the condition of having a Taylor term is the weakest sort of term condition of this type that an algebra (or variety) can satisfy.

A *Taylor polymorphism* of a structure $\mathbf{B}$ is a polymorphism of $\mathbf{B}$ that is also a Taylor operation. The only polymorphisms of the structure $\mathbf{B}_{\text{3SAT}}$ from Example 1 are the projection maps and so $\mathbf{B}_{\text{3SAT}}$ does not have a Taylor polymorphism (Barto et al., 2017). On the other hand, the structure $\mathbf{B}_{\text{HORN-3SAT}}$ from Example 2 has $x \wedge y$ as a Taylor polymorphism and $\mathbf{V}_{\mathbb{F}}$ from Example 3 has $x - y + z$ (this polymorphism ensures that $\mathcal{V}(\mathbb{A}(\mathbf{V}_{\mathbb{F}}))$ is congruence modular).

Taylor operations have played a central role in the investigation of the constraint satisfaction problem over a finite relational structure or finite algebra. The ultimate resolution of the Feder-Vardi Dichotomy Conjecture (Feder and Vardi, 1999) can be expressed in terms of Taylor polymorphisms, but also in terms of related conditions that are equivalent for finite structures and algebras. In Maróti and McKenzie (2008) it is shown that a finite algebra $\mathbb{A}$ will have a Taylor term if and only if it has a term that is a weak near-unanimity operation. A *weak near-unanimity operation* on a set A is an operation $w(x_1, x_2, \ldots, x_k)$ for some $k > 1$ that satisfies the equations $w(x, x, \ldots, x) = x$ and

$$w(y, x, \ldots, x) = w(x, y, \ldots, x) = \cdots = w(x, x, \ldots, x, y).$$

It is not hard to see that any weak near-unanimity operation is a Taylor operation but quite challenging to show that if a finite algebra has a Taylor term, then it also has a weak near-unanimity term operation. A similar result holds for finite structures: $\mathbf{B}$ will have a Taylor polymorphism if and only if it has a weak near-unanimity polymorphism.

Bulatov (2017) and Zhuk (2017) have shown that for a finite relational structure $\mathbf{B}$ such that for each $b \in B$, the unary relation $\{b\}$ is a relation of $\mathbf{B}$, $\mathsf{CSP}(\mathbf{B})$ is polynomial-time decidable if $\mathbf{B}$ has a Taylor polymorphism (or equivalently, a weak near-unanimity polymorphism). It has been known for some time (Jeavons, 1998) that in the absence of a Taylor polymorphism, this problem is NP-complete and so combining these results (and some other basic facts) leads to the following dichotomy: for a finite relational structure $\mathbf{B}$, $\mathsf{CSP}(\mathbf{B})$ is polynomial-time decidable or is NP-complete.

3.3. Propositional Formulas

By log, we indicate the logarithm base 2. When θ is a formula or a term, we define $\text{depth}(\theta)$ to be the maximum length of a path from the root of θ (viewed as a tree) to a leaf; we define $\text{leafsize}(\theta)$ to be the number of leaves of θ (again, viewed as a tree). Define $\mathcal{C}_{\text{log-MPF}}$ to be the subset of $\mathcal{C}_{\text{PF}}$ that contains a pair $(\theta, h) \in \mathcal{C}_{\text{PF}}$ when θ is *monotone* (that is, does not contain any instance of negation ($\neg$)) and when $\text{depth}(\theta) \leq 6 + 6\log(\text{leafsize}(\theta))$.

The following proposition is readily derivable using Spira's lemma and known techniques for representing a propositional formula as a monotone propositional formula.

Proposition 10 *There exists an oracular pwm-reduction from* $\mathcal{C}_{\text{PF}}$ *to* $\mathcal{C}_{\text{log-MPF}}$.

Let us prove this proposition, in two steps. We first observe a reduction to the following intermediate problem. Define $\mathcal{C}_{\text{log-PF}}$ to be the subset of $\mathcal{C}_{\text{PF}}$ that contains a pair $(\theta, h) \in \mathcal{C}_{\text{PF}}$ when θ has $\text{depth}(\theta) \leq 1 + 4\log(\text{leafsize}(\theta))$.

Lemma 11 *(derivable from Spira's Lemma; see the presentation/discussion in Bonet and Buss, 1994)*[3] *Let* ϕ *be a propositional formula; then there exists an equivalent propositional formula* ϕ' *such that* $\text{depth}(\phi') \leq 1 + 4\log(\text{leafsize}(\phi))$ *and such that* $\text{leafsize}(\phi) \leq \text{leafsize}(\phi') \leq \text{leafsize}(\phi)^3$. *It thus holds that* $\text{depth}(\phi') \leq 1 + 4\log(\text{leafsize}(\phi'))$.

The following proposition is readily derived from Lemma 11.

Proposition 12 *There exists an oracular pwm-reduction from* $\mathcal{C}_{\text{PF}}$ *to* $\mathcal{C}_{\text{log-PF}}$.

It then remains to give a reduction from $\mathcal{C}_{\text{log-PF}}$ to $\mathcal{C}_{\text{log-MPF}}$, which is what we now do.

Proposition 13 *There exists an oracular pwm-reduction from* $\mathcal{C}_{\text{log-PF}}$ *to* $\mathcal{C}_{\text{log-MPF}}$.

3. We remark that to guarantee $\text{leafsize}(\phi) \leq \text{leafsize}(\phi')$, one can repeatedly apply the transformation described in Bonet and Buss (1994) but leaving in the constants (0 and 1) that are introduced. At the end, one can then replace the constants 0 and 1 by $(v \wedge \neg v)$ and $(v \vee \neg v)$, respectively, where v is some variable.

Proof The proof proceeds along the lines of that of Dalmau (1999, Lemma 33). We describe the parts of the reduction. When θ is a propositional formula on variables $v_1, \ldots, v_m$, the mapping g is defined so that $\psi = g(s, n, \theta)$ is logically equivalent to

$$(v_1 \vee v_1') \wedge \cdots \wedge (v_m \vee v_m') \wedge [(v_1 \wedge v_1') \vee \cdots \vee (v_m \wedge v_m') \vee \theta].$$

Note that this expression is written using a conjunction of high fan-in and a disjunction of high fan-in. In each case, each can be rewritten as a formula where conjunction and disjunction have fan-in 2; then, the high fan-in conjunction and high fan-in disjunction are replaced with formulas having depth at most $1 + \log(m + 1)$. By rewriting in this fashion, we obtain ψ. As the depth of each disjunction $(v_i \vee v_i')$ has depth 1, we can naively bound the depth of ψ by $2 * (1 + \log(\text{leafsize}(\psi))) + 1 + \text{depth}(\theta)$ which is upper bounded by $6 + 6\log(\text{leafsize}(\psi))$. The function induced by the formula $g(s, n, \theta)$ evaluates to θ if for each i it holds that $v_i \neq v_i'$; to 0 if there exists an i such that $v_i = v_i'$; and, to 1 otherwise. For each assignment h from a set of variables $\{v_1, \ldots, v_m\}$ to $\{0, 1\}$, the mapping f is defined as $f(s, n, h) = h'$ where h' is the unique extension of h such that $h'(v_i') = \neg v_i$ for each i. The algorithm H, on input $(s, n, h' : \{v_1, v_1', \ldots, v_m, v_m'\} \to \{0, 1\})$, outputs the result of a query call on the restriction of h' to $\{v_1, \ldots, v_m\}$, if for each i it holds that $h'(v_i) \neq h'(v_i')$; 0 if there exists an i such that $h'(v_i) = h'(v_i')$; and, 1 otherwise. ■

4. Main Theorems

We are now in a position to present the main theorems and to explain how they will follow from the results in the following sections.

Theorem 14 *Let* $\mathbf{B}$ *be a finite relational structure.*

- *If the variety* $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ *is congruence modular, then the prediction problem* $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ *is polynomially exactly learnable with improper equivalence queries, using a concept representation that is polynomially evaluable.*
- *Otherwise, there is an oracular pwm-reduction from* $\mathcal{C}_{\text{PF}}$ *to* $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ *and so the prediction problem* $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ *is not polynomially predictable with membership queries unless* $\mathcal{C}_{\text{PF}}$ *is as well, and hence (by Theorem 1) not unless each of the Kearns-Valiant hypotheses fails.*

Let us remark that the following is known: each problem that is polynomially exactly learnable with improper equivalence queries under a polynomially evaluable concept representation is polynomially predictable with membership queries (see for example Angluin, 1988, Section 2.4).

Proof If the variety $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is congruence modular, then by Barto's theorem (Barto, 2018), it holds that this variety has *few subpowers* and that there is a k-edge polymorphism of $\mathbf{B}$; thus, the Idziak et al. result (Idziak et al., 2010, Corollary 5.6) applies. For more details, see the discussion after Definition 9 above. If this variety is not congruence modular, then Proposition 10, Theorem 18, Theorem 20, and Theorem 22 yield a sequence of

oracular pwm-reductions from the prediction problem $\mathcal{C}_{\text{PF}}$ to $\mathcal{C}_{\text{CQ}}(\mathbf{A})$, where $\mathbf{A}$ is a structure compatible with an algebra in the variety; an oracular pwm-reduction from $\mathcal{C}_{\text{CQ}}(\mathbf{A})$ to $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ exists by appeal to Propositions 5 and 4. Hence, $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ is not polynomially predictable with membership queries unless $\mathcal{C}_{\text{PF}}$ is as well, by Proposition 2. ■

The following theorem refines the understanding of the problems that are hard according to Theorem 14.

Theorem 15 *Let* $\mathbf{B}$ *be a finite relational structure.*

- *If* $\mathbf{B}$ *has a Taylor polymorphism, then there is an oracular pwm-reduction from* $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ *to* $\mathcal{C}_{\text{CIRC}}$.
- *Otherwise, there is an oracular pwm-reduction from* $\mathcal{C}_{\exists\text{CIRC}}$ *to* $\mathcal{C}_{\text{CQ}}(\mathbf{B})$.

We note that if $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is congruence modular, then $\mathbf{B}$ will have a Taylor polymorphism and so the positive result of Theorem 15 applies to $\mathcal{C}_{\text{CQ}}(\mathbf{B})$; however, the positive result of Theorem 14 is stronger than that of Theorem 15, since $\mathcal{C}_{\text{CIRC}}$ is not known to be learnable in the sense of the first item of Theorem 14. Hence, Theorem 15 is in essence a theorem concerning the problems that are hard according to Theorem 14.

We believe that it is natural to expect that Theorem 15 gives a true dichotomy in that the problem $\mathcal{C}_{\exists\text{CIRC}}$ is strictly harder than the problem $\mathcal{C}_{\text{CIRC}}$. A heuristic reason for believing this is that the computational decision problem of deciding if a given pair (T, f) belongs to $\mathcal{C}_{\text{CIRC}}$ is polynomial-time computable (instantiate the input variables according to f, and then evaluate the circuit), whereas the computational decision problem of deciding if a given pair $((T, U), f)$ belongs to $\mathcal{C}_{\exists\text{CIRC}}$ is NP-complete (in the case that $U = \emptyset$, this problem amounts to circuit satisfiability).

Before starting the proof of Theorem 15, recall that for a relational structure $\mathbf{B}$, $\mathsf{CSP}(\mathbf{B})$ is the computational problem of deciding, given a conjunctive query, whether or not its solution space over $\mathbf{B}$ is nonempty. Equivalently, $\mathsf{CSP}(\mathbf{B})$ is the problem of deciding, given a conjunctive query ψ whose variables are all existentially quantified, whether or not $\mathbf{B} \models \psi$, that is, whether or not ψ evaluates to true on $\mathbf{B}$. Define $\mathbf{B}^*$ to be the structure derived from $\mathbf{B}$ by adding, for each element $b \in B$, a relation $U_b^{\mathbf{B}^*} = \{b\}$.

Proof If $\mathbf{B}$ has a Taylor polymorphism, we argue as follows. The structure $\mathbf{B}^*$ also has a Taylor polymorphism, since a Taylor operation is idempotent. We obtain from Bulatov (2017) or Zhuk (2017) that $\mathsf{CSP}(\mathbf{B}^*)$ is polynomial-time decidable; hence, there is a polynomial-time computable function that, given a conjunctive query $\phi(V)$, outputs a Boolean circuit T such that for each assignment $f : V \to B$, it holds that $\mathbf{B}, f \models \phi$ if and only if f satisfies T. The existence of the claimed oracular pwm-reduction follows immediately.

If $\mathbf{B}$ does not have a Taylor polymorphism then by Theorem 23 there will be an oracular pwm-reduction from $\mathcal{C}_{\exists\text{CIRC}}$ to $\mathcal{C}_{\text{CQ}}(\mathbf{B})$. ■

From Theorems 14 and 15, we obtain the following corollary—a trichotomy theorem.

Corollary 16 *Let* $\mathbf{B}$ *be a finite relational structure.*

- *If the variety $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is congruence modular, then the prediction problem $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ is polynomially exactly learnable with improper equivalence queries, using a concept representation that is polynomially evaluable.*
- *If the variety $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is not congruence modular but $\mathbf{B}$ has a Taylor polymorphism, then there is an oracular pwm-reduction from $\mathcal{C}_{\text{PF}}$ to $\mathcal{C}_{\text{CQ}}(\mathbf{B})$, as well as an oracular pwm-reduction from $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ to $\mathcal{C}_{\text{CIRC}}$.*
- *Otherwise, there is an oracular pwm-reduction from $\mathcal{C}_{\exists\text{CIRC}}$ to $\mathcal{C}_{\text{CQ}}(\mathbf{B})$.*

We can apply this result to the examples presented in the introduction to determine the complexity of learning the solution spaces of conjunctive queries over them. Since the structure $\mathbf{B}_{\text{3SAT}}$ has no non-trivial polymorphisms, then it has no Taylor polymorphism and so there is an oracular pwm-reduction from $\mathcal{C}_{\exists\text{CIRC}}$ to $\mathcal{C}_{\text{CQ}}(\mathbf{B}_{\text{3SAT}})$. The structure $\mathbf{B}_{\text{HORN}-\text{3SAT}}$ has a Taylor polymorphism (the operation $x \wedge y$) but $\mathcal{V}(\mathbb{A}(\mathbf{B}_{\text{HORN}-\text{3SAT}}))$ is not congruence modular since this variety is essentially the variety of semilattices. So, there is an oracular pwm-reduction from $\mathcal{C}_{\text{PF}}$ to $\mathcal{C}_{\text{CQ}}(\mathbf{B}_{\text{HORN}-\text{3SAT}})$, as well as an oracular pwm-reduction from $\mathcal{C}_{\text{CQ}}(\mathbf{B}_{\text{HORN}-\text{3SAT}})$ to $\mathcal{C}_{\text{CIRC}}$. Since $\mathcal{V}(\mathbb{A}(\mathbf{V}_{\mathbb{F}}))$ is congruence modular for any finite field $\mathbb{F}$, then $\mathcal{C}_{\text{CQ}}(\mathbf{V}_{\mathbb{F}})$ is polynomially exactly learnable with improper equivalence queries, using a concept representation that is polynomially evaluable.

It is worth comparing the above trichotomy theorem with the CSP dichotomy theorem (Bulatov, 2017; Zhuk, 2017). The comparison works well up to a point, namely, if $\mathbf{B}$ falls into either of the first two cases of Corollary 16 then it will have a Taylor polymorphism, and so $\mathsf{CSP}(\mathbf{B})$ will be polynomial-time decidable. On the other hand, for any relational structure $\mathbf{B}$ that has a constant function as a polymorphism, $\mathsf{CSP}(\mathbf{B})$ is trivially polynomial-time decidable, even if $\mathbf{B}$ fails to have a Taylor polymorphism. So there are structures $\mathbf{B}$ such that $\mathsf{CSP}(\mathbf{B})$ is polynomial-time decidable while $\mathcal{C}_{\text{CQ}}(\mathbf{B})$ admits an oracular pwm-reduction from $\mathcal{C}_{\exists\text{CIRC}}$. It is the case that $\mathbf{B}$ will fall into the third case of Corollary 16 if and only if $\mathsf{CSP}(\mathbf{B}^*)$ is NP-complete.

Let us now present a theorem that addresses the effectivity of the dichotomy conditions of Theorems 14 and 15. That is, we address the complexity of deciding, given a relational structure $\mathbf{B}$, whether $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is congruence modular, and whether $\mathbf{B}$ has a Taylor polymorphism.

Theorem 17 *Let $\mathbf{B}$ be a finite relational structure.*

1. *There is an EXPTIME algorithm that decides if the variety $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is congruence modular.*
2. *The problem of deciding if $\mathbf{B}$ has a Taylor polymorphism is in NP.*

Proof Part 1 follows from the characterization of congruence modular varieties given by Day or Gumm (consult Section 8 of Freese and Valeriote, 2009). Using Gumm's characterization, to determine if $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is congruence modular one need only search amongst the ternary functions on B for a finite sequence of polymorphisms of $\mathbf{B}$ that satisfy a specified set of equations. This search can be carried out by an algorithm whose running time is bounded by an exponential function in the size of $\mathbf{B}$. A full discussion of the relevant details can be found in Section 8 of Freese and Valeriote (2009).

For Part 2, the result follows from a result of Siggers (2010) that shows that a finite structure will have a Taylor polymorphism if and only if it has one of arity 6. Thus, to certify that $\mathbf{B}$ has a Taylor polymorphism, we need only be able to quickly check that a given 6-ary operation on B preserves the relations of $\mathbf{B}$ and is a Taylor operation. ■

Kazda (2014) has shown that the decision problem addressed in Part 1 of Theorem 17 actually lies in the class NP. His algorithm is based on a "local" characterization of congruence modularity and a clever encoding of the problem into an instance of the constraint satisfaction problem over the structure. We note that Siggers' result used in the proof of this theorem has been refined in Kearnes et al. (2014) by reducing the arity of the term from 6 to 4.

5. Learning Lattice Terms

In this section, we prove the hardness of a class of prediction problems that deal with lattices, which will serve as a useful intermediate result on the way to our main hardness result; roughly speaking, the problems studied here involve learning the function induced by an unknown term. When $r \geq 1$ and $\mathcal{L}$ is a finite set of finite lattices, define $\mathcal{C}^r_{\text{TERM}}(\mathcal{L})$ to be the prediction problem containing a pair $(t, (\mathbf{L}, h, c))$ when the following conditions hold: t is a lattice term with $\text{depth}(t) \leq r + r\log(\text{leafsize}(t))$; $\mathbf{L} = (L; \wedge, \vee)$ is a lattice in $\mathcal{L}$; h is an assignment mapping each variable of t to an element of L; c is an element of L; and, $\mathbf{L}, h \models (t \geq c)$, that is, under the assignment h, the term t evaluates to a value greater than or equal to c in $\mathbf{L}$.

Theorem 18 *Suppose that $\mathcal{L}$ is a finite set of finite lattices containing a non-trivial lattice. Then, there exists $r > 1$ such that there exists an oracular pwm-reduction from the prediction problem $\mathcal{C}_{\text{log-MPF}}$ to the prediction problem $\mathcal{C}^r_{\text{TERM}}(\mathcal{L})$.*

It is helpful to first establish this theorem in the case of distributive lattices; the proof uses the fact that each finite distributive lattice can be embedded into a finite power of the two-element lattice.

Lemma 19 *Theorem 18 holds in the case that $\mathcal{L}$ contains only distributive lattices.*

Proof Let $\mathbf{L}_{\{0,1\}}$ denote the two-element lattice with bottom element 0 and top element 1. We first show that there exists a reduction from $\mathcal{C} = \mathcal{C}_{\text{log-MPF}}$ to $\mathcal{C}' = \mathcal{C}^6_{\text{TERM}}(\{\mathbf{L}_{\{0,1\}}\})$. The reduction (f, g, H) is defined as follows. The functions g and f are defined by $g(s, n, \theta) = \theta$ and $f(s, n, h) = (\mathbf{L}_{\{0,1\}}, h, 1)$. The algorithm H, on input $(s, n, (\mathbf{L}_{\{0,1\}}, h, b))$, does the following: if $b = 0$, it outputs 1, and if $b = 1$, it submits h as an oracle query and outputs the result. This reduction is correct, as for each monotone propositional formula θ and assignment h to the variables of θ, it always holds that $(\mathbf{L}_{\{0,1\}}, h, 0) \in \kappa_{\mathcal{C}'}(\theta)$; and, it holds that $(\mathbf{L}_{\{0,1\}}, h, 1) \in \kappa_{\mathcal{C}'}(\theta)$ if and only if h satisfies θ.

Now, suppose that $\mathbf{L}$ is a set of distributive lattices containing a non-trivial lattice. We exhibit a reduction from $\mathcal{C} = \mathcal{C}^6_{\text{TERM}}(\{\mathbf{L}_{\{0,1\}}\})$ to $\mathcal{C}' = \mathcal{C}^6_{\text{TERM}}(\mathcal{L})$, which suffices to give the lemma. It is well-known and straightforward to verify that each finite distributive lattice embeds into a finite power of the lattice $\mathbf{L}_{\{0,1\}}$. Let $D \geq 1$ be a sufficiently large

constant so that each $\mathbf{L} \in \mathcal{L}$ has an embedding into $\mathbf{L}_{\{0,1\}}^D$. For each $\mathbf{L} \in \mathcal{L}$, fix $e^{\mathbf{L}}$ to be such an embedding, and let $e_i^{\mathbf{L}}$ be the function that returns the ith coordinate of $e^{\mathbf{L}}$ (for $i = 1, \ldots, D$). Fix $\mathbf{L}^+ \in \mathcal{L}$ to be a non-trivial lattice, and let $\top$ and $\bot$ denote the top and bottom elements of $\mathbf{L}^+$, respectively. Let $e^+ : \{0,1\} \to \{\bot, \top\}$ be the mapping where $e^+(0) = \bot$ and $e^+(1) = \top$. The reduction is (f, g, H), defined as follows. The functions are defined by $g(s, n, t) = t$ and $f(s, n, (\mathbf{L}_{\{0,1\}}, h, c)) = (\mathbf{L}^+, e^+(h), e^+(c))$. The algorithm H, on input $(s, n, (\mathbf{L}, h, c))$, makes D oracle queries: for each $i = 1, \ldots, D$, it submits the query $(\mathbf{L}_{\{0,1\}}, e_i^{\mathbf{L}}(h), e_i^{\mathbf{L}}(c))$ and returns 1 if and only if all oracle calls were answered as 1. The correctness of H follows from the fact that, for any lattice term t, and any triple $(\mathbf{L}, h, c)$, it holds that $\mathbf{L}, h \models t \geq c$ if and only if $\mathbf{L}_{\{0,1\}}^D, e^{\mathbf{L}}(h) \models t \geq e^{\mathbf{L}}(c)$, which in turn is true if and only if for all $i = 1, \ldots, D$, it holds that $\mathbf{L}_{\{0,1\}}, e_i^{\mathbf{L}}(h) \models t \geq e_i^{\mathbf{L}}(c)$. ■

Proof (Theorem 18) By Lemma 19, it suffices to prove the theorem for each such set $\mathbf{L}$ that contains a non-distributive lattice. We prove this by induction on the maximum cardinality of a non-distributive lattice in $\mathbf{L}$. Define $s(x, y, z)$ to be the term $(x \wedge y) \vee (x \wedge z)$, and define $s'(x, y, z)$ to be the term $x \wedge (y \vee z)$. In the scope of this proof, when d and d' are elements of a lattice $\mathbf{L}$ with $d \leq d'$, we use $[d, d']$ to denote the set $\{c \mid d \leq c \leq d'\}$, and we use $\mathbf{L}[d, d']$ to denote the sublattice of $\mathbf{L}$ with universe $[d, d']$. Note that for any elements a, b, c of a lattice $\mathbf{L}$, it always holds that $s(a, b, c) \leq s'(a, b, c)$.

Define $\mathcal{L}^-$ as the set $\{\mathbf{L}[s(a, b, c), s'(a, b, c)] \mid a, b, c \in \mathbf{L} \text{ and } \mathbf{L} \in \mathcal{L}\}$. We will prove that, for any value $r > 1$, it holds that $\mathcal{C}_{\text{TERM}}^r(\mathcal{L}^-)$ has an oracular pwm-reduction to $\mathcal{C}_{\text{TERM}}^{r+4}(\mathcal{L})$. Let us argue that this suffices. Consider a lattice $\mathbf{L} \in \mathcal{L}$. If the lattice $\mathbf{L}$ is distributive, then for any elements $a, b, c \in \mathbf{L}$, it holds that $s(a, b, c) = s'(a, b, c)$ and thus that $\mathbf{L}[s(a, b, c), s'(a, b, c)]$ is a one-element lattice. If the lattice $\mathbf{L}$ is non-distributive, then for any elements $a, b, c \in \mathbf{L}$, if $s'(a, b, c)$ is the top element of $\mathbf{L}$, then a must be equal to the top element of $\mathbf{L}$, which in turn implies that $s(a, b, c) = s'(a, b, c)$. Hence (when $\mathbf{L}$ is non-distributive) each lattice of the form $\mathbf{L}[s(a, b, c), s'(a, b, c)]$ has cardinality strictly smaller than that of $\mathbf{L}$. Now consider two cases. If $\mathcal{L}^-$ contains a non-distributive lattice, then by the argumentation just given and by induction, there exists a value r such that $\mathcal{C}_{\text{TERM}}^r(\mathcal{L}^-)$ admits an oracular pwm-reduction from $\mathcal{C}_{\text{log-MPF}}$, and hence an oracular pwm-reduction from $\mathcal{C}_{\text{TERM}}^r(\mathcal{L}^-)$ to $\mathcal{C}_{\text{TERM}}^{r+4}(\mathcal{L})$ yields the theorem. If $\mathcal{L}^-$ contains only distributive lattices, we claim that $\mathcal{L}^-$ contains a non-trivial lattice, which completes the argument by appeal to Lemma 19. This claim holds because there exists (by assumption) a non-distributive lattice $\mathbf{L} \in \mathcal{L}$; by definition, there exist elements $a, b, c \in \mathbf{L}$ such that $s(a, b, c) \neq s'(a, b, c)$. Hence, the lattice $\mathbf{L}[s(a, b, c), s'(a, b, c)]$ is non-trivial.

It remains to give an oracular pwm-reduction (f, g, H) from $\mathcal{C}_{\text{TERM}}^r(\mathcal{L}^-)$ to $\mathcal{C}_{\text{TERM}}^{r+4}(\mathcal{L})$. First, define $g(r, n, t^-(x_1, \ldots, x_n))$ to be the term $t(z_1, z_2, z_3, x_1, \ldots, x_n)$ defined to be $t^-(x_1^*, \ldots, x_n^*)$, where each x_i^* is defined as the term $(x_i \vee s(z_1, z_2, z_3)) \wedge s'(z_1, z_2, z_3)$. Observe that $\text{depth}(t) \leq \text{depth}(t^-) + 4$. Define $f(r, n, (\mathbf{L}^-, h^-, c^-))$ to be $(\mathbf{L}, h, c^-)$ where $\mathbf{L}$ is a lattice in $\mathbf{L}$ such that there exist $a, b, c \in \mathbf{L}$ with $\mathbf{L}^- = \mathbf{L}[s(a, b, c), s'(a, b, c)]$, and where h is the extension of h^- defined on $\{z_1, z_2, z_3, x_1, \ldots, x_n\}$ where $h(z_1) = a$, $h(z_2) = b$, and $h(z_3) = c$. This f satisfies the needed property, as $\mathbf{L}^-, h^- \models t^- \geq c^-$ holds if and only if $\mathbf{L}, h^- \models t^- \geq c^-$ holds; this latter condition is equivalent to $\mathbf{L}, h \models t \geq c^-$, as $h(x_i^*)$ is equal to $h^-(x_i)$ for each i. Define the algorithm H on $(r, n, (\mathbf{L}, h, d))$ to perform

the following. Let $\mathbf{L}^-$ be the lattice $\mathbf{L}[s(h(z_1), h(z_2), h(z_3)), s'(h(z_1), h(z_2), h(z_3))]$. Define h^- on $\{x_1, \ldots, x_n\}$ by $h^-(x_i) = (h(x_i) \vee s(h(z_1), h(z_2), h(z_3))) \wedge s'(h(z_1), h(z_2), h(z_3))$. Set D^- to be the set $\{d^- \in \mathbf{L}^- \mid d^- \geq d\}$. The algorithm H makes, for each $d^- \in D^-$, the oracle query $(\mathbf{L}^-, h^-, d^-)$, and returns 1 if and only if at least one of the oracle responses was 1. Let us discuss why this algorithm satisfies the desired property. It is readily verified that, when t and t^- are terms with $g(s, n, t^-(x_1, \ldots, x_n)) = t(z_1, z_2, z_3, x_1, \ldots, x_n)$ and $(\mathbf{L}, h, d)$ is a triple, that $\mathbf{L}, h \models t \geq d$ if and only if $\mathbf{L}, h \models t^-(x_1^*, \ldots, x_n^*) \geq d$ if and only if $\mathbf{L}, h^- \models t^-(x_1, \ldots, x_n) \geq d$. Since all values in the image of h^- are in $\mathbf{L}^-$, the last condition $\mathbf{L}, h^- \models t^-(x_1, \ldots, x_n) \geq d$ holds if and only if there exists $d^- \in D^-$ such that $\mathbf{L}^-, h^- \models t^-(x_1, \ldots, x_n) \geq d^-$. ■

6. Hardness From Non-Congruence Modularity

In this section, we make use of 2-sorted relational structures. We thus first give an introduction to these. A 2-*sorted signature* is a finite set of relation symbols, where each symbol has an associated arity which is a string over $\{1, 2\}$. Below, the only 2-sorted signature to be used is $\{R\}$, where the symbol R has arity 122. A 2-*sorted relational structure* over a 2-sorted signature σ consists of two finite sets B_1, B_2 called the *first* and *second* universe, respectively, and, for each relation symbol $S \in \sigma$, a relation $S^{\mathbf{B}}$ that is a subset of $B_{w_1} \times \cdots \times B_{w_k}$, where $w_1 \ldots w_k$ is the arity of S. A conjunctive query on a 2-sorted signature is defined similarly to a conjunctive query on a usual relational signature, but with the differences that each variable has an associated sort (1 or 2), equality of variables may only be written between variables of the same sort, and when $S(v_1, \ldots, v_k)$ is a predicate application, letting $s(v_i)$ denote the sort of v_i, it must be that $s(v_1) \ldots s(v_k)$ is equal to the arity of S.

We now turn to the proper content of this section. Let A be a set. When θ and θ' are binary relations on A, we use $\theta \circ \theta'$ to denote their relational product. We use $\text{Eq}(A)$ to denote the lattice of equivalence relations on A, and we use $0_A = \{(a, a) \mid a \in A\}$ and $1_A = A^2$ to denote the bottom and top elements of $\text{Eq}(A)$, respectively. We define a *pentagon* to be a finite relational structure $\mathbf{P}$ over the signature $\{\alpha, \beta, \gamma\}$ containing three binary relation symbols such that $\alpha^{\mathbf{P}}$, $\beta^{\mathbf{P}}$, and $\gamma^{\mathbf{P}}$ are equivalence relations on P, and the following conditions hold in $\text{Eq}(P)$: $\alpha^{\mathbf{P}} \leq \beta^{\mathbf{P}}$, $\beta^{\mathbf{P}} \wedge \gamma^{\mathbf{P}} = 0_P$, $\beta^{\mathbf{P}} \circ \gamma^{\mathbf{P}} = 1_P$, and $\alpha^{\mathbf{P}} \vee \gamma^{\mathbf{P}} = 1_P$. The universe P of a pentagon $\mathbf{P}$ can be naturally decomposed as a direct product $P = B \times C$ in such a way that $\beta^{\mathbf{P}}$ and $\gamma^{\mathbf{P}}$ are the kernels of the projections of P onto B and C, respectively. Then, via the equivalence relation $\alpha^{\mathbf{P}}$, each element $b \in B$ induces an equivalence relation $\alpha_b^{\mathbf{P}} = \{(c, c') \in C \times C \mid ((b, c), (b, c')) \in \alpha^{\mathbf{P}}\}$ on C. For each pentagon $\mathbf{P}$, we define $\mathbf{L}(\mathbf{P})$ to be the lattice which is the sublattice of $\text{Eq}(C)$ generated by the equivalence relations $\alpha_b^{\mathbf{P}}$ (over $b \in B$); we extend this operator $\mathbf{L}(\cdot)$ to sets of pentagons in the natural fashion.

To each pentagon $\mathbf{P}$, we associate a 2-sorted relational structure, denoted by $\mathbf{P}_2$, which has $B_{\mathbf{P}}$ and $C_{\mathbf{P}}$ as first and second universe, respectively; here, $B_{\mathbf{P}}$ and $C_{\mathbf{P}}$ denote the sets in the decomposition of the universe P as described above. The structure $\mathbf{P}_2$ is defined on signature $\{R\}$ and has $R^{\mathbf{P}_2} = \{(b, c, c') \in B_{\mathbf{P}} \times C_{\mathbf{P}} \times C_{\mathbf{P}} \mid (c, c') \in \alpha_b^{\mathbf{P}}\}$. The definition of $\mathbf{P}_2$ comes from (Bova et al., 2013). In forming conjunctive queries over this signature $\{R\}$

each variable has a sort (first or second) associated with each variable; an atom $R(x, y, y')$ may be formed if x is of the first sort and y and y' are of the second sort. When $\mathcal{P}$ is a set of pentagons, we define the prediction problem $\mathcal{C}_{\text{CQ-2-PENT}}(\mathcal{P})$ to be the set

$$\{(\phi(V_1, V_2), (\mathbf{P}, (h_1, h_2))) \mid \mathbf{P} \in \mathcal{P} \text{ and } h_1 : V_1 \to B_{\mathbf{P}},\ h_2 : V_2 \to C_{\mathbf{P}} \\ \text{such that } \mathbf{P}_2, h_1, h_2 \models \phi\ \}.$$

Here, $\phi(V_1, V_2)$ denotes a conjunctive query over the signature $\{R\}$ with V_1 a set of variables of the first sort and V_2 a set of the second sort.

Theorem 20 *For $\mathcal{P}$ a finite set of pentagons, there exists an oracular pwm-reduction from the prediction problem $\mathcal{C}^r_{\text{TERM}}(\mathbf{L}(\mathcal{P}))$ for any $r > 1$ to the prediction problem $\mathcal{C}_{\text{CQ-2-PENT}}(\mathcal{P})$.*

Proof We make use of a version of a construction presented in the proof of Theorem 10 from Bova et al. (2013), which produces a 2-sorted conjunctive query $\phi_t(x_1, \ldots, x_m, y, y')$ over the signature $\{R\}$ from a lattice term $t(x_1, \ldots, x_m)$, where in ϕ_t the variables x_i are of sort 1 and the variables y and y' are of sort 2. The construction has the property that if $\mathbf{P} \in \mathcal{P}$, then for all $b_1, \ldots, b_m \in B_{\mathbf{P}}$ and for all c, $c' \in C_{\mathbf{P}}$, $\phi_t(b_1, \ldots, b_m, c, c')$ holds in $\mathbf{P}_2$ if and only if the pair (c, c') is in the equivalence relation given by $t^{\mathbf{L}(\mathbf{P})}(\alpha^{\mathbf{P}}_{b_1}, \ldots, \alpha^{\mathbf{P}}_{b_m})$. Let us specify the version of the construction used here.

- If $t = x_i$, then $\phi_t(x_1, \ldots, x_m, y, y') = R(x_i, y, y')$.
- If $t = t_1 \wedge t_2$, then $\phi_t(x_1, \ldots, x_m, y, y') = \phi_{t_1}(x_1, \ldots, x_m, y, y') \wedge \phi_{t_2} t(x_1, \ldots, x_m, y, y')$.
- If $t = t_1 \vee t_2$, then set M to be the maximum size of a second universe $C_{\mathbf{P}}$ over all pentagons $\mathbf{P} \in \mathcal{P}$. Let $z_{0,2}$ and $z_{i,j}$, where $i = 1, \ldots, M$ and $j = 1, 2$, be variables of the second sort, and identify $y = z_{0,2}$ and $y' = z_{M,2}$. Then $\phi_t(x_1, \ldots, x_m, y, y')$ is defined as the formula obtained by existentially quantifying the variables $z_{i,j}$ (other than y and y') before the conjunction

$$\bigwedge_{i=1}^{M} (\phi_{t_1}(x_1, \ldots, x_m, z_{i-1,2}, z_{i,1}) \wedge \phi_{t_2}(x_1, \ldots, x_m, z_{i,1}, z_{i,2})).$$

Note that in each of the latter two cases, the size $|\phi_t|$ of the created formula ϕ_t has size bounded above by a constant times $|\phi_{t_1}| + |\phi_{t_2}|$. Hence, the size of ϕ_t will be polynomial in that of t—when t has logarithmic depth, which will be the case in our application here.

To define the g component of the reduction, the construction just given is not applied directly to a lattice term t. Instead, we first start with a fixed lattice term $s(x_1, \ldots, x_q)$ with the property that for all $\mathbf{P} \in \mathcal{P}$ and all $\delta \in \mathbf{L}(\mathbf{P})$ there are $b_i \in B_{\mathbf{P}}$, for $1 \leq i \leq q$, so that $\delta = s^{\mathbf{L}(\mathbf{P})}(\alpha^{\mathbf{P}}_{b_1}, \ldots, \alpha^{\mathbf{P}}_{b_q})$. We let $\omega(\delta)$ be some sequence $(b_1, \ldots, b_q)$ for which this equality holds. The existence of such a term follows from the fact that $\mathcal{P}$ is a finite set of finite pentagons and that in each pentagon $\mathbf{P}$, the equivalence relations $\alpha^{\mathbf{P}}_b$ generate the lattice $\mathbf{L}(\mathbf{P})$. For future reference, let u be an integer such that $|\mathbf{P}| \leq u$ for all $\mathbf{P} \in \mathcal{P}$. For any lattice term $t(x_1, \ldots, x_m)$, we define $t \star s$ to be the mq-ary lattice term

$$t(s(x_{11}, \ldots, x_{1q}), \ldots, s(x_{m1}, \ldots, x_{mq})).$$

Let us now begin to describe the reduction; the parameters s and n (as described in the definition of oracular pwm-reduction) do not play a role, and we omit their mention. For each lattice term $t(x_1, \ldots, x_m)$, define $g(t)$ to be the following conjunctive query over the signature $\{R\}$:

$$\bigwedge_{1 \leq i \leq u^2} \phi_{t \star s}(x_{11}, x_{12}, \ldots, x_{mq}, y_i, y'_i).$$

For $\mathbf{P} \in \mathcal{P}$, an m-tuple $h = (\delta_1, \ldots, \delta_m)$ over $\mathbf{L}(\mathbf{P})$, and $\theta \in \mathbf{L}(\mathbf{P})$, define the function $f((\mathbf{L}(\mathbf{P}), h, \theta))$ to be $\mathbf{P}$ along with the assignment of the variables $(x_{11}, \ldots, x_{mq})$ to the concatenation of the q-tuples $\omega(\delta_i)$, for $1 \leq i \leq m$. The pairs of variables (y_i, y'_i), for $1 \leq i \leq u^2$, are assigned to pairs $(c, c') \in \theta$ so that each pair in θ is in the range of this assignment. By the choice of u, this is always possible to arrange. Note that $f((\mathbf{L}(\mathbf{P}), h, \theta))$ can be computed in time bounded by some polynomial in m.

It follows from the claim made about the construction ϕ_t in the first paragraph of this proof, and from the properties of the lattice term s that for all lattice terms $t(x_1, \ldots, x_m)$, if $\mathbf{P} \in \mathcal{P}$, $h = (\delta_1, \ldots, \delta_m) \in \mathbf{L}(\mathbf{P})^m$, and $\theta \in \mathbf{L}(\mathbf{P})$ then $t^{\mathbf{L}(\mathbf{P})}(\delta_1, \ldots, \delta_m) \geq \theta$ if and only if $\mathbf{P}_2$ satisfies the conjunctive query $g(t)$ under the assignments given by $f((\mathbf{L}(\mathbf{P}), h, \theta))$.

To complete the definition of our reduction from $\mathcal{C}^r_{\text{TERM}}(\mathbf{L}(\mathcal{P}))$ to $\mathcal{C}_{\text{CQ-2-PENT}}(\mathcal{P})$, define H to be the algorithm that when given $\mathbf{P} \in \mathcal{P}$ and h_1 and h_2 tuples over $B_{\mathbf{P}}$ and $C_{\mathbf{P}}$ respectively, will reject the input if h_1 is not an mq-tuple for some integer m or if h_2 is a tuple which is not of length $2u^2$. Otherwise, with $h_1 = (b_{11}, \ldots, b_{mq})$ and $h_2 = (c_1, c'_1, \ldots, c_{u^2}, c'_{u^2})$, construct in $\mathbf{L}(\mathbf{P})$ the elements $\delta_i = s^{\mathbf{L}(\mathbf{P})}(\alpha^{\mathbf{P}}_{b_{i1}}, \ldots, \alpha^{\mathbf{P}}_{b_{iq}})$ and find the smallest element $\theta \in \mathbf{L}(\mathbf{P})$ such that $(c_i, c'_i) \in \theta$ for all $1 \leq i \leq u^2$. For any lattice term t, the algorithm will return the result of testing if in $\mathbf{P}$, $t \geq \theta$ under the assignment $(\delta_1, \ldots, \delta_m)$. From the properties of g noted earlier, it follows that in $\mathbf{P}$, $t \geq \theta$ under this assignment if and only if $g(t)$ is true in $\mathbf{P}_2$ under the assignment (h_1, h_2). The run time of H can be bounded by a polynomial in the lengths of h_1 and h_2. ■

Lemma 21 *(Bova et al., 2013) Let $\mathbf{B}$ be a finite relational structure such that $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is not congruence modular. There exists a finite relational structure $\mathbf{A}$ defined on a signature including three binary relation symbols α, β, and γ which is compatible with an algebra in $\mathcal{V}(\mathbb{A}(\mathbf{B}))$, such that the following hold:*

- *There exists a finite set $\mathcal{P}$ of pentagons where for each $\mathbf{P} \in \mathcal{P}$, the universe P of $\mathbf{P}$ is a subset of A, and it holds that $\alpha^{\mathbf{P}} = \alpha^{\mathbf{A}} \cap P^2$, $\beta^{\mathbf{P}} = \beta^{\mathbf{A}} \cap P^2$, and $\gamma^{\mathbf{P}} = \gamma^{\mathbf{A}} \cap P^2$. Moreover, the set $\mathbf{L}(\mathcal{P})$ contains a non-trivial lattice.*
- *For each $k \geq 1$, there exists a relation $D_k \subseteq A^k$ which is cq-definable over $\mathbf{A}$ such that for any elements $a_1, \ldots, a_k \in A$, the tuple $(a_1, \ldots, a_k)$ is in D_k if and only if there exists a $\mathbf{P} \in \mathcal{P}$ such that all of the elements $a_1, \ldots, a_k$ are contained in the universe P of $\mathbf{P}$. In addition, there exists an algorithm that computes a cq-definition of D_k (over $\mathbf{A}$) in polynomial time, when given k as input.*

In the definition of the set $\mathcal{P}$ we may assume that if $\mathbf{P}$, $\mathbf{P}'$ are members, then $P \not\subseteq P'$. This additional property can be arranged by only including in $\mathcal{P}$ those pentagons

whose universes are maximal with respect to inclusion. Doing so will not change the other properties listed in the previous lemma.

Theorem 22 *Let* $\mathbf{A}$ *be a finite relational structure satisfying the conditions described in Lemma 21, and let* $\mathcal{P}$ *be the set of pentagons described there. There exists an oracular pwm-algorithm from* $\mathcal{C}_{\text{CQ-2-PENT}}(\mathcal{P})$ *to* $\mathcal{C}_{\text{CQ}}(\mathbf{A})$.

Essentially, Theorem 22 is proved in the following way. In order to translate a 2-sorted conjunctive query ϕ over pentagons to a conjunctive query ϕ' over $\mathbf{A}$, the relations β and γ are used to simulate the two sorts, and the relation α is used to simulate the behavior of the relation R. Also, in the resulting conjunctive query ϕ', all of the variables are related by the relation D_U (where U is the total number of variables), effectively localizing ϕ' to the pentagons found in the set $\mathcal{P}$.

Proof Let M be a natural number such that $|P| \leq M$ for all $\mathbf{P} \in \mathcal{P}$. For ϕ a 2-sorted conjunctive query over the signature $\{R\}$, let $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_m\}$ be the variables of the first and second sort, respectively, that appear in ϕ. Assume that the free variables of ϕ are $\{x_1, \ldots, x_{n'}\}$ and $\{y_1, \ldots, y_{m'}\}$ where $n' \leq n$ and $m' \leq m$. We construct a conjunctive query ϕ' from ϕ over the signature $\{\alpha, \beta, \gamma\}$ along the lines of the construction found in the proof of Theorem 7 in Bova et al. (2013); but, instead of adding the conjunct

$$\Delta_{n+m+k}(x'_1, \ldots, x'_n, y'_1, \ldots, y'_m, w'_1, \ldots, w'_k),$$

in the construction of ϕ' we add the conjunct

$$D_{n+m+k+M}(x'_1, \ldots, x'_n, y'_1, \ldots, y'_m, w'_1, \ldots, w'_k, v'_1, \ldots, v'_M),$$

where the v'_i's are fresh variables and the relation D is provided by the previous lemma. The resulting formula ϕ' will have as free variables

$$\{x'_1, \ldots, x'_{n'}\} \cup \{y'_1, \ldots, y'_{m'}\} \cup \{v'_1, \ldots, v'_M\}.$$

In giving the oracular pwm-reduction, the parameters s and n (as described in the definition of oracular pwm-reduction) do not play a role, and we omit their mention. To construct an oracular pwm-reduction (f, g, H) from $\mathcal{C}_{\text{CQ-2-PENT}}(\mathcal{P})$ to $\mathcal{C}_{\text{CQ}}(\mathbf{A})$ we define g to be the function that maps the 2-sorted conjunctive query ϕ to the conjunctive query ϕ'. As in the original construction from Bova et al. (2013) it follows that the size of $g(\phi)$ can be bounded by a polynomial in the size of ϕ. A key point, as appears in the statement of Lemma 21, is that a cq-definition of the relation D_k can be constructed in time bounded by some polynomial in k.

Given $\mathbf{P} \in \mathcal{P}$, let b^* be some fixed member of $B_{\mathbf{P}}$ and c^* some fixed member of $C_{\mathbf{P}}$ and let $(p_1, \ldots, p_M)$ be some listing of the elements of P. For the purposes of this discussion we regard P as being equal to the set $B_{\mathbf{P}} \times C_{\mathbf{P}}$, though strictly speaking, P is a subset of A. For assignments $h_1 = (b_1, \ldots, b_n) \in B_{\mathbf{P}}^n$ and $h_2 = (c_1, \ldots, c_m) \in C_{\mathbf{P}}^m$, define $f((\mathbf{P}, (h_1, h_2)))$ to be the tuple

$$((b_1, c^*), \ldots, (b_n, c^*), (b^*, c_1), \ldots, (b^*, c_m), p_1, \ldots, p_M).$$

Clearly $f((\mathbf{P}, (h_1, h_2)))$ can be computed in time bounded by a polynomial in the sizes of h_1 and h_2.

For ϕ a 2-sorted conjunctive query over the signature $\{R\}$, $\mathbf{P} \in \mathcal{P}$, and (h_1, h_2) a sorted assignment of the free variables of ϕ to $\mathbf{P}_2$, it follows from Claim 4 in the proof of Theorem 7 from Bova et al. (2013) (and referred to as just Claim 4 in the remainder of this proof) that if ϕ is true in $\mathbf{P}_2$ under the sorted assignment (h_1, h_2) then in $\mathbf{A}$, the formula $g(\phi) = \phi'$ is true under the assignment $f((\mathbf{P}, (h_1, h_2)))$. Claim 4 applies in this situation, since, using terminology from it, the assignments (h_1, h_2) and $f((\mathbf{P}, (h_1, h_2)))$, when restricted to its first $n + m$ components, *match*. The addition of the last M components to $f((\mathbf{P}, (h_1, h_2)))$ do not affect the satisfaction of ϕ', since they only appear in the D-relation conjunct of ϕ' and all of the elements lie in the set P and so, together will satisfy the D-relation that appears in ϕ'.

Conversely, if ϕ' is satisfied in $\mathbf{A}$ under the assignment $f((\mathbf{P}, (h_1, h_2)))$ then by Claim 4, there is some $\mathbf{P}' \in \mathcal{P}$ such that in $\mathbf{P}'_2$, the formula ϕ is satisfied under the assignment (h_1, h_2). By Lemma 21, all of the elements of A that appear in $f((\mathbf{P}, (h_1, h_2)))$ must all lie in $\mathbf{P}'$ since they jointly satisfy a D-relation. But by our modification of the set $\mathcal{P}$ and the fact that $(p_1, \ldots, p_M)$ is a listing of the elements of P, it follows that $\mathbf{P} = \mathbf{P}'$. Thus we have established that $\mathbf{P}$ satisfies ϕ under the assignment (h_1, h_2) if and only if $\mathbf{A}$ satisfies $g(\phi)$ under the assignment $f((\mathbf{P}, (h_1, h_2)))$.

To complete the construction of the oracular pwm-reduction, if we are given an assignment

$$h = (b_1, \ldots, b_n, c_1, \ldots, c_m, p_1, \ldots, p_M)$$

of elements from A, the algorithm H will reject h if there is no $\mathbf{P} \in \mathcal{P}$ that contains all of the elements that appear in h. Otherwise, let $\mathcal{P}_h$ be the set of pentagons in $\mathcal{P}$ that contains this set of elements. If $\phi(x_1, \ldots, x_n, y_1, \ldots, y_m)$ is a 2-sorted conjunctive query over the signature $\{R\}$, then by Claim 4, $g(\phi) = \phi'$ will be satisfied in $\mathbf{A}$ under the assignment h if and only if there is at least one $\mathbf{P} \in \mathcal{P}_h$ such that ϕ will be satisfied in $\mathbf{P}$ under the sorted assignment (h_1, h_2) that matches the assignment $(b_1, \ldots, b_n, c_1, \ldots, c_m)$ in P. By this we mean that $h_1 = (b'_1, \ldots, b'_n)$ and $h_2 = (c'_1, \ldots, c'_m)$, where, regarding the b_i and c_j as elements of $P = B_{\mathbf{P}} \times C_{\mathbf{P}}$, we have $b_i = (b'_i, v_i)$ and $c_j = (u_j, c'_j)$ for some elements v_i and u_j.

So, given the assignment h, and a conjunctive query ϕ, the algorithm H will, for each $\mathbf{P} \in \mathcal{P}_h$ compute the matching assignment (h_1, h_2) and query whether ϕ is satisfied in $\mathbf{P}$ under this assignment. If any of the queries are true, then H will return true to the query testing for the satisfaction of ϕ' in $\mathbf{A}$ under the assignment h. Otherwise, H will return false. Since the number of oracle calls that H will make is bounded by the fixed size of $\mathcal{P}$ and since each matching assignment can be quickly computed, the run time of H can be bounded by a polynomial in the size of h. ■

7. Hardness From the Absence of Taylor Polymorphisms

In this section we prove the following theorem.

Theorem 23 *If $\mathbf{B}$ is a finite relational structure that does not have a Taylor polymorphism then there is an oracular pwm-reduction from $\mathcal{C}_{\exists\text{CIRC}}$ to $\mathcal{C}_{\text{CQ}}(\mathbf{B})$.*

Our proof is based on the proof of Theorem 4 from Bova et al. (2013) and makes use of the fact that a finite relational structure $\mathbf{B}$ fails to have a Taylor polymorphism if and only if the variety generated by $\mathbb{A}(\mathbf{B})$ admits a certain kind of "bad" local behaviour. The theory developed by Hobby and McKenzie (Hobby and McKenzie, 1988), known as tame congruence theory, provides a way to analyze the local structure of finite algebras and identifies fives different types of local behaviour that a finite algebra can exhibit. These five types are: (1) the unary type, (2) the affine type, (3) the boolean type, (4) the lattice type, and (5) the semilattice type. A surprisingly useful invariant of a finite algebra $\mathbb{A}$, called the typeset of $\mathbb{A}$, is the set of types of local behaviour that $\mathbb{A}$ exhibits. The notion of a typeset can be extended to a variety $\mathcal{V}$ (or to any collection of algebras) by taking it to be the union of the typesets of its finite members.

Tame congruence theory establishes, for varieties $\mathcal{V}$ generated by a finite algebra, strong connections between the local behaviour of the finite members of $\mathcal{V}$ and global properties of $\mathcal{V}$. Chapter 9 of Hobby and McKenzie (1988) presents several theorems of this sort, including the following:

Theorem 24 *(Hobby and McKenzie, 1988, see Theorem 9.6) Let $\mathbb{A}$ be a finite algebra. The following are equivalent:*

- *$\mathbb{A}$ has a Taylor term,*
- *The typeset of $\mathcal{V}(\mathbb{A})$ omits the unary type.*

In the proof of Theorem 23 we will use this result to conclude that in the absence of a Taylor polymorphism, some finite member of the variety generated by $\mathbb{A}(\mathbf{B})$ will locally behave as a unary algebra, i.e., an algebra whose basic operations depend on at most one variable. As detailed in the proof, this will provide a way to produce an oracular pwm-reduction from $\mathcal{C}_{\text{CQ}}(\mathbf{B}_{3\text{SAT}})$ to $\mathcal{C}_{\text{CQ}}(\mathbf{B})$.

Recall the structure $\mathbf{B}_{3\text{SAT}}$ from Example 1. We will make use of the following fact, which follows immediately from the standard assignment-preserving conversion from a Boolean circuit to a 3-SAT formula where existential quantification may be used.

Proposition 25 *There exists an oracular pwm-reduction from $\mathcal{C}_{\exists\text{CIRC}}$ to $\mathcal{C}_{\text{CQ}}(\mathbf{B}_{3\text{SAT}})$.*

We also need the following fact.

Proposition 26 *There is a finite relational structure $\mathbf{C}$ with domain $\{0,1\}$ such that each relation $R \subseteq \{0,1\}^n$ of $\mathbf{C}$ contains the constant tuples $(0,0,\ldots,0)$ and $(1,1,\ldots,1)$ and such that there is an oracular pwm-reduction from $\mathcal{C}_{\text{CQ}}(\mathbf{B}_{3\text{SAT}})$ to $\mathcal{C}_{\text{CQ}}(\mathbf{C})$.*

Proof We apply the construction from Lemma 4 from Bova et al. (2012) to the structure $\mathbf{B}_{3\text{SAT}}$ to obtain the desired structure $\mathbf{C}$. For each ternary relation R of $\mathbf{B}_{3\text{SAT}}$, $\mathbf{C}$ will have a 5-ary relation R' such that

$$R' = \{(0,1,a,b,c) \mid (a,b,c) \in R\} \cup \{(0,0,0,0,0),(1,1,1,1,1)\}.$$

For any conjunctive query $\phi(x_1,\ldots,x_n)$ of $\mathbf{B}_{3\text{SAT}}$ we construct a corresponding conjunctive query $\phi'(w_0,w_1,x_1,\ldots,x_n)$ of $\mathbf{C}$ by replacing each occurrence of a relation $R(x,y,z)$ in ϕ with $R'(w_0,w_1,x,y,z)$. It is elementary to show that for any n-tuple $\bar{b} = (b_1,\ldots,b_n) \in \{0,1\}^n$, $\phi(\bar{b})$ holds in $\mathbf{B}_{3\text{SAT}}$ if and only if $\phi'(0,1,\bar{b})$ holds in $\mathbf{C}$. Furthermore if each variable x_i appears in at least one conjunct of ϕ, then the only other $(n+2)$-tuples that satisfy ϕ' are the constant tuples $(0,0,\ldots,0)$ and $(1,1,\ldots,1)$. From this it follows that there is an oracular pwm-reduction from $\mathcal{C}_{\text{CQ}}(\mathbf{B}_{3\text{SAT}})$ to $\mathcal{C}_{\text{CQ}}(\mathbf{C})$. ■

Proof [Proof of Theorem 23] Suppose that $\mathbf{B}$ is a finite relational structure that does not have a Taylor polymorphism. Then by Lemma 9.4 and Theorem 9.6 from Hobby and McKenzie (1988), the variety $\mathcal{V}$ generated by $\mathbb{A}(\mathbf{B})$ admits the unary type. Let $\mathbf{C}$ be a structure as described in Lemma 26. We argue that there is an oracular pwm-reduction from $\mathcal{C}_{\text{CQ}}(\mathbf{C})$ to $\mathcal{C}_{\text{CQ}}(\mathbf{A})$, where $\mathbf{A}$ is the finite structure from Lemma 1 from Bova et al. (2013). Since $\mathbf{A}$ is compatible with a finite algebra $\mathbb{A} \in \mathcal{V}$ then the Theorem follows from Propositions 3 and 5.

In addition to providing the structure $\mathbf{A}$ and the algebra $\mathbb{A}$, Lemma 1 also gives a polynomial-time construction that on input a conjunctive query ϕ over $\mathbf{C}$, produces a conjunctive query ϕ' over $\mathbf{A}$ that has the same sets of free and bound variables that ϕ has. If $X = \{x_1,\ldots,x_m\}$ are the free variables of ϕ (and of ϕ') then Proposition 2 from Bova et al. (2013) proves that for any assignment $g : X \to \{0,1\}$, $\mathbf{C}, g \models \phi$ if and only if $\mathbf{A}, g \models \phi'$. Conversely, for any $g : X \to A$, the equivalence of the following two statements is established in the proof of Proposition 2:

- $\mathbf{A}, g \models \phi'$ and
- $\mathbf{A}, g \models E_m$ and for every unary polynomial $p(x)$ of $\mathbb{A}$, if $p(g(x)) \in \{0,1\}$ for all $x \in X$, then $\mathbf{C}, p \circ g \models \phi$.

We use the above to produce an oracular pwm-reduction from $\mathcal{C}_{\text{CQ}}(\mathbf{C})$ to $\mathcal{C}_{\text{CQ}}(\mathbf{A})$ as follows. We need to design an algorithm H such that on input a conjunctive query $\phi(x_1,\ldots,x_m)$ over $\mathbf{C}$ and an assignment $g : \{x_1\ldots,x_m\} \to A$, decides if $\mathbb{A}, g \models \phi'$. Using the equivalence, it suffices to have H decide if $\mathbf{A}, g \models E_m$ and, for each unary polynomial $p(x)$ of $\mathbb{A}$ with $p(g(x_i)) \in \{0,1\}$ for all $1 \le i \le m$, if $\mathbf{C}, p \circ g \models \phi$. From Lemma 1 from Bova et al. (2013) it follows that determining if $\mathbf{A}, g \models E_m$ can be checked in time bounded by a polynomial in m, since for $m > k = |A|$, E_m is equal to the conjunction of $\binom{m}{k}$ instances of E_k, one for each k-element subset of variables from $\{x_1,\ldots,x_m\}$.

Since $\mathbb{A}$ is a finite algebra, then the set of unary polynomials $p(x)$ of $\mathbb{A}$ is also finite (of size bounded by k^k). For each such polynomial $p(x)$, H checks to see if $p(g(x_i)) \in \{0,1\}$ for all $1 \le i \le m$. For such a polynomial, H queries whether $\mathbf{C}, p \circ g \models \phi$. If any of these queries are false, then H returns false, otherwise, H returns true. Then $\mathbf{A}, g \models \phi'$ if and only if H returns true on input ϕ and g. The number of queries that H makes is bounded by the constant k^k and the run-time of H is bounded by m and the size of ϕ. ■

8. Open Issues

Let us conclude by highlighting two open issues. First, we ask if one can provide evidence, even of the conditional sort, that the dichotomy of Theorem 15 is a true dichotomy. Second, it would be desirable to pin down the exact complexity of the meta-problems of deciding whether or not an input relational structure satisfies each of the dichotomy conditions; we exhibited upper bounds on these meta-problems in Theorem 17 and the surrounding discussion.

Acknowledgments

An extended abstract of this work appeared in the Proceedings of the 28th Conference on Learning Theory (Chen and Valeriote, 2015). The second author acknowledges the support of the Natural Sciences and Engineering Research Council of Canada.

References

Eric Allender, Michael Bauland, Neil Immerman, Henning Schnoor, and Heribert Vollmer. The complexity of satisfiability problems: Refining Schaefer's theorem. *Journal of Computer and System Sciences*, 75(4):245–254, 2009.

Dana Angluin. Queries and concept learning. *Mach. Learn.*, 2(4):319–342, April 1988. ISSN 0885-6125. doi: 10.1023/A:1022821128753.

Dana Angluin and Michael Kharitonov. When won't membership queries help? *Journal of Computer and System Sciences*, 50(2):336 – 355, 1995. ISSN 0022-0000. doi: https://doi.org/10.1006/jcss.1995.1026.

Dana Angluin, Michael Frazier, and Leonard Pitt. Learning conjunctions of Horn clauses. *Mach. Learn.*, 9(2-3):147–164, July 1992. ISSN 0885-6125. doi: 10.1007/BF00992675.

Marta Arias and Roni Khardon. Learning closed Horn expressions. *Information and Computation*, 178(1):214 – 240, 2002. ISSN 0890-5401. doi: https://doi.org/10.1006/inco.2002.3162.

Libor Barto. Finitely related algebras in congruence modular varieties have few subpowers. *Journal of the European Mathematical Society*, 20, 04 2018. doi: 10.4171/JEMS/790.

Libor Barto and Marcin Kozik. Constraint satisfaction problems solvable by local consistency methods. *J. ACM*, 61(1):3:1–3:19, January 2014. ISSN 0004-5411. doi: 10.1145/2556646.

Libor Barto, Andrei Krokhin, and Ross Willard. Polymorphisms, and how to use them. In Andrei Krokhin and Stanislav Zivny, editors, *The Constraint Satisfaction Problem: Complexity and Approximability*, volume 7 of *Dagstuhl Follow-Ups*, pages 1–44. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2017. ISBN 978-3-95977-003-3.

Joel Berman, Paweł Idziak, Petar Marković, Ralph McKenzie, Matthew Valeriote, and Ross Willard. Varieties with few subalgebras of powers. *Transactions of the American Mathematical Society*, 362(3):1445–1473, 2010.

Arnab Bhattacharyya and Yuichi Yoshida. An algebraic characterization of testable Boolean CSPs. In *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part I*, pages 123–134, 2013.

Maria Luisa Bonet and Samuel Buss. Size-depth tradeoffs for Boolean formulae. *Information Processing Letters*, 49(3):151 – 155, 1994. ISSN 0020-0190. doi: https://doi.org/10.1016/0020-0190(94)90093-0.

Simone Bova, Hubie Chen, and Matthew Valeriote. On the expression complexity of equivalence and isomorphism of primitive positive formulas. *Theory Comput. Syst.*, 50(2):329–353, 2012.

Simone Bova, Hubie Chen, and Matthew Valeriote. Generic expression hardness results for primitive positive formula comparison. *Inf. Comput.*, 222:108–120, 2013.

Nader Bshouty. Exact learning from membership queries: Some techniques, results and new directions. In *Algorithmic Learning Theory - 24th International Conference, ALT 2013, Singapore, October 6-9, 2013. Proceedings*, pages 33–52, 2013.

Nader Bshouty, Jeffrey Jackson, and Christino Tamon. Exploring learnability between exact and PAC. *J. Comput. Syst. Sci.*, 70(4):471–484, 2005.

Andrei Bulatov. The complexity of the counting constraint satisfaction problem. *J. ACM*, 60(5):34:1–34:41, October 2013. ISSN 0004-5411. doi: 10.1145/2528400.

Andrei Bulatov. A dichotomy theorem for nonuniform CSPs. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 319–330, 2017. ISBN 978-1-5386-3464-6. doi: 10.1109/FOCS.2017.37.

Andrei Bulatov, Peter Jeavons, and Andrei Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM Journal on Computing*, 34(3):720–742, 2005.

Andrei Bulatov, Hubie Chen, and Victor Dalmau. Learning intersection-closed classes with signatures. *Theoretical Computer Science*, 382(3):209–220, 2007.

Stanley Burris and H. P. Sankappanavar. *A Course in Universal Algebra*. Springer, 1981.

Hubie Chen. Meditations on quantified constraint satisfaction. In Robert Constable and Alexandra Silva, editors, *Logic and Program Semantics*, volume 7230 of *Lecture Notes in Computer Science*, pages 35–49. Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-29484-6.

Hubie Chen and Benoit Larose. Asking the metaquestions in constraint tractability. *ACM Trans. Comput. Theory*, 9(3):11:1–11:27, October 2017. ISSN 1942-3454. doi: 10.1145/3134757.

Hubie Chen and Matthew Valeriote. Learnability of solutions to conjunctive queries: The full dichotomy. In Peter Grnwald, Elad Hazan, and Satyen Kale, editors, *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pages 326–337, Paris, France, 03–06 Jul 2015. PMLR.

Hubie Chen, Matthew Valeriote, and Yuichi Yoshida. Testing assignments to constraint satisfaction problems. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 525–534, Oct 2016. doi: 10.1109/FOCS.2016.63.

Nadia Creignou, Sanjeev Khanna, and Madhu Sudan. *Complexity Classification of Boolean Constraint Satisfaction Problems*. SIAM Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics, 2001.

Victor Dalmau. A dichotomy theorem for learning quantified Boolean formulas. *Machine Learning*, 35(3):207–224, 1999.

Victor Dalmau and Peter Jeavons. Learnability of quantified formulas. *Theor. Comput. Sci.*, 306(1-3):485–511, 2003.

Tomás Feder and Moshe Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory. *SIAM Journal on Computing*, 28:57–104, 1999.

Ralph Freese and Ralph McKenzie. *Commutator Theory for Congruence Modular Varieties*, volume 125 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 1987.

Ralph Freese and Matthew Valeriote. On the complexity of some Maltsev conditions. *Internat. J. Algebra Comput.*, 19(1):41–77, 2009. ISSN 0218-1967.

David Hobby and Ralph McKenzie. *The Structure of Finite Algebras*, volume 76 of *Contemporary Mathematics*. American Mathematical Society, 1988.

Paweł Idziak, Petar Marković, Ralph McKenzie, Matthew Valeriote, and Ross Willard. Tractability and learnability arising from algebras with few subpowers. *SIAM J. Comput.*, 39(7):3023–3037, 2010.

Jeffrey Jackson and Rocco Servedio. On learning random DNF formulas under the uniform distribution. *Theory of Computing*, 2(1):147–172, 2006.

Peter Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200:185–204, 1998.

Alexandr Kazda. Personal communication, 2014.

Keith Kearnes, Petar Marković, and Ralph McKenzie. Optimal strong Mal'cev conditions for omitting type 1 in locally finite varieties. *Algebra universalis*, 72(1):91–100, aug 2014. ISSN 0002-5240, 1420-8911. doi: 10.1007/s00012-014-0289-9.

Michael Kearns and Leslie Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *J. ACM*, 41(1):67–95, 1994.

Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, Apr 1988.

Miklós Maróti and Ralph McKenzie. Existence theorems for weakly symmetric operations. *Algebra Universalis*, 59(3-4):463–489, 2008. ISSN 0002-5240. doi: 10.1007/s00012-008-2122-9.

Ralph McKenzie, George McNulty, and Walter Taylor. *Algebras, Lattices, Varieties, vol. 1*. Wadsworth & Brooks/Cole, 1987.

Leonard Pitt and Manfred Warmuth. Prediction-preserving reducibility. *J. Comput. Syst. Sci.*, 41(3):430–467, 1990.

Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pages 245–254, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-047-0. doi: 10.1145/1374376.1374414.

Thomas Schaefer. The complexity of satisfiability problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, STOC '78, pages 216–226, New York, NY, USA, 1978. ACM. doi: 10.1145/800133.804350.

Mark Siggers. A strong Mal'cev condition for locally finite varieties omitting the unary type. *Algebra Universalis*, 64(1-2):15–20, 2010. ISSN 0002-5240. doi: 10.1007/s00012-010-0082-3.

Walter Taylor. Varieties obeying homotopy laws. *Canadian Journal of Mathematics*, 29(3):498–527, 1977. ISSN 1496-4279. doi: 10.4153/CJM-1977-054-9.

Dmitriy Zhuk. The proof of CSP dichotomy conjecture. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 331–342, 2017. doi: 10.1109/FOCS.2017.38.

Journal of Machine Learning Research 19 (2018) 1-55 Submitted 12/17; Revised 11/18; Published 5/19

Variance-based Regularization with Convex Objectives

John Duchi JDUCHI@STANFORD.EDU
Department of Statistics and Electrical Engineering
Stanford University
Stanford, CA 94305, USA

Hongseok Namkoong HNAMK@STANFORD.EDU
Department of Management Science and Engineering
Stanford University
Stanford, CA 94305, USA

Editor: Alexander Rakhlin

Abstract

We develop an approach to risk minimization and stochastic optimization that provides a convex surrogate for variance, allowing near-optimal and computationally efficient trading between approximation and estimation error. Our approach builds off of techniques for distributionally robust optimization and Owen's empirical likelihood, and we provide a number of finite-sample and asymptotic results characterizing the theoretical performance of the estimator. In particular, we show that our procedure comes with certificates of optimality, achieving (in some scenarios) faster rates of convergence than empirical risk minimization by virtue of automatically balancing bias and variance. We give corroborating empirical evidence showing that in practice, the estimator indeed trades between variance and absolute performance on a training sample, improving out-of-sample (test) performance over standard empirical risk minimization for a number of classification problems.

Keywords: variance regularization, robust optimization, empirical likelihood

1. Introduction

We propose and study a new approach to risk minimization that automatically trades between bias—or approximation error—and variance—or estimation error. Let $\mathcal{X}$ be a sample space, P_0 a distribution on $\mathcal{X}$, and Θ a parameter space. For a loss function $\ell : \Theta \times \mathcal{X} \to \mathbb{R}$, consider the problem of finding $\theta \in \Theta$ minimizing the risk

$$R(\theta) := \mathbb{E}[\ell(\theta, X)] = \int \ell(\theta, x) dP(x) \tag{1}$$

given a sample $\{X_1, \ldots, X_n\}$ drawn i.i.d. according to the distribution P. Under appropriate conditions on the loss ℓ, parameter space Θ, and random variables X, a number of researchers (Bartlett et al., 2005, 2006; Boucheron et al., 2005; Koltchinskii, 2006) have shown results of the form that with high probability,

$$R(\theta) \le \frac{1}{n} \sum_{i=1}^n \ell(\theta, X_i) + C_1 \sqrt{\frac{\mathrm{Var}(\ell(\theta, X))}{n}} + \frac{C_2}{n} \quad \text{for all } \theta \in \Theta \tag{2}$$

©2018 John Duchi and Hongseok Namkoong.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v19/17-750.html.

where C_1 and C_2 depend on the parameters of problem (1) and the desired confidence guarantee. Such bounds justify empirical risk minimization (ERM), which chooses $\widehat{\theta}_n$ to minimize $\frac{1}{n}\sum_{i=1}^n \ell(\theta, X_i)$ over $\theta \in \Theta$. Further, these bounds showcase a tradeoff between bias and variance, where we identify the bias (or approximation error) with the empirical risk $\frac{1}{n}\sum_{i=1}^n \ell(\theta, X_i)$, while the variance arises from the second term in the bound.

Given bounds of the form above and heuristically considering the classical "bias-variance" tradeoff in estimation and statistical learning, it is natural to instead choose θ to directly minimize a quantity trading between approximation and estimation error, say of the form

$$\frac{1}{n} \sum_{i=1}^n \ell(\theta, X_i) + C \sqrt{\frac{\mathrm{Var}_{\widehat{P}_n}(\ell(\theta, X))}{n}}, \tag{3}$$

where $\mathrm{Var}_{\widehat{P}_n}$ denotes the empirical variance of its argument. Maurer and Pontil (2009) considered precisely this idea, giving a number of guarantees on the convergence and good performance of such a procedure. Unfortunately, even when the loss ℓ is convex in θ, the formulation (3) is in general non-convex, yielding computationally intractable problems, which has limited the applicability of procedures that minimize the variance-corrected empirical risk (3). In this paper, we develop an approach that provides a tractable *convex* formulation whenever the loss ℓ is convex and very closely approximates the penalized risk (3). Our approach is based on Owen's empirical likelihood (Owen, 2001) and ideas from distributionally robust optimization (Ben-Tal et al., 2009; Bertsimas et al., 2014; Ben-Tal et al., 2015). Below, we give a number of theoretical guarantees and empirical evidence for its performance.

Before summarizing our contributions, we first describe our approach. Let $\phi : \mathbb{R}_+ \to \mathbb{R}$ be a convex function with $\phi(1) = 0$. Then the *ϕ-divergence* between distributions P and Q defined on a space $\mathcal{X}$ is

$$D_\phi(P \| Q) = \int \phi\left(\frac{dP}{dQ}\right) dQ = \int_{\mathcal{X}} \phi\left(\frac{p(x)}{q(x)}\right) q(x) d\mu(x),$$

where μ is any measure for which $P, Q \ll \mu$, and $p = \frac{dP}{d\mu}$, $q = \frac{dQ}{d\mu}$. Throughout this paper, we use $\phi(t) = \frac{1}{2}(t-1)^2$, which gives the χ^2-divergence (Tsybakov, 2009). Given ϕ and a sample $X_1, \ldots, X_n$, we define the *local neighborhood of the empirical distribution with radius* ρ by

$$\mathcal{P}_n := \left\{ \text{distributions } P \text{ such that } D_\phi\left(P \| \widehat{P}_n\right) \le \frac{\rho}{n} \right\},$$

where $\widehat{P}_n$ denotes the empirical distribution of the sample, and our choice of $\phi(t) = \frac{1}{2}(t-1)^2$ means that $\mathcal{P}_n$ consists of discrete distributions supported on the sample $\{X_i\}_{i=1}^n$. We then define the *robustly regularized risk*

$$R_n(\theta, \mathcal{P}_n) := \sup_{P \in \mathcal{P}_n} \mathbb{E}_P[\ell(\theta, X)] = \sup_P \left\{ \mathbb{E}_P[\ell(\theta, X)] : D_\phi(P \| \widehat{P}_n) \le \frac{\rho}{n} \right\}. \tag{4}$$

As it is the supremum of a family of convex functions, the robust risk $\theta \mapsto R_n(\theta, \mathcal{P}_n)$ is convex in θ whenever ℓ is convex, no matter the value of $\rho \ge 0$. Given the robust empirical risk (4), our proposed estimation procedure is to choose a parameter $\widehat{\theta}_n^{\mathrm{rob}}$ by minimizing $R_n(\theta, \mathcal{P}_n)$.

Let us now discuss a few of the properties of procedures minimizing the robust empirical risk (4). Our first main technical result, which we show in Section 2, is that for bounded loss functions, the robust risk $R_n(\theta, \mathcal{P}_n)$ is a good approximation to the variance-regularized quantity (3). That is,

$$R_n(\theta, \mathcal{P}_n) = \mathbb{E}_{\widehat{P}_n}[\ell(\theta, X)] + \sqrt{\frac{2\rho \mathrm{Var}_{\widehat{P}_n}(\ell(\theta, X))}{n}} + \varepsilon_n(\theta), \tag{5}$$

where $\varepsilon_n(\theta) \leq 0$ and is $O_P(1/n)$ uniformly in θ. We show specifically that whenever $\ell(\theta, X)$ has suitably large variance, with high probability we have $\varepsilon_n = 0$. From variance expansions of the form (5) and empirical Bernstein inequality (2), we see that $R_n(\theta, \mathcal{P}_n)$ is a $O(1/n)$-approximation to the population risk $R(\theta)$, in contrast to the cruder $O(1/\sqrt{n})$-approximation that the empirical risk $\mathbb{E}_{\widehat{P}_n}[\ell(\theta; X)]$ provides. Based on this intuition that the robustly regularized risk $R_n(\theta; \mathcal{P}_n)$ is a tighter approximation to the population risk $R(\theta)$, we show a number of finite-sample convergence guarantees for the estimator

$$\widehat{\theta}_n^{\mathrm{rob}} \in \operatorname*{argmin}_{\theta \in \Theta} \left\{ \sup_P \left\{ \mathbb{E}_P[\ell(\theta, X)] : D_\phi\left(P \| \widehat{P}_n\right) \leq \frac{\rho}{n} \right\} \right\} \tag{6}$$

that are often tighter than those available for ERM (see Section 3). The above problem is a *convex* optimization problem when the original loss $\ell(\cdot; X)$ is convex and Θ is a convex set.

Based on the expansion (5), solutions $\widehat{\theta}_n^{\mathrm{rob}}$ of problem (6) enjoy automatic finite sample optimality certificates: for $\rho \geq 0$, with probability at least $1 - C_1 \exp(-\rho)$ we have

$$R(\widehat{\theta}_n^{\mathrm{rob}}) = E[\ell(\widehat{\theta}_n^{\mathrm{rob}}; X)] \leq R_n(\widehat{\theta}_n^{\mathrm{rob}}; \mathcal{P}_n) + \frac{C_2\rho}{n} = \inf_{\theta \in \Theta} R_n(\theta, \mathcal{P}_n) + \frac{C_2\rho}{n}$$

where C_1, C_2 are constants (which we specify) that depend on the loss ℓ and domain Θ. That is, with high probability the robust solution has risk no worse than the optimal finite sample robust objective up to an $O(\rho/n)$ error term. To guarantee a desired level of risk performance with probability $1 - \delta$, we may specify the robustness penalty $\rho = O(\log \frac{1}{\delta})$.

Secondly, we show that the procedure (6) allows us to automatically and near-optimally trade between approximation and estimation error (bias and variance), so that

$$R(\widehat{\theta}_n^{\mathrm{rob}}) = E[\ell(\widehat{\theta}_n^{\mathrm{rob}}; X)] \leq \inf_{\theta \in \Theta} \left\{ \mathbb{E}[\ell(\theta; X)] + 2\sqrt{\frac{2\rho}{n} \mathrm{Var}(\ell(\theta; X))} \right\} + \frac{C\rho}{n} \tag{7}$$

with high probability. When there are parameters θ with small risk $R(\theta)$ and small variance $\mathrm{Var}(\ell(\theta, X))$, this guarantees that the excess risk $R(\widehat{\theta}_n^{\mathrm{rob}}) - \inf_{\theta \in \Theta} R(\theta)$ is essentially of order $O(\rho/n)$, where ρ governs our desired confidence level. Our bounds do not require the Bernstein-type condition $\mathrm{Var}(\ell(\theta; X)) \leq MR(\theta)$ often required for ERM. Since it is often the case that M depends on global information (e.g. size of parameter space Θ), we have $\mathrm{Var}(\ell(\theta; X)) \ll MR(\theta)$, in which case the bound (7) offers a tighter guarantee than that available for the ERM solution $\widehat{\theta}_n^{\mathrm{erm}}$. In particular, we give an explicit example in Section 3.3 where our robustly regularized procedure (6) converges at rate $O(\log n/n)$ compared to $O(1/\sqrt{n})$ of empirical risk minimization.

Bounds that trade between risk and variance are known in a number of cases in the empirical risk minimization literature (Mammen and Tsybakov, 1999; Tsybakov, 2004; Bartlett et al., 2005; Boucheron et al., 2005; Bartlett et al., 2006; Boucheron et al., 2013; Koltchinskii, 2006), which is relevant when one wishes to achieve "fast rates" of convergence for statistical learning algorithms (that is, faster than the $O(1/\sqrt{n})$ guaranteed by a number of uniform convergence results (Bartlett and Mendelson, 2002; Boucheron et al., 2005, 2013)). In many cases, however, such tradeoffs require either conditions such as the Mammen and Tsybakov's noise condition (Mammen and Tsybakov, 1999; Boucheron et al., 2005) or localization results made possible by curvature conditions that relate the loss/risk and variance (Bartlett et al., 2006, 2005; Mendelson, 2014). The robust solutions (6) enjoy a different tradeoff between variance and risk than that in this literature, but essentially without conditions except compactness of Θ.

In proposing any new estimator, it is essential to understand the limits of the proposed procedure and identify situations in which its performance may be worse than existing estimators. There are indeed situations in which minimizing the robust-regularized risk (4) yields some inefficiency (for example, in classical statistical estimation problems with correctly specified model). To understand limits of the inefficiency induced by using the distributionally-robustified estimator (6), in Section 4 we study explicit finite sample properties of the robust estimator for general stochastic optimization problems, and we also provide asymptotic normality results in classical problems. There are a number of situations, based on growth conditions on the population risk R, when convergence rates faster than $1/\sqrt{n}$ (or even $1/n$) are attainable (see Shapiro et al. (2009, Chapter 5)). We show that under these conditions, the robust procedure (6) still enjoys (near-optimal) fast rates of convergence, similar to empirical risk minimization (also known as sample average approximation in the stochastic programming literature). Our study of asymptotics makes precise the asymptotic efficiency loss of the robust procedure over minimizing the standard (asymptotically optimal) empirical expectation: there is a bias term that scales as $\sqrt{\rho/n}$ in the limiting distribution of $\widehat{\theta}_n^{\mathrm{rob}}$, though its variance is optimal. o

We complement our theoretical results in Section 5, where we conclude by providing three experiments comparing empirical risk minimization strategies to robustly-regularized risk minimization (6). These results validate our theoretical predictions, showing that the robust solutions are a practical alternative to empirical risk minimization. In particular, we observe that the robust solutions outperform their ERM counterparts on "harder" instances with higher variance. In classification problems, for example, the robustly regularized estimators exhibit an interesting tradeoff, where they improve performance on rare classes (where ERM usually sacrifices performance to improve the common cases—increasing variance slightly) at minor cost in performance on common classes.

Related Work

The theoretical foundations of empirical risk minimization are solid (Vapnik, 1998; Bartlett and Mendelson, 2002; Boucheron et al., 2005, 2013). When the expectation of the excess loss bounds its variance, it is possible to achieve faster rates than the $O(1/\sqrt{n})$ offered by standard uniform convergence arguments (Vapnik and Chervonenkis, 1971, 1974; Bartlett et al., 2006; Koltchinskii, 2006; Boucheron et al., 2013) (see Boucheron et al. (2005, Section

5) for an overview in the case of classification, and Shapiro et al. (2009, Chapter 5.3) for more general stochastic optimization problems). Vapnik and Chervonenkis (1971, 1974) first provided such results in the context of $\{0,1\}$-valued losses for classification (see also (Anthony and Shawe-Taylor, 1993)), where the expectation of the loss always upper bounds its variance, so that if there exists a perfect classifier the convergence rates of empirical risk minimization procedures are $O(1/n)$. Mammen and Tsybakov (Mammen and Tsybakov, 1999; Tsybakov, 2004) give low noise conditions for binary classification substantially generalizing these results, which yield a spectrum of fast rates. Under related conditions, Bartlett, Jordan, and McAuliffe (2006) show similar fast rates of convergence for convex risk minimization under appropriate curvature conditions on the loss. The robust procedure (6), on the other hand, is guaranteed to provide an at most $O(1/n)$ over-estimate of the population risk and a small increase of its variance regularized population counterpart. It may be the case that the variance-regularized risk $\inf_\theta\{R(\theta) + \sqrt{\mathrm{Var}(\ell(\theta, X))/n}\}$ decreases to $R(\theta^\star)$ more slowly than $1/n$. As we note above and detail in Section 4, however, in stochastic optimization problems the variance-regularized approach (6) suffers limited degradation with respect to empirical risk minimization strategies, even under convexity and curvature properties that allow faster rates of convergence than those achievable in classical regimes, as detailed by (Shapiro et al., 2009, Chapter 5.3).

Most related to our work is that of Maurer and Pontil (2009), who propose directly regularizing empirical risk minimization by variance, providing guarantees similar to ours and giving a natural foundation off of which many of our results build. In their setting, however—as they carefully note—it is unclear how to actually solve the variance-regularized problem, as it is generally non-convex. Shivaswamy and Jebara (2010, 2011) build on this and develop an elegant approach for boosting binary classifiers based on a variance penalty applied to the exponential loss; as it is a boosting approach, their approach provides a coordinate-wise strategy for decreasing the loss, but it is not guaranteed to converge to a global minimizer and applies to classification-like problems. Our approach, handling general stochastic optimization problems, removes these obstructions.

The robust procedure (6) is based on distributionally robust optimization ideas that many researchers have developed (Ben-Tal et al., 2013; Bertsimas et al., 2014; Lam and Zhou, 2015), where the goal (as in robust optimization more broadly (Ben-Tal et al., 2009)) is to protect against all deviations from a nominal data model. In the optimization literature, there is substantial work on tractability of the problem (6), including that of Ben-Tal et al. (2013), who show that the dual of (4) often admits a standard form (such as a second-order cone problem) to which standard polynomial-time interior point methods can be applied. Namkoong and Duchi (2016) develop stochastic-gradient-like procedures for solving the problem (6), which efficiently provide low accuracy solutions (which are still sufficient for statistical tasks). Work on the statistical analysis of such procedures is nascent; Bertsimas, Gupta, and Kallus (2014) and Lam and Zhou (2015) provide confidence intervals for solution quality under various conditions, and Duchi et al. (2016) give asymptotics showing that the optimal robust risk $R_n(\widehat{\theta}_n^{\mathrm{rob}}; \mathcal{P}_n)$ is a calibrated upper confidence bound for $\inf_{\theta\in\Theta} \mathbb{E}[\ell(\theta; X)]$. They and Gotoh et al. (2015) also provide a number of asymptotic results showing relationships between the robust risk $R_n(\theta; \mathcal{P}_n)$ and variance regularization, but they do not leverage these results for guarantees on the solutions $\widehat{\theta}_n^{\mathrm{rob}}$.

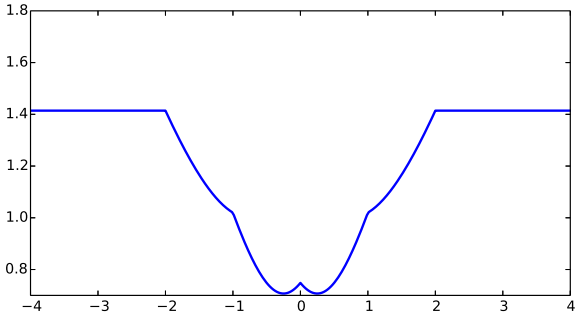


Figure 1. Plot of $\theta \mapsto \sqrt{\mathrm{Var}(\ell(\theta, X))}$ for $\ell(\theta; X) = |\theta - X|$ where $X \sim \mathsf{Uni}(\{-2, -1, 0, 1, 2\})$. The function is non-convex, with multiple local minima, inflection points, and does not grow as $\theta \to \pm\infty$.

Notation We collect our notation here. We let $\mathbb{B}$ denote a unit norm ball in $\mathbb{R}^d$, $\mathbb{B} = \{\theta \in \mathbb{R}^d : \|\theta\| \le 1\}$, where d and $\|\cdot\|$ are generally clear from context. Given sets $A \subset \mathbb{R}^d$ and $B \subset \mathbb{R}^d$, we let $A + B = \{a + b : a \in A, b \in B\}$ denote Minkowski addition. For a convex function f, the subgradient set $\partial f(x)$ of f at x is $\partial f(x) = \{g : f(y) \ge f(x) + g^\top(y - x) \text{ for all } y\}$. For a function $h : \mathbb{R}^d \to \mathbb{R}$, we let h^* denote its Fenchel (convex) conjugate, $h^*(y) = \sup_x\{y^\top x - h(x)\}$. For sequences a_n, b_n, we let $a_n \lesssim b_n$ denote that there is a numerical constant $C < \infty$ such that $a_n \le C b_n$ for all n. For a sequence of random vectors $X_1, X_2, \ldots$, we let $X_n \xrightarrow{d} X_\infty$ denote that X_n converges in distribution to X_∞. For a nonegative sequence $a_1, a_2, \ldots$, we say $X_n = O_P(a_n)$ if $\lim_{c\to\infty} \sup_n \mathbb{P}(\|X_n\| \ge c a_n) = 0$, and we say $X_n = o_P(a_n)$ if $\lim_{c\to 0} \limsup_n \mathbb{P}(\|X_n\| \ge c a_n) = 0$.

2. Variance Expansion

We begin our study of the robust regularized empirical risk $R_n(\theta, \mathcal{P}_n)$ by showing that it is a good approximation to the empirical risk plus a variance term, that is, studying the variance expansion (5). Although the variance of the loss is in general non-convex (see Figure 1 for a simple example), the robust formulation (6) is a convex optimization problem for variance regularization whenever the loss function is convex (the supremum of convex functions is convex (Hiriart-Urruty and Lemaréchal, 1993, Prop. 2.1.2.)).

2.1. Variance expansion for a single variable

To gain intuition for the variance expansion that follows, we begin with a slightly simpler problem, which is to study the quadratically constrained linear maximization problem

$$\underset{p}{\text{maximize}} \ \sum_{i=1}^n p_i z_i \ \ \text{subject to} \ \ p \in \mathcal{P}_n = \left\{p \in \mathbb{R}_+^n : \frac{1}{2}\left\|np - \mathbf{1}\right\|_2^2 \le \rho, \langle \mathbf{1}, p\rangle = 1\right\}, \quad (8)$$

where $z \in \mathbb{R}^n$ is a vector. For simplicity, let $s_n^2 = \frac{1}{n}\|z\|_2^2 - (\overline{z})^2 = \frac{1}{n}\|z - \overline{z}\|_2^2$ denote the empirical "variance" of the vector z, where $\overline{z} = \frac{1}{n}\langle \mathbf{1}, z\rangle$ is the mean value of z. Then by introducing the variable $u = p - \frac{1}{n}\mathbf{1}$, the objective in problem (8) satisfies $\langle p, z\rangle = \overline{z} + \langle u, z\rangle = \overline{z} + \langle u, z - \overline{z}\rangle$ because $\langle u, \mathbf{1}\rangle = 0$. Thus problem (8) is equivalent to solving

$$\operatorname*{maximize}_{u \in \mathbb{R}^n} \ \overline{z} + \langle u, z - \overline{z}\rangle \quad \text{subject to} \ \ \|u\|_2^2 \le \frac{2\rho}{n^2}, \ \ \langle \mathbf{1}, u\rangle = 0, \ u \ge -\frac{1}{n}.$$

Notably, by the Cauchy-Schwarz inequality, we have $\langle u, z - \overline{z}\rangle \le \sqrt{2\rho}\,\|z - \overline{z}\|_2 / n = \sqrt{2\rho s_n^2/n}$, and equality is attained if and only if

$$u_i = \frac{\sqrt{2\rho}(z_i - \overline{z})}{n\,\|z - \overline{z}\|_2} = \frac{\sqrt{2\rho}(z_i - \overline{z})}{n\sqrt{n s_n^2}}.$$

It is possible to choose such u_i while satisfying the constraint $u_i \ge -1/n$ if and only if

$$\min_{i \in [n]} \frac{\sqrt{2\rho}(z_i - \overline{z})}{\sqrt{n s_n^2}} \ge -1. \tag{9}$$

Thus, if inequality (9) holds for the vector z—that is, there is enough variance in z—we have

$$\sup_{p \in \mathcal{P}_n} \langle p, z\rangle = \overline{z} + \sqrt{\frac{2\rho s_n^2}{n}}.$$

For losses $\ell(\theta, X)$ with enough variance relative to $\ell(\theta, X_i) - \mathbb{E}_{\widehat{P}_n}[\ell(\theta, X_i)]$, that is, those satisfying inequality (9), then, we have

$$R_n(\theta, \mathcal{P}_n) = \mathbb{E}_{\widehat{P}_n}[\ell(\theta, X)] + \sqrt{\frac{2\rho \mathrm{Var}_{\widehat{P}_n}(\ell(\theta, X))}{n}}.$$

A slight elaboration of this argument, coupled with the application of a few concentration inequalities, yields the next theorem. The theorem as stated applies only to bounded random variables, but in subsequent sections we relax this assumption by applying the characterization (9) of the exact expansion. As usual, we assume that $\phi(t) = \frac{1}{2}(t-1)^2$ in our definition of the ϕ-divergence.

Theorem 1 *Let Z be a random variable taking values in $[0, M]$. Let $\sigma^2 = \mathrm{Var}(Z)$ and $s_n^2 = \mathbb{E}_{\widehat{P}_n}[Z^2] - \mathbb{E}_{\widehat{P}_n}[Z]^2$ denote the population and sample variance of Z, respectively. Fix $\rho \ge 0$. Then*

$$\left(\sqrt{\frac{2\rho}{n}s_n^2} - \frac{2M\rho}{n}\right)_+ \le \sup_P \left\{\mathbb{E}_P[Z] : D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}\right\} - \mathbb{E}_{\widehat{P}_n}[Z] \le \sqrt{\frac{2\rho}{n}s_n^2}. \tag{10}$$

Moreover, for $n \ge \max\left\{2, \frac{M^2}{\sigma^2}\max\{8\sigma, 44\}\right\}$, with probability at least $1 - \exp\left(-\frac{3n\sigma^2}{5M^2}\right)$

$$\sup_{P : D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[Z] = \mathbb{E}_{\widehat{P}_n}[Z] + \sqrt{\frac{2\rho}{n}s_n^2}. \tag{11}$$

See Section A for the proof of Theorem 1.

Inequality (10) and the exact expansion (11) show that, at least for bounded loss functions ℓ, the robustly regularized risk (4) is a natural (and convex) surrogate for empirical risk plus standard deviation of the loss, and the robust formulation approximates exact variance regularization with a convex penalty. In the sequel, we leverage this result to provide sharp guarantees for a number of stochastic risk minimization problems.

2.2. Uniform variance expansions

We now turn to a more uniform variant of Theorem 1, which depends on familiar notions of function complexity based on Rademacher averages. For a sample $x_1, \ldots, x_n$ and i.i.d. random signs $\varepsilon_i \in \{-1, 1\}$, independent of the x_i, the empirical Rademacher complexity of the class $\mathcal{F}$ is

$$\mathfrak{R}_n(\mathcal{F}) := \mathbb{E}\left[\sup_{f \in \mathcal{F}} \frac{1}{n}\sum_{i=1}^n \varepsilon_i f(x_i)\right].$$

The *worst-case* Rademacher complexity (Srebro et al., 2010) is

$$\mathfrak{R}_n^{\sup}(\mathcal{F}) := \sup_{x_1, \ldots, x_n \in \mathcal{X}} \mathbb{E}\left[\sup_{f \in \mathcal{F}} \left|\frac{1}{n}\sum_{i=1}^n \varepsilon_i f(x_i)\right|\right].$$

For example, when $\mathcal{F}$ is a class of functions bounded by M with VC-subgraph dimension d, we have the inequalities $\mathbb{E}[\mathfrak{R}_n(\mathcal{F})] \le \mathfrak{R}_n^{\sup}(\mathcal{F}) \lesssim M\sqrt{\frac{d}{n}}$. See van der Vaart and Wellner (1996, Chapter 2) and Bartlett and Mendelson (2002) for other bounds.

With this definition, we provide a result showing that the variance expansion (5) holds uniformly for all functions with *enough* variance.

Theorem 2 *Let $\mathcal{F}$ be a collection of bounded functions $f : \mathcal{X} \to [0, M]$, and $M \le n$. There exists a universal constant C such that if $\tau^2 > 0$ satisfies*

$$\tau^2 \ge \frac{4\rho M^2}{n} + C\left[\mathfrak{R}_n^{\sup}(\mathcal{F})^2 \log^3 n + \frac{M^2}{n}(t + \log\log n)\right].$$

Then with probability at least $1 - 3e^{-t}$

$$\sup_{P : D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[f(X)] = \mathbb{E}_{\widehat{P}_n}[f(X)] + \sqrt{\frac{2\rho}{n}\mathrm{Var}_{\widehat{P}_n}(f(X))} \tag{12}$$

for all $f \in \mathcal{F}$ such that $\mathrm{Var}(f) \ge \tau^2$.

We prove the theorem in Section B. Theorem 2 shows that the variance expansion of Theorem 1 holds uniformly for all functions f with sufficient variance. An asymptotic analogue of the equality (12) for heavier tailed random variables is also possible (Duchi et al., 2016). In the remainder of the section, we provide examples and applications of the theorem.

2.2.1. Linear and margin-based losses

Consider a standard margin-based classification problem (Bartlett and Mendelson, 2002), where we have data pairs $(x, y) \in \mathcal{X} \times \{-1, 1\}$, and $\mathcal{X} \subset \mathbb{R}^d$. Let $\Theta \subset \mathbb{R}^d$ be a norm ball of radius $r(\Theta)$, $\Theta = \{\theta \in \mathbb{R}^d \mid \|\theta\| \le r\}$, and let $\|\cdot\|_*$ be the associated dual norm, assuming also that $\mathcal{X} \subset \{x \in \mathbb{R}^d \mid \|x\|_* \le r(\mathcal{X})\}$. We may then consider the standard loss minimization setting, where for some non-increasing and 1-Lipschitz loss $\ell : \mathbb{R} \to \mathbb{R}_+$, we have the risk

$$R(\theta) := \mathbb{E}\left[\ell(Y \left\langle \theta, X \right\rangle)\right],$$

so that $\ell(y \left\langle x, \theta \right\rangle)$ is the loss suffered by making prediction $\langle \theta, x \rangle$ when the label is y. By taking the function class $\mathcal{F} = \{(x, y) \mapsto \ell(y \left\langle x, \theta \right\rangle) - \ell(0) \mid \theta \in \Theta\}$, in this case, an application of the Ledoux-Talagrand contraction inequality (Ledoux and Talagrand, 1991) implies for any $y_1, x_1, \ldots, y_n, x_n$ that

$$\mathbb{E}\left[\sup_{\theta \in \Theta} \left|\sum_{i=1}^n \varepsilon_i \left[\ell(y_i \left\langle \theta, x_i \right\rangle) - \ell(0)\right]\right|\right] \le \mathbb{E}\left[\sup_{\theta \in \Theta} \left|\sum_{i=1}^n \varepsilon_i \left\langle \theta, x_i \right\rangle\right|\right] \le r(\Theta)\mathbb{E}\left[\left\|\sum_{i=1}^n \varepsilon_i x_i\right\|_*\right]. \quad (13)$$

Example 1 (Euclidean norms) *In the above context, suppose that norm $\|\cdot\|$ is the standard ℓ_2 Euclidean norm so that Θ is contained in an ℓ_2-ball of radius $r(\Theta)$, and $\mathcal{X} \subset \mathbb{R}^d$ in an ℓ_2 ball of radius $r(\mathcal{X})$. Then Jensen's inequality and independence of ε_i's give the bound*

$$\mathbb{E}[\|\sum_{i=1}^n \varepsilon_i x_i\|] \le \sqrt{\mathbb{E}\sum_{j=1}^d \left(\sum_{i=1}^n \varepsilon_i x_{ij}\right)^2} \le r(\mathcal{X})\sqrt{n}.$$

Then, inequality (13) *and Theorem 1 imply that*

$$\sup_{P : D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[\ell(Y \left\langle \theta, X \right\rangle)] = \mathbb{E}_{\widehat{P}_n}[\ell(Y \left\langle \theta, X \right\rangle)] + \sqrt{\frac{2\rho}{n}\mathrm{Var}_{\widehat{P}_n}(\ell(Y \left\langle \theta, X \right\rangle))}$$

for all θ satisfying

$$\mathrm{Var}(\ell(Y \left\langle \theta, X \right\rangle)) \ge \frac{r(\mathcal{X})^2 r(\Theta)^2}{n}\left[4\rho + C\log^3 n + Ct\right],$$

with probability at least $1 - e^{-t}$.

Example 2 (High-dimensional problems) *In high dimensional problems, the Euclidean scaling of Example 1 may be problematic, so that using ℓ_1-constraints is preferred (Bühlmann and van de Geer, 2011). Thus, taking the norm $\|\cdot\|$ in the preceding to be the ℓ_1 norm, so that $\Theta \subset \{\theta \in \mathbb{R}^d \mid \|\theta\|_1 \le r_1(\Theta)\}$ and $\|\cdot\|_* = \|\cdot\|_\infty$, then $\mathbb{E}[\|\sum_{i=1}^n \varepsilon_i x_i\|_\infty] \le r(\mathcal{X})\sqrt{n\log(2d)}$, where $r_\infty(\mathcal{X})$ denotes the ℓ_∞-radius of $\mathcal{X} \subset \mathbb{R}^d$. Thus, if we take the loss class $\mathcal{F} = \{\ell(\langle \theta, \cdot \rangle) - \ell(0) \mid \theta \in \Theta\}$, we obtain*

$$\mathfrak{R}_n^{\sup}(\mathcal{F}) \lesssim \sup_{x_1, \ldots, x_n \in \mathcal{X}} \frac{r_1(\Theta)}{n}\mathbb{E}\left[\left\|\sum_{i=1}^n \varepsilon_i x_i\right\|_\infty\right] \le r_1(\Theta) r_\infty(\mathcal{X})\sqrt{\frac{\log(2d)}{n}}.$$

Then the exact variance expansion (12) *holds with probability at least $1 - e^{-t}$ uniformly over θ satisfying $\mathrm{Var}(\ell(Y \left\langle \theta, X \right\rangle)) \ge \frac{r_1(\Theta)^2 r_\infty(\mathcal{X})^2}{n}[4\rho + C\log d \cdot \log^3 n + Ct]$.*

2.2.2. Covering number guarantees

It is also possible to provide guarantees on the exact variance expansion using standard covering numbers, though careful arguments based on Rademacher complexity can be tighter. We begin by recalling the appropriate notions from approximation theory. Let $\mathcal{V}$ be a vector space and $V \subset \mathcal{V}$ be any collection of vectors in $\mathcal{V}$. Let $\|\cdot\|$ be a (semi)norm on $\mathcal{V}$. We say a collection $v_1, \ldots, v_N \subset \mathcal{V}$ is an *ϵ-cover* of $\mathcal{V}$ if for each $v \in \mathcal{V}$, there exists v_i such that $\|v - v_i\| \le \epsilon$. The *covering number* of V with respect to $\|\cdot\|$ is then

$$N(V, \epsilon, \|\cdot\|) := \inf\left\{N \in \mathbb{N} : \text{ there is an } \epsilon\text{-cover of } V \text{ with respect to } \|\cdot\|\right\}.$$

Now, let $\mathcal{F}$ be a collection of functions $f : \mathcal{X} \to \mathbb{R}$, and define the $L^\infty(\mathcal{X})$ norm on f by

$$\|f - g\|_{L^\infty(\mathcal{X})} := \sup_{x \in \mathcal{X}} |f(x) - g(x)|.$$

We also relax our covering number requirements to empirical ℓ_∞-covering numbers as follows. Define $\mathcal{F}(x) = \{(f(x_1), \ldots, f(x_n)) : f \in \mathcal{F}\}$ for $x \in \mathcal{X}^n$, and define the empirical ℓ_∞-covering numbers

$$N_\infty(\mathcal{F}, \epsilon, n) = \sup_{x \in \mathcal{X}^n} N\left(\mathcal{F}(x), \epsilon, \|\cdot\|_\infty\right),$$

which bound the number of ℓ_∞-balls of radius ϵ required to cover $\mathcal{F}(x)$. Note that we always have $N_\infty(\mathcal{F}, \epsilon, n) \le N(\mathcal{F}, \epsilon, \|\cdot\|_{L^\infty(\mathcal{X})})$ by definition. The classical Dudley entropy integral (Dudley, 1999; van der Vaart and Wellner, 1996) shows that, if P_n denotes the point masses on $x_1, \ldots, x_n$ and $\|\cdot\|_{L^2(P_n)}$ the empirical L^2-norm on functions $f : \mathcal{X} \to [-M, M]$, then

$$\begin{aligned}\mathbb{E}\left[\frac{1}{n}\sup_{f \in \mathcal{F}}\left|\sum_{i=1}^n \varepsilon_i f(x_i)\right|\right] &\lesssim \inf_{\delta \ge 0}\left\{\delta + \frac{1}{\sqrt{n}}\int_\delta^M \sqrt{\log N(\mathcal{F}, \epsilon, \|\cdot\|_{L^2(P_n)})}d\epsilon\right\} \\ &\le \inf_{\delta \ge 0}\left\{\delta + \frac{1}{\sqrt{n}}\int_\delta^M \sqrt{\log N_\infty(\mathcal{F}, \epsilon, n)}d\epsilon\right\}. \quad (14)\end{aligned}$$

Our main (essentially standard (van der Vaart and Wellner, 1996)) motivating example is that of Lipschitz loss functions for a parametric set Θ, as follows.

Example 3 *Let $\Theta \subset \mathbb{R}^d$ and assume that $\ell : \Theta \times \mathcal{X} \to [0, M]$ is L-Lipschitz in θ with respect to the ℓ_2-norm for all $x \in \mathcal{X}$, meaning that $|\ell(\theta, x) - \ell(\theta', x)| \le L\|\theta - \theta'\|_2$. Then taking $\mathcal{F} = \{\ell(\theta, \cdot) : \theta \in \Theta\}$, any ϵ-covering $\{\theta_1, \ldots, \theta_N\}$ of Θ in ℓ_2-norm guarantees that $\min_i |\ell(\theta, x) - \ell(\theta_i, x)| \le L\epsilon$ for all θ, x. That is,*

$$N(\mathcal{F}, \epsilon, \|\cdot\|_{L^\infty(\mathcal{X})}) \le N(\Theta, \epsilon/L, \|\cdot\|_2) \le \left(1 + \frac{\mathrm{diam}(\Theta)L}{\epsilon}\right)^d,$$

where $\mathrm{diam}(\Theta) = \sup_{\theta, \theta' \in \Theta}\|\theta - \theta'\|_2$. Thus ℓ_2-covering numbers of Θ control L^∞-covering numbers of the family $\mathcal{F}$, and we have by the entropy integral (14) *that*

$$\mathfrak{R}_n^{\sup}(\mathcal{F}) \lesssim \sqrt{\frac{d}{n}}\int_0^{\mathrm{diam}(\Theta)L}\sqrt{\log\frac{\mathrm{diam}(\Theta)L}{\epsilon}}d\epsilon \lesssim \mathrm{diam}(\Theta)L\sqrt{\frac{d}{n}}.$$

That is, with high probability, for all θ such that $\mathrm{Var}(\ell(\theta, X)) \ge \frac{4M^2\rho}{n} + \frac{Cd\,\mathrm{diam}(\Theta)^2L^2\log^3 n}{n}$, we have the exact variance expansion (12).

3. Optimization by Minimizing the Robust Loss

Based on the precise variance expansions in the preceding section, it is natural to expect that the robust solution (6) automatically trades between approximation and estimation error. This intuition is accurate, and we show that the robustly regularized objective $R_n(\theta; \mathcal{P}_n)$ overestimates the population risk $R(\theta)$ by at most $O(1/n)$. By virtue of optimizing this tighter approximation—as opposed to the usual $O(1/\sqrt{n})$-approximation given by the empirical risk $\mathbb{E}_{\widehat{P}_n}[\ell(\theta; X)]$—the robustly regularized solution (6) enjoys a number of favorable finite-sample properties, which are not always comparable to those for empirical risk minimization (ERM).

In Section 3.1, we present two versions of our main result that depend on covering numbers and discuss their consequences, and we provide an example where the robustly regularized solution $\widehat{\theta}_n^{\rm rob}$ achieves a tighter excess risk bound compared to those that a straightforward application of localized Rademacher complexities (Bartlett et al., 2005) show that the ERM solution $\widehat{\theta}_n^{\rm erm}$ achieves. As evidenced by the substantial work on Rademacher- and Gaussian-complexity and symmetrization, in some instances covering-number-based arguments do not provide the sharpest scaling (Bartlett and Mendelson, 2002; Bartlett et al., 2005; Srebro et al., 2010); thus, in Section 3.2 we present a version of our main result that depends on localized Rademacher complexities, which can allow more refined uniform concentration bounds than covering numbers. We also provide a concrete (but admittedly somewhat contrived) example where our robustly regularized procedure (6) achieves $R(\widehat{\theta}_n^{\rm rob}) - \inf_{\theta\in\Theta} R(\theta) \lesssim \frac{\log n}{n}$, while empirical risk minimization suffers $R(\widehat{\theta}_n^{\rm erm}) - \inf_{\theta\in\Theta} R(\theta) \gtrsim \frac{1}{\sqrt{n}}$, in Section 3.3. The robust "regularizer" has invariance properties other regularization procedures do not, and we mention these briefly in Section 3.4.

3.1. Covering arguments

Our first guarantee depends on the covering numbers of the function class $\mathcal{F}$ as we describe in Section 2.2.2. While we state our results abstractly, in the loss minimization setting we typically consider the function class $\mathcal{F} := \{\ell(\theta, \cdot) : \theta \in \Theta\}$ parameterized by θ. We have the following theorem, where as usual, we let $\mathcal{F}$ be a collection of functions $f : \mathcal{X} \to [M_0, M_1]$ with $M = M_1 - M_0$.

Theorem 3 *Let $n \geq 8M^2/t$, $t \geq \log 12$, $\epsilon > 0$, and $\rho \geq 9t$. Then with probability at least $1 - 2(3N_\infty(\mathcal{F}, \epsilon, 2n) + 1)e^{-t}$,*

$$\mathbb{E}[f(X)] \leq \sup_{P: D_\phi(P\|\widehat{P}_n) \leq \frac{\rho}{n}} \mathbb{E}_P[f(X)] + \frac{11}{3}\frac{M\rho}{n} + \left(2 + 4\sqrt{\frac{2t}{n}}\right)\epsilon \tag{15}$$

for all $f \in \mathcal{F}$. Defining the empirical minimizer

$$\widehat{f} \in \operatorname*{argmin}_{f\in\mathcal{F}} \left\{ \sup_P \left\{ \mathbb{E}_P[f(X)] : D_\phi(P\|\widehat{P}_n) \leq \frac{\rho}{n} \right\} \right\}$$

we have with the same probability that

$$\mathbb{E}[\widehat{f}(X)] \leq \inf_{f\in\mathcal{F}} \left\{ \mathbb{E}[f] + 2\sqrt{\frac{2\rho}{n}\mathrm{Var}(f)} \right\} + \frac{19M\rho}{3n} + \left(2 + 4\sqrt{\frac{2t}{n}}\right)\epsilon. \tag{16}$$

See Section C for a proof of the theorem. Because uniform L^∞-covering numbers upper bound empirical L^∞-covering numbers, it is immediate that covering $\mathcal{F}$ in $\|\cdot\|_{L^\infty(\mathcal{X})}$ provides an identical result.

3.1.1. Covering bounds: corollaries

We turn to a number of corollaries that expand on Theorem 3 to investigate its consequences. Our first corollary shows that Theorem 3 applies to standard Vapnik-Chervonenkis (VC) classes. As VC dimension is preserved through composition, this result also extends to the procedure (6) in typical empirical risk minimization scenarios.

Corollary 4 *In addition to the conditions of Theorem 3, let $\mathcal{F}$ have finite VC-dimension $\mathsf{VC}(\mathcal{F})$. Then for a numerical constant $c < \infty$, the bounds* (15) *and* (16) *hold with probability at least*

$$1 - \left(c\mathsf{VC}(\mathcal{F})\left(\frac{16Mne}{\epsilon}\right)^{\mathsf{VC}(\mathcal{F})-1} + 2\right)e^{-t}.$$

Proof Let $\|f\|_{L^1(Q)} := \int |f(x)| dQ(x)$ denote the L^1-norm on $\mathcal{F}$ for the probability distribution Q. Then by Theorem 2.6.7 of van der Vaart and Wellner (1996), we have

$$\sup_Q N(\mathcal{F}, \epsilon, \|\cdot\|_{L^1(Q)}) \leq c\mathsf{VC}(\mathcal{F})\left(\frac{8Me}{\epsilon}\right)^{\mathsf{VC}(\mathcal{F})-1}$$

for a numerical constant c. Because $\|x\|_\infty \leq \|x\|_1$, taking Q to be uniform on $x \in \mathcal{X}^{2n}$ yields $N(\mathcal{F}(x), \epsilon, \|\cdot\|_\infty) \leq N(\mathcal{F}, \frac{\epsilon}{2n}, \|\cdot\|_{L^1(Q)})$. The result is immediate. ■

Next, we focus more explicitly on the estimator $\widehat{\theta}_n^{\rm rob}$ defined by minimizing the robust regularized risk (6). Let us assume that $\Theta \subset \mathbb{R}^d$, and that we have a typical linear modeling situation, where a loss h is applied to an inner product, that is, $\ell(\theta, x) = h(\theta^\top x)$. In this case, by making the substitution that the class $\mathcal{F} = \{\ell(\theta, \cdot) : \theta \in \Theta\}$ in Corollary 4, we have $\mathsf{VC}(\mathcal{F}) \leq d$, and we obtain the following corollary. In the corollary, recall the definition (1) of the population risk $R(\theta) = \mathbb{E}[\ell(\theta, X)]$, and the uncertainty set $\mathcal{P}_n = \{P : D_\phi(P\|\widehat{P}_n) \leq \frac{\rho}{n}\}$, and that $R_n(\theta, \mathcal{P}_n) = \sup_{P\in\mathcal{P}_n} \mathbb{E}_P[\ell(\theta, X)]$. By setting $\epsilon = M/n$ in Corollary 4, we obtain the following result.

Corollary 5 *Let the conditions of the previous paragraph hold and let $\widehat{\theta}_n^{\rm rob} \in \operatorname{argmin}_{\theta\in\Theta} R_n(\theta, \mathcal{P}_n)$. Assume also that $\ell(\theta, x) \in [0, M]$ for all $\theta \in \Theta, x \in \mathcal{X}$. Then if $n \geq \rho \geq 9\log 12$,*

$$R(\widehat{\theta}_n^{\rm rob}) \leq R_n(\widehat{\theta}_n^{\rm rob}, \mathcal{P}_n) + \frac{11M\rho}{3n} + \frac{2M}{n}\left(1 + \sqrt{\frac{\rho}{n}}\right) \leq \inf_{\theta\in\Theta}\left\{ R(\theta) + 2\sqrt{\frac{2\rho}{n}\mathrm{Var}(\ell(\theta; X))} \right\} + \frac{11M\rho}{n}$$

with probability at least $1 - 2\exp(c_1 d\log n - c_2\rho)$, where c_i are universal constants with $c_2 \geq 1/9$.

To give an alternate concrete variant of Corollary 5 and Theorem 3, let $\Theta \subset \mathbb{R}^d$ and recall Example 3. We assume that for each $x \in \mathcal{X}$, $\inf_{\theta\in\Theta} \ell(\theta, x) = 0$ and that ℓ is L-Lipschitz in θ. If $D := \mathrm{diam}(\Theta) = \sup_{\theta,\theta'\in\Theta} \|\theta - \theta'\|_2 < \infty$, then $\ell(\theta, x) \leq L\,\mathrm{diam}(\Theta)$, and

for $\delta > 0$, we define

$$\rho = \log \frac{2}{\delta} + d \log(2nDL). \tag{17}$$

Setting $t = \rho$ and $\epsilon = \frac{1}{n}$ in Theorem 3 and assuming that $\delta \lesssim 1/n$, $D \lesssim n^k$ and $L \lesssim n^k$ for a numerical constant k, choosing $\delta = \frac{1}{n}$ we obtain that with probability at least $1-\delta = 1-1/n$,

$$\mathbb{E}[\ell(\widehat{\theta}_n^{\rm rob}; X)] = R(\widehat{\theta}_n^{\rm rob}) \le \inf_{\theta \in \Theta} \left\{ R(\theta) + C \sqrt{\frac{d \operatorname{Var}(\ell(\theta, X))}{n} \log n} \right\} + C \frac{dLD \log n}{n} \tag{18}$$

where C is a numerical constant.

3.1.2. Examples and heuristic discussion

Unpacking Theorem 3, the first result (15) (and its Corollary 5) provides a high-probability guarantee that the true expectation $\mathbb{E}[\widehat{f}]$ cannot be more than $O(1/n)$ worse than its robustly-regularized empirical counterpart. The second result (16) (and inequality (18)) guarantees convergence of the empirical minimizer to a parameter with risk at most $O(\log n/n)$ larger than the best possible variance-corrected risk.

To illustriate how variance regularization can yield tighter guarantees than empirical risk minimization by optimizing a $O(1/n)$ upper bound on the risk, we now compare the second bound (16) with an analogous result for empirical risk minimization (ERM). We first give a heuristic version, making it more precise in a coming example. For the ERM solution $\widehat{\theta}_n^{\rm erm} \in \operatorname{argmin}_{\theta \in \Theta} \mathbb{E}_{\widehat{P}_n}[\ell(\theta; X)]$, one common assumption is an upper bound of the variance by the risk; for example, when the losses take values in $[0, M]$, one has $\operatorname{Var}(\ell(\theta, X)) \le MR(\theta)$. In such cases, there is typically some complexity measure $\mathfrak{Comp}_n$ associated with the class of functions being learned, and it is possible to achieve bounds of the form

$$R(\widehat{\theta}_n^{\rm erm}) \le R(\theta^\star) + C \sqrt{\frac{\mathfrak{Comp}_n M R(\theta^\star)}{n}} + C \frac{\mathfrak{Comp}_n M}{n} \tag{19}$$

where $\theta^\star \in \operatorname{argmin}_{\theta \in \Theta} R(\theta)$, a type of result common for bounded nonnegative losses (Boucheron et al., 2005; Vapnik and Chervonenkis, 1971; Vapnik, 1998). For example, for classes of functions of VC-dimension d, we typically have $\mathfrak{Comp}_n \lesssim d \log \frac{n}{d}$. In this caricature, when $\operatorname{Var}(\ell(\theta^\star, X)) \ll MR(\theta^\star)$ and $\rho \gtrsim \mathfrak{Comp}_n$, the optimality guarantee (16) for variance regularization can be tighter than its ERM counterpart (19). This bound is certainly not always sharp, but yields minimax optimal rates in some cases.

Example 4 (Well-specified least-absolute-deviation regression) *For the least-absolute-deviation (LAD) regression, we compare rates of convergence for the ERM solution given by the localized Rademacher complexity against those for the robust solution. Let $Z = (X, Y) \in \mathbb{R}^d \times \mathbb{R}$, where $X \in \{x \in \mathbb{R}^d \mid \|x\|_2 \le L\}$, and let $D := \operatorname{diam}(\Theta)$ be the ℓ_2-diameter of Θ. The LAD loss is $\ell(\theta; (x, y)) := |y - \langle \theta, x \rangle|$, where we assume that $Y = \langle \theta^\star, X \rangle + \epsilon$ for some $\theta^\star \in \Theta$, and random noise $\epsilon \in [-B, B]$ independent of X. We then have the global bound $\ell(\theta; (X, Y)) \le DL + B =: M$. Suppose for simplicity that ϵ is uniform on $[-B, B]$; then $\theta^\star = \operatorname{argmin}_{\theta \in \Theta} R(\theta)$ and $R(\theta^\star) = \mathbb{E}[\ell(\theta^\star; Z)] = \frac{1}{2}B$. In this case,*

$$\operatorname{Var}\left(\ell(\theta^\star; Z)\right) = \frac{B^2}{12} \le \frac{1}{2}(DL + B)B = M \mathbb{E}[\ell(\theta^\star; Z)] = MR(\theta^\star).$$

Using that the loss is 1-Lipschitz, the L^∞ covering numbers for the set of functions $\mathcal{F} := \{f_\theta(x, y) = |\langle \theta, x \rangle - y| \mid \theta \in \Theta\}$ satisfy $\log N(\mathcal{F}, \epsilon, \|\cdot\|_{L^\infty(\mathcal{X})}) \lesssim d \log \frac{DL}{\epsilon}$, and so applying the bound (18) for the robustly regularized solution $\widehat{\theta}_n^{\rm rob}$ with $\epsilon = DL/n$, we obtain

$$R(\widehat{\theta}_n^{\rm rob}) \le R(\theta^\star) + C \sqrt{\frac{d \log n}{n} B^2} + C \frac{d(LD + B) \log n}{n}$$

with probability at least $1 - 1/n$. On the other hand, even an "optimistic" (but naive) ERM bound, achieved by taking $\mathfrak{Comp}_n \lesssim 1$ in the bound (19), yields

$$R(\widehat{\theta}_n^{\rm erm}) \le R(\theta^\star) + C \sqrt{\frac{\log n}{n}(BDL + B^2)} + C \frac{(LD + B) \log n}{n}$$

with probability at least $1-1/n$. We see that leading term for the robustly regularized solution $\widehat{\theta}_n^{\rm rob}$ only depends on the noise-level B^2 while the corresponding term for the ERM solution $\widehat{\theta}_n^{\rm erm}$ depends on global information like the size of the parameter space D, and a uniform bound over covariates L. For typical VC and other d-dimensional classes, the bound $\mathfrak{Comp}_n$ scales linearly in d (cf. (Bartlett et al., 2005, Corollary 3.7), in which case the bound (19) scales as $R(\theta^\star) + C\sqrt{d(BDL + B^2) \log n/n} + O(\log n/n)$, which is worse.

Example 5 (A hard median estimation problem) *To give a bit more insight into the behavior of the robust estimator, consider the simple 1-dimensional median problem, where $\ell(\theta; x) = |\theta - x|$, and assume that $x \in \{-B, B\}$ with $\mathbb{P}(X = B) = \frac{1+\delta}{2}$ for some $\delta > 0$, so that $\theta^\star = \operatorname{argmin} R(\theta) = B$ and $R(\theta^\star) = (1 - \delta)B$. In this case, taking $\theta_0 = 0$ yields $\operatorname{Var}(\ell(\theta; X)) = 0$ and $R(\theta_0) - R(\theta^\star) = \delta B$. For δ small (on the order of $1/\sqrt{n}$), with constant probability the empirical risk minimizer is $\widehat{\theta}_n^{\rm erm} = -B$, yielding risk $R(\widehat{\theta}_n^{\rm erm}) - R(\theta^\star) = 2\delta B$. On the other hand, with high probability $\widehat{\theta}_n^{\rm rob} \ge 0$ (because $\operatorname{Var}(\ell(\theta_0; X)) = 0$ as $\ell(0; X) \equiv B$), and so $R(\widehat{\theta}_n^{\rm rob}) - R(\theta^\star) \le \delta B$. This gap is of course small, but it shows that the robust solution is more conservative: it chooses $\widehat{\theta}_n^{\rm rob}$ so that large losses (of scale $2B$) are less frequent.*

When the population problem is "easy", it is often possible to achieve faster rates of convergence than the usual $O(1/\sqrt{n})$ rate. The simplest scenario where this occurs is if the problem is realizable $R(\theta^\star) = 0$, in which case $\widehat{\theta}_n^{\rm erm}$ has excess risk of the order $O(\log n/n)$; see the bound (19). The robustly regularized solution $\widehat{\theta}_n^{\rm rob}$ enjoys the same faster rates of convergence under the more general condition that $\operatorname{Var}(\ell(\theta^\star; X))$ is small. As a concrete instance of this, let $\ell(\theta; X) \in [0, M]$ and assume that $\ell(\theta; X)$ satisfies the conditions of the first part of Example 3, and let the problem be realizable $R(\theta^\star) = 0$. Since $\operatorname{Var}(\ell(\theta; X)) \le MR(\theta)$, we have from the bounds (18) and (19) that

$$R(\widehat{\theta}_n^{\rm erm}) \le \frac{CdDL \log n}{n} \quad \text{and} \quad R(\widehat{\theta}_n^{\rm rob}) \le \frac{CdDL \log n}{n}.$$

For example, $\operatorname{Var}(\ell(\theta; X)) = 0$ allows for the existence of some $\theta_0 \in \Theta$ such that $\ell(\theta_0; X) < \ell(\theta^\star; X)$ with positive probability.

3.2. Localized Rademacher Complexity

A somewhat more sophisticated approach to concentration inequalities and generalization bounds is based on localization ideas, motivated by the fact that near the optimum of an empirical risk, the complexity of the function class may be smaller than over the entire (global) class (van der Vaart and Wellner, 1996; Bartlett et al., 2005). With this in mind, we now present a refined version of Theorem 3 that depends on localized Rademacher averages.

The starting point for this approach is a notion of localized Rademacher complexity (we give a slightly less general notion than Bartlett et al. (2005), as it is sufficient for our derivations). For a function class $\mathcal{F}$ of functions $f : \mathcal{X} \to \mathbb{R}$, the localized Rademacher complexity at level r is

$$\mathbb{E}\left[\mathfrak{R}_n\left(\left\{cf \mid f \in \mathcal{F}, c \in [0,1], \mathbb{E}[c^2 f^2 \le r]\right\}\right)\right].$$

In addition, we require a few analytic notions, beginning with *sub-root* functions, where we recall (Bartlett et al., 2005) that a function $\psi : \mathbb{R}_+ \to \mathbb{R}_+$ is *sub-root* if it is nonnegative, nondecreasing, and $r \mapsto \psi(r)/\sqrt{r}$ is nonincreasing for all $r > 0$. Any non-constant sub-root function ψ is continuous and has a unique positive fixed point $r^\star = \psi(r^\star)$, where $r \ge \psi(r)$ for all $r \ge r^\star$. Lastly, we consider upper bounds $\psi_n : \mathbb{R}_+ \to \mathbb{R}_+$ on the localized Rademacher complexity satisfying

$$\psi_n(r) \ge \mathbb{E}[\mathfrak{R}_n(\{cf : f \in \mathcal{F}, c \in [0,1], \mathbb{E}[c^2 f^2] \le r\})], \tag{20}$$

where ψ_n is sub-root. (The localized Rademacher complexity itself is sub-root.) Roots of ψ_n play a fundamental role in providing uniform convergence guarantees, and Bartlett et al. (2005) and Koltchinskii (2006) provide careful analyses of localized Rademacher complexities, with typical results as follows. For a class of functions f with range bounded by 1, for any root $r_n^\star$ of ψ_n, with probability at least $1 - e^{-t}$ we have

$$\mathbb{E}[f] \le \mathbb{E}_{\widehat{P}_n}[f] + \frac{1}{\eta}\mathbb{E}_{\widehat{P}_n}[f] + C(1+\eta)\left(r_n^\star + \frac{1}{n}\right) + \frac{t}{n} \quad \text{for all } f \in \mathcal{F} \text{ and } \eta \ge 0.$$

As an example, when $\mathcal{F}$ is a bounded VC-class, we have $r_n^\star \asymp \frac{\mathsf{VC}(\mathcal{F})\log(n/\mathsf{VC}(\mathcal{F}))}{n}$ (Bartlett et al., 2005, Corollary 3.7).

With this motivation, we have the following theorem.

Theorem 6 *For $M \ge 1$, let $\mathcal{F}$ be a collection of functions $f : \mathcal{X} \to [0, M]$, let ψ_n be a sub-root function bounding the localized complexity* (20)*, and let $r_n^\star \ge \psi_n(r_n^\star)$. Let $t > 0$ be arbitrary and assume that ρ satisfies*

$$\frac{\rho}{n} \ge 8\left(\frac{45M}{n}\left(t + \log\left\lceil \log\frac{n}{t}\right\rceil\right) + 18 r_n^\star\right). \tag{21}$$

Then with probability at least $1 - e^{-t}$,

$$\mathbb{E}[f] \le \left(1 + 2\sqrt{\frac{2\rho}{n}}\right) \sup_{P : D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[f] + \left(13 + 4\sqrt{\frac{2\rho}{n}}\right)\frac{M\rho}{n} \quad \textit{for all } f \in \mathcal{F}. \tag{22}$$

Additionally, if $\widehat{f}$ minimizes $\sup_{P : D_\phi(P\|\widehat{P}_n) \le \rho/n} \mathbb{E}_P[f]$, then with probability at least $1 - 3e^{-t}$,

$$\mathbb{E}[\widehat{f}] \le \left(1 + 2\sqrt{\frac{2\rho}{n}}\right) \inf_{f \in \mathcal{F}} \left(\mathbb{E}[f] + \sqrt{\frac{91\rho}{45n}\mathrm{Var}(f)}\right) + \left(14 + 6\sqrt{\frac{2\rho}{n}}\right)\frac{M(3\rho + t)}{n}. \tag{23}$$

We provide the proof of Theorem 6 in Appendix D. It builds off of and parallels many of the techniques developed by Bartlett, Bousquet, and Mendelson (2005), but we require a bit of care to develop the precise variance bounds we provide.

Let us consider the additional $\sqrt{\frac{\rho}{n}}$ factors in Theorem 6 (as compared to Theorem 3). In general, these terms are negligible to the extent that the variance of f dominates the first moment of the function f—heuristically, in situations in which we expect penalizing the variance to improve performance. Let us make this more precise in a regime where n is large. Letting $f \in \mathcal{F}$, we see that we have the inequality

$$(1 + \sqrt{\rho/n})\left(\mathbb{E}[f] + \sqrt{\frac{\rho}{n}\mathrm{Var}(f)}\right) \le \mathbb{E}[f] + C\sqrt{\frac{\rho}{n}\mathrm{Var}(f)}$$

(for a constant $C > 1 + \sqrt{\rho/n}$) if and only if $(C - 1 - \sqrt{\rho/n})^2\mathrm{Var}(f) \ge \mathbb{E}[f]^2$. Equivalently, as n gets large, this occurs roughly when $\mathbb{E}[f^2] \ge \frac{C^2 - 2C + 2}{C^2 - 2C + 1}\mathbb{E}[f]^2$, which holds for large enough C whenever $\mathrm{Var}(f) > 0$.

In some scenarios, we can obtain substantially tighter bounds by using localized Rademacher averages instead of the covering number arguments considered in Section 3.1. (Recall also the discussion following Theorem 2.) To illustriate this point, we consider the case where $\mathcal{F}$ is a bounded subset of a reproducing kernel Hilbert space generated by some sufficiently nice kernel K; even for the Gaussian kernel $K(x, z) = \exp(-\frac{1}{2}\|x - z\|^2)$, log covering numbers for such function spaces grow at least exponentially in the dimension (Zhou, 2003; Kühn, 2011).

Example 6 (Reproducing kernels and least-absolute-deviation regression) *We now give an example using a non-parametric class of functionals in which covering number arguments do not apply, as the covering numbers of the associated classes are too large. Let $\mathcal{H}$ be a reproducing kernel Hilbert space (RKHS) with norm $\|\cdot\|_{\mathcal{H}}$ and associated kernel (representer of evaluation) $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Letting P be a distribution on $\mathcal{X}$, Mercer's theorem (e.g. Cristianini and Shawe-Taylor, 2004) implies that the integral operator $T_K : L^2(\mathcal{X}, P) \to L^2(\mathcal{X}, P)$ defined by $T_K(f)(x) = \int K(x, z)dP(z)$ is compact, and $K(x, x') = \sum_{j=1}^\infty \lambda_j \phi_j(x)\phi_j(z)$ where λ_j are the eigenvalues of T in decreasing order and ϕ_j form an orthonormal decomposition of $L^2(\mathcal{X}, P)$.*

Consider now the least absolute deviation (LAD) loss function $\ell(h; x, y) = |h(x) - y|$, defined for $h \in \mathcal{H}$, and let $\mathbb{B}_{\mathcal{H}}$ be the unit $\|\cdot\|_{\mathcal{H}}$-ball of $\mathcal{H}$. Assume additionally that the model is well-specified, and that $y = h^\star(x) + \xi$ for some random variable ξ with $\mathbb{E}[\xi \mid X] = 0$, $\mathbb{E}[\xi^2] \le \sigma^2$, and $h^\star \in \mathbb{B}_{\mathcal{H}}$. Let the function class

$$\{\ell \circ \mathcal{H}\}_{\le r} := \left\{(x, y) \mapsto c\ell(h(x), y) \mid c \in [0,1], c^2\mathbb{E}[\ell(h(X), Y)^2] \le r\right\}.$$

Based on inequality (20)*, we consider the localized complexity*

$$\mathfrak{R}_n(\{\ell \circ \mathcal{H}\}_{\le r}) = \mathbb{E}\left[\frac{1}{n}\sup_{h \in \mathbb{B}_{\mathcal{H}}, c \in [0,1]} \sum \varepsilon_i c\ell(h(x_i), y_i) \mid \mathbb{E}[\ell(h(X), Y)^2] \le r/c^2\right].$$

We claim that

$$\mathfrak{R}_n(\{\ell \circ \mathcal{H}\}_{\le r}) \lesssim \sqrt{r/n} + \left(\frac{1}{n}\sum_{j=1}^{\infty} \min\{\lambda_j, r\}\right)^{\frac{1}{2}}. \tag{24}$$

As this claim is not central to our development—but does show a slightly different localization result based on Gaussian comparison inequalities than available, for example, in Mendelson (2003)—we provide its proof in Appendix G.1.

Let us use inequality (24)*. To apply Theorem 3, we must find a bound on the fixed point of the localized complexity. To give this bound, we require some knowledge on the eigenvalues λ_j, for which there exists a body of work. For example (Mendelson, 2003), the Gaussian kernel $K(x, x') = \exp(-\frac{1}{2}\|x - x'\|_2^2)$ generates a class of smooth functions for which the eigenvalues λ_j decay exponentially, as $\lambda_j \lesssim e^{-j^2}$. Kernel operators underlying Sobolev spaces with different smoothness orders (Birman and Solomjak, 1967; Gu, 2002) typically have eigenvalues scaling as $\lambda_j \lesssim j^{-2\alpha}$ for some $\alpha > \frac{1}{2}$. As a concrete example, the first-order Sobolev (min) kernel $K(x, x') = 1 + \min\{x, x'\}$ generates an RKHS of Lipschitz functions with $\alpha = 1$. In the former case of $\lambda_j \lesssim e^{-j^2}$, $r_n^\star = \frac{\sqrt{\log n}}{n}$*

$$\left(\frac{1}{n}\sum_{j=1}^{\infty} \min\left\{e^{-j^2}, \frac{\log n}{n}\right\}\right)^{\frac{1}{2}} \approx \left(\frac{1}{n}\sum_{j=1}^{\sqrt{\log n}} \frac{\sqrt{\log n}}{n} + \frac{1}{n}\int_{\sqrt{\log n}}^{\infty} e^{-t^2} dt\right)^{\frac{1}{2}} \lesssim \frac{\sqrt{\log n}}{n} = r_n^\star.$$

In the latter case of polynomially decaying eigenvalues $\lambda_j \lesssim j^{-2\alpha}$, we have $j^{-2\alpha} = r$ when $r^{-\frac{1}{2\alpha}} = j$, so

$$\sum_{j=1}^{\infty} \min\{j^{-2\alpha}, r\} \approx r^{\frac{2\alpha-1}{2\alpha}} + \int_{r^{-1/2\alpha}}^{\infty} t^{-2\alpha} dt \asymp r^{\frac{2\alpha-1}{2\alpha}}.$$

Solving for $nr = r^{\frac{2\alpha-1}{2\alpha}}$, we find the fixed point $(r_n^\star)^{\frac{2\alpha-1}{4\alpha}} = r_n^\star\sqrt{n}$ yields $r_n^\star = n^{-\frac{2\alpha}{2\alpha+1}}$.

Ignoring constants, the above analysis shows that in the case that the kernel eigenvalues scale as $\lambda_j \lesssim e^{-j^2}$, as soon as $\rho \gtrsim \sqrt{\log n}$ we have

$$\mathbb{E}[\ell(h(X), Y)] \le (1+2\sqrt{2\rho/n})\left(\mathbb{E}_{\widehat{P}_n}[\ell(h(X), Y)] + \sqrt{\frac{2\rho}{n}\mathrm{Var}_{\widehat{P}_n}(\ell(h(X), Y))}\right) + \frac{C\rho}{n} \quad \text{for all } h \in \mathbb{B}_{\mathcal{H}}$$

with high probability. In the case of polynomial eigenvalues, if $\widehat{h}$ minimizes the robust empirical loss $\sup_{P: D_\phi(P\|\widehat{P}_n) \le \rho/n} \mathbb{E}_P[\ell(h(X), Y)]$ and $\rho \asymp n^{1-\frac{2\alpha}{2\alpha+1}}$, then

$$\mathbb{E}\left[\ell(\widehat{h}(X), Y)\right] \le \left(1 + Cn^{-\frac{\alpha}{2\alpha+1}}\right) \inf_{h \in \mathbb{B}_{\mathcal{H}}} \left(\mathbb{E}[\ell(h(X), Y)] + Cn^{-\frac{\alpha}{2\alpha+1}}\sqrt{\mathrm{Var}(\ell(h(X), Y))}\right) + Cn^{-\frac{2\alpha}{2\alpha+1}}.$$

This rate of convergence holds without any assumptions on the smoothness of the distribution of the noise ξ.

3.3. Beating empirical risk minimization

We now provide a concrete example where the robustly regularized estimator $\widehat{\theta}_n^{\mathrm{rob}}$ exhibits a substantial performance gap over empirical risk minimization. In the sequel, we bound the

performance degradation to show that the formulation (6) in general loses little over empirical risk minimization. For intuition in this section, consider the (admittedly contrived) setting in which we replace the loss $\ell(\theta, X)$ with $\ell(\theta, X) - \ell(\theta^\star, X)$, where $\theta^\star \in \mathrm{argmin}_{\theta\in\Theta} R(\theta)$. Then in this case, by taking $\theta = \theta^\star$ in Corollary 5, we have $R(\widehat{\theta}_n^{\mathrm{rob}}) \le R(\theta^\star) + O(1/n)$ with high probability. More broadly, we expect the robustly regularized approach to offer performance benefits in situations in which the empirical risk minimizer is highly sensitive to noise, say, because the losses are piecewise linear, and slight under- or over-estimates of slope may significantly degrade solution quality.

With this in mind, we construct a concrete 1-dimensional example—estimating the median of a discrete distribution supported on $\mathcal{X} = \{-1, 0, 1\}$—in which the robustly regularized estimator has convergence rate $\log n/n$, while empirical risk minimization is at best $1/\sqrt{n}$. Define the loss $\ell(\theta; x) = |\theta - x| - |x|$, and for $\delta \in (0, 1)$ let the distribution P be defined by

$$P(X = 1) = \frac{1-\delta}{2}, \quad P(X = -1) = \frac{1-\delta}{2}, \quad P(X = 0) = \delta. \tag{25}$$

Then for $\theta \in \mathbb{R}$, the risk of the loss is

$$R(\theta) = \delta|\theta| + \frac{1-\delta}{2}|\theta - 1| + \frac{1-\delta}{2}|\theta + 1| - (1 - \delta).$$

By symmetry, it is clear that $\theta^\star := \mathrm{argmin}_\theta R(\theta) = 0$, which satisfies $R(\theta^\star) = 0$. (Note also that $\ell(\theta, x) = \ell(\theta, x) - \ell(\theta^\star, x)$.) Without loss of generality, we assume that $\Theta = [-1, 1]$ in this problem.

Now, consider a sample $X_1, \ldots, X_n$ drawn i.i.d. from the distribution P, let $\widehat{P}_n$ denote its empirical distribution, and define the empirical risk minimizer

$$\widehat{\theta}_n^{\mathrm{erm}} := \operatorname*{argmin}_{\theta\in\mathbb{R}} \mathbb{E}_{\widehat{P}_n}[\ell(\theta, X)] = \operatorname*{argmin}_{\theta\in[-1,1]} \mathbb{E}_{\widehat{P}_n}[|\theta - X|].$$

If too many of the observations satisfy $X_i = 1$ or too many satisfy $X_i = -1$, then $\widehat{\theta}_n^{\mathrm{erm}}$ will be either 1 or -1; for small δ, such events become reasonably probable, as the following lemma makes precise. In the lemma, $\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{1}{2}t^2} dt$ denotes the standard Gaussian CDF. (See Section G.2 for a proof.)

Lemma 7 *Let the loss $\ell(\theta; x) = |\theta - x| - |x|$, $\delta \in [0, 1]$, and X follow the distribution* (25)*. Then $R(\widehat{\theta}_n^{\mathrm{erm}}) - R(\theta^\star) \ge \delta$ with probability at least*

$$2\Phi\left(-\sqrt{\frac{n\delta^2}{1-\delta^2}}\right) - (1-\delta^2)^{\frac{n}{2}}\sqrt{\frac{8}{\pi n}}.$$

On the other hand, we certainly have $\ell(\theta^\star; x) = 0$ for all $x \in \mathcal{X}$, so that $\mathrm{Var}(\ell(\theta^\star; X)) = 0$. Now, consider the bound in Theorem 3. We see that $\log N(\{\ell(\theta, \cdot) : \theta \in \Theta\}, \epsilon, \|\cdot\|_{L^\infty(\mathcal{X})}) \le 2\log\frac{1}{\epsilon}$, and taking $\epsilon = \frac{1}{n}$, we have that if $\widehat{\theta}_n^{\mathrm{rob}} \in \mathrm{argmin}_{\theta\in\Theta} R_n(\theta, \mathcal{P}_n)$, then

$$R(\widehat{\theta}_n^{\mathrm{rob}}) \le R(\theta^\star) + \frac{15\rho}{n} \quad \text{with probability } \ge 1 - 4\exp\left(2\log n - \rho\right).$$

In particular, taking $\rho = 3\log n$, we see that

$$R(\widehat{\theta}_n^{\rm rob}) \le R(\theta^\star) + \frac{45\log n}{n} \quad \text{with probability at least } 1 - \frac{4}{n}.$$

The risk for the empirical risk minimizer, as Lemma 7 shows, may be substantially higher; taking $\delta = 1/\sqrt{n}$ we see that with probability at least $2\Phi(-\sqrt{\frac{n}{n-1}}) - 2\sqrt{2}/\sqrt{\pi e n} \ge 2\Phi(-\sqrt{\frac{n}{n-1}}) - n^{-\frac{1}{2}}$,

$$R(\widehat{\theta}_n^{\rm erm}) \ge R(\theta^\star) + n^{-\frac{1}{2}}.$$

(For $n \ge 20$, the probability of this event is $\ge .088$.) For this (specially constructed) example, there is a gap of nearly $n^{\frac{1}{2}}$ in order of convergence.

3.4. Invariance properties

The robust regularization (4) technique enjoys a number of invariance properties. Standard regularization techniques (such as ℓ_1- and ℓ_2-regularization), which generally regularize a parameter toward a particular point in the parameter space, do not. While we leave deeper discussion of these issues to future work, we make two observations, which apply when $\Theta = \mathbb{R}^d$ is unconstrained. Throughout, we let $\widehat{\theta}_n^{\rm rob} \in \operatorname{argmin}_\theta R_n(\theta, \mathcal{P}_n)$ denote the robustly regularized empirical solution.

First, consider a location estimation problem in which we wish to estimate the minimizer of the expectation of a loss of the form $\ell(\theta, X) = h(\theta - X)$, where $h : \mathbb{R}^d \to \mathbb{R}$ is convex and symmetric about zero. Then the robust solution is by inspection shift invariant, as $\ell(\theta + c, X + c) = \ell(\theta, X)$ for any vector $c \in \mathbb{R}^d$. Concretely, in the example of the previous section, ℓ_1- or ℓ_2-regularization achieve better convergence guarantees than ERM does, but if we shift all data $x \mapsto x + c$, then non-invariant regularization techniques lose efficiency (while the robust regularization technique does not). Second, we may consider a generalized linear modeling problem, in which data comes in pairs $(x, y) \in \mathcal{X} \times \mathcal{Y}$ and $\ell(\theta, (x, y)) = h(y, \theta^\top x)$ for a function $h : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$ that is convex in its second argument. Then $\widehat{\theta}_n^{\rm rob}$ is invariant to invertible linear transformations, in the sense that for any invertible $A \in \mathbb{R}^{d\times d}$,

$$\operatorname*{argmin}_\theta \left\{ \sup_{P: D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[\ell(\theta, (X, Y))] \right\} = \operatorname*{argmin}_\theta \left\{ \sup_{P: D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[\ell(A^{-1}\theta, (AX, Y))] \right\} = \widehat{\theta}_n^{\rm rob}.$$

Our results in this section do not precisely apply as we require unbounded θ, however, the next section shows that localization approaches can address this.

4. Robust regularization cannot be too bad

The previous two sections provide guarantees on the performance of the robust regularized estimator (6), it does not—cannot—dominate classical approaches based on empirical risk minimization (also known as sample average approximation in the stochastic optimization literature), though it can improve on them in some cases. For example, with a correctly specified linear regression model with gaussian noise, least-squares—empirical risk minimization with the loss $\ell(\theta, (x, y)) = \frac{1}{2}(\theta^\top x - y)^2$—is essentially optimal. Our goal in this section is thus to provide more understanding of potential poor behavior of the procedure (6) with

respect to ERM, considering two scenarios. The first is in stochastic (convex) optimization problems, where we investigate the finite-sample convergence rates of the robust solution to the population optimal risk. We show that the robust solution $\widehat{\theta}_n^{\rm rob}$ enjoys fast rates of convergence in cases in which the risk has substantial curvature—precisely as with empirical risk minimization. The second is to consider the asymptotics of the robust solution $\widehat{\theta}_n^{\rm rob}$, where we show that in classical statistical scenarios the robust solution is nearly efficient, though there is an asymptotic bias of order $1/\sqrt{n}$ that scales with the confidence ρ.

4.1. Fast Rates

In cases in which the risk R has curvature, empirical risk minimization often enjoys faster rates of convergence (Boucheron et al., 2005; Shapiro et al., 2009). The robust solution $\widehat{\theta}_n^{\rm rob}$ similarly attains faster rates of convergence in such cases, even with approximate minimizers of $R_n(\theta, \mathcal{P}_n)$. For the risk R and $\epsilon \ge 0$, let

$$S_\star^\epsilon := \left\{ \theta \in \Theta : R(\theta) \le \inf_{\theta^\star \in \Theta} R(\theta^\star) + \epsilon \right\}$$

denote the ϵ-sub-optimal (solution) set, and similarly let

$$\widehat{S}_\star^\epsilon := \left\{ \theta \in \Theta : R_n(\theta, \mathcal{P}_n) \le \inf_{\theta' \in \Theta} R_n(\theta', \mathcal{P}_n) + \epsilon \right\}.$$

For a vector $\theta \in \Theta$, let $\pi_{S_\star}(\theta) = \operatorname{argmin}_{\theta^\star \in S_\star} \|\theta^\star - \theta\|_2$ denote the Euclidean projection of θ onto the set $S_\star$; this projection operator is very useful for showing faster rates of convergence in stochastic optimization (see Shapiro et al. (2009), whose techniques we closely follow). In the statement of the result, for $A \subset \Theta$, we let $\mathfrak{R}_n(A)$ denote the Rademacher complexity of the localized process $\{x \mapsto \ell(\theta; x) - \ell(\pi_{S_\star}(\theta); x) : \theta \in A\}$. We then have the following result, whose proof we provide in Section E.

Theorem 8 *Let Θ be convex and let $\ell(\cdot; x)$ be convex and L-Lipshitz in its first argument for all $x \in \mathcal{X}$. For constants $\lambda > 0$, $\gamma > 1$, and $r > 0$, assume the risk R satisfies*

$$R(\theta) - \inf_{\theta \in \Theta} R(\theta) \ge \lambda \operatorname{dist}(\theta, S_\star)^\gamma \quad \text{for all } \theta \text{ such that } \operatorname{dist}(\theta, S_\star) \le r. \tag{26}$$

Let $t > 0$. If $0 \le \epsilon \le \frac{1}{2}\lambda r^\gamma$ satisfies

$$\epsilon \ge \left(2\frac{8^\gamma L^\gamma}{\lambda}\right)^{\frac{1}{\gamma-1}} \left(\frac{\rho}{n}\right)^{\frac{\gamma}{2(\gamma-1)}} \quad \text{and} \quad \frac{\epsilon}{2} \ge 2\mathbb{E}[\mathfrak{R}_n(S_\star^{2\epsilon})] + L\left(\frac{2\epsilon}{\lambda}\right)^{\frac{1}{\gamma}} \sqrt{\frac{2t}{n}}, \tag{27}$$

then $\mathbb{P}(\widehat{S}_\star^\epsilon \subset S_\star^{2\epsilon}) \ge 1 - e^{-t}$,

We provide a brief discussion of this result as well as a corollary that gives more explicit rates of convergence. First, we note that (by an inspection of the proof) the L-Lipschitz assumption need only hold in the neighborhood $S_\star^{2\epsilon}$ for the result to hold. We also have the following

Corollary 9 *In addition to the conditions of Theorem 8, assume that $S_\star = \{\theta^\star\}$ is a single point and $\Theta \subset \mathbb{R}^d$. Then for any $\epsilon \le \frac{1}{2}\lambda r^\gamma$, we have $\mathbb{P}(\widehat{S}_\star^\epsilon \subset S_\star^{2\epsilon}) \ge 1 - e^{-t}$ for*

$$\epsilon \gtrsim \left(\frac{L^\gamma}{\lambda}\right)^{\frac{1}{\gamma-1}} \left(\frac{d}{n}\log\frac{n}{d} + \frac{t}{n} + \frac{\rho}{n}\right)^{\frac{\gamma}{2(\gamma-1)}}.$$

So long as $\rho \lesssim d \log \frac{n}{d}$, this rate of convergence is as good as that enjoyed by standard empirical risk minimization approaches (Shapiro et al., 2009, Ch. 5) under these types of growth conditions. The case that $\gamma = 2$ corresponds (roughly) to strong convexity, and in this case we get the approximate rate of convergence of $\frac{L^2}{\lambda}\frac{d\log\frac{n}{d}}{n}$, the familiar rate of convergence under these conditions. Of course, if there is too much variance penalization (i.e. ρ is too large), then the rates of convergence may be slower.

Proof That $S_\star$ is a singleton implies that $S_\star^{2\epsilon} \subset \{\theta \mid \|\theta - \theta^\star\| \le (2\epsilon/\lambda)^{\frac{1}{\gamma}}\}$. Moreover, in this case we also have that

$$\left|\mathbb{E}_{\widehat{P}_n}[\ell(\theta; X) - \ell(\theta^\star; X)] - \mathbb{E}_{\widehat{P}_n}[\ell(\theta'; X) - \ell(\theta^\star; X)]\right| \le L\left\|\theta - \theta'\right\|,$$

so that an ϵ/L-cover of $\{\theta \mid \|\theta - \theta^\star\| \le (2\epsilon/\lambda)^{\frac{1}{\gamma}}\}$ is an ϵ-cover of the function class $\mathcal{F} = \{f(x) = \ell(\theta; x) - \ell(\theta^\star; x) \mid \theta \in S_\star^{2\epsilon}\}$ in $\|\cdot\|_{L^2(P_n)}$ norm. Thus, the standard Dudley entropy integral (Dudley, 1999; van der Vaart and Wellner, 1996) yields

$$\begin{aligned}\mathbb{E}[\mathfrak{R}_n(S_\star^{2\epsilon})] &\lesssim \frac{1}{\sqrt{n}}\int_0^\infty \sqrt{\log N(\mathcal{F}, \delta, \|\cdot\|_{L^2(P_n)})}d\delta \\ &\lesssim \frac{1}{\sqrt{n}}\int_0^{L(2\epsilon/\lambda)^{\frac{1}{\gamma}}} \sqrt{d\log\frac{L}{\delta}}d\delta \le L\sqrt{\frac{d}{n}}\left(\frac{2\epsilon}{\lambda}\right)^{\frac{1}{\gamma}}\sqrt{1 + \frac{1}{\gamma}\log\frac{\lambda}{2L^\gamma\epsilon}}\end{aligned}$$

where we have used that $\int_0^\varepsilon \sqrt{\log\frac{L}{\delta}}d\delta \le \varepsilon\sqrt{1 + \log\frac{L}{\varepsilon}}$. Solving for ϵ in the localization inequality (27) then yields the corollary, showing that the specified choice of ϵ is sufficient for all the conditions (27) to hold. ∎

4.2. Asymptotics

It is important to understand the precise limiting behavior of the robust estimator in addition to its finite sample properties—this allows us to more precisely characterize when there may be *degradation* relative to classical risk minimization strategies. With that in mind, in this section we provide asymptotic results for the robust solution (6) to better understand the consequences of penalizing the variance of the loss itself. In particular, we would like to understand efficiency losses relative to (say) maximum likelihood in situations in which maximum likelihood is efficient. Before stating the results, we make a few standard assumptions on the risk $R(\theta)$, the loss ℓ, and the moments of ℓ and its derivatives. Concretely, we assume that

$$\theta^\star := \operatorname*{argmin}_\theta R(\theta) \quad \text{and} \quad \nabla^2 R(\theta^\star) \succ 0,$$

that is, the risk functional has strictly positive definite Hessian at $\theta^\star$, which is thus unique. Additionally, we have the following smoothness assumptions on the loss function, which are satisfied by common loss functions, including the negative log-likelihood for any exponential family or generalized linear model (Lehmann and Casella, 1998). In the assumption, we let $\mathbb{B}$ denote the ℓ_2-ball of radius 1 in $\mathbb{R}^d$.

Assumption A *For some $\epsilon > 0$, there exists a function $L : \mathcal{X} \to \mathbb{R}_+$ satisfying*

$$|\ell(\theta, x) - \ell(\theta', x)| \le L(x)\left\|\theta - \theta'\right\|_2 \quad \text{for } \theta, \theta' \in \theta^\star + \epsilon\mathbb{B}$$

and $\mathbb{E}[L(X)^2] \le L(P) < \infty$. Additionally, there is a function H such that the function $\theta \mapsto \ell(\theta, x)$ has $H(x)$-Lipschitz continuous Hessian (with respect to the Frobenius norm) on $\theta^\star + \epsilon\mathbb{B}$, where $\mathbb{E}[H(X)^2] < \infty$.

Then, recalling the robust estimator (6) as the minimizer of $R_n(\theta, \mathcal{P}_n)$, we have the following theorem, which we prove in Section F.

Theorem 10 *Let Assumption A hold, and let the sequence $\widehat{\theta}_n^{\rm rob}$ be defined by $\widehat{\theta}_n^{\rm rob} \in \operatorname{argmin}_\theta R_n(\theta, \mathcal{P}_n)$. Define*

$$b(\theta^\star) := \frac{\operatorname{Cov}(\nabla_\theta \ell(\theta^\star, X), \ell(\theta^\star, X))}{\sqrt{\operatorname{Var}(\ell(\theta^\star, X))}} \quad \text{and} \quad \Sigma(\theta^\star) = \left(\nabla^2 R(\theta^\star)\right)^{-1}\operatorname{Cov}(\nabla\ell(\theta^\star, X))\left(\nabla^2 R(\theta^\star)\right)^{-1}.$$

Then $\widehat{\theta}_n^{\rm rob} \stackrel{a.s.}{\to} \theta^\star$ and

$$\sqrt{n}(\widehat{\theta}_n^{\rm rob} - \theta^\star) \stackrel{d}{\to} \mathsf{N}\left(-\sqrt{2\rho}\, b(\theta^\star), \Sigma(\theta^\star)\right)$$

The asymptotic variance $\Sigma(\theta^\star)$ in Theorem 10 is generally unimprovable, as made apparent by Le Cam's local asymptotic normality theory and the Hájek-Le Cam local minimax theorems (van der Vaart and Wellner, 1996). Thus, Theorem 10 shows that the robust regularized estimator (6) has some efficiency loss, but it is only in the bias term. We explore this a bit more in the context of the risk of $\widehat{\theta}_n^{\rm rob}$. Letting $W \sim \mathsf{N}(0, \Sigma(\theta^\star))$, as an immediate corollary to this theorem, the delta-method implies that

$$n\left[R(\widehat{\theta}_n^{\rm rob}) - R(\theta^\star)\right] \stackrel{d}{\to} \frac{1}{2}\left\|\sqrt{2\rho}\, b(\theta^\star) + W\right\|^2_{\nabla^2 R(\theta^\star)}, \tag{28}$$

where we recall that $\|x\|_A^2 = x^\top A x$. This follows from a Taylor expansion, because $\nabla R(\theta^\star) = 0$ and so $R(\theta) - R(\theta^\star) = \frac{1}{2}(\theta - \theta^\star)^\top\nabla^2 R(\theta^\star)(\theta - \theta^\star) + o(\|\theta - \theta^\star\|^2)$, or

$$\begin{aligned}n(R(\widehat{\theta}_n^{\rm rob}) - R(\theta^\star)) &= n\left(\frac{1}{2}(\widehat{\theta}_n^{\rm rob} - \theta^\star)^\top\nabla^2 R(\theta^\star)(\widehat{\theta}_n^{\rm rob} - \theta^\star) + o(\|\widehat{\theta}_n^{\rm rob} - \theta^\star\|^2)\right) \\ &= \frac{1}{2}\left(\sqrt{n}(\widehat{\theta}_n^{\rm rob} - \theta^\star)\right)^\top\nabla^2 R(\theta^\star)\left(\sqrt{n}(\widehat{\theta}_n^{\rm rob} - \theta^\star)\right) + o_P(1) \\ &\stackrel{d}{\to} \frac{1}{2}(\sqrt{2\rho}\, b(\theta^\star) + W)^\top\nabla^2 R(\theta^\star)(\sqrt{2\rho}\, b(\theta^\star) + W)\end{aligned}$$

by Theorem 10.

The limiting random variable in expression (28) has expectation

$$\frac{1}{2}\mathbb{E}[\|\sqrt{2\rho}b(\theta^\star) + W\|^2_{\nabla^2 R(\theta^\star)}] = \rho b(\theta^\star)^\top \nabla^2 R(\theta^\star) b(\theta^\star) + \frac{1}{2}\operatorname{tr}(\nabla^2 R(\theta^\star)^{-1}\operatorname{Cov}(\ell(\theta^\star, X)),$$

while the classical empirical risk minimization procedure (standard M-estimation) (Lehmann and Casella, 1998; van der Vaart and Wellner, 1996) has limiting mean-squared error $\frac{1}{2}\operatorname{tr}(\nabla^2 R(\theta^\star)^{-1}\operatorname{Cov}(\ell(\theta^\star, X)))$. Thus there is an additional $\rho\,\|b(\theta^\star)\|^2_{\nabla^2 R(\theta^\star)}$ penalty in the asymptotic risk (at a rate of $1/n$) for the robustly-regularized estimator. An inspection of the proof of Theorem 10 reveals that $b(\theta^\star) = \nabla_\theta \sqrt{\operatorname{Var}(\ell(\theta^\star, X))}$; if the variance of the loss is stable near $\theta^\star$, so that moving to a parameter $\theta = \theta^\star + \Delta$ for some small Δ has little effect on the variance, then the standard loss terms dominate, and robust regularization has asymptotically little effect. On the other hand, highly unstable loss functions for which $\nabla_\theta \sqrt{\operatorname{Var}(\ell(\theta^\star, X))}$ is large yield substantial bias.

We conclude our study of the asymptotics with a (to us) somewhat surprising example. Consider the classical linear regression setting in which $y = x^\top\theta^\star + \varepsilon$, where $\varepsilon \sim \mathsf{N}(0, \sigma^2)$. Using the standard squared error loss $\ell(\theta, (x, y)) = \frac{1}{2}(\theta^\top x - y)^2$, we obtain that

$$\nabla\ell(\theta^\star, (x, y)) = (x^\top\theta^\star - y)x = (x^\top\theta^\star - x^\top\theta^\star - \varepsilon)x = -\varepsilon x,$$

while $\ell(\theta^\star, (x, y)) = \frac{1}{2}\varepsilon^2$. The covariance $\operatorname{Cov}(\varepsilon X, \varepsilon^2) = \mathbb{E}[\varepsilon X(\varepsilon^2 - \sigma^2)] = 0$ by symmetry of the error distribution, and so—in the special classical case of correctly specified linear regression—the bias term $b(\theta^\star) = 0$ for linear regression in Theorem 10. That is, the robustly regularized estimator (6) is asymptotically efficient.

5. Experiments

We present three experiments in this section. The first is a small simulation example, which serves as a proof of concept allowing careful comparison of standard empirical risk minimization (ERM) strategies to our variance-regularized approach. The latter two are classification problems on real datasets; for both of these we compare performance of robust solution (6) to its ERM counterpart.

5.1. Minimizing the robust objective

As a first step, we give a brief description of our (essentially standard) method for solving the robust risk problem. Our work in this paper focuses mainly on the properties of the robust objective (4) and its minimizers (6), so we only briefly describe the algorithm we use; we leave developing faster and more accurate specialized methods to further work. To solve the robust problem, we use a gradient descent-based procedure, and we focus on the case in which the empirical sampled losses $\{\ell(\theta, X_i)\}_{i=1}^n$ have non-zero variance for all parameters $\theta \in \Theta$, which is the case for all of our experiments.

Recall the definition of the subdifferential $\partial f(\theta) = \{g \in \mathbb{R}^d : f(\theta') \geq f(\theta) + \langle g, \theta' - \theta\rangle \text{ for all } \theta'\}$, which is simply the gradient for differentiable functions f. A standard result in convex analysis (Hiriart-Urruty and Lemaréchal, 1993, Theorem VI.4.4.2) is that if the vector $p^* \in \mathbb{R}^n_+$ achieving the supremum in the definition (4) of the robust risk is unique, then

$$\partial_\theta R_n(\theta, \mathcal{P}_n) = \partial_\theta \sup_{P \in \mathcal{P}_n} \mathbb{E}_P[\ell(\theta; X)] = \sum_{i=1}^n p_i^* \partial_\theta \ell(\theta; X_i),$$

where the final summation is the standard Minkowski sum of sets. As this maximizing vector p is indeed unique whenever $\operatorname{Var}_{\widehat{P}_n}(\ell(\theta; X)) \neq 0$, we see that for all our problems, so long as ℓ is differentiable, so too is $R_n(\theta, \mathcal{P}_n)$ and

$$\nabla_\theta R_n(\theta, \mathcal{P}_n) = \sum_{i=1}^n p_i^* \nabla_\theta \ell(\theta; X_i) \;\text{ where }\; p^* = \operatorname*{argmax}_{p \in \mathcal{P}_n} \left\{ \sum_{i=1}^n p_i \ell(\theta; X_i) \right\}. \tag{29}$$

In order to perform gradient descent on the risk $R_n(\theta, \mathcal{P}_n)$, then, by equation (29) we require only the computation of the worst-case distribution p^*. By taking the dual of the maximization (29), this is an efficiently solvable convex problem; for completeness, we provide a procedure for this computation in Section H that requires time $O(n \log n + \log\frac{1}{\epsilon}\log n)$ to compute an ϵ-accurate solution to the maximization (29). As all our examples have smooth objectives, we perform gradient descent on the robust risk $R_n(\cdot, \mathcal{P}_n)$, with stepsizes chosen by a backtracking (Armijo) line search (Boyd and Vandenberghe, 2004, Chapter 9.2).

5.2. Simulation experiment

For our simulation experiment, we use a quadratic loss with linear perturbation. For $v, x \in \mathbb{R}^d$, define the loss $\ell(\theta; x) = \frac{1}{2}\|\theta - v\|_2^2 + x^\top(\theta - v)$. We set $d = 50$ and take $X \sim \mathsf{Uni}(\{-B, B\}^d)$, varying B in the experiment. For concreteness, we let the domain $\Theta = \{\theta \in \mathbb{R}^d : \|\theta\|_2 \leq r\}$ and set $v = \frac{r}{2\sqrt{d}}\mathbf{1}$, so that $v \in \operatorname{int}\Theta$; we take $r = 10$. Notably, standard regularization strategies, such as ℓ_1 or ℓ_2-regularization, pull θ toward 0, while the variance of $\ell(\theta; X)$ is minimized by $\theta = v$ (thus naturally advantaging the variance-based regularization we consider, as $R(v) = \inf_\theta R(\theta) = 0$). Moreover, as X is pure noise, this is an example where we expect variance regularization to be particularly useful. We choose $\delta = .05$ and set ρ as in Eq. (17) (using that ℓ is $(3r + \sqrt{d}B)$-Lipschitz) to obtain robust coverage with probability at least $1 - \delta$. In our experiments, we obtained 100% coverage in the sense of (15), as the high probability bound is conservative.

Figure 2 summarizes the results. The robust solution $\widehat{\theta}_n^{\text{rob}} = \operatorname{argmin}_{\theta\in\Theta} R_n(\theta, \mathcal{P}_n)$ always outperforms the empirical risk minimizer $\widehat{\theta}_n^{\text{erm}} = \operatorname{argmin}_{\theta\in\Theta} \mathbb{E}_{\widehat{P}_n}[\ell(\theta, X)]$ in terms of the true risk $\mathbb{E}[\ell(\theta, X)] = \frac{1}{2}\|\theta - v\|_2^2$. Each experiment consists of 1,200 independent replications for each sample size n and value B. In Tables 1 and 2, we display the risks of $\widehat{\theta}_n^{\text{erm}}$ and $\widehat{\theta}_n^{\text{rob}}$ and variances, respectively, computed for the 1,200 independent trials. The gap between the risk of $\widehat{\theta}_n^{\text{erm}}$ and $\widehat{\theta}_n^{\text{rob}}$ is siginificant at level $p < .01$ for all sample sizes and values of B we considered according to a one-sided T-test. Notice also in Table 2 that the variance of the robust solutions is substantially smaller than that of the empirical risk minimizer—often several orders of magnitude smaller for large sample sizes n. This simulation shows that—in a simple setting favorable to it—our procedure outperforms standard alternatives.

. Code is available at `https://github.com/hsnamkoong/robustopt`.

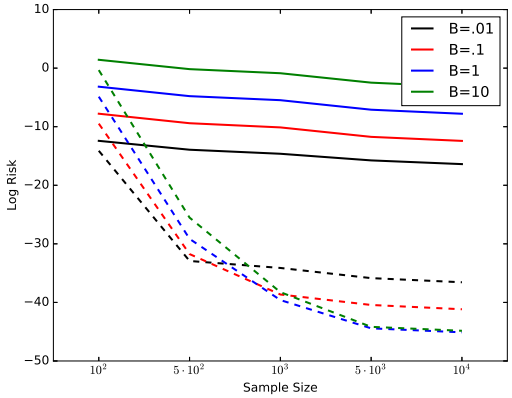


Figure 2. Simulation experiment. $\log \mathbb{E}[R(\widehat{\theta}_n^{\rm erm})]$ is the solid lines, in decreasing order from $B = 10$ (top) to $B = .01$ (bottom). $\log \mathbb{E}[R(\widehat{\theta}_n^{\rm rob})]$ is the dashed line, in the same vertical ordering at sample size $n = 10^2$.

Table 1: Simulation experiment: Mean risks over 1,200 simulations

	$B = .01$		$B = .1$		$B = 1$		$B = 10$	
n	$R(\widehat{\theta}_n^{\rm erm})$	$R(\widehat{\theta}_n^{\rm rob})$	$R(\widehat{\theta}_n^{\rm erm})$	$R(\widehat{\theta}_n^{\rm rob})$	$R(\widehat{\theta}_n^{\rm erm})$	$R(\widehat{\theta}_n^{\rm rob})$	$R(\widehat{\theta}_n^{\rm erm})$	$R(\widehat{\theta}_n^{\rm rob})$
100	4.06E-06	7.42E-07	4.17E-04	7.65E-05	4.20E-02	7.64E-03	4.15E+00	7.12E-01
500	8.91E-07	5.01E-15	8.22E-05	1.63E-14	8.36E-03	2.19E-13	8.41E-01	8.21E-12
1000	4.47E-07	1.52E-15	4.02E-05	1.64E-17	4.20E-03	6.32E-18	4.19E-01	2.45E-17
5000	1.44E-07	2.68E-16	8.00E-06	2.74E-18	8.27E-04	5.09E-20	8.38E-02	6.55E-20
10000	7.64E-08	1.32E-16	4.02E-06	1.32E-18	4.13E-04	2.57E-20	4.18E-02	3.34E-20

5.3. Protease cleavage experiments

For our second experiment, we compare our robust regularization procedure to other regularizers using the HIV-1 protease cleavage dataset from the UCI ML-repository (Lichman, 2013). In this binary classification task, one is given a string of amino acids (a protein) and a featurized representation of the string of dimension $d = 50960$, and the goal is to predict whether the HIV-1 virus will cleave the amino acid sequence in its central position. We have a sample of $n = 6590$ observations of this process, where the class labels are somewhat skewed: there are 1360 examples with label $Y = +1$ (HIV-1 cleaves) and 5230 examples with $Y = -1$ (does not cleave).

We use the logistic loss $\ell(\theta; (x, y)) = \log(1+\exp(-y\theta^\top x))$. We compare the performance of different constraint sets Θ by taking

$$\Theta = \left\{\theta \in \mathbb{R}^d : a_1 \left\|\theta\right\|_1 + a_2 \left\|\theta\right\|_2 \le r\right\},$$

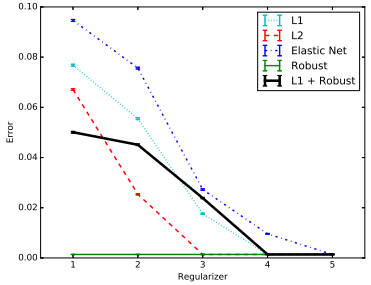
Table 2: Simulation experiment: Variances of $R(\widehat{\theta})$ over 1,200 simulations

	$B = .01$		$B = .1$		$B = 1$		$B = 10$	
n	ERM	Robust	ERM	Robust	ERM	Robust	ERM	Robust
100	7.06E-13	9.76E-14	6.58E-09	1.03E-09	7.09E-05	1.08E-05	7.37E-01	9.20E-02
500	5.98E-14	7.15E-28	3.04E-10	3.52E-26	2.80E-06	2.26E-24	2.92E-02	3.26E-21
1000	2.63E-14	1.07E-31	7.53E-11	1.99E-35	7.14E-07	3.44E-33	7.03E-03	4.78E-32
5000	7.34E-15	2.94E-33	2.70E-12	3.28E-37	2.95E-08	2.50E-39	2.74E-04	5.24E-38
10000	1.60E-15	6.54E-34	6.74E-13	7.59E-38	7.04E-09	3.34E-39	6.52E-05	2.25E-38

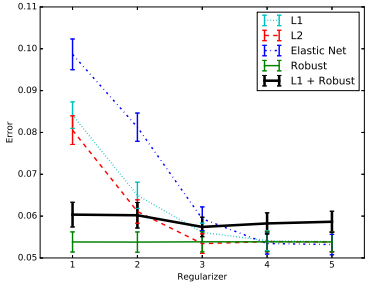
which is equivalent to elastic net regularization (Zou and Hastie, 2005), while varying a_1, a_2, and r. We experiment with ℓ_1-constraints ($a_1 = 1, a_2 = 0$) with $r \in \{50, 100, 500, 1000, 5000\}$, ℓ_2-constraints ($a_1 = 0, a_2 = 1$) with $r \in \{5, 10, 50, 100, 500\}$, elastic net ($a_1 = 1, a_2 = 10$) with $r \in \{100, 200, 1000, 2000, 10000\}$, our robust regularizer with $\rho \in \{100, 1000, 10000, 50000, 100000\}$ and our robust regularizer coupled with the ℓ_1-constraint ($a_1 = 1, a_2 = 0$) with $r = 100$. Though we use a convex surrogate (logistic loss), we measure performance of the classifiers using the 0-1 (misclassification) loss $1\{\text{sign}(\theta^T x)y \le 0\}$. For validation, we perform 50 experiments, where in each experiment we randomly select 9/10 of the data to train the model, evaluating its performance on the held out fraction (test).

We plot results summarizing these experiments in Figure 3. The horizontal axis in each figure indexes our choice of regularization value (so "`Regularizer = 1`" for the ℓ_1-constrained problem corresponds to $r = 50$). The figures show that the robustly regularized risk provides a different type of protection against overfitting than standard regularization or constraint techniques do: while other regularizers underperform in heavily constrained settings, the robustly regularized estimator $\widehat{\theta}_n^{\rm rob}$ achieves low classification error for all values of ρ (Figure 3(b)). Notably, even when coupled with a fairly stringent ℓ_1-constraint ($r = 100$), robust regularization has perofrmance better than ℓ_1 except for large values r, especially on the rare label $Y = +1$ (Figure 3 (d) and (f)).

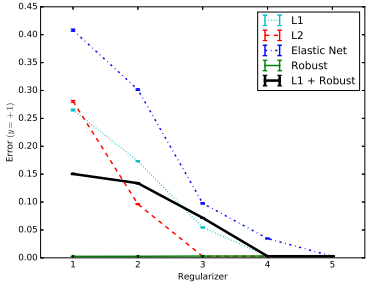
We investigate the effects of the robust regularizer with a slightly different perspective in Figure 4, where we use $\Theta = \{\theta : \|\theta\|_1 \le r\}$ with $r = 100$ for the constraint set for each experiment. The horizontal axis indicates the tolerance ρ we use in construction of the robust estimator $\widehat{\theta}_n^{\rm rob}$, where `ERM` means $\rho = 0$. In Fig. 4(a), we plot the logistic risk $R(\widehat{\theta}) = \mathbb{E}[\ell(\widehat{\theta}, (X, Y))]$ for the train and test distribution. We also plot the upper confidence bound $R_n(\theta, \mathcal{P}_n)$ in this plot, which certainly over-estimates the test risk—we hope to tighten this overestimate in future work. In Figure 4(b), we plot the misclassification error on train and test for different values of ρ, along with 2-standard-error intervals for the 50 runs. Figures 4(c) and (d) show the error rates restricted to examples from the uncommon (c) and common (d) classes. In Table 3 we give explicit error rates and logistic risk values for the different procedures. Due to the small size of the test dataset ($n_{\rm test} = 659$), the deviation across folds is somewhat large.



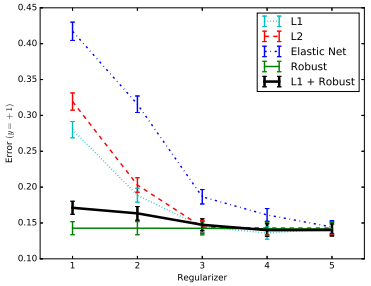
(a) Train error



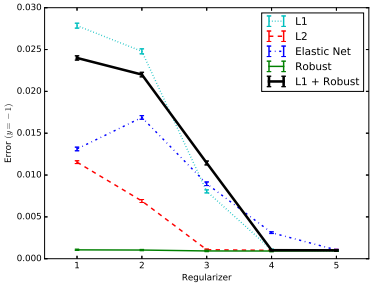
(b) Test error



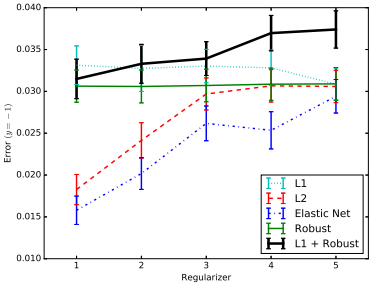
(c) Train error on rare class ($Y_i = +1$)



(d) Test error on rare class ($Y_i = +1$)

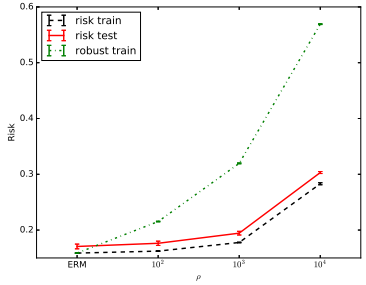


(e) Train error on common class ($Y_i = -1$)

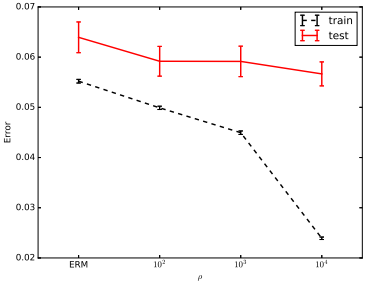


(f) Test error on common class ($Y_i = -1$)

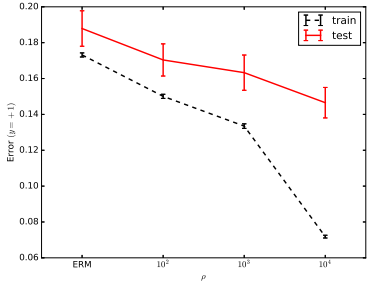
Figure 3. HIV-1 Protease Cleavage plots (2-standard error confidence bars). Comparison of misclassification error rates among different regularizers.



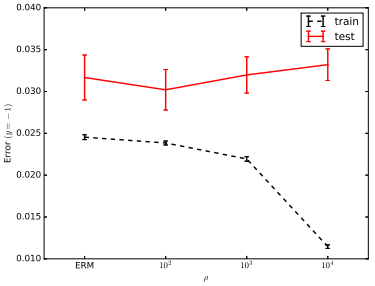
(a) Logistic risk and confidence bound



(b) Misclassification error rate



(c) Error on rare class ($Y_i = +1$)



(d) Error on common class ($Y_i = -1$)

Figure 4. HIV-1 Protease Cleavage plots (2-standard error confidence bars). Plot (a) shows the logistic risk $R(\theta) = \mathbb{E}[\log(1+e^{-Y\theta^\top X})]$ and confidence bounds computed from the robust risk (4). Plots (b)–(d) show misclassification error rates plotted against robustness parameter ρ.

In this experiment, we see (roughly) that the ERM solutions achieve good performance on the common class ($Y = -1$) but sacrifice performance on the uncommon class. As we increase ρ, performance of the robust solution $\widehat{\theta}_n^{\rm rob}$ on the rarer label $Y = +1$ improves (Fig. 4(c)), while the misclassification rate on the common class degrades a small (insignificant) amount (Fig. 4(d)); see also Table 3. This behavior is roughly what we might expect for the robust estimator: the poor performance of the ERM estimator $\widehat{\theta}_n^{\rm erm}$ on the rare class induces (relatively) more variance, which the robust solution reduces by via improved classification performance on the rare ($Y = +1$) class. This occurs at little expense over the more common label $Y = -1$ so that overall performance improves by a small amount. We remark—but are unable to explain—that this improvement on classification error for the rare labels comes despite increases in logistic risk; while the average logistic loss increases, misclassification errors decrease.

Table 3: HIV-1 Cleavage Error

	risk		error (%)		error ($Y=+1$)		error ($Y=-1$)	
ρ	train	test	train	test	train	test	train	test
erm	0.1587	0.1706	5.52	6.39	17.32	18.79	2.45	3.17
100	0.1623	0.1763	4.99	5.92	15.01	17.04	2.38	3.02
1000	0.1777	0.1944	4.5	5.92	13.35	16.33	2.19	3.2
10000	0.283	0.3031	2.39	5.67	7.18	14.65	1.15	3.32

5.4. Document classification in the Reuters corpus

For our final experiment, we consider a multi-label classification problem with a reasonably large dataset. The Reuters RCV1 Corpus (Lewis et al., 2004) has 804,414 examples with $d = 47{,}236$ features, where feature j is an indicator variable for whether word j appears in a given document. The goal is to classify documents as a subset of the 4 categories Corporate, Economics, Goverment, and Markets, and each document in the data is labeled with a subset of those. As each document can belong to multiple categories, we fit binary classifiers on each of the four categories. There are different numbers of documents labeled as each category, with the Economics category having the fewest number of positive examples. Table 4 gives the number of times a document is labeled as each of the four categories (so each document has about 1.18 associated classes). In this experiment, we expect the robust solution to outperform ERM on the rarer category (Economics), as the robustification (6) naturally upweights rarer (harder) instances, which disproportionally affect variance—as in the experiment on HIV-1 cleavage.

Table 4: Reuters Number of Examples

Corporate	Economics	Government	Markets
381,327	119,920	239,267	204,820

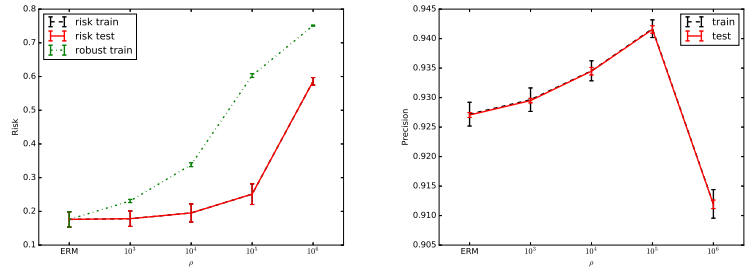
For each category $k \in \{1, 2, 3, 4\}$, we use the logistic loss $\ell(\theta_k; (x, y)) = \log(1+\exp(-y\theta_k^\top x))$. For each binary classifier, we use the ℓ_1 constraint set $\Theta = \left\{\theta \in \mathbb{R}^d : \|\theta\|_1 \le 1000\right\}$. To evaluate performance on this multi-label problem, we use precision (ratio of the number of correct positive labels to the number classified as positive) and recall (ratio of the number of correct positive labels to the number of actual positive labels):

$$\text{precision} = \frac{1}{n}\sum_{i=1}^n \frac{\sum_{k=1}^4 1\{\theta_k^\top x_i \ge 0, y_i = 1\}}{\sum_{k=1}^4 1\{\theta_k^\top x_i > 0\}},$$

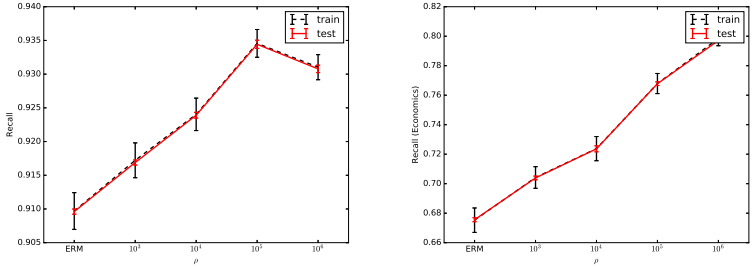
$$\text{recall} = \frac{1}{n}\sum_{i=1}^n \frac{\sum_{k=1}^4 1\{\theta_k^\top x_i \ge 0, y_i = 1\}}{\sum_{k=1}^4 1\{y_i = 1\}}.$$

We partition the data into ten equally-sized sub-samples and perform ten validation experiments, where in each experiment we use one of the ten subsets for fitting the logistic models and the remaining nine partitions as a test set to evaluate performance.

In Figure 5, we summarize the results of our experiment averaged over the 10 runs, with 2-standard error bars (computed across the folds). To facilitate comparison across the document categories, we give exact values of these averages in Tables 5 and 6. Both



(a) Logistic risk and confidence bound (b) Precision



(c) Recall (d) Recall on rare category (Economics)

Figure 5: Reuters Corpus (2-standard error deviations)

$\widehat{\theta}_n^{\text{rob}}$ and $\widehat{\theta}_n^{\text{erm}}$ have reasonably high precision across all categories, with increasing ρ giving a mild improvement in precision (from $.93 \pm .005$ to $.94 \pm .005$); see also Figure 5(a). On the other hand, we observe in Figure 5(d) that ERM has low recall (.69 on test) for the Economics category, which contains about 15% of documents. As we increase ρ from 0 (ERM) to 10^5, we see a smooth and substantial improvement in recall for this rarer category (without significant degradation in precision). This improvement in recall amounts to reducing variance in predictions on the rare class. We also note that while the robust solutions outperform ERM in classification performance for $\rho \le 10^5$, for very large $\rho = 10^6 \ge 10n$, the regularizing effects of robustness degrade the solution $\widehat{\theta}_n^{\text{rob}}$. This precision and recall improvement comes in spite of the increase in the average binary logistic risk for each of the 4 classes, which we show in Figure 5a, which plots the average binary logistic loss (on train and test sets) averaged over the 4 categories as well as the upper confidence bound $R_n(\theta, \mathcal{P}_n)$ as we vary ρ. The robust regularization effects reducing variance appear to improve the performance of the binary logistic loss as a surrogate for true misclassification error.

Table 5: Reuters Corpus Precision (%)

	Precision		Corporate		Economics		Government		Markets	
ρ	train	test	train	test	train	test	train	test	train	test
erm	92.72	92.7	93.55	93.55	89.02	89	94.1	94.12	92.88	92.94
1E3	92.97	92.95	93.31	93.33	87.84	87.81	93.73	93.76	92.56	92.62
1E4	93.45	93.45	93.58	93.61	87.6	87.58	93.77	93.8	92.71	92.75
1E5	94.17	94.16	94.18	94.19	86.55	86.56	94.07	94.09	93.16	93.24
1E6	91.2	91.19	92	92.02	74.81	74.8	91.19	91.25	89.98	90.18

Table 6: Reuters Corpus Recall (%)

	Recall		Corporate		Economics		Government		Markets	
ρ	train	test	train	test	train	test	train	test	train	test
erm	90.97	90.96	90.20	90.25	67.53	67.56	90.49	90.49	88.77	88.78
1E3	91.72	91.69	90.83	90.86	70.42	70.39	91.26	91.23	89.62	89.58
1E4	92.40	92.39	91.47	91.54	72.38	72.36	91.76	91.76	90.48	90.45
1E5	93.46	93.44	92.65	92.71	76.79	76.78	92.26	92.21	91.46	91.47
1E6	93.10	93.08	92.00	92.04	79.84	79.71	91.89	91.90	92.00	91.97

Summary

We have seen through multiple examples that robustification—our convex surrogate for variance regularization—is an effective tool in a number of applications. As we heuristically expect, variance-based regularization (robust regularization) yields predictors with better performance on "hard" instances, or subsets of the problem that induce higher variance, such as classes with relatively few training examples in classification problems. The robust regularization ρ gives a principled knob for tuning performance to trade between variance (uniform or across-the-board performance) and—sometimes—absolute performance.

6. Discussion

In this paper, we have developed theoretical results for robust regularization (6) that apply to general stochastic optimization and learning problems problems. The examples we describe in Section 3 illustrate our expectation that the robust solution $\widehat{\theta}_n^{\rm rob}$ should have good performance in cases in which $\mathrm{Var}(\ell(\theta^\star; X))$ is small (recall also Theorems 3 and 6). Identifying the separation between the performance empirical risk minimization and related estimators and that of the robustly-regularized estimators—as well as variance-regularized estimates—we consider more generally remains a challenge. We hope that this paper inspires work in this direction in machine learning and statistics, and more broadly, torward considering distributionally robust problems. Part of this is likely to come from making rigorous our empirical observations (Section 5) that robust regularization improves performance on "hard" instances without sacrificing performance on easier cases.

Our understanding of so-called "fast rates" for stochastic optimization problems, while considering robustness, is also limited. For empirical risk minimization, fast rates of convergence hold under conditions in which the the gap $R(\theta) - R(\theta^\star)$ controls the variance of the excess loss $\ell(\theta, X) - \ell(\theta^\star, X)$ (cf. Mammen and Tsybakov, 1999; Bartlett et al., 2005;

Boucheron et al., 2005; Bartlett et al., 2006), which usually requires some type of uniform convexity assumption. These bounds typically follow from localization guarantees (Bartlett et al., 2005, Section 5) on the function class

$$\{x \mapsto \ell(\theta, x) - \ell(\theta^\star, x) \mid \theta \in \Theta\}.$$

While in Section 4.1, we show that the robust estimate $\widehat{\theta}_n^{\rm rob}$ enjoys faster rates of convergence under growth conditions analogous to uniform convexity of the risk, as $\mathrm{Var}(\ell(\theta; X) - \ell(\theta^\star; X)) \neq \mathrm{Var}(\ell(\theta; X))$, it is not clear how to directly connect these guarantees to results of the form in Theorems 3 and 6. We leave investigation of these topics to future work.

The last point of our discussion is to revisit Theorem 6, which provides a guarantee for robustly regularized estimators based on localized Rademacher complexities. An investigation of our proof shows that our derivation proceeds by considering the complexity of self-normalized classes of functions of the form

$$\mathcal{G}_r = \left\{ \sqrt{\frac{r}{\mathbb{E}[f^2] \vee r}} f \mid f \in \mathcal{F} \right\}.$$

In contrast, the analogous result of Bartlett et al. (2005, Thereom 3.3) for empirical risk minimization considers the complexity of classes of functions of the form

$$\mathcal{G}_r = \left\{ \frac{r}{\mathbb{E}[f^2] \vee r} f \mid f \in \mathcal{F} \right\}.$$

The latter class normalizes functions f by $\sqrt{\mathbb{E}[f^2]}$—a type of self-normalization that arises in the computation of pivotal (asymptotically independent of the underlying distribution) statistics. While this choice *prima facie* is just a step in our proof, the robust objective $R_n(\theta, \mathcal{P}_n)$ defined in Eq. (4) is an empirical likelihood upper confidence bound on the optimal population risk (see also Duchi et al., 2016). One of the important characteristics of empirical likelihood confidence bounds is that they are self-normalizing and yield pivotal statistics (Owen, 2001). Investigating such self-normalization in complexity guarantees seems likely to yield fruitful insights.

Acknowledgments

We thank Feng Ruan for pointing out a much simpler proof of Theorem 1 than in our original paper. JCD and HN were partially supported by the SAIL-Toyota Center for AI Research and HN was partially supported by the Samsung Fellowship. JCD was also partially supported by the National Science Foundation award NSF-CAREER-1553086 and the Sloan Fellowship.

Appendix A. Proof of Theorem 1

The theorem is immediate if $s_n = 0$ or $\sigma^2 = 0$, as in this case $\sup_{P:D_\phi(P\|\widehat{P}_n)\le\rho/n} \mathbb{E}_P[Z] = \mathbb{E}_{\widehat{P}_n}[Z] = \mathbb{E}[Z]$. In what follows, we will thus assume that $\sigma^2, s_n^2 > 0$. We recall the maximization problem (8), which is

$$\operatorname*{maximize}_p \sum_{i=1}^n p_i z_i \quad \text{subject to } p \in \mathcal{P}_n = \left\{ p \in \mathbb{R}_+^n : \frac{1}{2}\left\|np - \mathbf{1}\right\|_2^2 \le \rho, \langle \mathbf{1}, p\rangle = 1 \right\},$$

and the solution criterion (9), which guarantees that the maximizing value of problem (8) is $\overline{z} + \sqrt{2\rho s_n^2/n}$ whenever

$$\sqrt{2\rho}\frac{z_i - \overline{z}}{\sqrt{n s_n^2}} \ge -1.$$

Letting $z = Z$, then under the conditions of the theorem, we have $|z_i - \overline{z}| \le M$, and to satisfy inequality (9) it is certainly sufficient that

$$2\rho\frac{M^2}{n s_n^2} \le 1, \quad \text{or} \quad n \ge \frac{2\rho M^2}{s_n^2}, \quad \text{or} \quad s_n^2 \ge \frac{2\rho M^2}{n}. \tag{30}$$

Conversely, suppose that $s_n^2 < \frac{2\rho M^2}{n}$. Then we have $\frac{2\rho s_n^2}{n} < \frac{4\rho^2 M^2}{n^2}$, which in turn implies that

$$\sup_{p\in\mathcal{P}_n} \langle p, z\rangle \ge \frac{1}{n}\langle \mathbf{1}, z\rangle + \left(\sqrt{\frac{2\rho s_n^2}{n}} - \frac{2M\rho}{n}\right)_+.$$

Combining this inequality with the condition (30) for the exact expansion to hold yields the two-sided variance bounds (10).

We now turn to showing the high-probability exact expansion (11), which occurs whenever the sample variance is large enough by expression (30). To that end, we show that s_n^2 is bounded from below with high probability. Define the event

$$\mathcal{E}_n := \left\{ s_n^2 \ge \frac{3}{64}\sigma^2 \right\},$$

and let $n \ge \frac{4M^2}{\sigma^2}\max\{2\sigma, 11\}$. Then, on event $\mathcal{E}_n$ we have $n \ge \frac{44\rho M^2}{\sigma^2} \ge \frac{2\rho M^2}{s_n^2}$, so that the sufficient condition (30) holds and expression (11) follows. We now argue that the event $\mathcal{E}_n$ has high probability via the following lemma due to Maurer and Pontil.

Lemma 11 (Maurer and Pontil (2009, Theorem 10)) *Let Z_i be i.i.d. random variables taking values in $[0, M]$, and let $s_n^2 = \frac{1}{n}\sum_{i=1}^n Z_i^2 - \left(\frac{1}{n}\sum_{i=1}^n Z_i\right)^2$. Then, for $n \ge 2$*

$$\mathbb{P}\left(s_n \le \sigma - t\right) \vee \mathbb{P}\left(s_n \ge \sigma + t\right) \le \exp\left(-\frac{nt^2}{2M^2}\right).$$

Setting $t = \left(1 - \frac{\sqrt{3}}{8}\right)\sigma$, the final result follows from noting that $\mathbb{P}(\mathcal{E}_n) \ge 1 - \exp\left(-\frac{nt^2}{2M^2}\right)$ from the lemma.

Appendix B. Proof of Theorem 2

Our starting point is to recall from inequality (30) in the proof of Theorem 1 that for each $f \in \mathcal{F}$, the empirical variance equality (12) holds if $n \ge \frac{4\rho M^2}{\mathrm{Var}_{\widehat{P}_n}(f)}$. As a consequence, Theorem 2 will follow if we can provide a uniform lower bound on the sample variances $\mathrm{Var}_{\widehat{P}_n}(f)$ that holds with high enough probability. We use C to denote a universal constant whose value may change from line to line. Noting that $\mathrm{Var}_{\widehat{P}_n}(f) = \mathbb{E}_{\widehat{P}_n}(f - \mathbb{E}[f])^2 - (\mathbb{E}_{\widehat{P}_n}(f - \mathbb{E}[f]))^2$, we proceed in two parts. First, we give a lower bound for $\mathbb{E}_{\widehat{P}_n}(f - \mathbb{E}[f])^2$.

Lemma 12 *Let $\mathcal{F}$ be a collection of bounded functions $f : \mathcal{X} \to [M_0, M_1]$ with $M := M_1 - M_0$. Then, with probability at least $1 - e^{-t}$, for every $f \in \mathcal{F}$*

$$\mathrm{Var}(f) \le 2\mathbb{E}_{\widehat{P}_n}(f - \mathbb{E}[f])^2 + C\left[\mathfrak{R}_n^{\sup}(\mathcal{F})^2 \log^3(nM) + \frac{M^2}{n}\left(t + \log\log n\right)\right].$$

Proof We follow the arguments of Srebro et al. (2010) and Bousquet (2002b, Thm. 6.1). For $x_1, \ldots, x_n \in \mathcal{X}$, let

$$\mathcal{F}_{n,r} := \left\{ f - \mathbb{E}[f] \mid f \in \mathcal{F}, \quad \mathbb{E}_{\widehat{P}_n}[(f - \mathbb{E}[f])^2] \le r \right\},$$

where $\widehat{P}_n$ is the empirical measure on $x_1, \ldots, x_n$. Let $\psi_n^{\sup}$ be a sub-root upper bound on the worst-case Rademacher complexity

$$\psi_n^{\sup}(r) \ge \mathfrak{R}_n^{\sup}(\mathcal{F}_{n,r}),$$

where implicitly in the right hand side we take the supremum over $x_1, \ldots, x_n$ definining $\mathcal{F}_{n,r}$ as well.

Lemma 13 (Srebro et al. (2010, Lemma 2.2)) *Let $\mathcal{H}$ be a class of bounded functions $\mathcal{X} \to [-M, M]$, and let $\kappa : [-M, M] \to \mathbb{R}_+$ be a bounded function with L-Lipschitz derivatives. Then,*

$$\mathfrak{R}_n^{\sup}\left(\left\{\kappa \circ h : h \in \mathcal{H}, \mathbb{E}_{\widehat{P}_n}[\kappa \circ h] \le r\right\}\right) \le C\sqrt{r}\mathfrak{R}_n^{\sup}(\mathcal{H})\log^{\frac{3}{2}}(Mn).$$

Since $\kappa(t) = t^2$ has Lipschitz derivatives, above lemma with $\mathcal{H} = \{f - \mathbb{E}[f] : f \in \mathcal{F}\}$ yields

$$\mathfrak{R}_n^{\sup}(\mathcal{F}_{n,r}^2) \le C\sqrt{r}\mathfrak{R}_n^{\sup}(\mathcal{F})\log^{\frac{3}{2}}(nM) \tag{31}$$

where we recall the notation that $\mathcal{G}^2 = \{g^2 \mid g \in \mathcal{G}\}$ for any function class $\mathcal{G}$. Thus we may take $\psi_n^{\sup}(r) = C\sqrt{r}\mathfrak{R}_n^{\sup}(\mathcal{F})\log^{\frac{3}{2}} n$, which has fixed point $r_n^{\sup} = C^2\mathfrak{R}_n^{\sup}(\mathcal{F})^2\log^3 n$. The following classical result then shows that the fixed point $r_n^{\sup}$ controls generalization of class of functions $\mathcal{F}_{n,r}^2$. Since $f^2 \ge 0$, Theorem 6.1 of Bousquet (2002b) yields that for all $f \in \mathcal{F}$,

$$\mathbb{E}(f - \mathbb{E}[f])^2 \le 2\mathbb{E}_{\widehat{P}_n}(f - \mathbb{E}[f])^2 + C\left[\mathfrak{R}_n^{\sup}(\mathcal{F})^2\log^3 nM + \frac{M^2}{n}\left(t + \log\log n\right)\right]$$

with probability at least $1 - e^{-t}$. ∎

Next, we give an upper bound for $(\mathbb{E}_{\widehat{P}_n}(f - \mathbb{E}[f]))^2$. We use the following version of Talagrand's inequality due to Bousquet (2002a, 2003). (See also Bartlett et al. (2005, Thm 2.1).)

Lemma 14 *Let $r > 0$ and $\mathcal{F}$ be a class of functions that map $\mathcal{X}$ into $[a, b]$ such that for every $f \in \mathcal{F}$, $\mathrm{Var}(f(X)) \le r$. Then, with probability at least $1 - e^{-t}$*

$$\sup_{f\in\mathcal{F}}\{\mathbb{E}[f] - \mathbb{E}_{\widehat{P}_n}[f]\} \le \inf_{\alpha>0}\left\{2(1+\alpha)\mathbb{E}[\mathfrak{R}_n(\mathcal{F})] + \sqrt{\frac{2rt}{n}} + \frac{t}{n}(b-a)\left(\frac{1}{3}+\frac{1}{\alpha}\right)\right\}.$$

The same statement holds with $\sup_{f\in\mathcal{F}}(\mathbb{E}_{\widehat{P}_n}[f] - \mathbb{E}[f])$ replacing the left-hand side of the inequalities.

Applying Lemma 14 and letting $\alpha = \frac{1}{2}$, with probability at least $1 - 2e^{-t}$

$$|\mathbb{E}_{\widehat{P}_n}[f] - \mathbb{E}[f]| \le 3\mathbb{E}[\mathfrak{R}_n(\mathcal{F})] + 2M\sqrt{\frac{2t}{n}}$$

holds for all $f \in \mathcal{F}$. Combining the above display with Lemma 12, we obtain the desired result.

Appendix C. Proof of Theorem 3

Before proving the theorem proper, we state a technical lemma that provides uniform Bernstein-like bounds for the class $\mathcal{F}$ using empirical ℓ_∞-covering numbers.

Lemma 15 (Maurer and Pontil (2009, Theorem 6)) *Let $n \ge \frac{8M^2}{t}$ and $t \ge \log 12$. Then with probability at least $1 - 6N_\infty(\mathcal{F}, \epsilon, 2n)e^{-t}$, we have*

$$\mathbb{E}[f] \le \mathbb{E}_{\widehat{P}_n}[f] + 3\sqrt{\frac{2\mathrm{Var}_{\widehat{P}_n}(f)t}{n}} + \frac{15Mt}{n} + 2\left(1 + 2\sqrt{\frac{2t}{n}}\right)\epsilon \tag{32}$$

for all $f \in \mathcal{F}$.

We return to the proof of Theorem 3. Let $\mathcal{E}_1$ denote that the event that the inequalities (32) hold. Then on $\mathcal{E}_1$ hold, uniformly over $f \in \mathcal{F}$ we have

$$\begin{aligned}
\mathbb{E}[f] &\le \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{\frac{18\mathrm{Var}_{\widehat{P}_n}(f(X))t}{n}} + \frac{15Mt}{n} + 2\left(1 + 2\sqrt{\frac{2t}{n}}\right)\epsilon \\
&\overset{(i)}{\le} \sup_{P:D_\phi(P\|\widehat{P}_n)\le\frac{\rho}{n}} \mathbb{E}_P[f(X)] + \sqrt{\frac{2\rho\mathrm{Var}_{\widehat{P}_n}(f(X))}{n}} \\
&\quad - \left(\sqrt{\frac{2\rho\mathrm{Var}_{\widehat{P}_n}(f(X))}{n}} - \frac{2M\rho}{n}\right)_+ + \frac{5M\rho}{3n} + 2\left(1 + 2\sqrt{\frac{2t}{n}}\right)\epsilon \\
&\le \sup_{P:D_\phi(P\|\widehat{P}_n)\le\frac{\rho}{n}} \mathbb{E}_P[f(X)] + \frac{11}{3}\frac{M\rho}{n} + 2\left(1 + 2\sqrt{\frac{2t}{n}}\right)\epsilon \text{ for all } f \in \mathcal{F},
\end{aligned} \tag{33}$$

where inequality (i) follows from the bounds (10) in Theorem 1 and the fact that $\rho \ge 9t$ by assumption. This gives the first result (15).

For the second result (16), we recall that $\widehat{f} \in \mathrm{argmin}_{f\in\mathcal{F}} \sup_P\{\mathbb{E}_P[f(X)] : D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}\}$, and we bound the supremum term in expression (33). First, we note that because $\widehat{f}$ minimizes the supremum term in expression (33), we have

$$\mathbb{E}[\widehat{f}] \le \sup_{P:D_\phi(P|\widehat{P}_n)\le\frac{\rho}{n}} \mathbb{E}_P[f(X)] + \frac{11M\rho}{3n} + 2\left(1 + 2\sqrt{\frac{2t}{n}}\right)\epsilon \text{ for all } f \in \mathcal{F}.$$

Now fix $f \in \mathcal{F}$. As the function f is fixed, by Bernstein's inequality, we have

$$\mathbb{E}_{\widehat{P}_n}[f] \le \mathbb{E}[f] + \sqrt{\frac{2\mathrm{Var}(f)t}{n}} + \frac{2Mt}{3n}$$

with probability at least $1 - e^{-t}$. Similarly, we have by Lemma 11 that

$$\sqrt{\mathrm{Var}_{\widehat{P}_n}(f)} \le \sqrt{\mathrm{Var}(f)} + \sqrt{\frac{2tM^2}{n}}$$

with probability at least $1 - e^{-t}$. That is, for any fixed $f \in \mathcal{F}$, we have with probability at least $1 - 2e^{-t}$ that

$$\begin{aligned}
\sup_{P:D_\phi(P|\widehat{P}_n)\le\frac{\rho}{n}} \mathbb{E}_P[f(X)] &\overset{(i)}{\le} \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{\frac{2\rho\mathrm{Var}_{\widehat{P}_n}(f)}{n}} \\
&\le \mathbb{E}[f] + \sqrt{\frac{2\mathrm{Var}(f)t}{n}} + \frac{2M}{3n}t + \sqrt{\frac{2\rho\mathrm{Var}(f)}{n}} + \frac{2\sqrt{M^2\rho t}}{n} \\
&\overset{(ii)}{\le} \mathbb{E}[f] + 2\sqrt{\frac{2\mathrm{Var}(f)\rho}{n}} + \frac{8}{3}\frac{M\rho}{n},
\end{aligned}$$

where inequality (i) follows from the uniform upper bound (10) of Theorem 1 and inequality (ii) from our assumption that $\rho \ge t$. Substituting this expression into our earlier bound (33) yields that for any $f \in \mathcal{F}$, with probability at least

$$1 - 2(3N_\infty(\mathcal{F}, \epsilon, 2n) + 1)e^{-t},$$

we have

$$\mathbb{E}[\widehat{f}(X)] \le \mathbb{E}[f(X)] + 2\sqrt{\frac{2\rho\mathrm{Var}(f(X))}{n}} + \frac{19}{3}\frac{M\rho}{n} + 2\left(1 + 2\sqrt{\frac{2t}{n}}\right)\epsilon.$$

This gives the theorem.

Appendix D. Proof of Theorem 6

We first show the following version of uniform Bernstein's inequality with Rademacher complexities. The proof uses a peeling technique (Bartlett et al., 2005; van de Geer, 2000), in conjuction with Talagrand's concentration inequality (Lemma 14).

Lemma 16 *Let $r > 0$ and $\mathcal{F}$ be a collection of bounded functions $f : \mathcal{X} \to [0, M]$ with $\mathrm{Var}(f(X)) \le r$. Then, with probability at least $1 - e^{-t}$, for every $f \in \mathcal{F}$*

$$\mathbb{E}[f] \le \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{\frac{2e\mathrm{Var}(f)}{n}\left(t + \log\left\lceil \log \frac{nr}{M^2 t}\right\rceil\right)} + 6\mathbb{E}[\mathfrak{R}_n(\mathcal{F})] + \frac{7M}{n}\left(t + \log\left\lceil \log \frac{nr}{M^2 t}\right\rceil\right).$$

The same statements hold with the roles of $\mathbb{E}[f]$ and $\mathbb{E}_{\widehat{P}_n}[f]$ reversed.

We defer the proof to section D at the end of this section. Because $\mathrm{Var}(f) \le M^2$ for all $f \in \mathcal{F}$, Lemma 16 also holds if we replace the terms $\left\lceil \log \frac{nr}{M^2 t}\right\rceil$ with $\left\lceil \log \frac{n}{t}\right\rceil \le 1 + \log \frac{n}{t}$.

Next, we show an important extension of Lemma 16 that replaces the Rademacher complexity term $\mathbb{E}[\mathfrak{R}_n(\mathcal{F})]$ by a local quantity $r_n^\star$, the fixed point of $\psi_n(r)$. To this end, we use another peeling argument and apply Lemma 16 to the self-normalized class

$$\mathcal{G}_r := \left\{ \sqrt{\frac{r}{\mathbb{E}[f^2] \vee r}} f : f \in \mathcal{F} \right\} \subseteq \left\{ cf : f \in \mathcal{F}, \mathbb{E}[c^2 f^2] \le r, c \in [0,1] \right\}.$$

This idea follows the techniques of Bartlett et al. (2005, Thm. 3.3), though we use a type of self-normalizing scale, that is, $f/\sqrt{\mathbb{E}[f^2]}$, whereas they use a variance-normalizing scaling by studying classes of functions of the form $f/\mathbb{E}[f^2]$. Our use of this alternative normalization is important in the next lemma, which allows us to obtain bounds that apply to the robustly regularized risk.

Lemma 17 *Let $\mathcal{F}$ be a collection of bounded functions $f : \mathcal{X} \to [0, M]$ satisfying the localization inequality (20) for some sub-root function $\psi_n(\cdot)$ with root $r_n^\star$. Let $B_n = \frac{1}{n}\left(t + \log\left\lceil \log \frac{n}{t}\right\rceil\right)$. Then with probability at least $1 - e^{-t}$, for every $f \in \mathcal{F}$*

$$\mathbb{E}[f] \le \mathbb{E}_{\widehat{P}_n}[f] + \left(\sqrt{2eB_n} + 6\sqrt{r_n^\star + 7MB_n/3}\right)\sqrt{\mathbb{E}[f^2]} + 6r_n^\star + 14MB_n.$$

The same statement holds with the roles of $\mathbb{E}[f]$ and $\mathbb{E}_{\widehat{P}_n}[f]$ reversed.

See Section D.1 for the proof.

Next, we give an analogous result for f^2.

Lemma 18 *Let $\mathcal{F}$ be a collection of bounded functions $f : \mathcal{X} \to [0, M]$ satisfying the localization inequality (20) for some sub-root function $\psi_n(\cdot)$ with root $r_n^\star$. Let $\eta > 0$. Then, with probability at least $1 - e^{-t}$, for every $f \in \mathcal{F}$*

$$\mathbb{E}[f^2] \le \mathbb{E}_{\widehat{P}_n}[f^2] + \frac{1}{\eta}\mathbb{E}_{\widehat{P}_n}[f^2] + 72M^2(1+\eta)r_n^\star + \frac{Mt}{n}\left(4 + \frac{7}{3}M\right).$$

Also, with probability at least $1 - e^{-t}$, for every $f \in \mathcal{F}$

$$\mathbb{E}_{\widehat{P}_n}[f^2] \le \mathbb{E}[f^2] + \frac{\eta}{1+\eta}\mathbb{E}[f^2] + 72M^2(1+\eta)r_n^\star + \frac{Mt}{n}\left(4 + \frac{7}{3}M\right).$$

See Section D.2 for the proof.

Now, we make two additional pieces of shorthand notation. Let

$$V_n = 4((2e + 84M)B_n + 36r_n^\star).$$

Then, Lemma 17 implies that

$$\mathbb{E}[f] \le \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{V_n\mathbb{E}[f^2]} + 6r_n^\star + 14MB_n$$

with probability at least $1 - e^{-t}$. Applying Lemma 18 to this bound with the choice $\eta = 1$ immediately yields that

$$\begin{aligned}\mathbb{E}[f] &\le \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{2V_n\mathbb{E}_{\widehat{P}_n}[f^2] + 144M^2V_nr_n^\star + 7V_nM\max\{M,1\}t/n} + 6r_n^\star + 14MB_n \\ &\le \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{2V_n\mathbb{E}_{\widehat{P}_n}[f^2]} + 12M\sqrt{V_n\left(r_n^\star + \frac{7\max\{M,1\}}{M}\frac{t}{n}\right)} + 6r_n^\star + 14MB_n\end{aligned}$$

for all $f \in \mathcal{F}$ with probability at least $1 - 2e^{-t}$. Subtracting and adding $(\mathbb{E}_{\widehat{P}_n}[f])^2$ to the second term, we have

$$\sqrt{2V_n\mathbb{E}_{\widehat{P}_n}[f^2]} = \sqrt{2V_n\mathrm{Var}_{\widehat{P}_n}(f) + 2V_n\mathbb{E}_{\widehat{P}_n}[f]^2} \le \sqrt{2V_n\mathrm{Var}_{\widehat{P}_n}(f)} + \sqrt{2V_n}\mathbb{E}_{\widehat{P}_n}[f],$$

where we have used that $f \ge 0$. We thus obtain

$$\begin{aligned}\mathbb{E}[f] &\le \left(1 + \sqrt{2V_n}\right)\mathbb{E}_{\widehat{P}_n}[f] + \sqrt{2V_n\mathrm{Var}_{\widehat{P}_n}(f)} + 12M\sqrt{V_n\left(r_n^\star + \frac{7\max\{M,1\}}{M}\frac{t}{n}\right)} + 6r_n^\star + 14MB_n \\ &\le \left(1 + \sqrt{2V_n}\right)\mathbb{E}_{\widehat{P}_n}[f] + \sqrt{2V_n\mathrm{Var}_{\widehat{P}_n}(f)} + 6MV_n + 6M\left(r_n^\star + \frac{7\max\{M,1\}t}{Mn}\right) + 6r_n^\star + 14MB_n,\end{aligned}$$

where the second inequality follows because $\sqrt{ab} \le \frac{1}{2}a + \frac{1}{2}b$ for $a, b \ge 0$. Recalling the bound (21), which implies $\rho \ge nV_n$, $\rho \ge n(r_n^\star + \frac{7\max\{M,1\}t}{Mn})$, and $\rho/n \ge 6r_n^\star + 14MB_n$, we obtain that

$$\mathbb{E}[f] \le \left(1 + \sqrt{\frac{2\rho}{n}}\right)\mathbb{E}_{\widehat{P}_n}[f] + \sqrt{\frac{2\rho}{n}\mathrm{Var}_{\widehat{P}_n}(f)} + \frac{13M\rho}{n}.$$

Theorem 1 implies $\mathbb{E}_{\widehat{P}_n}[f] + \sqrt{\frac{2\rho}{n}\mathrm{Var}_{\widehat{P}_n}(f)} \le \sup_{P: D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[f(X)] + \frac{2M\rho}{n}$, so we immediately we arrive at

$$\mathbb{E}[f] \le \left(1 + 2\sqrt{\frac{2\rho}{n}}\right)\sup_{P: D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[f(X)] + \left(13 + 4\sqrt{\frac{2\rho}{n}}\right)\frac{M\rho}{n}$$

for all $f \in \mathcal{F}$ with probability at least $1 - 2e^{-t}$. This is the first result (22).

To show the second result, we simply apply Bernstein's inequality and the concentration inequalities for the standard deviation in Lemma 11. For any fixed $f \in \mathcal{F}$, by Bernstein's inequality, we have

$$\mathbb{E}_{\widehat{P}_n}[f] \le \mathbb{E}[f] + \sqrt{\frac{2t\mathrm{Var}(f)}{n}} + \frac{2Mt}{3n}$$

with probability at least $1 - e^{-t}$. From Lemma 11, we have

$$\sqrt{\mathrm{Var}_{\widehat{P}_n}(f)} \le \sqrt{\mathrm{Var}(f)} + \sqrt{\frac{2tM^2}{n}}$$

with probability at least $1 - e^{-t}$.

We thus obtain that for any fixed f,

$$\sup_{P: D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[f] \le \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{\frac{2\rho}{n}\mathrm{Var}_{\widehat{P}_n}(f)} \le \mathbb{E}[f] + \sqrt{\frac{2t}{n}\mathrm{Var}(f)} + \sqrt{\frac{2\rho}{n}\mathrm{Var}(f)} + \frac{2M\sqrt{\rho t}}{n} + \frac{2Mt}{3n}$$

with probability at least $1 - 2e^{-t}$. Noting that $\rho \ge 45Mt$ by assumption (21), so $\sqrt{\rho} + \sqrt{t} \le \sqrt{46\rho/45 + 45t} \le \sqrt{91\rho/45}$ and that always $2\sqrt{\rho t} \le 3\rho + \frac{1}{3}t$, we have that with probability at least $1 - 2e^{-t}$ that

$$\sup_{P: D_\phi(P\|\widehat{P}_n) \le \frac{\rho}{n}} \mathbb{E}_P[f] \le \mathbb{E}[f] + \sqrt{\frac{91\rho}{45n}\mathrm{Var}(f)} + \frac{3M\rho}{n} + \frac{Mt}{n}.$$

Noting that we could take f to minimize the right hand side of the preceding expression and that $\widehat{f}$ minimizes $\sup_{P: D_\phi(P\|\widehat{P}_n) \le \rho/n} \mathbb{E}_P[f]$, we have the result (23).

We first show the claim for $g \in \mathcal{F}_{\text{centered}} = \{f - \mathbb{E}[f] : f \in \mathcal{F}\}$. To see the claim for $g \in \mathcal{F}_{\text{centered}}$, let us fix $L \in \mathbb{N}$ to be chosen later, and for $l = 1, \ldots, L-1$ define the classes

$$\mathcal{F}_l := \left\{g \in \mathcal{F}_{\text{centered}} : e^{-l} r < \mathbb{E}[g^2] \le e^{-(l-1)} r\right\}, \quad \mathcal{F}_L := \left\{g \in \mathcal{F}_{\text{centered}} : \mathbb{E}[g^2] \le e^{-L} r\right\}$$

so that $\mathcal{F}_{\text{centered}} = \cup_{l=1}^L \mathcal{F}_l$. Let $z > 0$ be such that $t \le z$. Applying Lemma 14 (with the choice $\alpha = \frac{1}{2}$) to $\mathcal{F}_l$ for each $l = 1, \ldots, L-1$, we have with probability at least $1 - e^{-t}$, for every $g \in \mathcal{F}_l$

$$\begin{aligned}\mathbb{E}[g] &\le \mathbb{E}_{\widehat{P}_n}[g] + \sqrt{\frac{2te^{-(l-1)}r}{n}} + 3\mathbb{E}[\mathfrak{R}_n(\mathcal{F}_l)] + 5M\frac{t}{n} \\ &\le \mathbb{E}_{\widehat{P}_n}[g] + \sqrt{\frac{2et}{n}\mathbb{E}[g^2]} + 3\mathbb{E}[\mathfrak{R}_n(\mathcal{F}_l)] + 5M\frac{t}{n}\end{aligned}$$

where in the last line we have used $e^{-l} r \le \mathbb{E}[g^2]$ for $g \in \mathcal{F}_l$. Similarly, applying Lemma 14 to $\mathcal{F}_L$, then with probability at least $1 - e^{-t}$, for every $g \in \mathcal{F}_L$

$$\begin{aligned}\mathbb{E}[g] &\le \mathbb{E}_{\widehat{P}_n}[g] + \sqrt{\frac{2te^{-L}r}{n}} + 3\mathbb{E}[\mathfrak{R}_n(\mathcal{F}_L)] + 5M\frac{t}{n} \\ &\le \mathbb{E}_{\widehat{P}_n}[g] + \sqrt{\frac{2et}{n}\mathbb{E}[g^2]} + \sqrt{\frac{2te^{-L}r}{n}} + 3\mathbb{E}[\mathfrak{R}_n(\mathcal{F}_L)] + 5M\frac{t}{n}.\end{aligned}$$

Taking a union bound, we have with probability at least $1 - Le^{-t}$, for every $g \in \mathcal{F}_{\text{centered}}$

$$\mathbb{E}[g] \le \mathbb{E}_{\widehat{P}_n}[g] + \sqrt{\frac{2et}{n}\mathbb{E}[g^2]} + 3\mathbb{E}[\mathfrak{R}_n(\mathcal{F}_{\text{centered}})] + 5M\frac{t}{n} + \sqrt{\frac{2te^{-L}r}{n}}.$$

Noting that $\mathbb{E}[\mathfrak{R}_n(\mathcal{F}_{\text{centered}})] \le 2\mathbb{E}[\mathfrak{R}_n(\mathcal{F})]$ by Jensen's inequality, we take $L = \lceil \log \frac{rn}{M^2 t} \rceil$ and map t to $t + \log L$ to obtain the lemma. The case when the roles of $\mathbb{E}[f]$ and $\mathbb{E}_{\widehat{P}_n}[f]$ are reversed follows similarly.

D.1. Proof of Lemma 17

Let $r \ge r_n^\star$ be an arbitrary but fixed value to be choosen later. Using this r, define the self-normalized class of functions

$$\mathcal{G}_r := \left\{\sqrt{\frac{r}{\mathbb{E}[f^2] \vee r}} f : f \in \mathcal{F}\right\} \subseteq \left\{cf : f \in \mathcal{F}, \mathbb{E}[c^2 f^2] \le r, c \in [0,1]\right\}.$$

From the truncation by r, we have $\mathbb{E}[g^2] \le r$ for all $g \in \mathcal{G}_r$. Lemma 16 implies that with probability at least $1 - e^{-t}$, uniformly over $g \in \mathcal{G}_r$

$$\mathbb{E}[g] \le \mathbb{E}_{\widehat{P}_n}[g] + \sqrt{\frac{2e}{n}\mathbb{E}[g^2]\left(t + \log\left\lceil \log \frac{n}{t}\right\rceil\right)} + 6\mathbb{E}[\mathfrak{R}_n(\mathcal{G}_r)] + \frac{7M}{n}\left(t + \log\left\lceil \log \frac{n}{t}\right\rceil\right). \quad (34)$$

Using the sub-root property of ψ_n and that $\psi_n(r_n^\star) = r_n^\star$, we have the inequality

$$\psi_n(r) = \sqrt{r}\psi_n(r)/\sqrt{r} \le \sqrt{r}\psi_n(r_n^\star)/\sqrt{r_n^\star} = \sqrt{r r_n^\star}$$

for any $r \ge r_n^\star$, so

$$\mathbb{E}[\mathfrak{R}_n \mathcal{G}_r] \le \mathbb{E}[\mathfrak{R}_n\left\{cf : f \in \mathcal{F}, \mathbb{E}[c^2 f^2] \le r, c \in [0,1]\right\}] \le \psi_n(r) \le \sqrt{r r_n^\star}$$

Using this upper bound in Eq. (34) and recalling the notation $B_n = \frac{1}{n}\left(t + \log\lceil \log \frac{n}{t}\rceil\right)$, we get

$$\mathbb{E}[g] \le \mathbb{E}_{\widehat{P}_n}[g] + \sqrt{2eB_n\mathbb{E}[g^2]} + 6\sqrt{r_n^\star r} + 7MB_n. \quad (35)$$

Now, we return to choose the value r to optimize the bound (35). let r be the largest solution to $6\sqrt{r_n^\star r} + 7MB_n = 6r$. The following elementary lemma provides a bound on r.

Lemma 19 *Let x be the largest solution to $ax + b = \frac{x^2}{d}$ where $a, b, d > 0$. Then $a^2d^2 \le x^2 \le a^2d^2 + 2bd$.*

Proof From the quadratic formula, we have $x = \frac{1}{2}\left(ad + \sqrt{a^2d^2 + 4b}\right)$ from which the lower bound follows. From convexity of $z \mapsto z^2$ and $\sqrt{z_1 + z_2} \le \sqrt{z_1} + \sqrt{z_2}$ for $z_1, z_2 > 0$, we obtain the upper bound. ■

Lemma 19 immediately yields

$$r_n^\star \le r \le r_n^\star + \frac{7MB_n}{3}.$$

For each $g \in \mathcal{G}_r$, there exists $f \in \mathcal{F}$ such that $g = \sqrt{\frac{r}{\mathbb{E}[f^2] \vee r}} f$. If $\mathbb{E}[f^2] \le r$, we have $g = f$ and the bound (35) yields

$$\mathbb{E}[f] \le \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{2eB_n\mathbb{E}[f^2]} + 6r_n^\star + 14MB_n.$$

If $\mathbb{E}[f^2] > r$, rescaling g in the bound (35) and using the choice $6r = 6\sqrt{r_n^\star r} + 7MB_n$ yields

$$\begin{aligned}\mathbb{E}[f] &\le \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{2eB_n\mathbb{E}[f^2]} + 6\sqrt{r\mathbb{E}[f^2]} \\ &\le \mathbb{E}_{\widehat{P}_n}[f] + \sqrt{2eB_n\mathbb{E}[f^2]} + 6\sqrt{(r_n^\star + 7MB_n/3)\mathbb{E}[f^2]}\end{aligned}$$

instead. Combining the cases $\mathbb{E}[f^2] \lesseqgtr r$, we conclude that for all $f \in \mathcal{F}$,

$$\mathbb{E}[f] \le \mathbb{E}_{\widehat{P}_n}[f] + \left(\sqrt{2eB_n} + 6\sqrt{r_n^\star + 7MB_n/3}\right)\sqrt{\mathbb{E}[f^2]} + 6r_n^\star + 14MB_n$$

with probability at least $1 - e^{-t}$. Similarly, we can reverse the roles of $\mathbb{E}[f]$ and $\mathbb{E}_{\widehat{P}_n}[f]$ to get the second result.

D.2. Proof of Lemma 18

We frequently use the Rademacher contraction principle (Ledoux and Talagrand, 1991, Thm. 4.12) in what follows.

Lemma 20 *Let $\phi : \mathbb{R} \to \mathbb{R}$ be L-Lipschitz. Then, for every class $\mathcal{G}$*

$$\mathbb{E}_\epsilon[\mathfrak{R}_n(\phi \circ \mathcal{G})] \le L\mathbb{E}_\epsilon[\mathfrak{R}_n(\mathcal{G})]$$

where $\phi \circ \mathcal{G} = \{\phi \circ f : f \in \mathcal{G}\}$.

As in Section D.1, define the self-normalized functions in $\mathcal{F}$

$$\mathcal{G}_r := \left\{\sqrt{\frac{r}{\mathbb{E}[f^2] \vee r}} f : f \in \mathcal{F}\right\} \subseteq \left\{cf : f \in \mathcal{F}, \mathbb{E}[c^2 f^2] \le r, c \in [0,1]\right\}$$

where $r \ge r_n^\star$ will be choosen later. Let $\mathcal{G}_r^2 = \{g^2 : g \in \mathcal{G}_r\}$. From the truncation by r, we have that for all $g^2 \in \mathcal{G}_r^2$, $\mathrm{Var}(g^2) \le \mathbb{E}[g^4] \le M^2\mathbb{E}[g^2] \le M^2 r$. Let $c_1 = 3$ and $c_2 = \frac{7}{3}$. Then by Lemma 14 applied to $\mathcal{G}_r^2$, with probability at least $1 - e^{-t}$, for every $g \in \mathcal{G}_r$

$$\begin{aligned}
\mathbb{E}[g^2] &\le \mathbb{E}_{\widehat{P}_n}[g^2] + c_1\mathbb{E}[\mathfrak{R}_n(\mathcal{G}_r^2)] + M\sqrt{\frac{2rt}{n}} + c_2\frac{M^2 t}{n} \\
&\overset{(a)}{\le} \mathbb{E}_{\widehat{P}_n}[g^2] + 2c_1 M\mathbb{E}[\mathfrak{R}_n(\mathcal{G}_r)] + M\sqrt{\frac{2rt}{n}} + \frac{c_2 M^2 t}{n} \\
&\overset{(b)}{\le} \mathbb{E}_{\widehat{P}_n}[g^2] + 2c_1 M\sqrt{r r_n^\star} + M\sqrt{\frac{2rt}{n}} + \frac{c_2 M^2 t}{n} \qquad (36)
\end{aligned}$$

where in step (a) we used the contraction principle (Lemma 20) and that $x \mapsto x^2$ is $2M$-Lipschitz on $[-M, M]$, and in step (b), we used that $\psi_n(r) \le \sqrt{r r_n^\star}$ as in the proof of Lemma 17 in Section D.1.

Let $A = 2c_1 M\sqrt{r_n^\star} + M\sqrt{\frac{2t}{n}}$ and $D = \frac{c_2 M^2 t}{n}$. For any fixed $K > 1$, choose r to be the largest solution to $A\sqrt{r} + D = \frac{r}{K}$ so that the bound (36) becomes

$$\mathbb{E}[g^2] \le \mathbb{E}_{\widehat{P}_n}[g^2] + \frac{r}{D}.$$

From Lemma 19, we have

$$K^2A^2 \le r \le K^2A^2 + 2KD$$

and in particular, $r \ge K^2A^2 \ge r_n^\star$. For each $g \in \mathcal{G}_r$, there exists $f \in \mathcal{F}$ such that $g = \sqrt{\frac{r}{\mathbb{E}[f^2] \vee r}} f$. If $\mathbb{E}[f^2] \le r$, rescaling the inequality (36) and using the upper bound on r, we obtain

$$\mathbb{E}[f^2] \le \mathbb{E}_{\widehat{P}_n}[f^2] + \frac{r}{K} \le \mathbb{E}_{\widehat{P}_n}[f^2] + KA^2 + 2D.$$

If $\mathbb{E}[f^2] > r$, rescaling instead yields

$$\mathbb{E}[f^2] \le \mathbb{E}_{\widehat{P}_n}[f^2] + \frac{\mathbb{E}[f^2]}{K}.$$

Combining the two cases, we obtain

$$\mathbb{E}[f^2] \le \frac{K}{K-1}\mathbb{E}_{\widehat{P}_n}[f^2] + KA^2 + 2D.$$

Noting that $A \le 2\left(4c_1^2 M^2 r_\eta^\star + 2\frac{M^2 t}{n}\right)$ by convexity, we have the first result once we replace K with $\eta = K - 1 > 0$. The second result similarly follows by reversing the roles of $\mathbb{E}[f]$ and $\mathbb{E}_{\widehat{P}_n}[f]$ in the above argument.

Appendix E. Proof of Theorem 8

Recall our shorthand notation that $\pi(\theta) = \mathrm{argmin}_{\theta^* \in S_\star}\{\|\theta - \theta^*\|_2\}$ denotes the Euclidean projection of θ onto $S_\star$, which is a closed convex set. Define also the localized empirical deviation function

$$\Delta_n(\theta) := \mathbb{E}\left[\ell(\theta; X) - \ell(\pi(\theta); X)\right] - \mathbb{E}_{\widehat{P}_n}\left[\ell(\theta; X) - \ell(\pi(\theta); X)\right]. \qquad (37)$$

We begin with the following

Claim 21 *If $\widehat{S}_\star^\epsilon \not\subset S_\star^{2\epsilon}$, then*

$$\sup_{\theta \in S_\star^{2\epsilon}}\left\{\Delta_n(\theta) + \sqrt{\frac{2\rho}{n}\mathrm{Var}_{\widehat{P}_n}(\ell(\theta; X) - \ell(\pi(\theta); X))}\right\} \ge \epsilon. \qquad (38)$$

Deferring the proof of the claim, let us prove the theorem. First, the growth condition (26) shows that

$$S_\star^{2\epsilon} \subset \left\{\theta \in \Theta : \|\theta - \pi(\theta)\|_2 \le \left(\frac{2\epsilon}{\lambda}\right)^{\frac{1}{\gamma}}\right\} = \left\{\theta \in \Theta : \mathrm{dist}(\theta, S_\star) \le \left(\frac{2\epsilon}{\lambda}\right)^{\frac{1}{\gamma}}\right\}.$$

Therefore, we have for all $\theta \in S_\star^{2\epsilon}$ that

$$\mathrm{Var}_{\widehat{P}_n}(\ell(\theta; X) - \ell(\pi(\theta); X)) \le L^2\,\mathrm{dist}(\theta, S_\star)^2 \le L^2\left(\frac{2\epsilon}{\lambda}\right)^{\frac{2}{\gamma}},$$

and so by the assumption (27) that $\epsilon \ge (\frac{8L^2\rho}{n})^{\frac{\gamma}{2(\gamma-1)}}(\frac{2}{\lambda})^{\frac{1}{\gamma-1}}$, we have

$$\sqrt{\frac{2\rho}{n}\mathrm{Var}_{\widehat{P}_n}(\ell(\theta; X) - \ell(\pi(\theta); X))} \le L\sqrt{\frac{2\rho}{n}}\left(\frac{2\epsilon}{\lambda}\right)^{\frac{1}{\gamma}} \le \frac{\epsilon}{2}.$$

In particular, if the event (38) holds then

$$\sup_{\theta \in S_\star^{2\epsilon}} \Delta_n(\theta) \ge \frac{\epsilon}{2},$$

and recalling the definition (37) of Δ_n, it then follows that

$$\mathbb{P}\left(\widehat{S}_\star^\epsilon \not\subset S_\star^{2\epsilon}\right) \le \mathbb{P}\left(\sup_{\theta \in S_\star^{2\epsilon}} \Delta_n(\theta) \ge \frac{\epsilon}{2}\right). \tag{39}$$

To bound the probability (39), we use standard bounded difference and symmetrization arguments (e.g. Boucheron et al., 2013, Theorem 6.5). Letting $f(X_1, \ldots, X_n) := \sup_{\theta \in S_\star^{2\epsilon}} \Delta_n(\theta)$, the function f satisfies bounded differences:

$$\begin{aligned}
&\sup_{x, x' \in \mathcal{X}} |f(X_1, \cdots, X_{j-1}, x, X_{j+1}, \cdots, X_n) - f(X_1, \cdots, X_{j-1}, x', X_{j+1}, \cdots, X_n)| \\
&\le \sup_{x, x' \in \mathcal{X}} \sup_{\theta \in S_\star^{2\epsilon}} \left| \frac{1}{n}(\ell(\theta; x) - \ell(\pi(\theta); x)) - \frac{1}{n}(\ell(\theta; x') - \ell(\pi(\theta); x')) \right| \\
&\le \frac{2L}{n} \sup_{\theta \in S_\star^{2\epsilon}} \operatorname{dist}(\theta, S_\star) \le \frac{2L}{n} \left(\frac{2\epsilon}{\lambda}\right)^{\frac{1}{\gamma}}
\end{aligned}$$

for $j = 1, \ldots, n$. Using the standard symmetrization inequality $\mathbb{E}[\sup_{\theta \in S_\star^{2\epsilon}} \Delta_n(\theta)] \le 2\mathbb{E}[\mathfrak{R}_n(S_\star^{2\epsilon})]$ and the bounded differences inequality (Boucheron et al., 2013, Theorem 6.5), we have

$$\mathbb{P}\left(\sup_{\theta \in S_\star^{2\epsilon}} \Delta_n(\theta) \ge 2\mathbb{E}[\mathfrak{R}_n(S_\star^{2\epsilon})] + t\right) \le \exp\left(-\frac{nt^2}{2L^2}\left(\frac{\lambda}{2\epsilon}\right)^{\frac{2}{\gamma}}\right)$$

for all $t \ge 0$. Letting $u = \frac{nt^2}{2L^2}\left(\frac{\lambda}{2\epsilon}\right)^{\frac{2}{\gamma}}$ above and recalling the assumption (27) upper bounding $\mathbb{E}[\mathfrak{R}_n(S_\star^{2\epsilon})]$, we have $\mathbb{P}(\sup_{\theta \in S_\star^{2\epsilon}} \Delta_n(\theta) \ge \frac{\epsilon}{2}) \le e^{-u}$. The theorem follows from the bound (39).

Proof of Claim 21 If $\widehat{S}_\star^\epsilon \not\subset S_\star^{2\epsilon}$, then certainly it is the case that there is some $\theta \in \Theta \setminus S_\star^{2\epsilon}$ such that

$$R_n(\theta, \mathcal{P}_n) \le \inf_{\theta \in \Theta} R_n(\theta, \mathcal{P}_n) + \epsilon \le R_n(\pi(\theta), \mathcal{P}_n) + \epsilon.$$

Using the convexity of R_n, we have for all $t \in [0, 1]$ that

$$R_n(t\theta + (1-t)\pi(\theta), \mathcal{P}_n) \le tR_n(\theta, \mathcal{P}_n) + (1-t)R_n(\pi(\theta), \mathcal{P}_n) \le R_n(\pi(\theta), \mathcal{P}_n) + t\epsilon.$$

For all $t \in [0, 1]$, we have by definition of orthogonal projection (because the vector $\theta - \pi(\theta)$ belongs to the normal cone to $S_\star$ at $\pi(\theta)$; cf. (Hiriart-Urruty and Lemaréchal, 1993, Prop. III.5.3.3)) that $\pi(t\theta + (1-t)\pi(\theta)) = \pi(\theta)$. Thus, choosing t appropriately, there exists $\theta' \in \operatorname{bd} S_\star^{2\epsilon}$ with $\theta' = t\theta + (1-t)\pi(\theta)$, $\pi(\theta') = \pi(\theta)$, and $R_n(\theta', \mathcal{P}_n) \le R_n(\pi(\theta'), \mathcal{P}_n) + \epsilon$.

Adding and subtracting the risk $R(\theta)$ and $R(\pi(\theta))$, we have that for some $\theta \in \operatorname{bd} S_\star^{2\epsilon}$ that

$$R_n(\theta, \mathcal{P}_n) - R(\theta) + R(\pi(\theta)) - R_n(\pi(\theta), \mathcal{P}_n) \le R(\pi(\theta)) - R(\theta) + \epsilon \le -\epsilon,$$

where we have used that $R(\theta) = R(\pi(\theta)) + 2\epsilon$ by construction. Multiplying by -1 on each side of the preceding display and taking suprema, we find that

$$\begin{aligned}
\epsilon &\le \sup_{\theta \in S_\star^{2\epsilon}} \left\{R(\theta) - R_n(\theta, \mathcal{P}_n) - (R(\pi(\theta)) - R_n(\pi(\theta), \mathcal{P}_n))\right\} \\
&\le \sup_{\theta \in S_\star^{2\epsilon}} \sup_{P : D_\phi(P \| \widehat{P}_n) \le \rho/n} \left\{R(\theta) - R(\pi) + \mathbb{E}_P\left[\ell(\pi(\theta); X) - \ell(\theta; X)\right]\right\}.
\end{aligned}$$

Applying the upper bound in inequality (10) of Theorem 1 gives the claim.

Appendix F. Proof of Theorem 10

We begin by establishing a few technical lemmas, after which the proof of the theorem follows essentially standard arguments in asymptotics. To prove Theorem 10, we first show that (eventually) we have the exact expansion

$$R_n(\theta, \mathcal{P}_n) = \mathbb{E}_{\widehat{P}_n}[\ell(\theta, X)] + \sqrt{\frac{2\rho \operatorname{Var}_{\widehat{P}_n}(\ell(\theta, X))}{n}}$$

for all θ in a neighborhood of $\theta^\star$. As in the proof of Theorem 1, this exact equality holds once there is suitable variability in the values $\ell(\theta, X_i)$ over $i = 1, \ldots, n$, however, we require a bit more care as the values $\ell(\theta, X_i)$ may be unbounded below and above. Heuristically, however, assuming that we have this exact expansion and that $\widehat{\theta}_n^{\rm rob} - \theta^\star = O_P(n^{-\frac{1}{2}})$, then we can write the expansions

$$\begin{aligned}
0 &= \nabla_\theta R_n(\widehat{\theta}_n^{\rm rob}, \mathcal{P}_n) \\
&= \nabla \frac{1}{n}\sum_{i=1}^n \ell(\theta^\star, X_i) + \nabla^2\left(\frac{1}{n}\sum_{i=1}^n \ell(\theta^\star, X_i)\right)(\widehat{\theta}_n^{\rm rob} - \theta^\star) + \nabla\sqrt{\frac{2\rho \operatorname{Var}_{\widehat{P}_n}(\ell(\widehat{\theta}_n^{\rm rob}, X))}{n}} + o_P(n^{-\frac{1}{2}}) \\
&= \frac{1}{n}\sum_{i=1}^n \nabla\ell(\theta^\star, X_i) + \nabla^2 R(\theta^\star)(\widehat{\theta}_n^{\rm rob} - \theta^\star) + \nabla\sqrt{\frac{2\rho \operatorname{Var}(\ell(\theta^\star, X))}{n}} + o_P(n^{-\frac{1}{2}}).
\end{aligned}$$

Multiplying by $\sqrt{n}$ and solving for $\widehat{\theta}_n^{\rm rob}$ in the preceding expression, computing $\nabla\sqrt{\operatorname{Var}(\ell(\theta^\star, X))}$ then yields the theorem.

The remainder of the proof makes this heuristic rigorous, and the outline is as follows:

1. We show that there is a uniform expansion of the form (12) in a neighborhood of $\theta^\star$. (See Section F.1.)

2. Using the uniform expansion, we can then leverage standard techniques for asymptotic analysis of finite-dimensional estimators (see, e.g. van der Vaart and Wellner (1996) or Lehmann and Casella (1998)), which proceed by performing a Taylor expansion of the objective in a neighborhood of the optimum and using local asymptotic normality arguments. (See Section F.2.)

F.1. The uniform variance expansion

To lighten notation, we define a few quantities similar to those used in the proof of Theorem 1. Let

$$Z(\theta) := \ell(\theta, X) - \mathbb{E}[\ell(\theta, X)]$$

be the deviation of $\ell(\theta, X)$ around its mean (the risk), and similarly let $Z_i(\theta)$ be the version of this quantity for observation X_i. In addition, let $s_n^2(\theta) = \operatorname{Var}_{\widehat{P}_n}(Z(\theta))$ be the empirical variance of $Z(\theta)$, which is identical to the empirical variance of $\ell(\theta, X)$.

Now, recall the problem

$$\operatorname*{maximize}_P \ \mathbb{E}_P[Z(\theta)] \ \text{ subject to } D_\phi(P \| \widehat{P}_n) \le \frac{\rho}{n},$$

and for each $\theta \in \Theta$, let $p(\theta) = \operatorname{argmax}_{p \in \mathcal{P}_n} \sum_{i=1}^n p_i Z_i(\theta)$ be the solution (probability) vectors. Following expression (9) we see for any $\epsilon \geq 0$ that

$$\min_{i \in [n]} \frac{\sqrt{2\rho}(Z_i(\theta) - \overline{Z}(\theta))}{\sqrt{n} s_n(\theta)} \geq -1 \quad \text{for all } \theta \in \theta^\star + \epsilon\mathbb{B}$$

is sufficient for the exact variance expansion to hold. We now show that this is indeed likely. Let $\epsilon > 0$ be small enough that Assumption A holds, that is, the random Lipschitz function $L(X)$ satisfies $|\ell(\theta, x) - \ell(\theta', x)| \leq L(x)\|\theta - \theta'\|$ for $\theta, \theta' \in \theta^\star + \epsilon\mathbb{B}$. Then because

$$\begin{aligned}\left|\sqrt{n} s_n(\theta) - \sqrt{n} s_n(\theta')\right| &\leq \sup_{u : \|u\|_2 \leq 1} \sum_{i=1}^n u_i \left(\ell(\theta, X_i) - \ell(\theta', X_i)\right) \\ &\leq \sup_{u : \|u\|_2 \leq 1} \sum_{i=1}^n u_i L(X_i) \left\|\theta - \theta'\right\| \leq \sqrt{\sum_{i=1}^n L^2(X_i)} \left\|\theta - \theta'\right\|\end{aligned}$$

so $\theta \mapsto s_n(\theta)$ is $\sqrt{\frac{1}{n}\sum_{i=1}^n L(X_i)^2}$-Lipschitz for $\theta \in \theta^\star + \epsilon\mathbb{B}$, we have

$$\inf_{\theta \in \theta^\star + \epsilon\mathbb{B}} \min_{i \in [n]} \left\{ \frac{\sqrt{2\rho}(Z_i(\theta) - \overline{Z}(\theta))}{\sqrt{n} s_n(\theta)} \right\} \geq \min_{i \in [n]} \frac{\sqrt{2\rho}(Z_i(\theta^\star) - \overline{Z}(\theta^\star) - 2\epsilon L(X_i))}{\sqrt{n\left(s_n(\theta^\star) - \epsilon\sqrt{\frac{1}{n}\sum_{j=1}^n L(X_j)^2}\right)}}.$$

Summarizing our development thus far, we have the following lemma.

Lemma 22 *Let the conditions of the previous paragraph hold. Then*

$$\min_{i \in [n]} \left\{ \sqrt{2\rho}(Z_i(\theta^\star) - \overline{Z}(\theta^\star) - 2\epsilon L(X_i)) \right\} \geq \sqrt{n} \sqrt{s_n(\theta^\star) - \epsilon\left(\frac{1}{n}\sum_{i=1}^n L(X_i)^2\right)^{\frac{1}{2}}}$$

implies that

$$R_n(\theta, \mathcal{P}_n) = \mathbb{E}_{\widehat{P}_n}[\ell(\theta, X)] + \sqrt{\frac{2\rho}{n}\operatorname{Var}_{\widehat{P}_n}(\ell(\theta, X))} \quad \textit{for all } \theta \in \theta^\star + \epsilon\mathbb{B}.$$

Now, we use the following standard result to show that the conditions of Lemma 22 eventually hold with probability one.

Lemma 23 (Owen (Owen, 1990), Lemma 3) *Let Y_i be independent random variables with $\sup_i \mathbb{E}[Y_i^2] < \infty$. Then $n^{-\frac{1}{2}} \max_{1 \leq i \leq n} |Y_i| \stackrel{a.s.}{\to} 0$.*

Based on Lemma 23 and the strong law of large numbers, we see immediately that

$$\frac{1}{\sqrt{n}} \max_{1 \leq i \leq n} |Z_i(\theta^\star)| \stackrel{a.s.}{\to} 0, \quad \text{and} \quad \frac{1}{\sqrt{n}} \max_{1 \leq i \leq n} L(X_i) \stackrel{a.s.}{\to} 0,$$

because $\mathbb{E}[Z(\theta^\star)^2] < \infty$ and $\mathbb{E}[L(X_i)^2] < \infty$. Applying the strong law of large numbers to obtain

$$s_n(\theta^\star) \stackrel{a.s.}{\to} \sqrt{\operatorname{Var}(\ell(\theta^\star, X))} \quad \text{and} \quad \sqrt{\frac{1}{n}\sum_{i=1}^n L(X_i)^2} \stackrel{a.s.}{\to} \sqrt{\mathbb{E}[L(X)^2]},$$

we see immediately that for small enough $\epsilon > 0$, the condition of Lemma 22 holds eventually with probability 1. That is, the following uniform expansion holds.

Lemma 24 *There exists $\epsilon > 0$ such that, with probability 1, there exists an N (which may be random) such that $n \geq N$ implies*

$$R_n(\theta, \mathcal{P}_n) = \mathbb{E}_{\widehat{P}_n}[\ell(\theta, X)] + \sqrt{\frac{2\rho \operatorname{Var}_{\widehat{P}_n}(\ell(\theta, X))}{n}} \quad \textit{for all } \theta \in \theta^\star + \epsilon\mathbb{B}.$$

F.2. Asymptotics and Taylor expansions

Let $\mathcal{E}_{n,\text{exact}}$ be the event that the exact variance expansion of Lemma 24 occurs for $\theta \in \theta^\star + \epsilon\mathbb{B}$. Now that we know that $\mathbb{P}(\mathcal{E}_{n,\text{exact}} \text{ eventually}) = 1$, we may perform a few asymptotic expansions of the variance-regularized objective to provide the convergence guarantees specified by the theorem. We use the following lemma.

Lemma 25 *Let the conditions of the theorem hold. If*

$$\widehat{\theta}_n^{\text{rob}} \in \operatorname*{argmin}_\theta R_n(\theta, \mathcal{P}_n) \quad \textit{then} \quad \widehat{\theta}_n^{\text{rob}} \stackrel{a.s.}{\to} \theta^\star. \tag{40}$$

The proof is standard, but for completeness we include it in Section G.3.

By combining Lemmas 24 and 25, we see that with probability 1, for any $\epsilon > 0$, we eventually have both

$$\|\widehat{\theta}_n^{\text{rob}} - \theta^\star\|_2 < \epsilon \quad \text{and} \quad R_n(\widehat{\theta}_n^{\text{rob}}, \mathcal{P}_n) = \mathbb{E}_{\widehat{P}_n}[\ell(\widehat{\theta}_n^{\text{rob}}, X)] + \sqrt{\frac{2\rho}{n}\operatorname{Var}_{\widehat{P}_n}(\ell(\widehat{\theta}_n^{\text{rob}}, X))}.$$

Assume for the remainder of the argument that both of these conditions hold. Standard results on subdifferentiability of maxima of collections of convex functions (Hiriart-Urruty and Lemaréchal, 1993, Chapter X) give that $R_n(\theta, \mathcal{P}_n)$ is differentiable near $\theta^\star$, and thus

$$\begin{aligned}0 &= \nabla R_n(\widehat{\theta}_n^{\text{rob}}, \mathcal{P}_n) = \mathbb{E}_{\widehat{P}_n}[\nabla\ell(\widehat{\theta}_n^{\text{rob}}, X)] + \nabla\sqrt{\frac{2\rho}{n}\operatorname{Var}_{\widehat{P}_n}(\ell(\widehat{\theta}_n^{\text{rob}}, X))} \\ &= \frac{1}{n}\sum_{i=1}^n \nabla\ell(\widehat{\theta}_n^{\text{rob}}, X_i) + \sqrt{\frac{2\rho}{n}} \frac{\mathbb{E}_{\widehat{P}_n}\left[(\nabla\ell(\widehat{\theta}_n^{\text{rob}}, X) - \mathbb{E}_{\widehat{P}_n}[\nabla\ell(\widehat{\theta}_n^{\text{rob}}, X)])(\ell(\widehat{\theta}_n^{\text{rob}}, X) - \mathbb{E}_{\widehat{P}_n}[\ell(\widehat{\theta}_n^{\text{rob}}, X)])\right]}{\sqrt{\operatorname{Var}_{\widehat{P}_n}(\ell(\widehat{\theta}_n^{\text{rob}}, X))}}.\end{aligned} \tag{41}$$

Because $\widehat{\theta}_n^{\text{rob}} \stackrel{a.s.}{\to} \theta^\star$, by the continuous mapping theorem and local uniform convergence of the empirical expectations $\mathbb{E}_{\widehat{P}_n}[\cdot]$ to $\mathbb{E}[\cdot]$, the second term of expression (41) satisfies

$$\frac{\mathbb{E}_{\widehat{P}_n}\left[(\nabla\ell(\widehat{\theta}_n^{\text{rob}}, X) - \mathbb{E}_{\widehat{P}_n}[\nabla\ell(\widehat{\theta}_n^{\text{rob}}, X)])(\ell(\widehat{\theta}_n^{\text{rob}}, X) - \mathbb{E}_{\widehat{P}_n}[\ell(\widehat{\theta}_n^{\text{rob}}, X)])\right]}{\sqrt{\operatorname{Var}_{\widehat{P}_n}(\ell(\widehat{\theta}_n^{\text{rob}}, X))}} = \underbrace{\frac{\operatorname{Cov}(\nabla\ell(\theta^\star, X), \ell(\theta^\star, X))}{\sqrt{\operatorname{Var}(\ell(\theta^\star, X))}}}_{=: b(\theta^\star)} + o_P(1).$$

For simplicity, we let $b(\theta^\star)$ denote the final term, which we shall see becomes an asymptotic bias. Thus, performing a Taylor expansion of the terms $\nabla\ell(\widehat{\theta}_n^{\text{rob}}, X_i)$ around $\theta^\star$ in equality (41), there exist (random) error matrices $E_n(X_i)$, where $\|E_n(X_i)\| \leq H(X_i)\|\widehat{\theta}_n^{\text{rob}} - \theta^\star\|$

by Assumption A, such that

$$0 = \mathbb{E}_{\widehat{P}_n}[\nabla \ell(\theta^\star, X)] + \frac{1}{n}\sum_{i=1}^n \left(\nabla^2 \ell(\theta^\star, X_i) + E_n(X_i)\right)(\widehat{\theta}_n^{\rm rob} - \theta^\star) + \sqrt{\frac{2\rho}{n}}(b(\theta^\star) + o_P(1))$$
$$= \mathbb{E}_{\widehat{P}_n}[\nabla \ell(\theta^\star, X)] + \left(\nabla^2 R(\theta^\star) + o_P(1)\right)(\widehat{\theta}_n^{\rm rob} - \theta^\star) + \sqrt{\frac{2\rho}{n}}(b(\theta^\star) + o_P(1)).$$

Multiplying both sides by $\sqrt{n}$, using that $\nabla^2 R(\theta^\star) + o_P(1)$ is eventually invertible, and applying the continuous mapping theorem, we have

$$\sqrt{n}(\widehat{\theta}_n^{\rm rob} - \theta^\star) = -(\nabla^2 R(\theta^\star) + o_P(1))^{-1}\frac{1}{\sqrt{n}}\sum_{i=1}^n \nabla \ell(\theta^\star, X_i) - \sqrt{2\rho} b(\theta^\star) + o_P(1).$$

The first term on the right side of the above display converges in distribution to a $\mathsf{N}(0, \Sigma)$ distribution, where

$$\Sigma = (\nabla^2 R(\theta^\star))^{-1}\operatorname{Cov}(\nabla \ell(\theta^\star, X))(\nabla^2 R(\theta^\star))^{-1},$$

so that

$$\sqrt{n}(\widehat{\theta}_n^{\rm rob} - \theta^\star) \stackrel{d}{\to} \mathsf{N}\left(-\sqrt{2\rho}\, b(\theta^\star), \Sigma\right)$$

as claimed in the theorem statement.

Appendix G. Proofs of Technical Lemmas

G.1. Proof of Inequality (24)

Define the Gaussian complexity

$$\mathfrak{G}_n(\{\ell \circ \mathcal{H}\}_{\le r}) := \mathbb{E}\left[\sup_{h \in \mathbb{B}_{\mathcal{H}}, c \in [0,1]} \sum g_i c \ell(h(x_i), y_i) \mid \mathbb{E}[\ell(h(X), Y)^2] \le r/c^2\right], \tag{42}$$

where $g_i \stackrel{\rm iid}{\sim} \mathsf{N}(0,1)$ (here we recall the standard result (Bartlett and Mendelson, 2002) that Gaussian complexity upper bounds Rademacher complexities up to a constant). Now, the set $h - h^\star$ such that $h \in \mathbb{B}_{\mathcal{H}}$ is contained in $2\mathbb{B}_{\mathcal{H}}$, which is convex. Moreover, we have $\mathbb{E}[\ell(h(X), Y)^2] = \mathbb{E}[(h(X) - h^\star(X))^2] + \sigma^2$, and so we have for any c that

$$\{h \in \mathbb{B}_{\mathcal{H}} \mid c^2 \mathbb{E}[\ell(h(X), Y)^2] \le r\} \subset \{h \in \mathbb{B}_{\mathcal{H}} \mid \mathbb{E}[(h(X) - h^\star(X))^2] \le r/c^2\},$$

and $\mathbb{E}[\ell(h(X), Y)^2] \le r/c^2$ also implies $\sigma^2 \le r/c^2$. Returning to expression (42) and enlarging the sets over which we take suprema, we thus obtain

$$\mathfrak{G}_n(\ell \circ \mathcal{H}) \le \mathbb{E}\left[\sup_{h \in \mathbb{B}_{\mathcal{H}}, c_1, c_2 \in [0,1]} \sum_{i=1}^n g_i |c_1(h(x_i) - h^\star(x_i)) - c_2 \xi_i| \mid \mathbb{E}[(h(X) - h^\star(X))^2] \le \frac{r}{c_1^2}, \sigma^2 \le \frac{r}{c_2^2}\right]$$
$$\le \mathbb{E}\left[\sup_{f \in 2\mathbb{B}_{\mathcal{H}}, c \in [0,1]} \sum_{i=1}^n g_i |f(x_i) - c\xi_i| \mid \mathbb{E}[f(X)^2] \le r, \sigma^2 \le r/c^2\right],$$

where we have used that $h - h^\star \in 2\mathbb{B}_{\mathcal{H}}$ and that the set $\mathbb{B}_{\mathcal{H}}$ is convex to obtain the second inequality. We now upper bound the final display using the classical Sudakov-Fernique comparison theorem (e.g. Chatterjee, 2005). Indeed, define the two Gaussian processes indexed by $f \in \mathcal{H}$ and $c \in [0,1]$ by $Y_{f,c} = \sum_{i=1}^n g_i |f(x_i) - c\xi_i|$ and $Z_{f,c} = \sum_{i=1}^n g_i f(x_i) + c\sum_{i=1}^n w_i \xi_i$, where $g_i \stackrel{\rm iid}{\sim} \mathsf{N}(0,1)$ and $w_i \stackrel{\rm iid}{\sim} \mathsf{N}(0,1)$. Then we have for any $f_1, f_2 \in \mathcal{H}$ and $c_1, c_2 \in [0,1]$ that

$$\mathbb{E}[(Y_{f_1,c_1} - Y_{f_2,c_2})^2] = \sum_{i=1}^n \left(|f_1(x_i) - c_1\xi_i| - |f_2(x_i) - c_2\xi_i|\right)^2$$
$$\le \sum_{i=1}^n \left(f_1(x_i) - f_2(x_i) + (c_2 - c_1)\xi_i\right)^2$$
$$\le 2\sum_{i=1}^n (f_1(x_i) - f_2(x_i))^2 + 2(c_2 - c_1)^2 \sum_{i=1}^n \xi_i^2.$$

Moreover, $\mathbb{E}[(Z_{f_1,c_1} - Z_{f_2,c_2})^2] = \sum_{i=1}^n (f_1(x_i) - f_2(x_i))^2 + (c_1 - c_2)^2 \sum_{i=1}^n \xi_i^2$. Thus, the Sudakov-Fernique inequality guarantees that $\mathbb{E}[\sup_{f,c} Y_{f,c}] \le \sqrt{2}\mathbb{E}[\sup_{f,c} Z_{f,c}]$, and

$$\mathfrak{G}_n(\ell \circ \mathcal{H}) \lesssim \mathbb{E}\left[\sup_{f \in 2\mathbb{B}_{\mathcal{H}}} \sum_{i=1}^n g_i f(x_i) \mid \mathbb{E}[f(X)^2] \le r\right] + \mathbb{E}\left[\sup_{c \in [0,1]} c\sum_{i=1}^n w_i \xi_i \mid c^2\sigma^2 \le r\right].$$

The last term in the expression has bound $\sqrt{nr}$ by Jensen's inequality and the relaxation that $c \in [-1,1]$. For the first term, Mendelson (2003, Thm. 2.1) shows that for RKHS with kernel eigenvalues $\lambda_1, \lambda_2, \ldots$, we have

$$\mathbb{E}\left[\sup_{f \in 2\mathbb{B}_{\mathcal{H}}} \sum_{i=1}^n g_i f(X_i) \mid \mathbb{E}[f(X)^2] \le r\right] \lesssim \sqrt{n}\left(\sum_{j=1}^\infty \min\{\lambda_j, r\}\right)^{\frac{1}{2}},$$

which yields our desired claim (24).

G.2. Proof of Lemma 7

Defining $N_y := \operatorname{card}\{i \in [n] : X_i = y\}$ for $y \in \{-1, 0, 1\}$, we immediately obtain

$$\mathbb{E}_{\widehat{P}_n}[\ell(\theta; X)] = \frac{1}{n}\left[N_{-1}|\theta + 1| + N_1|\theta - 1| + N_0|\theta| - (n - N_0)\right],$$

because $N_1 + N_{-1} + N_0 = n$. In particular, we find that the empirical risk minimizer θ satisfies

$$\widehat{\theta}_n^{\rm erm} := \operatorname*{argmin}_{\theta \in \mathbb{R}} \mathbb{E}_{\widehat{P}_n}[\ell(\theta; X)] = \begin{cases} 1 & \text{if } N_1 > N_0 + N_{-1} \\ -1 & \text{if } N_{-1} > N_0 + N_1 \\ \in [-1,1] & \text{otherwise.} \end{cases}$$

On the events $N_1 > N_{-1} + N_0$ or $N_{-1} > N_0 + N_1$, which are disjoint, then, we have

$$R(\widehat{\theta}_n^{\rm erm}) = \delta = R(\theta^\star) + \delta.$$

Let us give a lower bound on the probability of this event. Noting that marginally $N_1 \sim \mathsf{Bin}(n, \frac{1-\delta}{2})$ and using $N_0 + N_{-1} = n - N_1$, we have $N_1 > N_0 + N_{-1}$ if and only if $N_1 > \frac{n}{2}$, and we would like to lower bound

$$\mathbb{P}\left(N_1 > \frac{n}{2}\right) = \mathbb{P}\left(\mathsf{Bin}\left(n, \frac{1-\delta}{2}\right) > \frac{n}{2}\right) = \mathbb{P}\left(\mathsf{Bin}\left(n, \frac{1+\delta}{2}\right) < \frac{n}{2}\right).$$

Letting $\Phi(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{t} e^{-u^2/2} du$ denote the standard Gaussian CDF, then Zubkov and Serov (2013) show that

$$\mathbb{P}\left(N_1 \geq \frac{n}{2}\right) \geq \Phi\left(-\sqrt{2n D_{\rm kl}\left(\frac{1}{2} \| \frac{1+\delta}{2}\right)}\right)$$

where $D_{\rm kl}\left(p\|q\right) = p \log \frac{p}{q} + (1-p) \log \frac{1-p}{1-q}$ denotes the binary KL-divergence. We have by standard bounds on the KL-divergence (Tsybakov, 2009, Lemma 2.7) that $D_{\rm kl}(\frac{1}{2} \| \frac{1+\delta}{2}) \leq \frac{\delta^2}{2(1-\delta^2)}$, so that

$$\mathbb{P}\left(N_1 > \frac{n}{2} \text{ or } N_{-1} > \frac{n}{2}\right) \geq 2\Phi\left(-\sqrt{\frac{n\delta^2}{1-\delta^2}}\right) - 2\mathbb{P}\left(N_1 = \frac{n}{2}\right).$$

For n odd, the final probability is 0, while for n even, we have

$$\mathbb{P}\left(N_1 = \frac{n}{2}\right) = 2^{-n} \binom{n}{n/2} (1-\delta^2)^{n/2} \leq (1-\delta^2)^{n/2} \sqrt{\frac{2}{\pi n}},$$

where the inequality uses that $\binom{2n}{n} \leq \frac{4^n}{\sqrt{\pi n}}$ by Stirling's approximation. Summarizing, we find that

$$\mathbb{P}\left(N_1 > \frac{n}{2} \text{ or } N_{-1} > \frac{n}{2}\right) \geq 2\Phi\left(-\sqrt{\frac{n\delta^2}{1-\delta^2}}\right) - (1-\delta^2)^{n/2} \sqrt{\frac{8}{\pi n}}.$$

G.3. Proof of Lemma 25

Under the conditions of the theorem, the compactness of $\theta^\star + \epsilon\mathbb{B}$ guarantees that

$$\sup_{\theta \in \theta^\star + \epsilon\mathbb{B}} |\mathbb{E}_{\widehat{P}_n}[\ell(\theta, X)] - R(\theta)| \stackrel{a.s.}{\to} 0,$$

as the functions $\theta \mapsto \ell(\theta, x)$ are Lipschitz in a neighborhood of $\theta^\star$ by Assumption A. Similarly,

$$\sup_{\theta \in \theta^\star + \epsilon\mathbb{B}} \left|\mathrm{Var}_{\widehat{P}_n}(\ell(\theta, X)) - \mathrm{Var}(\ell(\theta, X))\right| \stackrel{a.s.}{\to} 0,$$

using the local Lipschitzness of $\nabla^2 \ell$. (See, for example, the Glivenko-Cantelli results in Chapters 2.4–2.5 of van der Vaart and Wellner (1996).) Thus, using the two-sided bounds (10) of Theorem 1, we have that

$$\begin{aligned}
&\sup_{\theta \in \theta^\star + \epsilon\mathbb{B}} |R_n(\theta, \mathcal{P}_n) - R(\theta)| \\
&\leq \sup_{\theta \in \theta^\star + \epsilon\mathbb{B}} \left|\mathbb{E}_{\widehat{P}_n}[\ell(\theta, \mathcal{P}_n)] - R(\theta)\right| + \sqrt{\frac{2\rho}{n}} \sup_{\theta \in \theta^\star + \epsilon\mathbb{B}} \sqrt{\mathrm{Var}_{\widehat{P}_n}(\ell(\theta, X))} \stackrel{a.s.}{\to} 0.
\end{aligned}$$

Now, we use the fact that $\nabla^2 R(\theta^\star) \succ 0$, and that $\theta \mapsto \nabla^2 R(\theta)$ is continuous in a neighborhood of $\theta^\star$. Fix $\epsilon > 0$ small enough that the preceding uniform convergence guarantees hold over $\theta^\star + 2\epsilon\mathbb{B}$ and $\nabla^2 R(\theta) \succeq \lambda I$ for some $\lambda > 0$ and all $\theta \in \theta^\star + 2\epsilon\mathbb{B}$. Let $\theta \notin \theta^\star + \epsilon\mathbb{B}$, but $\theta \in \theta^\star + 2\epsilon\mathbb{B}$. Then for sufficently large n, we have that

$$\begin{aligned}
R_n(\theta, \mathcal{P}_n) \geq \mathbb{E}_{\widehat{P}_n}[\ell(\theta, X)] &\stackrel{(i)}{\geq} R(\theta) - \frac{\lambda}{4}\epsilon^2 \\
&\stackrel{(ii)}{\geq} R(\theta^\star) + \frac{\lambda}{2}\left\|\theta - \theta^\star\right\|_2^2 - \frac{\lambda}{4}\epsilon^2 \stackrel{(iii)}{\geq} R(\theta^\star) + \frac{\lambda}{4}\epsilon^2 \\
&\stackrel{(iv)}{\geq} \mathbb{E}_{\widehat{P}_n}[\ell(\theta^\star, X)] + \frac{\lambda}{4}\epsilon^2 - \frac{\lambda}{8}\epsilon^2 = \mathbb{E}_{\widehat{P}_n}[\ell(\theta^\star, X)] + \frac{\lambda}{8}\epsilon^2,
\end{aligned}$$

where inequalities (i) and (iv) follow from the uniform convergence guarantee, inequality (ii) from the strong convexity of R near $\theta^\star$, and (iii) because $\|\theta - \theta^\star\|_2 \geq \epsilon$. Finally, we have that

$$\mathbb{E}_{\widehat{P}_n}[\ell(\theta^\star, X)] \geq R_n(\theta^\star, \mathcal{P}_n) - \underbrace{\sqrt{\frac{2\rho}{n}\mathrm{Var}_{\widehat{P}_n}(\ell(\theta^\star, X))}}_{\stackrel{a.s.}{\to} 0},$$

so that eventually $R_n(\theta, \mathcal{P}_n) > R_n(\theta^\star, \mathcal{P}_n)$ for all $\theta \in \theta^\star + 2\epsilon\mathbb{B} \setminus \epsilon\mathbb{B}$. By convexity, then this inequality holds for all $\theta \notin \theta^\star + \epsilon\mathbb{B}$. Thus if $\widehat{\theta}_n^{\rm rob} \in \operatorname{argmin}_\theta R_n(\theta, \mathcal{P}_n)$, then for any $\epsilon > 0$ we must eventually have $\|\widehat{\theta}_n^{\rm rob} - \theta^\star\|_2 < \epsilon$.

Appendix H. Efficient solutions to computing the robust expectation

In this appendix, we give a detailed description of the procedure we use to compute the supremum problem (8). In particular, our procedure requires time $O(n \log n + \log \frac{1}{\epsilon} \log n)$, where ϵ is the desired solution accuracy. Let us reformulate this as a minimization problem in a variable $p \in \mathbb{R}^n$ for simplicity. Then we wish to solve

$$\text{minimize } p^\top z \quad \text{subject to } \frac{1}{2n}\left\|np - \mathbf{1}\right\|_2^2 \leq \rho,\ p \geq 0,\ p^\top \mathbf{1} = 1.$$

We take a partial dual of this minimization problem, then maximize this dual to find the optimizing p. Introducing the dual variable $\lambda \geq 0$ for the constraint that $\frac{1}{2}\|p - \frac{1}{n}\mathbf{1}\|_2^2 \leq \frac{\rho}{n}$ and performing the standard min-max swap (Boyd and Vandenberghe, 2004) (strong duality obtains for this problem because the Slater condition is satisfied by $p = \frac{1}{n}\mathbf{1}$) yields the maximization problem

$$\underset{\lambda \geq 0}{\text{maximize}}\ f(\lambda) := \inf_p \left\{\frac{\lambda}{2}\left\|p - \frac{1}{n}\mathbf{1}\right\|_2^2 - \frac{\lambda\rho}{n} + p^\top z \mid p \geq 0,\ \mathbf{1}^\top p = 1\right\}. \tag{43}$$

If we can efficiently compute the infimum (43), then it is possible to binary search over λ. Recall the standard fact (Hiriart-Urruty and Lemaréchal, 1993, Chapter VI.4.4) that for a collection $\{f_p\}_{p \in \mathcal{P}}$ of concave functions, if the infimum $f(x) = \inf_{p \in \mathcal{P}} f_p(x)$ is attained at some p_0 then any vector $\nabla f_{p_0}(x)$ is a supergradient of $f(x)$. Thus, letting $p(\lambda)$ be the (unique) minimizing value of p for any $\lambda > 0$, the objective (43) becomes $f(\lambda) = \frac{\lambda}{2}\|p(\lambda) - \frac{1}{n}\mathbf{1}\|_2^2 - \frac{\lambda\rho}{n} + p(\lambda)^\top z$, whose derivative with respect to λ (holding p fixed) is $f'(\lambda) = \frac{1}{2}\|p(\lambda) - \frac{1}{n}\mathbf{1}\|_2^2 - \frac{\rho}{n}$.

Now we use well-known results on the Euclidean projection of a vector to the probability simplex (Duchi et al., 2008) to provide an efficient computation of the infimum (43). First, we assume with no loss of generality that $z_1 \le z_2 \le \cdots \le z_n$ and that $\mathbf{1}^\top z = 0$, because neither of these changes the original optimization problem (as $\mathbf{1}^\top p = 0$ and the objective is symmetric). Then we define the two vectors $s, \sigma^2 \in \mathbb{R}^n$, which we use for book-keeping in the algorithm, by

$$s_i = \sum_{j \le i} z_j, \quad \sigma_i^2 = \sum_{j \le i} z_j^2,$$

and we let z^2 be the vector whose entries are z_i^2. The infimum problem (43) is equivalent to projecting the vector $v(\lambda) \in \mathbb{R}^n$ defined by

$$v_i = \frac{1}{n} - \frac{1}{\lambda} z_i$$

onto the probability simplex. Notably (Duchi et al., 2008), the projection $p(\lambda)$ has the form $p_i(\lambda) = (v_i - \eta)_+$ for some $\eta \in \mathbb{R}$, where η is chosen such that $\sum_{i=1}^n p_i(\lambda) = 1$. Finding such a value η is equivalent (Duchi et al., 2008, Figure 1) to finding the unique index i such that

$$\sum_{j=1}^{i} (v_j - v_i) < 1 \quad \text{and} \quad \sum_{j=1}^{i+1} (v_j - v_{i+1}) \ge 1,$$

taking $i = n$ if no such index exists (the sum $\sum_{j=1}^i (v_j - v_i)$ is increasing in i and $v_1 - v_1 = 0$). Given the index i, algebraic manipulations show that $\eta = \frac{1}{n} - \frac{1}{i} - \frac{1}{i} \sum_{j=1}^i z_j/\lambda = \frac{1}{n} - \frac{1}{i} - \frac{1}{i} s_i/\lambda$ satisfies the equality $\sum_{i=1}^n (v_i - \eta)_+ = 1$ and that $v_j - \eta \ge 0$ for all $j \le i$ while $v_j - \eta \le 0$ for $j > i$. Of course, given the index i and η, we may calculate the derivative $\frac{\partial}{\partial \lambda} f(\lambda)$ efficiently as well:

$$\begin{aligned} f'(\lambda) &= \frac{\partial}{\partial \lambda} \left\{ \frac{\lambda}{2} \left\| p(\lambda) - n^{-1}\mathbf{1} \right\|_2^2 - \frac{\lambda \rho}{n} + p(\lambda)^\top z \right\} \\ &= \frac{1}{2} \left\| p(\lambda) - n^{-1}\mathbf{1} \right\|_2^2 - \frac{\rho}{n} = \frac{1}{2} \sum_{j=1}^{i} (v_j - \eta - n^{-1})^2 + \frac{1}{2} \sum_{j=i+1}^{n} \frac{1}{n^2} - \frac{\rho}{n} \\ &= \frac{1}{2} \sum_{j=1}^{i} \left(\frac{1}{\lambda} z_j + \eta \right)^2 + \frac{n-i}{2n^2} - \frac{\rho}{n} = \frac{\sigma_i^2}{2\lambda^2} + \frac{i\eta^2}{2} + \frac{s_i \eta}{\lambda} + \frac{n-i}{2n^2} - \frac{\rho}{n}. \end{aligned}$$

Finding the index optimal i can be done by a binary search, which requires $O(\log n)$ time, and $f'(\lambda)$ is then computable in $O(1)$ time using the vectors s and σ^2. It is then possible to perform a binary search over λ using $f'(\lambda)$, which which requires $\log \frac{1}{\epsilon}$ iterations to find λ within accuracy ϵ, from which it is easy to compute $p(\lambda)$ via $p_i(\lambda) = (v_i - \eta)_+ = \left(n^{-1} - \lambda^{-1} z_i - \eta\right)_+$.

We summarize this discussion with pseudo-code in Figures 6 and 7, which provide a main routine and sub-routine for finding the optimal vector p. These routines show that, once provided the sorted vector z with $z_1 \le z_2 \le \cdots \le z_n$ (which requires $n \log n$ time to compute), we require only $O(\log \frac{1}{\epsilon} \cdot \log n)$ computations.

Inputs: Sorted vector $z \in \mathbb{R}^n$ with $\mathbf{1}^\top z = 0$, parameter $\rho > 0$, solution accuracy ϵ

Set $\lambda_{\min} = 0$ and $\lambda_{\max} = \lambda_\infty = \max\{n \|z\|_\infty, \sqrt{n/2\rho}\, \|z\|_2\}$
Set $s_i = \sum_{j \le i} z_j$ and $\sigma_i^2 = \sum_{j \le i} z_j^2$
While $|\lambda_{\max} - \lambda_{\min}| > \epsilon \lambda_\infty$
  Set $\lambda = \frac{\lambda_{\max} + \lambda_{\min}}{2}$
  Set $(\eta, i) = \textsc{FindShift}(z, \lambda, s)$ // (Figure 7)
  Set $f'(\lambda) = \frac{1}{2\lambda^2}\sigma_i^2 + \frac{\eta^2}{2} i^2 + \frac{\eta}{\lambda} s_i + \frac{n-i}{2n^2} - \frac{\rho}{n}$
  If $f'(\lambda) > 0$
    Set $\lambda_{\min} = \lambda$
  Else
    Set $\lambda_{\max} = \lambda$
Set $\lambda = \frac{1}{2}(\lambda_{\max} + \lambda_{\min})$, $(\eta, i) = \textsc{FindShift}(z, \lambda, s)$
Set $p_i = \left(\frac{1}{n} - \frac{1}{\lambda} z_i - \eta\right)_+$ and return p

Figure 6. Procedure FindP to find the vector p minimizing $\sum_{i=1}^n p_i z_i$ subject to the constraint $\frac{1}{2n} \|np - \mathbf{1}\|_2^2 \le \rho$. Method takes $\log \frac{1}{\epsilon}$ iterations of the loop.

Inputs: Sorted vector z with $\mathbf{1}^\top z = 0$, $\lambda > 0$, vector s with $s_i = \sum_{j \le i} z_j$

Set $i_{\text{low}} = 1, i_{\text{high}} = n$
If $\frac{1}{n} - \frac{z_n}{\lambda} \ge 0$
  Return $(\eta = 0, i = n)$
While $i_{\text{low}} \ne i_{\text{high}}$
  $i = \frac{1}{2}(i_{\text{low}} + i_{\text{high}})$
  $s_{\text{left}} = \frac{1}{\lambda}(iz_i - s_i)$ // (this is $s_{\text{left}} = \sum_{j=1}^i (v_j - v_i)$)
  $s_{\text{right}} = \frac{1}{\lambda}((i+1)z_{i+1} - s_{i+1})$ // (this is $s_{\text{right}} = \sum_{j=1}^{i+1} (v_j - v_{i+1})$)
  If $s_{\text{right}} \ge 1$ and $s_{\text{left}} < 1$
    Set $\eta = \frac{1}{n} - \frac{1}{i} - \frac{1}{\lambda i} s_i$ and return (η, i)
  Else if $s_{\text{left}} \ge 1$
    Set $i_{\text{high}} = i - 1$
  Else
    Set $i_{\text{low}} = i + 1$
Set $i = i_{\text{low}}$ and $\eta = \frac{1}{n} - \frac{1}{i} - \frac{1}{\lambda i} s_i$ and return (η, i)

Figure 7. Procedure FindShift to find index i and parameter η such that, for the definition $v_i = \frac{1}{n} - \frac{1}{\lambda} z_i$, we have $v_j - \eta \ge 0$ for $j \le i$, $v_j - \eta \le 0$ for $j > i$, and $\sum_{j=1}^n (v_j - \eta)_+ = 1$. Method requires time $O(\log n)$.

References

M. Anthony and J. Shawe-Taylor. A result of Vapnik with applications. *Discrete Applied Mathematics*, 47:207–217, 1993.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

P. L. Bartlett, O. Bousquet, and S. Mendelson. Local Rademacher complexities. *Annals of Statistics*, 33(4):1497–1537, 2005.

P. L. Bartlett, M. I. Jordan, and J. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101:138–156, 2006.

A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

A. Ben-Tal, D. den Hertog, A. D. Waegenaere, B. Melenberg, and G. Rennen. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357, 2013.

A. Ben-Tal, E. Hazan, T. Koren, and S. Mannor. Oracle-based robust optimization via online learning. *Operations Research*, 63(3):628–638, 2015.

D. Bertsimas, V. Gupta, and N. Kallus. Robust SAA. *arXiv:1408.4445 [math.OC]*, 2014. URL `http://arxiv.org/abs/1408.4445`.

M. Birman and M. Solomjak. Piecewise-polynomial approximations of functions of the classes W_p^α. *Sbornik: Mathematics*, 2(3):295–317, 1967.

S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: a survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 2005.

S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities: a Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

O. Bousquet. A bennett concentration inequality and its application to suprema of empirical processes. *Comptes Rendus Mathematique*, 334(6):495–500, 2002a.

O. Bousquet. *Concentration inequalities and empirical processes theory applied to the analysis of learning algorithms*. PhD thesis, L'Ecole Polytechnique, 2002b.

O. Bousquet. Concentration inequalities for sub-additive functions using the entropy method. In *Stochastic inequalities and applications*, pages 213–247. Springer, 2003.

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

P. Bühlmann and S. van de Geer. *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer, 2011.

S. Chatterjee. An error bound in the Sudakov-Fernique inequality. *arXiv:0510424 [math.PR]*, 2005.

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

J. C. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, 2008.

J. C. Duchi, P. W. Glynn, and H. Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *arXiv:1610.03425 [stat.ML]*, 2016.

R. M. Dudley. *Uniform Central Limit Theorems*. Cambridge University Press, 1999.

J.-y. Gotoh, M. J. Kim, and A. Lim. Robust empirical optimization is almost the same as mean-variance optimization. *Available at SSRN 2827400*, 2015.

C. Gu. *Smoothing spline ANOVA models*. Springer, 2002.

J. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms I & II*. Springer, New York, 1993.

V. Koltchinskii. Local Rademacher complexities and oracle inequalities in risk minimization. *Annals of Statistics*, 34(6):2593–2656, 2006.

T. Kühn. Covering numbers of Gaussian reproducing kernel Hilbert spaces. *Journal of Complexity*, 27(5):489–499, 2011.

H. Lam and E. Zhou. Quantifying input uncertainty in stochastic optimization. In *Proceedings of the 2015 Winter Simulation Conference*. IEEE, 2015.

M. Ledoux and M. Talagrand. *Probability in Banach Spaces*. Springer, 1991.

E. L. Lehmann and G. Casella. *Theory of Point Estimation, Second Edition*. Springer, 1998.

D. Lewis, Y. Yang, T. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

E. Mammen and A. B. Tsybakov. Smooth discrimination analysis. *Annals of Statistics*, 27: 1808–1829, 1999.

A. Maurer and M. Pontil. Empirical Bernstein bounds and sample variance penalization. In *Proceedings of the Twenty Second Annual Conference on Computational Learning Theory*, 2009.

S. Mendelson. On the performance of kernel classes. *Journal of Machine Learning Research*, 4(Oct):759–771, 2003.

S. Mendelson. Learning without concentration. In *Proceedings of the Twenty Seventh Annual Conference on Computational Learning Theory*, 2014.

H. Namkoong and J. C. Duchi. Stochastic gradient methods for distributionally robust optimization with f-divergences. In *Advances in Neural Information Processing Systems 30*, 2016.

A. Owen. Empirical likelihood ratio confidence regions. *The Annals of Statistics*, 18(1): 90–120, 1990.

A. B. Owen. *Empirical likelihood*. CRC press, 2001.

A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory*. SIAM and Mathematical Programming Society, 2009.

P. K. Shivaswamy and T. Jebara. Empirical Bernstein boosting. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010.

P. K. Shivaswamy and T. Jebara. Variance penalizing AdaBoost. In *Advances in Neural Information Processing Systems 25*, 2011.

N. Srebro, K. Sridharan, and A. Tewari. Smoothness, low noise and fast rates. In *nips2010*, pages 2199–2207, 2010.

A. B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics*, 32(1):135–166, 2004.

A. B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009.

S. van de Geer. *Empirical Processes in M-Estimation*. Cambridge University Press, 2000.

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes: With Applications to Statistics*. Springer, New York, 1996.

V. N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, XVI(2):264–280, 1971.

V. N. Vapnik and A. Y. Chervonenkis. *Theory of Pattern Recognition*. Nauka, Moscow, 1974. (In Russian).

D.-X. Zhou. Capacity of reproducing kernel spaces in learning theory. *IEEE Transactions on Information Theory*, 49(7):1743–1752, 2003.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67(2):301–320, 2005.

A. Zubkov and A. Serov. A complete proof of universal inequalities for the distribution function of the binomial law. *Theory of Probability & Its Applications*, 57(3):539–544, 2013.

Journal of Machine Learning Research 20 (2019) 1-39 Submitted 1/18; Revised 12/18; Published 4/19

On Consistent Vertex Nomination Schemes

Vince Lyzinski VLYZINSKI@UMASS.EDU
Department of Mathematics and Statistics
University of Massachusetts Amherst
Amherst, MA 01003, USA

Keith Levin KLEVIN@UMICH.EDU
Department of Statistics
University of Michigan
Ann Arbor, MI 48109, USA

Carey E. Priebe CEP@JHU.EDU
Department of Applied Mathematics and Statistics
Johns Hopkins University
Baltimore, MD 21218-2608, USA

Editor: Edo Airoldi

Abstract

Given a vertex of interest in a network G_1, the vertex nomination problem seeks to find the corresponding vertex of interest (if it exists) in a second network G_2. A vertex nomination scheme produces a list of the vertices in G_2, ranked according to how likely they are judged to be the corresponding vertex of interest in G_2. The vertex nomination problem and related information retrieval tasks have attracted much attention in the machine learning literature, with numerous applications to social and biological networks. However, the current framework has often been confined to a comparatively small class of network models, and the concept of statistically consistent vertex nomination schemes has been only shallowly explored. In this paper, we extend the vertex nomination problem to a very general statistical model of graphs. Further, drawing inspiration from the long-established classification framework in the pattern recognition literature, we provide definitions for the key notions of Bayes optimality and consistency in our extended vertex nomination framework, including a derivation of the Bayes optimal vertex nomination scheme. In addition, we prove that no universally consistent vertex nomination schemes exist. Illustrative examples are provided throughout.

Keywords: Vertex nomination, graph inference, recommender systems

1. Introduction

Statistical inference on graphs is an important branch of modern statistics and machine learning. In recent years, there have been numerous papers in the literature developing graph analogues of statistical inference tasks such as hypothesis testing (Asta and Shalizi, 2015; Tang et al., 2017b), classification (Tang et al., 2013; Chen et al., 2016a), and clustering (Luxburg, 2007; Rohe et al., 2011; Sussman et al., 2012; Newman and Clauset, 2016). Moreover, growth in the size and complexity of network data sets have necessitated techniques for network-specific data mining tasks such as link prediction (Liben-Nowell and Kleinberg, 2007; Lü and Zhou, 2011); entity resolution and network alignment (Conte et al., 2004; Lyzinski, 2018); and vertex nomination (Coppersmith and Priebe, 2012; Coppersmith, 2014; Suwan et al., 2015; Fishkind et al., 2015; Lyzinski et al., 2016). Akin to the development of classical statistics, algorithmic advancement has, in many ways, outpaced theoretical developments in these emerging graph-driven domains. This development has been necessitated by the dizzying pace of data generation, but there is nevertheless the need for a firm theoretical context in which to frame algorithmic progress. In this paper, drawing inspiration from the long-established classification framework in the pattern recognition literature (Devroye et al., 1997), we provide a rigorous theoretical framework for understanding statistical consistency in the vertex nomination inference task.

©2019 Vince Lyzinski, Keith Levin, and Carey E. Priebe.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-048.html.

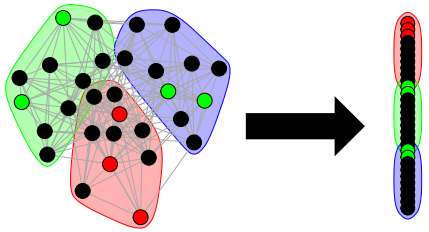


Figure 1: A visual representation of the classical Vertex Nomination framework: Given a community of interest in a network (here the red community) and some examples of vertices that are/are not part of the community of interest (colored red and green, respectively), rank the remaining vertices in the network into a nomination list, with those vertices from the community of interest concentrating at the top of the nomination list.

The vertex nomination (VN) task, which can be viewed as the graph analogue of the more classical recommender system task (Ricci et al., 2011), has traditionally been stated as follows: given a community of interest in a network and some examples of vertices that are or are not part of a community of interest, vertex nomination seeks to rank the remaining vertices in the network into a nomination list, with those vertices from the community of interest (ideally) concentrating at the top of the nomination list. See Figure 1 for a visual representation of this classical Vertex Nomination framework. In limited-resource settings, vertex nomination tools have proven to be effective in efficiently searching and querying large networks, with applications including detecting fraudsters in the Enron email network (Coppersmith and Priebe, 2012; Marchette et al., 2011; Suwan et al., 2015), uncovering web advertisements that have association with human trafficking (Fishkind et al., 2015), and identifying latent structure in connectome data (Fishkind et al., 2015; Yoder et al., 2018).

While related to the community detection problem (Newman, 2006; Luxburg, 2007; Bickel and Chen, 2009; Newman and Clauset, 2016), this traditional formulation of the VN problem is a semi-supervised inference task whose output is not an assignment of vertices to communities, but rather a ranked estimate of which vertices belong to a particular community of interest. That is, in contrast to community detection, the VN problem does

not aim to recover the community memberships of any vertices not in the community of interest. Clearly, any method that can recover the community memberships of all vertices in a graph can recover the interesting community, and hence any community detection algorithm can be repurposed for the VN problem just described with minor adaptation (e.g., by ranking vertices according to their probability of membership in the community of interest); see, for example, the spectral vertex nomination scheme of Fishkind et al. (2015). The specific performance of such an adaptation is highly dependent on the fidelity of the base clustering procedure, and the performance is often below that of the semi-supervised VN specific analogues (see Yoder et al., 2018).

The above formulation of the VN task assumes the presence of strong community structure among the vertices of interest in the graph. In practice, this is often a reasonable assumption, particularly if it is expected that interesting vertices will behave similarly to one another in the network. However, the particular features that mark a vertex as interesting are entirely task-dependent. To paraphrase the common proverb, interestingness is in the eye of the practitioner. Interesting vertices may be, for example, those with large network centrality (Jeong et al., 2001; Newman, 2005), those with a particular role in the network (Lusseau and Newman, 2004), or those corresponding to a given user across social networks (Patsolic et al., 2017). In these applications, interesting vertices need not correspond precisely to the community structure captured by a generative network model, and hence such cases are ill-described by the community-based VN problem described above. To accommodate this task-dependency and broader notion of interesting vertices, we consider the following generalization and extension of the previously-presented VN problem: Given a vertex of interest v^* in a graph $G_1 = (V_1, E_1)$, find the corresponding vertex of interest u^* (if it exists) in a second graph $G_2 = (V_2, E_2)$ by ranking the vertices of G_2 according to our confidence that they correspond to v^* in graph G_1; see Figure 2 for a visual representation of this VN framework. In this formulation, which is an (potentially) unsupervised inference task, what defines v^* as interesting is entirely model-dependent, and different network models can highlight different characteristics of interest in the graph. Potential application domains for this VN generalization abound, including identifying users of interest across social network platforms (see, for example, Patsolic et al., 2017), identifying structural signal across connectomes (see, for example, Sussman et al., 2018), and identifying topics of interest across graphical knowledge bases (see, for example, Sun and Priebe, 2013).

In Fishkind et al. (2015) and Lyzinski et al. (2016), the notion of a consistent vertex nomination scheme (i.e., an asymptotically optimal solution to the VN problem) was proposed for the original formulation of the VN problem, in which community membership entirely determines whether or not a given vertex is interesting. This definition of consistency was based on the *mean average precision* (MAP) of a nomination scheme operating on a graph model with explicit community structure encoded by the the Stochastic Block Model (SBM) of Holland et al. (1983). Under this restricted notion of consistency, Fishkind et al. (2015) derived the analogue of universal Bayes optimality in the VN setting, namely a scheme that achieves the optimal mean average precision for all parameterizations of the underlying SBM. While this derivation of the Bayes optimal scheme somewhat parallels the derivation of the Bayes optimal classifier in the classical pattern recognition literature, the SBM model assumption and MAP formulation greatly narrow the set of models and sets of interesting vertices we can consider. In this paper, we revamp and generalize the concept

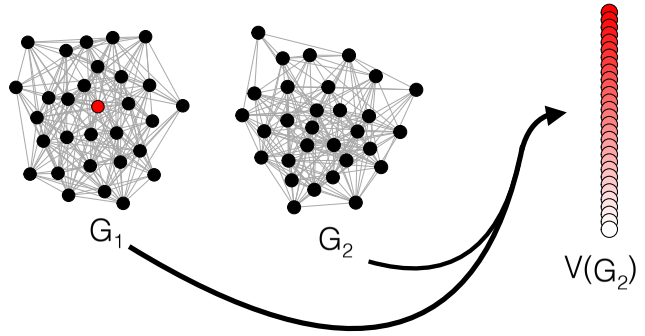


Figure 2: A visual representation of the generalized Vertex Nomination framework: Given a vertex of interest v^* (colored red) in a graph $G_1 = (V_1, E_1)$, find the corresponding vertex of interest u^* (if it exists) in a second graph $G_2 = (V_2, E_2)$, ranking the vertices of G_2 into a nomination list so that u^* ideally appears at the top of the nomination list.

of VN consistency—and of VN Bayes optimality—in the two-graph VN framework. This framework is quite general, and further allows us to highlight the similarities and differences between our new VN consistency formulation and its analogue in the classification literature defined in, for example, Devroye et al. (1997).

The paper is laid out as follows. In the remainder of this section, we provide brief overviews of information retrieval as it relates to vertex nomination (Section 1.1) and the Bayes optimal classifier in the classical setting (Section 1.2), and conclude the introduction by establishing notation for the remainder of the paper (Section 1.3). In Section 2, we define the VN problem framework that is the focus of this paper, and in Section 3 we derive the VN analogue of a Bayes optimal scheme. In Section 4, we define a new notion of VN consistency, and we prove that no universally consistent VN scheme exists, providing an interesting contrast to the standard classification setting. We conclude in Section 5 with a short summary comparing and contrasting VN with classical classification and a discussion of implications and future directions.

1.1. Connections to Information Retrieval

The vertex nomination task is, in some ways, similar to the task faced by recommender systems (Resnick and Varian, 1997; Ricci et al., 2011), in which the aim is to retrieve objects (e.g., documents or images) likely to be of interest to a user based on his or her previous behavior. For example, the celebrated PageRank algorithm (Brin and Page, 1998) recommends webpages based on random walks on the world wide web graph, in which websites are nodes and (directed) edges reflect hyperlinks between pages. The information retrieval (IR) literature includes many such graph-based approaches. We refer the reader to

Ricci et al. (2011) and Mihalcea and Radev (2011) for the state of the art circa 2010, and concentrate here on recapping more recent graph-based information retrieval techniques.

Many graph-based IR techniques rely on the assumption that similar objects (i.e., documents, webpages, etc.) will lie near one another in a suitably-constructed graph. Indeed, this intuition underlies many graph-based approaches throughout machine learning and related disciplines (see, for example, Belkin and Niyogi, 2003; Zhou et al., 2004). Techniques along these lines have been applied toward many tasks in natural language processing, typically inspired by PageRank (Rothe and Schütze, 2014). Along similar lines, Ma et al. (2012) applied a diffusion-based method (Coifman and Lafon, 2006) to the world wide web graph to yield an approach to ranking for query completion and recommendation. These information retrieval techniques can be naturally adapted to the vertex nomination problem by treating the vertex or vertices of interest as the object or objects to be retrieved.

The vertex nomination problem also bears similarities to the task of *learning to rank* (Duh, 2009; Liu, 2009; Li, 2011), in which the goal is to learn an ordering on a set of objects (e.g., documents, images, videos, etc.) according to (estimated) similarity or relevance to a given query object. In the learning to rank literature, graphs usually appear as training instances, with nodes corresponding to objects and edges encoding preferences or similarities among them elicited from users (e.g., an undirected weighted edge may join two documents judged to be similar). The work in Agarwal et al. (2006) is among the earliest to consider the problem of ranking objects in a network. The authors modified the PageRank algorithm to take preference information into account, rather than working solely with the hyperlink graph. In Agarwal (2010), the authors used a *data graph* encoding object similarities to obtain a regularizer similar to Belkin et al. (2006) on the empirical ranking error, with the target ranking encoded in a *preference graph*. More recent efforts along these lines have focused on the problem of incorporating network structure present between entities of different types, for example, between users and events in a social network (Luo et al., 2014; Pham et al., 2016). Here again, any learning to rank algorithm has a natural adaptation to the VN problem by using the first graph, in which some vertices are labeled, as training data to learn a ranking on the vertices of the second graph.

1.2. Bayes Error in Classical Pattern Recognition

In this section, we review the concepts of consistency and Bayes error from the statistical classification literature. We do not aim to give an exhaustive overview of the subject, but only to provide a rough outline as to the structures that we would like to replicate in the context of vertex nomination. For a more thorough treatment, we refer the interested reader to Devroye et al. (1997), whose presentation we follow below.

We begin by recalling the classical definition of Bayes error. Note that we will restrict our attention to the two-class problem to maximally bring forth the similarities (and differences) between statistical classification and VN, as in VN vertices are either of interest or not.

Definition 1 *Consider a set of potential observations $\mathcal{X}$ and a set of unknown class labels $\{0,1\}$ for objects in $\mathcal{X}$. A classifier is a function $h : \mathcal{X} \to \{0,1\}$, which aims to predict the class label of a given observation in $\mathcal{X}$. Given a distribution F supported on $\mathcal{X} \times \{0,1\}$, the error for the classifier h is given by $L(h) = \mathbb{P}(h(X) \neq Y)$ where $(X,Y) \sim F$.*

Any classifier that achieves the lowest possible error is said to be a *Bayes optimal* classifier. We write h^* for any such optimal classifier, which by definition satisfies $h^* \in \arg\min_{h:\mathcal{X}\to\{0,1\}} L(h)$. It is easily seen in this two-class framework that the Bayes optimal classifier is given by

$$h^*(x) = \begin{cases} 1 & \text{if } \mathbb{E}(Y|X=x) = \mathbb{P}(Y=1|X=x) > 1/2; \\ 0 & \text{else.} \end{cases} \tag{1}$$

Practically speaking, the Bayes optimal scheme chooses the label which maximizes the class-conditional probability of the observed data. The corresponding error, $L^* = L(h^*)$, is called the *Bayes error*. Of course, h^* depends on the distribution F of (X,Y), and, when appropriate, we will make this dependence explicit by writing L^*_F.

In practice, a classifier is often constructed based on training data $(X_1,Y_1),(X_2,Y_2),\ldots,(X_n,Y_n)$, where the data (X_i,Y_i) are drawn i.i.d. according to F. This supervised classification framework is defined as follows.

Definition 2 *Consider a set of potential observations $\mathcal{X}$ and a set of unknown class labels $\{0,1\}$ for objects in $\mathcal{X}$. A (supervised) classifier is a function*

$$h_n : \mathcal{X} \times \{\mathcal{X} \times \{0,1\}\}^n \to \{0,1\},$$

which aims to predict the class label of a given observation in $\mathcal{X}$ based on n training observations $(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n) \in \mathcal{X} \times \{0,1\}$. Given a distribution F supported on $\mathcal{X} \times \{0,1\}$, the error for the classifier h_n is given by

$$L_F(h_n) = \mathbb{P}\big[h_n(X,(X_i,Y_i)_{i=1}^n) \neq Y \mid (X_i,Y_i)_{i=1}^n\big]$$

where $(X,Y),(X_1,Y_1),(X_2,Y_2),\ldots,(X_n,Y_n) \overset{i.i.d.}{\sim} F$.

Note that $L_F(h_n)$ is a random variable in which $\{(X_i,Y_i)\}_{i=1}^n$ are drawn i.i.d. from F, but then held fixed as we average over $(X,Y) \sim F$.

A sequence of classifiers $\mathbf{h} = (h_n)_{n=1}^\infty$ is called a *classification rule*. Informally, a good classification rule is one for which the probability of error becomes arbitrarily close to Bayes optimal as $n \to \infty$. The precise nature of what we mean by close is codified in the concept of statistical consistency.

Definition 3 *A classification rule $\mathbf{h} = (h_n)_{n=1}^\infty$ is* consistent *with respect to F if*

$$\mathbb{E}_F(L(h_n)) \to L^*_F.$$

The rule $\mathbf{h}$ is strongly consistent *if*

$$L_F(h_n) \overset{a.s.}{\to} L^*_F.$$

A rule that is (strongly) consistent for all distributions F on $\mathcal{X} \times \{0,1\}$ is called (strongly) universally consistent.

Perhaps surprisingly, given that F can have arbitrary structure on $\mathcal{X} \times \{0,1\}$, universally consistent classification rules exist; see Stone (1977) for the first proof of this phenomenon.

In Fishkind et al. (2015), a notion of consistency for vertex nomination was presented, roughly analogous to Definition 3. In contrast to the classification task presented above, vertex nomination requires a ranking of the vertices, rather than merely the classification of a single vertex. As such, a vertex nomination scheme is evaluated in Fishkind et al. (2015) based on *average precision* (Manning et al., 2008), rather than simply a fraction of correctly-classified vertices. In Fishkind et al. (2015), VN consistency is defined in the context of stochastic block model (SBM) random graphs with respect to a provably optimal *canonical* nomination scheme. This canonical scheme plays an analogous role of Bayes optimal classifiers in this restricted model framework (see Section 3 below). The goal of this paper is to explore and further develop a broader notion of VN consistency that encompasses a more expressive class of models than the SBM.

1.3. Notation and Background

We conclude this section by establishing notation and reviewing a few of the more popular statistical network models that we will make use of as examples in the sequel.

1.3.1. Notation

For a set S, we let $|S|$ denote its cardinality and $\binom{S}{2}$ denote the set of all unordered pairs of distinct elements from S. Throughout, we will denote graphs via the ordered pair $G = (V, E)$, with vertices V and edges $E \subseteq \binom{V}{2}$. All graphs considered herein will be labeled, hollow (i.e., containing no self-edges), and undirected. We let $\mathcal{G}_n$ denote the set of all labeled, hollow, undirected graphs on n vertices. Given a graph G, we will let $V(G)$ denote the vertices of G and $E(G)$ denote its edges. We note that when G is random, this latter set is a random subset of $\binom{V}{2}$. For a set of vertices $S \subseteq V(G)$, we let $G[S]$ denote the subgraph of G induced by S, i.e., the graph $G' = (S, E)$ with $\{u, v\} \in E$ if and only if $\{u, v\} \in E(G)$. In a few places, we will require the notion of an *asymmetric* graph. A graph $G \in \mathcal{G}_n$ is *asymmetric* if it has no nontrivial automorphisms (Erdős and Rényi, 1963). For a positive integer $n \in \mathbb{Z}$, we will define $[n] = \{1, 2, \ldots, n\}$, and $\mathcal{G}_n$ to be the be the set of labeled graphs on n vertices. Throughout this paper, we will often, in order to simplify notation, suppress dependence of parameters on n. Throughout, the reader should assume that, unless specified otherwise, all parameters depend on the number of vertices n.

1.3.2. Stochastic block models

The stochastic block model (SBM) is a widely studied model for edge-independent random graphs with latent community structure (Holland et al., 1983; Hoff et al., 2002; Karrer and Newman, 2011).

Definition 4 *We say that a random graph $G = (V, E) \in \mathcal{G}_n$ is an instantiation of a stochastic block model with parameters (K, B, b), written $G \sim \mathrm{SBM}(K, B, b)$, if*

i. V is partitioned into K classes (called communities or blocks), $V = V_1 \cup V_2 \cup \cdots \cup V_K$.

ii. The block membership vector $b \in [K]^{|V|}$ is such that for all $k \in [K]$, $b_v = k$ if and only if $v \in V_k$.

iii. The symmetric matrix $B \in [0,1]^{K \times K}$ denotes the edge probabilities between and within blocks, with

$$\mathbb{1}_{\{\{u,v\} \in E(G)\}} \overset{ind.}{\sim} Bernoulli(B_{b_u, b_v}).$$

We note that when $K = 1$, we recover the Erdős-Rényi random graph (Erdős and Rényi, 1959), in which the edges of G are present or absent independently with probability p. In this special case, we write $G \sim \mathrm{ER}(n, p)$. By a slight abuse of notation, for a symmetric matrix $P \in [0,1]^{n \times n}$, we will write $G \sim \mathrm{ER}(P)$ if, identifying the vertices of G with $[n]$, we have $\{i, j\} \in E(G)$ with probability $P_{i,j}$ independent of the other edges. With no restrictions on P, $\mathrm{ER}(P)$ random graphs can be viewed as n-block SBMs and are the most general edge-independent random graph model.

The latent community structure inherent to SBMs makes them a natural model for use in the traditional vertex nomination framework. Recall the traditional VN task: given a community of interest in a network and some examples of vertices that are or are not part of the community of interest, vertex nomination seeks to rank the remaining vertices in the network into a nomination list, with those vertices from the community of interest (ideally) concentrating at the top of the nomination list. As a result, previous work on VN consistency (Fishkind et al., 2015) has been posed within the SBM framework, with the optimal scheme only obtaining its optimality for SBMs. We note that we consider herein the SBM setting where communities are disjoint and each vertex can only belong to a single community. However, the results contained herein translate immediately to the mixed membership SBM setting (Airoldi et al., 2008); details are omitted for brevity.

1.3.3. Random dot product graphs

In stochastic block models, the block assignment vector can be viewed as a latent feature vector for the vertices in the network, with these features (i.e., block memberships) defining the connectivity structure in the network. The random dot product graph (RDPG) model (Young and Scheinerman, 2007) allows for more nuanced vertex features to be incorporated into the model and has been used as the setting for a VN formulation similar to the one proposed here; see Patsolic et al. (2017) for details.

Definition 5 *We say that a random graph $G = (V, E) \in \mathcal{G}_n$ is an instantiation of a d-dimensional random dot product graph with parameters X, written $G \sim \mathrm{RDPG}(X)$, if*

i. The matrix $X \in \mathbb{R}^{n \times d}$ is such that $0 \leq (XX^T)_{i,j} \leq 1$ for all $i, j \in [n]$. The rows of X provide the latent features for the vertices in V.

ii. The edges of G are present or absent independently, with $\{i, j\} \in E(G)$ with probability $(XX^T)_{i,j}$. Written succinctly, $G \sim \mathrm{ER}(XX^T)$.

We can view the RDPG model as a example of the more general *latent position* random graph model (Hoff et al., 2002), in which edge probabilities are determined by hidden vertex-level geometry.

Estimating the latent position structure in RDPGs is particularly amenable to spectral methods, and this model has played a prominent role in recent theoretical developments of spectral graph methods (see, for example, Rohe et al., 2011; Sussman et al., 2012; Tang

et al., 2017a). Note that the RDPG can be extended to a broader class of models, in which edge probabilities are given by evaluating a positive definite link function at vertices' latent positions as in, for example, Tang et al. (2013). While incorporating this more general family of latent position graphs into the present VN framework would be straightforward, we restrict our focus to the RDPG model of Definition 5 for ease of exposition.

1.3.4. Correlation across networks

The vertex nomination problem we consider in this paper presupposes the existence of a vertex of interest in a network G_1 and, ideally, a corresponding vertex of interest in a second network G_2. Often, such correspondences across networks are encoded into random graph models via edge-wise graph correlation (see, for example, Fishkind et al., 2019). Arguably the simplest such structure is seen in the ρ-correlated Erdős-Rényi model (see, for example, Lyzinski et al., 2015).

Definition 6 *We say that bivariate random graphs $(G_1, G_2) \in \mathcal{G}_n \times \mathcal{G}_n$ are an instantiation of a ρ-correlated $\mathrm{ER}(P)$ model, written $(G_1, G_2) \sim \rho\text{-}\mathrm{ER}(P)$, if*

i. Marginally, $G_1 \sim \mathrm{ER}(P)$ and $G_2 \sim \mathrm{ER}(P)$.

ii. Edges are independent across G_1 and G_2 except that the indicators of the events $\{u, v\} \in E(G_1)$ and $\{u, v\} \in E(G_2)$ are jointly distributed as a pair of Bernoulli random variables with success probability $P_{u,v}$ and correlation ρ. If the correlation is allowed to vary across edges, so that these two events have correlation $\rho_{u,v}$, then collecting these correlations in a symmetric matrix $R = [\rho_{i,j}]_{i,j=1}^{n}$, we write $(G_1, G_2) \sim R\text{-}\mathrm{ER}(P)$ (see Lyzinski and Sussman, 2018).

Ranging the values in R from 0 to 1 allows for the consideration of graphs that range from independent ($R = 0$) to isomorphic ($R = 1$). Intermediate values of R allow for the encoding of a correspondence across networks between these two extremes. We will also consider $R < 0$, in which case edges across networks are anti-correlated. This is particularly useful for modeling situations in which corresponding vertices stochastically behave differently across networks.

2. Vertex Nomination

Loosely stated, the vertex nomination problem we consider in this paper can be summarized as follows: Given a vertex of interest v^* in a graph $G_1 = (V_1, E_1)$, find the corresponding vertex of interest u^* (if it exists) in a second graph $G_2 = (V_2, E_2)$ by ranking the vertices of G_2 according to our confidence that they correspond to v^* in graph G_1. To formally define this version of vertex nomination, we will need to consider distributions on graphs with partially-overlapping node sets that have a built-in notion of vertex correspondence across graphs. To this end, we will consider distributions on $\mathcal{G}_n \times \mathcal{G}_m$, where $\mathcal{G}_n$ is the set of labeled graphs on n vertices, with vertex labels given by $\{v_1, v_2, \ldots, v_n\}$, and $\mathcal{G}_m$ is the set of labeled graphs on m vertices, with vertex labels given by $\{u_1, u_2, \ldots, u_m\}$. Note that for $i \in [n] \cap [m]$, v_i and u_i are merely vertex labels, and it is not necessarily the case that $v_i = u_i$. We follow this labeling convention in order to emphasize the reality that the vertex sets of G_1 and G_2 may only partially overlap.

Definition 7 (Nominatable Distributions) *For a given $n, m \in \mathbb{Z} > 0$, the set of* Nominatable Distributions of order (n, m), *denoted $\mathcal{N}_{n,m}$, is the collection of all families of distributions of the following form*

$$\mathbf{F}_{\Theta}^{(n,m)} = \{F_{c,\theta}^{(n,m)} \quad s.t.\ 0 \leq c \leq \min(n, m) \in \mathbb{Z}, \theta \in \Theta \subset \mathbb{R}^{d(n,m)}\}$$

where $F_{c,\theta}^{(n,m)}$ is a distribution on $\mathcal{G}_n \times \mathcal{G}_m$ parameterized by $\theta \in \Theta$ satisfying:

i. The vertex sets $V_1 = \{v_1, v_2, \ldots, v_n\}$ and $V_2 = \{u_1, u_2, \ldots, u_m\}$ satisfy $v_i = u_i$ for $0 < i \leq c$. We refer to $C = \{v_1, v_2, \ldots, v_c\} = \{u_1, u_2, \ldots, u_c\}$ as the core vertices. *These are the vertices that are shared across the two graphs and imbue the model with a natural notion of corresponding vertices.*

ii. Vertices in $J_1 = V_1 \setminus C$ and $J_2 = V_2 \setminus C$, satisfy $J_1 \cap J_2 = \emptyset$. We refer to J_1 and J_2 as junk *vertices. These are the vertices in each graph that have no corresponding vertex in the other graph.*

iv. The induced subgraphs $G_1[J_1]$ and $G_2[J_2]$ are conditionally independent given θ.

A few examples will serve to illustrate this definition. We will return to the three example settings below several times throughout the rest of the paper in order to highlight and illustrate phenomena of interest.

Example 1 (R-ER(P)) Let $(G_1, G_2) \sim R\text{-}\mathrm{ER}(P)$ with $P, R \in \mathbb{R}^{n \times n}$ and $R > 0$ entrywise, so that G_1 and G_2 have correlated edges as described in Section 1.3.4. In this example, the model parameter is $\theta = (P, R)$, and the vertex sets of the two graphs can be thought of as fully overlapping, i.e., $V_1 = V_2 = C = [n]$ and $J_1 = J_2 = \emptyset$, since the correlation structure conveyed in the entries of R encodes an explicit correspondence between the edges of G_1 and the edges of G_2 (and hence also a correspondence between V_1 and V_2). Note that if we consider $C = [k]$ with $k < n$, then we would require (after suitably ordering the vertices) $R_{u,v} = 0$ for $u, v > k$. This highlights the way in which θ (and hence the distribution $F_{c,\theta}^{(n,m)}$) can vary with c, and vice-versa.

Example 2 (RDPG) Let $m > n$ and suppose that $Y \in \mathbb{R}^{m \times d}$ has distinct rows and satisfies $(YY^T)_{i,j} \in [0, 1]$ for all $i, j \in [m]$. Let $X \in \mathbb{R}^{n \times d}$ be a submatrix of Y, and consider $G_1 \sim \mathrm{RDPG}(X)$ and $G_2 \sim \mathrm{RDPG}(Y)$, where G_1 and G_2 are conditionally independent given Y. In this example, we can consider $\theta = Y$, $V_1 = [n] = C$, $J_1 = \emptyset$, $V_2 = [n] \cup J_2$, and $J_2 = \{u_{n+1}, u_{n+2}, \ldots, u_m\}$. Note that as G_1 and G_2 are conditionally independent given θ, we could also consider $0 < c < n$ here as well. This illustrates that θ need not necessarily vary with c, and hence $F_{c,\theta}^{(n,m)}$ need not vary with c, either.

Example 3 (Independent Erdős-Rényi graphs) Let (G_1, G_2) be independent $\mathrm{ER}(n, p)$ random graphs. In this example, we can consider any $c \in [n]$. Note that if $c = 0$ here, then there is no corresponding vertex of interest in G_2, and this example serves as a natural boundary case between models in which nomination is possible and those in which it is not. As we will see below in Theorem 23, $c > 0$ may still yield chance performance for any nomination scheme, and the existence of a vertex correspondence does not necessarily imply any performance guarantees.

Remark 8 In addition to the edge-independent and conditionally edge-independent network models considered above, the class of nominatable distributions contains a host of other popular random graph models, including the Exponential Random Graph Model (Frank and Strauss, 1986; Snijders et al., 2006; Robins et al., 2007), the preferential attachment model (Albert and Barabási, 2002), and the Watts-Strogatz small world model (Watts and Strogatz, 1998), among others. Indeed, if we consider the case where $c = n = m$, then any parametric distribution on $\mathcal{G}_n \times \mathcal{G}_n$ is a nominatable distribution.

Remark 9 The core vertices C in a nominatable distribution correspond to the vertices that can be sensibly identified across graphs. Note that this set does not require any further structure, aside from the conditional independence of $G_1[J_1]$ and $G_2[J_2]$ given the parameter θ. Thus, we are largely free to specify any notion of correspondence we please. Depending on the application, this correspondence may be that of vertices playing similar structural roles, belonging to the same community, or some more complicated application-specific notion of correspondence. That is, the notion of cross-graph correspondence, and hence the notion of vertex similarity, is largely left to the practitioner to specify when she or he specifies an appropriate random graph model.

Given a pair of graphs $(G_1, G_2) \sim F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$, if the vertices in C are known across graphs then identifying the corresponding vertex to $v^* \in C$ is immediate from the vertex labels. In practice, this information is unknown and the correspondences across graphs are only partially observed or even unobserved entirely. To model this added uncertainty, we consider passing the vertex labels of G_2 through an obfuscating function.

Definition 10 *Let $(G_1, G_2) \sim F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$. An* obfuscating function $\mathfrak{o} : V_2 \to W$ *is a bijection from V_2 to W with $W \cap V_i = \emptyset$ for $i = 1, 2$. We call a set W satisfying $W \cap V_i = \emptyset$ for $i = 1, 2$ an obfuscating set, and for a given obfuscating set W, we let $\mathfrak{O}_W$ be the set of all obfuscating functions $\mathfrak{o} : V_2 \to W$.*

Here, $\mathfrak{o}$ models the practical reality that the correspondence of labels across graph is unknown a priori. Note that to ease notation, we shall write $\mathfrak{o}(G_2)$ (resp., $\mathfrak{o}(g_2)$ and $\mathfrak{o}(\mathcal{G}_m)$) to denote the graph G_2 (respectively, g_2 and $\mathcal{G}_m$) whose labels have been obfuscated via $\mathfrak{o}$.

Before defining a VN scheme, we must make one additional definition: for a graph $g \in \mathcal{G}_m$ and $u \in V(g)$, define

$$\mathcal{I}(u; g) = \{w \in V(g) \text{ s.t. } \exists \text{ an automorphism } \sigma \text{ of } g, \text{ s.t. } \sigma(u) = w\}.$$

Note that by taking σ to be the identity, we have $u \in \mathcal{I}(u; g)$. The vertices in $\mathcal{I}(u; g)$ are those that are, in a sense, topologically equivalent to the vertex u in g, and hence, in the absence of labels, indistinguishable from one another. As such, any sensibly-defined vertex nomination scheme should view all vertices in $\mathcal{I}(u; g)$ as being equally good matches to a vertex of interest v^*. Thus, a well-defined VN scheme should be "label-independent" in the following sense: The set of ranks of each set of equivalent vertices (i.e., each $\mathcal{I}(u; g_2)$) needs to be invariant to the particular choice of obfuscating function; see Figure 3 for an illustration of this consistency criterion. Formally, we have the following.

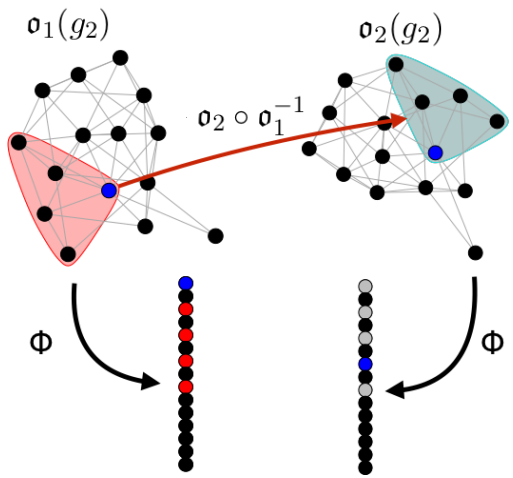


Figure 3: An illustration of the "label-independence" property of VN schemes. If the blue vertex in $\mathfrak{o}_1(g_2)$ (resp., $\mathfrak{o}_2(g_2)$) is $\mathfrak{o}_1(u)$ (resp., $\mathfrak{o}_2(u)$) for $u \in V_2$, then we require the ranks of $\mathcal{I}(\mathfrak{o}_1(u); \mathfrak{o}_1(g_2))$ (outlined in red in the network $\mathfrak{o}_1(g_2)$ and colored red/blue in the ordering provided by Φ) to be equal to the ranks of $\mathcal{I}(\mathfrak{o}_2(u); \mathfrak{o}_2(g_2))$ (outlined in grey in the network $\mathfrak{o}_2(g_2)$ and colored grey/blue in the ordering provided by Φ). Indeed, the set of ranks of $\mathfrak{o}(\mathcal{I}(u; g_2))$ via Φ is independent of the choice of obfuscation function $\mathfrak{o}$.

Definition 11 (Vertex Nomination (VN) Scheme) *For a set A, let $\mathcal{T}_A$ denote the set of all total orderings of the elements of A. For $n, m > 0$ fixed, obfuscating set W, and obfuscating function $\mathfrak{o} \in \mathfrak{O}_W$, a* vertex nomination scheme *is a function $\Phi : \mathcal{G}_n \times \mathfrak{o}(\mathcal{G}_m) \times V_1 \to \mathcal{T}_W$ satisfying the following consistency property: If for each $u \in V_2$, we define $rank_{\Phi(g_1, \mathfrak{o}(g_2), v^*)}\big(\mathfrak{o}(u)\big)$ to be the position of $\mathfrak{o}(u)$ in the total ordering provided by $\Phi(g_1, \mathfrak{o}(g_2), v^*)$, and we define $\mathfrak{r}_\Phi : \mathcal{G}_n \times \mathcal{G}_m \times \mathfrak{O}_W \times V_1 \times 2^{V_2} \mapsto 2^{[m]}$ via*

$$\mathfrak{r}_\Phi(g_1, g_2, \mathfrak{o}, v^*, S) = \{rank_{\Phi(g_1, \mathfrak{o}(g_2), v^*)}\big(\mathfrak{o}(u)\big) \text{ s.t. } u \in S\},$$

then we require that for any $g_1 \in \mathcal{G}_n$, $g_2 \in \mathcal{G}_m$, $v^* \in V_1$, obfuscating functions $\mathfrak{o}_1, \mathfrak{o}_2 \in \mathfrak{O}_W$ and any $u \in V(g_2)$,

$$\mathfrak{r}_\Phi\big(g_1, g_2, \mathfrak{o}_1, v^*, \mathcal{I}(u; g_2)\big) = \mathfrak{r}_\Phi\big(g_1, g_2, \mathfrak{o}_2, v^*, \mathcal{I}(u; g_2)\big) \tag{2}$$
$$\Leftrightarrow \mathfrak{o}_2 \circ \mathfrak{o}_1^{-1}\Big[\mathcal{I}\Big(\Phi(g_1, \mathfrak{o}_1(g_2), v^*)[k]\big); \mathfrak{o}_1(g_2)\Big)\Big] = \mathcal{I}\Big(\Phi(g_1, \mathfrak{o}_2(g_2), v^*)[k]; \mathfrak{o}_2(g_2)\Big), \text{ for all } k \in [m],$$

where $\Phi(g_1, \mathfrak{o}(g_2), v^*)[k]$ denotes the k-th element (i.e., the rank-k vertex) in the ordering $\Phi(g_1, \mathfrak{o}(g_2), v^*)$. We let $\mathcal{V}_{n,m}$ denote the set of all such VN schemes.

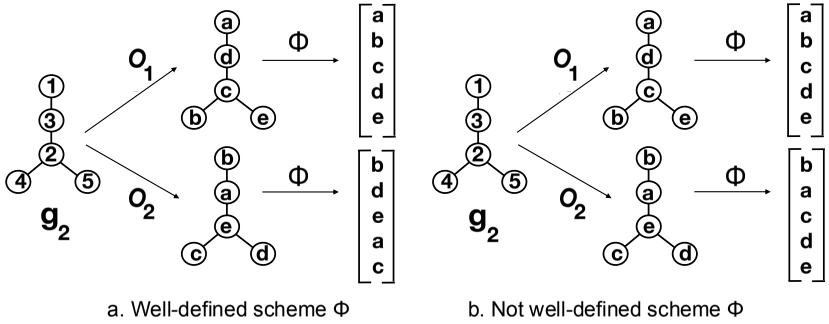


Figure 4: An example of the consistency criterion, Equation (2), in action. The left panel (a) shows a well-defined nomination scheme while the right panel (b) shows an ill-defined scheme. The key in this example is that any scheme satisfying Equation (2) must have $\mathfrak{r}_\Phi(g_1, g_2, \mathfrak{o}_1, v^*, \{1\}) = \mathfrak{r}_\Phi(g_1, g_2, \mathfrak{o}_2, v^*, \{1\})$; $\mathfrak{r}_\Phi(g_1, g_2, \mathfrak{o}_1, v^*, \{3\}) = \mathfrak{r}_\Phi(g_1, g_2, \mathfrak{o}_2, v^*, \{3\})$; $\mathfrak{r}_\Phi(g_1, g_2, \mathfrak{o}_1, v^*, \{2\}) = \mathfrak{r}_\Phi(g_1, g_2, \mathfrak{o}_2, v^*, \{2\})$; and $\mathfrak{r}_\Phi(g_1, g_2, \mathfrak{o}_1, v^*, \{4, 5\}) = \mathfrak{r}_\Phi(g_1, g_2, \mathfrak{o}_2, v^*, \{4, 5\})$.

Given $(G_1, G_2) \sim F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$ realized as $G_1 = g_1$ and $G_2 = g_2$ with $v^* \in V_1$ the vertex of interest in G_1, a VN scheme $\Phi(\cdot, \cdot, \cdot)$ produces a ranked list $\Phi(g_1, \mathfrak{o}(g_2), v^*)$ of the vertices of $\mathfrak{o}(g_2)$ (i.e., the set W), ordered according to how likely each vertex in $V(\mathfrak{o}(g_2))$ is judged to correspond to v^*, with optimal performance corresponding to

$$\Phi(g_1, \mathfrak{o}(g_2), v^*)[1] = \begin{cases} \mathfrak{o}(v^*) & \text{if } v^* \in C \\ \text{arbitrary } v \in W & \text{if } v^* \notin C. \end{cases}$$

Less formally, one can think of a VN scheme as ranking the vertices of G_2 according to how well they resemble the vertex of interest v^* under some task-dependent measure.

Remark 12 Note that if $u \in V_2$ is such that $\mathcal{I}(u; g_2) = \{u\}$ (i.e., u is topologically distinct within g_2), then Equation (2) implies that

$$\text{rank}_{\Phi(g_1, \mathfrak{o}_1(g_2), v^*)}\big(\mathfrak{o}_1(u)\big) = \text{rank}_{\Phi(g_1, \mathfrak{o}_2(g_2), v^*)}\big(\mathfrak{o}_2(u)\big)$$

for any $\mathfrak{o}_1$, $\mathfrak{o}_2$ in $\mathfrak{S}_W$. If $\mathcal{I}(u; g_2)$ contains vertices in addition to u, then Equation (2) implies that the set of vertices topologically equivalent to u (namely, those in $\mathcal{I}(u; g_2)$) must achieve the same ranks via Φ under any two obfuscating functions; see Figure 4 for a simple example of this consistency criterion in action.

Remark 13 (Relation to Fishkind et al. (2015); Lyzinski et al. (2016)) Recall the one-graph vertex nomination task considered in earlier works (Coppersmith, 2014; Fishkind et al., 2015; Lyzinski et al., 2016) and described in Section 1, in which vertices are

considered interesting precisely when they belong to one of K communities in a stochastic block model. While the two-graph VN formulation we consider in the present work (modulo symmetries) involves a single vertex of interest across graphs, the framework is easily extended to the setting where one may have multiple vertices of interest (and not of interest), and in particular it can encode instances of the one-graph version VN problem. To see this, consider an instance of the single-graph VN problem on graph $G = (V, E)$ where V is partitioned into K communities as $V = V_1 \cup V_2 \cup \cdots \cup V_K$ and each of the communities is comprised of labeled (i.e., seed vertices, whose community memberships are observed) and unlabeled (i.e., nonseed, whose community memberships are unobserved) vertices, $V_k = S_k \cup U_k$, where $S_k \subseteq V_k$ is the set of seeds from the k-th block and $U_k \subseteq V_k$ is the set of nonseed vertices. We can encode this one-graph VN instance as an instance of the two-graph problem by encoding additional information in the graph G_1. Construct a vertex set $V' = V \cup \{\ell_1, \ell_2, \ldots, \ell_K\}$. The K new vertices $\{\ell_k\}_{k=1}^K$ will encode the label information present in the graph G. Let $E' = E \cup L$, where $L = \{\{\ell_k, s\} : s \in S_k, k \in [K]\}$, so that edges connect from seed vertices in $S = S_1 \cup S_2 \cup \cdots \cup S_K$ to their corresponding label vertices. Take $G_1 = (V', E')$, and let the interesting vertices (and possible uninteresting vertices) be given by the elements of $S \subseteq V'$. The second graph G_2 is then the subgraph of G induced by the unlabeled vertices $U \subseteq V$ passed through an appropriate obfuscating function. This pair (G_1, G_2), with any $s \in S_1$ chosen to be the interesting vertex, encodes the label information present in the one-graph VN problem as well as the graph structure of G, as required.

3. Bayes Error and Bayes Optimality in Vertex Nomination

Viewing a VN scheme as an information retrieval system suggests that a scheme that puts $\mathfrak{o}(v^*)$ close to the top of the nomination list is potentially of great practical value, even if it fails to obtain perfect performance. Motivated by this, we adapt the recall-at-k metric from classical information retrieval as a measure of performance. To wit, we define the level-k loss function and error for VN as follows.

Definition 14 (VN loss function, level-k error) *Let $\Phi \in \mathcal{V}_{n,m}$ be a vertex nomination scheme and $\mathfrak{o}$ an obfuscating function. For (g_1, g_2) realized from $(G_1, G_2) \sim F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$ with vertex of interest $v^* \in C$, and for $k \in [m-1]$, we define the* level-k nomination loss *via*

$$\begin{aligned} \ell_k(\Phi, g_1, g_2, \mathfrak{o}, v^*) &= \mathbb{1}\{rank_{\Phi(g_1, \mathfrak{o}(g_2), v^*)}(\mathfrak{o}(v^*)) \geq k+1\} \\ &= 1 - \mathbb{1}\{rank_{\Phi(g_1, \mathfrak{o}(g_2), v^*)}(\mathfrak{o}(v^*)) \leq k\}. \end{aligned} \tag{3}$$

The level-k error *of Φ at v^* is then defined to be*

$$\begin{aligned} L_k(\Phi, v^*) &= \mathbb{E}_{(G_1, G_2) \sim F_{c,\theta}^{(n,m)}} \left[\ell_k(\Phi, G_1, G_2, \mathfrak{o}, v^*)\right] \\ &= \mathbb{P}_{(G_1, G_2) \sim F_{c,\theta}^{(n,m)}} \left[rank_{\Phi(G_1, \mathfrak{o}(G_2), v^*)}(\mathfrak{o}(v^*)) \geq k+1\right]. \end{aligned} \tag{4}$$

From the definition of the level-k error in Eq. (4), it is immediate that

$$\begin{aligned} L_1(\Phi, v^*) &= 1 - \mathbb{P}_{(G_1,G_2)\sim F_{c,\theta}^{(n,m)}}\left[\text{rank}_{\Phi(G_1,\mathfrak{o}(G_2),v^*)}(\mathfrak{o}(v^*)) = 1\right] \\ &\geq L_2(\Phi, v^*) = 1 - \mathbb{P}_{(G_1,G_2)\sim F_{c,\theta}^{(n,m)}}\left[\text{rank}_{\Phi(G_1,\mathfrak{o}(G_2),v^*)}(\mathfrak{o}(v^*)) \in \{1,2\}\right] \\ &\geq L_3(\Phi, v^*) = 1 - \mathbb{P}_{(G_1,G_2)\sim F_{c,\theta}^{(n,m)}}\left[\text{rank}_{\Phi(G_1,\mathfrak{o}(G_2),v^*)}(\mathfrak{o}(v^*)) \in \{1,2,3\}\right] \\ &\quad\vdots \\ &\geq L_{m-1}(\Phi, v^*) = 1 - \mathbb{P}_{(G_1,G_2)\sim F_{c,\theta}^{(n,m)}}\left[\text{rank}_{\Phi(G_1,\mathfrak{o}(G_2),v^*)}(\mathfrak{o}(v^*)) \in [m-1]\right], \end{aligned} \tag{5}$$

The level-1 loss function is analogous to the classical 0/1 loss function in classification, as $L_1(\Phi, v^*)$ is simply the probability that Φ fails to "classify" $\mathfrak{o}(v^*)$ as the vertex corresponding to v^* in $\mathfrak{o}(G_2)$ (i.e., fails to rank it first). Considering $1 < k \ll m$ enables us to model the practical loss associated with using a VN scheme to search for $\mathfrak{o}(v^*)$ in $\mathfrak{o}(V_2)$ given limited resources.

Remark 15 Unlike in the classification setting described in Section 1.2, where $L_F(h_n)$ is a random variable indexed by n, the nomination errors defined in Definition 14 are sequences indexed by m and n and are not random. In the classical setting, $L_F(h_n)$ denotes the error rate of a classifier that classifies a single observation X based on n training instances $\{(X_i, Y_i)\}_{i=1}^n$. In the case of VN, the notion of labeled training instances is, at best, more hazy. Indeed, in the present setting, the training data and test data are inseparable. The graphs (or, more specifically, their edges) *are* the training data, and in the present work, the graph orders n, m are better thought of as measuring problem dimension rather than training set size.

Analogous to the classification literature, we are now able to define the concept of Bayes optimality in the VN framework.

Definition 16 (Bayes error of a VN scheme) *Let $(G_1, G_2) \sim F_{c,\theta}^{(n,m)}$ with vertex of interest $v^* \in C$ (where we recall that C is the set of core vertices; see Definition 7), and let $\mathfrak{o} \in \mathfrak{O}_W$ be an obfuscating function. For $k \in [m-1]$, we define the* level-k Bayes optimal *VN scheme to be any element $\Psi \in \arg\min_{\Phi\in\mathcal{V}_{n,m}} L_k(\Phi, v^*)$, and define the* level-$k$ Bayes error *to be $L_k^*(v^*) = L_k(\Psi, v^*)$ for level-k Bayes optimal Ψ.*

Now that we have a notion of Bayes error for VN, it is natural to ask whether an optimal VN scheme exists analogous to the Bayes optimal classifier of Equation (1). Toward this end, let (g_1, g_2) be realized from $(G_1, G_2) \sim F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$, and consider a vertex of interest $v^* \in C$ and obfuscating function $\mathfrak{o} : V_2 \to W$. In order to avoid the technical complexities associated with graph automorphisms, in what follows we will assume that $F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$ is supported on $\mathcal{G}_n^a \times \mathcal{G}_m^a$, where $\mathcal{G}_n^a$ (resp., $\mathcal{G}_m^a$) is the set of asymmetric graphs in $\mathcal{G}_n$ (resp., $\mathcal{G}_m$). For analogous results in networks with symmetries, see Remark 18.

Letting $\simeq$ denote graph isomorphism, for $(g_1, g_2) \in \mathcal{G}_n \times \mathcal{G}_m$ define the set

$$\begin{aligned} (g_1, [\mathfrak{o}(g_2)]) &= \left\{\left(g_1, \tilde{g}_2\right) \text{ s.t. } \mathfrak{o}(\tilde{g}_2) \simeq \mathfrak{o}(g_2)\right\} \\ &= \left\{\left(g_1, \tilde{g}_2\right) \text{ s.t. } \tilde{g}_2 \simeq g_2\right\}. \end{aligned} \tag{6}$$

In order to define the Bayes optimal scheme, we will also need the following restriction of $(g_1, [\mathfrak{o}(g_2)])$: for each $w \in W$, we define

$$(g_1,[\mathfrak{o}(g_2)])_{w=\mathfrak{o}(v^*)} = \left\{\left(g_1, \tilde{g}_2\right) \text{ s.t. } \exists \text{ isomorphism } \sigma \text{ s.t. } \mathfrak{o}(\tilde{g}_2) = \sigma(\mathfrak{o}(g_2)), \sigma(w) = \mathfrak{o}(v^*)\right\}. \tag{7}$$

We are now ready to define a Bayes optimal VN scheme.

For ease of notation, in the sequel we will write $\mathbb{P}_{F_{c,\theta}^{(n,m)}}$ or even simply $\mathbb{P}$ in place of $\mathbb{P}_{(G_1,G_2)\sim F_{c,\theta}^{(n,m)}}$ where there is no risk of ambiguity. Let

$$\mathbf{g} = \left\{\left(g_1^{(i)}, g_2^{(i)}\right)\right\}_{i=1}^h \tag{8}$$

be such that the sets

$$\left\{\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right\}_{i=1}^h$$

partition $\mathcal{G}_n^a \times \mathcal{G}_m^a$. We will call this partition $\mathcal{P}_{n,m}$, where we suppress dependence on $\mathbf{g}$ and $\mathfrak{o}$ for ease of notation. We will define a Bayes optimal scheme Φ^* (independent of the choice of $\mathbf{g}$) piecewise on each element of this partition, and we will prove in Theorem 17 that Φ^* is level-k Bayes optimal for all $k \in [m-1]$:

$$\begin{aligned} \Phi^*\left(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*\right)[1] &\in \underset{u\in W}{\text{argmax}}\, \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)_{u=\mathfrak{o}(v^*)} \;\middle|\; \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right] \\ \Phi^*\left(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*\right)[2] &\in \underset{u\in W\setminus\{\Phi^*[1]\}}{\text{argmax}}\, \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)_{u=\mathfrak{o}(v^*)} \;\middle|\; \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right] \\ &\quad\vdots \\ \Phi^*\left(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*\right)[m] &\in \underset{u\in W\setminus\{\cup_{j\in[m-1]}\Phi^*[j]\}}{\text{argmax}}\, \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)_{u=\mathfrak{o}(v^*)} \;\middle|\; \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right], \end{aligned} \tag{9}$$

with ties broken arbitrarily but deterministically. We refer the interested reader to Appendix B.1 for discussion of the case where ties are allowed in the ranking function. For each element

$$(g_1, g_2) \in \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right) \setminus \left\{\left(g_1^{(i)}, g_2^{(i)}\right)\right\},$$

choose the permutation σ such that $\mathfrak{o}(g_2) = \sigma(\mathfrak{o}(g_2^{(i)}))$, and define

$$\Phi^*(g_1^{(i)}, \mathfrak{o}(g_2), v^*) = \sigma(\Phi^*(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)).$$

Lastly, the following theorem shows that this scheme (uniquely defined up to tie-breaking) is indeed Bayes optimal. A proof can be found in Appendix A.1.

Theorem 17 *Let $\mathfrak{o} \in \mathfrak{O}_W$ be an obfuscating function, and let*

$$\mathbf{g} = \left\{\left(g_1^{(i)}, g_2^{(i)}\right)\right\}_{i=1}^h$$

be such that the sets

$$\left\{\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right\}_{i=1}^h$$

partition $\mathcal{G}_n^a \times \mathcal{G}_m^a$. *Let* $\Phi^* = \Phi^*_{\mathbf{g}}$ *be as defined in Equation* (9). *Suppose that* $(G_1, G_2) \sim F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$ *with* $F_{c,\theta}^{(n,m)}$ *supported on* $\mathcal{G}_n^a \times \mathcal{G}_m^a$, *and consider a vertex of interest* $v^* \in C$. *We have that* $L_k(\Phi^*, v^*) = L_k^*(v^*)$ *for all* $k \in [m-1]$, *partitions* $\mathbf{g}$, *and all obfuscating functions* $\mathfrak{o}$.

Remark 18 The effect of symmetries on Theorem 23 is both subtle and cumbersome, as the specific tie-breaking procedures used in the analogue of Eq. (9) are of great import. To this end, consider $\mathbf{g}$ to be defined as above, and let $T \in \mathcal{T}_W$ be the ordering that specifies the (fixed but otherwise arbitrary) scheme by which elements within each $\mathcal{I}(v; \mathfrak{o}(g_2))$ are ordered. Informally, we will first rank the sets $\mathcal{I}(v; \mathfrak{o}(g_2))$ (rather than the individual vertices), and then use T to rank within and across each of the $\mathcal{I}(v; \mathfrak{o}(g_2))$. Full detail is provided below.

For each $w \in W$ and $v \in V(g_2)$, define

$$(g_1, [\mathfrak{o}(g_2)])_{\mathcal{I}(w;\mathfrak{o}(g_2))=\mathfrak{o}(v)} = \Big\{ \big(g_1, \tilde{g}_2\big) \text{ s.t. } \exists \text{ iso. } \sigma \text{ with } \mathfrak{o}(\tilde{g}_2) = \sigma(\mathfrak{o}(g_2)), \\ \text{and } \sigma(u) = \mathfrak{o}(v) \text{ for some } u \in \mathcal{I}(w; \mathfrak{o}(g_2)) \Big\}. \tag{10}$$

As above, we will define the Bayes' optimal VN scheme on each element of the partition provided via $\mathbf{g}$. We first define a ranking Ψ of the sets

$$\left\{ \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)_{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))=\mathfrak{o}(v^*)} \right\}$$

indexed by u, and then will use T to give the total ordering from Ψ. To wit, for each $i \in [k]$, iteratively define (where ties in the argmax are broken in an arbitrary but nonrandom manner)

$$\begin{aligned}
\Psi\Big(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*\Big)[1] &\in \underset{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))\subset W}{\operatorname{argmax}} \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)_{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))=\mathfrak{o}(v^*)} \,\middle|\, \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right] \\
\Psi\Big(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*\Big)[2] &\in \underset{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))\subset W\setminus\{\Psi[1]\}}{\operatorname{argmax}} \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)_{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))=\mathfrak{o}(v^*)} \,\middle|\, \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right] \\
&\vdots \\
\Psi\Big(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*\Big)[\ell] &\in \underset{\mathcal{I}_{\mathfrak{o}(g_2)}(u)\subset W\setminus\{\cup_{j=1}^{\ell-1}\Psi[j]\}}{\operatorname{argmax}} \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)_{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))=\mathfrak{o}(v^*)} \,\middle|\, \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right].
\end{aligned} \tag{11}$$

For each element

$$(g_1, g_2) \in (g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]) \setminus \{(g_1^{(i)}, g_2^{(i)})\},$$

choose an isomorphism σ such that $\mathfrak{o}(g_2) = \sigma(\mathfrak{o}(g_2^{(i)}))$, and define

$$\Psi(g_1, \mathfrak{o}(g_2), v^*) = \sigma(\Psi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)).$$

Note that the choice of isomorphism σ does not impact the definition of Ψ.

For each $(g_1, g_2) \in \mathcal{G}_n \times \mathcal{G}_m$, we define a VN scheme Φ_T^* from Ψ as follows:
1. Initialize $\Phi_T^*(g_1, \mathfrak{o}(g_2), v^*)$ as an empty list; initialize $j = 1$;
2. If $\Psi(g_1, \mathfrak{o}(g_2), v^*)[j]$ is nonempty, add the top ranked (according to T) element from $\Psi(g_1, \mathfrak{o}(g_2), v^*)[j]$ to the end of $\Phi_T^*(g_1, \mathfrak{o}(g_2), v^*)$, else do nothing; set $j = j + 1 \pmod{|\Psi(g_1, \mathfrak{o}(g_2), v^*)|}$
3. Repeat Step 2 until there are no more vertices to add to $\Phi_T^*(g_1, \mathfrak{o}(g_2), v^*)$.
If $T[1] = \mathfrak{o}(v^*)$, then $\Phi_T^*(g_1, \mathfrak{o}(g_2), v^*)$ is Bayes optimal (as in Theorem 17) in the sense of Definition 16. See Appendix A.1 for details.

Example 1, continued. Let $(G_1, G_2) \sim R\text{-ER}(P)$ for $P, R \in \mathbb{R}^{n\times n}$. Under mild model assumptions, we have that $\lim_{n\to\infty} L_k^*(v^*) = 0$ for any fixed k. This is due to the fact that the optimal graph matching of G_1 to $\mathfrak{o}(G_2)$ will almost surely recover the true vertex labels of $\mathfrak{o}(G_2)$ for n suitably large; i.e.,

$$\operatorname{argmin}_{Q\in\Pi_n} \|AQ - QB\|_F = \{I_n\} \text{ with probability} \to 1,$$

where Π_n is the set of $n \times n$ permutation matrices, A is the adjacency matrix for G_1 and B the adjacency matrix for G_2. More concretely, we have the following theorem adapted to our present setting from Lyzinski and Sussman (2018). A proof sketch can be found in Appendix B.

Theorem 19 *Let* $(A, B) \sim R\text{-ER}(P)$, *and for any fixed permutation matrix* Q *define the random variable* $\delta(Q) := \|AQ - QB\|_F$. *Define* $0 < \epsilon := \min_{i,j;i\neq j} 2R_{i,j}P_{i,j}(1 - P_{i,j})$. *There exist positive constants* c_1, c_2 *such that if* $\epsilon^2 > c_1 \log(n)/n$, *then for sufficiently large* n,

$$\mathbb{P}\left(\exists\ Q \in \Pi_n \setminus \{I_n\} \text{ s.t. } \delta(Q) \le \delta(I_n)\right) \le 2\exp\left\{-c_2\epsilon^2 n\right\}.$$

Similarly to the Bayes optimal scheme, we define the graph matching VN scheme, denoted Φ_M, separately on each element of $\mathcal{P}_{n,n}$. For a given $(\tilde{g}_1, [\mathfrak{o}(\tilde{g}_2)]) \in \mathcal{P}_{n,n}$, let (g_1, g_2) be an fixed element in $(\tilde{g}_1, [\mathfrak{o}(\tilde{g}_2)])$. Define $\Phi_M(g_1, \mathfrak{o}(g_2), v^*)[1]$ to be a fixed but arbitrary element from

$$\left\{\mathfrak{r}^{-1}(v^*) \text{ s.t. } Q_{\mathfrak{r}} \in \operatorname{argmin}_{Q\in\Pi_n} \|AQ - QB\|_F\right\}, \tag{12}$$

where each $\mathfrak{r} : W \to V_2$ appearing above is a bijection and $Q_{\mathfrak{r}}$ its associated permutation matrix (having identified both W and V_2 with the set $[n]$). Define

$$\mathfrak{R}_1 = \Big\{\mathfrak{r} : W \to V_2 \text{ s.t. } \mathfrak{r} \text{ is a bijection with } \mathfrak{r}^{-1}(v^*) = \Phi_M(g_1, \mathfrak{o}(g_2), v^*)[1]\Big\}.$$

If $j > 1$, define $\Phi_M(g_1, \mathfrak{o}(g_2), v^*)[j]$ to be any element of

$$\left\{\mathfrak{r}^{-1}(v^*) \text{ s.t. } Q_{\mathfrak{r}} \in \underset{Q\in\Pi(n)\setminus\{Q_{\mathfrak{r}}:\mathfrak{r}\in\bigcup_{\ell=1}^{j-1}\mathfrak{R}_\ell\}}{\operatorname{argmin}} \|AQ - QB\|_F\right\},$$

where $\mathfrak{R}_j$ is defined analogously to $\mathfrak{R}_1$. For each element $(g_1', g_2') \in (\tilde{g}_1, [\mathfrak{o}(\tilde{g}_2)]) \setminus \{(g_1, g_2)\}$, choose a permutation σ such that $\mathfrak{o}(g_2') = \sigma(\mathfrak{o}(g_2))$, and we then define $\Phi_M(g_1, \mathfrak{o}(g_2'), v^*) = \sigma(\Phi_M(g_1, \mathfrak{o}(g_2), v^*))$. Theorem 19 states that under mild model assumptions, we have that $\Phi_M(G_1, \mathfrak{o}(G_2), v^*)[1] = \mathfrak{o}(v^*)$ asymptotically almost surely, and thus $\lim_{n\to\infty} L_1(\Phi_M, v^*) = 0$. Indeed, in this setting, for any fixed $k \ge 1$, $\lim_{n\to\infty} L_k(\Phi_M, v^*) = 0$. It is then immediate

that $\lim_{n\to\infty} L_k^*(v^*) = 0$ in this model for any fixed k.

The next two examples serve to illustrate how the level-k Bayes error behaves in the presence of stochastically indistinguishable vertices. In essence, we cannot hope to perform better than randomly ordering stochastically equivalent vertices.

Example 3, continued. Let G_1 and G_2 be independent ER(n,p) graphs. Since the vertices are stochastically indistinguishable within each of the two graphs, no nomination scheme can do better than random chance in this model. Thus, with $c = n$, we have that $L_k^*(v^*) = (1 - k/n)(1 + o(1))$ for all $k \in [n-1]$ and all $v^* \in [n]$.

Example 4 Let $p_1, p_2, q \in [0,1]$ with $1 \geq p_1 > p_2 \geq 0$ and $q \neq p_1, p_2$. Define the matrix

$$B = \begin{pmatrix} p_1 & q \\ q & p_2 \end{pmatrix},$$

and let G_1, G_2 be independent SBM$\,(2, B, b_n)$ graphs where $b_n(i) = 1$ if $i \leq n/2$ (n even) and $b_n(i) = 2$ if $i > n/2$. With $c = n$ and the correspondence equal to the identity function, let $(k_n)_{n=2}^{\infty}$ be a nondecreasing divergent sequence satisfying $k_n \leq n/2$ for all $n > 1$, then $\lim_n L_{k_n}^*(v^*) = \lim_n [(1 - 2k_n/n) \vee 0]$ for all v^*. Indeed, $L_{k_n}^*$ is asymptotically equivalent to randomly ordering the $n/2$ vertices in G_2 that are stochastically equivalent to v^*.

4. VN Consistency

With the definition of Bayes optimality and the Bayes optimal scheme in hand, it is now possible to define a notion of consistent vertex nomination analogous to consistent classification in the pattern recognition literature. Before defining a consistent VN rule (i.e., a sequence of VN schemes), we must first define the notion of sequences of nominatable distributions with *nested cores*. Such sequences of distributions are necessary in order to speak sensibly of a sequence of vertex nomination problem instances.

Definition 20 *Let* $\mathbf{F} = \left(F_{c_n,\theta_n}^{(n,m_n)}\right)_{n=n_0}^{\infty}$ *be a sequence of distributions such that* $F_{c_n,\theta_n}^{(n,m_n)} \in \mathcal{N}_{n,m_n}$ *for all* n. *We say that* $\mathbf{F}$ *has* nested cores *if for all* $n_0 \leq n_1 < n_2$, *if*

$$(G_1, G_2) \sim F_{n_1} = F_{c_{n_1},\theta_{n_1}}^{(n_1,m_{n_1})} \text{ and } (\widetilde{G}_1, \widetilde{G}_2) \sim F_{n_2} = F_{c_{n_2},\theta_{n_2}}^{(n_2,m_{n_2})},$$

we have, letting C_1 *and* C_2 *be the core vertices associated with* F_{n_1} *and* F_{n_2} *respectively, and denoting the junk vertices* $J_{1,1}, J_{2,1}, J_{1,2}, \widetilde{J}_{2,2}$ *analogously,*
[i.] $V(G_1) = C_1 \cup J_{1,1} \subset V(\widetilde{G}_1) = C_2 \cup J_{2,1}$;
[ii.] $V(G_2) = C_2 \cup J_{1,2} \subset V(\widetilde{G}_2) = C_2 \cup J_{2,2}$;
[iii.] $C_1 \subset C_2$.

We are now ready to define a consistent VN rule.

Definition 21 *Let*

$$\mathbf{F} = \left(F_n = F_{c_n,\theta_n}^{(n,m_n)}\right)_{n=n_0}^{\infty}$$

be a sequence of distributions such that $F_n \in \mathcal{N}_{n,m(n)}$ *with nested cores satisfying* $\lim_{n\to\infty} m_n = \infty$. *For a given non-decreasing sequence* (k_n), *we say that a VN rule* $\mathbf{\Phi} = (\Phi_{n,m_n})_{n=n_0}^{\infty}$ *is* level-(k_n) consistent *at* v^* *with respect to* $\mathbf{F}$ *if*

$$\lim_{n\to\infty} L_{k_n}(\Phi_{n,m_n}, v^*) - L_{k_n}^*(v^*) = 0,$$

for any sequence of obfuscating functions of V_2 *with* $|V_2| = m_n$. *If a rule* $\mathbf{\Phi}$ *is level-*(k_n) *consistent at* v^* *for a constant sequence* $k_n = k$, $n = 1, 2, \ldots$, *then we say simply that* $\mathbf{\Phi}$ *is* level-k consistent.

Remark 22 Equation (5) has an interesting implication for VN consistency in the setting where $L_{k_n}^*(v^*) \to 0$. In this case, level-(k_n) consistency of a VN rule $\mathbf{\Phi}$ implies that $\mathbf{\Phi}$ is (k'_n)-consistent for all (k'_n) such that $\liminf \frac{k'_n}{k_n} \geq 1$. We conjecture that this implication holds true for the case where $L_{k_n}^*(v^*) \to c > 0$, but this problem remains open at present.

Example 1, continued. Let $\mathbf{F} = (F_{n,\theta_n=(P_n,R_n)}^{(n,n)})$ be a sequence of R_n-ER(P_n) random graph models in $\mathcal{N}_{n,n}$ for some sequence of probability matrices $(P_n)_{n=n_0}^{\infty}$ and correlation matrices $(R_n)_{n=n_0}^{\infty}$. Under mild model assumptions (see Theorem 19), the graph matching vertex nomination rule $\mathbf{\Phi}_M$ defined in Equation (12) above is level-1 consistent, and hence level-(k_n) consistent for all (k_n) sequences.

Example 3, continued. Let $\mathbf{F} = (F_{n,\theta_n=p}^{(n,n)})$ be a sequence of independent ER(n,p) random graph models in $\mathcal{N}$. All vertex nomination rules are level-(k_n) consistent for all (k_n) sequences. This holds for all possible values of $c \in [n]$ in the nested sequence of ER(n,p) distributions, as all VN rules have effectively chance performance, regardless of core size under this model.

We define the consistency of a VN rule with respect to a broad class of graph sequences, and it is perhaps no surprise that there cannot be any level-(k_n) universally consistent VN rules. Indeed, even for constant sequences $k_n := k$ (i.e., those that are level-(k_n) consistent for all sequences of nominatable distributions $\mathbf{F}$ with nested cores) there cannot be any level-k universally consistent VN rules. To prove this result, we will first establish an analogue to the "arbitrary poor performance" theorems for classifiers (see, e.g., Devroye et al., 1997, Theorem 7.1), which state that for a fixed n and m, any VN scheme can be shown to have arbitrarily poor performance with respect to a well-chosen adversarial distribution $F_{c,\theta}^{(n,m)}$. Our theorem mirrors the classical classification literature, as for a given classification rule, there exists "a sufficiently complex distribution for which the sample size n is hopelessly small" (Devroye et al., 1997, pg. 111), so that a classification rule can be made to perform arbitrarily poorly by selecting a suitably complex data distribution. Nonetheless, in the case of classification, this model complexity and the implicit dependence on n can be overcome asymptotically by a classification rule. That is, universally consistent classifiers exist (see, for example, Stone, 1977; Steinwart, 2002; Tang et al., 2013). In contrast, in the VN problem, the complexity of the model generating the data can also grow in n, which effectively thwarts the ability of a VN rule to asymptotically overcome a sequence of adversarial graph models. Formalizing the above, we arrive at the following theorem, a proof of which can be found in Appendix A.2.

Theorem 23 *Let n and m be large enough to guarantee the existence of asymmetric graphs $g_1 \in \mathcal{G}_n$ and $g_2 \in \mathcal{G}_m$. Consider a VN scheme $\Phi \in \mathcal{V}_{n,m}$, obfuscating function $\mathfrak{o}$, and strictly increasing sequence $(\epsilon_i)_{i=1}^m$ satisfying $\epsilon_i \in (0, \frac{i}{m})$. Then there exists a distribution $F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$ over $\mathcal{G}_n \times \mathcal{G}_m$ and $v^* \in C$ such that for each $k \in [c-1]$,*

$$L_k^*(v^*) \leq \epsilon_{m-k} < 1 - \frac{k}{m} < 1 - \epsilon_k < L_k(\Phi, v^*),$$

where $1 - \frac{k}{m}$ represents the error probability of chance performance, i.e., the error probability of a VN scheme in the independent Erdős-Rényi setting.

In the remainder of the section, we will suppress the dependence of $m = m_n$ on n. If we consider sequences $(\epsilon_{m,i})_{i=1}^m$ satisfying the assumptions of Theorem 23 and $\lim_n \epsilon_{m,m-k_n} = \epsilon \in (0,1)$ for a given (k_n) satisfying $k_n = o(m)$, we arrive at the following Corollary, namely that level-(k_n) universally consistent VN schemes do not exist for any sequence $(k_n)_{n=n_0}^\infty$ that does not grow as fast as $m = |V(G_2)|$.

Corollary 24 *Let $\epsilon \in (0,1)$ be arbitrary, and consider a VN rule $\mathbf{\Phi} = (\Phi_{n,m})$. For any nondecreasing sequence $(k_n)_{n=n_0}^\infty$ satisfying $k_n = o(m)$, there exists a sequence of distributions $(F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m})_{n=n_0}^\infty$ with nested cores such that*

$$\limsup_{n\to\infty} L_{k_n}^*(v^*) = \epsilon < 1 = \lim_{n\to\infty} L_{k_n}(\Phi_{n,m}, v^*).$$

Corollary 24 has a number of practical implications. Below, we will briefly outline two such implications. Unlike in the classification setting, where universally consistent rules (e.g., k-nearest neighbors) are theoretically guaranteed to perform well in big-data settings, the VN practitioner enjoys no such certainty. Indeed, in VN, the practitioner first needs to identify the consistency class of a VN rule (i.e., the set of models for which the VN rule is consistent) before applying it in real settings. Unfortunately, identifying and enumerating these consistency classes is theoretically and practically nontrivial, and we are investigating theory and heuristics for this at present. In a streaming data setting, the performance of a universally consistent classifier will approach Bayes optimality for the distribution governing the data, and the classifier will be guaranteed to successfully adapt itself to any changes in the underlying data distribution. The lack of universal consistency in the VN setting implies that his is not the case for graph data, as the performance, as the performance of a consistent VN scheme in the streaming setting could precipitously decline in the presence of distributional shifts in the data. Recognizing these shifts and their potential impact on VN performance is paramount and is the subject of current research.

4.1. Global Consistency

We have just seen that no universally consistent VN schemes exist. This is a consequence of the complexity of the models available when choosing a sequence of nominatable distributions. Indeed, nested-core nominatable sequences $\mathbf{F}$ allow for (nearly) arbitrary dependence structure and model complexity as n increases: corresponding vertex behavior may be correlated (see Example 1), independent (see Example 4), or negatively correlated (see Example 5) across networks. This model flexibility is in service of modeling the complexity of real

world networks, but, as we will demonstrate below, restricting our model class to simpler dependency structures still does not necessarily guarantee the existence of universally consistent schemes.

It is thus natural to explore a weaker notion of consistency, namely consistency for a sufficiently large family of nominatable sequences rather than for all nominatable sequences.

Definition 25 *Let*

$$\mathfrak{F} = \left\{ \mathbf{F}_\alpha = \left(F_{c_n,\theta_n}^{(n,m_n),\alpha} \right)_{n=n_0}^\infty : \alpha \in \mathcal{A} \right\}$$

be a family of nominatable sequences, indexed by some set $\mathcal{A}$. We say that VN scheme Φ is level-(k_n) $\mathfrak{F}$-globally consistent *if Φ is level-(k_n) consistent for every $\mathbf{F} \in \mathfrak{F}$. We call such a family* level-(k_n) globally consistent.

The question of the maximal family $\mathfrak{F}$ for which a level-(k_n) $\mathfrak{F}$-globally consistent rule exists is of prime interest. While we cannot offer a satisfactory complete answer to that question in the present work, we do offer some examples of globally consistent families.

Example 1 continued: In settings where corresponding vertices have correlated neighborhood structures across networks, there is hope for finding globally consistent rules. In the ongoing Example 1, we have seen a simple example of this in the R-ER(P) model, in which the matrix of correlations R encodes a correspondence across the two graphs. As mentioned previously, Theorem 19 asserts that under some mild model assumptions on R and P in the R-ER(P) model, level-1 globally consistent VN rules exist (namely the graph matching VN scheme). If $\mathfrak{F}$ denotes the set of distributions obeying these model assumptions, then we have that level-(k_n) $\mathfrak{F}$-globally consistent rules exist for all sequences (k_n). While we do not expect the conditions of Theorem 19 to produce a maximal level-(k_n) globally consistent family for any given (k_n), this example nonetheless provides an important intuition for the properties such maximal families might possess.

Example 4 continued: The SBM provides a prime example of global consistency. Working in the one-graph framework of Remark 13, under appropriate growth conditions on the parameters of every sequence in family $\mathfrak{F}$, Theorem 6 in Lyzinski et al. (2016) implies the existence of a likelihood-based nomination scheme that is level-$(|U_1|)$ globally consistent for this family of models. Under similar growth conditions, Theorem 6 in Lyzinski et al. (2014) implies the existence of a level-$(|U_1|)$ globally consistent scheme based on spectral clustering, in which vertices are nominated based on their proximity to the vertex or vertices of interest.

Remark 26 An attempt at systematically constructing a maximal globally consistent family might begin by putting model restrictions onto elements of $\mathcal{N}_{n,m}$. A natural restriction to consider would be to demand that the models in $\mathfrak{F}$ be nested in the following sense: For $\mathbf{F} \in \mathfrak{F}$, if $(G_1, G_2) \sim F_{c_{n_2},\theta_{n_2}}^{(n_2,m_{n_2})}$ with $n_1 < n_2$, then $(G_1[[n_1]], G_2[[m_{n_1}]]) \stackrel{\mathcal{L}}{=} (G_1', G_2')$ where $(G_1', G_2') \sim F_{c_{n_1},\theta_{n_1}}^{(n_1,m_{n_1})}$. This property would allow us to consider "streaming" network models $\mathbf{F}$, where for $n_1 < n_2$, if (g_1, g_2) is realized from $(G_1, G_2) \sim F_{c_{n_2},\theta_{n_2}}^{(n_2,m_{n_2})}$, and (g_1', g_2') is realized from $(G_1', G_2') \sim F_{c_{n_1},\theta_{n_1}}^{(n_1,m_{n_1})}$ then (g_1, g_2) can be constructed by appropriately adding

$n_2 - n_1$ vertices to (g_1', g_2'). Additionally, this would serve to mimic the nested nature of the data in the classification consistency literature. However, as we will see in Example 5, global consistency depends both on the dependency structure within each graph (as seen in Theorem 23) *and* the vertex correspondence (i.e., the potential dependency structure across graphs) encoded in the model.

4.2. Behavioral (In)consistency and Global (In)consistency

We suspect that if the vertices of interest have a common distinguishing probabilistic and/or topological characteristic (e.g., correlated neighborhoods, common SBM block structure, high network centrality, etc.) then a globally consistent rule may exist. Indeed, under mild model assumptions, this is the case in the R-ER(P) of Example 1; in the i.i.d. SBM of Example 4 where the correspondence is the identity function (Lyzinski et al., 2014); and in the i.i.d. ER of Example 3, to name a few. In each of these examples, there is a stochastic/topological similarity (or in the ER case, uniformity) between corresponding vertices across networks. In each, corresponding vertices behave similarly across networks. While we suspect that this behavioral similarity is not sufficient for global consistency, Example 5 demonstrates that behavioral inconsistencies within a family of nominatable distributions can preclude the existence of globally consistent nomination rules.

Example 5 For each n, consider n-vertex random graphs $G_1 \sim$ asym-SBM$(2, B_1, b_n^{(1)})$ independent of $G_2 \sim$ asym-SBM$(2, B_2, b_n^{(2)})$, where asym-SBM denotes the stochastic blockmodel distribution restricted to have support on asymmetric graphs. This restriction is made to avoid the unpleasantries of symmetries, and is asymptotically negligible as the SBMs considered in this example are asymptotically almost surely asymmetric.

Case 1. In this case, corresponding vertices behave similarly across networks. To wit, let $\mathbf{F} = (F_n)_{n=n_0}^{\infty}$ be the sequence of models where

$$B_1 = B_2 = \begin{pmatrix} p_1 & q \\ q & p_2 \end{pmatrix}, \; b_n^{(1)}(i) = b_n^{(2)}(i) = \begin{cases} 1 & \text{if } i \leq n/2; \\ 2 & \text{if } i > n/2, \end{cases}$$

$p_1 \neq p_2$, $c = n$, and the correspondence is the identity function. As stated before, in this model $L^*_{n/2}(v^*) \to 0$ for all $v^* \in C$. Without loss of generality, consider $v^* = v_1 = u_1 = 1$. If $\mathbf{\Phi}$ is consistent with respect to $\mathbf{F}$ then

$$\mathbb{P}_{F_n}(\text{rank}_{\Phi(G_1, \mathfrak{o}(G_2), v_1)}(\mathfrak{o}(u_1)) \geq n/2 + 1) \to 0.$$

By the distributional equivalence of vertices within the same block, and the consistency property in the definition of a VN scheme, for any $u, v \in b_n^{-1}(1)$, $k \in [n]$ we have that

$$\mathbb{P}_{F_n}(\text{rank}_{\Phi(G_1, \mathfrak{o}(G_2), v_1)}(\mathfrak{o}(u)) = k) = \mathbb{P}_{F_n}(\text{rank}_{\Phi(G_1, \mathfrak{o}(G_2), v_1)}(\mathfrak{o}(v)) = k).$$

Letting this common value be set to $\alpha_{k,n}$ (with $\beta_{k,n}$ defined similarly as the common value of $\mathbb{P}_{F_n}(\text{rank}_{\Phi(G_1, \mathfrak{o}(G_2), v_1)}(\mathfrak{o}(u)) = k)$ for u in block 2), we have that

$$\sum_{i=1}^{n} \mathbb{P}_{F_n}(\text{rank}_{\Phi(G_1, \mathfrak{o}(G_2), v_1)}(\mathfrak{o}(u_i)) = k) = 1 = \frac{n(\alpha_{k,n} + \beta_{k,n})}{2}$$

giving us that $\alpha_{k,n} = 2/n - \beta_{k,n}$. Consistency implies that

$$\sum_{k=1}^{n/2} \alpha_{k,n} \to 1,$$

which implies that

$$\sum_{k=1}^{n/2} \alpha_{k,n} = \sum_{k=1}^{n/2} (2/n - \beta_{k,n}) = 1 - \sum_{k=1}^{n/2} \beta_{k,n} \to 1,$$

implying $\sum_{k=1}^{n/2} \beta_{k,n} \to 0$. Therefore, for any u in block 2,

$$\mathbb{P}_{F_n}(\text{rank}_{\Phi(G_1, \mathfrak{o}(G_2), v_1)}(\mathfrak{o}(u)) \geq n/2 + 1) \to 1.$$

Case 2. In this case, corresponding vertices behave differently across networks. To wit, let $\tilde{\mathbf{F}} = (\tilde{F}_n)_{n=n_0}^{\infty}$ be the sequence of models where

$$B_1 = \begin{pmatrix} p_1 & q \\ q & p_2 \end{pmatrix}, \quad B_2 = \begin{pmatrix} p_2 & q \\ q & p_1 \end{pmatrix}, \; b_n^{(1)}(i) = b_n^{(2)}(i) = \begin{cases} 1 \text{ if } i \leq n/2; \\ 2 \text{ if } i > n/2, \end{cases}$$

$c = n$, and the correspondence is the identity function. As in Case 1 considered above, in this model $L^*_{n/2}(v^*) \to 0$ for all $v^* \in C$, and, as above, consider $v^* = v_1 = u_1$. If $\mathbf{\Phi}$ is consistent with respect to $\tilde{\mathbf{F}}$ then

$$\mathbb{P}_{\tilde{F}_n}(\text{rank}_{\Phi(G_1, \mathfrak{o}(G_2), v_1)}(\mathfrak{o}(u_1)) \geq n/2 + 1) \to 0.$$

Note that if σ is the permutation such that

$$\sigma(i) = \begin{cases} i + n/2 & \text{if } i \leq n/2; \\ i - n/2 & \text{if } i > n/2, \end{cases},$$

then $\mathbb{P}_{F_n}(g_1, g_2) = \mathbb{P}_{\tilde{F}_n}(g_1, \sigma(g_2))$. Define

$$E_n = \{(g_1, g_2) \text{ s.t. } \text{rank}_{\Phi(g_1, \mathfrak{o}(g_2), v_1)}(\mathfrak{o}(u_1)) \geq n/2 + 1\}\}$$

i.e., and $\tilde{E}_n = \{(g_1, g_2) \text{ s.t. } (g_1, \sigma(g_2)) \in E_n\}$, i.e.,

$$\tilde{E}_n = \{(g_1, g_2) \text{ s.t. } \text{rank}_{\Phi(g_1, \mathfrak{o}(g_2), v_1)}(\mathfrak{o}(u_{n/2+1})) \geq n/2 + 1\}.$$

If $\mathbf{\Phi}$ is consistent with respect to $\mathbf{F}$ we have that $\mathbb{P}_{F_n}(E_n) \to 0$ which implies (as $(g_1, g_2) \in E_n \Leftrightarrow (g_1, \sigma(g_2)) \in \tilde{E}_n$) $\mathbb{P}_{\tilde{F}_n}(\tilde{E}_n) \to 0$. If $\mathbf{\Phi}$ is consistent with respect to $\tilde{\mathbf{F}}$ then $\mathbb{P}_{\tilde{F}_n}(E_n) \to 0$ and $\mathbb{P}_{\tilde{F}_n}(\tilde{E}_n) \to 1$. We arrive at a contradiction, and $\mathbf{\Phi}$ cannot be $(n/2)$-consistent with respect to both $\mathbf{F}$ and $\tilde{\mathbf{F}}$.

Although Example 5 may seem artificial, it is a simple representation of a common phenomenon observed in network data. Often the same entity can behave quite differently across networks; see Patsolic et al. (2017) for an example of this in social networks and Chen et al. (2016b) for an example of this in connectomics. In such a setting, intuition says that a universal scheme that works in both behavioral settings should not exist. Indeed, at least in the simple block model setting considered above, we see that no such scheme exists. Example 5 also highlights an important difference between the VN setting and the more standard classification framework. We already noted that classification's universal consistency relies on the distribution not changing in n, whereas in VN the distributions must vary with n (indeed, the graph sizes grow in n). Further, Example 5 shows that the nonexistence of a universally consistent scheme is not simply a consequence of changing the underlying distributional parameters with n, as these two SBM distributions are (essentially) fixed, in that the matrix B does not change with n. In this example the "training data" provided by G_1 cannot be uniformly beneficial for a single VN scheme across the two differing model settings we consider. Contrast this with the classification setting of Devroye et al. (1997), where the training data uniformly provides progressively better estimates of the class-conditional distributions. Indeed, the training data helps delineate potentially interesting vertices from non-interesting ones in G_2 in Case (1) for one VN scheme, and in Case (2) for another VN scheme, but there does not exist a VN scheme that achieves this desired class separation across both cases.

Remark 27 In the cases considered in Example 5, if we introduce positive edge-wise correlation of

$$\rho \leq \sqrt{\min\left(\frac{p_1(1-p_2)}{p_2(1-p_1)}, \frac{p_2(1-p_1)}{p_1(1-p_2)}\right)}$$

into both Case 1 and Case 2, then under mild assumptions on the growth of p_1 and p_2, joint consistency can be recovered via a USVT centered graph matching nomination scheme (see Lyzinski and Sussman, 2018, for details). This example demonstrates that it is sometimes possible to toggle a family of models to allow for global consistency. A note of caution is needed, however, as in this particular example the correlation ρ is introducing a behavioral consistency across networks that addresses the precise issue brought forth by the behavioral inconsistency in Example 5. In other, more nuanced model families, we do not expect the global-consistency modification (if it indeed exists) to be as straightforward as adding additional edge-correlation into the model.

4.3. Vertex Nomination on Networks with Node Covariates

It is natural to ask if incorporating vertex features into the VN framework can resolve the lack of universally consistent VN schemes. While straightforward to implement, the ameliorating effect of features is significantly more nuanced. Before defining the VN scheme with features, we need the following extension of $\mathcal{I}(v;g)$ for $g \in \mathcal{G}_n$ and $v \in V(g)$. Letting $\mathcal{X}$ be the space of vertex features for graphs in $\mathcal{G}_n$, for $g \in \mathcal{G}_n$, $v \in V(g)$, and $X \in \mathcal{X}^n$ we define

$$\widetilde{\mathcal{I}}(v;g,X) = \{u \in V(g) : \exists \text{ automorphism } \tau \text{ of } g \text{ s.t. } \tau(v) = u \text{ and } X_u = X_v\},$$

where X_v is the feature associated to v via X.

Definition 28 (Vertex Nomination (VN) Scheme with features) *Let $\mathcal{X}$ (resp., $\mathcal{Y}$) be the space of vertex features of graphs in $\mathcal{G}_n$ (resp., $\mathcal{G}_m$). For $n, m > 0$ fixed, obfuscating set W and obfuscating function $\mathfrak{o} \in \mathfrak{O}_W$, a* vertex nomination scheme with features *is a function*

$$\Phi : \mathcal{G}_n \times \mathcal{X}^n \times \mathfrak{o}(\mathcal{G}_m) \times \mathcal{Y}^m \times V_1 \to \mathcal{T}_W$$

satisfying the following consistency property: If for each $u \in V_2$, we define

$$rank_{\Phi(g_1,X,\mathfrak{o}(g_2),\mathfrak{o}(Y),v^*)}\big(\mathfrak{o}(u)\big)$$

to be the position of $\mathfrak{o}(u)$ in the total ordering provided by $\Phi(g_1, X, \mathfrak{o}(g_2), \mathfrak{o}(Y), v^*)$, and we define $\mathfrak{r}_\Phi : \mathcal{G}_n \times \mathcal{X}^n \times \mathcal{G}_m \times \mathcal{Y}^m \times \mathfrak{O}_W \times V_1 \times 2^{V_2} \mapsto 2^{[m]}$ via

$$\mathfrak{r}_\Phi(g_1, X, \mathfrak{o}(g_2), \mathfrak{o}(Y), v^*, S) = \{rank_{\Phi(g_1,X,\mathfrak{o}(g_2),\mathfrak{o}(Y),v^*)}\big(\mathfrak{o}(u)\big) \text{ s.t. } u \in S\},$$

then we require that for any $g_1 \in \mathcal{G}_n$, $g_2 \in \mathcal{G}_m$, $v^* \in V_1$, $X \in \mathcal{X}^n$, $Y \in \mathcal{Y}^m$, obfuscating functions $\mathfrak{o}_1, \mathfrak{o}_2 \in \mathfrak{O}_W$ and any $u \in V(g_2)$,

$$\mathfrak{r}_\Phi\big(g_1, X, \mathfrak{o}_1(g_2), \mathfrak{o}_1(Y), v^*, \widetilde{\mathcal{I}}(u; g_2, Y)\big) = \mathfrak{r}_\Phi\big(g_1, X, \mathfrak{o}_2(g_2), \mathfrak{o}_2(Y), v^*, \widetilde{\mathcal{I}}(u; g_2, Y)\big). \quad (13)$$

We let $\widetilde{\mathcal{V}}_{n,m}$ denote the set of all such VN schemes.

It is immediate that if the features are sufficiently informative, consistency can be established with features where it could not be without. Indeed, consider in Example 5 features that encode the community memberships of a few vertices (e.g., a few vertices whose correspondences across the two graphs are known a priori). Combined with spectral methods, these would be sufficient for consistent VN under either behavior regime. It is also immediate that the fundamental idea presented in Example 5 has an analogue when vertex features are available, as illustrated by the following example.

Example 6 For each n, consider n-vertex random graphs $G_1 \sim$ asym-SBM$(3, B_1, b_n^{(1)})$ independent of $G_2 \sim$ asym-SBM$(3, B_2, b_n^{(2)})$, where asym-SBM again indicates the stochastic block model with support restricted to the asymmetric graphs.
Case 1. In this case, corresponding vertices behave similarly across networks. To wit, let $\mathbf{F} = (F_n)_{n=n_0}^{\infty}$ be the sequence of models where $3|n$ and

$$B_1 = B_2 = \begin{pmatrix} p_1 & q & q \\ q & p_2 & q \\ q & q & p_2 \end{pmatrix}, \; b_n^{(1)}(i) = b_n^{(2)}(i) = \begin{cases} 1 & \text{if } i \leq n/3; \\ 2 & \text{if } i \in (n/3, 2n/3] \\ 3 & \text{if } i > n/3, \end{cases}$$

$p_1 \neq p_2$, $c = n$, the correspondence is the identity function, and $v^* = 1$ is in block 1.
Case 2. In this case, corresponding vertices behave differently across networks. To wit, let $\mathbf{F} = (F_n)_{n=n_0}^{\infty}$ be the sequence of models where $3|n$ and

$$B_1 = \begin{pmatrix} p_1 & q & q \\ q & p_2 & q \\ q & q & p_2 \end{pmatrix}, \; B_2 = \begin{pmatrix} p_2 & q & q \\ q & p_1 & q \\ q & q & p_2 \end{pmatrix}, \; b_n^{(1)}(i) = b_n^{(2)}(i) = \begin{cases} 1 & \text{if } i \leq n/3; \\ 2 & \text{if } i \in (n/3, 2n/3] \\ 3 & \text{if } i > n/3, \end{cases}$$

$p_1 \neq p_2$, $c = n$, the correspondence is the identity function, and $v^* = 1$ is in block 1. Similar to Example 5, without features no VN scheme can be $(n/3)$-consistent for both Cases 1 and 2 simultaneously. In a similar fashion, if we consider features X and Y defined via

$$X_v = Y_v = \begin{cases} 1 & \text{if } b^{(1)}(v) = 1 \\ -1 & \text{if } b^{(1)}(v) = 2, 3, \end{cases}$$

then joint consistency is achievable for both Cases 1 and 2, for example by relying on features and ignoring graph structure. However, if we consider features X and Y defined via

$$X_v = Y_v = \begin{cases} 1 & \text{if } b^{(1)}(v) = 1, 2 \\ -1 & \text{if } b^{(1)}(v) = 3, \end{cases}$$

then joint consistency is again not achievable for both Cases 1 and 2 simultaneously.

This example demonstrates that features, in general, are not enough to ensure universal consistency. Nevertheless, insofar as features supply additional information, they have the potential to improve VN performance. A more thorough examination of the effect and effectiveness of vertex features in VN is beyond the scope of this work, and is the subject of current research.

5. Discussion

In this work, we have introduced a notion of consistency for the vertex nomination task that better reflects the broad range of models under which VN may be deployed. Rather than being restricted to the stochastic block model structure required in previous formulations of the problem, our framework allows for arbitrary dependence structure both within and between graph pairs, while encompassing the original SBM formulation of the problem. Additionally, we have demonstrated how this framework relates to the well-studied notion of Bayes optimal classifiers in the pattern recognition literature. Unlike in the classification setting, we have seen that while Bayes optimal VN schemes always exist, no universally consistent scheme exists. This fact is due essentially to the additional leeway provided by the graph model, in which observing more vertices does not necessarily correspond to receiving more information about the underlying distribution. This is in contrast to the classification setting studied in Stone (1977) and others (Devroye et al., 1997), in which observing more samples allows more accurate estimation of the underlying distribution and class boundary. For this reason, one especially interesting line of investigation concerns the nominatable distributions for which larger n does indeed correspond to more information about the underlying graph distribution. A simple example of this is the initial formulation of the vertex nomination problem, in which observing more vertices allows one to better estimate the model parameters, including the block memberships, and thus more accurately identify the vertices from the interesting block. We suspect that the essential property at play here is that under models of this sort, each vertex is analogous to a sample from a single distribution, though this may not be in and of itself a sufficient condition for consistency. For example, in the case of (G_1, G_2) being i.i.d. or ρ-correlated marginally identical draws from a random dot product graph model with the identity correspondence, each vertex (along with its incident edges) is, in a sense, a noisy sample from the underlying latent

position distribution. Hence, for large n, one can estimate the latent positions or their distribution to arbitrary accuracy, and provided that the latent positions of the interesting vertices are suitably separated from those of the rest of the graph, one should have VN consistency for the collection of these latent position models.

More broadly, it would be good to better understand whether there exist families of nominatable distributions $\mathfrak{F}$ for which certain VN schemes are consistent, and precisely how large these families can be made to be. In a similar vein, it would be of interest to explore how the dependence structure allowed both within and between graphs influences vertex nomination. In particular, if one rules out certain pathologically hard dependence structures as considered in Example 5, can one obtain global consistency with respect to this restricted set of distributions? We hope to explore these questions in future work.

We are also exploring alternative formulations of the VN problem and alternate formulations of the VN loss function. While the extension to multiple vertices of interest in each network G_1 and G_2 is straightforward, we are considering several generalizations of the VN problem considered here. One formulation of prime interest in applications (especially in connectomics and social networks) is as follows: given a collection of vertices of interest in one graph, find those that play a similar structural (based on the topology of the underlying network) or functional (based on vertex or edge covariates) role in the other graph. In addition, as seen in Section 4.3 the impact on VN consistency (and the potential existence of universally consistent schemes) when incorporating edge and vertex covariates into the VN framework is of prime interest, and a deeper analysis of the VN inference task in this framework is the subject of our current and future work.

The loss function considered in the present work is an analogue of the 0/1 recall-at-k loss function in the information retrieval literature. Under this loss function, we have shown that no universally consistent VN rule exists. It is natural to ask whether alternative loss functions can be considered under which universal consistency is achievable. While we conjecture that Example 5 will nearly always provide a counterexample to universal consistency, this question remains open and is the subject of current research.

Acknowledgments

This work is supported in part by the D3M program of the Defense Advanced Research Projects Agency (DARPA), NSF grant DMS-1646108, and by the Air Force Research Laboratory and DARPA, under agreement number FA8750-18-2-0035. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory and DARPA, or the U.S. Government.

Appendix A. Proofs of Main Results

Here we collect the proofs of the two main theorems in this work, Theorems 17 and 23.

A.1. Theorem 17 and Remark 18

In this section, we present proofs of Theorem 17 and the claim in Remark 18.

Proof [Proof of Theorem 17]

Let Φ be an arbitrary VN scheme and Φ^* the scheme defined at Eq. (9). Recall the definition

$$(g_1, [\mathfrak{o}(g_2)])_{w=\mathfrak{o}(v)} = \Big\{ \big(g_1, \tilde{g}_2\big) \text{ s.t. } \exists \text{ iso. } \sigma \text{ with } \mathfrak{o}(\tilde{g}_2) = \sigma(\mathfrak{o}(g_2)), \text{ and } \sigma(w) = \mathfrak{o}(v^*) \Big\}.$$

With $\mathbf{g}$ defined as in the theorem, note that for each $i \in [h]$,

$$\begin{aligned}
U^j_{i,\mathbf{g}} :&= \Big\{ (g_1, g_2) \in \Big(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \Big) \text{ s.t. } \operatorname{rank}_{\Phi(g_1, \mathfrak{o}(g_2), v^*)}(\mathfrak{o}(v^*)) = j \Big\} \\
&= \Big\{ (g_1, g_2) \in \Big(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \Big) \text{ s.t. } \Phi(g_1, \mathfrak{o}(g_2), v^*)[j] = \mathfrak{o}(v^*) \Big\} \\
&= \Big\{ (g_1, g_2) \in \Big(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \Big) \text{ s.t. } \exists \text{ iso. } \sigma \text{ s.t. } \sigma(\mathfrak{o}(g_2^{(i)})) = \mathfrak{o}(g_2) \text{ and} \\
&\qquad\qquad \sigma(\Phi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j]) = \mathfrak{o}(v^*) \Big\} \\
&= \Big(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \Big)_{\Phi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j] = \mathfrak{o}(v^*)}.
\end{aligned}$$

For each $i \in [h]$ define $p_{i,\Phi} \in [0,1]^m$ via

$$p_{i,\Phi}[j] := \mathbb{P}_{F^{(n,m)}_{c,\theta}} \left(U^j_{i,\mathbf{g}} \,\middle|\, (g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]) \right).$$

Observe that for each $i \in [h]$, it is immediate that p_{i,Φ^*} majorizes $p_{i,\Phi}$. To see this, note that for any fixed $\xi \in [m]$, letting $q^{\xi}_{i,\Phi}$ be $(p_{i,\Phi}[j])_{j=1}^{\xi}$ with entries sorted in descending order, we have $p_{i,\Phi^*}[j] \geq q^{\xi}_{i,\Phi}[j]$ for all $j \in [\xi]$, and majorization follows immediately. With $\mathcal{P}_{\mathbf{g}}$ denoting the partition induced by $\mathbf{g}$, this majorization property implies

$$\begin{aligned}
L_\xi(\Phi, v^*) &= 1 - \sum_{j=1}^{\xi} \mathbb{P}\left(\operatorname{rank}_{\Phi(G_1, \mathfrak{o}(G_2), v^*)}(\mathfrak{o}(v^*)) = j \right) \\
&= 1 - \sum_{\mathcal{P}_{\mathbf{g}}} \sum_{j=1}^{\xi} \mathbb{P}\left[U^j_{i,\mathbf{g}} \,\middle|\, (g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]) \right] \mathbb{P}\left[(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]) \right] \\
&= \sum_{\mathcal{P}_{\mathbf{g}}} \left(1 - \sum_{j=1}^{\xi} p_{i,\Phi}[j] \right) \mathbb{P}\left[(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]) \right] \\
&\geq \sum_{\mathcal{P}_{\mathbf{g}}} \left(1 - \sum_{j=1}^{\xi} p_{i,\Phi^*}[j] \right) \mathbb{P}\left[(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]) \right] \\
&= L_\xi(\Phi^*, v^*).
\end{aligned}$$

As Φ, $\mathbf{g}$, and $\mathfrak{o}$ were arbitrary, the proof follows. ■

Proof [Proof of Remark 18] Fix i, and let $\xi_i = |\Psi(g_1^{(i)}, g_2^{(i)}, v^*)|$. Note that for each $j \leq \xi_i$, the set of graphs $(g_1, g_2) \in \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)$ for which $\Psi(g_1, g_2, v^*)[j] = \mathcal{I}(\mathfrak{o}(v^*); \mathfrak{o}(g_2))$ is precisely

$$\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)_{\Psi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j] = \mathfrak{o}(v^*)}.$$

If the tie breaking scheme T satisfies $T[1] = \mathfrak{o}(v^*)$, then the set set of graphs $(g_1, g_2) \in \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)$ for which $\Phi^*_T(g_1, g_2, v^*)[j] = \mathfrak{o}(v^*)$ is then also

$$\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)_{\Psi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j] = \mathfrak{o}(v^*)}.$$

The proof proceeds as follows. For an arbitrary VN scheme Φ, and for each $i \in [h]$,

$$\begin{aligned}
U^j_{i,\mathbf{g}} :&= \Big\{ (g_1, g_2) \in \Big(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \Big) \text{ s.t. } \operatorname{rank}_{\Phi(g_1, \mathfrak{o}(g_2), v^*)}(\mathfrak{o}(v^*)) = j \Big\} \\
&\subset \Big\{ (g_1, g_2) \in \Big(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \Big) \text{ s.t. } \exists \text{ iso. } \sigma \text{ s.t. } \sigma(\mathfrak{o}(g_2^{(i)})) = \mathfrak{o}(g_2) \text{ and} \\
&\qquad\qquad \sigma\left(\mathcal{I}\left(\Phi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j]; \mathfrak{o}(g_2^{(i)}) \right) \right) \ni \mathfrak{o}(v^*) \Big\} \\
&= \Big(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \Big)_{\mathcal{I}\left(\Phi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j]; \mathfrak{o}(g_2^{(i)}) \right) = \mathfrak{o}(v^*)}
\end{aligned}$$

We then have that $T[1] = \mathfrak{o}(v^*)$ implies that

$$\begin{aligned}
1 - L_\ell(\Phi, v^*) &= \sum_{\mathcal{P}_{\mathbf{g}}} \mathbb{P}\left[\left(\bigcup_{j=1}^{\ell} U^j_{i,\mathbf{g}} \right) \mid \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right] \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right] \\
&\leq \sum_{\mathcal{P}_{\mathbf{g}}} \mathbb{P}\left[\bigcup_{j=1}^{\ell} \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)_{\mathcal{I}\left(\Phi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j]; \mathfrak{o}(g_2^{(i)}) \right) = o(v^*)} \mid \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right] \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right] \\
&= \sum_{\mathcal{P}_{\mathbf{g}}} \sum_{j \in \mathcal{J}_i} \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)_{\mathcal{I}\left(\Phi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j]; \mathfrak{o}(g_2^{(i)}) \right) = o(v^*)} \mid \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right] \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right] \\
&\leq \sum_{\mathcal{P}_{\mathbf{g}}} \sum_{j \in \mathcal{J}_i} \underbrace{\mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right)_{\Psi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j] = \mathfrak{o}(v^*)} \mid \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right]}_{= \mathbb{P}\left[(g_1, g_2) \in \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \text{ s.t. } \Phi^*_T(g_1, \mathfrak{o}(g_2), v^*)[j] = \mathfrak{o}(v^*) \,\middle|\, \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right]} \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})] \right) \right] \\
&\leq 1 - L_\ell(\Phi^*_T, v^*),
\end{aligned}$$

where $\mathcal{J}_i \subset [\ell]$ is the lexicographically smallest set of indices for which

$$\left\{ \left(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}) \right)_{\mathcal{I}\left(\Phi(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[j]; \mathfrak{o}(g_2^{(i)}) \right) = o(v^*)} \right\}_{j \in \mathcal{J}_i}$$

are distinct. ■

A.2. Proof of Theorem 23

Proof Define a probability vector $\xi \in \mathbb{R}^m$ by $\xi_i = \epsilon_i - \epsilon_{i-1}$ for $i \in [m-1]$ (where we take $\epsilon_0 := 0$), and let $\xi_m = 1 - \epsilon_{m-1}$. Consider asymmetric graphs $(g_1, g_2) \in \mathcal{G}_n \times \mathcal{G}_m$ and construct a distribution $F_{c,\theta}^{(n,m)} \in \mathcal{N}$ as follows.

i. $c = n \wedge m$;

ii. The support of $F_{c,\theta}^{(n,m)}$ is $(g_1, [\mathfrak{o}(g_2)])$;

iii. For each $k \in [m]$ define

$$R_{\Phi,k} = \left\{(g_1, \tilde{g}_2) \in (g_1, [\mathfrak{o}(g_2)]) \text{ s.t. } \Phi(g_1, \mathfrak{o}(\tilde{g}_2), v^*)[k] = \mathfrak{o}(v^*)\right\}.$$

Then we define $\mathbb{P}_{F_{c,\theta}^{(n,m)}}(R_{\Phi,k}) := \xi[k]$ with all elements of $R_{\Phi,k}$ being assigned equal mass under $F_{c,\theta}^{(n,m)}$.

It is clear then that $L_k(\Phi, v^*) = 1 - \epsilon_k > 1 - \frac{k}{m}$. It is also clear that $L_k^*(v^*) \leq \epsilon_{m-k}$. Indeed, consider Φ' which is defined by reversing the order provided by Φ; then $L_k(\Phi', v^*) = \epsilon_{m-k}$; which completes the proof. ■

Appendix B. Proof of Theorem 19

Herein we will provide a sketch of the proof of Theorem 19 for completeness. Let Q be a permutation matrix in Π_n (with associated permutation σ_Q) that permutes precisely k labels (i.e., $\sum_i Q_{i,i} = n - k$), and let T denote the number of transpositions induced by σ_Q, and note $|T| \leq k/2$. By exploiting the cyclic structure of Q acting on $\text{vec}(B)$, we have that

$$\mathbb{E}\delta(Q) - \mathbb{E}\delta(I_n) = \mathbb{E}\|AQ - QB\|_F^2 - \mathbb{E}\|A - B\|_F^2 \geq \epsilon\left((n-k)k + \binom{k}{2} - T\right).$$

Combining this expectation bound with the following McDiarmid-like concentration result will yield the proof of Theorem 19.

Proposition 29 (Proposition 3.2 from Kim et al. (2002)) *Let $X_1, \ldots, X_n$ be a sequence of independent Bernoulli random variables where $\mathbb{E}[X_i] = p_i$. Let $f : \{0,1\}^n \mapsto \Re$ be such that changing any X_i to $1 - X_i$ changes f by at most*

$$M = \sup_i \sup_{X_1,\ldots,X_n} |f(X_1, \ldots, X_i, \ldots, X_n) - f(X_1, \ldots, 1 - X_i, \ldots, X_n)|.$$

Let $\sigma^2 = M^2 \sum_i p_i(1 - p_i)$ and let $Y = f(X_1, \ldots, X_n)$.
Then

$$\Pr[|Y - \mathbb{E}[Y]| \geq t\sigma] \leq 2e^{-t^2/4}$$

for all $0 < t < 2\sigma/M$.

Indeed, we see that $X_Q := \delta(Q) - \delta(I_n)$ is a function of N_Q independent Bernoulli random variables, where $N_Q = 3\left(\binom{k}{2} + k(n-k) - |T|\right) \leq 3kn$. Let S_Q be the sum of these N_Q Bernoulli random variables, and it follows that $\text{Var}(S_Q) \leq N_Q/4$. By setting $t = C\frac{\epsilon kn}{\sigma}$ for an appropriate constant $C > 0$ in Proposition 29, we have for n sufficiently large

$$\Pr(X_Q \leq 0) \leq \Pr(|X_Q - \mathbb{E}(X_Q)| \geq \mathbb{E}(X_Q)) \leq 2\exp\left\{-\Theta(\epsilon^2 kn)\right\}.$$

A union bound over all such Q (of which there are $\leq n^k$) and over k yields

$$\mathbb{P}(\exists\, Q \in \Pi_n \setminus \{I_n\} \text{ s.t. } \delta(Q) \leq \delta(I_n)) \leq \sum_{k=2}^{n} 2\exp\left\{k\log(n) - \Theta(\epsilon^2 kn)\right\} = 2\exp\left\{-\Theta(\epsilon^2 n)\right\},$$

as desired.

B.1. VN Schemes with Ties

We can incorporate ties into the VN framework as follows. With ties allowed, any sensibly-defined vertex nomination scheme should view all vertices in $\mathcal{I}(u; g)$ as being equally good matches to a vertex of interest v^*. To this end, we will view VN schemes as providing *weak orderings* of the elements of W:

Definition 30 *For a set A, let $\mathfrak{W}_A$ denote the set of all weak orderings of the elements of A (i.e., the set of all total orderings where ties are allowed). For $x \in \mathfrak{W}_A$, let t_x be any maximum-length total ordering induced by x. For each $a \in A$, we define*

$$rank_x(a) = rank_{t_x}(a'),$$

where $a = a'$ according to the ordering x.

Example 7 *If $A = \{a, b, c, d, e\}$ and $x : a > c > d = e > b$, then $t(x) : a > c > d > b$, or $t(x) : a > c > e > b$; in either case, $rank_x(a) = 1$, $rank_x(c) = 2$, $rank_x(d) = 3$, $rank_x(e) = 3$, and $rank_x(b) = 4$.*

A well-defined VN scheme should be "label-independent" in the following sense: Each element of each $\mathcal{I}(\mathfrak{o}(u); \mathfrak{o}(g_2))$ should be ranked identically by Φ, and these ranks should be independent of the obfuscation function $\mathfrak{o}$. Formally, we have the following.

Definition 31 (Vertex Nomination (VN) Scheme) *For $n, m > 0$ fixed, obfuscating set W, and $\mathfrak{o} \in \mathfrak{O}_W$, a* vertex nomination scheme *is a function $\Phi : \mathcal{G}_n \times \mathfrak{o}(\mathcal{G}_m) \times V_1 \to \mathfrak{W}_W$ satisfying the following properties: For all $(g_1, g_2) \in \mathcal{G}_n \times \mathcal{G}_m$,*

i. If $u_1 \notin \mathcal{I}(u_2; g_2)$ then either $\mathfrak{o}(u_1) > \mathfrak{o}(u_2)$ or $\mathfrak{o}(u_1) < \mathfrak{o}(u_2)$ in the ordering provided by $\Phi(g_1, \mathfrak{o}(g_2), v^*)$;

ii. If $u_1 \in \mathcal{I}(u_2; g_2)$ then $\mathfrak{o}(u_1) = \mathfrak{o}(u_2)$ in the ordering provided by $\Phi(g_1, \mathfrak{o}(g_2), v^*)$;

iii. (consistency criterion) If $\mathfrak{o}_1, \mathfrak{o}_2 \in \mathfrak{O}_W$, then for each $v \in V(g_2)$

$$rank_{\Phi(g_1, \mathfrak{o}_1(g_2), v^*)}(\mathfrak{o}_1(v)) = rank_{\Phi(g_1, \mathfrak{o}_2(g_2), v^*)}(\mathfrak{o}_2(v)). \tag{14}$$

We let $\mathcal{V}_{n,m}$ denote the set of all such VN schemes.

The VN loss functions and level-k errors are defined as in the totally ordered setting. To define the Bayes optimal scheme, let (g_1, g_2) be realized from $(G_1, G_2) \sim F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$, and consider a vertex of interest $v^* \in C$ and obfuscating function $\mathfrak{o} : V_2 \to W$. Letting $\simeq$ denote graph isomorphism, for $(g_1, g_2) \in \mathcal{G}_n \times \mathcal{G}_m$ define the set

$$\begin{aligned}(g_1, [\mathfrak{o}(g_2)]) &= \big\{\big(g_1, \tilde{g}_2\big) \text{ s.t. } \mathfrak{o}(\tilde{g}_2) \simeq \mathfrak{o}(g_2)\big\} \\ &= \big\{\big(g_1, \tilde{g}_2\big) \text{ s.t. } \tilde{g}_2 \simeq g_2\big\}\end{aligned} \tag{15}$$

In order to define the Bayes optimal scheme, we will also need the following restrictions of $(g_1, [\mathfrak{o}(g_2)])$: for each $w \in W$, and $v \in V_2$ we define

$$\begin{aligned}(g_1, [\mathfrak{o}(g_2)])_{\mathcal{I}(w;\mathfrak{o}(g_2))=\mathfrak{o}(v)} = \Big\{\big(g_1, \tilde{g}_2\big) \text{ s.t. } \exists \text{ iso. } \sigma \text{ with } \mathfrak{o}(\tilde{g}_2) = \sigma(\mathfrak{o}(g_2)), \\ \text{and } \sigma(u) = \mathfrak{o}(v) \text{ for some } u \in \mathcal{I}(w; \mathfrak{o}(g_2))\Big\}.\end{aligned} \tag{16}$$

Note that for a fixed v, if $\{\mathcal{I}(w; \mathfrak{o}(g_2))\}_{w \in W'}$ partitions W (for some suitable $W' \subseteq W$), then

$$\big\{(g_1, [\mathfrak{o}(g_2)])_{\mathcal{I}(w;\mathfrak{o}(g_2))=\mathfrak{o}(v)}\big\}_{w \in W'}$$

partitions $(g_1, [\mathfrak{o}(g_2)])$. We are now ready to define a Bayes optimal VN scheme.

For ease of notation, in the sequel we will write $\mathbb{P}_{F_{c,\theta}^{(n,m)}}$ or even simply $\mathbb{P}$ in place of $\mathbb{P}_{(G_1,G_2)\sim F_{c,\theta}^{(n,m)}}$ where there is no risk of ambiguity. Let

$$\mathbf{g} = \left\{\left(g_1^{(i)}, g_2^{(i)}\right)\right\}_{i=1}^{h} \tag{17}$$

be such that the sets

$$\left\{\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right\}_{i=1}^{h}$$

partition $\mathcal{G}_n \times \mathcal{G}_m$. We will call this partition $\mathcal{P}_{n,m}$, where we suppress dependence on $\mathbf{g}$ and $\mathfrak{o}$ for ease of notation. We will define a Bayes optimal scheme $\Phi^* = \Phi_{\mathbf{g}}^*$ piecewise on each element of this partition. For each $i \in [h]$, define (where ties in the argmax's are broken in an arbitrary but nonrandom manner)

$$\begin{aligned}\Phi^*\Big(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*\Big)[1] &\in \operatorname*{argmax}_{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))\subset W} \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)_{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))=\mathfrak{o}(v^*)} \,\middle|\, \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right] \\ \Phi^*\Big(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*\Big)[2] &\in \operatorname*{argmax}_{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))\subset W\setminus\{\Phi^*[1]\}} \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)_{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))=\mathfrak{o}(v^*)} \,\middle|\, \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right] \\ &\vdots \\ \Phi^*\Big(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*\Big)[\ell] &\in \operatorname*{argmax}_{\mathcal{I}_{\mathfrak{o}(g_2)}(u)\subset W\setminus\{\cup_{j=1}^{\ell-1}\Phi^*[j]\}} \mathbb{P}\left[\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)_{\mathcal{I}(u;\mathfrak{o}(g_2^{(i)}))=\mathfrak{o}(v^*)} \,\middle|\, \left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right]\end{aligned} \tag{18}$$

so that the ranking provided by Φ^* is (where $\Phi^*(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)[i] = \{u_1^{(i)}, \ldots, u_{n_i}^{(i)}\}$)

$$\underbrace{u_1^{(1)} = u_2^{(1)} = \cdots = u_{n_1}^{(1)}}_{n_1} > \underbrace{u_1^{(2)} = u_2^{(2)} = \cdots = u_{n_2}^{(2)}}_{n_2} > \cdots > \underbrace{u_1^{(\ell)} = u_2^{(\ell)} = \cdots = u_{n_\ell}^{(\ell)}}_{n_\ell}.$$

For each element

$$(g_1', g_2') \in (g_1^{(i)}, [\mathfrak{o}(\tilde{g}_2^{(i)})]) \setminus \{(g_1^{(i)}, g_2^{(i)})\},$$

choose any isomorphism σ such that $\mathfrak{o}(g_2') = \sigma(\mathfrak{o}(g_2^{(i)}))$, and define

$$\Phi^*(g_1, \mathfrak{o}(g_2'), v^*) = \sigma(\Phi^*(g_1^{(i)}, \mathfrak{o}(g_2^{(i)}), v^*)),$$

noting that $\Phi^*(g_1, \mathfrak{o}(g_2'), v^*)$ is independent of the choice of isomorphism σ. The next proposition states that, modulo ties, the definition of $\Phi_{\mathbf{g}}^*$ is independent of the choice of $\mathbf{g}$.

Proposition 32 *Let $\mathfrak{o} \in \mathfrak{O}_W$ be an obfuscating function, and let*

$$\mathbf{g} = \left\{\left(g_1^{(i)}, g_2^{(i)}\right)\right\}_{i=1}^{h} \neq \tilde{\mathbf{g}} = \left\{\left(g_1^{(i)}, \tilde{g}_2^{(i)}\right)\right\}_{i=1}^{h}$$

be such that the sets

$$\left\{\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right\}_{i=1}^{h}, \left\{\left(g_1^{(i)}, [\mathfrak{o}(\tilde{g}_2^{(i)})]\right)\right\}_{i=1}^{h}$$

partition $\mathcal{G}_n \times \mathcal{G}_m$. Suppose that $(G_1, G_2) \sim F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$, and consider a vertex of interest $v^* \in C$. Then there exists a fixed strategy for breaking ties in the argmax's for $\Phi_{\mathbf{g}}^*$ and $\Phi_{\tilde{\mathbf{g}}}^*$ that yields $\Phi_{\mathbf{g}}^* = \Phi_{\tilde{\mathbf{g}}}^*$. In particular, under any such tie-breaking strategy, we have that $L_k(\Phi_{\mathbf{g}}^*, v^*) = L_k(\Phi_{\tilde{\mathbf{g}}}^*, v^*)$ for all $k \in [m-1]$.

Lastly, the following theorem shows that this scheme (or schemes) is indeed Bayes optimal. The proof is analogous to the totally ordered setting and is thus omitted.

Theorem 33 *Let $\mathfrak{o} \in \mathfrak{O}_W$ be an obfuscating function, and let*

$$\mathbf{g} = \left\{\left(g_1^{(i)}, g_2^{(i)}\right)\right\}_{i=1}^{h}$$

be such that the sets

$$\left\{\left(g_1^{(i)}, [\mathfrak{o}(g_2^{(i)})]\right)\right\}_{i=1}^{h}$$

partition $\mathcal{G}_n \times \mathcal{G}_m$. Let $\Phi^* = \Phi_{\mathbf{g}}^*$ be as defined in Equation (9). Suppose that $(G_1, G_2) \sim F_{c,\theta}^{(n,m)} \in \mathcal{N}_{n,m}$, and consider a vertex of interest $v^* \in C$. We have that $L_k(\Phi^*, v^*) = L_k^*(v^*)$ for all $k \in [m-1]$ and all obfuscating functions $\mathfrak{o}$.

References

A. Agarwal, S. Chakrabarti, and S. Aggarwal. Learning to rank networked entities. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining*, pages 14–23, 2006.

S. Agarwal. Learning to rank on graphs. *Machine Learning*, 10(3):333–357, 2010.

E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *The Journal of Machine Learning Research*, 9:1981–2014, 2008.

R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47, 2002.

D. Asta and C. R. Shalizi. Geometric network comparison. In M. Meila and T. Haskes, editors, *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*, pages 102–110, 2015.

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

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold Regularization: A Geometric Framework for Learning from Examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

P. J. Bickel and A. Chen. A nonparametric view of network models and Newman-Girvan and other modularities. *Proc. National Academy of Sciences, USA*, 106:21068–21073, 2009.

S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the 7th International World Wide Web Conference*, 1998.

L. Chen, C. Shen, J. T. Vogelstein, and C. E. Priebe. Robust vertex classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3):578–590, 2016a.

L. Chen, J. T. Vogelstein, V. Lyzinski, and C. E. Priebe. A joint graph inference case study: The C. elegans chemical and electrical connectomes. *Worm*, 5(2), 2016b.

R. R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21:5–30, 2006.

D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 18 (03):265–298, 2004.

G. Coppersmith. Vertex nomination. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(2):144–153, 2014.

G. A. Coppersmith and C. E. Priebe. Vertex nomination via content and context. *arXiv preprint arXiv:1201.4118*, 2012.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1997.

K. K. Duh. *Learning to Rank with Partially-Labeled Data*. PhD thesis, University of Washington, 2009.

P. Erdős and A. Rényi. On random graphs, I. *Publicationes Mathematicae*, 6:290–297, 1959.

P. Erdős and A. Rényi. Asymmetric graphs. *Acta Mathematica Academiae Scientiarum Hungarica*, 14(3–4):295–315, 1963.

D. E. Fishkind, V. Lyzinski, H. Pao, L. Chen, and C. E. Priebe. Vertex nomination schemes for membership prediction. *The Annals of Applied Statistics*, 9(3):1510–1532, 2015.

D. E. Fishkind, S. Adali, H. G. Patsolic, L. Meng, D. Singh, V. Lyzinski, and C. E. Priebe. Seeded graph matching. *Pattern Recognition*, 87:203–215, 2019.

O. Frank and D. Strauss. Markov graphs. *Journal of the american Statistical association*, 81(395):832–842, 1986.

P. D. Hoff, A. E. Raftery, and M. S. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098, 2002.

P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

H. Jeong, S. P. Mason, A.-L. Barabási, and Z. N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411(6833):41–42, 2001.

B. Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83, 2011.

J. H. Kim, B. Sudakov, and V. H. Vu. On the asymmetry of random regular graphs and random graphs. *Random Structures and Algorithms*, 21:216–224, 2002.

H. Li. Learning to rank for information retrieval and natural language processing. *Synthesis Lectures on Human Language Technologies*, 4(1), 2011.

D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the Association for Information Science and Technology*, 58(7):1019–1031, 2007.

T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

L. Lü and T. Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.

C. Luo, W. Pang, Z. Wang, and C. Lin. Hete-CF: Social-based collaborative filtering recommendation using heterogeneous relations. In *Proceedings of the IEEE International Conference on Data Mining*, pages 917–922, 2014.

D. Lusseau and M. E. J. Newman. Identifying the role that animals play in their social networks. *Proceedings of the Royal Society of London B: Biological Sciences*, 271(Suppl 6):S477–S481, 2004.

U. Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

V. Lyzinski. Information recovery in shuffled graphs via graph matching. *IEEE Transactions on Information Theory*, 64(5):3254–3273, 2018.

V. Lyzinski and D. L. Sussman. Matchability of heterogeneous networks pairs. *arXiv preprint arXiv:1705.02294*, 2018.

V. Lyzinski, D. L. Sussman, M. Tang, A. Athreya, and C. E. Priebe. Perfect clustering for stochastic blockmodel graphs via adjacency spectral embedding. *Electronic Journal of Statistics*, 8:2905–2922, 2014.

V. Lyzinski, D. Fishkind, M. Fiori, J.T. Vogelstein, C.E. Priebe, and G. Sapiro. Graph matching: Relax at your own risk. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, In press, 2015.

V. Lyzinski, K. Levin, D. E. Fishkind, and C. E. Priebe. On the consistency of the likelihood maximization vertex nomination scheme: Bridging the gap between maximum likelihood estimation and graph matching. *Journal of Machine Learning Research*, 17(179):1–34, 2016.

H. Ma, I. King, and M. R. Lyu. Mining web graphs for recommendations. *IEEE Transactions on Knowledge and Data Engineering*, 24(6):1051–1064, 2012.

C. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

D. Marchette, C. E. Priebe, and G. Coppersmith. Vertex nomination via attributed random dot product graphs. In *Proceedings of the 57th ISI World Statistics Congress*, volume 6, page 16, 2011.

R. Mihalcea and D. Radev. *Graph-based Natural Language Processing and Information Retrieval*. Cambridge University Press, 2011.

M. E. J. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27(1):39–54, 2005.

M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74(3):036104, 2006.

M. E. J. Newman and A. Clauset. Structure and inference in annotated networks. *Nature Communications*, 7(11863), 2016.

H. G. Patsolic, Y. Park, V. Lyzinski, and C. E. Priebe. Vertex nomination via local neighborhood matching. *arXiv preprint arXiv:1705.00674*, 2017.

T. A. N. Pham, X. Li, G. Cong, and Z. Zhang. A general recommendation model for heterogeneous networks. *IEEE Transactions on Knowledge and Data Engineering*, 28 (12):3140–3153, 2016.

P. Resnick and H. R. Varian. Recommender systems. *Communications of the ACM*, 40(3): 56–58, 1997.

F. Ricci, L. Rokach, and B. Shapira. *Introduction to Recommender Systems Handbook*. Springer, 2011.

G. Robins, P. Pattison, Y. Kalish, and D. Lusher. An introduction to exponential random graph (p^*) models for social networks. *Social Networks*, 29(2):173–191, 2007.

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *Annals of Statistics*, 39:1878–1915, 2011.

S. Rothe and H. Schütze. CoSimRank: A flexible & efficient graph-theoretic similarity measure. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1392–1402, 2014.

T. Snijders, P. Pattison, G. Robins, and M. Handcock. New specifications for exponential random graph models. *Sociological Methodology*, 36(1):99–153, 2006.

I. Steinwart. Support vector machines are universally consistent. *Journal of Complexity*, 18(3):768–791, 2002.

C. Stone. Consistent nonparametric regression. *Annals of Statistics*, 5:595–645, 1977.

M. Sun and C. E. Priebe. Efficiency investigation of manifold matching for text document classification. *Pattern Recognition Letters*, 34(11):1263–1269, 2013.

D. L. Sussman, M. Tang, D. E. Fishkind, and C. E. Priebe. A consistent adjacency spectral embedding for stochastic blockmodel graphs. *Journal of the American Statistical Association*, 107(499):1119–1128, 2012.

D. L. Sussman, V. Lyzinski, Y. Park, and C. E. Priebe. Matched filters for noisy induced subgraph detection. *arXiv preprint arXiv:1803.02423*, 2018.

S. Suwan, D. S. Lee, and C. E. Priebe. Bayesian vertex nomination using content and context. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(6):400–416, 2015.

M. Tang, D. L. Sussman, and C. E. Priebe. Universally consistent vertex classification for latent positions graphs. *The Annals of Statistics*, 41(3):1406–1430, 2013.

M. Tang, A. Athreya, D. L. Sussman, V. Lyzinski, Y. Park, and C. E. Priebe. A semiparametric two-sample hypothesis testing problem for random graphs. *Journal of Computational and Graphical Statistics*, 26(2):344–354, 2017a.

M. Tang, A. Athreya, D. L. Sussman, V. Lyzinski, and C. E. Priebe. A nonparametric two-sample hypothesis testing problem for random dot product graphs. *Bernoulli*, 23(3): 1599–1630, 2017b.

D. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393 (6684):440, 1998.

J. Yoder, L. Chen, H. Pao, E. Bridgeford, K. Levin, D. E. Fishkind, C. E. Priebe, and V. Lyzinski. Vertex nomination: The canonical sampling and the extended spectral nomination schemes. *arXiv preprint arXiv:1802.04960*, 2018.

S. Young and E. Scheinerman. Random dot product graph models for social networks. In *Proceedings of the 5th International Conference on Algorithms and Models for the Web-graph*, pages 138–149, 2007.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In S. Thrun, L. K. Saul, and P. B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 321–328, 2004.

Journal of Machine Learning Research 20 (2019) 1-33 Submitted 2/18; Revised 4/19; Published 4/19

Semi-Analytic Resampling in Lasso

Tomoyuki Obuchi obuchi@c.titech.ac.jp

Yoshiyuki Kabashima kaba@c.titech.ac.jp
Department of Mathematical and Computing Science
Tokyo Institute of Technology
2-12-1, Ookayama, Meguro-ku, Tokyo, Japan

Editor: Jon McAuliffe

Abstract

An approximate method for conducting resampling in Lasso, the ℓ_1 penalized linear regression, in a semi-analytic manner is developed, whereby the average over the resampled datasets is directly computed without repeated numerical sampling, thus enabling an inference free of the statistical fluctuations due to sampling finiteness, as well as a significant reduction of computational time. The proposed method is based on a message passing type algorithm, and its fast convergence is guaranteed by the state evolution analysis, when covariates are provided as zero-mean independently and identically distributed Gaussian random variables. It is employed to implement bootstrapped Lasso (Bolasso) and stability selection, both of which are variable selection methods using resampling in conjunction with Lasso, and resolves their disadvantage regarding computational cost. To examine approximation accuracy and efficiency, numerical experiments were carried out using simulated datasets. Moreover, an application to a real-world dataset, the wine quality dataset, is presented. To process such real-world datasets, an objective criterion for determining the relevance of selected variables is also introduced by the addition of noise variables and resampling. MATLAB codes implementing the proposed method are distributed in (Obuchi, 2018).

Keywords: bootstrap method, Lasso, variable selection, message passing algorithm, replica method

1. Introduction

Variable selection is an important problem in statistics, signal processing, and machine learning. A desire for useful techniques of variable selection has recently been growing, as cumulated advances in measurement and information technologies have started to steadily produce a large amount of high-dimensional data in science and engineering. A naive method for selecting relevant variables requires solving discrete optimization problems. This involves a serious computational difficulty as the dimensionality of the variables increases, even in the simplest case of linear models (Natarajan, 1995). Hence, certain relaxations or approximations are required for handling such large high-dimensional datasets.

A great deal of progress has been made with regard to relaxation techniques by introducing an ℓ_1 penalty (Tibshirani, 1996; Meinshausen and Bühlmann, 2004; Banerjee et al., 2006; Friedman et al., 2008). Its model consistency, that is, whether the estimated model converges to the true model in the large size limit of data, has been extensively studied, particularly in the case of linear models (Knight and Fu, 2000; Zhao and Yu, 2006; Yuan and Lin, 2007; Wainwright, 2009; Meinshausen and Yu, 2009). These studies show that a naive usage of the ℓ_1 penalty affects model consistency in realistic settings: The resultant estimator cannot completely reject variables not present in the true model if there exist non-trivial correlations between covariates, although the normal consistency in the ℓ_2 sense is retained. This fact motivated the development of further techniques for recovering model consistency in the variable selection context, particularly in the last decade, including traditional approaches using confidence interval and hypothesis testing (Zou, 2006; Bach, 2008; Meinshausen and Bühlmann, 2010; Javanmard and Montanari, 2014a,b, 2015; Lockhart et al., 2014; G'Sell et al., 2013; Takahashi and Kabashima, 2018). If the distribution of the estimator is accurately figured out and the corresponding P-value is efficiently computed, those traditional approaches are quite efficient and powerful enough to correctly perform variable selection. However in many practical situations, the functional form of the distribution is not clear. In such cases, certain numerical approaches are employed to approximately estimate the distribution. Resampling is one of the representative methods and is focused on in the present paper.

Resampling is a versatile idea applicable to a wide range of problems in statistical modeling and is broadly used in machine learning algorithms; some examples can be found in boosting and bagging (Trevor et al., 2009). In statistics, Efron's bootstrap method is a pioneering example in which resampling is efficiently and systematically used (Efron and Tibshirani, 1994). In the case of the ℓ_1-penalized linear regression or Lasso (Tibshirani, 1996), through a fine evaluation of each variable's probability to be selected (positive probability) in a fairly general setting, Bach showed that it is possible to perform statistically consistent variable selection by utilizing the bootstrap method (Bach, 2008); the associated algorithm is called *bootstrapped Lasso* (Bolasso). Another algorithm based on a similar idea, called *stability selection* (SS), was also implemented by randomizing the penalty coefficient in Lasso (Meinshausen and Bühlmann, 2010). All these examples demonstrate that resampling is powerful and versatile.

A common disadvantage of such resampling approaches is their computational cost. For example, in the Bolasso case, the ℓ_1-penalized linear regression should be recursively solved according to the number of resampled datasets, which should be sufficiently large so that the positive probability of variables may be estimated. This multiple computational cost precludes the application of such resampling techniques to large datasets. The aim of this study is to avoid this problem by developing an approximation whereby the resampling is conducted in a semi-analytic manner, and to implement it in Lasso.

Such an approximation was already obtained by Malzahn and Opper (2003), where a general framework of semi-analytic resampling was developed using the replica method from statistical mechanics, and was demonstrated in Gaussian process regression. We follow their idea and pursue how it works in Lasso with certain resampling manners.

The remaining of the paper is organized as follows. In the next section, the general framework for semi-analytic resampling using the replica method is reviewed. Concrete formulas in the Lasso case are also provided. After taking the average with respect to the resampling, there remains an intractable statistical model. Handling this model is another key issue, and certain approximations (expected to be exact under certain conditions) should

©2019 Tomoyuki Obuchi and Yoshiyuki Kabashima.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-109.html.

be adopted. In this study, Gaussian approximation is used in conjunction with the so-called cavity method, providing a message passing type algorithm. Its dynamical behavior is analyzed by the so-called state evolution, which guarantees that the algorithm convergence is free from the model dimensionality and the dataset size when covariates are provided as zero-mean independently and identically distributed (i.i.d.) Gaussian random variables. They are explained in Section 3. In Section 4, numerical experiments are carried out on both simulated and real-world datasets to examine the accuracy and efficiency of the proposed semi-analytic method. For processing real-world datasets, an objective criterion for determining the relevance of selected variables is also proposed, based on the addition of noise variables and resampling. The last section concludes the paper.

2. Formulation for semi-analytic resampling

Let us start from preparing notations and definitions. We denote by D a given dataset, which usually consists of a set of inputs $\boldsymbol{x}_\mu$ and outputs y_μ as $D = \{(\boldsymbol{x}_\mu, y_\mu)\}_{\mu=1}^M$, and denote our basic statistics by $\hat{\boldsymbol{\beta}}(D) = (\hat{\beta}_1(D), \cdots, \hat{\beta}_N(D))^\top$. They are interpreted as an estimator of true parameters in the true model that is inferred. Based on the Bayesian inference framework, it is assumed that the basic estimator can be expressed as an average over a certain posterior distribution $P(\boldsymbol{\beta}|\boldsymbol{\lambda}, D)$, that is,

$$\hat{\boldsymbol{\beta}}(\boldsymbol{\lambda}, D) = \int d\boldsymbol{\beta}\, \boldsymbol{\beta} P(\boldsymbol{\beta}|\boldsymbol{\lambda}, D) \equiv \langle \boldsymbol{\beta} \rangle \,, \tag{1}$$

where $\langle \cdot \rangle$ represents the average over the posterior distribution which is supposed to be constructed as follows:

$$P(\boldsymbol{\beta}|\boldsymbol{\lambda}, D) = \frac{1}{Z(\boldsymbol{\lambda}, D)} P_0(\boldsymbol{\beta}|\boldsymbol{\lambda}) P(D|\boldsymbol{\beta}). \tag{2}$$

Here, $P_0(\boldsymbol{\beta}|\boldsymbol{\lambda})$ is the prior distribution characterized by the parameters $\boldsymbol{\lambda}$, and $P(D|\boldsymbol{\beta})$ represents the likelihood. The normalization constant or the partition function can be explicitly expressed by

$$Z(\boldsymbol{\lambda}, D) = \int d\boldsymbol{\beta}\, P_0(\boldsymbol{\beta}|\boldsymbol{\lambda}) P(D|\boldsymbol{\beta}). \tag{3}$$

Considering the construction of a subsample by resampling from the full data D of size M with replacement, let an indicator vector $\boldsymbol{c} = (c_1, \cdots, c_M)^\top$ specify any such subsample; each c_μ is a non-negative integer counting the number of occurrences of the μ-th datapoint of D in the subsample. A subsample identified by $\boldsymbol{c}$ is denoted by $D_{\boldsymbol{c}}$. Specifying a resampling defines the distribution $P(\boldsymbol{c})$ of $\boldsymbol{c}$, and the question is the behavior of the basic estimators and their functions with respect to the average over $P(\boldsymbol{c})$. In some resampling techniques, additional randomization is introduced in the prior distribution (Meinshausen and Bühlmann, 2010). Hence, an average over the parameters $\boldsymbol{\lambda}$ is further considered, and the distribution $P(\boldsymbol{\lambda})$ of $\boldsymbol{\lambda}$ is introduced. For notational simplicity, the average over these distributions is denoted by square brackets with appropriate subscripts as follows:

$$[(\cdots)]_{\boldsymbol{c},\boldsymbol{\lambda}} \equiv \sum_{\boldsymbol{c}} \int d\boldsymbol{\lambda} (\cdots) P(\boldsymbol{c}) P(\boldsymbol{\lambda}).$$

The average of this type is called *configurational average* throughout this paper.

2.1. General theory of semi-analytic resampling

The purpose of resampling is to obtain the distribution of the basic estimator

$$P(\beta_i) = \left[\delta\left(\beta_i - \hat{\beta}_i(\boldsymbol{\lambda}, D_c) \right) \right]_{\boldsymbol{c},\boldsymbol{\lambda}}, \tag{4}$$

which is reduced to computing the moments $\left[\hat{\beta}_i^r(\boldsymbol{\lambda}, D_{\boldsymbol{c}}) \right]_{\boldsymbol{c},\boldsymbol{\lambda}}$ of arbitrary degree $\forall r \in \mathbb{N}$. We thus show that these moments can be evaluated in the following manner.

By definition, the moment $\left[\hat{\beta}_i^r(\boldsymbol{\lambda}, D_{\boldsymbol{c}}) \right]_{\boldsymbol{c},\boldsymbol{\lambda}}$ with $r \in \mathbb{N}$ can be written as

$$\left[\hat{\beta}_i^r(\boldsymbol{\lambda}, D_{\boldsymbol{c}}) \right]_{\boldsymbol{c},\boldsymbol{\lambda}} = \left[\frac{1}{Z^r(\boldsymbol{\lambda}, D_{\boldsymbol{c}})} \int \left\{ \prod_{a=1}^r d\boldsymbol{\beta}^a\, \beta_i^a P_0(\boldsymbol{\beta}^a|\boldsymbol{\lambda}) P(D_{\boldsymbol{c}}|\boldsymbol{\beta}^a) \right\} \right]_{\boldsymbol{c},\boldsymbol{\lambda}} \tag{5}$$

$$= \lim_{n\to 0} \left[Z^{n-r}(\boldsymbol{\lambda}, D_{\boldsymbol{c}}) \int \left\{ \prod_{a=1}^r d\boldsymbol{\beta}^a\, \beta_i^a P_0(\boldsymbol{\beta}^a|\boldsymbol{\lambda}) P(D_{\boldsymbol{c}}|\boldsymbol{\beta}^a) \right\} \right]_{\boldsymbol{c},\boldsymbol{\lambda}} \tag{6}$$

$$\doteq \lim_{n\to 0} \left[\int \left\{ \prod_{b=1}^n d\boldsymbol{\beta}^b \right\} \left\{ \prod_{a=1}^r \beta_i^a \right\} \left\{ \prod_{b=1}^n P_0(\boldsymbol{\beta}^b|\boldsymbol{\lambda}) P(D_{\boldsymbol{c}}|\boldsymbol{\beta}^b) \right\} \right]_{\boldsymbol{c},\boldsymbol{\lambda}}. \tag{7}$$

The evaluation of (5) is technically difficult owing to the existence of $Z^r(\boldsymbol{\lambda}, D_{\boldsymbol{c}})$ in the denominator: The negative power of the partition function does not allow the analytical evaluation of the average over $\boldsymbol{c}$ and $\boldsymbol{\lambda}$, even if $P(\boldsymbol{c})$ and $P(\boldsymbol{\lambda})$ take reasonable forms. To overcome this difficulty, an auxiliary parameter n is introduced in (6). The exponent n is assumed to be a positive integer larger than r in (7); thus, the power of the partition function can be expanded in the integral form (3). Integral variables $\{\boldsymbol{\beta}^1, \boldsymbol{\beta}^2, \ldots, \boldsymbol{\beta}^n\}$ are termed *replicas* since they are regarded as constituting n copies of the original system. In (7), the configurational average can be analytically computed under appropriate approximations. This yields an expression as a function of n that is analytically continuable from $\mathbb{N}$ to $\mathbb{R}$. The $n \to 0$ limit is taken by employing the analytically continued expression after all computations. The evaluation technique based on these procedures is often called the *replica method* (Mézard et al., 1987; Nishimori, 2001; Dotsenko, 2005). Evidently, this technique is not justified in a strict sense, but it is known that the replica method gives correct results for many problems. Justification of the replica method is known to be difficult (Talagrand, 2003), and here we leave it as a future work and just employ the method for our purpose. In the present problem, it is far from trivial to obtain an analytically continuable expression, and below we concentrate on its derivation.

Accordingly, the configurational average of any power of the basic estimator can be evaluated from the following quantities:

$$\Xi\left(\{\boldsymbol{\beta}^a\}_{a=1}^n \mid D \right) \equiv \int \left\{ \prod_{a=1}^n d\boldsymbol{\beta}^a \right\} \left[\prod_{a=1}^n P_0(\boldsymbol{\beta}^a|\boldsymbol{\lambda}) P(D_{\boldsymbol{c}}|\boldsymbol{\beta}^a) \right]_{\boldsymbol{c},\boldsymbol{\lambda}},$$

$$P\left(\{\boldsymbol{\beta}^a\}_{a=1}^n \mid D \right) \equiv \frac{1}{\Xi\left(\{\boldsymbol{\beta}^a\}_{a=1}^n \mid D \right)} \left[\prod_{a=1}^n P_0(\boldsymbol{\beta}^a|\boldsymbol{\lambda}) P(D_{\boldsymbol{c}}|\boldsymbol{\beta}^a) \right]_{\boldsymbol{c},\boldsymbol{\lambda}},$$

where the former is called replicated partition function and the latter is called replicated Boltzmann distribution. The average over the replicated Boltzmann distribution is denoted by

$$\overline{\langle\cdots\rangle} \equiv \int \left\{\prod_{a=1}^{n} d\boldsymbol{\beta}^a\right\} (\cdots)\, P\left(\{\boldsymbol{\beta}^a\}_{a=1}^n \,|\, D\right). \tag{8}$$

It follows from (5)–(7) that this average converges to the desired total average over the posterior and the configurational variables $\boldsymbol{c}$ and $\boldsymbol{\lambda}$ as $n \to 0$. The next step is to compute this average. The nontrivial technical difficulties are in the integrations with respect to replicas $\{\boldsymbol{\beta}^a\}_{a=1}^n$ and in deriving a functional form that is analytically continuable with respect to n. These will be handled after considering the specific case of Lasso.

2.2. Specifics to Lasso with independent sampling

The above general theory is now rephrased in the context of Lasso. Given a set of covariates $\boldsymbol{x}_\mu \in \mathbb{R}^N$ and responses $y_\mu \in \mathbb{R}$, $D = \{(\boldsymbol{x}_\mu, y_\mu)\}_{\mu=1}^M$, the usual estimator by Lasso is

$$\hat{\boldsymbol{\beta}}(\lambda, D) = \arg\min_{\boldsymbol{\beta}} \left\{ \frac{1}{2} \sum_{\mu=1}^{M} \left(y_\mu - \sum_i x_{\mu i} \beta_i \right)^2 + \lambda ||\boldsymbol{\beta}||_1 \right\}.$$

In contrast to this, considering Bolasso and SS, given the resampled data $D_{\boldsymbol{c}}$ and the randomized penalty coefficients $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_N)^\top$, the following estimator is introduced:

$$\hat{\boldsymbol{\beta}}(\boldsymbol{\lambda}, D_{\boldsymbol{c}}) = \arg\min_{\boldsymbol{\beta}} \left\{ \frac{1}{2} \sum_{\mu=1}^{M} c_\mu \left(y_\mu - \sum_i x_{\mu i} \beta_i \right)^2 + \sum_{i=1}^{N} \lambda_i |\beta_i| \right\}. \tag{9}$$

For representing this in terms of the posterior average, the following quantities are introduced:

$$\begin{aligned}
\mathcal{H}(\boldsymbol{\beta}|\boldsymbol{\lambda}, D_{\boldsymbol{c}}) &= \frac{1}{2} \sum_{\mu=1}^{M} c_\mu \left(y_\mu - \sum_i x_{\mu i} \beta_i \right)^2 + \sum_{i=1}^{N} \lambda_i |\beta_i|, \\
Z_\gamma(\boldsymbol{\lambda}, D_{\boldsymbol{c}}) &= \int d\boldsymbol{\beta}\, e^{-\gamma \mathcal{H}(\boldsymbol{\beta}|\boldsymbol{\lambda}, D_{\boldsymbol{c}})}, \\
P_\gamma(\boldsymbol{\beta}|\boldsymbol{\lambda}, D_{\boldsymbol{c}}) &= \frac{1}{Z_\gamma} e^{-\gamma \mathcal{H}(\boldsymbol{\beta}|\boldsymbol{\lambda}, D_{\boldsymbol{c}})},
\end{aligned}$$

where these quantities are called (in the order they appear) Hamiltonian, partition function, and Boltzmann distribution, in accordance with physics terminology. As $\gamma \to \infty$, the Boltzmann distribution converges to a pointwise measure at $\hat{\boldsymbol{\beta}}(\boldsymbol{\lambda}, D_{\boldsymbol{c}})$ and thereby becomes the desired posterior distribution, allowing the identification of the Boltzmann distribution with the posterior distribution; thus, the average over the Boltzmann distribution is thereafter denoted by $\langle\cdots\rangle$ introduced in (1)[1]. The prior distribution $P_0(\boldsymbol{\beta}|\boldsymbol{\lambda})$ and the likelihood

1. It is assumed that the order of the integration and the $\gamma \to \infty$ limit may be changed.

$P(D|\boldsymbol{\beta})$ in (2) correspond to the factors $e^{-\gamma \sum_i \lambda_i |\beta_i|}$ and $e^{-\frac{\gamma}{2} \sum_{\mu=1}^M c_\mu \left(y_\mu - \sum_i x_{\mu i} \beta_i\right)^2}$ in the Boltzmann distribution, respectively.

Considering Bolasso and SS, we draw the subsample $\boldsymbol{c}$ of a fixed size m in an unbiased manner. Each sample comes out with a probability of $M^{-m} m!/(c_1! \cdots c_M!)$, which has a weak dependency among $\{c_\mu\}_\mu$. However, this dependency is not essential for large M and m; thus, it is ignored. Using Stirling's formula $m! \approx (\frac{m}{e})^m$ in conjunction with the relation $m = \sum_{\mu=1}^M c_\mu$, c_μ is approximately regarded as an i.i.d. variable from a Poisson distribution of mean $\tau = m/M$, namely,

$$P(\boldsymbol{c}) = \prod_{\mu=1}^{M} \frac{\tau^{c_\mu}}{c_\mu!} e^{-\tau} = \prod_{\mu=1}^{M} P(c_\mu).$$

It is also natural to require that $P(\boldsymbol{\lambda})$ is factorized into a batch of identical distributions as follows:

$$P(\boldsymbol{\lambda}) = \prod_{i=1}^{N} P(\lambda_i).$$

At this point, the explicit form of $P(\lambda_i)$ is not specified. These factorized natures allow us to express the replicated Boltzmann distribution as

$$P_\gamma(\{\boldsymbol{\beta}^a\}_{a=1}^n | D) \propto \left[e^{-\gamma \sum_{a=1}^n \mathcal{H}(\boldsymbol{\beta}^a|\boldsymbol{\lambda}, D_{\boldsymbol{c}})} \right]_{\boldsymbol{c},\boldsymbol{\lambda}} = \prod_{\mu=1}^{M} \Phi_\mu\left(\{\boldsymbol{\beta}_i\}_{i=1}^N\right) \prod_{i=1}^{N} \Psi\left(\boldsymbol{\beta}_i\right), \tag{10}$$

where

$$\begin{aligned}
\Phi_\mu\left(\{\boldsymbol{\beta}_i\}_{i=1}^N\right) &= \left[e^{-\frac{1}{2}\gamma c \sum_a (y_\mu - \sum_i x_{\mu i} \beta_i^a)^2} \right]_c, \\
\Psi\left(\boldsymbol{\beta}_i\right) &= \left[e^{-\gamma \lambda \sum_{a=1}^n |\beta_i^a|} \right]_\lambda,
\end{aligned}$$

and the vector notation $\boldsymbol{\beta}_i = (\beta_i^a)_a$ was introduced for later convenience. Φ_μ is hereafter called μ-th potential function.

To proceed further, an approximation should be introduced to make the right-hand side of (10) tractable. Although there are several ways for that, we here make an approximation based on the cavity method from statistical physics. The details are in the next section.

3. Handling the replicated system

We below work with the cavity method, or the belief propagation (BP) in computer science, and use a Gaussian approximation. This treatment is essentially identical to that used in deriving the known approximate message passing (AMP) (Kabashima, 2003; Donoho et al., 2009), and can be justified if each covariate is i.i.d. from Gaussian distributions (Bayati and Montanari, 2011; Barbier et al., 2017). For treating nontrivial correlations between covariates, more sophisticated approximations (Opper and Winther, 2001a,b, 2005; Kabashima and Vehkapera, 2014; Çakmak et al., 2014; Cespedes et al., 2014; Rangan et al., 2016; Ma and Ping, 2017; Takeuchi, 2017) are required. Such extensions are left as future work.

(10) implies that the replicated Boltzmann distribution naturally has a factor graph structure. Hence, by the cavity method, (10) can be "approximately" decomposed into two messages as follows:

$$\widetilde{\phi}_{\mu\to i}(\boldsymbol{\beta}_i) = \frac{1}{Z_{\mu\to i}} \int \prod_{j(\neq i)} d\boldsymbol{\beta}_j \; \Phi_\mu\left(\{\boldsymbol{\beta}_i\}_{i=1}^N\right) \prod_{j(\neq i)} \phi_{j\to\mu}(\boldsymbol{\beta}_j), \tag{11}$$

$$\phi_{i\to\mu}(\boldsymbol{\beta}_i) = \frac{1}{Z_{i\to\mu}} \Psi\left(\boldsymbol{\beta}_i\right) \prod_{\nu(\neq\mu)} \widetilde{\phi}_{\nu\to i}(\boldsymbol{\beta}_i), \tag{12}$$

where $Z_{\mu\to i}, Z_{i\to\mu}$ are normalization factors that are not relevant and will be discarded below. The average of (8) can be computed by employing this set of equations.

3.1. Gaussian approximation on cavity method

A crucial observation for assessing (11),(12) is that the residual $R_\mu^a = y_\mu - \sum_j x_{\mu j}\beta_j^a$ appearing in Φ_μ has a sum of a large number of random variables; thus, the central limit theorem justifies treating it as a Gaussian variable with the appropriate mean and variance. This consideration leads to the following decomposition in (11)

$$R_\mu^a + x_{\mu i}\beta_i^a \equiv y_\mu - \sum_{j(\neq i)} x_{\mu j}\beta_j^a \approx y_\mu - \sum_{j(\neq i)} x_{\mu j}\overline{\beta}_j^{\backslash\mu} + Z_{\mu i}^a \equiv r_{\mu i} + Z_{\mu i}^a,$$

where $Z_{\mu i}^a$ is a zero-mean Gaussian variable whose covariance is equivalent to that of $\sum_{j(\neq i)} x_{\mu j}\beta_j^a$, and $\overline{(\cdots)}^{\backslash\mu}$ is the average over $\prod_j \phi_{j\to\mu}(\boldsymbol{\beta}_j)$ or the replicated Boltzmann distribution without μ-th potential function. This shows that it is difficult to consider the correlation between β_i and β_j with different i, j in the present framework. This could be overcome even in the cavity method framework (Opper and Winther, 2001a,b); however, it is beyond the scope of this study.

The so-called replica symmetry is now assumed in the messages. This symmetry implies that any permutation of the replicas $\{\beta_i^1, \cdots, \beta_i^n\}$ yields an identical message, which in turn implies, by De Finetti's theorem (Hewitt and Savage, 1955), that the messages can be expressed as

$$\phi_{i\to\mu}(\boldsymbol{\beta}_i) = \int d\mathcal{F}\rho_{i\to\mu}(\mathcal{F}) \prod_{a=1}^n \mathcal{F}(\beta_i^a),$$
$$\widetilde{\phi}_{\mu\to i}(\boldsymbol{\beta}_i) = \int d\mathcal{G}\rho_{\mu\to i}(\mathcal{G}) \prod_{a=1}^n \mathcal{G}(\beta_i^a),$$

where $d\mathcal{F}\rho_{i\to\mu}(\mathcal{F})$ and $d\mathcal{G}\rho_{i\to\mu}(\mathcal{G})$ are probability measures over the probability distribution functions $\mathcal{F}$ and $\mathcal{G}$, respectively. From this form, two different variances naturally emerge:

$$V_i^{\backslash\mu} \equiv \overline{(\beta_i^a)^2 - \beta_i^a\beta_i^b}^{\backslash\mu} = \int d\boldsymbol{\beta}_i \left((\beta_i^a)^2 - \beta_i^a\beta_i^b\right) \int d\mathcal{F}\rho_{i\to\mu}(\mathcal{F}) \prod_{c=1}^n \mathcal{F}(\beta_i^c)$$
$$= \int d\mathcal{F}\rho_{i\to\mu}(\mathcal{F}) \left\{ \int \mathcal{F}(\beta)\beta^2 d\beta - \left(\int \mathcal{F}(\beta)\beta d\beta\right)^2 \right\},$$
$$W_i^{\backslash\mu} \equiv \overline{\beta_i^a\beta_i^b}^{\backslash\mu} - \overline{\beta_i^a}^{\backslash\mu}\overline{\beta_i^b}^{\backslash\mu}$$
$$= \int d\mathcal{F}\rho_{i\to\mu}(\mathcal{F}) \left(\int \mathcal{F}(\beta)\beta d\beta\right)^2 - \left(\int d\mathcal{F}\rho_{i\to\mu}(\mathcal{F}) \int \mathcal{F}(\beta)\beta d\beta\right)^2,$$

where it was assumed that $a \neq b$. Below, the μ-dependence of $V_i^{\backslash\mu}$ and $W_i^{\backslash\mu}$ is ignored, as it is small, and let

$$V_i^{\backslash\mu} \approx V_i = \overline{(\beta_i^a)^2 - \beta_i^a\beta_i^b},$$
$$W_i^{\backslash\mu} \approx W_i = \overline{\beta_i^a\beta_i^b} - \overline{\beta_i^a}\,\overline{\beta_i^b}.$$

It can be implied that V_i describes the average of the variance inside a fixed resampled dataset, whereas W_i represents the inter-sample variance. Using V_i and W_i, the covariance of $\boldsymbol{\beta}_i$ is generally written as

$$\mathrm{Cov}^{\backslash\mu}\left(\beta_i^a, \beta_i^b\right) \equiv \overline{\beta_i^a\beta_i^b}^{\backslash\mu} - \overline{\beta_i^a}^{\backslash\mu}\overline{\beta_i^b}^{\backslash\mu} = W_i^{\backslash\mu} + V_i^{\backslash\mu}\delta_{ab} \approx W_i + V_i\delta_{ab}.$$

Using this relation, the covariance of $\{Z_{\mu i}^a\}_a$ can be expressed as

$$\mathrm{Cov}^{\backslash\mu}\left(Z_{\mu i}^a, Z_{\mu i}^b\right) = \sum_{j,k(\neq i)} x_{\mu i}x_{\mu j}\mathrm{Cov}^{\backslash\mu}\left(\beta_j^a, \beta_k^b\right) \approx \sum_j x_{\mu j}^2 \mathrm{Cov}^{\backslash\mu}\left(\beta_j^a, \beta_j^b\right)$$
$$\approx \sum_j x_{\mu j}^2 (W_j + V_j\delta_{ab}) \equiv W_\mu + V_\mu\delta_{ab}.$$

The correlation between β_j and β_k for $j \neq k$ is neglected in the second sum because it is not taken into account in the present framework, as explained above. Moreover, the addition of the i-th term in the same sum does not affect the following discussion because it is sufficiently small in the summation. Based on these observations, $Z_{\mu i}^a$ is decomposed as follows:

$$Z_{\mu i}^a = D_{\mu i} + \Delta_{\mu i}^a,$$

where $D_{\mu i}$ and $\Delta_{\mu i}^a$ are zero-mean Gaussian variables whose covariances are $\mathrm{Cov}^{\backslash\mu}(D_{\mu i}, D_{\mu i}) = W_\mu$, $\mathrm{Cov}^{\backslash\mu}\left(D_{\mu i}, \Delta_{\mu i}^a\right) = 0$, $\mathrm{Cov}^{\backslash\mu}\left(\Delta_{\mu i}^a, \Delta_{\mu i}^b\right) = V_\mu\delta_{ab}$.

$\widetilde{\phi}_{\mu\to i}$ can now be computed. The integration with respect to $\{\boldsymbol{\beta}_j\}_{j(\neq i)}$ in (11) is replaced by that over D_i and Δ_i^a. The result is

$$\widetilde{\phi}_{\mu\to i}(\boldsymbol{\beta}_i) \approx \int dD_{\mu i}P(D_{\mu i}) \int \prod_a d\Delta_{\mu i}^a P(\Delta_{\mu i}^a) \left[e^{-\frac{1}{2}\gamma c\sum_a (r_{\mu i} - x_{\mu i}\beta_i^a + D_{\mu i} + \Delta_{\mu i}^a)^2}\right]_c$$
$$= \left[U(c)^{-\frac{1}{2}}\left(1 + c\gamma V_\mu\right)^{-\frac{n}{2}} e^{-\frac{1}{2}S(c)\sum_a\left(r_{\mu i} - x_{\mu i}\beta_i^a\right)^2 + \frac{1}{2}T(c)\left(\sum_a (r_{\mu i} - x_{\mu i}\beta_i^a)\right)^2}\right]_c,$$

where

$$S(c) = \frac{c\gamma}{1+c\gamma V_\mu},$$
$$T(c) = \frac{c^2\gamma^2 W_\mu}{(1+c\gamma V_\mu)(1+c\gamma(V_\mu+nW_\mu))},$$
$$U(c) = \frac{1+c\gamma(V_\mu+nW_\mu)}{1+c\gamma V_\mu}.$$

$\ln\widetilde{\phi}_{\mu\to i}(\boldsymbol{\beta}_i)$ can be expanded with respect to $x_{\mu i}\beta_i^a$, and the expansion up to the second order leads to an effective Gaussian approximation of the message, yielding

$$\widetilde{\phi}_{\mu\to i}(\boldsymbol{\beta}_i) \approx e^{-\frac{1}{2}\gamma\tilde{A}_{\mu\to i}\sum_a\left(\beta_i^a\right)^2+\frac{1}{2}\gamma^2\tilde{C}_{\mu\to i}\left(\sum_a\beta_i^a\right)^2+\gamma\tilde{B}_{\mu\to i}\sum_a\beta_i^a},$$

where

$$\tilde{A}_{\mu\to i} = \left[\frac{c}{1+c\gamma V_\mu}\right]_c x_{\mu i}^2, \tag{16a}$$
$$\tilde{B}_{\mu\to i} = \left[\frac{c}{1+c\gamma V_\mu}\right]_c r_{\mu i}x_{\mu i}, \tag{16b}$$
$$\tilde{C}_{\mu\to i} = \left\{\left[\left(\frac{c}{1+c\gamma V_\mu}\right)^2\right]_c W_\mu + \left(\left[\left(\frac{c}{1+c\gamma V_\mu}\right)^2\right]_c - \left[\frac{c}{1+c\gamma V_\mu}\right]_c^2\right)(r_{\mu i})^2\right\} x_{\mu i}^2. \tag{16c}$$

It should be noted that the $n\to 0$ limit was already taken in these coefficients.

Owing to the Gaussian approximation of $\widetilde{\phi}_{\mu\to i}(\boldsymbol{\beta}_i)$, the marginal distribution of the replicated Boltzmann distribution can be simply written as

$$P_i(\boldsymbol{\beta}_i|D) \equiv \int\prod_{j(\neq i)} d\boldsymbol{\beta}_j P(\{\boldsymbol{\beta}_i\}_i|D) \propto \Psi(\boldsymbol{\beta}_i)\prod_\mu \widetilde{\phi}_{\mu\to i}(\boldsymbol{\beta}_i)$$
$$\approx \left[\int Dz\; e^{\gamma\left(-\frac{1}{2}A_i\sum_a\left(\beta_i^a\right)^2+(B_i+\sqrt{C_i}z)\sum_a\beta_i^a-\lambda\sum_a|\beta_i^a|\right)}\right]_\lambda, \tag{17}$$

where $Dz = dze^{-\frac{1}{2}z^2}/\sqrt{2\pi}$, and the following identity was used:

$$e^{\frac{1}{2}Cx^2} = \int Dz\; e^{\sqrt{C}zx}.$$

Moreover, $A_i = \sum_\mu \tilde{A}_{\mu\to i}$, $B_i = \sum_\mu \tilde{B}_{\mu\to i}$, and $C_i = \sum_\mu \tilde{C}_{\mu\to i}$. All replicas are now factorized, and the average can be taken for each replica independently, allowing passing to the $n\to 0$ limit by analytic continuation from $\mathbb{N}$ to $\mathbb{R}$ with respect to n. For example, the mean of β_i^a is computed as

$$\overline{\beta_i^a} = \left[\int Dz\left(\int d\beta\, e^{\gamma\left(-\frac{1}{2}A_i\beta^2+(B_i+\sqrt{C_i}z)\beta_i-\lambda|\beta|\right)}\right)^n\right]_\lambda^{-1}$$
$$\times\left[\int Dz\int d\beta\;\beta e^{\gamma\left(-\frac{1}{2}A_i\beta^2+(B_i+\sqrt{C_i}z)\beta_i-\lambda|\beta|\right)}\left(\int d\beta\, e^{\gamma\left(-\frac{1}{2}A_i\beta^2+(B_i+\sqrt{C_i}z)\beta_i-\lambda|\beta|\right)}\right)^{n-1}\right]_\lambda$$
$$\xrightarrow{n\to 0}\left[\int Dz\frac{\int d\beta\;\beta e^{\gamma\left(-\frac{1}{2}A_i\beta^2+(B_i+\sqrt{C_i}z)\beta_i-\lambda|\beta|\right)}}{\int d\beta\, e^{\gamma\left(-\frac{1}{2}A_i\beta^2+(B_i+\sqrt{C_i}z)\beta_i-\lambda|\beta|\right)}}\right]_\lambda \xrightarrow{\gamma\to\infty}\left[\int DzS_\lambda(B_i+\sqrt{C_i}z;A_i)\right]_\lambda,$$

where S_λ is the so-called soft thresholding function

$$S_\lambda(x;A) = \frac{1}{A}(x-\lambda\mathrm{sgn}\,(x))\theta(|x|-\lambda),$$

and $\theta(x)$ is the step function that is equal to 1 if $x>0$ and 0 otherwise. Thus, the mean of the basic estimator takes a reasonable form.

In the above average, it is assumed that the coefficients A_i, B_i, C_i remain finite as $\gamma\to\infty$. This is the case if the following holds:

$$\chi_\mu \equiv \gamma V_\mu = \sum_i x_{\mu i}^2\gamma V_i \to O(1),\ (\gamma\to\infty).$$

This scaling is consistent, which can be shown by

$$\chi_i \equiv \gamma V_i = \gamma\left\{\overline{\left(\beta_i^a\right)^2-\beta_i^a\beta_i^b}\right\}$$
$$\xrightarrow{n\to 0}\left[\int Dz\,\gamma\left\{\frac{\int d\beta\,\beta^2 e^{\gamma\left(-\frac{1}{2}A_i\beta^2+(B_i+\sqrt{C_i}z)\beta_i-\lambda|\beta|\right)}}{\int d\beta\, e^{\gamma\left(-\frac{1}{2}A_i\beta^2+(B_i+\sqrt{C_i}z)\beta_i-\lambda|\beta|\right)}}-\left(\frac{\int d\beta\,\beta e^{\gamma\left(-\frac{1}{2}A_i\beta+(B_i+\sqrt{C_i}z)\beta_i-\lambda|\beta|\right)}}{\int d\beta\, e^{\gamma\left(-\frac{1}{2}A_i\beta^2+(B_i+\sqrt{C_i}z)\beta_i-\lambda|\beta|\right)}}\right)^2\right\}\right]_\lambda$$
$$\xrightarrow{\gamma\to\infty}\frac{1}{A_i}\left[\int Dz\theta\left(|B_i+\sqrt{C_i}z|-\lambda\right)\right]_\lambda = \frac{\partial\overline{\beta_i}}{\partial B_i}.$$

In contrast to V_i, the inter-sample fluctuation W_i takes a finite value even as $\gamma\to\infty$. Its explicit form is

$$W_i = \overline{\beta_i^a\beta_i^b}-\overline{\beta_i^a}\,\overline{\beta_i^b}$$
$$\xrightarrow{n\to 0,\ \gamma\to\infty}\left[\int DzS_\lambda^2\left(B_i+\sqrt{C_i}z;A_i\right)\right]_\lambda-\left(\left[\int DzS_\lambda\left(B_i+\sqrt{C_i}z;A_i\right)\right]_\lambda\right)^2.$$

It follows that any moment of the basic estimator can be computed from (17), once the coefficients $\{A_i,B_i,C_i\}_{i=1}^N$ are correctly estimated. Hence, the next task is to derive a set of self-consistent equations for the coefficients and to construct an algorithm for solving it.

3.2. Self-consistent equations and a message passing algorithm

Using (12), we obtain

$$\phi_{i\to\mu}(\boldsymbol{\beta}_i) \propto \Psi\left(\boldsymbol{\beta}_i\right)\prod_{\nu(\neq\mu)}\widetilde{\phi}_{\nu\to i}(\boldsymbol{\beta}_i)$$
$$\approx\left[\int Dz\; e^{\gamma\left(-\frac{1}{2}A_{i\to\mu}\sum_a\left(\beta_i^a\right)^2+(B_{i\to\mu}+\sqrt{C_{i\to\mu}}z)\sum_a\beta_i^a-\lambda\sum_a|\beta_i^a|\right)}\right]_\lambda,$$

where $A_{i\to\mu}=\sum_{\nu(\neq\mu)}\tilde{A}_{\nu\to i}$, $B_{i\to\mu}=\sum_{\nu(\neq\mu)}\tilde{B}_{\nu\to i}$, and $C_{i\to\mu}=\sum_{\nu(\neq\mu)}\tilde{C}_{\nu\to i}$. Inserting this into (11), we can derive a set of self-consistent equations determining all the cavity coefficients $\{A_{i\to\mu},B_{i\to\mu},C_{i\to\mu},\tilde{A}_{\mu\to i},\tilde{B}_{\mu\to i},\tilde{C}_{\mu\to i}\}_{i,\mu}$. This procedure suggests an iterative

algorithm, conventionally called BP algorithm, which is schematically described as follows:

$$\{\tilde{A}_{\mu\to i}, \tilde{B}_{\mu\to i}, \tilde{C}_{\mu\to i}\}^{(t)} \leftarrow \{\overline{\beta_i}^{\setminus\mu}, V_i, W_i\}^{(t)}, \tag{18a}$$
$$\{A_{i\to\mu}, B_{i\to\mu}, C_{i\to\mu}\}^{(t+1)} \leftarrow \{\tilde{A}_{\mu\to i}, \tilde{B}_{\mu\to i}, \tilde{C}_{\mu\to i}\}^{(t)}, \tag{18b}$$
$$\{\overline{\beta_i}^{\setminus\mu}, V_i, W_i\}^{(t+1)} \leftarrow \{A_{i\to\mu}, B_{i\to\mu}, C_{i\to\mu}\}^{(t+1)}, \tag{18c}$$

where $t = 0, 1, \cdots$, denotes the algorithm time step. If the BP algorithm converges, the full coefficients $\{A_i, B_i, C_i\}_{i=1}^N$ are given from the converged values of $\{\tilde{A}_{\mu\to i}, \tilde{B}_{\mu\to i}, \tilde{C}_{\mu\to i}\}_{i,\mu}$. However, this algorithm is not particularly efficient because its computational cost is $O(NM^2)$.

A more efficient algorithm is derived by approximately rewriting the cavity coefficients and the cavity mean $\overline{\boldsymbol{\beta}}^{\setminus\mu}$ using the full coefficients $\{A_i, B_i, C_i\}_i$. To this end, $\phi_{i\to\mu}(\boldsymbol{\beta}_i)$ is connected with $P_i(\boldsymbol{\beta}_i|D)$ in a perturbative manner. Comparing the coefficients, it follows that the difference between A_i and $A_{i\to\mu}$ is negligibly small, as it is proportional to $x_{\mu i}^2 = O(1/N)$. The same is true between C_i and $C_{i\to\mu}$. Hence, the relevant difference is only $\Delta B_i^{(t)} = B_i^{(t)} - B_{i\to\mu}^{(t)}$ and is expressed as

$$\begin{aligned}\Delta B_i^{(t)} &= \left[\frac{c}{1+c\chi_\mu^{(t-1)}}\right]_c r_{\mu i}^{(t-1)} x_{\mu i} = a_\mu^{(t-1)} x_{\mu i} + \left[\frac{c}{1+c\chi_\mu^{(t-1)}}\right]_c \left(\overline{\beta_i}^{\setminus\mu}\right)^{(t-1)} x_{\mu i}^2 \\ &\approx a_\mu^{(t-1)} x_{\mu i},\end{aligned}$$

where

$$a_\mu^{(t)} \equiv \left[\frac{c}{1+c\chi_\mu^{(t)}}\right]_c \left(y_\mu - \sum_j x_{\mu j}\left(\overline{\beta_j}^{\setminus\mu}\right)^{(t)}\right) = \left[\frac{c}{1+c\chi_\mu^{(t)}}\right]_c \left(r_{\mu i}^{(t)} - x_{\mu i}\left(\overline{\beta_i}^{\setminus\mu}\right)^{(t)}\right). \tag{19}$$

Accordingly, the difference between $\overline{\beta_i}^{\setminus\mu}$ and $\overline{\beta_i}$ is computed as

$$\left(\overline{\beta_i}^{\setminus\mu}\right)^{(t)} \approx \overline{\beta_i}^{(t)} - \frac{\partial\overline{\beta_i}^{(t)}}{\partial B_i^{(t)}}\Delta B_i^{(t)} = \overline{\beta_i}^{(t)} - \chi_i^{(t)} a_\mu^{(t-1)} x_{\mu i}, \tag{20}$$

Inserting (20) into (19) yields

$$a_\mu^{(t)} = \left[\frac{c}{1+c\chi_\mu^{(t)}}\right]_c \left(y_\mu - \sum_j x_{\mu j}\overline{\beta_j}^{(t)} + \chi_\mu^{(t)} a_\mu^{(t-1)}\right),$$

where the last term is interpreted as the Onsager reaction term in physics. Rewriting $r_{\mu i}$ using a_μ in the right-hand sides of (16) and collecting several factors, a simplified message

passing algorithm corresponding to (18) is obtained as follows:

$$\chi_\mu^{(t)} = \sum_i x_{\mu i}^2 \chi_i^{(t)}, \tag{21a}$$
$$W_\mu^{(t)} = \sum_i x_{\mu i}^2 W_i^{(t)}, \tag{21b}$$
$$\left(f_{1\mu}^{(t)}, f_{2\mu}^{(t)}\right) = \left(\left[\frac{c}{1+c\chi_\mu^{(t)}}\right]_c, \left[\left(\frac{c}{1+c\chi_\mu^{(t)}}\right)^2\right]_c\right), \tag{21c}$$
$$a_\mu^{(t)} = f_{1\mu}^{(t)}\left(y_\mu - \sum_j x_{\mu j}\overline{\beta_j}^{(t)} + \chi_\mu^{(t)} a_\mu^{(t-1)}\right), \tag{21d}$$
$$A_i^{(t+1)} = \sum_\mu x_{\mu i}^2 f_{1\mu}^{(t)}, \tag{21e}$$
$$B_i^{(t+1)} = \sum_\mu x_{\mu i} a_\mu^{(t)} + \left(\sum_\mu x_{\mu i}^2 f_{1\mu}^{(t)}\right)\overline{\beta_i}^{(t)}, \tag{21f}$$
$$C_i^{(t+1)} = \sum_\mu x_{\mu i}^2 \left\{f_{2\mu}^{(t)} W_\mu^{(t)} + \left(f_{2\mu}^{(t)} - \left(f_{1\mu}^{(t)}\right)^2\right)\left(\frac{a_\mu^{(t)}}{f_{1\mu}^{(t)}}\right)^2\right\}, \tag{21g}$$
$$\overline{\beta_i}^{(t+1)} = \left[\int Dz S_\lambda\left(B_i^{(t+1)} + \sqrt{C_i^{(t+1)}}z; A_i^{(t+1)}\right)\right]_\lambda, \tag{21h}$$
$$\chi_i^{(t+1)} = \frac{1}{A_i^{(t+1)}}\left[\int Dz\ \theta\left(\left|B_i^{(t+1)} + \sqrt{C_i^{(t+1)}}z\right| - \lambda\right)\right]_\lambda, \tag{21i}$$
$$W_i^{(t+1)} = \left[\int Dz S_\lambda^2\left(B_i^{(t+1)} + \sqrt{C_i^{(t+1)}}z; A_i^{(t+1)}\right)\right]_\lambda - \left(\overline{\beta_i}^{(t+1)}\right)^2. \tag{21j}$$

We call the algorithm (21) AMPR (Approximate Message Passing with Resampling) because it can be regarded as an extension of the AMP in the usual Lasso to the resampling case. The computational cost is $O(NM)$ per iteration and is significantly reduced compared with the BP algorithm. This yields the main result of this study.

There is an ambiguity in the initial condition for AMPR. Here we assume that we are given an initial estimate $\{\overline{\beta_i}^{(0)}, \chi_i^{(0)}, W_i^{(0)}\}_i$ and conduct the iteration based on (21) until convergence. Still, there is an ambiguity in computing $a_\mu^{(0)}$, due to the presence of $a_\mu^{(-1)}$ in (21d). To resolve this, we assume $a_\mu^{(-1)} = 0$, yielding

$$a_\mu^{(0)} = \left[\frac{c}{1+c\chi_\mu^{(0)}}\right]_c \left(y_\mu - \sum_j x_{\mu j}\overline{\beta_j}^{(0)}\right). \tag{22}$$

These completely determine the initial condition.

As explained at the end of Section 3.1, the convergent solution of AMPR, $\{A_i^*, B_i^*, C_i^*\}_i$, enables the computation of any moment of the basic estimator as follows:

$$[\langle\beta_i\rangle^r]_{\mathbf{c},\boldsymbol{\lambda}} = \lim_{n\to 0}\overline{\prod_{a=1}^r \beta_i^a} = \left[\int Dz S_\lambda^r\left(B_i^* + \sqrt{C_i^*}z; A_i^*\right)\right]_\lambda,$$

which indicates that the marginal distribution of the basic estimator is obtained as

$$P(\beta_i) = \left[\int Dz \ \delta \left(\beta_i - S_\lambda \left(B_i^* + \sqrt{C_i^*} z; A_i^* \right) \right) \right]_\lambda .$$

This yields the positive probability, which is important for variable selection techniques, as follows:

$$\Pi_i \equiv \text{Prob}\left(|\hat{\beta}_i| \neq 0 \right) = \left[\int Dz \ \theta \left(\left| B_i^* + \sqrt{C_i^*} z \right| - \lambda \right) \right]_\lambda . \tag{23}$$

Applications using these relations are provided in Section 4.

3.3. State evolution for AMPR

A benefit of the AMP type algorithms is that it is possible to track the macroscopic dynamical behavior of the algorithm. This can be done by using the so-called state evolution (SE) equations. Here we derive the SE equations associated with AMPR. The derivation relies on the i.i.d. assumption of the covariates, and hence we assume each $x_{\mu i}$ is i.i.d. from the zero-mean Gaussian as

$$x_{\mu i} \sim \mathcal{N}(0, \sigma_x^2). \tag{24}$$

The variance value is arbitrary in general, but for notational simplicity it is chosen as $\sigma_x^2 = 1/N$ in this section. Furthermore, we assume the data is generated from the following linear process:

$$\boldsymbol{y} = X\boldsymbol{\beta}_0 + \boldsymbol{\xi},$$

where $\boldsymbol{\xi}$ is the noise vector whose component is i.i.d. from $\mathcal{N}(0, \sigma_\xi^2)$ and $\boldsymbol{\beta}_0$ is the true parameters whose component is also i.i.d. from a certain distribution $P_{\beta_0}(\cdot)$.

Under the assumption (24) with $\sigma_x^2 = 1/N$, the intra- and inter-sample variances can be simplified as

$$\chi_\mu = \sum_i x_{\mu i}^2 \chi_i \approx \sum_i \mathrm{E}\left[x_{\mu i}^2 \chi_i \right] \approx \frac{1}{N} \sum_i \chi_i \equiv \tilde{\chi}, \tag{25}$$

$$W_\mu = \sum_i x_{\mu i}^2 W_i \approx \sum_i \mathrm{E}\left[x_{\mu i}^2 W_i \right] \approx \frac{1}{N} \sum_i W_i \equiv \tilde{W}, \tag{26}$$

where we have neglected the correlations between $x_{\mu i}$ and the variances[2]. Accordingly, many quantities appearing in (21) become independent of the subscripts μ and i. Terms retaining the dependence are only the linear terms with respect to $x_{\mu i}$ such as $\overline{\beta}_i, B_i$ and a_μ. To derive the SE equations, we need to handle those terms.

2. Remembering the discussion in Section 3.2, these variances are actually the ones computed in the absence of the μ-th potential function, $\chi_i^{\setminus \mu}$ and $W_i^{\setminus \mu}$, and hence this neglect can be justified. The same discussion can be applied to the following.

We start from the following form of $B_i^{(t+1)}$:

$$B_i^{(t+1)} = \sum_\nu \left[\frac{c}{1 + c\chi_\nu} \right]_c r_{\nu i}^{(t)} x_{\nu i} \approx f_1^{(t)} \sum_\nu x_{\nu i} \left(y_\nu - \sum_{j(\neq i)} x_{\nu j} \left(\overline{\beta}_j^{\setminus \nu} \right)^{(t)} \right). \tag{27}$$

The righthand side is the sum of a large number of random variables, and hence we can treat it as a Gaussian variable with appropriate mean and variance. The mean is

$$\begin{aligned} &\mathrm{E}\left[f_1^t \sum_\nu x_{\nu i} \left(y_\nu - \sum_{j(\neq i)} x_{\nu j} \left(\overline{\beta}_j^{\setminus \nu} \right)^{(t)} \right) \right] \\ &= f_1^t \sum_\nu \left(\mathrm{E}\left[x_{\nu i}^2 \right] \beta_{0i} + \sum_{j(\neq i)} \mathrm{E}\left[x_{\nu i} x_{\nu j} \right] \left(\beta_{0j} - \left(\overline{\beta}_j^{\setminus \mu} \right)^{(t)} \right) + \mathrm{E}\left[x_{\nu i} \xi_\nu \right] \right) \\ &= f_1^t \sum_\nu \frac{1}{N} \beta_{0i} = \alpha f_1^t \beta_{0i}, \end{aligned} \tag{28}$$

where $\alpha = M/N$ is the ratio of the dataset size to the dimensionality. In the same way, the variance becomes

$$\mathrm{V}\left[f_1^t \sum_\nu x_{\nu i} \left(y_\nu - \sum_{j(\neq i)} x_{\nu i} \left(\overline{\beta}_j^{\setminus \nu} \right)^{(t)} \right) \right] \approx \alpha \left(f_1^t \right)^2 \left(\mathrm{MSE}^{(t)} + \sigma_\xi^2 \right) \equiv v_0^{(t+1)}, \tag{29}$$

where $\mathrm{MSE}^{(t)}$ denotes the mean-squared error (MSE) between the true and averaged parameters:

$$\mathrm{MSE}^{(t)} \equiv \frac{1}{N} \sum_{i=1}^N \left(\beta_{0i} - \overline{\beta}_i^{(t)} \right)^2 \approx \frac{1}{N} \sum_{i=1}^N \left(\beta_{0i} - \left(\overline{\beta}_i^{\setminus \mu} \right)^{(t)} \right)^2 . \tag{30}$$

Hence, we may write

$$B_i^{(t+1)} = \alpha f_1^t \beta_{0i} + \sqrt{v_0^{(t+1)}} u_i, \tag{31}$$

where $u_i \sim \mathcal{N}(0, 1)$. Besides, in the computation of $C_i^{(t+1)}$, we have

$$\sum_\nu x_{\nu i}^2 f_{2\nu}^{(t)} W_\nu^{(t)} \approx \alpha f_2^{(t)} \tilde{W}^{(t)}, \tag{32}$$

$$\sum_\nu x_{\nu i}^2 \left(r_{\nu i}^{(t)} \right)^2 \approx \alpha \left(\mathrm{MSE}^{(t)} + \sigma_\xi^2 \right). \tag{33}$$

This yields

$$C_i^{(t+1)} \approx C^{(t+1)} = \alpha f_2^{(t)} \tilde{W}^{(t)} + \alpha \left(f_2^{(t)} - \left(f_1^{(t)} \right)^2 \right) \left(\mathrm{MSE}^{(t)} + \sigma_\xi^2 \right). \tag{34}$$

In the same level of approximation, we get $A_i^{(t+1)} \approx A^{(t+1)} = \alpha f_1^{(t)}$.

To derive a closed set of equations, we have to compute $\tilde{\chi}^{(t+1)}, \tilde{W}^{(t+1)}$ and $\mathrm{MSE}^{(t+1)}$ from $\{A^{(t+1)}, \{B_i^{(t+1)}\}_i, C^{(t+1)}\}$. As an example, we show the derivation of $\tilde{\chi}^{(t+1)}$ from (21i) in the following:

$$\tilde{\chi}^{(t+1)} = \frac{1}{N}\sum_{i=1}^{N}\chi_i^{(t+1)} \approx \frac{1}{N}\sum_{i=1}^{N}\frac{1}{A^{(t+1)}}\left[\int Dz\ \theta\left(\left|B_i^{(t+1)} + \sqrt{C^{(t+1)}}z\right| - \lambda\right)\right]_\lambda$$
$$\approx \frac{1}{A^{(t+1)}}\int d\beta P_{\beta_0}(\beta)\int Du\left[\int Dz\ \theta\left(\left|\alpha f_1^{(t)}\beta + \sqrt{v_0^{(t+1)}}u + \sqrt{C^{(t+1)}}z\right| - \lambda\right)\right]_\lambda, \quad (35)$$

where we have applied the law of large numbers. The other quantities $\tilde{W}, \mathrm{MSE}$ are computed in the same manner. Overall, we reach the following set of equations:

$$\left(f_1^{(t)}, f_2^{(t)}\right) = \left(\left[\frac{c}{1+c\tilde{\chi}^{(t)}}\right]_c, \left[\left(\frac{c}{1+c\tilde{\chi}^{(t)}}\right)^2\right]_c\right), \quad (36a)$$

$$A^{(t+1)} = \alpha f_1^{(t)}, \quad (36b)$$

$$C^{(t+1)} = \alpha f_2^{(t)}\tilde{W}^{(t)} + \alpha\left(f_2^{(t)} - \left(f_1^{(t)}\right)^2\right)\left(\mathrm{MSE}^{(t)} + \sigma_\xi^2\right), \quad (36c)$$

$$v_0^{(t+1)} = \alpha\left(f_1^{(t)}\right)^2\left(\mathrm{MSE}^{(t)} + \sigma_\xi^2\right), \quad (36d)$$

$$\tilde{\chi}^{(t+1)} = \frac{1}{A^{(t+1)}}\int d\beta P_{\beta_0}(\beta)\int Du$$
$$\times\left[\int Dz\ \theta\left(\left|A^{(t+1)}\beta + \sqrt{v_0^{(t+1)}}u + \sqrt{C^{(t+1)}}z\right| - \lambda\right)\right]_\lambda, \quad (36e)$$

$$\tilde{W}^{(t+1)} = \int d\beta P_{\beta_0}(\beta)\int Du\left\{\left[\int DzS_\lambda^2\left(A^{(t+1)}\beta + \sqrt{v_0^{(t+1)}}u + \sqrt{C^{(t+1)}}z; A^{(t+1)}\right)\right]_\lambda\right.$$
$$\left. - \left[\int DzS_\lambda\left(A^{(t+1)}\beta + \sqrt{v_0^{(t+1)}}u + \sqrt{C^{(t+1)}}z; A^{(t+1)}\right)\right]_\lambda^2\right\}, \quad (36f)$$

$$\mathrm{MSE}^{(t+1)} = \int d\beta P_{\beta_0}(\beta)\int Du$$
$$\times\left\{\beta - \left[\int DzS_\lambda\left(A^{(t+1)}\beta + \sqrt{v_0^{(t+1)}}u + \sqrt{C^{(t+1)}}z; A^{(t+1)}\right)\right]_\lambda\right\}^2. \quad (36g)$$

Given an initial condition $\{\tilde{\chi}^{(0)}, \tilde{W}^{(0)}, \mathrm{MSE}^{(0)}\}$, we can track the dynamical evolution of those quantities according to (36). This is the SE equations for AMPR.

A direct consequence of the SE equations is the convergence property of AMPR: Its convergence depends on neither the dataset size M nor the model dimensionality N. Hence, we can assume the iteration steps required for convergence is $O(1)$ and the total computational cost of AMPR is thus guaranteed to be $O(NM)$. This reinforces the superiority of the present approach.

We, however, warn that the discussion based on the SE equations strongly relies on the i.i.d. assumption among covariates, which is not necessarily satisfied in real-world datasets. For covariates with non-trivial correlations and heterogeneity, it is known that

AMP type algorithms tend to show slow convergence, or even not to converge in particular cases (Caltagirone et al., 2014). A common prescription to overcome this difficulty is to introduce a damping factor in the update of the messages (Rangan et al., 2014), which is also employed in our implementation (Obuchi, 2018). In Sections 4.1.5 and 4.2, we see how this prescription works for datasets with nontrivial covariates in numerical simulations.

4. Numerical experiments

In this section, the accuracy and the computational time of the proposed semi-analytic method based on AMPR is examined by a comparison with direct numerical resampling. We also check how nontrivial correlations among covariates affect the performance of AMPR. Both simulated and real-world datasets (from UCI machine learning repository, Lichman, 2013) are used.

For all experiments involving numerical resampling, *Glmnet* (Friedman et al., 2010), implemented as an *MEX* subroutine in MATLAB®, was employed for solving (9), given a sample $\{\boldsymbol{\lambda}, D_c\}$. Moreover, the proposed AMPR algorithm was implemented as raw code in MATLAB. This is not the most optimized approach because AMPR uses a number of *for* and *while* loops which are slow in MATLAB; hence, the comparison of computational time is not necessarily fair. However, even in this comparison, there is a significant difference in the computational time between the proposed semi-analytic method and the numerical resampling approach. For reference, it should be noted that all experiments below were conducted in a single thread on a single CPU of Intel(R) Xeon(R) E5-2630 v3 2.4GHz.

For actual computations, the distribution $P(\boldsymbol{\lambda})$ should be specified. In SS, the following distribution is used (Meinshausen and Bühlmann, 2010):

$$P(\boldsymbol{\lambda}) = \prod_{i=1}^{N}\left\{p_w\delta\left(\lambda_i - \lambda/w\right) + (1-p_w)\delta\left(\lambda_i - \lambda\right)\right\},$$

with $0 < w \leq 1$ and $0 < p_w < 1$. The case of the non-random penalty coefficient, in which Bolasso is included (Bach, 2008), is recovered at $w = 1$, irrespective of the value of p_w. This distribution is adopted below.

4.1. Simulated dataset

Here, simulated datasets are treated. The data is supposed to be generated from the following linear model:

$$\boldsymbol{y} = X\boldsymbol{\beta}_0 + \boldsymbol{\xi},$$

where each component of the design matrix $X = (\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N)$ is i.i.d. from $\mathcal{N}\left(0, N^{-1}\right)$, and $\boldsymbol{\xi}$ is the noise vector, whose component is i.i.d. from $\mathcal{N}(0, \sigma_\xi^2)$. The ratio of the dataset size M to the model dimensionality N is denoted as $\alpha \equiv M/N$ hereafter. These settings are identical to the ones assumed in Section 3.3. The true signal $\boldsymbol{\beta}_0 \in \mathbb{R}^N$ is assumed to be $K_0(= N\rho_0)$-sparse vector, and the non-zero components are i.i.d. from $\mathcal{N}(0, 1/\rho_0)$, setting the power of the signal unity. The index set of non-zero components is denoted by $S_0 = \{i | |\beta_{0i}| \neq 0\}$ and is called true support. Any estimator of the true support is simply called support and is hereafter denoted by S.

For simplicity, the number of resampling times is fixed at $N_{\mathrm{res}} = 1000$ in the experiments with numerical resampling.

4.1.1. Accuracy of the semi-analytic method

Let us first check the consistency between the results of our semi-analytic method and of the direct numerical resampling.

Figure 1 shows the plots of the experimental values of the quantities $\{\overline{\beta_i}, W_i, \Pi_i\}_i$ against their semi-analytic counterparts for the non-random penalty case $w = 1$ with the bootstrap resampling $\tau = 1$, which is the situation considered in Bolasso. The same plots in the SS situation, $w = 0.5(< 1)$, $p_w = 0.5$, and $\tau = 0.5$, are shown in Figure 2. Other detailed parameters are provided in the captions. These results show that the proposed semi-analytic

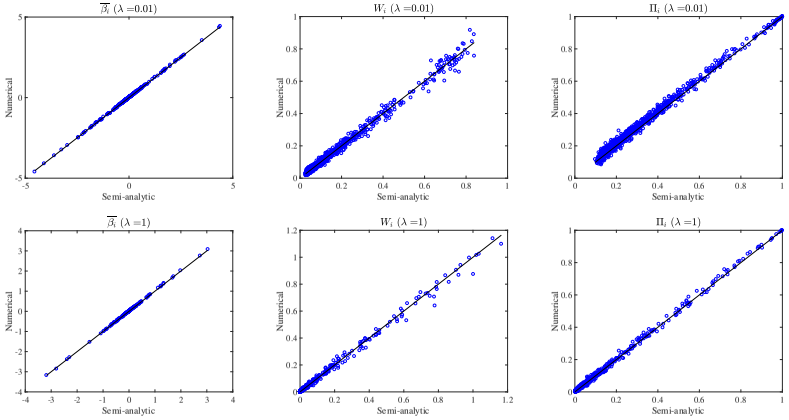


Figure 1: Experimental values of $\overline{\beta_i}$ (left), W_i (middle), and Π_i (right) are plotted against those computed by the semi-analytic method in the non-random penalty $w = 1$ and $\tau = 1$ case. The upper panels are for $\lambda = 0.01$ and the lower are for $\lambda = 1$. The other parameters are set to be $(N, \alpha, \rho_0, \sigma_\xi^2) = (1000, 0.5, 0.2, 0.01)$.

method reproduces the numerical results fairly accurately. As far as it was examined, results of similar accuracy were obtained for a very wide range of parameters. These validate the proposed semi-analytic method.

In the above experiments using Glmnet, we set a threshold ϵ to judge the algorithm convergence as $\epsilon = 10^{-10}$, which is rather tighter than the default value. This is necessary for examining consistency with the proposed semi-analytic method. For example, for $\lambda = 0.01$ (the upper panels in Figures 1 and 2), a systematic deviation from the proposed semi-analytic method ($\overline{\beta}_i$ tends to be underestimated) emerges at the default value $\epsilon = 10^{-7}$.

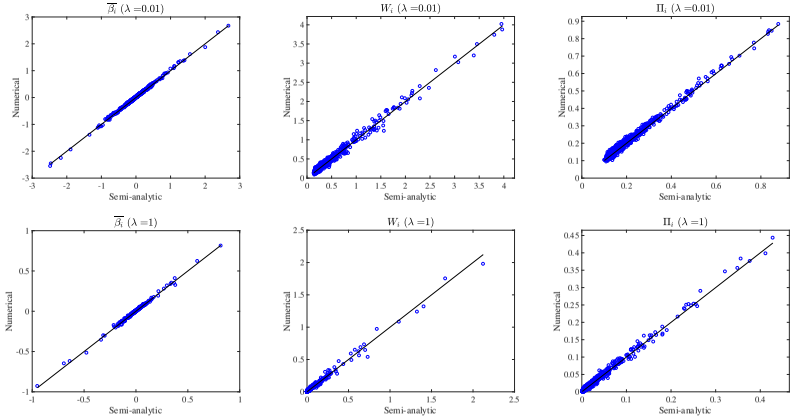


Figure 2: Same plots as in Figure 1 in the random penalty case with $w = 0.5$, $p_w = 0.5$, and $\tau = 0.5$. The parameters of each panel are identical to the corresponding parameters in Figure 1. In comparison to Figure 1, $|\overline{\beta}_i|$ and Π_i tend to be smaller, whereas W_i tends to be larger. This is probably due to the additional stochastic variation coming from the randomization on the penalty coefficient and the difference between $\tau = 1$ and $1/2$.

This implies that a rather tight threshold is required for microscopic quantities such as $\overline{\beta_i}$ and W_i. Unless explicitly mentioned, this value $\epsilon = 10^{-10}$ is used below.

4.1.2. Comparison with state evolution

To examine the convergence properties of AMPR, we next see the dynamical behavior of the macroscopic quantities $(\tilde{\chi}^{(t)}, \tilde{W}^{(t)}, \mathrm{MSE}^{(t)})$ as the algorithm step t proceeds, in comparison with the SE equations (36). Figure 3 shows the plots of them against t for different parameters. In all the cases, these macroscopic quantities rapidly take stable values and

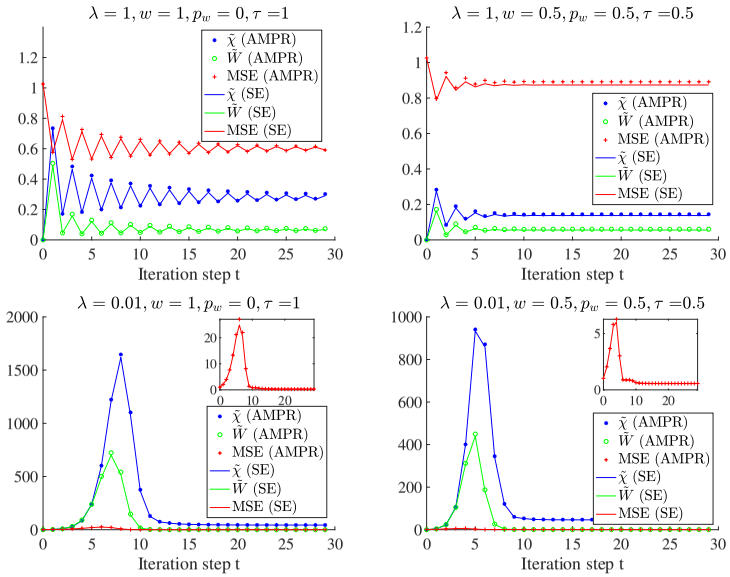


Figure 3: Dynamical behavior of macroscopic parameters $(\tilde{\chi}^{(t)}, \tilde{W}^{(t)}, \mathrm{MSE}^{(t)})$ initialized at $(\tilde{\chi}^{(0)}, \tilde{W}^{(0)}) = (0, 0)$ and $\mathrm{MSE}^{(0)} \approx 1$. The result computed by SE is denoted by lines while the one by AMPR is represented by markers, and the agreement is fairly good. (Left) The non-randomized penalty case ($w = 1, p_w = 0, \tau = 1$). (Right) The randomized penalty case ($w = p_w = \tau = 1/2$). The upper panels are for $\lambda = 1$, while the lower ones are of $\lambda = 0.01$ for which insets are given to show the MSE values in visible scales. The AMPR result is obtained at $N = 20000$. The other parameters are fixed at $(\alpha, \rho_0, \sigma_\xi^2) = (0.5, 0.2, 0.01)$.

the agreement between the AMPR and SE results is excellent. This demonstrates the fast convergence of AMPR, which does not depend on both N and M. To see a good agreement

with the SE result by just one sample of $\{\boldsymbol{\beta}_0, X, \boldsymbol{\xi}\}$, the model dimensionality N is chosen as a rather large value of $N = 20000$ in the AMPR experiment. Although in Figure 3 the initial condition is fixed at $\boldsymbol{\chi}^{(0)} = \boldsymbol{W}^{(0)} = \overline{\boldsymbol{\beta}}^{(0)} = \boldsymbol{0}$ corresponding to $(\tilde{\chi}^{(0)}, \tilde{W}^{(0)}) = (0, 0)$ and $\mathrm{MSE}^{(0)} \approx 1$, we also examined several other initial conditions and confirmed the good agreement between the AMPR and SE results in all the cases.

4.1.3. Application to Bolasso

Bolasso is a variable selection method utilizing the positive probability (23) evaluated by the bootstrap resampling $\tau = 1$ with no penalty randomization $w = 1$. Its soft version, abbreviated as Bolasso-S (Bach, 2008), selects variables with $\Pi_i \geq 0.9$ as active variables and other variables are rejected from the support. We here adopt this manner and see the performance of AMPR used for implementing this.

The left panel of Figure 4 shows the plot of the true positive ratio (TP) against the false positive ratio (FP) as the values of α change. Bolasso requires scaling the regularization parameter as $\lambda \propto \sqrt{\alpha}$, and here, it is set to $\lambda = (1/2)\sqrt{\alpha}$. The other parameters are fixed at $(N, \rho_0, \sigma_\xi^2) = (1000, 0.2, 0.01)$. As was theoretically shown (Bach, 2008), TP converges to unity, whereas FP tends to zero in this setup. This demonstrates that the

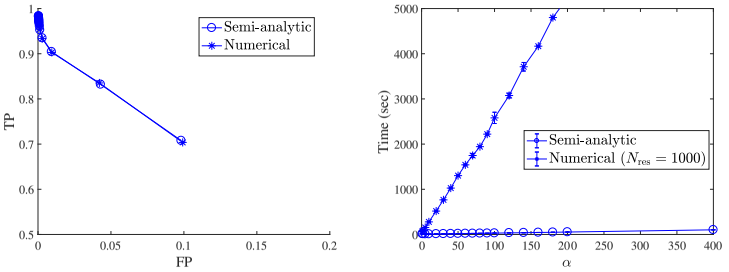


Figure 4: Bolasso experiment at $(N, \rho_0, \sigma_\xi^2) = (1000, 0.2, 0.01)$. The semi-analytic result (circle) is plotted with the numerical resampling result (asterisk). (Left) TP is plotted against FP as α increases: TP approaches unity, whereas FP tends to zero. The semi-analytic result completely overlaps that of the numerical resampling. Error bars are omitted for clarity. (Right) Computational time plotted against α. The observed large difference is attributed to the numerical resampling cost.

model consistency in the variable selection context is recovered. The contribution of this study is a significant reduction of the computational time: In the right panel, the actual computational time is compared between the semi-analytic method using AMPR and the direct numerical resampling, by plotting it against α. The computational costs of AMPR and Glmnet are both scaled as $O(NM)$ and thus should be scaled linearly with respect to α for fixed N. Figure 4 clearly shows this linearity. The significant difference in compu-

tational time is fully attributed to the numerical resampling cost. This demonstrates the efficiency of AMPR. It should be noted that the average over 10 different samples of D is taken in Figure 4 to obtain smooth curves and error bars.

4.1.4. Application to stability selection

SS is another variable selection method utilizing positive probability. The difference from Bolasso is the presence of the penalty coefficient randomization ($w < 1$) and that τ is set to 0.5. These introduce a further stochastic variation in the method, and consequently they tend to show a clearer discrimination in the positive probabilities between the variables in and outside the true support. AMPR can easily implement this, and here, it is compared with numerical resampling.

Figure 5 shows the plots of the positive probability values against λ, the so-called stability path (Meinshausen and Bühlmann, 2010), and the computational time for obtaining the stability path against the covariate dimensionality N. The other parameters are fixed at $(\alpha, \rho_0, \sigma_\xi^2, w, p_w) = (2, 0.2, 0.01, 0.5, 0.5)$. Drawing all stability paths $\{\Pi_i\}_i$ leads to an

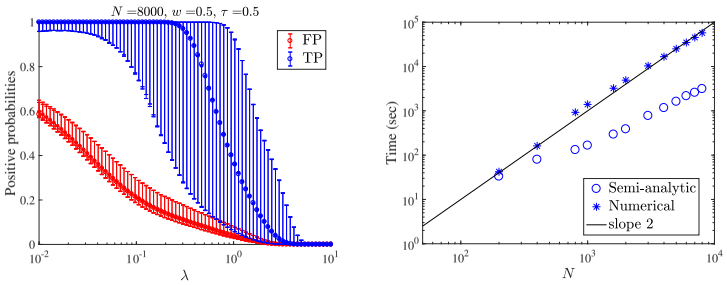


Figure 5: SS experiment at $(\alpha, \rho_0, \sigma_\xi^2, w, p_w) = (2, 0.2, 0.01, 0.5, 0.5)$. The semi-analytic result (circle) is plotted with the numerical resampling result (asterisk). (Left) Plots of the medians (points) and q-percentiles (bars) of $\{\Pi_i\}_{i\in S_0}$ (TP) and $\{\Pi_i\}_{i\notin S_0}$ (FP) with $q = 16$ and 84 against λ for $N = 8000$. The semi-analytic result well overlaps that of the numerical resampling. There is a clear gap between TP and FP, suggesting that an accurate variable selection is possible. (Right) Computational time plotted against N on double-logarithmic scale. The dominant reason for the difference is again the numerical resampling cost.

unclear plot; thus, only the median and q-percentiles of $\{\Pi_i\}_{i\in S_0}$ and those of $\{\Pi_i\}_{i\notin S_0}$ are plotted, which are denoted by TP and FP in the left panel of Figure 5, respectively. The medians are represented by points, and the percentiles of $q = 16$ and 84, approximately corresponding to one-sigma points in normal distributions, are represented by bars. There is a clear gap between TP and FP, suggesting that an accurate variable selection is possible by setting an appropriate threshold of probability value, as discussed by Meinshausen and

Bühlmann (2010). The right panel shows a plot of the computational time by AMPR and by the numerical resampling. Both are supposed to be scaled as $O(NM = N^2)$. Although such a scaling is not observed in the AMPR result, it is expected to appear for larger N. Again, a significant difference in computational time is present between the semi-analytic and the numerical resampling methods, demonstrating the effectiveness of the proposed method.

4.1.5. Correlated covariates

In the derivation of AMPR and the associated SE equations, the weakness of correlations between covariates is assumed, but this assumption does not necessarily hold in realistic situations. Hence, it is important to check the accuracy of results obtained by AMPR for datasets with correlated covariates. Here we numerically examine this point.

To introduce correlations into the simulated dataset described above in a systematic way, we generate our covariates $\{\boldsymbol{x}_i\}_{i=1}^N$ in the following manner: As a common component we first generate a vector $\boldsymbol{x}^{\rm com} \in \mathbb{R}^M$ each component of which is i.i.d. from $\mathcal{N}(0, 1/N)$; choose a number $0 \le r^{\rm com} < 1$ controlling the ratio of the common component and generate a binary vector $\boldsymbol{m}_i$ each component of which independently takes 1 with probability $r^{\rm com}$; take another vector $\tilde{\boldsymbol{x}}_i \in \mathbb{R}^M$ each component of which is i.i.d. from $\mathcal{N}(0, 1/N)$, and generate a covariate vector $\boldsymbol{x}_i$ as a linear combination between $\boldsymbol{x}^{\rm com}$ and $\tilde{\boldsymbol{x}}_i$ with using $\boldsymbol{m}_i$ as a mask. These operations can be summarized in the following equation:

$$\boldsymbol{x}_i = \boldsymbol{x}^{\rm com} \circ \boldsymbol{m}_i(r^{\rm com}) + \tilde{\boldsymbol{x}}_i \circ (1 - \boldsymbol{m}_i(r^{\rm com})), \tag{37}$$

where $\circ$ denotes Hadamard (component-wise) product. The covariates' correlations monotonically increase as $r^{\rm com}$ grows. To quantify this, we compute an overlap between the covariates, $\mathrm{overlap}_{ij} = \boldsymbol{x}_i^\top \boldsymbol{x}_j / \left(\sqrt{\boldsymbol{x}_i^\top \boldsymbol{x}_i}\sqrt{\boldsymbol{x}_j^\top \boldsymbol{x}_j}\right)$, and plot its mean value over i and j ($\neq i$) in Figure 6. Keeping in mind this quantitative information, we check the accuracy of AMPR below.

To directly see the accuracy of AMPR for correlated covariates, we plot the experimental values of the quantities $\{\overline{\beta}_i, W_i, \Pi_i\}_i$ against those of AMPR in Figure 7 for two different values of the common component ratio, $r^{\rm com} = 0.4$ and $r^{\rm com} = 0.8$. For $r^{\rm com} = 0.8$, the AMPR result shows a clear deviation from the experimental one, while for $r^{\rm com} = 0.4$ it still exhibits a good agreement with the experiment. Although Figure 7 is the specific result to the non-random penalty case ($w = 1, \tau = 1$) with parameters $(N, \alpha, \rho_0, \sigma_\xi^2, \lambda) = (1000, 0.5, 0.2, 0.01, 1)$, we have tested several different cases and observed similar tendency. To capture more global and quantitative information, we introduce a normalized MSE of $\overline{\beta}$ between the experimental $\overline{\beta}^{\rm exp}$ and AMPR $\overline{\beta}^{\rm AMPR}$ values as

$$\frac{\sum_{i=1}^N \left(\overline{\beta}_i^{\rm exp} - \overline{\beta}_i^{\rm AMPR}\right)^2}{\sum_{i=1}^N \left(\overline{\beta}_i^{\rm AMPR}\right)^2}. \tag{38}$$

The counterparts for W and Π are defined in the same way. Plots of the normalized MSEs for these quantities against $r^{\rm com}$ are given in Figure 8. Here the results for both the randomized ($w = p_w = \tau = 0.5$) and non-randomized ($w = \tau = 1, p_w = 0$) penalty

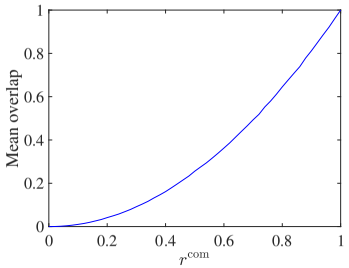


Figure 6: Plot of mean overlap of covariates against the common component ratio r^{com}. This is a result of a single numerical simulation at $(N, \alpha) = (1000, 0.5)$, but the invariability of the result against changing the parameters has been also checked.

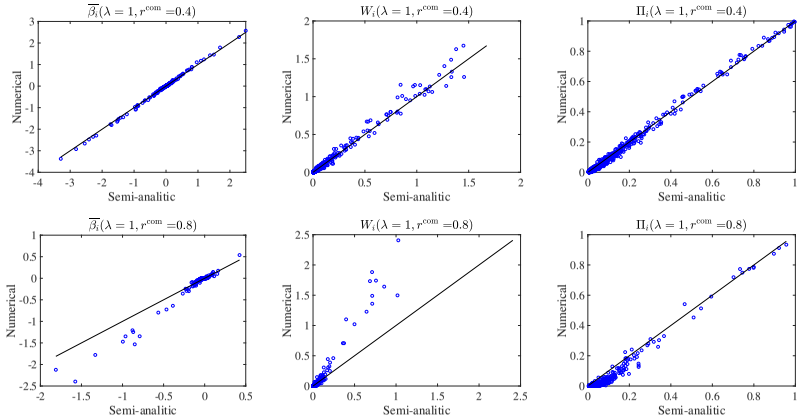


Figure 7: Experimental values of $\overline{\beta_i}$ (left), W_i (middle), and Π_i (right) are plotted against those computed by AMPR for correlated covariates with $r^{\mathrm{com}} = 0.4$ (upper) and $r^{\mathrm{com}} = 0.8$ (lower). For the lower panels, there exists a systematic deviation in the AMPR result. Here the non-random penalty case $(w = 1, \tau = 1)$ is treated and the other parameters are $(N, \alpha, \rho_0, \sigma_\xi^2, \lambda) = (1000, 0.5, 0.2, 0.01, 1)$.

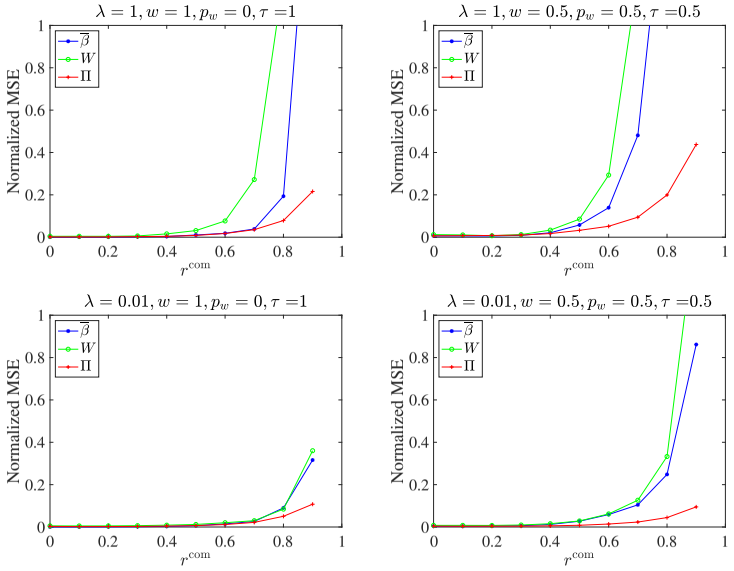


Figure 8: Plots of normalized MSEs of $\overline{\beta}, W, \Pi$ against r^{com} of the non-randomized (left, $w = 1, p_w = 0, \tau = 1$) and randomized (right, $w = p_w = \tau = 1/2$) penalty cases. The upper panels are for $\lambda = 1$ while the lower ones are of $\lambda = 0.01$. The other parameters are $(N, \alpha, \rho_0, \sigma_\xi^2) = (1000, 0.5, 0.2, 0.01)$.

cases are presented, with two different values of λ. This suggests that AMPR is practically reliable up to a certain level of correlations: For example if $r^{\mathrm{com}} \leq 0.6$, the normalized MSE of $\overline{\beta}$ is well suppressed and is commonly less than 0.2. According to Figure 6, $r^{\mathrm{com}} = 0.6$ corresponds to the mean overlap value of about 0.36 which is not small. This speaks for the effectiveness of AMPR even for real-world datasets, as long as the correlations are not too large.

The above discussion clarifies the accuracy of AMPR for correlated covariates, but a more crucial issue is in its convergent property. For the non-correlated case of $r^{\mathrm{com}} = 0$, AMPR converges very rapidly as shown in Figure 3, but for correlated cases it tends to badly converge and can even diverge. A common way to overcome this is to introduce a damping factor γ in the update (Caltagirone et al., 2014; Rangan et al., 2014), and we adopted this in the above experiments. The update with the damping factor can be symbolized as

$$\boldsymbol{\theta}^{(t+1)} = (1-\gamma)\boldsymbol{\theta}^{(t)} + \gamma G(\boldsymbol{\theta}^{(t)}), \tag{39}$$

where $\boldsymbol{\theta}$ is a variable summarizing $(\overline{\boldsymbol{\beta}}, \boldsymbol{\chi}, \boldsymbol{W})$ and G represents the AMPR operations. The original message passing update corresponds to $\gamma = 1$. Smaller values of γ are better for the stability of the algorithm but takes a longer time until the convergence. Our experiments on the simulated dataset seemingly show that a smaller value of γ is needed for larger r^{com} and smaller λ. For example for obtaining Figure 7, we needed to set $\gamma < 0.1$ for avoiding divergence. This value is found experimentally, and it is desired to find a more principled way to choose an appropriate value of γ or a more effective way to better control the convergence of AMP type algorithms. Some earlier studies tackled this problem (Caltagirone et al., 2014; Rangan et al., 2014), but the complete understanding of the convergent property is still missing. Investigation along this direction is an important topic but is beyond the purpose of the present paper.

4.2. Real-world dataset

In this subsection, AMPR is applied to a real-world dataset, and the stability path is computed for examining the relevant covariates or variables. The dataset treated here is the wine quality dataset (Lichman, 2013): The data size is $M = 4898$ (only white wine is treated), and the number of covariates, which represent physicochemical aspects of wine, is $N_0 = 11$. The basic aim of this dataset is to model wine quality in terms of the physicochemical aspects only, and the response is an integer-valued quality score from zero to ten obtained by human expert evaluation. This dataset can be used in both classification and regression, and the linear model was also tested in earlier studies (Cortez et al., 2009; Melkumova and Shatskikh, 2017). Lasso and SS are applied to this dataset.

As preprocessing, N_{noise} noise variables are added into the dataset. Each component of the noise variables is i.i.d. from $\mathcal{N}(0, 1/N)$. The usual standardization, zeroing the mean of all variables and responses and normalizing the variables to be of unit norm, is also conducted.

The reason of the introduction of the noise variables is to make an objective criterion for judging relevant stability paths. The noise variables are by definition irrelevant for describing the responses and hence if some stability paths of original variables behave similarly to the ones of the noise variables then those variables can be regarded as irrelevant. A distribution of stability paths of the noise variables thus defines a kind of "rejection region",

resolving the arbitrariness of judging criterion of the positive probability both in Bolasso and SS. As far as we have searched, this kind of active construction of rejection regions has never been observed in literatures, and hence this proposition itself can be regarded as a part of new results of this paper.

This sort of manipulation to dataset is usually unfavored because it increases the computational cost and tends to contaminate the dataset. However thanks to AMPR, the first computational issue becomes less serious since the computational time of AMPR increases just linearly with respect to the number of added noise variables N_{noise}. The second issue does not seem to be serious either for the wine quality dataset, because the dataset size $M = 4898$ is very large compared to the number of original variables $N_0 = 11$. Below, we experimentally check if this strategy works well or not.

Let us start from checking how the noise variables influence on the dataset. To this end, we plot the solution paths and the generalization errors estimated by 10-fold cross validation (CV) in Figure 9 in the cases with and without the noise variables. This figure

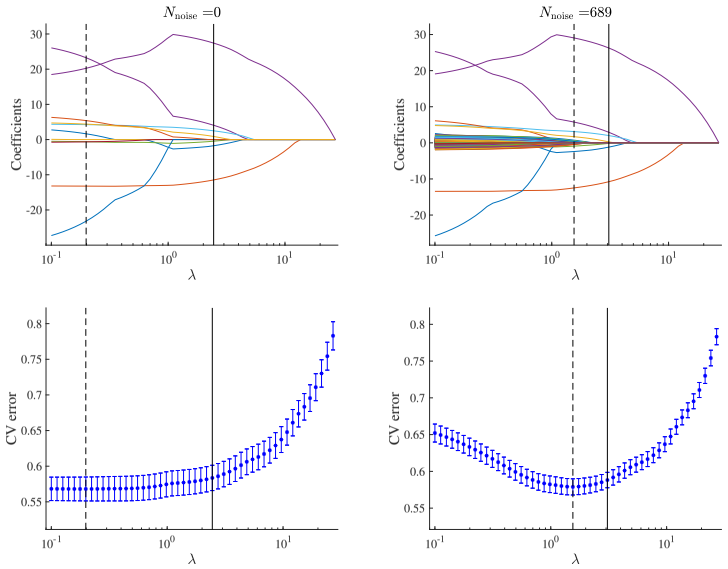


Figure 9: Solution paths (top) and CV errors plotted against λ (bottom), for Nnoise $= 0$ (left) and Nnoise $= 689$ (right). The vertical dotted line denotes the location of the CV error minimum, λ_{min}, whereas the vertical straight line represents the location selected by the one-standard-error rule that "defines" the optimal λ value, λ_{opt}. The effect of the noise variables addition is weak around λ_{opt}.

Table 1: Overlap of the 3rd covariate (citric acid) and the others

index i	1	2	3	4	5	6	7	8	9	10	11
overlap $\boldsymbol{x}_i^\top \boldsymbol{x}_3$	0.29	-0.15	1.00	0.09	0.11	0.09	0.12	0.15	-0.16	0.06	-0.07

indicates that the solution paths of the original variables and the CV error value are stable against the introduction of noise variables, particularly in the relevant range of λ. At the optimal λ ($\lambda_{\rm opt}$), chosen by the one-standard-error rule (Trevor et al., 2009), the variables in the support are common in both cases: They are indexed as $1, 2, 4, 5, 6, 10$, and 11, which represent fixed acidity, volatile acidity, residual sugar, chlorides, free sulfur dioxide, sulphates, and alcohol, respectively. Hence, we can conclude that the introduction of the noise variables hardly affects the estimates of the original variables, and we can effectively use the noise variables to judge the significance of each original variable.

Subsequently, the result of applying SS with $(\tau, w, p_w) = (0.5, 0.5, 0.5)$ to this dataset is examined. Figure 10 shows the plots of the stability paths according to the three categories of variables: The "important" variables are defined as being in the support, at $\lambda_{\rm opt}$ in the Lasso analysis, and thus their indices are $1, 2, 4, 5, 6, 10, 11$; the "neutral" variables are the other variables in the original dataset, and the corresponding indices are $3, 7, 8, 9$; the remaining are the noise variables. Both the results of AMPR (circle) and the direct numerical resampling (asterisk) are shown (the same variable is depicted in the same color).

Consequences of Figure 10 are three-fold. The first one is consistent agreement between the results of AMPR and the direct numerical resampling. For all categories, the two curves of each stability path $\Pi_i(\lambda)$ by AMPR and the direct numerical resampling are very close to each other. This is an additional evidence supporting the accuracy of the proposed semi-analytic method in real-world datasets with correlated covariates. The computational time for obtaining these results in an experiment is 1077 s by AMPR and 2859 s by the numerical resampling, demonstrating the efficiency of AMPR. It should be stressed that this efficiency can be enhanced by optimizing the implementation of AMPR. The second consequence is the different behaviors of stability paths in different categories. The positive probabilities of the important variables (upper left) are growing largely even for $\lambda > \lambda_{\rm min}$, while those of the noise variables (lower left) do not grow well unless λ drops below $\lambda_{\rm min}$. The behavior of the neutral variables (upper right) is somewhat elusive, and a better interpretation is provided by utilizing the noise variables, which is the third consequence: As discussed above, we can define a rejection region from the distribution of stability paths of the noise variables; an actual definition here is the q-percentiles with $q = 16$ and 84 of the distribution, which is in the lower right panel depicted by red bars with the median (red markers) of the distribution, the legend of which is given as FP, in the same manner as Figure 5. This analysis shows that citric acid and total sulfur dioxide of the neutral variables tend to be in the rejection region, implying that they are irrelevant for modeling wine quality. Moreover, density and pH are well beyond the interval of the rejection region, and they can be regarded as relevant, even though the behavior of density's path is rather tricky.

The above conclusion of citric acid's irrelevance contradicts that in Cortez et al. (2009), where citric acid was concluded to be the fourth most important variable. This may be explained by the collinearity of the covariates. In Table 1, the covariates' overlap between citric acid and the others is summarized. This shows a strong collinearity of the 3rd variable

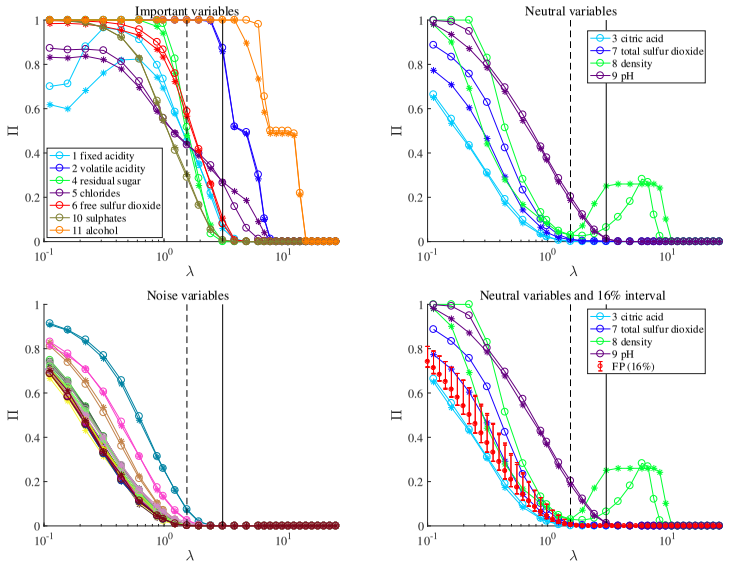


Figure 10: Stability paths for the wine quality data with $N_{\rm noise} = 689$. The paths of the important variables are in the upper left, those of the neutral variables are in the upper right, and those of the noise variables are in the lower left panels. For the lower left panel, the paths of only a part of the noise variables are shown. The vertical dotted and straight lines denote $\lambda_{\rm min}$ and $\lambda_{\rm opt}$, respectively, as in Figure 9. Circles denote the result by AMPR and asterisks represent that by the direct numerical resampling, and the same variable is depicted in the same color; the semi-analytic and numerical results show a consistent agreement for all cases. The lower right panel is the simultaneous plot of the stability paths of the neutral variables and the FP interval computed from paths of the noise variables.

with several others, particularly with the 1st variable. This implies that citric acid can be replaced by the 1st variable or fixed acidity. This is a plausible explanation because the proposed model puts considerably more weight on fixed acidity than on citric acid, whereas the opposite is the case in Cortez et al. (2009). Further thorough comparison will be required to determine which model is better.

Table 1 also implies why AMPR works well for the wine quality dataset: The maximum value of the covariates's overlap is about 0.29 which corresponds $r^{\rm com} \approx 0.5$ in Figure 6; Figure 8 shows that the accuracy of AMPR is commonly good around that value of $r^{\rm com}$, explaining the accuracy of AMPR. This consideration indicates that given a new dataset we can judge whether AMPR will give an accurate result or not for the dataset from the covariates' overlap and Figures 6 and 8.

Overall, the analysis using stability paths provides richer information that cannot be obtained by solely using Lasso. A new objective criterion could be proposed for determining the relevance of variables by utilizing the distribution of stability paths of the added noise variables. These facts highlight the effectiveness of the resampling strategy in variable selection, and the proposed semi-analytic method can implement this strategy in a computationally efficient manner.

5. Conclusion

An approximate method was developed for performing resampling in Lasso in a semi-analytic manner. The replica method enables us to analytically take the resampling average over the given data and the average over the penalty coefficient randomness, and the resultant replicated model is approximately handled by a Gaussian approximation using the cavity method. A message passing algorithm named AMPR is thus derived, the computational cost of which is $O(NM)$ per iteration and is sufficiently reasonable. Its convergence in $O(1)$ iterations is guaranteed by a state evolution analysis when covariates are given as zero-mean i.i.d. Gaussian random variables. We demonstrated how it actually works through numerical experiments using simulated and real-world datasets. Comparison with direct numerical resampling has evidenced its approximation accuracy and efficiency in terms of computational cost, even for covariates with correlations of a moderate level. AMPR was also employed to approximately perform Bolasso and SS, and it was applied to the wine quality dataset (Lichman, 2013; Cortez et al., 2009). To provide a finer quantitative analysis of the dataset, an objective criterion was proposed for determining the relevance of the stability paths by processing the added noise variables, yielding reasonable results in satisfactory computational time.

An advantage of the present framework is its generality. For example, its extension to a generalized linear model is straightforward. This is an immediate future research direction. Extensions to other resampling techniques, such as the multiscale bootstrap method (Shimodaira et al., 2004), would also be interesting.

An unsatisfactory aspect of the present AMPR is that the correlations between covariates are neglected by the approximation. This is a clear drawback, and certain issues arise when the present AMPR is applied to problems involving significantly correlated covariates, although numerical experiments showed that the accuracy of AMPR is still good in the presence of correlations of a moderate level. This drawback may be overcome using more

sophisticated approximations, such as the expectation propagation or the adaptive TAP method (Opper and Winther, 2001a,b, 2005; Kabashima and Vehkapera, 2014; Çakmak et al., 2014; Cespedes et al., 2014; Rangan et al., 2016; Ma and Ping, 2017; Takeuchi, 2017). Applying those approximations with retaining the benefit of message passing algorithms, the low computational cost, is still a nontrivial challenge and promising future work.

Resampling is a very versatile framework applicable to various contexts and models in statistics and machine learning. Reducing its computational cost by extending the present method will thus be beneficial in various fields, and can even be imperative, as available data in society will continue to increase rapidly.

Acknowledgement

This work was supported by JSPS KAKENHI Nos. 25120013 and 17H00764. TO is also supported by a Grant for Basic Science Research Projects from the Sumitomo Foundation. The authors thank Hideitsu Hino for useful comments.

References

Francis R Bach. Bolasso: model consistent lasso estimation through the bootstrap. In *Proceedings of the 25th international conference on Machine learning*, pages 33–40. ACM, 2008.

Onureena Banerjee, Laurent El Ghaoui, Alexandre d'Aspremont, and Georges Natsoulis. Convex optimization techniques for fitting sparse gaussian graphical models. In *Proceedings of the 23rd international conference on Machine learning*, pages 89–96. ACM, 2006.

Jean Barbier, Nicolas Macris, Mohamad Dia, and Florent Krzakala. Mutual information and optimality of approximate message-passing in random linear estimation. *arXiv preprint arXiv:1701.05823*, 2017.

Mohsen Bayati and Andrea Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Transactions on Information Theory*, 57 (2):764–785, 2011.

Francesco Caltagirone, Lenka Zdeborová, and Florent Krzakala. On convergence of approximate message passing. In *2014 IEEE International Symposium on Information Theory*, pages 1812–1816, June 2014. doi: 10.1109/ISIT.2014.6875146.

Burak Çakmak, Ole Winther, and Bernard H Fleury. S-amp: Approximate message passing for general matrix ensembles. In *Information Theory Workshop (ITW), 2014 IEEE*, pages 192–196. IEEE, 2014.

Javier Cespedes, Pablo M Olmos, Matilde Sánchez-Fernández, and Fernando Perez-Cruz. Expectation propagation detection for high-order high-dimensional mimo systems. *IEEE Transactions on Communications*, 62(8):2840–2849, 2014.

Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 2009.

David L Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009.

Viktor Dotsenko. *Introduction to the replica theory of disordered statistical systems*, volume 4. Cambridge University Press, 2005.

Bradley Efron and Robert J Tibshirani. *An introduction to the bootstrap*. CRC press, 1994.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

Max Grazier G'Sell, Jonathan Taylor, and Robert Tibshirani. Adaptive testing for the graphical lasso. *arXiv preprint arXiv:1307.4765*, 2013.

Edwin Hewitt and Leonard J Savage. Symmetric measures on cartesian products. *Transactions of the American Mathematical Society*, 80(2):470–501, 1955.

Adel Javanmard and Andrea Montanari. Confidence intervals and hypothesis testing for high-dimensional regression. *The Journal of Machine Learning Research*, 15(1):2869–2909, 2014a.

Adel Javanmard and Andrea Montanari. Hypothesis testing in high-dimensional regression under the gaussian random design model: Asymptotic theory. *IEEE Transactions on Information Theory*, 60(10):6522–6554, 2014b.

Adel Javanmard and Andrea Montanari. De-biasing the lasso: Optimal sample size for gaussian designs. *arXiv preprint arXiv:1508.02757*, 2015.

Yoshiyuki Kabashima. A cdma multiuser detection algorithm on the basis of belief propagation. *Journal of Physics A: Mathematical and General*, 36(43):11111, 2003.

Yoshiyuki Kabashima and Mikko Vehkapera. Signal recovery using expectation consistent approximation for linear observations. In *Information Theory (ISIT), 2014 IEEE International Symposium on*, pages 226–230. IEEE, 2014.

Keith Knight and Wenjiang Fu. Asymptotics for lasso-type estimators. *Annals of statistics*, pages 1356–1378, 2000.

M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Richard Lockhart, Jonathan Taylor, Ryan J Tibshirani, and Robert Tibshirani. A significance test for the lasso. *Annals of statistics*, 42(2):413, 2014.

Junjie Ma and Li Ping. Orthogonal amp. *IEEE Access*, 5:2020–2033, 2017.

Dörthe Malzahn and Manfred Opper. An approximate analytical approach to resampling averages. *Journal of Machine Learning Research*, 4(Dec):1151–1173, 2003.

Nicolai Meinshausen and Peter Bühlmann. Consistent neighbourhood selection for sparse high-dimensional graphs with the lasso. Seminar für Statistik, Eidgenössische Technische Hochschule (ETH), Zürich, 2004.

Nicolai Meinshausen and Peter Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

Nicolai Meinshausen and Bin Yu. Lasso-type recovery of sparse representations for high-dimensional data. *The Annals of Statistics*, pages 246–270, 2009.

LE Melkumova and S Ya Shatskikh. Comparing ridge and lasso estimators for data analysis. *Procedia Engineering*, 201:746–755, 2017.

Marc Mézard, Giorgio Parisi, and Miguel Virasoro. *Spin glass theory and beyond: An Introduction to the Replica Method and Its Applications*, volume 9. World Scientific Publishing Company, 1987.

Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

Hidetoshi Nishimori. *Statistical physics of spin glasses and information processing: an introduction*, volume 111. Clarendon Press, 2001.

Tomoyuki Obuchi. Matlab package of AMPR. `https://github.com/T-Obuchi/AMPR_lasso_matlab`, 2018.

Manfred Opper and Ole Winther. Adaptive and self-averaging thouless-anderson-palmer mean-field theory for probabilistic modeling. *Physical Review E*, 64(5):056131, 2001a.

Manfred Opper and Ole Winther. Tractable approximations for probabilistic models: The adaptive thouless-anderson-palmer mean field approach. *Physical Review Letters*, 86(17):3695, 2001b.

Manfred Opper and Ole Winther. Expectation consistent approximate inference. *Journal of Machine Learning Research*, 6(Dec):2177–2204, 2005.

Sundeep Rangan, Philip Schniter, and Alyson Fletcher. On the convergence of approximate message passing with arbitrary matrices. In *Information Theory (ISIT), 2014 IEEE International Symposium on*, pages 236–240. IEEE, 2014.

Sundeep Rangan, Philip Schniter, and Alyson Fletcher. Vector approximate message passing. *arXiv preprint arXiv:1610.03082*, 2016.

Hidetoshi Shimodaira et al. Approximately unbiased tests of regions using multistep-multiscale bootstrap resampling. *The Annals of Statistics*, 32(6):2616–2641, 2004.

Takashi Takahashi and Yoshiyuki Kabashima. A statistical mechanics approach to de-biasing and uncertainty estimation in lasso for random measurements. *Journal of Statistical Mechanics: Theory and Experiment*, 2018(7):073405, 2018.

Keigo Takeuchi. Rigorous dynamics of expectation-propagation-based signal recovery from unitarily invariant measurements. In *Information Theory (ISIT), 2017 IEEE International Symposium on*, pages 501–505. IEEE, 2017.

Michel Talagrand. *Spin glasses: a challenge for mathematicians: cavity and mean field models*, volume 46. Springer Science & Business Media, 2003.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Hastie Trevor, Tibshirani Robert, and Friedman Jerome. *The Elements of Statistical Learning; Data Mining, Inference, and Prediction*. Springer-Verlag New York, 2009. doi: 10.1007/978-0-387-84858-7.

Martin J Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1–constrained quadratic programming (lasso). *IEEE transactions on information theory*, 55(5):2183–2202, 2009.

Ming Yuan and Yi Lin. On the non-negative garrotte estimator. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):143–161, 2007.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *Journal of Machine learning research*, 7(Nov):2541–2563, 2006.

Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429, 2006.

Journal of Machine Learning Research 20 (2019) 1-42 Submitted 3/18; Revised 2/19; Published 3/19

Lazifying Conditional Gradient Algorithms

Gábor Braun GABOR.BRAUN@ISYE.GATECH.EDU
Sebastian Pokutta SEBASTIAN.POKUTTA@ISYE.GATECH.EDU
Daniel Zink DANIEL.ZINK@GATECH.EDU
School of Industrial & Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA

Editor: Mark Schmidt

Abstract

Conditional gradient algorithms (also often called Frank–Wolfe algorithms) are popular due to their simplicity of only requiring a linear optimization oracle and more recently they also gained significant traction for online learning. While simple in principle, in many cases the actual implementation of the linear optimization oracle is costly. We show a general method to *lazify* various conditional gradient algorithms, which in actual computations leads to several orders of magnitude of speedup in wall-clock time. This is achieved by using a faster separation oracle instead of a linear optimization oracle, relying only on *few* linear optimization oracle calls.

Keywords: Frank–Wolfe algorithm, conditional gradient, caching, linear optimization oracle, convex optimization

1. Introduction

Convex optimization is an important technique both from a theoretical and an applications perspective. Gradient descent based methods are widely used due to their simplicity and easy applicability to many real-world problems. We are interested in solving constraint convex optimization problems of the form

$$\min_{x \in P} f(x), \tag{1}$$

where f is a smooth convex function and P is a polytope, with access to f being limited to first-order information, i.e., we can obtain $\nabla f(x)$ and $f(x)$ for a given $x \in P$ and access to P via a linear minimization oracle, which returns $\mathrm{LP}_P(c) = \operatorname{argmin}_{x \in P} cx$ for a given linear objective c.

When solving Problem (1) using gradient descent approaches in order to maintain feasibility, typically a projection step is required. This projection back into the feasible region P is potentially computationally expensive, especially for complex feasible regions in very large dimensions. As such, projection-free methods gained a lot of attention recently, in particular the Frank–Wolfe algorithm Frank and Wolfe (1956) (also known as conditional gradient descent Levitin and Polyak (1966); see also Jaggi (2013) for an overview) and its online version Hazan and Kale (2012) due to their simplicity. We recall the basic Frank–Wolfe algorithm in Algorithm 1. These methods eschew the projection step and rather use a

©2019 Gábor Braun, Sebastian Pokutta, and Daniel Zink.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-114.html.

Algorithm 1 Frank–Wolfe Algorithm Frank and Wolfe (1956)
Input: smooth convex function f with curvature C, start vertex $x_1 \in P$, linear minimization oracle LP_P
Output: points x_t in P
1: **for** $t = 1$ **to** $T - 1$ **do**
2: $v_t \leftarrow \mathrm{LP}_P(\nabla f(x_t))$
3: $x_{t+1} \leftarrow (1 - \gamma_t)x_t + \gamma_t v_t$ with $\gamma_t \coloneqq \frac{2}{t+2}$
4: **end for**

linear optimization oracle to stay within the feasible region. While convergence rates and regret bounds are often suboptimal, in many cases the gain due to only having to solve a *single* linear optimization problem over the feasible region in every iteration still leads to significant computational advantages (see e.g., (Hazan and Kale, 2012, Section 5)). This led to conditional gradient algorithms being used for e.g., online optimization and more generally machine learning. Also the property that these algorithms naturally generate sparse distributions over the extreme points of the feasible region is often helpful. Further increasing the relevance of these methods, it was shown recently that conditional gradient methods can also achieve linear convergence (see e.g., Garber and Hazan (2013); Lacoste-Julien and Jaggi (2015); Garber and Meshi (2016)) as well as that the number of total gradient evaluations can be reduced while maintaining the optimal number of oracle calls as shown in Lan and Zhou (2014).

Unfortunately, for complex feasible regions even solving the linear optimization problem might be time-consuming and as such the cost of solving the LP might be non-negligible. This could be the case, e.g., when linear optimization over the feasible region is a hard problem or when solving large-scale optimization or learning problems. As such it is natural to ask the following questions:

(i) Does the linear optimization oracle have to be called in every iteration?

(ii) Does one need approximately optimal solutions for convergence?

(iii) Can one reuse information across iterations?

We will answer these questions in this work, showing that (i) the LP oracle is not required to be called in every iteration, (ii) much weaker guarantees are sufficient, and (iii) we can reuse information. To significantly reduce the cost of oracle calls *while* maintaining identical convergence rates up to small constant factors, we replace the linear optimization oracle by a *(weak) separation oracle* (Oracle 1) which approximately solves a *separation problem*

Oracle 1 Weak Separation Oracle $\mathrm{LPsep}_P(c, x, \Phi, K)$
Input: linear objective $c \in \mathbb{R}^n$, point $x \in P$, accuracy $K \geq 1$, objective value $\Phi > 0$;
Output: Either (1) vertex $y \in P$ with $c(x - y) > \Phi/K$, or (2) **false**: $c(x - z) \leq \Phi$ for all $z \in P$.

within a multiplicative factor and returns improving vertices. We stress that the weak separation oracle is significantly weaker than approximate minimization, which has been

already considered in Jaggi (2013). In fact, there is no guarantee that the improving vertices returned by the oracle are near to the optimal solution to the linear minimization problem. It is this relaxation of dual bounds and approximate optimality that will provide a significant speedup as we will see later. However, if the oracle does not return an improving vertex (returns **false**), then this fact can be used to derive a reasonably small dual bound of the form: $f(x_t) - f(x^*) \leq \nabla f(x_t)(x_t - x^*) \leq \Phi_t$ for some $\Phi_t > 0$. While the accuracy K is presented here as a formal argument of the oracle, an oracle implementation might restrict to a fixed value $K > 1$, which often makes implementation easier. We point out that the cases (1) and (2) potentially overlap if $K > 1$. This is intentional and in this case it is unspecified which of the cases the oracle should choose (and it does not matter for the algorithms).

This new oracle encapsulates the smart use of the *original* linear optimization oracle, even though for some problems it could potentially be implemented directly without relying on a linear programming oracle. Concretely, a weak separation oracle can be realized by a single call to a linear optimization oracle and as such is no more complex than the original oracle. However it has two important advantages: it allows for *caching* and *early termination*. Caching refers to storing previous solutions, and first searching among them to satisfy the oracle's separation condition. The underlying linear optimization oracle is called only, when none of the cached solutions satisfy the condition. Algorithm 2 formalizes this process. Early termination is the technique to stop the linear optimization algorithm before it finishes at an appropriate stage, when from its internal data a suitable oracle answer can be easily recovered; this is clearly an implementation dependent technique. The two techniques can be combined, e.g., Algorithm 2 could use an early terminating linear oracle or other implementation of the weak separation oracle in line 4.

Algorithm 2 $\mathrm{LPsep}_P(c, x, \Phi, K)$ via LP oracle

Input: linear objective $c \in \mathbb{R}^n$, point $x \in P$, accuracy $K \geq 1$, objective value $\Phi > 0$;
Output: Either (1) vertex $y \in P$ with $c(x - y) > \Phi/K$, or (2) **false**: $c(x - z) \leq \Phi$ for all $z \in P$.

1: **if** $y \in P$ cached with $c(x - y) > \Phi/K$ exists **then**
2: **return** y {Cache call}
3: **else**
4: $y \leftarrow \operatorname{argmax}_{x \in P} cx$ {LP call}
5: **if** $c(x - y) > \Phi/K$ **then**
6: add y to cache
7: **return** y
8: **else**
9: **return false**
10: **end if**
11: **end if**

We call *lazification* the technique of replacing a linear programming oracle with a much weaker one, and we will demonstrate significant speedups in wall-clock performance (see e.g., Figure 12), while maintaining identical theoretical convergence rates.

To exemplify our approach we provide conditional gradient algorithms employing the weak separation oracle for the standard Frank–Wolfe algorithm as well as the variants in

Hazan and Kale (2012); Garber and Meshi (2016); Garber and Hazan (2013), which have been chosen due to requiring modified convergence arguments that go beyond those required for the vanilla Frank–Wolfe algorithm. Complementing the theoretical analysis we report computational results demonstrating effectiveness of our approach via a significant reduction in wall-clock time compared to their linear optimization counterparts.

Related Work

There has been extensive work on Frank–Wolfe algorithms and conditional gradient algorithms, so we will restrict to review work most closely related to ours. The Frank–Wolfe algorithm was originally introduced in Frank and Wolfe (1956) (also known as conditional gradient descent Levitin and Polyak (1966) and has been intensely studied in particular in terms of achieving stronger convergence guarantees as well as affine-invariant versions. We demonstrate our approach for the vanilla Frank–Wolfe algorithm Frank and Wolfe (1956) (see also Jaggi (2013)) as an introductory example. We then consider more complicated variants that require non-trivial changes to the respective convergence proofs to demonstrate the versatility of our approach. This includes the linearly convergent variant via local linear optimization Garber and Hazan (2013) as well as the pairwise conditional gradient variant of Garber and Meshi (2016), which is especially efficient in terms of implementation. However, our technique also applies to the *Away-Step Frank–Wolfe* algorithm, the *Fully-Corrective Frank–Wolfe* algorithm, the *Pairwise Conditional Gradient* algorithm, as well as the *Block-Coordinate Frank–Wolfe* algorithm. Recently, in Freund and Grigas (2016) guarantees for arbitrary step-size rules were provided and an analogous analysis can be also performed for our approach. On the other hand, the analysis of the inexact variants, e.g., with approximate linear minimization does not apply to our case as our oracle is significantly weaker than approximate minimization as pointed out earlier. For more information, we refer the interested reader to the excellent overview in Jaggi (2013) for Frank–Wolfe methods in general as well as Lacoste-Julien and Jaggi (2015) for an overview with respect to global linear convergence.

It was also recently shown in Hazan and Kale (2012) that the Frank–Wolfe algorithm can be adjusted to the online learning setting and in this work we provide a lazy version of this algorithm. Combinatorial convex online optimization has been investigated in a long line of work (see e.g., Kalai and Vempala (2005); Audibert et al. (2013); Neu and Bartók (2013)). It is important to note that our regret bounds hold in the *structured online learning* setting, i.e., our bounds depend on the ℓ_1-diameter or sparsity of the polytope, rather than its ambient dimension for arbitrary convex functions (see e.g., Cohen and Hazan (2015); Gupta et al. (2016)). We refer the interested reader to Hazan (2016) for an extensive overview.

A key component of the new oracle is the ability to cache and reuse old solutions, which accounts for the majority of the observed speed up. The idea of caching of oracle calls was already explored in various other contexts such as cutting plane methods (see e.g., Joachims et al. (2009)) as well as the *Block-Coordinate Frank–Wolfe* algorithm in Shah et al. (2015); Osokin et al. (2016). Our lazification approach (which uses caching) is however much more lazy, requiring no multiplicative approximation guarantee; see (Osokin et al., 2016, Proof of Theorem 3. Appendix F) and Lacoste-Julien et al. (2013) for comparison to our setup.

Contribution

The main technical contribution of this paper is a new approach, whereby instead of finding the optimal solution, the oracle is used only to find a *good enough solution* or a *certificate* that such a solution does not exist, both ensuring the desired convergence rate of the conditional gradient algorithms.

Our contribution can be summarized as follows:

(i) *Lazifying approach.* We provide a general method to lazify conditional gradient algorithms. For this we replace the linear optimization oracle with a weak separation oracle, which allows us to reuse feasible solutions from previous oracle calls, so that in many cases the oracle call can be skipped. In fact, once a simple representation of the underlying feasible region is learned no further oracle calls are needed. We also demonstrate how parameter-free variants can be obtained.

(ii) *Lazified conditional gradient algorithms.* We exemplify our approach by providing lazy versions of the vanilla Frank–Wolfe algorithm as well as of the conditional gradient methods in Hazan and Kale (2012); Garber and Hazan (2013); Garber and Meshi (2016).

(iii) *Weak separation through augmentation.* We show in the case of 0/1 polytopes how to implement a weak separation oracle with at most k calls to an augmentation oracle that on input $c \in \mathbb{R}^n$ and $x \in P$ provides either an improving solution $\overline{x} \in P$ with $c\overline{x} < cx$ or ensures optimality, where k denotes the ℓ_1-diameter of P. This is useful when the solution space is sparse.

(iv) *Computational experiments.* We demonstrate computational superiority by extensive comparisons of the weak separation based versions with their original versions. In all cases we report significant speedups in wall-clock time often of several orders of magnitude.

It is important to note that in all cases, we inherit the same requirements, assumptions, and properties of the baseline algorithm that we lazify. This includes applicable function classes, norm requirements, as well as smoothness and (strong) convexity requirements. We also maintain identical convergence rates up to (small) constant factors.

A previous version of this work appeared as extended abstract in Braun et al. (2017); this version has been significantly revised over the conference version including a representative subset of more extensive computational results, full proofs for all described variants, as well as a variant that uses an augmentation oracle instead of linear optimization oracle (see Section 7).

Outline

We briefly recall notation and notions in Section 2 and consider conditional gradient algorithms in Section 3. In Section 4 we consider parameter-free variants of the proposed algorithms, and in Section 5 we examine online versions. Finally, in Section 7 we show a realization of a weak separation oracle with an even weaker oracle in the case of combinatorial problem and we provide extensive computational results in Section 8.

2. Preliminaries

Let $\|\cdot\|$ be an arbitrary norm on $\mathbb{R}^n$, and let $\|\cdot\|^*$ denote the dual norm of $\|\cdot\|$. A function f is *L-Lipschitz* if $|f(y) - f(x)| \leq L\|y - x\|$ for all $x, y \in \operatorname{dom} f$. A convex function f is *smooth* with *curvature* at most C if

$$f(\gamma y + (1-\gamma)x) \leq f(x) + \gamma \nabla f(x)(y - x) + C\gamma^2/2$$

for all $x, y \in \operatorname{dom} f$ and $0 \leq \gamma \leq 1$. A function f is *S-strongly convex* if

$$f(y) - f(x) \geq \nabla f(x)(y - x) + \frac{S}{2}\|y - x\|^2$$

for all $x, y \in \operatorname{dom} f$. Unless stated otherwise Lipschitz continuity and strong convexity will be measured in the norm $\|\cdot\|$. Moreover, let $\mathbb{B}_r(x) \coloneqq \{y \mid \|x - y\| \leq r\}$ be the ball around x with radius r with respect to $\|.\|$. In the following, P will denote the feasible region, a polytope and the vertices of P will be denoted by $v_1, \ldots, v_N$.

3. Lazy Conditional Gradient

We start with the most basic Frank–Wolfe algorithm as a simple example for lazifying by means of a *weak separation oracle*. We then lazify more complex Frank–Wolfe algorithms in Garber and Hazan (2013) and Garber and Meshi (2016). Throughout this section $\|\cdot\|$ denotes the ℓ_2-norm.

3.1. Lazy Conditional Gradient: a basic example

We start with lazifying the original Frank–Wolfe algorithm (arguably the simplest Conditional Gradient algorithm), adapting the baseline argument from (Jaggi, 2013, Theorem 1). While the vanilla version has suboptimal convergence rate $O(1/T)$, its simplicity makes it an illustrative example of the main idea of lazification. The lazy algorithm (Algorithm 3) maintains an upper bound Φ_t on the convergence rate, guiding its eagerness for progress when searching for an improving vertex v_t. If the weak separation oracle provides an improving vertex v_t we refer to this as a *positive call* and if the oracle claims there are no improving vertices we call it a *negative call*.

The step size γ_t is chosen to (approximately) minimize Φ_t in Line 2; roughly Φ_{t-1}/KC.

Theorem 1 *Assume f is convex and smooth with curvature C. Then Algorithm 3 with $\gamma_t = \frac{2(K^2+1)}{K(t+K^2+2)}$ and $f(x_1) - f(x^*) \leq \Phi_0$ has convergence rate*

$$f(x_t) - f(x^*) \leq \frac{2\max\{C, \Phi_0\}(K^2+1)}{t + K^2 + 2},$$

where x^* is a minimum point of f over P.

Proof *We prove by induction that*

$$f(x_t) - f(x^*) \leq \Phi_{t-1}.$$

Algorithm 3 Lazy Conditional Gradient

Input: smooth convex function f with curvature C, start vertex $x_1 \in P$, weak linear separation oracle LPsep_P, accuracy $K \geq 1$, step sizes γ_t, initial upper bound Φ_0
Output: points x_t in P

1: **for** $t = 1$ **to** $T - 1$ **do**
2: $\quad \Phi_t \leftarrow \frac{\Phi_{t-1} + \frac{C\gamma_t^2}{2}}{1 + \frac{\gamma_t}{K}}$
3: $\quad v_t \leftarrow \text{LPsep}_P(\nabla f(x_t), x_t, \Phi_t, K)$
4: $\quad$ **if** $v_t =$ **false then**
5: $\quad\quad x_{t+1} \leftarrow x_t$
6: $\quad$ **else**
7: $\quad\quad x_{t+1} \leftarrow (1 - \gamma_t)x_t + \gamma_t v_t$
8: $\quad$ **end if**
9: **end for**

The claim is clear for $t = 1$ by the choice of Φ_0. Assuming the claim is true for t, we prove it for $t + 1$. We distinguish two cases depending on the return value of the weak separation oracle in Line 3.

In case of a positive call, i.e., when the oracle returns an improving solution v_t, then $\nabla f(x_t)(x_t - v_t) \geq \Phi_t/K$, which is used in the second inequality below. The first inequality follows by smoothness of f, and the second inequality by the induction hypothesis and the fact that v_t is an improving solution:

$$\begin{aligned} f(x_{t+1}) - f(x^*) &\leq \underbrace{f(x_t) - f(x^*)}_{\leq \Phi_{t-1}} + \gamma_t \underbrace{\nabla f(x_t)(v_t - x_t)}_{\leq -\Phi_t/K} + \frac{C\gamma_t^2}{2} \\ &\leq \Phi_{t-1} - \gamma_t \frac{\Phi_t}{K} + \frac{C\gamma_t^2}{2} \\ &= \Phi_t, \end{aligned}$$

In case of a negative call, i.e., when the oracle returns no improving solution, then in particular $\nabla f(x_t)(x_t - x^*) \leq \Phi_t$, hence by Line 5

$$f(x_{t+1}) - f(x^*) = f(x_t) - f(x^*) \leq \nabla f(x_t)(x_t - x^*) \leq \Phi_t.$$

Finally, using the specific values of γ_t we prove the upper bound

$$\Phi_{t-1} \leq \frac{2\max\{C, \Phi_0\}(K^2 + 1)}{t + K^2 + 2}$$

by induction on t. The claim is obvious for $t = 1$. The induction step is an easy computation relying on the definition of Φ_t on Line 2:

$$\Phi_t = \frac{\Phi_{t-1} + \frac{C\gamma_t^2}{2}}{1 + \frac{\gamma_t}{K}} \leq \frac{\frac{2\max\{C,\Phi_0\}(K^2+1)}{t+K^2+2} + \frac{\max\{C,\Phi_0\}\gamma_t^2}{2}}{1 + \frac{\gamma_t}{K}} \leq \frac{2\max\{C, \Phi_0\}(K^2 + 1)}{t + K^2 + 3}.$$

Here the last inequality follows from the concrete value of γ_t. ■

Note that by design, the algorithm converges at the worst-case rate that we postulate due to the negative calls when it does not move. Clearly, this is highly undesirable, therefore the algorithm should be understood as the *textbook variant* of lazy conditional gradient. We will present an improved, parameter-free variant of Algorithm 3 in Section 4 that converges at the best possible rate that the non-lazy variant would achieve (up to a small constant factor).

3.2. Lazy Pairwise Conditional Gradient

In this section we provide a lazy variant (Algorithm 4) of the Pairwise Conditional Gradient algorithm from Garber and Meshi (2016), using separation instead of linear optimization. We make identical assumptions: the feasible region is a 0/1 polytope, i.e., all vertices of P have only 0/1 entries, and moreover it is given in the form $P = \{x \in \mathbb{R}^n \mid 0 \leq x \leq \mathbb{1}, Ax = b\}$, where $\mathbb{1}$ denotes the all-one vector.

Algorithm 4 Lazy Pairwise Conditional Gradient (LPCG)

Input: polytope P, smooth and S-strongly convex function f with curvature C, accuracy $K \geq 1$, non-increasing step-sizes η_t, eagerness Δ_t
Output: points x_t

1: $x_1 \in P$ arbitrary and $\Phi_0 \geq f(x_1) - f(x^*)$
2: **for** $t = 1, \ldots, T$ **do**
3: $\quad \widetilde{\nabla} f(x_t)_i \coloneqq \begin{cases} \nabla f(x_t)_i & \text{if } x_{t,i} > 0 \\ -\infty & \text{if } x_{t,i} = 0 \end{cases}$
4: $\quad \Phi_t \leftarrow \frac{2\Phi_{t-1} + \eta_t^2 C}{2 + \frac{\eta_t}{K\Delta_t}}$
5: $\quad c_t \leftarrow \left(\nabla f(x_t), -\widetilde{\nabla} f(x_t)\right)$
6: $\quad (v_t^+, v_t^-) \leftarrow \text{LPsep}_{P\times P}\left(c_t,\, (x_t, x_t),\, \frac{\Phi_t}{\Delta_t},\, K\right)$
7: $\quad$ **if** $(v_t^+, v_t^-) =$ **false then**
8: $\quad\quad x_{t+1} \leftarrow x_t$
9: $\quad$ **else**
10: $\quad\quad \tilde{\eta}_t \leftarrow \max\{2^{-\delta} \mid \delta \in \mathbb{Z}_{\geq 0},\ 2^{-\delta} \leq \eta_t\}$
11: $\quad\quad x_{t+1} \leftarrow x_t + \tilde{\eta}_t(v_t^+ - v_t^-)$
12: $\quad$ **end if**
13: **end for**

Observe that Algorithm 4 calls the linear separation oracle LPsep on the cartesian product of P with itself. Choosing the objective function as in Line 5 allows us to simultaneously find an improving direction and an away-step direction.

Let $\operatorname{card} x$ denote the number of non-zero entries of the vector x.

Theorem 2 *Let x^* be a minimum point of f in P, and Φ_0 an upper bound of $f(x_1) - f(x^*)$. Furthermore, let $\operatorname{card}(x^*) \leq \alpha$, $M_1 \coloneqq \sqrt{\frac{S}{8\alpha}}$, $\kappa \coloneqq \min\left\{\frac{M_1}{KC}, 1/\sqrt{\Phi_0}\right\}$, $\eta_t \coloneqq \kappa\sqrt{\Phi_{t-1}}$ and*

$\Delta_t \coloneqq \sqrt{\frac{2\alpha\Phi_{t-1}}{S}}$, *then Algorithm 4 has convergence rate*

$$f(x_{t+1}) - f(x^*) \leq \Phi_t \leq \Phi_0 \left(\frac{1+B}{1+2B}\right)^t,$$

where $B \coloneqq \kappa \cdot \frac{M_1}{2K}$.

We recall a technical lemma for the proof.

Lemma 3 ((Garber and Meshi, 2016, Lemma 2)) *Let $x, y \in P$. Then x is a liner combination $x = \sum_{i=1}^k \lambda_i v_i$ of some vertices v_i of P (in particular, $\sum_{i=1}^k \lambda_i = 1$) with $x - y = \sum_{i=1}^k \gamma_i (v_i - z)$ for some $0 \leq \gamma_i \leq \lambda_i$ and $z \in P$ such that $\sum_{i=1}^k \gamma_i \leq \sqrt{\operatorname{card}(y)}\|x-y\|$.*

Proof [Proof of Theorem 2] The feasibility of the iterates x_t is ensured by Line 10 and the monotonicity of the sequence $\{\eta_t\}_{t\geq 1}$ with the same argument as in (Garber and Meshi, 2016, Lemma 1 and Observation 2).

We first show by induction that

$$f(x_{t+1}) - f(x^*) \leq \Phi_t.$$

For $t = 0$ we have $\Phi_0 \geq f(x_1) - f(x^*)$. Now assume the statement for some $t \geq 0$. In case of a negative call (Line 8), we use the guarantee of Oracle 1 to get

$$c_t[(x_t, x_t) - (z_1, z_2)] \leq \frac{\Phi_t}{\Delta_t}$$

for all $z_1, z_2 \in P$, which is equivalent to (as $c_t(x_t, x_t) = 0$)

$$\widetilde{\nabla} f(x_t) z_2 - \nabla f(x_t) z_1 \leq \frac{\Phi_t}{\Delta_t}$$

and therefore

$$\nabla f(x_t)(\tilde{z}_2 - z_1) \leq \frac{\Phi_t}{\Delta_t}, \tag{2}$$

for all $\tilde{z}_2, z_1 \in P$ with $\operatorname{supp}(\tilde{z}_2) \subseteq \operatorname{supp}(x_t)$, where $\operatorname{supp}(x)$ denotes the set of non-zero coordinates of x. We use Lemma 3 for the decompositions $x_t = \sum_{i=1}^k \lambda_i v_i$ and $x_t - x^* = \sum_{i=1}^k \gamma_i (v_i - z)$ with $0 \leq \gamma_i \leq \lambda_i$, $z \in P$ and

$$\sum_{i=1}^k \gamma_i \leq \sqrt{\operatorname{card}(x^*)}\|x_t - x^*\| \leq \sqrt{\frac{2\operatorname{card}(x^*)\Phi_{t-1}}{S}} \leq \Delta_t,$$

using the induction hypothesis and strong convexity in the second inequality. Then

$$f(x_{t+1}) - f(x^*) = f(x_t) - f(x^*) \leq \nabla f(x_t)(x_t - x^*) = \sum_{i=1}^k \gamma_i \cdot \underbrace{\nabla f(x_t)(v_i - z)}_{\leq \Phi_t/\Delta_t} \leq \Phi_t,$$

where we used Equation (2) for the last inequality.

In case of a positive call (Lines 10 and 11) we get, using first smoothness of f, then $\eta_t/2 < \tilde{\eta}_t \leq \eta_t$ and $\nabla f(x_t)(v_t^+ - v_t^-) \leq -\Phi_t/(K\Delta_t)$, and finally the definition of Φ_t:

$$\begin{aligned} f(x_{t+1}) - f(x^*) &= f(x_t) - f(x^*) + f(x_t + \tilde{\eta}_t(v_t^+ - v_t^-)) - f(x_t) \\ &\leq \Phi_{t-1} + \tilde{\eta}_t \nabla f(x_t)(v_t^+ - v_t^-) + \frac{\tilde{\eta}_t^2 C}{2} \\ &\leq \Phi_{t-1} - \frac{\eta_t}{2} \cdot \frac{\Phi_t}{K\Delta_t} + \frac{\eta_t^2 C}{2} = \Phi_t. \end{aligned}$$

Plugging in the values of η_t and Δ_t to the definition of Φ_t gives the desired bound.

$$\Phi_t = \frac{2\Phi_{t-1} + \eta_t^2 C}{2 + \frac{\eta_t}{K\Delta_t}} = \Phi_{t-1}\frac{1 + \kappa^2 C/2}{1 + \kappa M_1/K} \leq \Phi_{t-1}\frac{1+B}{1+2B} \leq \Phi_0 \left(\frac{1+B}{1+2B}\right)^t.$$

■

3.3. Lazy Local Conditional Gradient

In this section we provide a lazy version (Algorithm 5) of the conditional gradient algorithm from Garber and Hazan (2013). Let $P \subseteq \mathbb{R}^n$ be any polytope, D denote an upper bound on the ℓ_2-diameter of P, and $\mu \geq 1$ be an affine invariant parameter of P satisfying Lemma 4 below, see (Garber and Hazan, 2013, Section 2) for a possible definition. As the algorithm is not affine invariant by nature, we need a non-invariant version of smoothness: Recall that a convex function f is *β-smooth* if

$$f(y) - f(x) \leq \nabla f(x)(y - x) + \beta\|y - x\|^2/2.$$

Algorithm 5 Lazy Local Conditional Gradient (LLCG)

Input: feasible polytope P, β-smooth and S-strongly convex function f, parameters K, S, β, μ; diameter D
Output: points x_t

1: $x_1 \in P$ arbitrary and $\Phi_0 \geq f(x_1) - f(x^*)$
2: $\alpha \leftarrow \frac{S}{2K\beta n\mu^2}$
3: **for** $t = 1, \ldots, T$ **do**
4: $\quad r_t \leftarrow \sqrt{\frac{2\Phi_{t-1}}{S}}$
5: $\quad \Phi_t \leftarrow \frac{\Phi_{t-1} + \frac{\beta}{2}\alpha^2 \min\{n\mu^2 r_t^2, D^2\}}{1+\alpha/K}$
6: $\quad p_t \leftarrow \mathrm{LLPsep}_P\left(\nabla f(x_t), x_t, r_t, \Phi_t, K\right)$
7: $\quad$ **if** $p_t =$ **false then**
8: $\quad\quad x_{t+1} \leftarrow x_t$
9: $\quad$ **else**
10: $\quad\quad x_{t+1} \leftarrow x_t + \alpha(p_t - x_t)$
11: $\quad$ **end if**
12: **end for**

Algorithm 6 Weak Local Separation $\mathrm{LLPsep}_P(c, x, r, \Phi, K)$

Input: polytope P together with invariant μ, linear objective $c \in \mathbb{R}^n$, point $x \in P$, radius $r > 0$, objective value $\Phi > 0$, accuracy $K \geq 1$

Output: Either (1) $y \in P$ with $\|x - y\| \leq \sqrt{n}\mu r$ and $c(x - y) > \Phi/K$, or (2) **false**: $c(x - z) \leq \Phi$ for all $z \in P \cap \mathbb{B}_r(x)$.

1: $\Delta \leftarrow \min\left\{\frac{\sqrt{n}\mu}{D} r, 1\right\}$
2: Decompose x: $x = \sum_{j=1}^{M} \lambda_j v_j$, $\lambda_j > 0$, $\sum_j \lambda_j = 1$.
3: Sort vertices: $i_1, \ldots, i_M$ $\quad cv_{i_1} \geq \cdots \geq cv_{i_M}$.
4: $k \leftarrow \min\{k : \sum_{j=1}^{k} \lambda_{i_j} \geq \Delta\}$
5: $p_- \leftarrow \sum_{j=1}^{k-1} \lambda_{i_j} v_{i_j} + \left(\Delta - \sum_{j=1}^{k-1} \lambda_{i_j}\right) v_{i_k}$
6: $v^* \leftarrow \mathrm{LPsep}_P\left(c, \frac{p_-}{\Delta}, \frac{\Phi}{\Delta}\right)$
7: **if** $v^* =$ **false then**
8: **return false**
9: **else**
10: **return** $y \leftarrow x - p_- + \Delta v^*$
11: **end if**

As an intermediary step, we first implement a *local weak separation oracle* in Algorithm 6, a *local* version of Oracle 1, which finds improving points only in a small neighborhood of the original point, analogously to the local linear optimization oracle in Garber and Hazan (2013). To this end, we recall a technical lemma from Garber and Hazan (2013).

Lemma 4 *(Garber and Hazan, 2013, Lemma 7) Let $P \subseteq \mathbb{R}^n$ be a polytope and $v_1, \ldots, v_N$ be its vertices. Let $x, y \in P$ and $x = \sum_{i=1}^{N} \lambda_i v_i$ a convex combination of the vertices of P. Then there are numbers $0 \leq \gamma_i \leq \lambda_i$ and $z \in P$ satisfying*

$$x - y = \sum_{i \in [N]} \gamma_i (z - v_i)$$

$$\sum_{i \in [N]} \gamma_i \leq \frac{\sqrt{n}\mu}{D} \|x - y\|.$$

Now we prove the correctness of the weak local separation algorithm.

Lemma 5 *Algorithm 6 is correct. In particular* $\mathrm{LLPsep}_P(c, x, r, \Phi, K)$

(i) returns either an $y \in P$ with $\|x - y\| \leq \sqrt{n}\mu r$ and $c(x - y) \geq \Phi/K$,

(ii) or returns ***false****, and then $c(x - z) \leq \Phi$ for all $z \in P \cap \mathbb{B}_r(x)$.*

Proof *We first consider the case when the algorithm exits in Line 10. Observe that $y \in P$ since y is a convex combination of vertices of by construction: $y = \sum_{j=1}^{M} (\lambda_{i_j} - \gamma_j) v_{i_j} + \Delta v^*$ with $\Delta = \sum_{j=1}^{M} \gamma_j \leq \frac{\sqrt{n}\mu}{D} r$, where $\gamma_j = \lambda_{i_j}$ for $j < k$, and $\gamma_k = \Delta - \sum_{j=1}^{k-1} \lambda_{i_j}$, and $\gamma_j = 0$ for $j > k$. Therefore*

$$\|x - y\| = \left\| \sum_{j=1}^{M} \gamma_j (v_{i_j} - v^*) \right\| \leq \sum_{j=1}^{M} \gamma_j \|v_{i_j} - v^*\| \leq \sqrt{n}\mu r.$$

Finally using the guarantee of LPsep_P *we get*

$$c(x - y) = \Delta c\left(\frac{p_-}{\Delta} - v^*\right) \geq \frac{\Phi}{K}.$$

If the algorithm exits in Line 8, we use Lemma 4 to decompose any $y \in P \cap \mathbb{B}_r(x)$:

$$x - y = \sum_{i=1}^{M} \gamma_i (v_i - z),$$

with $z \in P$ and $\sum_{i=1}^{M} \gamma_i \leq \frac{\sqrt{n}\mu}{D} \|x - y\| \leq \Delta$. Since $\sum_{i=1}^{M} \lambda_i = 1 \geq \Delta$, there are numbers $\gamma_i \leq \eta_i^- \leq \lambda_i$ with $\sum_{i=1}^{M} \eta_i^- = \Delta$. Let

$$\tilde{p}_- \coloneqq \sum_{i=1}^{M} \eta_i^- v_i,$$

$$\tilde{p}_+ \coloneqq y - x + \tilde{p}_- = \sum_{i=1}^{M} (\eta_i^- - \gamma_i) v_i + \sum_{i=1}^{M} \gamma_i z,$$

so that $\tilde{p}_+/\Delta \in P$. To bound the function value we first observe that the choice of p_- in the algorithm assures that $cu \leq cp_-$ for all $u = \sum_{i=1}^{M} \eta_i v_i$ with $\sum_{i=1}^{M} \eta_i = \Delta$ and all $0 \leq \eta_i \leq \lambda_i$. In particular, $c\tilde{p}_- \leq cp_-$. The function value of the positive part $\tilde{p}_+$ can be bounded with the guarantee of LPsep_P*:*

$$c\left(\frac{p_-}{\Delta} - \frac{\tilde{p}_+}{\Delta}\right) \leq \frac{\Phi}{\Delta},$$

i.e., $c(p_- - \tilde{p}_+) \leq \Phi$. Finally combining these bounds gives

$$c(x - y) = c(\tilde{p}_- - \tilde{p}_+) \leq c(p_- - \tilde{p}_+) \leq \Phi$$

as desired. ■

We are ready to examine the Conditional Gradient Algorithm based on LLPsep_P:

Theorem 6 *Algorithm 5 converges with the following rate:*

$$f(x_{t+1}) - f(x^*) \leq \Phi_t \leq \Phi_0 \left(\frac{1 + \alpha/(2K)}{1 + \alpha/K}\right)^t.$$

Proof *The proof is similar to the proof of Theorem 2. We prove this rate by induction. For $t = 0$ the choice of Φ_0 guarantees that $f(x_1) - f(x^*) \leq \Phi_0$. Now assume the theorem holds for $t \geq 0$. With strong convexity and the induction hypothesis we get*

$$\|x_t - x^*\|^2 \leq \frac{2}{S}(f(x_t) - f(x^*)) \leq \frac{2}{S}\Phi_{t-1} = r_t^2,$$

i.e., $x^* \in P \cap \mathbb{B}_{r_t}(x_t)$. In case of a negative call, i.e., when $p_t =$ **false***, then case (ii) of Lemma 5 applies:*

$$f(x_{t+1}) - f(x^*) = f(x_t) - f(x^*) \leq \nabla f(x_t)(x_t - x^*) \leq \Phi_t.$$

In case of a positive call, i.e., when Line 10 is executed, we get the same inequality via:

$$\begin{aligned} f(x_{t+1}) - f(x^*) &\leq \Phi_{t-1} + \alpha \nabla f(x_t)(p_t - x_t) + \frac{\beta}{2}\alpha^2 \|x_t - p_t\|^2 \\ &\leq \Phi_{t-1} - \alpha \frac{\Phi_t}{K} + \frac{\beta}{2}\alpha^2 \min\{n\mu^2 r_t^2, D^2\} \\ &= \Phi_t. \end{aligned}$$

Therefore using the definition of α and r_t we get the desired bound:

$$\Phi_t \leq \frac{\Phi_{t-1} + \frac{\beta}{2}\alpha^2 r_t^2 n\mu^2}{1 + \alpha/K} = \Phi_{t-1}\left(\frac{1 + \alpha/(2K)}{1 + \alpha/K}\right) \leq \Phi_0 \left(\frac{1 + \alpha/(2K)}{1 + \alpha/K}\right)^t.$$

■

4. Parameter-free Conditional Gradient via Weak Separation

In this section we provide a parameter-free variant of the Lazy Frank–Wolfe Algorithm, which is inspired by Pokutta (2017) and which exhibits a very favorable behavior in computations; the same technique applies to all other variants from Section 3 as well. The idea is that instead of using predetermined values for progress parameters, like Φ_t and γ_t in Algorithm 3, to optimize worst-case progress, the parameters are adjusted adaptively, using data encountered by the algorithm, and avoiding hard-to-estimate parameters, like the curvature C. In practice, this leads to faster convergence, as usual for adaptive methods, while the theoretical convergence rate is worse only by a small constant factor. See Figure 14 for a comparison and Section 8.1.1 for more experimental results.

The occasional halving of the Φ_t is reminiscent of an adaptive restart strategy, considering successive iterates with the same Φ_t as an epoch. It ensures that Φ_t is always at least half of the primal gap, while quickly reducing it if it is too large for the algorithm to make progress, and as such it represents a reasonable amount of expected progress throughout the whole run of the algorithm, not just at the initial iterates.

Remark 7 (Additional LP call for initial bound) *Note that Algorithm 7 finds a tight initial bound Φ_0 with a single extra LP call. If this is undesired, this can be also done approximately as long as Φ_0 is a valid upper bound, for example by means of binary search via the weak separation oracle.*

Theorem 8 *Let f be a smooth convex function with curvature C. Algorithm 7 converges at a rate proportional to $1/t$. In particular to achieve a bound $f(x_t) - f(x^*) \leq \varepsilon$, given an initial upper bound $f(x_1) - f(x^*) \leq 2\Phi_0$, the number of required steps is upper bounded by*

$$t \leq \left\lceil \log \frac{\Phi_0}{\varepsilon} \right\rceil + 1 + 4K \left\lceil \log \frac{\Phi_0}{KC} \right\rceil + \frac{16K^2 C}{\varepsilon}.$$

Proof *The main idea of the proof is to maintain an approximate upper bound on the optimality gap. We then show that negative calls halve the upper bound $2\Phi_t$ and positive oracle calls make significant objective function improvement.*

Algorithm 7 Parameter-free Lazy Conditional Gradient (LCG)

Input: smooth convex function f, start vertex $x_1 \in P$, weak linear separation oracle LPsep_P, accuracy $K \geq 1$
Output: points x_t in P

1: $\Phi_0 \leftarrow \max_{x \in P} \nabla f(x_1)(x_1 - x)/2$ {Initial bound}
2: **for** $t = 1$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \text{LPsep}_P(\nabla f(x_t), x_t, \Phi_{t-1}, K)$
4: $\quad$ **if** $v_t =$ **false then**
5: $\quad\quad x_{t+1} \leftarrow x_t$
6: $\quad\quad \Phi_t \leftarrow \frac{\Phi_{t-1}}{2}$ {Update Φ}
7: $\quad$ **else**
8: $\quad\quad \gamma_t \leftarrow \text{argmin}_{0 \leq \gamma \leq 1} f((1-\gamma)x_t + \gamma v_t)$ {Line search}
9: $\quad\quad x_{t+1} \leftarrow (1 - \gamma_t)x_t + \gamma_t v_t$ {Update iterate}
10: $\quad\quad \Phi_t \leftarrow \Phi_{t-1}$
11: $\quad$ **end if**
12: **end for**

*We analyze iteration t of the algorithm. If Oracle 1 in Line 3 returns a negative answer (i.e., **false**, case (2)), then this guarantees $\nabla f(x_t)(x_t - x) \leq \Phi_{t-1}$ for all $x \in P$, in particular, using convexity, $f(x_{t+1}) - f(x^*) = f(x_t) - f(x^*) \leq \nabla f(x_t)(x_t - x^*) \leq \Phi_{t-1} = 2\Phi_t$.*

If Oracle 1 returns a positive answer (case (1)), then we have $f(x_t) - f(x_{t+1}) \geq \gamma_t \Phi_{t-1}/K - (C/2)\gamma_t^2$ by smoothness of f. By minimality of γ_t, therefore $f(x_t) - f(x_{t+1}) \geq \min_{0 \leq \gamma \leq 1}(\gamma \Phi_{t-1}/K - (C/2)\gamma^2)$, which is $\Phi_{t-1}^2/(2CK^2)$ if $\Phi_{t-1} < KC$, and $\Phi_{t-1}/K - C/2 \geq \frac{C}{2}$ if $\Phi_{t-1} \geq KC$.

Now we bound the number t' of consecutive positive oracle calls immediately following an iteration t with a negative oracle call. Note that the same argument bounds the number of initial consecutive positive oracle calls with the choice $t = 0$, as we only use $f(x_{t+1}) - f(x^*) \leq 2\Phi_t$ below.

Note that $\Phi_t = \Phi_{t+1} = \cdots = \Phi_{t+t'}$. Therefore

$$2\Phi_t \geq f(x_{t+1}) - f(x^*) \geq \sum_{\tau=t+1}^{t+t'} (f(x_\tau) - f(x_{\tau+1})) \geq \begin{cases} t' \frac{\Phi_t^2}{2CK^2} & \text{if } \Phi_t < KC \\ t' \left(\frac{\Phi_t}{K} - \frac{C}{2}\right) & \text{if } \Phi_t \geq KC \end{cases},$$

which gives in the case $\Phi_t < KC$ that $t' \leq 4CK^2/\Phi_t$, and in the case $\Phi_t \geq KC$ that

$$t' \leq \frac{2\Phi_t}{\frac{\Phi_t}{K} - \frac{C}{2}} = \frac{4K\Phi_t}{2\Phi_t - KC} \leq \frac{4K\Phi_t}{2\Phi_t - \Phi_t} = 4K.$$

Thus iteration t is followed by at most $4K$ consecutive positive oracle calls as long as $\Phi_t \geq KC$, and $4CK^2/\Phi_t < 2^{\ell+1} \cdot 4K$ ones for $2^{-\ell-1}KC < \Phi_t \leq 2^{-\ell}KC$ with $\ell \geq 0$.

Adding up the number of oracle calls gives the desired rate: in addition to the positive oracle calls we also have at most $\lceil \log(\Phi_0/\varepsilon) \rceil + 1$ negative oracle calls, where $\log(\cdot)$ is the binary logarithm and ε is the (additive) accuracy. Thus after a total of

$$\left\lceil \log \frac{\Phi_0}{\varepsilon} \right\rceil + 1 + 4K \left\lceil \log \frac{\Phi_0}{KC} \right\rceil + \sum_{\ell=0}^{\lceil \log(KC/\varepsilon) \rceil} 2^{\ell+1} \cdot 4K \leq \left\lceil \log \frac{\Phi_0}{\varepsilon} \right\rceil + 1 + 4K \left\lceil \log \frac{\Phi_0}{KC} \right\rceil + \frac{16K^2 C}{\varepsilon}$$

iterations (or equivalently oracle calls) we have $f(x_t) - f(x^*) \leq \varepsilon$. ■

As seen from the proof, the algorithm receives few negative oracle calls by design; these are usually more expensive than positive ones as the oracle has to compute a certificate by, e.g., executing a full linear optimization oracle call.

Corollary 9 *Algorithm 7 receives at most* $\lceil \log \Phi_0/\varepsilon \rceil + 1$ *negative oracle answers.*

Remark 10 (Improved use of Linear Optimization oracle) *A possible improvement to Line 6 is* $\Phi_t \leftarrow \max_{x \in P} \nabla f(x_t)(x_t - x)/2$*, assuming that at a negative call the oracle also provides the dual gap* $\max_{x \in P} \nabla f(x_t)(x_t - x)$ *as well as the minimizer* $\bar{x} \in P$ *of the oracle call. This is the case e.g., when the weak separation oracle is implemented as in Algorithm 2. Clearly, the minimizer* $\bar{x}$ *can be also used to perform a progress step; albeit without guarantee w.r.t. to* Φ_t*.*

Remark 11 (Line Search) *If line search is too expensive we can choose* $\gamma_t = \min\{1, \Phi_t/KC\}$ *in Algorithm 7. In this case an estimate of the curvature* C *is required.*

5. Lazy Online Conditional Gradient

In this section we lazify the online conditional gradient algorithm of Hazan and Kale (2012) over arbitrary polytopes $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$, resulting in Algorithm 8. We slightly improve constant factors by replacing (Hazan and Kale, 2012, Lemma 3.1) with a better estimation via solving a quadratic inequality arising from strong convexity. In this section the norm $\|\cdot\|$ can be arbitrary.

Theorem 12 *Let* $0 \leq b, s < 1$*. Let* $K \geq 1$ *be an accuracy parameter. Assume* f_t *is L-Lipschitz, and smooth with curvature at most* Ct^{-b}*. Let* $D \coloneqq \max_{y_1, y_2 \in P} \|y_1 - y_2\|$ *denote the diameter of* P *in norm* $\|\cdot\|$*. Then the following hold for the points* x_t *computed by Algorithm 8 where* x_T^* *is the minimizer of* $\sum_{t=1}^T f_t$*:*

(i) With the choice

$$\gamma_t = t^{-(1-b)/2},$$

the x_t *satisfy*

$$\frac{1}{T}\sum_{t=1}^{T}(f_t(x_T) - f_t(x_T^*)) \leq AT^{-(1-b)/2},$$

where

$$A \coloneqq \frac{CK}{2(1-b)} + L(K+1)D.$$

(ii) Moreover, if all the f_t *are* St^{-s}*-strongly convex, then with the choice*

$$\gamma_t = t^{(b+s-2)/3},$$

Algorithm 8 Lazy Online Conditional Gradient (LOCG)

Input: functions f_t, start vertex $x_1 \in P$, weak linear separation oracle LPsep_P, parameters K, C, b, S, s; diameter D

Output: points x_t

1: **for** $t = 1$ **to** $T - 1$ **do**
2: $\quad \nabla_t \leftarrow \nabla f_t(x_t)$
3: $\quad$ **if** $t = 1$ **then**
4: $\quad\quad h_1 \leftarrow \min\{\|\nabla_1\|^* D, 2\|\nabla_1\|^{*2}/S\}$
5: $\quad$ **else**
6: $\quad\quad h_t \leftarrow \Phi_{t-1} + \min\left\{\|\nabla_t\|^* D, \frac{\|\nabla_t\|^{*2}}{St^{1-s}} + 2\sqrt{\frac{\|\nabla_t\|^{*2}}{2St^{1-s}}\left(\frac{\|\nabla_t\|^{*2}}{2St^{1-s}} + \Phi_{t-1}\right)}\right\}$
7: $\quad$ **end if**
8: $\quad \Phi_t \leftarrow \frac{h_t + \frac{Ct^{1-b}\gamma_t^2}{2(1-b)}}{1+\frac{\gamma_t}{K}}$
9: $\quad v_t \leftarrow \text{LPsep}_P(\sum_{i=1}^t \nabla f_i(x_t), x_t, \Phi_t, K)$
10: $\quad$ **if** $v_t =$ **false then**
11: $\quad\quad x_{t+1} \leftarrow x_t$
12: $\quad$ **else**
13: $\quad\quad x_{t+1} \leftarrow (1-\gamma_t)x_t + \gamma_t v_t$
14: $\quad\quad \Phi_t \leftarrow h_t - \sum_{i=1}^t f_i(x_t) + \sum_{i=1}^t f_i(x_{t+1})$
15: $\quad$ **end if**
16: **end for**

the x_t *satisfy*

$$\frac{1}{T}\sum_{t=1}^{T}(f_t(x_T) - f_t(x_T^*)) \leq AT^{-(2(1+b)-s)/3}, \tag{3}$$

where

$$A \coloneqq 2\left((K+1)(K+2)\frac{L^2}{S} + \frac{CK}{2(1-b)}\right).$$

Proof *We prove only Claim (ii), as the proof of Claim (i) is similar and simpler. Let* $F_T \coloneqq \sum_{t=1}^T f_t$*. Furthermore, let* $\overline{h}_T \coloneqq AT^{1-(2(1+b)-s)/3}$ *be* T *times the right-hand side of Equation* (3)*. In particular,* F_T *is* S_T*-strongly convex, and smooth with curvature at most* C_{F_T} *where*

$$C_{F_T} \coloneqq \frac{CT^{1-b}}{1-b} \geq C\sum_{t=1}^{T} t^{-b}, \qquad S_T \coloneqq ST^{1-s} \leq S\sum_{t=1}^{T} t^{-s}.$$

We prove $F_t(x_t) - F_t(x_t^*) \leq h_t \leq \overline{h}_t$ *by induction on* t*. The case* $t = 1$ *is clear. Let* $\overline{\Phi}_t$ *denote the value of* Φ_t *in Line 8, while we reserve* Φ_t *to denote its value as used in Line 6. We start by showing* $F_t(x_{t+1}) - F_t(x_t^*) \leq \Phi_t \leq \overline{\Phi}_t$*. We distinguish two cases depending on the oracle answer* v_t *from Line 9. For a negative oracle answer (*$v_t =$ ***false****), we have* $\Phi_t = \overline{\Phi}_t$ *and the weak separation oracle asserts* $\max_{y \in P} \nabla F_t(x_t)(x_t - y) \leq \Phi_t$*, which combined with the convexity of* F_t *provides*

$$F_t(x_{t+1}) - F_t(x_t^*) = F_t(x_t) - F_t(x_t^*) \leq \nabla F_t(x_t)(x_t - x_{t^*}) \leq \Phi_t = \overline{\Phi}_t.$$

Otherwise, for a positive oracle answer, Line 14 and the induction hypothesis provides $F_t(x_{t+1}) - F_t(x_t^*) \leq h_t + F_t(x_{t+1}) - F_t(x_t) = \Phi_t$. *To prove* $\Phi_t \leq \overline{\Phi}_t$, *we apply the smoothness of* F_t *followed by the inequality provided by the choice of* v_t:

$$F_t(x_{t+1}) - F_t(x_t) - \frac{C_{F_t}\gamma_t^2}{2} \leq \nabla F_t(x_t)(x_{t+1} - x_t) = \gamma_t \nabla F_t(x_t)(v_t - x_t) \leq -\frac{\gamma_t \overline{\Phi}_t}{K}.$$

Rearranging provides the inequality:

$$\Phi_t = h_t + F_t(x_{t+1}) - F_t(x_t) \leq h_t - \frac{\gamma_t \overline{\Phi}_t}{K} + \frac{C_{F_t}\gamma_t^2}{2} = \overline{\Phi}_t.$$

For later use, we bound the difference between $\overline{h}_t$ *and* $\overline{\Phi}_t$ *using the value of parameters,* $h_t \leq \overline{h}_t$, *and* $\gamma_t \leq 1$:

$$\overline{h}_t - \overline{\Phi}_t \geq \overline{h}_t - \frac{\overline{h}_t + \frac{C_{F_t}\gamma_t^2}{2}}{1 + \frac{\gamma_t}{K}} = \frac{\overline{h}_t \gamma_t}{K} - \frac{\frac{C_{F_t}\gamma_t^2}{2}}{1 + \frac{\gamma_t}{K}} \geq \frac{\frac{\overline{h}_t \gamma_t}{K} - \frac{C_{F_t}\gamma_t^2}{2}}{1 + \frac{1}{K}} = \frac{A - \frac{CK}{2(1-b)}}{K+1} t^{[2s-(1+b)]/3}.$$

We now apply $F_t(x_{t+1}) - F_t(x_t^*) \leq \Phi_t$, *together with convexity of* f_{t+1}, *and the minimality* $F_t(x_t^*) \leq F_t(x_{t+1}^*)$ *of* x_t^*, *followed by strong convexity of* F_{t+1}:

$$\begin{aligned} F_{t+1}(x_{t+1}) - F_{t+1}(x_{t+1}^*) &\leq (F_t(x_{t+1}) - F_t(x_t^*)) + (f_{t+1}(x_{t+1}) - f_{t+1}(x_{t+1}^*)) \\ &\leq \Phi_t + \|\nabla_{t+1}\|^* \cdot \|x_{t+1} - x_{t+1}^*\| \\ &\leq \Phi_t + \|\nabla_{t+1}\|^* \sqrt{\frac{2}{S_{t+1}}(F_{t+1}(x_{t+1}) - F_{t+1}(x_{t+1}^*))}. \end{aligned} \tag{4}$$

Solving the quadratic inequality provides

$$F_{t+1}(x_{t+1}) - F_{t+1}(x_{t+1}^*) \leq \Phi_t + \frac{\|\nabla_{t+1}\|^{*2}}{S_{t+1}} + 2\sqrt{\frac{\|\nabla_{t+1}\|^{*2}}{2S_{t+1}}\left(\frac{\|\nabla_{t+1}\|^{*2}}{2S_{t+1}} + \Phi_t\right)}. \tag{5}$$

From Equation (4), *ignoring the last line, we also obtain* $F_{t+1}(x_{t+1}) - F_{t+1}(x_{t+1}^*) \leq \Phi_t + \|\nabla_{t+1}\|^* D$ *via the estimate* $\|x_{t+1} - x_{t+1}^*\| \leq D$. *Thus* $F_{t+1}(x_{t+1}) - F_{t+1}(x_{t+1}^*) \leq h_{t+1}$, *by Line 6, as claimed.*

Now we estimate the right-hand side of Equation (5) *by using the actual value of the parameters, the estimate* $\|\nabla_{t+1}\|^* \leq L$, *and the inequality* $s + b \leq 2$. *In fact, we estimate a proxy for the right-hand side. Note that* A *was chosen to satisfy the second inequality:*

$$\begin{aligned} \frac{L^2}{S_{t+1}} + 2\sqrt{\frac{L^2}{2S_{t+1}}\overline{h}_t} &\leq \frac{L^2}{St^{1-s}} + 2\sqrt{\frac{L^2}{2St^{1-s}}\overline{h}_t} \leq \frac{L^2}{S} t^{[2s-(1+b)]/3} + 2\sqrt{\frac{L^2}{2St^{1-s}}\overline{h}_t} \\ &= \left(\frac{L^2}{S} + \sqrt{2\frac{L^2}{S}A}\right) t^{[2s-(1+b)]/3} \leq \frac{A - \frac{CK}{2(1-b)}}{K+1} t^{[2s-(1+b)]/3} \\ &\leq \overline{h}_t - \overline{\Phi}_t \leq \overline{h}_t - \Phi_t. \end{aligned}$$

In particular, $\frac{L^2}{2S_{t+1}} + \Phi_t \leq \overline{h}_t$ *hence combining with Equation* (5) *we obtain*

$$\begin{aligned} h_{t+1} &\leq \Phi_t + \frac{L^2}{S_{t+1}} + 2\sqrt{\frac{L^2}{2S_{t+1}}\left(\frac{L^2}{2S_{t+1}} + \Phi_t\right)} \\ &\leq \Phi_t + \frac{L^2}{S_{t+1}} + 2\sqrt{\frac{L^2}{2S_{t+1}}\overline{h}_t} \\ &\leq \overline{h}_t \leq \overline{h}_{t+1}. \end{aligned}$$

■

5.1. Stochastic and Adversarial Versions

Complementing the offline algorithms from Section 3, we will now derive various online versions. The presented cases here are similar to those in Hazan and Kale (2012) and thus we state them without proof.

For stochastic cost functions f_t, we obtain bounds from Theorem 12 (i) similar to (Hazan and Kale, 2012, Theorems 4.1 and 4.3) (with δ replaced by δ/T in the bound to correct an inaccuracy in the original argument). The proof is analogous and hence omitted, but note that $\|y_1 - y_2\|_2 \leq \sqrt{\|y_1 - y_2\|_1 \|y_1 - y_2\|_\infty} \leq \sqrt{k}$ for all $y_1, y_2 \in P$.

Corollary 13 *Let* f_t *be convex functions sampled i.i.d. with expectation* $\mathbb{E}[f_t] = f^*$, *and* $\delta > 0$. *Assume that the* f_t *are* L*-Lipschitz in the* 2*-norm.*

(i) If all the f_t *are smooth with curvature at most* C, *then Algorithm 8 applied to the* f_t *(with* $b = 0$*) yields with probability* $1 - \delta$

$$\sum_{t=1}^{T} f^*(x_t) - \min_{x \in P} \sum_{t=1}^{T} f^*(x) \leq O\left(C\sqrt{T} + Lk\sqrt{nT\log(nT^2/\delta)\log T}\right).$$

(ii) Without any smoothness assumption, Algorithm 8 (applied to smoothenings of the f_t*) provides with probability* $1 - \delta$

$$\sum_{t=1}^{T} f^*(x_t) - \min_{x \in P} \sum_{t=1}^{T} f^*(x) \leq O\left(\sqrt{n}LkT^{2/3} + Lk\sqrt{nT\log(nT^2/\delta)\log T}\right).$$

Similar to (Hazan and Kale, 2012, Theorem 4.4), from Theorem 12 (ii) we obtain the following regret bound for adversarial cost functions with an analogous proof.

Corollary 14 *For any* L*-Lipschitz convex cost functions* f_t, *Algorithm 8 applied to the functions* $\tilde{f}_t(x) := \nabla f_t(x_t)x + \frac{2L}{\sqrt{k}} t^{-1/4} \|x - x_1\|_2^2$ *(with* $b = s = 1/4$, $C = L\sqrt{k}$, $S = L/\sqrt{k}$, *and Lipschitz constant* $3L$*) achieving regret*

$$\sum_{t=1}^{T} f_t(x_t) - \min_{x \in P} \sum_{t=1}^{T} f_t(x) \leq O(L\sqrt{k}T^{3/4})$$

with at most T *calls to the weak separation oracle.*

Note that the gradient of the $\tilde{f}_t$ are easily computed via the formula $\nabla \tilde{f}_t(x) = \nabla f_t(x_t) + 4Lt^{-1/4}(x - x_1)/\sqrt{k}$, particularly because the gradient of the f_t need not be recomputed, so that we obtain a weak separation-based stochastic gradient descent algorithm, where we only have access to the f_t through a stochastic gradient oracle, while retaining all the favorable properties of the Frank–Wolfe algorithm with a convergence rate $O(T^{-1/4})$ (c.f., Garber and Hazan (2013)).

6. Non-polytopal domains

So far we have formulated our results for the polytopal case as most of the base methods that we lazify are usually formulated for polytopal domains. However, whenever a base method extends to general compact convex sets as domains then so does our lazification of the base method. In fact it is not even required to use vertices or extremal points as answers to either the LP oracle or weak-separation oracle calls; this is usually only required to obtain iterates as convex combinations of *extremal points* but it is not necessary for convergence.

Base methods that extend to general compact convex sets in particular include the vanilla Frank–Wolfe algorithm, the Away-Step Frank–Wolfe algorithm, and the Pairwise Frank–Wolfe algorithm. Note, however that for the Away-Step Frank–Wolfe algorithm and the Pairwise Frank–Wolfe algorithm (irrespective of lazification) it is not known whether a linear rate of convergence can be achieved for strongly convex functions over general compact convex sets. Base variants that do not readily apply to the non-polyhedral case are the variants in Garber and Meshi (2016) (see Section 3.2) and Garber and Hazan (2013) (see Section 3.3).

We further would like to mention, that often (but not always) for non-polyhedral domains the linear optimization oracle might be more expensive, so that lazification might offer attractive benefits in this case as the effect of caching and early termination might be even more pronounced. We present computational tests for non-polyedral domains for matrix completion, where the feasible domain is given via $\|X\|_* \leq R$, where $\|\cdot\|_*$ is the (non-polyhedral) nuclear norm (see Section 8.1).

7. Weak Separation through Augmentation

So far we realized the weak separation oracle via lazy optimization. We will now create a (weak) separation oracle for integral polytopes, employing an even weaker, so-called augmentation oracle, which only provides an improving solution but provides no guarantee with respect to optimality. We call this approach *lazy augmentation*. This is especially useful when a fast augmentation oracle is available or the vertices of the underlying polytope P are particularly sparse, i.e., $\|y_1 - y_2\|_1 \leq k \ll n$ for all $y_1, y_2 \in P$, where n is the ambient dimension of P. As before theoretical convergence rates are maintained.

For simplicity of exposition we restrict to 0/1 polytopes P here. For general integral polytopes, one considers a so-called *directed augmentation oracle*, which can be similarly linearized after splitting variables in positive and negative parts; we refer the interested reader to see Schulz and Weismantel (2002); Bodic et al. (2015) for an in-depth discussion.

Let k denote the ℓ_1-diameter of P. Upon presentation with a 0/1 solution x and a linear objective $c \in \mathbb{R}^n$, an augmentation oracle either provides an improving 0/1 solution $\bar{x}$ with $c\bar{x} < cx$ or asserts optimality for c:

Oracle 2 Linear Augmentation Oracle AUG_P(c, x)
Input: linear objective $c \in \mathbb{R}^n$, vertex $x \in P$
Output: vertex $\bar{x} \in P$ with $c\bar{x} < cx$ when exists, otherwise $\bar{x} = x$

Such an oracle is significantly weaker than a linear optimization oracle but also significantly easier to implement and much faster; we refer the interested reader to Grötschel and Lovász (1993); Schulz et al. (1995); Schulz and Weismantel (2002) for an extensive list of examples. While augmentation and optimization are polynomially equivalent (even for convex integer programming Oertel et al. (2014)) the current best linear optimization algorithms based on an augmentation oracle are slow for general objectives. While optimizing an *integral* objective $c \in \mathbb{R}^n$ needs $O(k \log\|c\|_\infty)$ calls to an augmentation oracle (see Schulz et al. (1995); Schulz and Weismantel (2002); Bodic et al. (2015)), a general objective function, such as the gradient in Frank–Wolfe algorithms has only an $O(kn^3)$ guarantee in terms of required oracle calls (e.g., via simultaneous diophantine approximations Frank and Tardos (1987)), which is not desirable for large n. In contrast, here we use an augmentation oracle to perform separation, without finding the optimal solution. Allowing a multiplicative error $K > 1$, we realize an augmentation-based weak separation oracle (see Algorithm 9), which decides given a linear objective function $c \in \mathbb{R}^n$, an objective value $\Phi > 0$, and a starting point $x \in P$, whether there is a $y \in P$ with $c(x - y) > \Phi/K$ or $c(x - y) \leq \Phi$ for all $y \in P$. In the former case, it actually provides a certifying $y \in P$, i.e., with $c(x - y) > \Phi/K$. Note that a constant accuracy K requires a linear number of oracle calls in the diameter k, e.g., $K = (1 - 1/e)^{-1} \approx 1.582$ needs at most $N \leq k$ oracle calls, which can be much smaller than the ambient dimension of the polytope.

At the beginning, in Line 2, the algorithm has to replace the input point x with an integral point x_0. If the point x is given as a convex combination of integral points, then a possible solution is to evaluate the objective c on these integral points, and choose x_0 the first one with $cx_0 \leq cx$. This can be easily arranged for Frank–Wolfe algorithms as they maintain convex combinations.

Proposition 15 *Assume $\|y_1 - y_2\|_1 \leq k$ for all $y_1, y_2 \in P$. Then Algorithm 9 is correct, i.e., it outputs either (1) $y \in P$ with $c(x - y) > \Phi/K$, or (2)* **false**. *In the latter case $c(x - y) \leq \Phi$ for all $y \in P$ holds. The algorithm calls AUG_P at most $N \leq \lceil \log(1 - 1/K)/\log(1 - 1/k) \rceil$ many times.*

Proof *First note that $(\mathbb{1} - 2x)v + \|x\|_1 = \|v - x\|_1$ for $x, v \in \{0,1\}^n$, hence Line 7 is equivalent to $x_i \leftarrow \text{AUG}_P(c + \frac{\Phi - c(x - x_{i-1})}{k}\|\cdot - x_{i-1}\|_1, x_{i-1})$.*

The algorithm obviously calls the oracle at most N times by design, and always returns a value, so we need to verify only the correctness of the returned value. We distinguish cases according to the output.

Algorithm 9 Augmenting Weak Separation $\mathrm{LPsep}_P(c, x, \Phi, K)$

Input: linear objective $c \in \mathbb{R}^n$, point $x \in P$, objective value $\Phi > 0$; accuracy $K > 1$
Output: Either (1) $y \in P$ vertex with $c(x-y) > \Phi/K$, or (2) **false**: $c(x-z) \leq \Phi$ for all $z \in P$.

1: $N \leftarrow \lceil \log(1-1/K)/\log(1-1/k) \rceil$
2: Choose $x_0 \in P$ vertex with $cx_0 \leq cx$.
3: **for** $i = 1$ **to** N **do**
4: **if** $c(x - x_{i-1}) \geq \Phi$ **then**
5: **return** x_{i-1}
6: **end if**
7: $x_i \leftarrow \mathrm{AUG}_P(c + \frac{\Phi - c(x-x_{i-1})}{k}(\mathbb{1} - 2x_{i-1}), x_{i-1})$
8: **if** $x_i = x_{i-1}$ **then**
9: **return false**
10: **end if**
11: **end for**
12: **return** x_N

Clearly, Line 5 always returns an x_{i-1} with $c(x - x_{i-1}) \geq \Phi > [1-(1-1/k)^N]\Phi$. When Line 9 is executed, the augmentation oracle just returned $x_i = x_{i-1}$, i.e., for all $y \in P$

$$cx_{i-1} \leq cy + \frac{\Phi - c(x-x_{i-1})}{k}\|y - x_{i-1}\|_1 \leq cy + \frac{\Phi - c(x-x_{i-1})}{k}k = c(y-x) + cx_{i-1} + \Phi,$$

so that $c(x-y) \leq \Phi$, as claimed.

Finally, when Line 12 is executed, the augmentation oracle has found an improving vertex x_i at every iteration, i.e.,

$$cx_{i-1} > cx_i + \frac{\Phi - c(x-x_{i-1})}{k}\|x_i - x_{i-1}\|_1 \geq cx_i + \frac{\Phi - c(x-x_{i-1})}{k},$$

using $\|x_i - x_{i-1}\|_1 \geq 1$ by integrality. Rearranging provides the convenient form

$$\Phi - c(x - x_i) < \left(1 - \frac{1}{k}\right)[\Phi - c(x - x_{i-1})],$$

which by an easy induction provides

$$\Phi - c(x - x_N) < \left(1 - \frac{1}{k}\right)^N [\Phi - c(x - x_0)] \leq \left(1 - \frac{1}{K}\right)\Phi,$$

i.e., $c(x - x_N) \geq \frac{\Phi}{K}$, finishing the proof. ■

8. Experiments

We implemented and compared the parameter-free variant of LCG (Algorithm 7) to the standard Frank–Wolfe algorithm (CG), then Algorithm 4 (LPCG) to the Pairwise Conditional Gradient algorithm (PCG) of Garber and Meshi (2016), as well as Algorithm 8 (LOCG) to the Online Frank–Wolfe algorithm (OCG) of Hazan and Kale (2012). While we did implement the Local Conditional Gradient algorithm of Garber and Hazan (2013) as well, the very large constants in the original algorithms made it impractical to run. Unless stated otherwise the weak separation oracle is implemented as sketched in Algorithm 2 through caching and early termination of the original LP oracle.

We have used $K = 1.1$ and $K = 1$ as multiplicative factors for the weak separation oracle; for the impact of the choice of K see Section 8.2.2. For the baseline algorithms we use inexact variants, i.e., we solve linear optimization problems only approximately. This is a significant speedup in favor of non-lazy algorithms at the (potential) cost of accuracy, while neutral to lazy optimization as it solves an even more relaxed problem anyways. To put things in perspective, the non-lazy baselines could not complete even a single iteration for a significant fraction of the considered problems in the given time frame if we were to exactly solve the linear optimization problems. In terms of using line search, for all tests we treated all algorithms equally: either *all* or *none* used line search. If not stated otherwise, we used (simple backtracking) line search.

The linear optimization oracle over $P \times P$ for LPCG was implemented by calling the respective oracle over P twice: once for either component. Contrary to the non-lazy version, the lazy algorithms depend on the initial upper bound Φ_0. For the instances that need a very long time to solve the (approximate) linear optimization even once, we used a binary search for Φ_0 for the lazy algorithms: starting from a conservative initial value, using the update rule $\Phi_0 \leftarrow \Phi_0/2$ until the separation oracle returns an improvement for the first time and then we start the algorithm with $2\Phi_0$, which is an upper bound on the Wolfe gap and hence also on the primal gap. This initial phase is also included in the reported wall-clock time. Alternatively, if the linear optimization was less time consuming we used a single (approximate) linear optimization at the start to obtain an initial bound on Φ_0 (see e.g., Section 4).

In some cases, especially when the underlying feasible region has a high dimension and the (approximate) linear optimization can be solved relatively fast compared to the cost of computing an inner product, we observed that the costs of maintaining the cache was very high. In these cases we reduced the cache size every 100 steps by keeping only the 100 points that were used the most so far. Both the number of steps and the approximate size of the cache were chosen arbitrarily, however 100 for both worked very well for all our examples. Of course there are many different strategies for maintaining the cache, which could be used here and which could lead to further improvements in performance.

The stopping criteria for each of the experiments was a given wall clock time limit in seconds. The time limit was enforced separately for the main code and the oracle code, so in some cases the actual time used can be larger, when the last oracle call started before the time limit was reached and took longer than the time left.

We implemented all algorithms in Python 2.7 with critical functions *cythonized* for performance employing Numpy. We used these packages from the Anaconda 4.2.0 distribution as well as Gurobi 7.0 Gurobi Optimization (2016) as a black box solver for the linear optimization oracle. The weak separation oracle was implemented via a callback function to stop linear optimization as soon as a good enough feasible solution has been found in a schema as outlined in Algorithm 2. The parameters for Gurobi were kept at their

default settings except for enforcing the time limit of the tests and setting the acceptable duality gap to 10%, allowing Gurobi to terminate the linear optimization early avoiding the expensive *proof* of optimality. This is used to realize the inexact versions of the baseline algorithms. All experiments were performed on a 16-core machine with Intel Xeon E5-2630 v3 @ 2.40GHz CPUs and 128GB of main memory. While our code does not explicitly use multiple threads, both Gurobi and the numerical libraries use multiple threads internally.

8.1. Computational results

We performed computational tests on a large variety of different problems that are instances of the three machine learning tasks *video colocalization*, *matrix completion*, and *structured regression*.

Video colocalization. Video colocalization is the problem of identifying objects in a sequence of multiple frames in a video. In Joulin et al. (2014) it is shown that video colocalization can be reduced to optimizing a quadratic objective function over a flow or a path polytope, which is the problem we are going to solve. The resulting linear program is an instance of the minimum-cost network flow problem, see (Joulin et al., 2014, Eq. (3)) for the concrete linear program and more details. The quadratic functions are of the form $\|Ax - b\|^2$ where we choose the non-zero entries in A according to a density parameter at random and then each of these entries to be $[0, 1]$-uniformly distributed, while b is chosen as a linear combination of the columns of A with random multipliers from $[0, 1]$. For some of the instances we also use $\|x - b\|^2$ as the objective function with $b_i \in [0, 1]$ uniformly at random.

Matrix completion. The formulation of the matrix completion problem we are going to use is the following:

$$\min_X \sum_{(i,j)\in\Omega} |X_{i,j} - A_{i,j}|^2 \qquad \text{s.t.} \quad \|X\|_* \leq R, \tag{6}$$

where $\|\cdot\|_*$ denotes the nuclear norm, i.e., $\|A\|_* = \operatorname{Tr}(\sqrt{A^t A})$. The set Ω, the matrix A and R are given parameters. Similarly to Lan and Zhou (2014) we generate the $m \times n$ matrix A as the product of A_L of size $m \times r$ and A_R of size $r \times n$. The entries in A_L and A_R are chosen from a standard Gaussian distribution. The set Ω is chosen uniformly of size $s = \min\{5r(m+n-r), \lceil 0.99mn \rceil\}$. The linear optimization oracle is implemented in this case by a singular value decomposition of the linear objective function and we essentially solve the LP to (approximate) optimality. The matrix completion tests will only demonstrate the impact of caching solutions. Note that this test is also informative as due to the 'roundness' of the feasible region the solution of the actual LP oracle will induce a direction that is equal to the true gradient and as such it provides insight into how much per-iteration progress is lost due to working with gradient approximations from the weak separation oracle.

Structured regression. The structured regression problem consists of solving a quadratic function of the form $\|Ax - b\|^2$ over some structured feasible set or a polytope P, i.e., we solve $\min_{x\in P} \|Ax - b\|^2$. We construct the objective functions in the same way as for the video colocalization problem.

Tests. In the following two sections we will present our results for various problems grouped by the versions of the considered algorithms. Every figure contains two columns, each containing one experiment. We use different measures to report performance: we report progress of loss or function value in wall-clock time in the first row (including time spent by the oracle), in the number of iterations in the second row, and in the number of linear optimization calls in the last row. Obviously, the latter only makes sense for the lazy algorithms. In some other cases we report in another row the dual bound or Wolfe gap in wall-clock time. The red line denotes the non-lazy algorithm and the green line denotes the lazy variants. For each experiment we also report the cache hit rate, which is the number of oracle calls answered with a point from the cache over all oracle calls given in percent.

While we found convergence rates in the number of iterations quite similar (as expected!), we consistently observe a significant speedup in wall-clock time. In particular for many large-scale or hard combinatorial problems, lazy algorithms performed several thousand iterations whereas the non-lazy versions completed only a handful of iterations due to the large time spent approximately solving the linear optimization problem. The observed cache hit rate was at least 90% in most cases, and often even above 99%.

Compared to the non-lazy variants, the lazy variants might use weaker descent directions (due to employing the weak-separation oracle instead of the LP oracle), and hence one expects that the lazy algorithms in general require more iterations despite being faster in wall-clock time. However, the lazy and non-lazy algorithms usually generate different sequences of iterates, and hence the lazy algorithms may converge faster even in the number of iterations by chance; this is even expected in a small number of cases by the law of large numbers. This happens for example in Figures 3 and 14.

8.1.1. Offline Results

We describe the considered instances in the offline case separately for the vanilla Frank–Wolfe method and the Pairwise Conditional Gradient method.

Vanilla Frank–Wolfe Method We tested the vanilla Frank–Wolfe algorithm on the six video colocalization instances with underlying path polytopes from `http://lime.cs.elte.hu/~kpeter/data/mcf/netgen/` (Figure 1). In these instances we additionally report the dual bound or Wolfe gap in wall clock time. We further tested the vanilla Frank–Wolfe algorithm on eight instances of the matrix completion problem generated as described above, for which we did not use line search; the parameter-free lazy variant is run with approximate minimization as described in Remark 11, the others use their respective standard step sizes. We provide the used parameters for each example in the figures below (Figures 2 and 3). The last tests for this version were performed on three instances of the structured regression problem, two with the feasible region containing flow-based formulations of Hamiltonian cycles in graphs (Figure 4), and further tests on two spanning tree instances of different size (Figure 5).

We observed a significant speedup of LCG compared to CG, due to the faster iteration of the lazy algorithm.

Pairwise Conditional Gradient Algorithm As we inherit structural restrictions of PCG on the feasible region, the problem repertoire is limited in this case. We tested the

Pairwise Conditional Gradient algorithm on the structured regression problem with feasible regions from the MIPLIB instances `eil33-2`, `air04` (Figure 6).

Again similarly to the vanilla Frank–Wolfe algorithm, we observed a significant improvement in wall-clock time of LPCG compared to CG, due to the faster iteration of the lazy algorithm.

8.1.2. Online Results

Additionally to the quadratic objective functions above we tested the online version on random linear functions $cx + b$ with $c \in [-1, +1]^n$ and $b \in [0, 1]$. For online algorithms, each experiment used a random sequence of 100 different random loss functions. In every figure the left column uses linear loss functions, while the right one uses quadratic loss functions over the same polytope. As customary, we did not use line search here but used the respective prescribed step sizes.

As an instance of the structured regression problem we used the standard formulation of the cut polytope for graphs with 28 nodes as the feasible region (Figure 7). We also tested our algorithm on the quadratic unconstrained boolean optimization (QUBO) instances defined on Chimera graphs Dash (2013), which are available at `http://researcher.watson.ibm.com/researcher/files/us-sanjeebd/chimera-data.zip`. The instances are relatively hard albeit their rather small size and in general the problem is NP-hard. (Figure 8).

One instance of the video colocalization problem uses a path polytope from `http://lime.cs.elte.hu/~kpeter/data/mcf/netgen/` that was generated with the `netgen` graph generator (Figure 9). Most of these instances are very large-scale minimum cost flow instances with several tens of thousands nodes in the underlying graphs, therefore solving still takes considerable time despite the problem being in P. Finally, for the spanning tree problem, we used the well-known extended formulation with $O(n^3)$ inequalities for an n-node graph. We considered graphs with 25 nodes (Figures 10).

We observed that similarly to the offline case while OCG and LOCG converge comparably in the number of iterations, the lazy LOCG performed significantly more iterations; for hard problems, where linear optimization is costly and convergence requires a large number of iterations, this led LOCG converging much faster in wall-clock time. In extreme cases OCG could not complete even a single iteration. This is due to LOCG only requiring *some* good enough solution, whereas OCG requires a stronger guarantee. This is reflected in faster oracle calls for LOCG.

Weak-Separation via Augmentation As discussed in Section 7 in some cases it can be very beneficial to realize the weak-separation oracle by means of augmentation instead of linear optimization. To verify this we implemented a weak-separation oracle via an augmentation oracle for quadratic unconstrained boolean optimization (QUBO) instances (see above for details). For those instances, the primal heuristics of Gurobi (or CPLEX) can find improving solutions very fast due to the structure of the instances. We obtain the augmentation oracle then by exiting the Gurobi call as soon as the first improving solution is found. For comparability, apart from using the augmentation oracle we used the same configuration as above. The results can be found in Figure 11, where we can observe a significant speedup of the lazified variant of OCG using augmentation over the base OCG algorithm. Also observe that this is in contrast to Figure 8, where the advantage of LOCG

(without augmentation) over OCG was not that pronounced. To provide further insight in this case we also report *oracle time*, which denotes actual time spent in the augmentation oracle, which is minimal in both cases as can be seen.

8.2. Performance improvements, parameter sensitivity, and tuning

8.2.1. Effect of caching

As mentioned before, lazy algorithms have two improvements: caching and early termination. Here we depict the effect of caching in Figure 12, comparing OCG (no caching, no early termination), LOCG (caching and early termination) and LOCG (only early termination) (see Algorithm 8). We did not include a caching-only OCG variant, because caching without early termination does not make much sense: in each iteration a new linear optimization problem has to be solved; previous solutions can hardly be reused as they are unlikely to be optimal for the new linear optimization problem.

8.2.2. Effect of K

If the parameter K of the oracle can be chosen, which depends on the actual oracle implementation, then we can increase K to bias the algorithm towards performing more positive calls. At the same time the steps get shorter. As such there is a natural trade-off between the cost of many positive calls vs. a negative call. We depict the impact of the parameter choice for K in Figure 13.

8.2.3. Parameter-free vs. textbook variant

For illustrative purposes, we compare the textbook variant of the lazy conditional gradient (Algorithm 3) with its parameter-free counterpart (Algorithm 7) in Figure 14. The parameter-free variant outperforms the textbook variant due to the active management of Φ combined with line search.

Similar parameter-free variants can be derived for the other algorithms; see discussion in Section 4.

9. Final Remarks

As discussed above in Section 6, if a given baseline algorithm works over general compact convex sets P, then so does the lazified version. In fact, as the lazified algorithm runs, it produces a polyhedral approximation of the set P with very few vertices (subject to optimality vs. sparsity tradeoffs; see (Jaggi, 2013, Appendix C)).

Moreover, the weak separation oracle does not need to return extreme points. All algorithms also work with maximal solutions that are not necessarily extremal (e.g., lying in a higher-dimensional face). However, in that case we lose the desirable property that the final solution is a sparse convex combination of extreme points (typically vertices in the polyhedral setup).

We would also like to briefly address potential downsides of our approach. In fact, we believe the right perspective is the following: when using the lazy oracle over the LP oracle, we obtain potentially *weaker* approximations $v_t - x_t$ of the true gradient $\nabla f(x_t)$ compared

to solving the actual LP, but the computation might be much faster. This is the tradeoff that one has to consider: working with weaker approximations (which implies potentially less progress per iteration) vs. potentially significantly faster computation of the approximations. If solving the LP is expensive, then lazification will be usually very beneficial, if the LP is very cheap as in the case of $P = [0, 1]^n$ or $P = \Delta_n$ being the probability simplex, then lazification might be slower.

A related remark in this context is that once the lazified algorithm has obtained vertices $x_1, \ldots, x_m$ of P, so that the minimizer x^* of f satisfies $x^* \in \operatorname{conv}\{x_1, \ldots, x_m\}$, then from that point onwards no actual calls to the true LP oracle have to be performed anymore for primal progress and the algorithm will only use cache calls; the only remaining true LP calls are at most a logarithmic number for dual progress updates of the Φ_t.

Acknowledgements

We are indebted to Alexandre D'Aspremont, Simon Lacoste-Julien, and George Lan for the helpful discussions and for providing us with relevant references. Research reported in this paper was partially supported by NSF CAREER award CMMI-1452463.

References

Jean-Yves Audibert, Sébastien Bubeck, and Gábor Lugosi. Regret in online combinatorial optimization. *Mathematics of Operations Research*, 39(1):31–45, 2013.

Pierre Le Bodic, Jeffrey W Pavelka, Marc E Pfetsch, and Sebastian Pokutta. Solving MIPs via scaling-based augmentation. *arXiv preprint arXiv:1509.03206*, 2015.

G. Braun, S. Pokutta, and D. Zink. Lazifying Conditional Gradient Algorithms. *Proceedings of ICML*, 2017.

Alon Cohen and Tamir Hazan. Following the perturbed leader for online structured learning. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1034–1042, 2015.

Sanjeeb Dash. A note on QUBO instances defined on Chimera graphs. *preprint arXiv:1306.1202*, 2013.

András Frank and Éva Tardos. An application of simultaneous Diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987.

Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

Robert M. Freund and Paul Grigas. New analysis and results for the Frank–Wolfe method. *Mathematical Programming*, 155(1):199–230, 2016. ISSN 1436-4646. doi: 10.1007/s10107-014-0841-6. URL `http://dx.doi.org/10.1007/s10107-014-0841-6`.

Dan Garber and Elad Hazan. A linearly convergent conditional gradient algorithm with applications to online and stochastic optimization. *arXiv preprint arXiv:1301.4666*, 2013.

Dan Garber and Ofer Meshi. Linear-memory and decomposition-invariant linearly convergent conditional gradient algorithm for structured polytopes. *arXiv preprint, arXiv:1605.06492v1*, May 2016.

Martin Grötschel and László Lovász. Combinatorial optimization: A survey, 1993.

Swati Gupta, Michel Goemans, and Patrick Jaillet. Solving combinatorial games using products, projections and lexicographically optimal bases. *arXiv preprint arXiv:1603.00522*, 2016.

Gurobi Optimization. Gurobi optimizer reference manual version 6.5, 2016. URL `https://www.gurobi.com/documentation/6.5/refman/`.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3–4):157–325, 2016. doi: 10.1561/2400000013. URL `http://ocobook.cs.princeton.edu/`.

Elad Hazan and Satyen Kale. Projection-free online learning. *arXiv preprint arXiv:1206.4657*, 2012.

Martin Jaggi. Revisiting Frank–Wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 427–435, 2013.

Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, 2009.

Armand Joulin, Kevin Tang, and Li Fei-Fei. Efficient image and video co-localization with Frank-Wolfe algorithm. In *European Conference on Computer Vision*, pages 253–268. Springer, 2014.

Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

Simon Lacoste-Julien and Martin Jaggi. On the global linear convergence of Frank–Wolfe optimization variants. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28, pages 496–504. Curran Associates, Inc., 2015. URL `http://papers.nips.cc/paper/5925-on-the-global-linear-convergence-of-frank-wolfe-optimization-variants.pdf`.

Simon Lacoste-Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. Block-coordinate Frank–Wolfe optimization for structural SVMs. In *ICML 2013 International Conference on Machine Learning*, pages 53–61, 2013.

Guanghui Lan and Yi Zhou. Conditional gradient sliding for convex optimization. *Optimization-Online preprint (4605)*, 2014.

Evgeny S Levitin and Boris T Polyak. Constrained minimization methods. *USSR Computational mathematics and mathematical physics*, 6(5):1–50, 1966.

Gergely Neu and Gábor Bartók. An efficient algorithm for learning with semi-bandit feedback. In *Algorithmic Learning Theory*, pages 234–248. Springer, 2013.

Timm Oertel, Christian Wagner, and Robert Weismantel. Integer convex minimization by mixed integer linear optimization. *Oper. Res. Lett.*, 42(6-7):424–428, 2014.

Anton Osokin, Jean-Baptiste Alayrac, Isabella Lukasewitz, Puneet K Dokania, and Simon Lacoste-Julien. Minding the gaps for block Frank–Wolfe optimization of structured SVMs. *ICML 2016 International Conference on Machine Learning / arXiv preprint arXiv:1605.09346*, 2016.

Sebastian Pokutta. Smooth convex optimization via geometric scaling. *preprint*, 2017.

Andreas S Schulz and Robert Weismantel. The complexity of generic primal algorithms for solving general integer programs. *Mathematics of Operations Research*, 27(4):681–692, 2002.

Andreas S. Schulz, Robert Weismantel, and Günter M. Ziegler. 0/1-integer programming: Optimization and augmentation are equivalent. In *Algorithms – ESA '95, Proceedings*, pages 473–483, 1995.

Neel Shah, Vladimir Kolmogorov, and Christoph H Lampert. A multi-plane block-coordinate Frank–Wolfe algorithm for training structural SVMs with a costly max-oracle. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2737–2745, 2015.

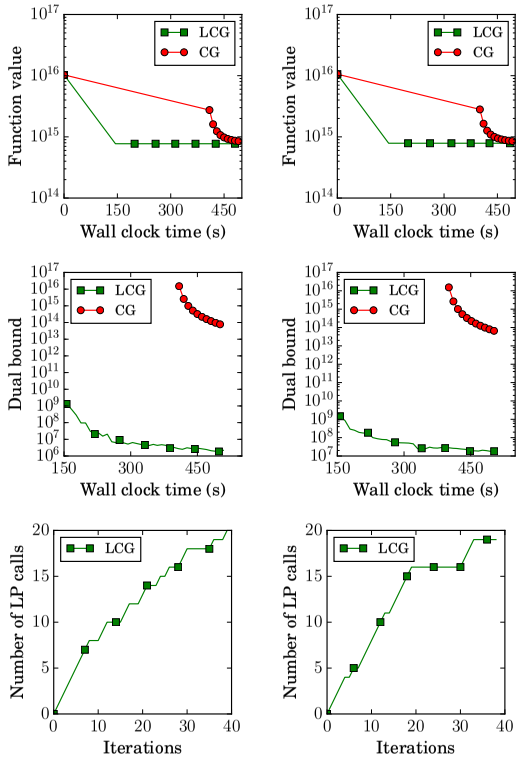


Figure 1: LCG vs. CG on large netgen instances `netgen 16a` (left) and `netgen 16b` (right) with quadratic objective functions. In both cases the difference in function value between the two versions of the algorithm is large. In the dual bound the performance of the lazy version is multiple orders of magnitude better than the performance of the non-lazy counterpart. The cache hit rates for these two instances are lower due to the high dimension of the polytope.

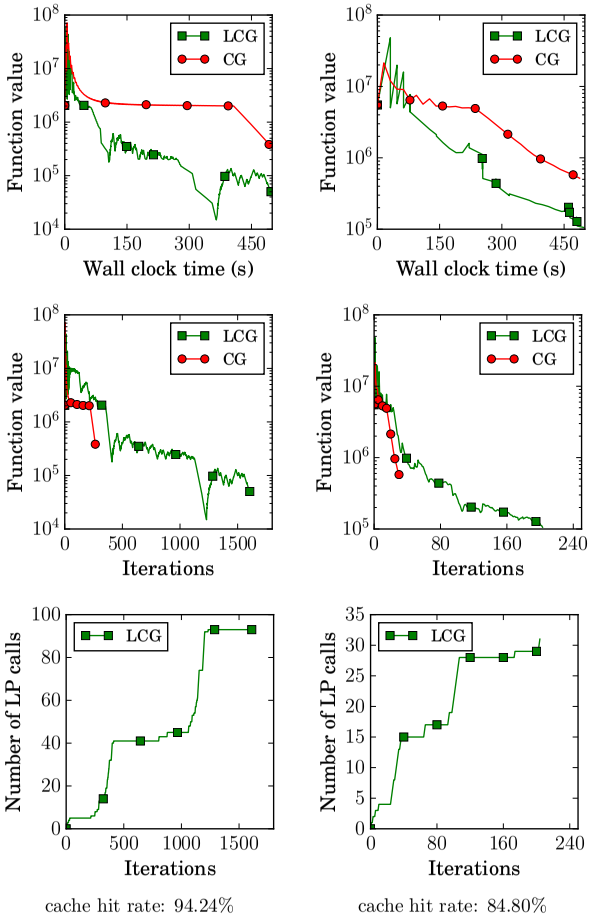


Figure 2: LCG vs. CG on two matrix completion instances. We solve the problem as given in Equation (6) with the parameters $n = 3000$, $m = 1000$, $r = 10$ and $R = 30000$ for the left instance and $n = 10000$, $m = 100$, $r = 10$ and $R = 10000$ for the right instance. In both cases the lazy version is slower in iterations, however significantly faster in wall clock time. Note that we used the short-step rule for step sizes for both algorithms as line search for matrix completion is very expensive.

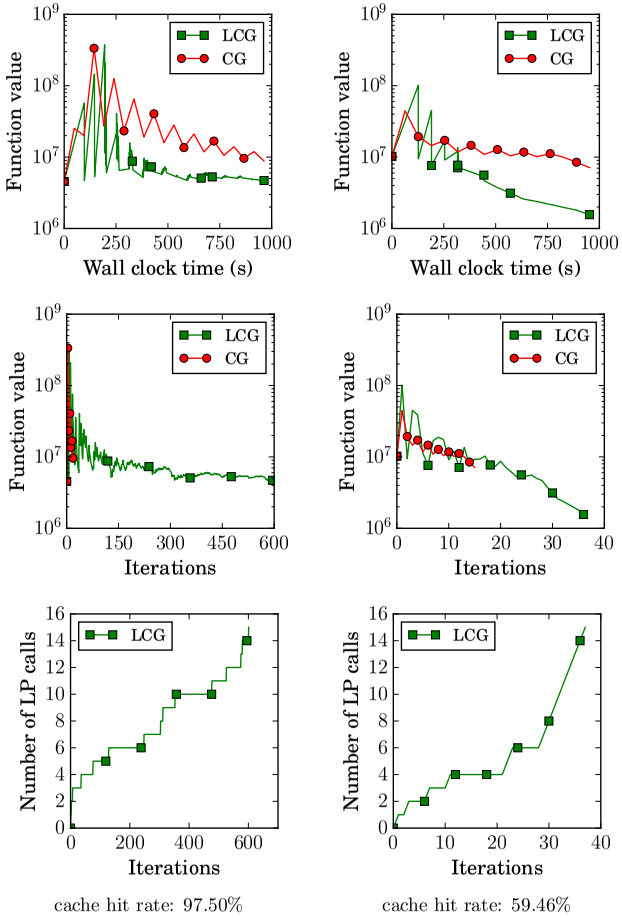


Figure 3: LCG vs. CG on two matrix completion instances. The parameters for Equation (6) are given by $n = 5000$, $m = 4000$, $r = 10$ and $R = 50000$ for the left instance and $n = 100$, $m = 20000$, $r = 10$ and $R = 15000$ for the right instance. In both of these cases the performance of the lazy and the non-lazy version are comparable in iterations, however in wall clock time the lazy version reaches lower function values faster. Note that we used the short-step rule for step sizes for both algorithms as line search for matrix completion is very expensive.

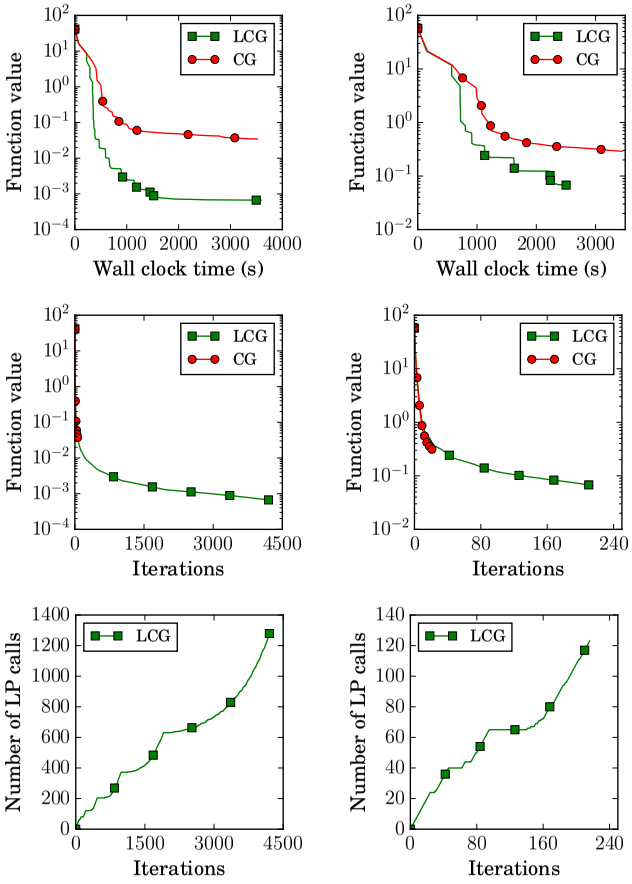


Figure 4: LCG vs. CG on structured regression problems with feasible regions being a TSP polytope over 11 nodes (left) and 12 nodes (right). In both cases LCG is significantly faster in wall-clock time.

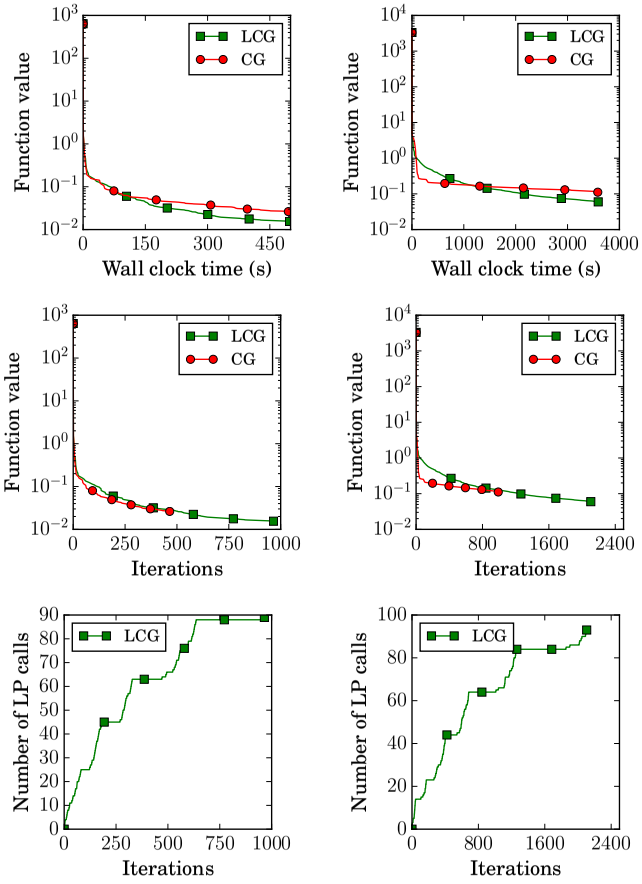


Figure 5: LCG vs. CG on structured regression instances with extended formulation of the spanning tree problem on a 10 node graph on the left and a 15 node graph on the right.

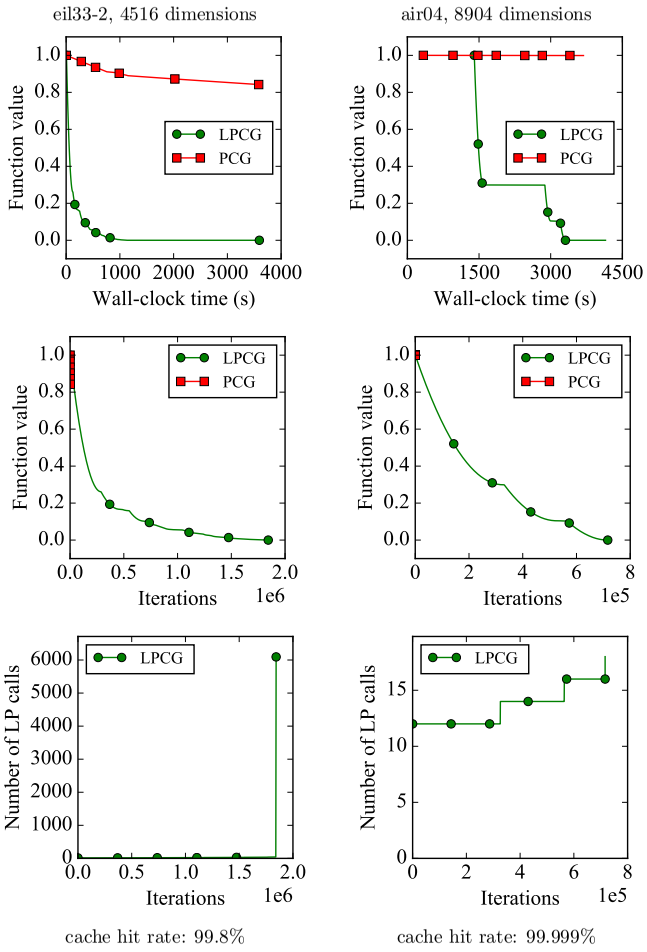


Figure 6: LPCG vs. PCG on two MIPLIB instances `eil33-2` and `air04`. LPCG converges very fast, making millions of iterations with a relatively few oracle calls, while PCG completed only comparably few iterations due to the time-consuming oracle calls. This clearly illustrates the advantage of lazy methods when the cost of linear optimization is non-negligible. On the left, when reaching ε-optimality, LPCG performs many (negative) oracle calls to (re-)prove optimality; at that point one might opt for stopping the algorithm. On the right LPCG needed a rather long time for the initial bound tightening of Φ_0, before converging significantly faster than PCG.

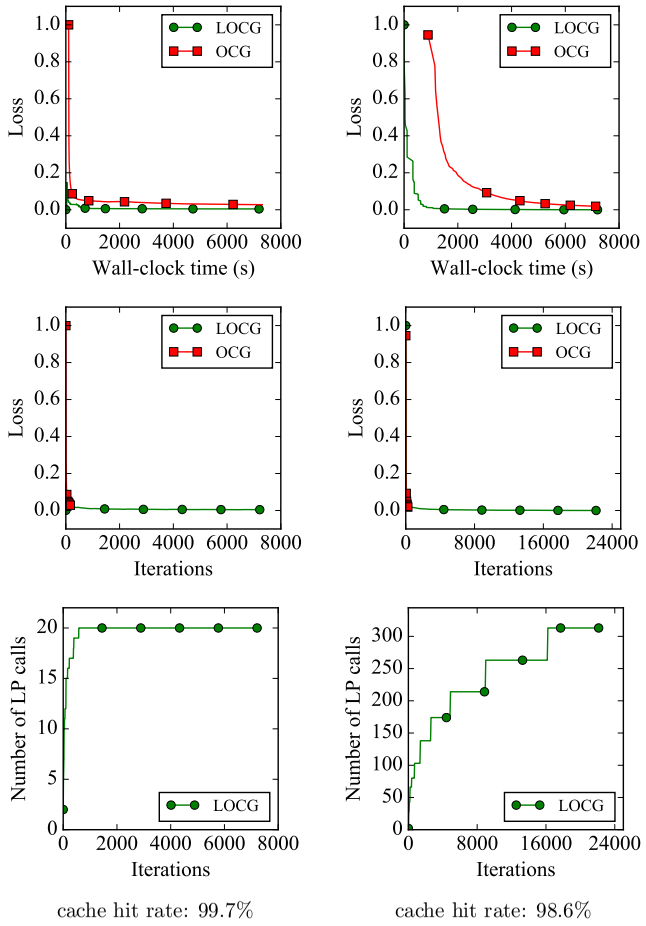


Figure 7: LOCG vs. OCG over cut polytope for a 28-node graph. As for the smaller problem, this also illustrates the advantage of lazy algorithms when linear optimization is expensive. Again, LOCG needed no oracle calls after a small initial amount of time.

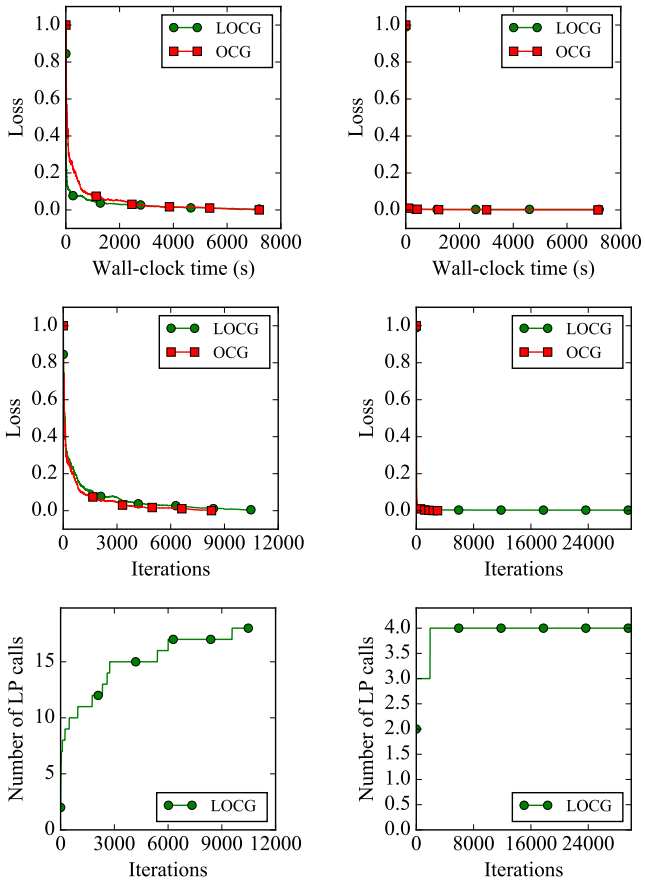


Figure 8: LOCG vs. OCG on a large QUBO instance. Both algorithms converge fast to the optimum. Interestingly, LOCG only performs 4 LP calls.

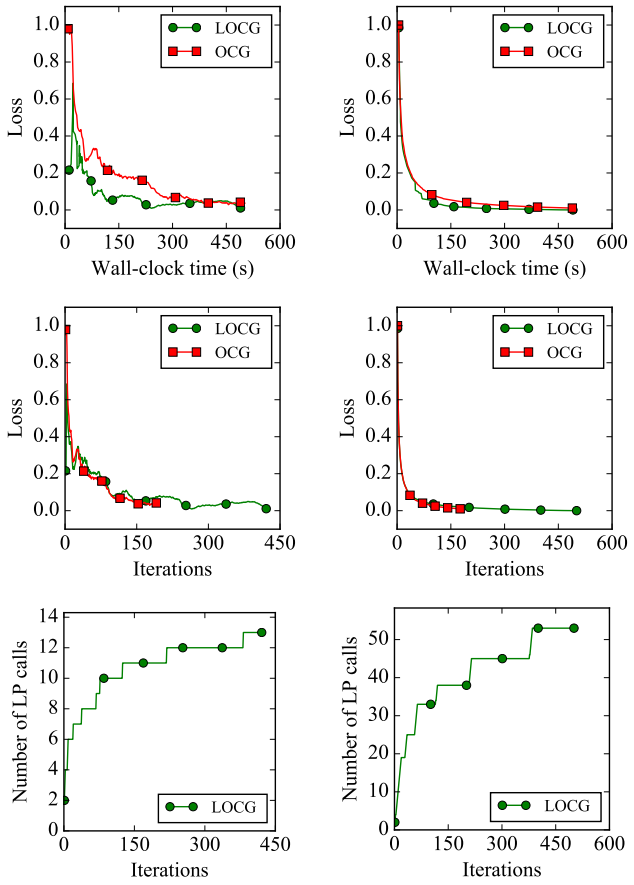


Figure 9: LOCG vs. OCG over a path polytope. Similar convergence rate in the number of iterations, but significant difference in terms of wall-clock time.

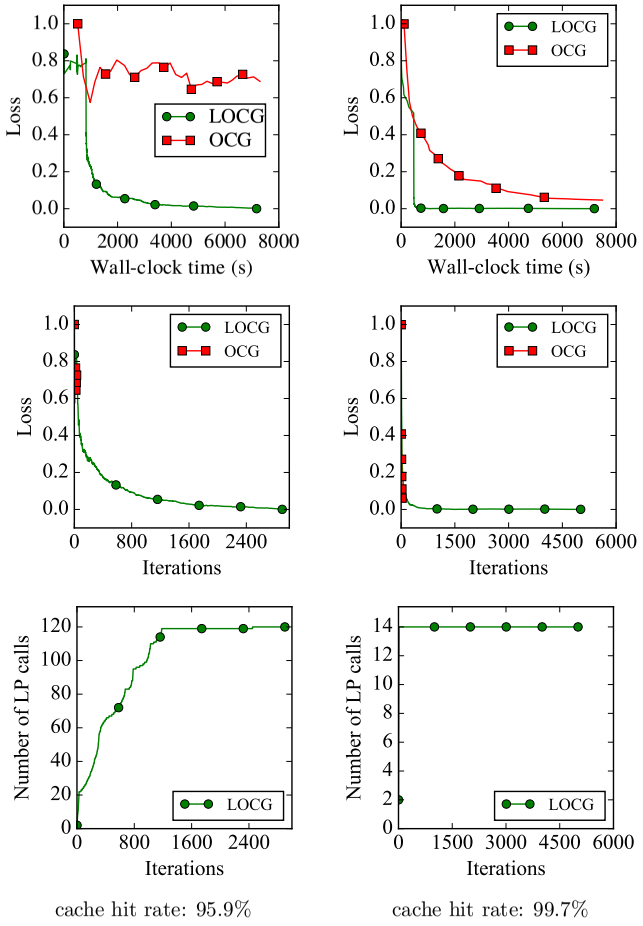


Figure 10: LOCG vs. OCG over a spanning tree instance for a 25-node graph. On the left, early fluctuation can be observed, bearing no consequence for later convergence rate. OCG did not get past this early stage. In both cases LOCG converges significantly faster.

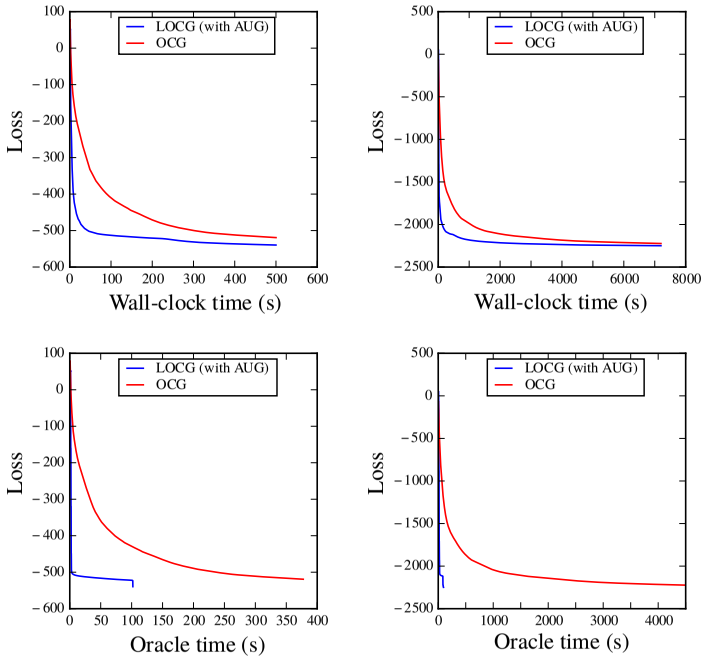


Figure 11: Lazy OCG (with augmentation) vs. OCG on two QUBO instances. In both cases the lazy variant together with augmentation significantly outperforms OCG.

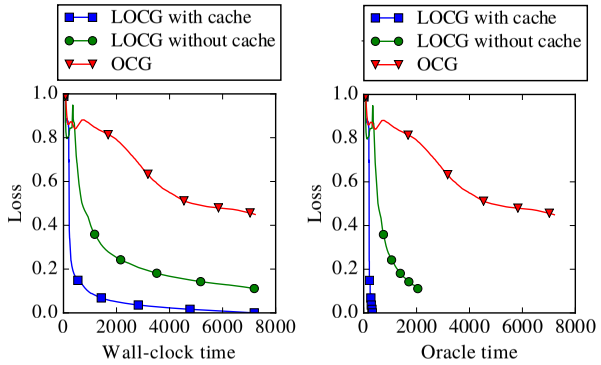


Figure 12: Performance gain due to caching and early termination for online optimization over a maximum cut problem with linear losses. The red line is the OCG baseline, the green one is the lazy variant using only early termination, and the blue one uses caching and early termination. Left: loss vs. wall-clock time. Right: loss vs. total time spent in oracle calls. Time limit was 7200 seconds. Caching allows for a significant improvement in loss reduction in wall-clock time. The effect is even more obvious in oracle time as caching cuts out a large number of oracle calls.

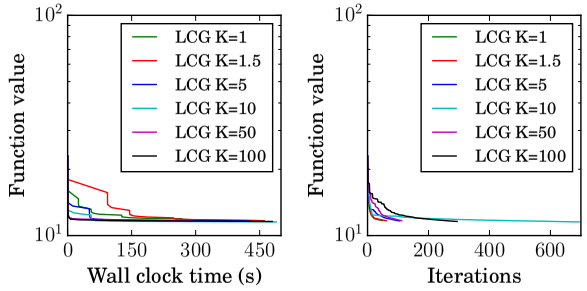


Figure 13: Impact of the oracle approximation parameter K depicted for the Lazy CG algorithm. We can see that increasing K leads to a deterioration of progress in iterations but improves performance in wall-clock time. The behavior is similar for other algorithms.

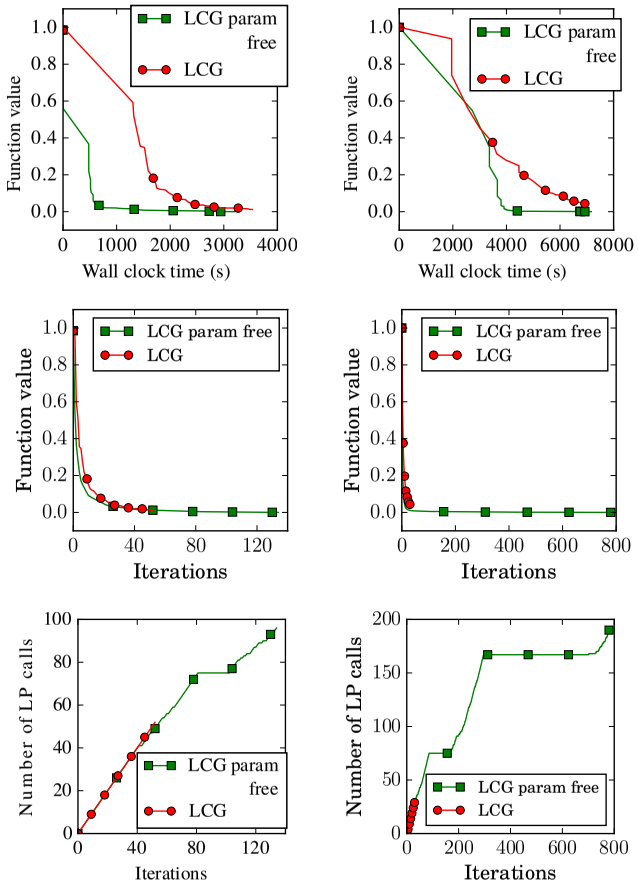


Figure 14: Comparison of the 'textbook' variant of the Lazy CG algorithm (Algorithm 3) vs. the Parameter-free Lazy CG (Algorithm 7) depicted for two sample instances to demonstrate behavior. The parameter-free variant usually has a slightly improved behavior in terms of iterations and a significantly improved behavior in terms of wall-clock performance. In particular, the parameter-free variant can execute significantly more oracle calls, due to the Φ-halving strategy and the associated bounded number of negative calls (see Theorem 9).

Journal of Machine Learning Research 20 (2019) 1-47 Submitted 3/18; Revised 4/19; Published 4/19

Redundancy Techniques for Straggler Mitigation in Distributed Optimization and Learning

Can Karakus CAKARAK@AMAZON.COM
Amazon Web Services
East Palo Alto, CA 94303, USA

Yifan Sun YSUN13@CS.UBC.CA
Department of Computer Science
University of British Columbia
Vancouver, BC, Canada

Suhas Diggavi SUHAS@EE.UCLA.EDU
Department of Electrical and Computer Engineering
University of California, Los Angeles
Los Angeles, CA 90095, USA

Wotao Yin WOTAOYIN@MATH.UCLA.EDU
Department of Mathematics
University of California, Los Angeles
Los Angeles, CA 90095, USA

Editor: Tong Zhang

Abstract

Performance of distributed optimization and learning systems is bottlenecked by "straggler" nodes and slow communication links, which significantly delay computation. We propose a distributed optimization framework where the dataset is "encoded" to have an overcomplete representation with built-in redundancy, and the straggling nodes in the system are dynamically treated as missing, or as "erasures" at every iteration, whose loss is compensated by the embedded redundancy. For quadratic loss functions, we show that under a simple encoding scheme, many optimization algorithms (gradient descent, L-BFGS, and proximal gradient) operating under data parallelism converge to an approximate solution even when stragglers are ignored. Furthermore, we show a similar result for a wider class of convex loss functions when operating under model parallelism. The applicable classes of objectives covers several popular learning problems such as linear regression, LASSO, support vector machine, collaborative filtering, and generalized linear models including logistic regression. These convergence results are deterministic, *i.e.*, they establish sample path convergence for arbitrary sequences of delay patterns or distributions on the nodes, and are independent of the tail behavior of the delay distribution. We demonstrate that equiangular tight frames have desirable properties as encoding matrices, and propose efficient mechanisms for encoding large-scale data. We implement the proposed technique on Amazon EC2 clusters, and demonstrate its performance over several learning problems, including matrix factorization, LASSO, ridge regression and logistic regression, and compare the proposed method with uncoded, asynchronous, and data replication strategies.

∗. Work done while at University of California, Los Angeles.

©2019 Can Karakus, Yifan Sun, Suhas N. Diggavi, Wotao Yin.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-148.html.

Keywords: Distributed optimization, straggler mitigation, proximal gradient, coordinate descent, restricted isometry property

1. Introduction

Solving learning and optimization problems at present scale often requires parallel and distributed implementations to deal with otherwise infeasible computational and memory requirements. However, such distributed implementations often suffer from system-level issues such as slow communication and unbalanced computational nodes. The runtime of many distributed implementations are therefore throttled by that of a few slow nodes, called stragglers, or a few slow communication links, whose delays significantly encumber the overall learning task Dean and Barroso (2013). In this paper, we propose a distributed optimization framework based on proceeding with each iteration without waiting for the stragglers, and encoding the dataset across nodes to add redundancy in the system in order to mitigate the resulting potential performance degradation due to lost updates.

We consider the master-worker architecture, where the dataset is distributed across a set of worker nodes which directly communicate to a master node to optimize a global objective. The encoding framework consists of an efficient linear transformation (coding) of the dataset that results in an overcomplete representation, which is then partitioned and distributed across the worker nodes. The distributed optimization algorithm is then performed directly on the encoded data, with all worker nodes oblivious to the encoding scheme, *i.e.*, no explicit decoding of the data is performed, and nodes simply solve the effective optimization problem on the encoded data. In order to mitigate the effect of stragglers, in each iteration, the master node only waits for the first k updates to arrive from the m worker nodes (where $k \leq m$ is a design parameter) before moving on; the remaining $m - k$ node results are treated as missing (erasures), whose loss is compensated by the data encoding.

The framework is applicable to both the data parallelism and model parallelism paradigms of distributed learning, and can be applied to distributed implementations of several popular optimization algorithms, including gradient descent, limited-memory-BFGS, proximal gradient, and block coordinate descent. Under data parallelism, we focus on quadratic loss functions with arbitrary convex regularization terms or arbitrary convex constraint sets, whereas for model parallelism, we consider general convex objectives where the vector of parameters act linearly on input data, such as logistic regression. We show that if the linear transformation is designed to satisfy a spectral condition resembling the restricted isometry property, the iterates resulting from the encoded version of these algorithms deterministically converge to an exact solution for the case of model paralellism, and an approximate one under data parallelism, where the approximation quality only depends on the properties of encoding and the parameter k. These convergence guarantees are deterministic in the sense that they hold for any pattern of node delays, *i.e.*, even if an adversary chooses which nodes to delay at every iteration. In addition, the convergence behavior is independent of the tail behavior of the node delay distribution, and can tolerate unbounded delay. Such a worst-case guarantee is not possible for the asynchronous versions of these algorithms, whose convergence rates deteriorate with increasing node delays. We point out that our approach is particularly suited to computing networks with a high degree of variability and

unpredictability, where a large number of nodes can delay their computations for arbitrarily long periods of time.

Our contributions are as follows: (i) we propose the encoded distributed optimization framework, and prove deterministic convergence guarantees under this framework for gradient descent, L-BFGS, proximal gradient and block coordinate descent algorithms; (ii) we provide three classes of encoding matrices, and discuss their properties, and describe how to efficiently encode with such matrices on large-scale data; (iii) we implement the proposed technique on Amazon EC2 clusters and compare their performance to uncoded, replication, and asynchronous strategies for problems such as ridge regression, collaborative filtering, logistic regression, and LASSO. In these tasks we show that in the presence of stragglers, the technique can result in significant speed-ups (specific amounts depend on the underlying system, and examples are provided in Section 5) compared to the uncoded case when we wait for all workers in each iteration, to achieve the same test error.

Related work. The approaches to mitigating the effect of stragglers can be broadly classified into three categories: replication-based techniques, asynchronous optimization, and coding-based techniques.

Replication-based techniques consist of either re-launching a certain task if it is delayed, or pre-emptively assigning each task to multiple nodes and moving on with the copy that completes first. Such techniques have been proposed and analyzed in Gardner et al. (2015); Ananthanarayanan et al. (2013); Shah et al. (2016); Wang et al. (2015); Yadwadkar et al. (2016), among others. Our framework does not preclude the use of such system-level strategies, which can still be built on top of our encoded framework to add another layer of robustness against stragglers. However, it is not possible to achieve the worst-case guarantees provided by encoding with such schemes, since it is still possible for both replicas to be delayed.

Perhaps the most popular approach in distributed learning to address the straggler problem is asynchronous optimization, where each worker node asynchronously pushes updates to and fetches iterates from a parameter server independently of other workers, hence the stragglers do not hold up the entire computation. This approach was studied in Recht et al. (2011); Agarwal and Duchi (2011); Dean et al. (2012); Li et al. (2014) (among many others) for the case of data parallelism, and Liu et al. (2015); You et al. (2016); Peng et al. (2016); Sun et al. (2017) for coordinate descent methods (model parallelism). Although this approach has been largely successful, all asynchronous convergence results depend on either a hard bound on the allowable delays on the updates, or a bound on the moments of the delay distribution, and the resulting convergence rates explicitly depend on such bounds. In contrast, our framework allows for completely unbounded delays. Further, as in the case of replication, one can still consider asynchronous strategies on top of the encoding, although we do not focus on such techniques within the scope of this paper.

A more recent line of work that address the straggler problem is based on coding-theory-inspired techniques Tandon et al. (2017); Lee et al. (2018); Dutta et al. (2016); Karakus et al. (2017a,b); Yang et al. (2017); Halbawi et al. (2017); Reisizadeh et al. (2017). Some of these works focus exclusively on coding for distributed linear operations, which are considerably simpler to handle. The works in Tandon et al. (2017); Halbawi et al. (2017) propose coding techniques for distributed gradient descent that can be applied more generally. However,

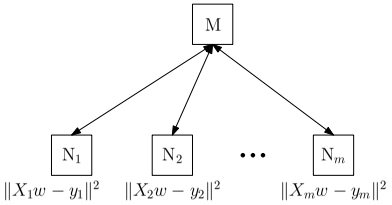


Figure 1: Uncoded distributed optimization with data parallelism, where X and y are partitioned as $X = [X_i]_{i\in[m]}$ and $y = [y_i]_{i\in[m]}$.

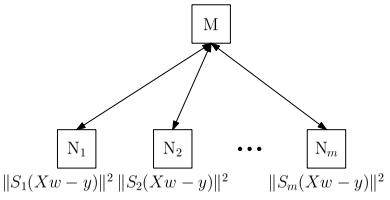


Figure 2: Encoded setup with data parallelism, where node i stores (S_iX, S_iy), instead of (X_i, y_i). The uncoded case corresponds to $S = I$.

the approach proposed in these works require a redundancy factor of $r + 1$ in the code, to mitigate r stragglers. Our approach relaxes the exact gradient recovery requirement of these works, consequently reducing the amount of redundancy required by the code.

The proposed technique, especially under data parallelism, is also closely related to randomized linear algebra and sketching techniques in Mahoney et al. (2011); Drineas et al. (2011); Pilanci and Wainwright (2015), used for dimensionality reduction of large convex optimization problems. The main difference between this literature and the proposed coding technique is that the former focuses on reducing the problem dimensions to lighten the computational load, whereas encoding *increases* the dimensionality of the problem to provide robustness. As a result of the increased dimensions, coding can provide a much closer approximation to the original solution compared to sketching techniques. In addition, unlike these works, our model allows for an arbitrary convex regularizer in addition to the encoded loss term.

2. Encoded Distributed Optimization

We will use the notation $[j] = \{i \in \mathbb{Z} : 1 \leq i \leq j\}$. All vector norms refer to 2-norm, and all matrix norms refer to spectral norm, unless otherwise noted. The superscript ^{c} will refer to complement of a subset, *i.e.*, for $A \subseteq [m]$, $A^c = [m]\backslash A$. For a sequence of matrices $\{M_i\}$ and a set A of indices, we will denote $[M_i]_{i\in A}$ to mean the matrix formed by stacking the matrices M_i vertically. The main notation used throughout the paper is provided in Table 1.

We consider a distributed computing network where the dataset $\{(x_i, y_i)\}_{i=1}^n$ is stored across a set of m worker nodes, which directly communicate with a single master node. In practice the master node can be implemented using a fully-connected set of nodes, but this can still be abstracted as a single master node.

It is useful to distinguish between two paradigms of distributed learning and optimization; namely, data parallelism, where the dataset is partitioned across data samples, and model parallelism, where it is partitioned across features (see Figures 1 and 3). We will describe these two models in detail next.

Notation	Explanation
$[j]$	The set $\{i \in \mathbb{Z} : 1 \leq i \leq j\}$
m	Number of worker nodes
n, p	The dimensions of the data matrix $X \in \mathbb{R}^{n\times p}$, vector $y \in \mathbb{R}^{n\times 1}$
k_t	Number of updates the master node waits for in iteration t, before moving on
η_t	Fraction of nodes waited for in iteration, *i.e.*, $\eta_t = \frac{k_t}{m}$
A_t	The subset of nodes $[m]$ which send the fastest k_t updates at iteration t
$f(w), \widetilde{f}(w)$	The original and encoded objectives, respectively, under data parallelism
$g(w) = \phi(Xw)$	The original objective under model parallelism
$\widetilde{g}(v) = \phi(XS^\top v)$	The encoded objective under model parallelism
$h(w)$	Regularization function (potentially non-smooth)
ν	Strong convexity parameter
L	Smoothness parameter for $h(w)$ (if smooth), and $g(w)$
λ	Regularization parameter
Ψ_t	Mapping from gradient updates to step $\{\nabla f_i(t)\}_{i\in A_t} \mapsto d_t$
d_t	Descent direction chosen by the algorithm
α_t, α	Step size
M, μ	Largest and smallest eigenvalues of $X^\top X$, respectively
β	Redundancy factor ($\beta \geq 1$)
S	Encoding matrix with dimensions $\beta n \times n$
S_i	ith row-block of S, corresponding to worker i
S_A	Submatrix of S formed by $\{S_i\}_{i\in A\subseteq[m]}$ stacked vertically

Table 1: Notation used in the paper.

2.1. Data parallelism

We focus on objectives of the form

$$f(w) = \frac{1}{2n}\|Xw - y\|^2 + \lambda h(w), \tag{1}$$

where $X \in \mathbb{R}^{n\times p}$ and $y \in \mathbb{R}^n$ are the data matrix and data vector, respectively. We assume each row of X corresponds to a data sample, and the data samples and response variables can be horizontally partitioned as $X = \begin{bmatrix} X_1^\top & X_2^\top & \cdots & X_m^\top \end{bmatrix}^\top$ and $y = \begin{bmatrix} y_1^\top & y_2^\top & \cdots & y_m^\top \end{bmatrix}^\top$. In the uncoded setting, machine i stores the row-block X_i (Figure 1). We denote the largest and smallest eigenvalues of $X^\top X$ with $M > 0$, and $\mu \geq 0$, respectively. We assume $\lambda \geq 0$, and $h(w) \geq 0$ is a convex, extended real-valued function of w that does not depend on data. Since $h(w)$ can take the value $h(w) = \infty$, this model covers arbitrary convex constraints on the optimization. Several important learning problems, such as ridge regression, LASSO, collaborative filtering (solved via alternating minimization), and support vector machine[1] fall within this formulaton.

The encoding consists of solving the proxy problem

$$\widetilde{f}(w) = \frac{1}{2n}\|S\left(Xw - y\right)\|^2 + \lambda h(w) = \frac{1}{2n}\sum_{i=1}^{m} \underbrace{\|S_i\left(Xw - y\right)\|^2}_{f_i(w)} + \lambda h(w), \tag{2}$$

1. The dual formulation of SVM with squared hinge loss can be solved as a quadratic minimization over the simplex (Jaggi (2013); Li et al. (2009))

instead, where $S \in \mathbb{R}^{\beta n\times n}$ is a designed encoding matrix with redundancy factor $\beta \geq 1$, partitioned as $S = \begin{bmatrix} S_1^\top & S_2^\top & \cdots & S_m^\top \end{bmatrix}^\top$ across m machines. Based on this partition, worker node i stores (S_iX, S_iy), and operates to solve the problem (2) in place of (1) (Figure 2). We will denote $\hat{w} \in \arg\min \widetilde{f}(w)$, and $w^* \in \arg\min f(w)$.

In general, the regularizer $h(w)$ can be non-smooth. We will say that $h(w)$ is L-smooth if $\nabla h(w)$ exists everywhere and satisfies

$$h(w') \leq h(w) + \langle \nabla h(w), w' - w\rangle + \frac{L}{2}\|w' - w\|^2$$

for some $L > 0$, for all w, w'. The objective f is ν-strongly convex if, for all x, y,

$$f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{\nu}{2}\|x - y\|^2.$$

Once the encoding is done and appropriate data is stored in the nodes, the optimization process works in iterations. At iteration t, the master node broadcasts the current iterate w_t to the worker nodes, and wait for k_t gradient updates $\nabla f_i(w)$ to arrive, corresponding to that iteration, and then chooses a step direction d_t and a step size α_t (based on algorithm Ψ_t that maps the set of gradients updates to a step) to update the parameters. We will denote $\eta_t = \frac{k_t}{m}$. We will also drop the time dependence of k and η whenever it is kept constant.

The set of fastest k_t nodes to send gradients for iteration t will be denoted as A_t. Once k_t updates have been collected, the remaining nodes, denoted A_t^c, are interrupted by the master node[2]. Algorithms 1 and 2 describe the generic mechanism of the proposed distributed optimization scheme at the master node and a generic worker node, respectively.

The intuition behind the encoding idea is that waiting for only $k_t < m$ workers prevents the stragglers from holding up the computation, while the redundancy provided by using a tall matrix S compensates for the information lost by proceeding without the updates from stragglers (the nodes in the subset A_t^c).

We next describe the three specific algorithms that we consider under data parallelism, to compute d_t.

Gradient descent. In this case, we assume that $h(w)$ is L-smooth. Then we simply set the descent direction

$$d_t = -\left(\frac{1}{2n\eta}\sum_{i\in A_t} \nabla f_i(w_t) + \lambda \nabla h(w_t)\right).$$

We keep $k_t = k$ constant, chosen based on the number of stragglers in the network, or based on the desired operating regime.

2. If the communication is already in progress at the time when k_t faster gradient updates arrive, the communication can be finished without interruption, and the late update can be dropped upon arrival. Otherwise, such interruption can be implemented by having the master node send an interrupt signal, and having one thread at each worker node keep listening for such a signal.

Limited-memory-BFGS. We assume that $h(w) = \|w\|^2$, and assume $\mu + \lambda > 0$. Although L-BFGS is traditionally a batch method, requiring updates from all nodes, its stochastic variants have also been proposed by Mokhtari and Ribeiro (2015); Berahas et al. (2016). The key modification to ensure convergence in this case is that the Hessian estimate must be computed via gradient components that are common in two consecutive iterations, *i.e.*, from the nodes in $A_t \cap A_{t-1}$. We adapt this technique to our scenario. For $t > 0$, define $u_t := w_t - w_{t-1}$, and[3]

$$r_t := \frac{m}{2n\left|A_t \cap A_{t-1}\right|} \sum_{i \in A_t \cap A_{t-1}} \left(\nabla f_i(w_t) - \nabla f_i(w_{t-1})\right).$$

Then once the gradient terms $\{\nabla f_i(w_t)\}_{i \in A_t}$ are collected, the descent direction is computed by $d_t = -B_t \widetilde{g}_t$, where $\widetilde{g}_t = \frac{1}{2\eta n} \sum_{i \in A_t} \nabla f_i(w_t)$, and B_t is the inverse Hessian estimate for iteration t, which is computed by

$$B_t^{(\ell+1)} = V_{j_{\ell,t}}^\top B_t^{(\ell)} V_{j_{\ell,t}} + \rho_{j_{\ell,t}} u_{j_{\ell,t}} u_{j_{\ell,t}}^\top, \quad \rho_j = \frac{1}{r_j^\top u_j}, \quad V_j = I - \rho_j r_j u_j^\top$$

with $j_{\ell,t} = t - \widetilde{\sigma} + \ell$, $B_t^{(0)} = \frac{r_t^\top r_t}{r_t^\top u_t} I$, and $B_t := B_t^{(\widetilde{\sigma})}$ with $\widetilde{\sigma} := \min\{t, \sigma\}$, where σ is the L-BFGS memory length. Once the descent direction d_t is computed, the step size is determined through exact line search[4]. To do this, each worker node computes $S_i X d_t$, and sends it to the master node. Once again, the master node only waits for the fastest k_t nodes, denoted by $D_t \subseteq [m]$ (where in general $D_t \neq A_t$), to compute the step size that minimizes the function along d_t, given by

$$\alpha_t = -\rho \frac{d_t^\top \widetilde{g}_t}{d_t^\top \widetilde{X}_D^\top \widetilde{X}_D d_t}, \tag{3}$$

where $\widetilde{X}_D = [S_i X]_{i \in D_t}$, and $0 < \rho < 1$ is a back-off factor of choice.

Proximal gradient. Here, we consider the general case of non-smooth $h(w) \geq 0, \lambda \geq 0$. The descent direction d_t is given by

$$d_t = \arg\min_w \widetilde{F}_t(w) - w_t,$$

where

$$\widetilde{F}_t(w) := \frac{1}{2\eta n} \sum_{i \in A_t} f_i(w_t) + \left\langle \frac{1}{2\eta n} \sum_{i \in A_t} \nabla f_i(w_t), w - w_t \right\rangle + \lambda h(w) + \frac{1}{2\alpha} \|w - w_t\|^2.$$

We keep the step size $\alpha_t = \alpha$ and $k_t = k$ constant.

3. Note that we assume here that the intersection $A_t \cap A_{t-1}$ is non-empty. This can be ensured by adaptively choosing k_t (waiting for sufficiently many workers) so that $A_t \cap A_{t-1} \neq \varnothing$. This is further discussed in Section 3. However, our experiments indicate that convergence is still possible even when this condition does not hold.
4. Note that exact line search is not more expensive than backtracking line search for a quadratic loss, since it only requires a single matrix-vector multiplication.

Algorithm 1 Generic encoded distributed optimization procedure under data parallelism, at the master node.

Given: Ψ_t, a sequence of functions that map gradients $\{\nabla f_i(w_t)\}_{i \in A_t}$ to a descent direction d_t
Initialize w_0, α_0
for $t = 1, \ldots, T$ **do**
  broadcast w_t to all worker nodes
  wait to receive k_t gradient updates $\{\nabla f_i(w_t)\}_{i \in A_t}$
  send interrupt signal to the nodes in A_t^c
  compute the descent direction $d_t = \Psi_t\left(\{\nabla f_i(w_t)\}_{i \in A_t}\right)$
  determine step size α_t
  take the step $w_{t+1} = w_t + \alpha_t d_t$
end for

Algorithm 2 Generic encoded distributed optimization procedure under data parallelism, at worker node i.

Given: $f_i(w) = \|S_i(Xw - y)\|^2$
for $t = 1, \ldots, T$ **do**
  wait to receive w_t
  while not interrupted by master **do**
    compute $\nabla f_i(w_t)$
  end while
  if computation was interrupted **then**
    continue
  else
    send $\nabla f_i(w_t)$
  end if
end for

2.2. Model parallelism

Under the model parallelism paradigm, we focus on objectives of the form

$$\min g(w) := \min_w \phi(Xw) = \min_w \phi\left(\sum_{i=1}^m X_i w_i\right), \tag{4}$$

where the data matrix is partitioned as $X = [X_1\, X_2\, \cdots\, X_m]$, the parameter vector is partitioned as $w = \left[w_1^\top\, w_2^\top\, \cdots\, w_m^\top\right]^\top$, ϕ is convex, and $g(w)$ is L-smooth. Note that the data matrix X is partitioned horizontally, meaning that the dataset is split across features, instead of data samples (see Figure 3). This class of objectives is applicable to any classification or regression problem with generalized linear models, such as logistic regression, softmax regression, and multinomial regression.

We encode the problem (4) by setting $w = S^\top v$, and solving the problem

$$\min_v \widetilde{g}(v) := \phi\left(XS^\top v\right) = \min_v \phi\left(\sum_{i=1}^m X S_i^\top v_i\right), \tag{5}$$

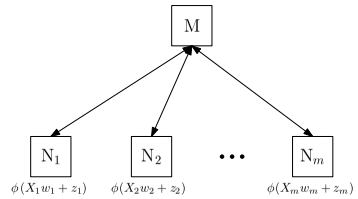


Figure 3: Uncoded distributed optimization with model parallelism, where ith node stores the ith partition of the model w_i. For $i = 1, \ldots, m$, $z_i = \sum_{j \neq i} X_j w_j$.

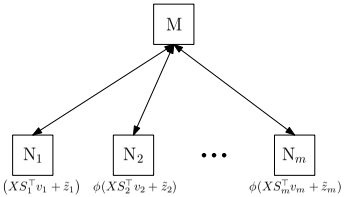


Figure 4: Encoded setup with model parallelism, where ith node stores the partition v_i of the model in the "lifted" space. For $i = 1, \ldots, m$, $\widetilde{z}_i = \sum_{j \neq i} u_j = \sum_{j \neq i} X S_j^\top v_j$.

where $w \in \mathbb{R}^p$ and $S^\top = \begin{bmatrix} S_1^\top & S_2^\top & \cdots & S_m^\top \end{bmatrix} \in \mathbb{R}^{p \times \beta p}$ (see Figure 4). As a result, worker i stores the column-block $XS_i^\top$, as well as the iterate partition v_i. Note that we increase the dimensions of the parameter vector by multiplying the dataset X with a wide encoding matrix $S^\top$ from the right, and as a result we have redundant coordinates in the system. As in the case of data parallelism, such redundant coordinates provide robustness against erasures arising due to stragglers. Such increase in coordinates means that the problem is simply lifted onto a larger dimensional space, while preserving the original geometry of the problem. We will denote $u_{i,t} = XS_i^\top v_{i,t}$, where $v_{i,t}$ is the parameter iterates of worker i at iteration t. In order to compute updates to its parameters v_i, worker i needs the up-to-date value of $\widetilde{z}_i := \sum_{j \neq i} u_j$, which is provided by the master node at every iteration.

Let $\mathcal{S} = \arg\min_w g(w)$, and given w, let w^* be the projection of w onto $\mathcal{S}$. We will say that $g(w)$ satisfies ν-restricted-strong convexity (Lai and Yin (2013)) if

$$\langle \nabla g(w), w - w^* \rangle \geq \nu \|w - w^*\|^2$$

for all w. Note that this is weaker than (implied by) strong convexity since w^* is restricted to be the projection of w, but unlike strong convexity, it is satisfied under the case where ϕ is strongly convex, but X has a non-trivial null space, *e.g.*, when it has more columns than rows.

For a given $w \in \mathbb{R}^p$, we define the level set of g at w as $D_g(w) := \{w' : g(w') \leq g(w)\}$. We will say that the level set at w_0 has diameter R if

$$\sup \left\{ \|w - w'\| : w, w' \in D_g(w_0) \right\} \leq R.$$

As in the case of data parallelism, we assume that the master node waits for k updates at every iteration, and then moves onto the next iteration (see Algorithms 3 and 4). We similarly define A_t as the set of k fastest nodes in iteration t, and also define

$$I_{i,t} = \begin{cases} 1 & i \in A_t \\ 0 & i \notin A_t. \end{cases}$$

Algorithm 3 Encoded block coordinate descent at worker node i.

1: Given: X_i, v_i.
2: **for** $t = 1, \ldots, T$ **do**
3: wait to receive $(I_{i,t-1}, \widetilde{z}_{i,t})$
4: **if** $I_{i,t} == 1$ **then**
5: take step $v_{i,t} = v_{i,t-1} + d_{i,t-1}$
6: **else**
7: set $v_{i,t} = v_{i,t-1}$
8: **end if**
9: **while** not interrupted by master **do**
10: compute next step $d_{i,t} = \alpha S_i X^\top \nabla \phi \left(X S_i^\top v_{i,t} + \widetilde{z}_{i,t} \right)$
11: compute $u_{i,t} = X S_i^\top v_{i,t}$
12: **end while**
13: **if** computation was interrupted **then**
14: continue
15: **else**
16: send $u_{i,t}$ to master node
17: **end if**
18: **end for**

Algorithm 4 Encoded block coordinate descent at the master node.

1: **for** $t = 1, \ldots, T$ **do**
2: **for** $i = 1, \ldots, m$ **do**
3: send $(I_{i,t-1}, \widetilde{z}_{i,t})$ to worker i
4: **end for**
5: wait to receive k updated parameters $\{u_{i,t}\}_{i \in A_t}$
6: send interrupt signal to the nodes in A_t^c
7: set $u_{i,t} = u_{i,t-1}$ for $i \in A_t^c$
8: compute $\widetilde{z}_{i,t} = \sum_{j \neq i} u_{j,t}$ for all i
9: **end for**

Under model parallelism, we consider block coordinate descent, described in Algorithm 3, where worker i stores the current values of the partition v_i, and performs updates on it, given the latest values of the rest of the parameters. The parameter estimate at time t is denoted by $v_{i,t}$, and we also define $\widetilde{z}_{i,t} = \sum_{j \neq i} u_{i,t} = \sum_{j \neq i} X S_j^\top v_j$. The iterates are updated by

$$v_{i,t} - v_{i,t-1} = \Delta_{i,t} := \begin{cases} -\alpha \nabla_i \widetilde{g}(v_{t-1}), & \text{if } i \in A_t \\ 0, & \text{otherwise,} \end{cases}$$

for a step size parameter $\alpha > 0$, where ∇_i refers to gradient only with respect to the variables v_i, *i.e.*, $\nabla \widetilde{g} = [\nabla_i \widetilde{g}]_{i \in [m]}$. Note that if $i \notin A_t$ then v_i does not get updated in worker i, which ensures the consistency of parameter values across machines. This is achieved by lines 4–8 in Algorithm 3. Worker i learns about this in the next iteration, when $I_{i,t-1}$ is sent by the master node.

3. Main Theoretical Results: Convergence Analysis

In this section, we prove convergence results for the algorithms described in Section 2. Note that since we modify the original optimization problem and solve it obliviously to this change, it is not obvious that the solution has any optimality guarantees with respect to the original problem. We show that, it is indeed possible to provide convergence guarantees in terms of the *original* objective under the encoded setup.

3.1. A spectral condition

In order to show convergence under the proposed framework, we require the encoding matrix S to satisfy a certain spectral criterion on S. Let S_A denote the submatrix of S associated with the subset of machines A, *i.e.*, $S_A = [S_i]_{i\in A}$. Then the criterion in essence requires that for any sufficiently large subset A, S_A behaves approximately like a matrix with orthogonal columns. We make this precise in the following statement.

Definition 1 *Let $\beta \geq 1$, and $\frac{1}{\beta} \leq \eta \leq 1$ be given. A matrix $S \in \mathbb{R}^{\beta n\times n}$ is said to satisfy the (m,η,ϵ)-block-restricted isometry property ((m,η,ϵ)-BRIP) if for any $A \subseteq [m]$ with $|A| \geq \eta m$,*

$$(1-\epsilon)I_n \preceq \frac{1}{\eta}S_A^\top S_A \preceq (1+\epsilon)I_n. \tag{6}$$

Note that this is similar to the restricted isometry property used in compressed sensing (Candes and Tao (2005)), except that we do not require (6) to hold for every submatrix of S of size $\mathbb{R}^{\eta n\times n}$. Instead, (6) needs to hold only for the submatrices of the form $S_A = [S_i]_{i\in A}$, which is a less restrictive condition. In general, it is known to be difficult to analytically prove that a structured, deterministic matrix satisfies the general RIP condition. Such difficulty extends to the BRIP condition as well. However, it is known that i.i.d. sub-Gaussian ensembles and randomized Fourier ensembles satisfy this property (Candes and Tao (2006)). In addition, numerical evidence suggests that there are several families of constructions for S whose submatrices have eigenvalues that mostly tend to concentrate around 1. We point out that although the strict BRIP condition is required for the theoretical analysis, in practice the algorithms perform well as long as the bulk of the eigenvalues of S_A lie within a small interval $(1-\epsilon, 1+\epsilon)$, even though the extreme eigenvalues may lie outside of it (in the non-adversarial setting). In Section 4, we explore several classes of matrices and discuss their relation to this condition.

3.2. Convergence of encoded gradient descent

We first consider the algorithms described under data parallelism architecture. The following theorem summarizes our results on the convergence of gradient descent for the encoded problem.

Theorem 2 *Let w_t be computed using encoded gradient descent with an encoding matrix that satisfies (m,η,ϵ)-BRIP, with step size $\alpha_t = \frac{2\zeta}{M(1+\epsilon)+\lambda L}$ for some $0 < \zeta \leq 1$, for all t. Let $\{A_t\}$ be an arbitrary sequence of subsets of $[m]$ with cardinality $|A_t| \geq \eta m$ for all t. Then, for f as given in (1),*

1.

$$\frac{1}{t}\sum_{\tau=1}^{t} f(w_\tau) - \kappa_1 f(w^*) \leq \frac{4\epsilon f(w_0) + \frac{1}{2\alpha}\|w_0 - w^*\|^2}{(1-7\epsilon)\,t}$$

2. If f is in addition ν-strongly convex, then

$$f(w_t) - \frac{\kappa_2^2(\kappa_2-\gamma)}{1-\kappa_2\gamma} f(w^*) \leq (\kappa_2\gamma)^t f(w_0), \quad t = 1, 2, \ldots,$$

where $\kappa_1 = \frac{1+3\epsilon}{1-7\epsilon}$, $\kappa_2 = \frac{1+\epsilon}{1-\epsilon}$, and $\gamma = \left(1 - \frac{4\nu\zeta(1-\zeta)}{M(1+\epsilon)+\lambda L}\right)$, where ϵ is assumed to be small enough so that $\kappa_2\gamma < 1$.

The proof is provided in Appendix A, which relies on the fact that the solution to the effective "instantaneous" problem corresponding to the subset A_t lies in a bounded set $\{w : f(w) \leq \kappa f(w^*)\}$ (where κ depends on the encoding matrix and strong convexity assumption on f), and therefore each gradient descent step attracts the iterate towards a point in this set, which must eventually converge to this set. Theorem 2 shows that encoded gradient descent can achieve the standard $O\left(\frac{1}{t}\right)$ convergence rate for the general case[5], and linear convergence rate for the strongly convex case, up to an approximate minimum. For the convex case, the convergence is shown on the running mean of past function values, whereas for the strongly convex case we can bound the function value at every step. Note that although the nodes actually minimize the encoded objective $\widetilde{f}(w)$, the convergence guarantees are given in terms of the original objective $f(w)$. Note that under the strongly convex case, linear convergence requires condition $\kappa_2\gamma < 1$. Since γ, which is slightly smaller than 1, is a decreasing function of the fraction of the strong convexity parameter to the smoothness parameter, the practical implication of this is that the less the curvature of the function (the smaller the strong convexity parameter ν), the more workers we need to wait in each iteration, so that κ_2 is small enough to satisfy $\kappa_2\gamma < 1$, which ensures the linear convergence guarantee of the theorem. Note that even if this condition is not satisfied, the algorithm can still converge, but not necessarily at the linear rate promised by the theorem.

Theorem 2 provides deterministic, sample path convergence guarantees under any (adversarial) sequence of active nodes $\{A_t\}$, which is in contrast to the stochastic methods, which show convergence typically in expectation. Further, the convergence rate is not affected by the tail behavior of the delay distribution, since the delayed updates of stragglers are not applied to the iterates.

Note that since we do not seek exact solutions under data parallelism, we can keep the redundancy factor β fixed regardless of the number of stragglers. Increasing number of stragglers in the network simply results in a looser approximation of the solution, allowing for a graceful degradation. This is in contrast to existing work Tandon et al. (2017) seeking exact convergence under coding, which shows that the redundancy factor must grow linearly with the number of stragglers.

5. Since the convergence result is deterministic, the variance reduction factor $\frac{1}{m}$ present in stochastic algorithms does not appear in our convergence rate.

3.3. Convergence of encoded L-BFGS

We consider the variant of L-BFGS described in Section 2. For our convergence result for L-BFGS, we need another assumption on the matrix S, in addition to (6). Defining $\breve{S}_t = [S_i]_{i\in A_t\cap A_{t-1}}$ for $t > 0$, we assume that for some $\delta > 0$,

$$\delta I \preceq \breve{S}_t^\top \breve{S}_t \tag{7}$$

for all $t > 0$. Note that this requires that one should wait for sufficiently many nodes to send updates so that the overlap set $A_t \cap A_{t-1}$ has more than $\frac{1}{\beta}$ nodes, and thus the matrix $\breve{S}_t$ can be full rank. When the columns of X are linearly independent, this is satisfied if $\eta \geq \frac{1}{2} + \frac{1}{2\beta}$ in the worst-case, and in the case where node delays are i.i.d. across machines, it is satisfied in expectation if $\eta \geq \frac{1}{\sqrt{\beta}}$. One can also choose k_t adaptively so that $k_t = \min\left\{k : |A_t(k) \cap A_{t-1}| > \frac{1}{\beta}\right\}$. We note that although this condition is required for the theoretical analysis, the algorithm may perform well in practice even when this condition is not satisfied.

We first show that this algorithm results in stable inverse Hessian estimates under the proposed model, under arbitrary realizations of $\{A_t\}$ (of sufficiently large cardinality), which is done in the following lemma.

Lemma 3 *Let $\mu + \lambda > 0$. Then there exist constants $c_1, c_2 > 0$ such that for all t, the inverse Hessian estimate B_t satisfies $c_1 I \preceq B_t \preceq c_2 I$.*

The proof, provided in Appendix A, is based on the well-known trace-determinant method. Using Lemma 3, we can show the following convergence result.

Theorem 4 *Let $\mu + \lambda > 0$, and let w_t be computed using the L-BFGS method described in Section 2, with an encoding matrix that satisfies (m, η, ϵ)-BRIP. Let $\{A_t\}, \{D_t\}$ be arbitrary sequences of subsets of $[m]$ with cardinality $|A_t|, |D_t| \geq \eta m$ for all t. Then, for f as described in Section 2,*

$$f(w_t) - \frac{\kappa^2(\kappa-\gamma)}{1-\kappa\gamma} f(w^*) \leq (\kappa\gamma)^t f(w_0),$$

where $\kappa = \frac{1+\epsilon}{1-\epsilon}$, and $\gamma = \left(1 - \frac{4(\mu+\lambda)c_1c_2}{(M+\lambda)(1+\epsilon)(c_1+c_2)^2}\right)$, where c_1 and c_2 are the constants in Lemma 3.

Similar to Theorem 2, the proof is based on the observation that the solution of the effective problem at time t lies in a bounded set around the true solution w^*. As in gradient descent, coding enables linear convergence deterministically, unlike the stochastic and multi-batch variants of L-BFGS, *e.g.*, Mokhtari and Ribeiro (2015); Berahas et al. (2016).

3.4. Convergence of encoded proximal gradient

Next we consider the encoded proximal gradient algorithm, described in Section 2, for objectives with potentially non-smooth regularizers $h(w)$. The following theorem characterizes our convergence results under this setup.

Theorem 5 *Let w_t be computed using encoded proximal gradient with an encoding matrix that satisfies (m, η, ϵ)-BRIP, with step size $\alpha_t = \alpha < \frac{1}{M}$, and where $\epsilon < \frac{1}{7}$. Let $\{A_t\}$ be an arbitrary sequence of subsets of $[m]$ with cardinality $|A_t| \geq \eta m$ for all t. Then, for f as described in Section 2,*

1. *For all t,*

$$\frac{1}{t}\sum_{\tau=1}^{t} f(w_\tau) - \kappa f(w^*) \leq \frac{4\epsilon f(w_0) + \frac{1}{2\alpha}\|w_0 - w^*\|^2}{(1-7\epsilon)\,t},$$

2. *For all t,*

$$f(w_{t+1}) \leq \kappa f(w_t),$$

where $\kappa = \frac{1+7\epsilon}{1-3\epsilon}$.

As in the previous algorithms, the convergence guarantees hold for arbitrary sequences of active nodes $\{A_t\}$. Note that as in the gradient descent case, the convergence is shown on the mean of past function values. Since this does not prevent the iterates from having a sudden jump at a given iterate, we include the second part of the theorem to complement the main convergence result, which implies that the function value cannot increase by more than a small factor of its current value.

3.5. Convergence of encoded block coordinate descent

Finally, we consider the convergence of encoded block coordinate descent algorithm. The following theorem characterizes our main convergence result for this case.

Theorem 6 *Let $w_t = S^\top v_t$, where v_t is computed using encoded block coordinate descent as described in Section 2. Let S satisfy (m, η, ϵ)-BRIP, and the step size satisfy $\alpha < \frac{1}{L(1+\epsilon)}$. Let $\{A_t\}$ be an arbitrary sequence of subsets of $[m]$ with cardinality $|A_t| \geq \eta m$ for all t. Let the level set of g at the first iterate $D_g(w_0)$ have diameter R. Then, for $g(w) = \phi(Xw)$ as described in Section 2, the following hold.*

1. *If ϕ is convex, then*

$$g(w_t) - g(w^*) \leq \frac{1}{\frac{1}{\pi_0} + Ct},$$

where $\pi_0 = g(w_0) - g(w^*)$, and $C = \frac{(1-\epsilon)\alpha}{R^2}\left(1 - \frac{\alpha L'}{2}\right)$.

2. *If g is ν-restricted-strongly convex, then*

$$g(w_t) - g(w^*) \leq \left(1 - \frac{1}{\xi}\right)^t (g(w_0) - g(w^*)),$$

where $\xi = \frac{2}{\nu(1-\epsilon)\alpha}\left(1 - \frac{L(1+\epsilon)\alpha}{2}\right)^{-1}$.

Theorem 6 demonstrates that the standard $O\left(\frac{1}{t}\right)$ rate for the general convex, and linear rate for the strongly convex case can be obtained under the encoded setup. Similar to the previous cases, encoding allows for deterministic convergence guarantees under adversarial failure patterns.

Note that unlike the data parallelism setup, we can achieve exact minimum under model parallelism, since the underlying geometry of the problem does not change under encoding; the same objective is simply mapped onto a higher-dimensional space, which has redundant coordinates. In every iteration, such redundancy serves as a "soft" error correction mechanism, by allowing each worker to receive sufficient information about the up-to-date values of the rest of the parameters hosted in the other nodes, even though a subset of them are not able to send their updates sufficiently fast. The smaller the ϵ is in the BRIP condition, the more representative the redundant coordinates are for the missing ones, which is reflected in the convergence rate, as a non-zero ϵ slightly weakens the constants in the convergence expressions. Still, note that this penalty in convergence rate only depends on the encoding matrix and not on the delay profile in the system. This is in contrast to the asynchronous coordinate descent methods; for instance, in Liu et al. (2015), the step size is required to shrink *exponentially* in the maximum allowable delay, and thus the guaranteed convergence rate can exponentially degrade with increasing worst-case delay in the system. The same is true for the linear convergence guarantee in Peng et al. (2016).

4. Code Design

4.1. Block RIP condition and code design

We first discuss two classes of encoding matrices with regard to the BRIP condition; namely equiangular tight frames, and random matrices.

Tight frames. A unit-norm *frame* for $\mathbb{R}^n$ is a set of vectors $F = \{a_i\}_{i=1}^{n\beta}$ with $\|a_i\| = 1$, where $\beta \geq 1$, such that there exist constants $\xi_2 \geq \xi_1 > 0$ such that, for any $u \in \mathbb{R}^n$,

$$\xi_1 \|u\|^2 \leq \sum_{i=1}^{n\beta} |\langle u, a_i \rangle|^2 \leq \xi_2 \|u\|^2.$$

The frame is *tight* if the above is satisfied with $\xi_1 = \xi_2$. In this case, it can be shown that the constants are equal to the redundancy factor of the frame, *i.e.*, $\xi_1 = \xi_2 = \beta$. If we form $S \in \mathbb{R}^{(\beta n)\times n}$ by rows that form a *tight frame*, then we have $S^\top S = \beta I$, which ensures $\|Xw - y\|^2 = \frac{1}{\beta}\|SXw - Sy\|^2$. Then for any solution $\hat{w}$ to the encoded problem (with $k = m$),

$$\nabla \widetilde{f}(\hat{w}) = X^\top S^\top S(X\hat{w} - y) = \beta X^\top (X\hat{w} - y) = \beta \nabla f(\hat{w}).$$

Therefore, the solution to the encoded problem satisfies the optimality condition for the original problem as well:

$$-\nabla \widetilde{f}(\hat{w}) \in \partial h(\hat{w}), \quad \Leftrightarrow \quad -\nabla f(\hat{w}) \in \partial h(\hat{w}),$$

and if f is also strongly convex, then $\hat{w} = w^*$ is the unique solution. This means that for $k = m$, obliviously solving the encoded problem results in the same objective value as in the original problem.

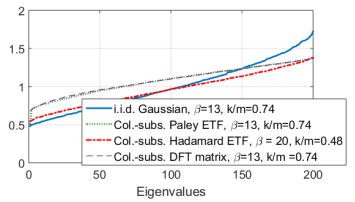


Figure 5: Sample spectrum of $S_A^\top S_A$ for various constructions with high redundancy, and small k (normalized).

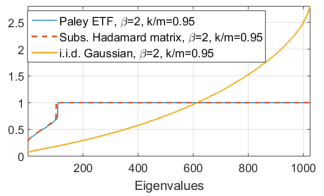


Figure 6: Sample spectrum of $S_A^\top S_A$ for various constructions with moderate redundancy, and large k (normalized).

Define the maximal inner product of a unit-norm tight frame $F = \{a_i\}_{i=1}^{n\beta}$, where $a_i \in \mathbb{R}^n, \forall i$, by

$$\omega(F) := \max_{\substack{a_i, a_j \in F \\ i \neq j}} |\langle a_i, a_j \rangle|.$$

Definition 7 (Equiangular tight frame (ETF)) *A tight frame is called an* equiangular tight frame *(ETF) if* $|\langle a_i, a_j \rangle| = \omega(F)$ *for every* $i \neq j$.

Proposition 8 (Welch (1974)) *Let* $F = \{a_i\}_{i=1}^{n\beta}$ *be a tight frame, where* $a_i \in \mathbb{R}^n$ *for* $i = 1, \ldots, n\beta$. *Then* $\omega(F) \geq \sqrt{\frac{\beta - 1}{n\beta - 1}}$. *Moreover, equality is satisfied if and only if* F *is an equiangular tight frame.*

Therefore, an ETF minimizes the correlation between its individual elements, making each submatrix $S_A^\top S_A$ as close to orthogonal as possible. This, combined with the property that tight frames preserve the optimality condition when all nodes are waited for ($k = m$), make ETFs good candidates for encoding, in light of the required property (6). We specifically evaluate the Paley ETF from Paley (1933) and Goethals and Seidel (1967); Hadamard ETF from Szöllősi (2013) (not to be confused with Hadamard matrix); and Steiner ETF from Fickus et al. (2012) in our experiments.

Although the derivation of tight eigenvalue bounds for subsampled ETFs is a long-standing problem, numerical evidence (see Figures 5, 6) suggests that they tend to have their eigenvalues more tightly concentrated around 1 than random matrices (also supported by the fact that they satisfy Welch bound, Proposition 8 with equality).

Random matrices. Another natural choice of encoding could be to use i.i.d. random matrices. Although encoding with such random matrices can be computationally expensive and may not have the desirable properties of encoding with tight frames, their eigenvalue behavior can be characterized analytically. In particular, using the existing results on the eigenvalue scaling of large i.i.d. Gaussian matrices from Geman (1980); Silverstein (1985) and union bound, it can be shown that

$$\mathbb{P}\left(\max_{A:|A|=k} \lambda_{\max}\left(\frac{1}{\beta\eta m} S_A^\top S_A\right) > \left(1 + \sqrt{\frac{1}{\beta\eta}}\right)^2\right) \to 0 \tag{8}$$

$$\mathbb{P}\left(\min_{A:|A|=k} \lambda_{\min}\left(\frac{1}{\beta\eta n} S_A^\top S_A\right) < \left(1-\sqrt{\frac{1}{\beta\eta}}\right)^2\right) \to 0, \tag{9}$$

as $n \to \infty$, if the elements of S_A are drawn i.i.d. from $N(0,1)$. Hence, for sufficiently large redundancy and problem dimension, i.i.d. random matrices are good candidates for encoding as well. However, for finite β, even if $k = m$, in general the optimum of the original problem is not recovered exactly, for such matrices.

4.2. Discussion on required redundancy

A common problem with worst-case bounds is that they are often pessimistic when compared against actual performance on real instances. In our case, directly computing the redundancy for small enough $\epsilon < 1$ (as required by Theorem 5, for instance) suggests a required redundancy β of more than 200. However, as we also show in Section 5, in practice much lower values values of redundancy (such as $\beta = 2$) results in good performance. The reason for this apparent discrepancy is that in the proofs of the theorems, we take a conservative approach and bound quadratic terms of the form

$$\xi(w) = \frac{\left|(Xw-y)^\top (S_A^\top S_A - I)(Xw-y)\right|}{\|Xw-y\|^2},$$

which is at worst

$$\xi(w) \leq \epsilon = \max\left\{\lambda_{\max}\left(S_A^\top S_A - I\right), -\lambda_{\min}\left(S_A^\top S_A - I\right)\right\}.$$

Note that the inequality is tight if $Xw - y$ is an extremal eigenvector of $S_A^T S_A - I$, which, although possible, is highly unlikely. We illustrate this point through an example experiment shown in Figure 7, which plots $\xi(w_t)$ for a run of gradient descent on linear regression with random data, where w_t is the t^{th} iterate. In the entire optimization process, it is clear that ϵ is a very pessimistic upper bound on $\xi(w_t)$.

In particular, most of the energy of the vector $Xw_t - y$ lies in the eigenspace associated with the bulk of the eigenvalues of $S_A^\top S_A$. From another perspective, if we were to use the "empirical" maximum of the value ϵ for this particular run (instead of the worst-case bound, the largest eigenvalue), we would achieve $\epsilon = 0.074$ at redundancy $\beta = 2$, a dramatic improvement compared to the above requirement of $\beta > 200$ for $\epsilon = \frac{1}{7}$.

This observation implies that for good practical performance, what is more important is to ensure that most eigenvalues lie close to, or is exactly 1. The following proposition shows that for ETFs, the bulk of the eigenvalues can be identically 1.

Proposition 9 *If the rows of S are chosen to form an ETF with redundancy β, then for $\eta \geq 1 - \frac{1}{\beta}$, $\frac{1}{\beta} S_A^\top S_A$ has $n(1-\beta(1-\eta))$ eigenvalues equal to 1.*

This follows immediately from Cauchy interlacing theorem, using the fact that $S_A S_A^\top$ and $S_A^\top S_A$ have the same spectra except zeros. Therefore for sufficiently large η, ETFs have a mostly flat spectrum even for low redundancy, and thus in practice one would expect ETFs to perform well even for small amounts of redundancy. This is also confirmed by Figure 6, as well as our numerical results.

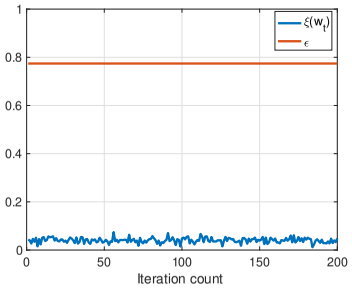


Figure 7: The evolution of $\xi(w_t)$ and ϵ against iteration count. The parameters are $(n, d, \beta) = (2001, 1400, 2)$, where we encode the data variables with a Paley ETF. We initialize the elements of w_0 with a standard normal distribution. We set $\eta = 0.9$, *i.e.*, we wait for 90% of the machines at every step.

4.3. Efficient encoding

In this section we discuss some of the possible practical approaches to encoding. Some of the practical issues involving encoding include the computational complexity of encoding, as well as the loss of sparsity in the data due to the multiplication with S, and the resulting increase in time and space complexity. We address these issues in this section.

4.3.1. Efficient distributed encoding with sparse matrices

Let the dataset (X, y) lie in a database, accessible to each worker node, where each node is responsible for computing their own encoded partitions $S_i X$ and $S_i y$. We assume that S has a sparse structure. Given S, define $B_i(S) = \{j : S_{ij} \neq 0\}$ as the set of indices of the non-zero elements of the ith row of S. For a set $\mathcal{I}$ of rows, we define $B_{\mathcal{I}}(S) = \cup_{i\in\mathcal{I}} B_i(S)$.

Let us partition the set of rows of S, $[\beta n]$, into m machines, and denote the partition of machine k as $\mathcal{I}_k$, *i.e.*, $\bigsqcup_{k=1}^m \mathcal{I}_k = [\beta n]$, where $\sqcup$ denotes disjoint union. Then the set of non-zero columns of S_k is given by $B_{\mathcal{I}_k}(S)$. Note that in order to compute $S_k X$, machine k only requires the rows of X in the set $B_{\mathcal{I}_k}(S)$. In what follows, we will denote this submatrix of X by $\widetilde{X}_k$, *i.e.*, if $x_i^\top$ is the ith row of X, $\widetilde{X}_k := \left[x_i^\top\right]_{i\in B_{\mathcal{I}_k}(S)}$. Similarly $\widetilde{y}_k = [y_i]_{i\in B_{\mathcal{I}_k}(S)}$, where y_i is the ith element of y.

Consider the specific computation that needs to be done by worker k during the iterations, for each algorithm. Under the data parallelism setting, worker k computes the following gradient:

$$\nabla f_k(w) = X^\top S_k^\top S_k (Xw - y) \overset{(a)}{=} \widetilde{X}_k^\top S_k^\top S_k (\widetilde{X}_k w - \widetilde{y}_k) \tag{10}$$

where (a) follows since the rows of X that are not in $B_{\mathcal{I}_k}$ get multiplied by zero vector. Note that the last expression can be computed without any matrix-matrix multiplication. This gives a natural storage and computation scheme for the workers. Instead of computing $S_k X$ offline and storing it, which can result in a loss of sparsity in the data, worker k

can store $\widetilde{X}_k$ in uncoded form, and compute the gradient through (10) whenever needed, using only matrix-vector multiplications. Since S_k is sparse, the overhead associated with multiplications of the form $S_k v$ and $S_k^\top v$ is small.

Similarly, under model parallelism, the computation required by worker k is

$$\nabla_k \widetilde{g}(v) = S_k X^\top \nabla_k \phi\left(X S_k^\top v_k + \widetilde{z}_k\right) = S_k \widetilde{X}_k^\top \nabla_k \phi\left(\widetilde{X}_k S_k^\top v_k + \widetilde{z}_k\right), \tag{11}$$

and as in the data parallelism case, the worker can store $\widetilde{X}_k$ uncoded, and compute (11) online through matrix-vector multiplications.

Example: Steiner ETF. We illustrate the described technique through Steiner ETF, based on the construction proposed in Fickus et al. (2012), using $(2, 2, v)$-Steiner systems. Let v be a power of 2, let $H \in \mathbb{R}^{v\times v}$ be a real Hadamard matrix, and let h_i be the ith column of H, for $i = 1, \ldots, v$. Consider the matrix $V \in \{0, 1\}^{v\times v(v-1)/2}$, where each column is the incidence vector of a distinct two-element subset of $\{1, \ldots, v\}$. For instance, for $v = 4$,

$$V = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Note that each of the v rows have exactly $v - 1$ non-zero elements. We construct Steiner ETF S as a $v^2 \times \frac{v(v-1)}{2}$ matrix by replacing each 1 in a row with a distinct column of H, and normalizing by $\sqrt{v-1}$. For instance, for the above example, we have

$$S = \frac{1}{\sqrt{3}} \begin{bmatrix} h_2 & h_3 & h_4 & 0 & 0 & 0 \\ h_2 & 0 & 0 & h_3 & h_4 & 0 \\ 0 & h_2 & 0 & h_3 & 0 & h_4 \\ 0 & 0 & h_2 & 0 & h_3 & h_4 \end{bmatrix}.$$

We will call a set of rows of S that arises from the same row of V a block. In general, this procedure results in a matrix S with redundancy factor $\beta = \frac{2v}{v-1}$. In full generality, Steiner ETFs can be constructed for larger redundancy levels; we refer the reader to Fickus et al. (2012) for a full discussion of these constructions.

We partition the rows of the V matrix into m machines, so that each machine gets assigned $\frac{v}{m}$ rows of V, and thus the corresponding $\frac{v}{m}$ blocks of S.

This construction and partitioning scheme is particularly attractive for our purposes for two reasons. First, it is easy to see that for any node k, $|B_{\mathcal{I}_k}|$ is upper bounded by $\frac{v(v-1)}{m} = \frac{2n}{m}$, which means the memory overhead compared to the uncoded case is limited to a factor[6] of β. Second, each block of S_k consists of (almost) a Hadamard matrix, so the multiplication $S_k v$ can be efficiently implemented through Fast Walsh-Hadamard Transform.

6. In practice, we have observed that the convergence performance improves when the blocks are broken into multiple machines, so one can, for instance, assign half-blocks to each machine.

Example: Haar matrix. Another possible choice of sparse matrix is column-subsampled Haar matrix, which is defined recursively by

$$H_{2n} = \frac{1}{\sqrt{2}} \begin{bmatrix} H_n \otimes [1\ 1] \\ I_n \otimes [1\ -1] \end{bmatrix}, \quad H_1 = 1,$$

where $\otimes$ denotes Kronecker product. Given a redundancy level β, one can obtain S by randomly sampling $\frac{n}{\beta}$ columns of H_n. It can be shown that in this case, we have $|B_{\mathcal{I}_k}| \leq \frac{\beta n \log(n)}{m}$, and hence encoding with Haar matrix incurs a memory cost by logarithmic factor.

4.3.2. Fast transforms

Another computationally efficient method for encoding is to use fast transforms: Fast Fourier Transform (FFT), if S is chosen as a subsampled DFT matrix, and the Fast Walsh-Hadamard Transform (FWHT), if S is chosen as a subsampled real Hadamard matrix. In particular, one can insert rows of zeroes at random locations into the data pair (X, y), and then take the FFT or FWHT of each column of the augmented matrix. This is equivalent to a randomized Fourier or Hadamard ensemble, which is known to satisfy the RIP with high probability by Candes and Tao (2006). However, such transforms do not have the memory advantages of the sparse matrices, and thus they are more useful for the setting where the dataset is dense, and the encoding is done offline.

4.4. Cost of encoding

Since encoding increases the problem dimensions, it clearly comes with the cost of increased space complexity. The memory and storage requirement of the optimization still increases by a factor of 2, if the encoding is done offline (for dense datasets), or if the techniques described in the previous subsection are applied (for sparse datasets)[7]. Note that the added redundancy can come by increasing the amount of effective data points per machine, by increasing the number of machines while keeping the load per machine constant, or a combination of the two. In the first case, the computational load per machine increases by a factor of β. Although this can make a difference if the system is bottlenecked by the computation time, distributed computing systems are typically communication-limited, and thus we do not expect this additional cost to dominate the speed-up from the mitigation of stragglers.

5. Numerical Results

We implement the proposed technique on four problems: ridge regression, matrix factorization, logistic regression, and LASSO.

7. Note that the increase in space complexity is not higher for sparse matrices, since the sparsity loss can be avoided using the technique described in Section 4.3.1

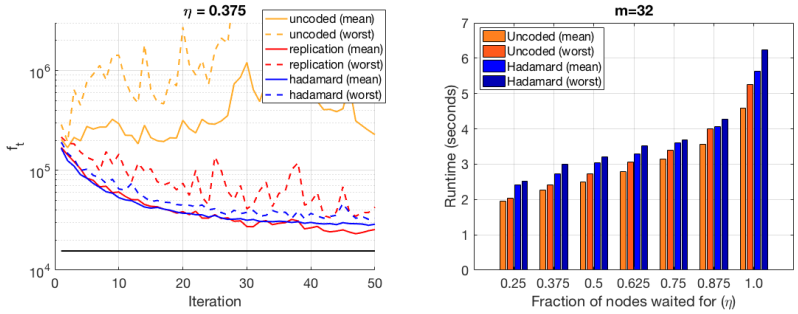


Figure 8: **Left:** Sample evolution of uncoded, replication, and Hadamard (FWHT)-coded cases, for $k = 12$, $m = 32$. **Right:** Runtimes of the schemes for different values of η, for the same number of iterations for each scheme. Note that this essentially captures the delay profile of the network, and does not reflect the relative convergence rates of different methods.

5.1. Data parallelism

5.1.1. Ridge regression

We generate the elements of matrix X i.i.d. $\sim N(0,1)$, and the elements of y are generated from X and an i.i.d. $N(0,1)$ parameter vector w^*, through a linear model with Gaussian noise, for dimensions $(n,p) = (4096, 6000)$. We solve the problem $\min_w \frac{1}{2n}\|S(Xw-y)\|^2 + \frac{\lambda}{2}\|w\|^2$, for regularization parameter $\lambda = 0.05$. We evaluate column-subsampled Hadamard matrix with redundancy $\beta = 2$ (encoded using FWHT), replication and uncoded schemes. We implement distributed L-BFGS as described in Section 3 on an Amazon EC2 cluster using `mpi4py` Python package, over $m = 32$ `m1.small` instances as worker nodes, and a single `c3.8xlarge` instance as the central server.

Figure 8 shows the result of our experiments, which are aggregated from 20 trials. In addition to uncoded scheme, we consider data replication, where each uncoded partition is replicated $\beta = 2$ times across nodes, and the server discards the duplicate copies of a partition, if received in an iteration. It can be seen that for low η, uncoded L-BFGS may not converge when a fixed number of nodes are waited for, whereas the Hadamard-coded case stably converges. We also observe that the data replication scheme converges on average, but its performance may deteriorate if both copies of a partition are delayed. Figure 8 suggests that this performance can be achieved with an approximately 40% reduction in the runtime, compared to waiting for all the nodes.

Note that in the left plot of Figure 8, iteration count does not equal time. The combined message of the two plots is that although encoding incurs a computational overhead due to increased dimensions (as can be seen by the fact that for the same η, Hadamard-encoded case takes longer than the uncoded case), the time savings due to waiting for fewer nodes per iteration (reduced η) dominates this overhead, especially for communication-limited scenarios. However, when η is reduced, uncoded optimization can destabilize or lead to

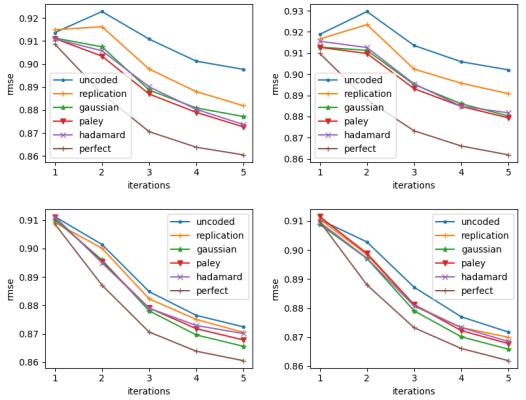


Figure 9: Test RMSE for $m = 8$ (left) and $m = 24$ (right) nodes, where the server waits for $k = m/8$ (top) and $k = m/2$ (bottom) responses. "Perfect" refers to the case where $k = m$.



Figure 10: Total runtime with $m = 8$ and $m = 24$ nodes for different values of k, under fixed 100 iterations for each scheme.

less accurate solutions, as can be seen from the aggregated sample evolution on the left. The left plot shows that, encoding effectively counteracts this effect, maintaining smooth convergence due to the redundant data in the nodes.

5.1.2. Matrix factorization

We next apply matrix factorization on the MovieLens-1M dataset (Riedl and Konstan (1998)) for the movie recommendation task. We are given R, a sparse matrix of movie ratings 1–5, of dimension $\#users \times \#movies$, where R_{ij} is specified if user i has rated movie j. We withhold randomly 20% of these ratings to form an 80/20 train/test split. The goal is to recover user vectors $x_i \in \mathbb{R}^p$ and movie vectors $y_i \in \mathbb{R}^p$ (where p is the embedding dimension) such that $R_{ij} \approx x_i^T y_j + u_i + v_j + b$, where u_i, v_j, and b are user, movie, and global biases, respectively. The optimization problem is given by

$$\min_{x_i,y_j,u_i,v_j} \sum_{i,j:\ \text{observed}} (R_{ij} - u_i - v_j - x_i^T y_j - b)^2 + \lambda \left(\sum_i \|x_i\|_2^2 + \|u\|_2^2 + \sum_j \|y_j\|_2^2 + \|v\|_2^2 \right). \tag{12}$$

We choose $b = 3$, $p = 15$, and $\lambda = 10$, which achieves test RMSE 0.861, close to the current best test RMSE on this dataset using matrix factorization[8].

Problem (12) is often solved using alternating minimization, minimizing first over all (x_i, u_i), and then all (y_j, v_j), in repetition. Each such step further decomposes by row and column, made smaller by the sparsity of R. To solve for (x_i, u_i), we first extract

8. `http://www.mymedialite.net/examples/datasets.html`

$I_i = \{j \mid r_{ij} \text{ is observed}\}$, and minimize

$$\left([y_{I_i}^T, \mathbf{1}] \begin{bmatrix} x_i \\ u_i \end{bmatrix} - (R_{i,I_i}^T - v_{I_i} - b\mathbf{1}) \right)^2 + \lambda \left(\sum_i \|x_i\|_2^2 + \|u\|_2^2 \right) \tag{13}$$

for each i, which gives a sequence of regularized least squares problems with variable $w = [x_i^T, u_i]^T$, which we solve distributedly using coded L-BFGS; and repeat for $w = [y_j^T, v_j]^T$, for all j.

The Movielens experiment is run on a single 32-core machine with Linux 4.4. In order to simulate network latency, an artificial delay of $\Delta \sim \exp(10 \text{ ms})$ is imposed each time the worker completes a task. Small problem instances ($n < 500$) are solved locally at the central server, using the built-in function `numpy.linalg.solve`. To reduce overhead, we create a bank of encoding matrices $\{S_n\}$ for Paley ETF and Hadamard ETF, for $n = 100, 200, \ldots, 3500$, and then given a problem instance, subsample the columns of the appropriate matrix S_n to match the dimensions. Overall, we observe that encoding overhead is amortized by the speed-up of the distributed optimization.

Figure 9 gives the final performance of our distributed L-BFGS for various encoding schemes, for each of the 5 epochs, which shows that coded schemes are most robust for small k. A full table of results is given in Appendix D.

5.1.3. LASSO

We solve the LASSO problem, with the objective

$$\min_w \frac{1}{2n} \|Xw - y\|^2 + \lambda \|w\|_1,$$

where $X \in \mathbb{R}^{130{,}000 \times 100{,}000}$ is a matrix with i.i.d. $N(0,1)$ entries, and y is generated from X and a parameter vector w^* through a linear model with Gaussian noise:

$$y = Xw^* + \sigma z,$$

where $\sigma = 40$, $z \sim N(0,1)$. The parameter vector w^* has 7695 non-zero entries out of 100,000, where the non-zero entries are generated i.i.d. from $N(0,4)$. We choose $\lambda = 0.6$ and consider the sparsity recovery performance of the corresponding LASSO problem, solved using proximal gradient (iterative shrinkage/thresholding algorithm).

We implement the algorithm over 128 `t2.medium` worker nodes which collectively store the matrix X, and a `c3.4xlarge` master node. We measure the sparsity recovery performance of the solution using the F1 score, defined as the harmonic mean

$$F1 = \frac{2PR}{P+R},$$

where P and R are precision recall of the solution vector $\hat{w}$ respectively, defined as

$$P = \frac{|\{i : w_i^* \neq 0, \hat{w}_i \neq 0\}|}{|i : \hat{w}_i \neq 0|}, \quad R = \frac{|\{i : w_i^* \neq 0, \hat{w}_i \neq 0\}|}{|i : w_i^* \neq 0|}$$

.

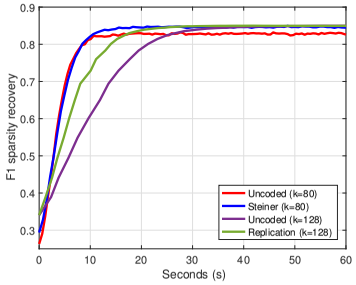


Figure 11: Evolution of F1 sparsity recovery performance for each scheme.

Figure 11 shows the sample evolution of the F1 score of the model under uncoded, replication, and Steiner encoded scenarios, with artificial multi-modal communication delay distribution $q_1\mathcal{N}(\mu_1, \sigma_1^2) + q_2\mathcal{N}(\mu_s, \sigma_2^2) + q_3\mathcal{N}(\mu_3, \sigma_3^2)$, where $q_1 = 0.8$, $q_2 = 0.1$, $q_3 = 0.1$; $\mu_1 = 0.2$s, $\mu_s = 0.6$s, $\mu_3 = 1$s; and $\sigma_1 = 0.1$s, $\sigma = 0.2$s, $\sigma_3 = 0.4$s, independently at each node. We observe that the uncoded case $k = 80$ results in a performance loss in sparsity recovery due to data dropped from delayed noes, and uncoded and replication with $k = 128$ converges slow due to stragglers, while Steiner coding with $k = 80$ is not delayed by stragglers, while maintaining almost the same sparsity recovery performance as the solution of the uncoded $k = 128$ case.

5.2. Model parallelism

5.2.1. Logistic regression

In our next experiment, we apply logistic regression for document classification for Reuters Corpus Volume 1 (`rcv1.binary`) dataset from Lewis et al. (2004), where we consider the binary task of classifying the documents into corporate/industrial/economics vs. government/social/markets topics. The dataset has 697,641 documents, and 47,250 term frequency-inverse document frequency (tf-idf) features. We randomly select 32,500 features for the experiment, and reserve 100,000 documents for the test set. We use logistic regression with ℓ_2-regularization for the classification task, with the objective

$$\min_{w,b} \frac{1}{n} \sum_{i=1}^{n} \log\left(1 + \exp\left\{-z_i^\top w + b\right\}\right) + \lambda \|w\|^2,$$

where $z_i = y_i x_i$ is the data sample x_i multiplied by the label $y_i \in \{-1, 1\}$, and b is the bias variable. We solve this optimization using encoded distributed block coordinate descent as described in Section 2, and implement Steiner and Haar encoding as described in Section 4, with redundancy $\beta = 2$. In addition we implement the asynchronous coordinate descent, as well as replication, which represents the case where each partition Z_i is replicated across two nodes, and the faster copy is used in each iteration. We use $m = 128$ `t2.medium` instances as worker nodes, and a single `c3.4xlarge` instance as the master node, which

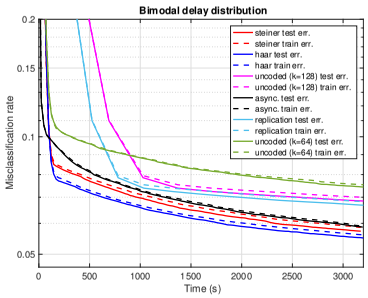


Figure 12: Test and train errors over time (in seconds) for each scheme, for the bimodal delay distribution. Steiner and Haar encoding is done with $k = 64$, $\beta = 2$.

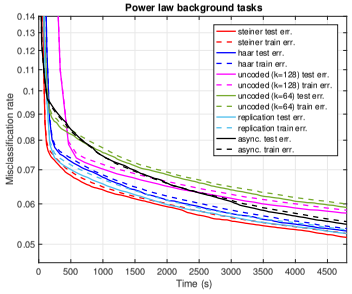


Figure 13: Test and train errors over time (in seconds) for each scheme. Number of background tasks follow a power law. Steiner and Haar encoding is done with $k = 80$, $\beta = 2$.

# Nodes (m)	Dimensions ($n \times d$)
8	8000×3000
32	32000×12000
128	128000×48000

Table 2: Problem dimensions for each experiment.

communicate using the `mpi4py` package. We consider two models for stragglers. In the first model, at each node, we add a random delay drawn from a Gaussian mixture distribution $q\mathcal{N}(\mu_1, \sigma_1^2) + (1-q)\mathcal{N}(\mu_s, \sigma_2^2)$, where $q = 0.5$, $\mu_1 = 0.5$s, $\mu_s = 20$s, $\sigma_1 = 0.2$s, $\sigma_2 = 5$s. In the second model, we do not directly add any delay, but at each machine we launch a number of dummy background tasks (matrix multiplication) that are executed throughout the computation. The number of background tasks across the nodes is distributed according to a power law with exponent $\alpha = 1.5$. The number of background tasks launched is capped at 50.

Figures 12 and 13 shows the evolution of training and test errors as a function of wall clock time. We observe that for each straggler model, either Steiner or Haar encoded optimization dominates all schemes. Figures 14 and 15 show the statistics of how frequent each node participates in an update, for the case with background tasks, for encoded and asynchronous cases, respectively. We observe that the stark difference in the relative speeds of different machines result in vastly different update frequencies for the asynchronous case, which results in updates with large delays, and a corresponding performance loss.

5.3. Speed tests

In order to quantify the speed-up achieved by the encoding scheme, we design a set of controlled distributed optimization experiments which compares it against other commonly

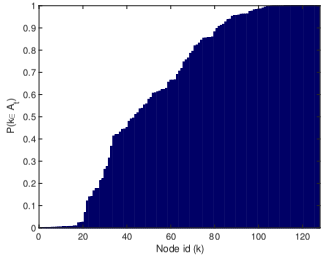


Figure 14: The fraction of iterations each worker node participates in (the empirical probability of the event $\{k \in A_t\}$), plotted for Steiner encoding with $k = 80$, $m = 128$. The number of background tasks are distributed by a power law with $\alpha = 1.5$ (capped at 50).

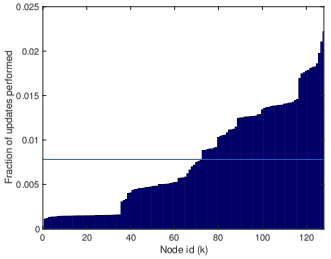


Figure 15: The fraction of updates performed by each node, for asynchronous block coordinate descent. The horizontal line represents the uniformly distributed case. The number of background tasks are distributed by a power law with $\alpha = 1.5$ (capped at 50).

Scheme	Runtime (s) $\mu_s = 0$	Runtime (s) $\mu_s = \gamma$	Runtime (s) $\mu_s = 2\gamma$
Steiner	1.71	2.10	2.13
Synchronous	1.00	7.93	14.89
Replication	1.03	2.11	3.16
Asynchronous	0.61	-	-

Table 3: The average wall-clock time taken to achieve $1.05 \times$ MMSE for each scheme, for $m = 8$ workers.

Scheme	Runtime (s) $\mu_s = 0$	Runtime (s) $\mu_s = \gamma$	Runtime (s) $\mu_s = 2\gamma$
Steiner	1.84	2.58	3.09
Synchronous	1.12	9.06	17.24
Replication	1.17	4.96	8.69
Asynchronous	0.66	-	-

Table 4: The average wall-clock time taken to achieve $1.05 \times$ MMSE for each scheme, for $m = 32$ workers.

Scheme	Runtime (s) $\mu_s = 0$	Runtime (s) $\mu_s = \gamma$	Runtime (s) $\mu_s = 2\gamma$
Steiner	6.64	15.18	18.51
Synchronous	3.69	43.60	84.55
Replication	4.56	26.55	49.75
Asynchronous	8.77	-	-

Table 5: The average wall-clock time taken to achieve $1.05 \times$ MMSE for each scheme, for $m = 128$ workers.

straggler mitigation schemes, namely replication and asynchronous optimization, as well as the uncoded synchronous case. To do this, we focus on ridge regression, and use the mean-squared error (MSE) in predicting the test-set response variable y as our metric. We compare the methods against each other by measuring the wall-clock time of optimization required to achieve a fixed mean-squared error.

Note that since the speed-up is due to the mitigation of stragglers, the performance of the proposed technique depends strongly on many cluster- and problem-specific factors, such as computational power of nodes, number of nodes, background processes running at nodes, networking configurations, latency and bandwidth available in the network, compute-to-communication ratio required by the problem, availability of nodes[9]. Since these experiments are intended purely as speed measurements, we make several design decisions to build a highly-controlled computing environment, in order to isolate the effects of the straggler mitigation technique and side-step problem- or network-specific artifacts that might affect the results.

First, to have direct control over the delays induced by the workers and measure the exact effect of the delay distribution on the runtime, we use compute-optimized, high-performance, high-bandwidth `c4.4xlarge` and `c4.large` instances available through Amazon EC2, set a node to be a straggler with probability $q = 0.25$, and for stragglers in each step we add an artificial delay drawn from Gaussian distribution $\mathcal{N}\left(\mu_s, \sigma_s^2\right)$, to simulate more unreliable clusters. In our tests, we vary μ_s to measure the effect of the amount of straggler delay (keeping $\sigma_s = 0.4\mu_s$). We sweep μ_s over the values $\{0, \gamma, 2\gamma\}$, where γ is chosen as the amount of time it takes for a non-straggler node to perform 5-6 iterations for the given problem dimensions. Note that such delays can be observed in practice in unreliable/burstable clusters (see, for instance, the EC2 experiments in Tandon et al. (2016) conducted on `t2.micro` instances, where the stragglers are $8\times$ slower, or Ananthanarayanan et al. (2013) which similarly reports $8\times$ straggler delay for latency-limited tasks).

Second, as we scale the number of workers, we scale the problem proportionally as well, in order to maintain the same computation-communication ratio. To be able to control exact problem dimensions, we use synthetic data, where the data matrix $X \in \mathbb{R}^{n\times d}$ is generated i.i.d. $\mathcal{N}(0,1)$, and $y = X\bar{w} + \eta$ for some ground truth $\bar{w}$ and noise $\eta \in \mathcal{N}\left(0, \sigma_\eta^2\right)$, with σ_η^2 proportional to n.

Third, since under data-parallelism the proposed method does not converge to the exact minimum, we first find the minimum MSE (MMSE) achieved for the given problem through a preliminary experiment using fully synchronous optimization with all nodes, as well as hyper-parameter optimization to determine a good value for the regularization parameter λ, which we determine to be $\lambda = 0.025$. Then, we set the MSE bar as $1.05 \times \mathsf{MMSE}$. Note that if the exact minimum is desired for encoded scenarios in a real application, one can always increase k_t to m (*i.e.*, start using all nodes) towards the end of the optimization, after this approximate target is reached.

Under this setup, we perform three experiments, sweeping the number of workers in the set $m \in \{8, 32, 128\}$. The problem dimensions for each case is given in Table 2. For encoding we use Steiner ETFs described in Section 4 with redundancy $\beta = 2$, and $k_t = 0.75m$. In

9. For instance, in Amazon EC2 clusters, burstable `t2` and `t3`-type instances can provide high performance for a limited period of time, before slowing down to a baseline level of performance; or low-cost spot instances, which can instantly become available or unavailable based on supply and demand.

all cases, we use gradient descent with step size $\alpha_t = 0.2$, which is run for 120 steps. γ is determined to be 0.2s, 0.3s, and 1.2s for the three problem sizes, respectively.

The results are provided in Tables 3–5, where four methods are compared: Steiner encoding, uncoded synchronous optimization, uncoded asynchronous optimization, and replication, where each data shard is replicated across two nodes, and the faster copy is used in optimization. Note that Steiner encoding and replication schemes have double the data per worker. The results are averaged over 5 trials.

We observe that when there is no delay ($\mu_s = 0$), the additional computation overhead of encoding (as well as the perturbation of data) results in relatively high runtime to achieve the desired MSE. We see a similar overhead for replication, although replication enjoys the advantage of having access to raw, uncoded data. However, in the regime where the system is bottlenecked by stragglers ($\mu_s = \gamma$ and $\mu_s = 2\gamma$), the scheme results in significant speed up, ranging from approximately $3\times$ to $7\times$ against the uncoded synchronous; and up to $2\times$ against the replication baseline. Note that replication still gets impacted by straggler nodes if both copies of a shard happen to be delayed. When there are no stragglers, and for modest cluster sizes, asynchronous optimization achieves the best performance. However, since the gradient staleness scales linearly with the cluster size, asynchronous performance falls behind Steiner for $m = 128$ workers even in the absence of stragglers. We also observe that when stragglers are introduced, outdated gradients significantly degrade the asynchronous performance, as the desired MSE bar is not reached at the end of 120 steps. We have run the same experiment for the uncoded case with erasures as well; however, the method failed to achieve the MSE target in all the experiments, including the case with $\mu_s = 0$. This is because the straggler nodes tend to persist their slow computation throughout optimization. In other words, it is usually approximately the same subset of nodes that provide the updates, effectively removing the data stored in straggler nodes from the optimization. Finally, we note that the runtime for the encoded case also increases in the presence of stragglers (though less severely than the other schemes), due to various implementation overheads such as the interruption of the stragglers for the next iteration. However, we believe that part of this overhead could be mitigated in a more efficient implementation.

We conclude from these experiments that encoding can result in large speed-ups in highly unreliable, low-cost clusters which can induce large delays (such as EC2 `t2` instances), or clusters with intermittent node availability (such as EC2 spot instances). It is less suited to reliable clusters with high-capacity, high-bandwidth nodes due to the computational overhead associated with it.

Acknowledgments

The work of Can Karakus and Suhas Diggavi was supported in part by NSF grants #1314937 and #1514531, and by UC-NL grant LFR-18-548554. The work of Wotao Yin was supported by ONR Grant N000141712162, and NSF Grant DMS-1720237.

Appendix A. Proofs of Theorems 2 and 4

In the proofs, we will ignore the normalization constants on the objective functions for brevity. We will assume the normalization $\frac{1}{\sqrt{\eta}}$ is absorbed into the encoding matrix S_A. Let $f(w) = \|Xw - y\|^2 + \lambda h(w)$. Let $\widetilde{f}_t^A := \|S_{A_t}(Xw_t - y)\|^2 + \lambda h(w_t)$, and $\widetilde{f}^A(w) := \|S_{A_t}(Xw - y)\|^2 + \lambda h(w)$, where we set $A \equiv A_t$. Let $\widetilde{w}_t^*$ denote the solution to the effective "instantaneous" problem at iteration t, *i.e.*, $\widetilde{w}_t^* = \arg\min_w \widetilde{f}^A(w)$.

Unless otherwise stated, throughout this appendix, we will also denote

$$w^* = \arg\min_w \|Xw - y\|^2 + \lambda h(w)$$
$$\hat{w} = \arg\min_w \|S_A(Xw - y)\|^2 + \lambda h(w)$$

unless otherwise noted, where A is a fixed subset of $[m]$.

A.1. Lemmas

Lemma 10 (Approximation error) *If S satisfies (6) (BRIP) for any $A \subseteq [m]$ with $|A| \geq k$, for any convex set C,*

$$\|X\hat{w} - y\|^2 \leq \kappa^2 \|Xw^* - y\|^2,$$

where $\kappa = \frac{1+\epsilon}{1-\epsilon}$, $\hat{w} = \arg\min_{w\in C} \|S_A(Xw - y)\|^2$, and $w^* = \arg\min_{w\in C} \|Xw - y\|^2$.

Proof Define $e = \hat{w} - w^*$ and note that

$$\|X\hat{w} - y\| = \|Xw^* - y + Xe\| \leq \|Xw^* - y\| + \|Xe\|$$

by triangle inequality, which implies

$$\|X\hat{w} - y\|^2 \leq \left(1 + \frac{\|Xe\|}{\|Xw^* - y\|}\right)^2 \|Xw^* - y\|^2. \tag{14}$$

Note that we have

$$\|S_A(X\hat{w} - y)\|^2 \leq \|S_A(Xw^* - y)\|^2$$

by the definition of $\hat{w}$. Expanding the quadratic terms and canceling the common term $y^\top S_A^\top S_A y$ from both sides, we get

$$\hat{w}^\top X^\top S_A^\top S_A X\hat{w} - 2y^\top S_A^\top S_A X\hat{w} \leq w^{*\top} X^\top S_A^\top S_A Xw^* - 2y^\top S_A^\top S_A Xw^*.$$

Adding $-2w^{*\top} X^\top S_A^\top S_A X\hat{w} + w^{*\top} X^\top S_A^\top S_A Xw^*$ to both sides,

$$\begin{aligned}\hat{w}^\top X^\top S_A^\top S_A X\hat{w} - 2w^{*\top} X^\top S_A^\top S_A X\hat{w} + w^{*\top} X^\top S_A^\top S_A Xw^* - 2y^\top S_A^\top S_A X\hat{w} \\ \leq 2w^{*\top} X^\top S_A^\top S_A Xw^* - 2y^\top S_A^\top S_A Xw^* - 2w^{*\top} X^\top S_A^\top S_A X\hat{w},\end{aligned}$$

which can be re-arranged into

$$\|S_A X(\hat{w} - w^*)\|^2 \leq 2y^\top S_A^\top S_A X(\hat{w} - w^*) - 2w^{*\top} X^\top S_A^\top S_A X(\hat{w} - w^*),$$

which, using the definition $e = \hat{w} - w^*$, is equivalent to

$$\|S_A Xe\|^2 \leq -2(Xw^* - y)^\top S_A^\top S_A Xe = -2e^\top X^\top S_A^\top S_A(Xw^* - y). \tag{15}$$

Now, for any $c > 0$, consider

$$\begin{aligned}\|Xe\|^2 &\overset{(a)}{\leq} \frac{\|S_A Xe\|^2}{1-\epsilon} \overset{(b)}{\leq} -2\frac{e^\top X^\top S_A^\top S_A(Xw^* - y)}{1-\epsilon} \\ &= -2\frac{e^\top X^\top \left(S_A^\top S_A - cI\right)(Xw^* - y)}{1-\epsilon} - \frac{2c}{1-\epsilon} e^\top X^\top (Xw^* - y) \\ &\overset{(c)}{\leq} -2\frac{e^\top X^\top \left(S_A^\top S_A - cI\right)(Xw^* - y)}{1-\epsilon} \\ &\overset{(d)}{\leq} 2\frac{\left\|e^\top X^\top \left(cI - S_A^\top S_A\right)\right\|}{1-\epsilon} \|Xw^* - y\| \\ &\overset{(e)}{\leq} 2\frac{\left\|cI - S_A^\top S_A\right\|}{1-\epsilon} \|Xw^* - y\| \|Xe\|,\end{aligned}$$

where (a) follows from the BRIP property for S_A; (b) follows by (15); (c) follows by the fact that since $\hat{w} \in C$, e represents a feasible direction of the constrained optimization, and thus the convex optimality condition implies $\langle \nabla p(w^*), \hat{w} - w^* \rangle = 2e^\top X^\top (Xw^* - y) \geq 0$, where $p(w) := \|Xw - y\|^2$; (d) follows by Cauchy-Schwarz inequality; and (e) follows by the definition of matrix norm.

The above chain of inequalities then imply that

$$\frac{\|Xe\|}{\|Xw^* - y\|} \leq \frac{2\left\|cI - S_A^\top S_A\right\|}{1-\epsilon}.$$

Since this is true for any $c > 0$, we make the minimizing choice $c = \frac{\lambda_{\max} + \lambda_{\min}}{2}$ (where $\lambda_{\max}$ and $\lambda_{\min}$ represent the largest and smallest eigenvalues of $S_A^\top S_A$, respectively), which gives

$$\frac{\|Xe\|}{\|Xw^* - y\|} \leq \frac{\lambda_{\max} - \lambda_{\min}}{1-\epsilon} \leq \frac{2\epsilon}{1-\epsilon}.$$

Plugging this back in (14), we get the desired result. ∎

Lemma 11 (Approximation error with regularization) *If S satisfies (6) (BRIP) for any $A \subseteq [m]$ with $|A| \geq k$,*

$$f(\hat{w}) \leq \kappa^2 f(w^*),$$

where $\kappa = \frac{1+\epsilon}{1-\epsilon}$, $\hat{w} = \arg\min_w \|S_A(Xw - y)\|^2 + \lambda h(w)$, and $w^* = \arg\min_w \|Xw - y\|^2 + \lambda h(w)$.

Proof Consider a fixed $A_t = A$, and a corresponding

$$\hat{w} = \widetilde{w}_t^* \in \arg\min_w \|S_A(Xw - y)\|^2 + \lambda h(w)$$

Define

$$\hat{w}(r) = \underset{w:\lambda h(w)\le r}{\arg\min} \|S_A(Xw-y)\|^2$$
$$w^*(r) = \underset{w:\lambda h(w)\le r}{\arg\min} \|Xw-y\|^2.$$

Finally, define

$$r^* = \arg\min_r \|Xw^*(r)-y\|^2 + r.$$

Now, consider

$$\begin{aligned} f(\hat{w}) &= \|X\hat{w}-y\|^2 + \lambda h(\hat{w}) = \min_r \left(\|X\hat{w}(r)-y\|^2 + r\right) \\ &\le \|X\hat{w}(r^*)-y\|^2 + r^* \overset{(a)}{\le} \kappa^2 \|Xw^*(r^*)-y\|^2 + r^* \\ &\le \kappa^2 \left(\|Xw^*(r^*)-y\|^2 + r^*\right) = \kappa^2 f(w^*), \end{aligned}$$

which shows the desired result, where (a) follows by Lemma 10, and by the fact that the set $\{w : \lambda h(w) \le r\}$ is a convex set. ■

Lemma 12 (Error evolution) *Let $\kappa = \frac{1+\epsilon}{1-\epsilon}$. If*

$$\widetilde{f}^A_{t+1} - \widetilde{f}^A(\widetilde{w}^*_t) \le \gamma\left(\widetilde{f}^A_t - \widetilde{f}^A(\widetilde{w}^*_t)\right)$$

for all $t > 0$, and for some $0 < \gamma < 1$ with $\kappa\gamma < 1$, where $\widetilde{w}^*_t \in \arg\min_w \widetilde{f}^A_t$, then

$$f(w_t) \le (\kappa\gamma)^t f(w_0) + \frac{\kappa^2(\kappa-\gamma)}{1-\kappa\gamma} f(w^*).$$

Proof Since for any w,

$$(1-\epsilon)\|Xw-y\|^2 \le (Xw-y)^\top S_A^\top S_A (Xw-y),$$

we have

$$(1-\epsilon) f(w) \le \widetilde{f}^A(w).$$

Similarly $\widetilde{f}^A(w) \le (1+\epsilon) f(w)$, and therefore, using the assumption of the lemma

$$(1-\epsilon) f(w_{t+1}) - (1+\epsilon) f(\widetilde{w}^*_t) \le \gamma\left((1+\epsilon) f(w_t) - (1-\epsilon) f(\widetilde{w}^*_t)\right),$$

which can be re-arranged into the linear recursive inequality

$$f(w_{t+1}) \le \kappa\gamma f(w_t) + (\kappa-\gamma) f(\widetilde{w}^*_t) \overset{(a)}{\le} \kappa\gamma f(w_t) + \kappa^2(\kappa-\gamma) f(w^*),$$

where $\kappa = \frac{1+\epsilon}{1-\epsilon}$ and (a) follows by Lemma 11. By considering such inequalities for $0 \le \tau \le t$, multiplying each by $(\kappa\gamma)^{t-\tau}$ and summing, we get

$$\begin{aligned} f(w_t) &\le (\kappa\gamma)^t f(w_0) + \kappa^2(\kappa-\gamma) f(w^*) \sum_{\tau=0}^{t-1} (\kappa\gamma)^\tau \\ &\le (\kappa\gamma)^t f(w_0) + \frac{\kappa^2(\kappa-\gamma)}{1-\kappa\gamma} f(w^*). \end{aligned}$$

■

Lemma 13 (Strong convexity parameter) *Under the assumptions of Theorem 4, $\widetilde{f}^A(w)$ is $(1-\epsilon)(\mu+\lambda)$-strongly convex.*

Proof It is sufficient to show that the minimum eigenvalue of $\widetilde{X}_A^\top \widetilde{X}_A$ is bounded away from zero. This can easily be shown by the fact that

$$u^\top \widetilde{X}_A^\top \widetilde{X}_A u = u^\top \hat{X}^\top S_A^\top S_A \hat{X} u \ge (1-\epsilon)\|\hat{X}u\|^2 \ge (1-\epsilon)(\mu+\lambda)\|u\|^2,$$

for any unit vector u. ■

Lemma 14 (Rotation bound) *Let $M \in \mathbb{R}^{p\times p}$ be a symmetric positive definite matrix, with the condition number (ratio of maximum eigenvalue to the minimum eigenvalue) given by κ. Then, for any unit vector u,*

$$\frac{u^\top M u}{\|Mu\|} \ge \frac{2\sqrt{\kappa}}{\kappa+1}.$$

Proof We point out that this is a special case of Kantorovich inequality, but provide a dedicated proof here for completeness.

Let M have the eigen-decomposition $M = Q^\top D Q$, where Q has orthonormal columns, and D is a diagonal matrix with positive, decreasing entries $d_1 \ge d_2 \ge \cdots \ge d_n$, with $\frac{d_1}{d_n} = \kappa$. Let $y = (Qu)^{\circ 2}$, where $^{\circ 2}$ denotes entry-wise square. Then the quantity we are interested in can be represented as

$$\hat{y} = \arg\inf_{y\in\Delta} \frac{\sum_{i=1}^n d_i y_i}{\sqrt{\sum_{i=1}^n d_i^2 y_i}},$$

and $\frac{u^\top M u}{\|Mu\|}$ is lower bounded by the optimal objective value of the above optimization problem. The set $\Delta = \{y \ge 0 : \mathbf{1}^\top y = 1\}$ is the simplex. Note that $\hat{y}$ is invariant to any positive scaling on the entire vector d, so we can consider a convexification, where the minimum is attained since y is confined to a compact set, and where $\hat{y}$ is the solution to:

$$\begin{array}{ll} \textbf{min.} & \hat{d}^T y \\ \textbf{subj.to} & \sum_{i=1}^n \hat{d}_i^2 y_i = 1 \\ & y \ge 0,\ y^T \mathbf{1} = 1 \end{array}$$

and $\hat{d}$ is the scaled version of d such that $\sum_{i=1}^n \hat{d}_i^2 y_i = 1$. The Lagrange dual is

$$\begin{array}{ll} \textbf{max.} & \alpha + \beta \\ \textbf{subj.to} & \hat{d} - \beta \hat{d}^{\circ 2} - \alpha \mathbf{1} \geq \mathbf{0} \end{array}$$

where by complementary slackness, the gap in the inequality corresponds to the nonzero pattern of y, e.g.

$$y_i(\hat{d}_i - \beta \hat{d}_i^2 - \alpha) = 0, \ \forall i \in [n].$$

Now we consider the quadratic function $d - \beta d^2 - \alpha$ in terms of a scalar d, with fixed α and β. It is clear that this concave function can have at most 2 zeros. In other words, there can be at most two indices i, j where

$$\hat{d}_k - \beta \hat{d}_k^2 - \alpha = 0, \ k \in \{i, j\}, \quad y_k = 0 \ k \notin \{i, j\}$$

and it is possible that $i = j$. (At least one solution must exist, since $y^T \mathbf{1} = 1$, so y cannot be all 0.) This reduces our primal optimization problem significantly, to two variables y_i and y_j, where the two linear constraints can be inverted directly

$$y_i = \frac{1 - \hat{d}_j^2}{\hat{d}_i^2 - \hat{d}_j^2}, \ y_j = \frac{\hat{d}_i^2 - 1}{\hat{d}_i^2 - \hat{d}_j^2}$$

Plugging back into the primal objective,

$$y_i \hat{d}_i + y_j \hat{d}_j = \frac{\hat{d}_i - \hat{d}_i \hat{d}_j^2 + \hat{d}_j \hat{d}_i^2 - \hat{d}_j}{\hat{d}_i^2 - \hat{d}_j^2} = \frac{1 + (\hat{d}_i \hat{d}_j)}{\hat{d}_i + \hat{d}_j}$$

which is minimized when $\hat{d}_i$ and $\hat{d}_j$ are as far apart as possible; e.g., $i = 1$ and $j = n$.

We can equivalently reparametrize y_1 and y_n in terms of d_1 and d_n as

$$y_1(c) = \frac{c^2 - d_n^2}{d_1^2 - d_n^2}, \ y_n(c) = \frac{d_1^2 - c^2}{d_1^2 - d_n^2}$$

where $c^2 = d_1^2 y_1 + d_n^2 y_n$. Now we consider our original optimization problem, significantly simplified

$$\inf_{y \in \Delta} \frac{\sum_{i=1}^n d_i y_i}{\sqrt{\sum_{i=1}^n d_i^2 y_i}} = \inf_{c \geq 0} \frac{d_1 y_1(c) + d_n y_n(c)}{c} = \inf_{c \geq 0} \frac{c^2 + d_1 d_n}{c(d_1 + d_n)}.$$

This is now a scalar nonconvex optimization problem, with minimum at either the boundaries ($c = 0$ or $c \to +\infty$) or the one stationary point ($c = d_1 d_n$). The stationary point is the only one in which this quantity is not positive infinity, and is thus the solution. Therefore, simplifying with $\kappa = d_1/d_n$, we arrive at

$$\hat{y}_1 = \frac{1}{\kappa + 1}, \ \hat{y}_n = \frac{\kappa}{\kappa + 1}.$$

and for any unit-normed u,

$$\frac{u^\top M u}{\|Mu\|} \geq \frac{c^2 + d_1 d_n}{c(d_1 + d_n)} = \frac{(1 + d_1 d_n)}{(d_1 + d_n)} = \frac{(1/d_n + d_1)}{(\kappa + 1)} \geq \frac{2\sqrt{d_1/d_n)}}{(\kappa + 1)} = \frac{2\sqrt{\kappa})}{(\kappa + 1)}.$$

where the inequality comes from the arithmetic-geometric mean inequality. ■

A.2. Proof of Theorem 2

The proof of the first part of the theorem is a special case of the proof of Theorem 5 (with $\lambda = 0$, and the smooth regularizer incorporated into $p(w)$) and thus we omit this proof and refer the reader to Appendix B. We prove the second part here.

Note that because of the condition in (6) (BRIP), we have

$$(1 - \epsilon) I \preceq S_A^\top S_A \preceq (1 + \epsilon) I.$$

Using smoothness of the objective, and the choices $d_t = -\nabla \widetilde{f}^A(w_t)$ and $\alpha_t = \alpha$, we have

$$\begin{aligned} &\widetilde{f}^A(w_{t+1}) - \widetilde{f}^A(w_t) \leq \alpha \nabla \widetilde{f}^A(w_t)(w_t)^\top d_t + \frac{1}{2}\alpha^2 d_t^\top X^\top S_A^\top S_A X d_t + \lambda \frac{L}{2}\alpha^2 \|d_t\|^2 \\ &\leq -\alpha \left(1 - \frac{(1+\epsilon)M + \lambda L}{2}\alpha\right) \left\|\nabla \widetilde{f}^A(w_t)\right\|^2 = -\frac{2\zeta(1-\zeta)}{(1+\epsilon)M + \lambda L} \|\nabla \widetilde{f}^A(w_t)\|^2 \\ &\overset{(a)}{\leq} -\frac{4\nu\zeta(1-\zeta)}{M(1+\epsilon) + \lambda L} \left(\widetilde{f}^A(w_t) - \widetilde{f}^A(\widetilde{w}_t^*)\right), \end{aligned}$$

where (a) follows by strong convexity. Re-arranging this inequality, and using the definition of γ, we get

$$\widetilde{f}_{t+1}^A - \widetilde{f}^A(\widetilde{w}_t^*) \leq \gamma \left(\widetilde{f}_t^A - \widetilde{f}^A(\widetilde{w}_t^*)\right),$$

which, using Lemma 12, implies the result.

A.3. Proof of Theorem 4

We first present the proof of Lemma 3, then move on to the main proof.

Proof [Proof of Lemma 3] The reader is referred to Section 2.1 for notation. Define $\breve{S}_t := S_{A_t \cap A_{t-1}}$. First note that

$$\begin{aligned} r_t^\top u_t &= \left(X^\top \breve{S}_t^\top \breve{S}_t \left[(Xw_t - y) - (Xw_{t-1} - y)\right]\right)^\top (w_t - w_{t-1}) \\ &= (w_t - w_{t-1})^\top X^\top \breve{S}_t^\top \breve{S}_t X (w - w_{t-1}) \\ &\geq \delta \mu \|u_t\|^2, \end{aligned} \tag{16}$$

by (7), where recall $\delta \geq 0$ is the smallest eigenvalue of $\breve{S}_t^\top \breve{S}_t$. Also consider

$$\frac{\|r_t\|^2}{r_t^\top u_t} = \frac{(w_t - w_{t-1})^\top \left(X^\top \breve{S}_t^\top \breve{S}_t X\right) \left(X^\top \breve{S}_t^\top \breve{S}_t X\right) (w_t - w_{t-1})}{(w_t - w_{t-1})^\top X^\top \breve{S}_t^\top \breve{S}_t X (w_t - w_{t-1})},$$

which implies

$$\delta \mu \leq \frac{\|r_t\|^2}{r_t^\top u_t} \leq (1 + \epsilon) M,$$

again by (6). Now, setting $j_\ell = t - \tilde{\sigma} + \ell$, consider the trace

$$\mathbf{tr}\left(B_t^{(\ell+1)}\right) = \mathbf{tr}\left(B_t^{(\ell)}\right) - \mathbf{tr}\left(\frac{B_t^{(\ell)} u_{j_\ell} u_{j_\ell}^\top B_t^{(\ell)}}{u_{j_\ell}^\top B_t^{(\ell)} u_{j_\ell}}\right) + \mathbf{tr}\left(\frac{r_{j_\ell} r_{j_\ell}^\top}{r_{j_\ell}^\top u_{j_\ell}}\right)$$

$$\begin{aligned}
&\overset{(a)}{\leq} \mathbf{tr}\left(B_t^{(\ell)}\right) + \mathbf{tr}\left(\frac{r_{j_\ell} r_{j_\ell}^\top}{r_{j_\ell}^\top u_{j_\ell}}\right) \\
&= \mathbf{tr}\left(B_t^{(\ell)}\right) + \frac{\|r_{j_\ell}\|^2}{r_{j_\ell}^\top u_{j_\ell}} \\
&\leq \mathbf{tr}\left(B_t^{(\ell)}\right) + (1+\epsilon)M,
\end{aligned}$$

where (a) follows since the trace of a positive semidefinite matrix is always non-negative. This implies $\mathbf{tr}\left(B_t\right) \leq (1+\epsilon)M\left(\widetilde{\sigma}+d\right)$. We now do a similar exercise for the determinant. Note first that for two arbitrary vectors a and b,

$$\begin{aligned}
\det(I - aa^T + bb^T) &= \det\left(I + \begin{bmatrix}-a & b\end{bmatrix}\begin{bmatrix}a^T \\ b^T\end{bmatrix}\right) \overset{(a)}{=} \det\left(I + \begin{bmatrix}a^T \\ b^T\end{bmatrix}\begin{bmatrix}-a & b\end{bmatrix}\right) \\
&= (1 - a^T a)(1 + b^T b) + (a^T b)^2
\end{aligned}$$

where (a) is by Sylvester's identity. Since $B_t^{(\ell)} \succ 0$, we can take the Cholesky factorization $B_t^{(\ell)} = LL^T$ and write the recursion for $B_t^{(\ell)}$ as a product:

$$B_t^{(\ell+1)} \quad = \quad L\left(I - L^T \frac{u_{j_\ell} u_{j_\ell}^T}{u_{j_\ell}^T B_t^{(\ell)} u_{j_\ell}} L + L^{-1} \frac{r_{j_\ell} r_{j_\ell}^T}{r_{j_\ell}^T u_{j_\ell}} L^{-T}\right) L^T.$$

Using the above identity for

$$a = \frac{1}{u_{j_\ell}^T B_t^{(\ell)} u_{j_\ell}} L^T u_{j_\ell}, \qquad b = \frac{1}{\sqrt{r_{j_\ell}}^T u_{j_\ell}} L^T r_{j_\ell}$$

we can see that $a^T a = 1$, and $(a^T b)^2 = \frac{r_{j_\ell}^T u_{j_\ell}}{u_{j_\ell}^T B_t^{(\ell)} u_{j_\ell}}$. Therefore

$$\begin{aligned}
\det\left(B_t^{(\ell+1)}\right) &= \det\left(B_t^{(\ell)}\right) \cdot \frac{r_{j_\ell}^\top u_{j_\ell}}{u_{j_\ell}^\top B_t^{(\ell)} u_{j_\ell}} \\
&= \det\left(B_t^{(\ell)}\right) \cdot \frac{r_{j_\ell}^\top u_{j_\ell}}{\|u_{j_\ell}\|^2} \cdot \frac{\|u_{j_\ell}\|^2}{u_{j_\ell}^\top B_t^{(\ell)} u_{j_\ell}} \\
&\geq \det\left(B_t^{(\ell)}\right) \cdot \delta\mu \cdot \frac{\|u_{j_\ell}\|^2}{u_{j_\ell}^\top B_t^{(\ell)} u_{j_\ell}} \\
&\geq \det\left(B_t^{(\ell)}\right) \cdot \delta\mu \cdot \frac{1}{\lambda_{\min}(B_t^{(\ell)})} \\
&\geq \det\left(B_t^{(\ell)}\right) \cdot \delta\mu \cdot \frac{1}{\mathbf{tr}\left(B_t^{(\ell)}\right)} \\
&\geq \det\left(B_t^{(\ell)}\right) \frac{\delta\mu}{(1+\epsilon)M\left(\widetilde{\sigma}+d\right)},
\end{aligned}$$

which implies $\det\left(B_t\right) \geq \det\left(B_t^{(0)}\right)\left(\frac{\delta\mu}{(1+\epsilon)M(\widetilde{\sigma}+d)}\right)^{\widetilde{\sigma}}$. By construction, $\mathbf{tr}\left(B_t^{(0)}\right)$ is bounded above and $\det(B_t^{(0)})$ is bounded below, which means $\mathbf{tr}\left(B_t\right)$ is bounded above and $\det(B_t)$ is bounded below. Therefore there must exist $0 < c_1 \leq c_2$ such that

$$c_1 I \preceq B_t \preceq c_2 I.$$

■

Proof [Proof of Theorem 4]

Since $h(w)$ is constrained to be quadratic, we can absorb this term into the error term to get

$$\min_w \left\| \begin{bmatrix} S & 0 \\ 0 & I \end{bmatrix} \left(\begin{bmatrix} X \\ \sqrt{\lambda} I \end{bmatrix} w - \begin{bmatrix} y \\ 0 \end{bmatrix} \right) \right\|.$$

Note that as long as S satisfies (6), the effective encoding matrix $\mathrm{diag}\left([S, I]\right)$ also satisfies the same. Therefore, without loss of generality we can ignore $h(w)$, and consider minimizing the unregularized quadratic problem

$$\min_w \|\hat{X}w - y\|, \qquad \hat{X} = \begin{bmatrix} X \\ \sqrt{\lambda} I \end{bmatrix}.$$

Here,

$$(\mu + \lambda) I \preceq \hat{X}^\top \hat{X} \preceq (M + \lambda) I.$$

We also define the extreme eigenvalues of the encoding matrices as $\lambda_{\min} = 1 - \epsilon$ and $\lambda_{\max} = 1 + \epsilon$ for convenience. Using convexity and the closed-form expression for the step size, we have

$$\begin{aligned}
&\widetilde{f}^A\left(w_{t+1}\right) - \widetilde{f}^A(w_t) \leq \alpha_t \nabla \widetilde{f}^A(w_t)^\top d_t + \frac{1}{2}\alpha_t^2 d_t^\top \hat{X}^\top S_A^\top S_A \hat{X} d_t \\
&= -\frac{\rho\left(\nabla \widetilde{f}^A(w_t)^\top d_t\right)^2}{d_t^\top \hat{X}^\top S_D^\top S_D \hat{X} d_t} + \frac{1}{2}\frac{\rho^2\left(\nabla \widetilde{f}^A(w_t)^\top d_t\right)^2}{d_t^\top \hat{X}^\top S_D^\top S_D \hat{X} d_t} \cdot \frac{d_t^\top \hat{X}^\top S_A^\top S_A \hat{X} d_t}{d_t^\top \hat{X}^\top S_D^\top S_D \hat{X} d_t} \\
&= \left(\frac{d_t^\top \hat{X}^\top \left(\rho^2 S_A^\top S_A - 2\rho S_D^\top S_D\right) \hat{X} d_t}{2\left(d_t^\top \hat{X}^\top S_D^\top S_D \hat{X} d_t\right)^2}\right)\left(d_t^\top \nabla \widetilde{f}^A(w_t)\right)^2 \\
&\overset{(a)}{=} -\rho\left(\frac{z^\top\left(S_D^\top S_D - \frac{\rho}{2} S_A^\top S_A\right) z}{\left(z^\top S_D^\top S_D z\right)^2}\right)\frac{\left(d_t^\top \nabla \widetilde{f}^A(w_t)\right)^2}{\|\hat{X} d_t\|^2} \\
&\overset{(b)}{\leq} -\rho\left(\frac{\lambda_{\min} - \frac{\rho}{2}\lambda_{\max}}{\lambda_{\min}^2}\right)\frac{\left(d_t^\top \nabla \widetilde{f}^A(w_t)\right)^2}{\|\hat{X} d_t\|^2} \overset{(c)}{\leq} -\frac{\rho}{M+\lambda}\left(\frac{\lambda_{\min} - \frac{\rho}{2}\lambda_{\max}}{\lambda_{\min}^2}\right)\frac{\left(d_t^\top \nabla \widetilde{f}^A(w_t)\right)^2}{\|d_t\|^2} \\
&\overset{(d)}{=} -\frac{\rho}{M+\lambda}\left(\frac{\lambda_{\min} - \frac{\rho}{2}\lambda_{\max}}{\lambda_{\min}^2}\right)\frac{\left(\nabla \widetilde{f}^A(w_t)^\top B_t \nabla \widetilde{f}^A(w_t)\right)^2}{\|B_t \nabla \widetilde{f}^A(w_t)\|^2}
\end{aligned}$$

$$\overset{(e)}{\leq} -\frac{4\rho}{M+\lambda}\left(\frac{\lambda_{\min}-\frac{\rho}{2}\lambda_{\max}}{\lambda_{\min}^2}\right)\frac{c_1c_2}{(c_1+c_2)^2}\|\nabla\widetilde{f}^A(w_t)\|^2$$
$$\overset{(f)}{\leq} -\frac{8(\mu+\lambda)\rho}{M+\lambda}\left(\frac{\lambda_{\min}-\frac{\rho}{2}\lambda_{\max}}{\lambda_{\min}^2}\right)\frac{c_1c_2}{(c_1+c_2)^2}\left(\widetilde{f}\left(w_t\right)-\widetilde{f}\left(\widetilde{w}_t^*\right)\right)$$
$$\overset{(g)}{=} -\frac{4(\mu+\lambda)c_1c_2}{(M+\lambda)(1+\epsilon)\left(c_1+c_2\right)^2}\left(\widetilde{f}\left(w_t\right)-\widetilde{f}\left(\widetilde{w}_t^*\right)\right)\overset{(h)}{=}-\left(1-\gamma\right)\left(\widetilde{f}^A\left(w_t\right)-\widetilde{f}^A\left(\widetilde{w}_t^*\right)\right).$$

where (a) follows by defining $z=\frac{\hat{X}d_t}{\|\hat{X}d_t\|}$; (b) follows by (6); (c) follows by the assumption that $\hat{X}^\top\hat{X}\preceq(M+\lambda)I$; (d) follows by the definition of d_t; (e) follows by Lemmas 14 and 3; (f) follows by strong convexity of $\widetilde{f}$ (by Lemma 13), which implies $\|\nabla\widetilde{f}^A(w_t)\|^2\geq 2(\mu+\lambda)\left(\widetilde{f}\left(\theta_t\right)-\widetilde{f}\left(\widetilde{w}_t^*\right)\right)$; (g) follows by choosing $\rho=\frac{\lambda_{\min}}{\lambda_{\max}}$; and (h) follows using the definition of γ.

Re-arranging the inequality, we obtain

$$\widetilde{f}_{t+1}^A-\widetilde{f}^A\left(\widetilde{w}_t^*\right)\leq\gamma\left(\widetilde{f}_t^A-\widetilde{f}^A\left(\widetilde{w}_t^*\right)\right),$$

and hence applying Lemma 12, we get the desired result. ■

Appendix B. Proof of Theorem 5

Throughout this appendix, we will define $p(w)=\frac{1}{2}\|Xw-y\|^2$ and $\widetilde{p}_t(w)=\frac{1}{2}\|S_{A_t}\left(Xw-y\right)\|^2$ for convenience, where the normalization by $\sqrt{\eta}$ is absorbed into S_A. We will omit the normalization by n for brevity. Let us also define

$$w^*=\arg\min_w p(w)+\lambda h(w)$$

to be the true solution of the optimization problem.

By M-smoothness of $p(w)$,

$$\begin{aligned}p(w_{t+1})&\leq p(w_t)+\langle\nabla p(w_t),w_{t+1}-w_t\rangle+\frac{M}{2}\|w_{t+1}-w_t\|^2\\&\leq p(w^*)-\langle\nabla p(w_t),w^*-w_t\rangle+\langle\nabla p(w_t),w_{t+1}-w_t\rangle+\frac{M}{2}\|w_{t+1}-w_t\|^2\\&\leq p(w^*)-\langle\nabla p(w_t),w^*-w_t\rangle+\langle\nabla p(w_t),w_{t+1}-w_t\rangle+\frac{1}{2\alpha}\|w_{t+1}-w_t\|^2\\&\leq p(w^*)+\langle\nabla p(w_t),w_{t+1}-w^*\rangle+\frac{1}{2\alpha}\|w_{t+1}-w_t\|^2\end{aligned}\tag{17}$$

where the second line follows by convexity of p, and the third line follows since $\alpha<\frac{1}{M}$. Since $w_{t+1}=\arg\min_w\widetilde{F}_t(w)$, by optimality conditions

$$0\in\lambda\cdot\partial h(w_{t+1})+\nabla\widetilde{p}_t(w_t)+\frac{1}{\alpha}\left(w_{t+1}-w_t\right).\tag{18}$$

Since h is convex, any subgradient $g\in\partial h$ at $w=w_{t+1}$ satisfies

$$h(w^*)\geq h(w_{t+1})+\langle g,w^*-w_{t+1}\rangle,$$

and therefore (18) implies

$$\lambda h(w^*)\geq\lambda h(w_{t+1})-\langle\nabla\widetilde{p}_t(w_t),w^*-w_{t+1}\rangle-\frac{1}{\alpha}\langle w_{t+1}-w_t,w^*-w_{t+1}\rangle.\tag{19}$$

Combining (17) and (19),we have

$$f(w_{t+1})=p(w_{t+1})+\lambda h(w_{t+1})\tag{20}$$
$$\begin{aligned}&=p(w^*)+\langle\nabla p(w_t),w_{t+1}-w^*\rangle+\frac{1}{2\alpha}\|w_{t+1}-w_t\|^2\\&\leq f(w^*)+\langle\nabla p(w_t)-\nabla\widetilde{p}_t(w_t),w_{t+1}-w^*\rangle\\&\quad-\frac{1}{\alpha}\langle w_t-w_{t+1},w^*-w_{t+1}\rangle+\frac{1}{2\alpha}\|w_t-w_{t+1}\|^2\\&=f(w^*)+\langle\nabla p(w_t)-\nabla\widetilde{p}_t(w_t),w_{t+1}-w^*\rangle\\&\quad+\frac{1}{2\alpha}\left(\|w_t\|^2-2w_t^\top w^*+\|w^*\|^2+2w_{t+1}^\top w^*-\|w^*\|^2-\|w_{t+1}\|^2\right)\\&=f(w^*)+\langle\nabla p(w_t)-\nabla\widetilde{p}_t(w_t),w_{t+1}-w^*\rangle\\&\quad+\frac{1}{2\alpha}\left(\|w_t-w^*\|^2-\|w_{t+1}-w^*\|^2\right)\end{aligned}\tag{21}$$
$$\tag{22}$$

Define $\Delta=I-S_A^\top S_A$, and consider the second term on the right-hand side of (22).

$$\begin{aligned}&\langle\nabla p(w_t)-\nabla\widetilde{p}_t(w_t),w_{t+1}-w^*\rangle=\left\langle X^\top\Delta(Xw_t-y),w_{t+1}-w^*\right\rangle\\&=\langle\Delta(Xw_t-y),Xw_{t+1}-y\rangle-\langle\Delta(Xw_t-y),Xw^*-y\rangle\\&=\frac{1}{2}\Big[\left(X\left(w_t+w_{t+1}\right)-2y\right)^\top\Delta\left(X\left(w_t+w_{t+1}\right)-2y\right)\\&\quad-\left(Xw_{t+1}-y\right)^\top\Delta\left(Xw_{t+1}-y\right)+\left(Xw^*-y\right)^\top\Delta\left(Xw^*-y\right)\\&\quad-\left(X\left(w_t+w^*\right)-2y\right)^\top\Delta\left(X\left(w_t+w^*\right)-2y\right)\Big]\\&=2\left(X\left(\frac{w_t+w_{t+1}}{2}\right)-y\right)^\top\Delta\left(X\left(\frac{w_t+w_{t+1}}{2}\right)-y\right)\\&\quad-2\left(X\left(\frac{w_t+w^*}{2}\right)-y\right)^\top\Delta\left(X\left(\frac{w_t+w^*}{2}\right)-y\right)\\&\quad-\frac{1}{2}\left(Xw_{t+1}-y\right)^\top\Delta\left(Xw_{t+1}-y\right)+\frac{1}{2}\left(Xw^*-y\right)^\top\Delta\left(Xw^*-y\right)\\&\leq 4\epsilon p\left(\frac{w_t+w_{t+1}}{2}\right)+4\epsilon p\left(\frac{w_t+w^*}{2}\right)+\epsilon p(w_{t+1})+\epsilon p(w^*)\\&\overset{(a)}{\leq}2\epsilon p(w_t)+2\epsilon p(w_{t+1})+2\epsilon p(w_t)+2\epsilon p(w^*)+\epsilon p(w_{t+1})+\epsilon p(w^*)\\&=\epsilon\left[4p(w_t)+3p(w_{t+1})+3p(w^*)\right]\\&\overset{(b)}{\leq}\epsilon\left[4f(w_t)+3f(w_{t+1})+3f(w^*)\right],\end{aligned}$$

where (a) is by convexity of $p(w)$ and Jensen's inequality, and (b) follows by non-negativity of h. Plugging this back in (22),

$$(1-3\epsilon)\, f(w_{t+1}) - 4\epsilon f(w_t) \leq (1+3\epsilon)\, f(w^*) + \frac{1}{2\alpha}\left(\|w_t - w^*\|^2 - \|w_{t+1} - w^*\|^2\right).$$

Adding this for $t = 0, \ldots, (T-1)$,

$$(1-7\epsilon)\sum_{t=1}^{T-1} f(w_t) + (1-3\epsilon) f(w_T) \leq T\,(1+3\epsilon)\, f(w^*) + 4\epsilon f(w_0) + \frac{1}{2\alpha}\left(\|w_0 - w^*\|^2 - \|w_T - w^*\|^2\right)$$

$$\Rightarrow (1-7\epsilon)\sum_{t=1}^{T} f(w_t) \leq T\,(1+3\epsilon)\, f(w^*) + 4\epsilon f(w_0) + \frac{1}{2\alpha}\left(\|w_0 - w^*\|^2 - \|w_T - w^*\|^2\right)$$

$$\leq T\,(1+3\epsilon)\, f(w^*) + 4\epsilon f(w_0) + \frac{1}{2\alpha}\|w_0 - w^*\|^2.$$

Defining $\bar{f}_t = \frac{1}{T}\sum_{t=1}^{T} f(w_t)$, and $\kappa = \frac{1+3\epsilon}{1-7\epsilon}$, we get

$$\bar{f}_T - \kappa f(w^*) \leq \frac{4\epsilon f(w_0) + \frac{1}{2\alpha}\|w_0 - w^*\|^2}{(1-7\epsilon)\, T},$$

which proves the first part of the theorem. To establish the second part of the theorem, note that the convexity of h implies

$$h(w_t) \geq h(w_{t+1}) + \langle g, w_t - w_{t+1}\rangle,$$

where $g \in \partial h(w_{t+1})$. By the optimality condition (18), this implies

$$\lambda h(w_t) \geq \lambda h(w_{t+1}) - \langle \nabla \widetilde{p}_t(w_t), w_t - w_{t+1}\rangle + \frac{1}{\alpha}\|w_{t+1} - w_t\|^2.$$

Combining this with the smoothness condition of $p(w)$,

$$p(w_{t+1}) \leq p(w_t) + \langle \nabla p(w_t), w_{t+1} - w_t\rangle + \frac{M}{2}\|w_{t+1} - w_t\|^2$$

and using the fact that $\alpha < \frac{1}{M}$, we have

$$f(w_{t+1}) \leq f(w_t) + \langle \nabla p(w_t) - \nabla \widetilde{p}_t(w_t), w_{t+1} - w_t\rangle - \frac{1}{2\alpha}\|w_t - w_{t+1}\|^2.$$

As in the previous analysis, we can show that

$$\langle \nabla p(w_t) - \nabla \widetilde{p}_t(w_t), w_{t+1} - w_t\rangle \leq \epsilon\left[7 f(w_t) + 3 f(w_{t+1})\right],$$

and therefore

$$f(w_{t+1}) \leq \frac{1+7\epsilon}{1-3\epsilon} f(w_t) - \frac{1}{2\alpha(1-3\epsilon)}\|w_t - w_{t+1}\|^2$$

$$\leq \frac{1+7\epsilon}{1-3\epsilon} f(w_t).$$

Appendix C. Proof of Theorem 6

For an iterate v_t, let $w_t := S v_t$. Define the solution set $\mathcal{S} = \arg\min_w g(w)$, and $w_t^* = \mathcal{P}_{\mathcal{S}}(w_t)$, where $\mathcal{P}_{\mathcal{S}}(\cdot)$ is the projection operator onto the set $\mathcal{S}$. Let v_t^* be such that $w_t^* = S^\top v_t^*$, which always exists since S has full column rank.

We also define $L' := L(1+\epsilon)$, and $g^* = \min_w g(w) = g(w_t^*)$ for any t.

C.1. Lemmas

Lemma 15 *$\widetilde{g}(v)$ is L'-smooth.*

Proof For any u, v,

$$\widetilde{g}(u) = g(S^\top u) \leq g(S^\top v) + \langle \nabla g(S^\top v), S^\top(u-v)\rangle + \frac{L}{2}\|S^\top(u-v)\|^2,$$

$$\overset{(a)}{\leq} g(S^\top v) + \langle S\nabla g(S^\top v), u - v\rangle + \frac{L(1+\epsilon)}{2}\|u-v\|^2,$$

$$\overset{(b)}{=} \widetilde{g}(v) + \langle \nabla \widetilde{g}(v), u - v\rangle + \frac{L(1+\epsilon)}{2}\|u-v\|^2,$$

where (a) follows from smoothness of g, and from (m, η, ϵ)-BRIP property, and (b) is by the chain rule of derivatives and the definition of $\widetilde{g}(v)$. Therefore $\widetilde{g}$ is $L(1+\epsilon)$-smooth. ■

Lemma 16 *For any t,*

$$\widetilde{g}^* := \min_v \widetilde{g}(v) = \min_w g(w) =: g^*.$$

Proof It is clear that

$$\min_v \tilde{g}(v) = \min_v g(S^\top v) \geq \min_w g(w).$$

To show the other direction, set $v^* = S(S^\top S)^{-1} w^*$, where $S^\top S$ is invertible since S has full column rank. Then $g(w^*) = \tilde{g}(v^*) \geq \min_v \tilde{g}(v)$. ■

Lemma 17 *If g is ν-restricted-strongly convex, then*

$$g(w) - g^* \geq \frac{\nu}{2}\|w - w^*\|^2,$$

where $w^* = \mathcal{P}_{\mathcal{S}}(w)$.

Proof We follow the proof technique in Zhang and Yin (2013). We have

$$g(w) = g^* + \int_0^1 \langle \nabla g(w^* + \tau(w - w^*)), w - w^*\rangle d\tau$$

$$= g^* + \int_0^1 \frac{1}{\tau}\langle \nabla g(w^* + \tau(w - w^*)), \tau(w - w^*)\rangle d\tau$$

$$\overset{(a)}{\geq} g^* + \int_0^1 \frac{1}{\tau}\nu\tau^2\|w - w^*\|^2 d\tau$$
$$= g^* + \nu\|w - w^*\|^2 \int_0^1 \tau d\tau$$
$$= g^* + \frac{1}{2}\nu\|w - w^*\|^2,$$

which is the desired result, where in (a) we used ν-restricted strong convexity, and the fact that

$$\mathcal{P}_{\mathcal{S}}(w^* + \tau(w - w^*)) = w^*,$$

for all $\tau \in [0, 1]$, since $w^* = \mathcal{P}_{\mathcal{S}}(w)$ is the orthogonal projection. ■

C.2. Proof of Theorem 6

Recall that the step for block i at time t, $\Delta_{i,t}$, is defined by

$$\Delta_{i,t} := \begin{cases} -\alpha\nabla_i \widetilde{g}(v_{t-1}), & \text{if } i \in A_t \\ 0, & \text{otherwise.} \end{cases}$$

By smoothness and definition of Δ_t,

$$\begin{aligned}\widetilde{g}(v_{t+1}) - \widetilde{g}(v_t) &\leq \langle \nabla\widetilde{g}(v_t), \Delta_t\rangle + \frac{L'}{2}\|\Delta_t\|^2 \\ &= \sum_{i\in A_t}\left(\langle \nabla_i\widetilde{g}(v_t), \Delta_{i,t}\rangle + \frac{L'}{2}\|\Delta_{i,t}\|^2\right) \\ &= \sum_{i\in A_t}\left(-\frac{1}{\alpha}\langle \Delta_{i,t}, \Delta_{i,t}\rangle + \frac{L'}{2}\|\Delta_{i,t}\|^2\right) \\ &= -\left(\frac{1}{\alpha} - \frac{L'}{2}\right)\|\Delta_t\|^2. \end{aligned} \tag{23}$$

Now, for any t,

$$\begin{aligned}\widetilde{g}(v_t) - \widetilde{g}^* &\leq \langle \nabla\widetilde{g}(v_t), v_t^* - v_t\rangle = \left\langle S\nabla g(S^\top v_t), v_t^* - v_t\right\rangle \\ &\overset{(a)}{\leq} \left\|\nabla g(S^\top v_t)\right\| \cdot \left\|S^\top(v_t^* - v_t)\right\| = \left\|\nabla g(S^\top v_t)\right\| \cdot \|w_t^* - w_t\|, \end{aligned} \tag{24}$$

where (a) is due to Cauchy-Schwartz inequality. Using

$$\Delta_t = -\alpha P_t \begin{bmatrix} S_{A_t}\nabla g(S^\top v_t) \\ 0 \end{bmatrix},$$

where P_t is a block permutation matrix mapping $\{1, \ldots, k\}$ to the node indices in A_t, we have

$$\|\Delta_t\|^2 = \alpha^2 \nabla g(S^\top v_t)^\top S_{A_t}^\top P_t^\top P_t S_{A_t}\nabla g(S^\top v_t) \geq (1-\epsilon)\alpha^2\left\|\nabla g(S^\top v_t)\right\|^2. \tag{25}$$

Because of (23), we have

$$\widetilde{g}(v_{t+1}) - \widetilde{g}(v_t) = g(w_{t+1}) - g(w_t) \leq 0$$

when $\alpha < 2/L'$, and hence w_t is contained in the level set defined by the initial iterate for all t, *i.e.*,

$$w_t \in \{w : g(w) \leq g(w_0)\}.$$

By the diameter assumption on this set, we have $\|w_t - w_t^*\| \leq R$ for all t. Using this and (25) in (24), we get

$$\widetilde{g}(v_t) - \widetilde{g}^* \leq \frac{R}{\alpha}\sqrt{\frac{1}{1-\epsilon}}\|\Delta_t\|.$$

Note that the fact that $\epsilon > 0$ weakens the per-step bound here, since we do not take the steepest descent direction. Another penalty from non-zero $\epsilon > 0$ comes due to the increase in the smoothness parameter of the objective $L' = (1+\epsilon)L$, which slightly limits the maximum step size.

Combining the above with (23),

$$\widetilde{g}(v_{t+1}) - \widetilde{g}(v_t) \leq -\frac{(1-\epsilon)\alpha}{R^2}\left(1 - \frac{\alpha L'}{2}\right)\left(\widetilde{g}(v_t) - \widetilde{g}^*\right)^2.$$

Defining $\pi_t := \widetilde{g}(v_t) - \widetilde{g}^*$, and $C := \frac{(1-\epsilon)\alpha}{R^2}\left(1 - \frac{\alpha L'}{2}\right)$, this implies

$$\pi_{t+1} \leq \pi_t - C\pi_t^2.$$

Dividing both sides by $\pi_t\pi_{t+1}$, and noting that $\pi_{t+1} \leq \pi_t$ due to (23),

$$\frac{1}{\pi_t} \leq \frac{1}{\pi_{t+1}} - C\frac{\pi_t}{\pi_{t+1}} \leq \frac{1}{\pi_{t+1}} - C$$

Therefore

$$\frac{1}{\pi_t} \geq \frac{1}{\pi_0} + Ct,$$

which implies

$$\pi_t \leq \frac{1}{\frac{1}{\pi_0} + Ct}.$$

Since $g(w_t) = g(S^\top v_t) = \widetilde{g}(v_t)$ by definition, and $g^* = \widetilde{g}^*$ by Lemma 16, $\pi_t = g(w_t) - g^*$, and therefore we have established the first part of the theorem.

To prove the second part, we make the additional assumption that g satisfies ν-restricted-strong convexity, which, through Lemma 17, implies $g(w) - g^* \geq \frac{\nu}{2}\|w - w^*\|^2$, for $w^* = \mathcal{P}_{\mathcal{S}}(w)$. Plugging in $w = w_t$ then gives the bound

$$\|w_t - w_t^*\|^2 \leq \frac{2}{\nu}\pi_t.$$

Using this bound as well as (25) in (24), we have

$$\pi_t^2 \leq \frac{2\|\Delta_t\|^2}{\nu(1-\epsilon)\alpha^2}\pi_t.$$

Using (23), this gives

$$\pi_t \leq \frac{2}{\nu(1-\epsilon)\alpha^2}\left(\frac{1}{\alpha}-\frac{L'}{2}\right)^{-1}(\pi_t - \pi_{t+1}),$$

which, defining $\xi = \frac{2}{\nu(1-\epsilon)\alpha}\left(1-\frac{L'\alpha}{2}\right)^{-1}$, results in

$$\pi_t \leq \left(1-\frac{1}{\xi}\right)^t \pi_0,$$

which shows the desired result.

Appendix D. Full results of the Matrix factorization experiment

Tables 6 and 7 give the test and train RMSE for the Movielens 1-M recommendation task, with a random 80/20 train/test split.

	uncoded	replication	gaussian	paley	hadamard
	$m = 8$, $k = 1$				
train RMSE	0.804	0.783	0.781	**0.775**	0.779
test RMSE	0.898	0.889	0.877	**0.873**	0.874
runtime	1.60	1.76	2.24	1.82	1.82
	$m = 8$, $k = 4$				
train RMSE	0.770	0.766	0.765	**0.763**	0.765
test RMSE	0.872	0.872	**0.866**	0.868	0.870
runtime	2.96	3.13	3.64	3.34	3.18
	$m = 8$, $k = 6$				
train RMSE	0.762	0.760	0.762	**0.758**	0.760
test RMSE	0.866	0.871	0.864	**0.860**	0.864
runtime	5.11	4.59	5.70	5.50	5.33

Table 6: Full results for Movielens 1-M, distributed over $m = 8$ nodes total. Runtime is in hours. An uncoded scheme running full batch L-BFGS has a train/test RMSE of 0.756 / 0.861, and a runtime of 9.58 hours.

	uncoded	replication	gaussian	paley	hadamard
	$m = 24$, $k = 3$				
train RMSE	0.805	0.791	0.783	**0.780**	0.782
test RMSE	0.902	0.893	0.880	**0.879**	0.882
runtime	2.60	3.22	3.98	3.49	3.49
	$m = 24$, $k = 12$				
train RMSE	0.770	**0.764**	0.767	**0.764**	0.765
test RMSE	0.872	0.870	**0.866**	0.868	0.868
runtime	4.24	4.38	4.92	4.50	4.61

Table 7: Full results for Movielens 1-M, distributed over $m = 24$ nodes total. Runtime is in hours. An uncoded scheme running full batch L-BFGS has a train/test RMSE of 0.757 / 0.862, and a runtime of 14.11 hours.

References

Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 873–881, 2011.

Ganesh Ananthanarayanan, Ali Ghodsi, Scott Shenker, and Ion Stoica. Effective straggler mitigation: Attack of the clones. In *NSDI*, volume 13, pages 185–198, 2013.

Albert S Berahas, Jorge Nocedal, and Martin Takác. A multi-batch l-bfgs method for machine learning. In *Advances in Neural Information Processing Systems*, pages 1055–1063, 2016.

Emmanuel J Candes and Terence Tao. Decoding by linear programming. *IEEE transactions on information theory*, 51(12):4203–4215, 2005.

Emmanuel J Candes and Terence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE transactions on information theory*, 52(12): 5406–5425, 2006.

Jeffrey Dean and Luiz André Barroso. The tail at scale. *Communications of the ACM*, 56 (2):74–80, 2013.

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. Large scale distributed deep networks. In *Advances in neural information processing systems*, pages 1223–1231, 2012.

Petros Drineas, Michael W Mahoney, S Muthukrishnan, and Tamás Sarlós. Faster least squares approximation. *Numerische mathematik*, 117(2):219–249, 2011.

Sanghamitra Dutta, Viveck Cadambe, and Pulkit Grover. Short-dot: Computing large linear transforms distributedly using coded short dot products. In *Advances In Neural Information Processing Systems*, pages 2092–2100, 2016.

Matthew Fickus, Dustin G Mixon, and Janet C Tremain. Steiner equiangular tight frames. *Linear algebra and its applications*, 436(5):1014–1027, 2012.

Kristen Gardner, Samuel Zbarsky, Sherwin Doroudi, Mor Harchol-Balter, and Esa Hyytia. Reducing latency via redundant requests: Exact analysis. *ACM SIGMETRICS Performance Evaluation Review*, 43(1):347–360, 2015.

Stuart Geman. A limit theorem for the norm of random matrices. *The Annals of Probability*, pages 252–261, 1980.

J.M. Goethals and J Jacob Seidel. Orthogonal matrices with zero diagonal. *Canad. J. Math*, 1967.

Wael Halbawi, Navid Azizan-Ruhi, Fariborz Salehi, and Babak Hassibi. Improving distributed gradient descent using reed-solomon codes. *arXiv preprint arXiv:1706.05436*, 2017.

Martin Jaggi. An equivalence between the lasso and support vector machines. *arXiv preprint arXiv:1303.1152*, 2013.

Can Karakus, Yifan Sun, and Suhas Diggavi. Encoded distributed optimization. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 2890–2894. IEEE, 2017a.

Can Karakus, Yifan Sun, Suhas Diggavi, and Wotao Yin. Straggler mitigation in distributed optimization through data encoding. In *Advances in Neural Information Processing Systems*, pages 5440–5448, 2017b.

Ming-Jun Lai and Wotao Yin. Augmented \ell_1 and nuclear-norm models with a globally linearly convergent algorithm. *SIAM Journal on Imaging Sciences*, 6(2):1059–1091, 2013.

Kangwook Lee, Maximilian Lam, Ramtin Pedarsani, Dimitris Papailiopoulos, and Kannan Ramchandran. Speeding up distributed machine learning using codes. *IEEE Transactions on Information Theory*, 64(3):1514–1529, 2018.

David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397, 2004.

Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *OSDI*, volume 14, pages 583–598, 2014.

Yu-Feng Li, Ivor W Tsang, Jame Kwok, and Zhi-Hua Zhou. Tighter and convex maximum margin clustering. In *Artificial Intelligence and Statistics*, pages 344–351, 2009.

Ji Liu, Stephen J Wright, Christopher Ré, Victor Bittorf, and Srikrishna Sridhar. An asynchronous parallel stochastic coordinate descent algorithm. *The Journal of Machine Learning Research*, 16(1):285–322, 2015.

Michael W Mahoney et al. Randomized algorithms for matrices and data. *Foundations and Trends® in Machine Learning*, 3(2):123–224, 2011.

Aryan Mokhtari and Alejandro Ribeiro. Global convergence of online limited memory BFGS. *Journal of Machine Learning Research*, 16:3151–3181, 2015.

Raymond EAC Paley. On orthogonal matrices. *Studies in Applied Mathematics*, 12(1-4): 311–320, 1933.

Zhimin Peng, Yangyang Xu, Ming Yan, and Wotao Yin. Arock: an algorithmic framework for asynchronous parallel coordinate updates. *SIAM Journal on Scientific Computing*, 38(5):A2851–A2879, 2016.

Mert Pilanci and Martin J Wainwright. Randomized sketches of convex programs with sharp guarantees. *IEEE Transactions on Information Theory*, 61(9):5096–5115, 2015.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 693–701, 2011.

Amirhossein Reisizadeh, Saurav Prakash, Ramtin Pedarsani, and Salman Avestimehr. Coded computation over heterogeneous clusters. In *Information Theory (ISIT), 2017 IEEE International Symposium on*, pages 2408–2412. IEEE, 2017.

J Riedl and J Konstan. Movielens dataset, 1998.

Nihar B Shah, Kangwook Lee, and Kannan Ramchandran. When do redundant requests reduce latency? *IEEE Transactions on Communications*, 64(2):715–722, 2016.

Jack W Silverstein. The smallest eigenvalue of a large dimensional wishart matrix. *The Annals of Probability*, pages 1364–1368, 1985.

Tao Sun, Robert Hannah, and Wotao Yin. Asynchronous coordinate descent under more realistic assumptions. In *Advances in Neural Information Processing Systems*, pages 6183–6191, 2017.

Ferenc Szöllősi. Complex hadamard matrices and equiangular tight frames. *Linear Algebra and its Applications*, 438(4):1962–1967, 2013.

Rashish Tandon, Qi Lei, Alexandros G Dimakis, and Nikos Karampatziakis. Gradient coding. *ML Systems Workshop (MLSyS), NIPS*, 2016.

Rashish Tandon, Qi Lei, Alexandros G Dimakis, and Nikos Karampatziakis. Gradient coding: Avoiding stragglers in distributed learning. In *International Conference on Machine Learning*, pages 3368–3376, 2017.

Da Wang, Gauri Joshi, and Gregory Wornell. Using straggler replication to reduce latency in large-scale parallel computing. *ACM SIGMETRICS Performance Evaluation Review*, 43(3):7–11, 2015.

Lloyd Welch. Lower bounds on the maximum cross correlation of signals (corresp.). *IEEE Transactions on Information theory*, 20(3):397–399, 1974.

N J. Yadwadkar, B. Hariharan, J. Gonzalez, and R H. Katz. Multi-task learning for straggler avoiding predictive job scheduling. *Journal of Machine Learning Research*, 17(4):1–37, 2016.

Yaoqing Yang, Pulkit Grover, and Soummya Kar. Coded distributed computing for inverse problems. In *Advances in Neural Information Processing Systems*, pages 709–719, 2017.

Yang You, Xiangru Lian, Ji Liu, Hsiang-Fu Yu, Inderjit S Dhillon, James Demmel, and Cho-Jui Hsieh. Asynchronous parallel greedy coordinate descent. In *Advances in Neural Information Processing Systems*, pages 4682–4690, 2016.

Hui Zhang and Wotao Yin. Gradient methods for convex minimization: better rates under weaker conditions. *arXiv preprint arXiv:1303.4645*, 2013.

Journal of Machine Learning Research 20 (2019) 1-46 Submitted 3/18; Revised 1/19; Published 2/19

Analysis of Langevin Monte Carlo via Convex Optimization

Alain Durmus alain.durmus@cmla.ens-cachan.fr
CMLA - École normale supérieure Paris-Saclay,
CNRS, Université Paris-Saclay, 94235 Cachan, France

Szymon Majewski smajewski@impan.pl
Institute of Mathematics, Polish Academy of Sciences
ul. Śniadeckich 8, 00-656 Warszawa, Poland

Błażej Miasojedow B.Miasojedow@mimuw.edu.pl
Institute of Applied Mathematics and Mechanics,
University of Warsaw, ul. Banacha 2, 02-097 Warszawa, Poland

Editor: Francois Caron

Abstract

In this paper, we provide new insights on the Unadjusted Langevin Algorithm. We show that this method can be formulated as the first order optimization algorithm for an objective functional defined on the Wasserstein space of order 2. Using this interpretation and techniques borrowed from convex optimization, we give a non-asymptotic analysis of this method to sample from log-concave smooth target distribution on $\mathbb{R}^d$. Based on this interpretation, we propose two new methods for sampling from a non-smooth target distribution. These new algorithms are natural extensions of the Stochastic Gradient Langevin Dynamics (SGLD) algorithm, which is a popular extension of the Unadjusted Langevin Algorithm for largescale Bayesian inference. Using the optimization perspective, we provide non-asymptotic convergence analysis for the newly proposed methods.

Keywords: Unadjasted Langevin Algorithm, convex optimization, Bayesian inference, gradient flow, Wasserstein metric

1. Introduction

This paper deals with the problem of sampling from a probability measure π on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$ which admits a density, also denoted by π, with respect to the Lebesgue measure given for all $x \in \mathbb{R}^d$ by

$$\pi(x) = \mathrm{e}^{-U(x)} \Big/ \int_{\mathbb{R}^d} \mathrm{e}^{-U(y)} \mathrm{d}y \ ,$$

where $U : \mathbb{R}^d \to \mathbb{R}$. This problem arises in various fields such that Bayesian statistical inference (Gelman et al., 2014), machine learning (Andrieu et al., 2003), ill-posed inverse problems (Stuart, 2010), and computational physics (Krauth, 2006). Common and current methods to tackle this problem are Markov Chain Monte Carlo methods (Brooks et al., 2011), for example the Hastings-Metropolis algorithm (Metropolis et al., 1953; Hastings, 1970) or Gibbs sampling (Geman and Geman, 1984). All these methods boil down to building a Markov kernel on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$ whose invariant probability distribution is π. Yet, choosing an appropriate proposal distribution for the Hastings-Metropolis algorithm is a tricky subject. For this reason, it has been proposed to consider continuous dynamics which naturally leave the target distribution π invariant. Perhaps, one of the most famous examples is the over-damped Langevin diffusion (Rossky et al., 1978; Parisi, 1981) associated with U, which is assumed to be continuously differentiable:

$$\mathrm{d}\mathbf{Y}_t = -\nabla U(\mathbf{Y}_t)\mathrm{d}t + \sqrt{2}\mathrm{d}B_t \ , \tag{1}$$

where $(B_t)_{t\geq 0}$ is a d-dimensional Brownian motion. With appropriate conditions on U, this SDE admits a unique strong solution $(\mathbf{Y}_t)_{t\geq 0}$ and defines a strong Markov semi-group $(P_t)_{t\geq 0}$ which converges to π in total variation (Roberts and Tweedie, 1996, Theorem 2.1) or Wasserstein distance (Bolley et al., 2012). However, simulating exact solutions of such stochastic differential equations is not possible in most cases, and discretizations of these equations are used instead. In addition, numerical solutions associated with these schemes define Markov kernels for which π is not invariant anymore. Therefore quantifying the error introduced by these approximations is crucial to justify their use to sample from the target π. We consider in this paper the Euler-Maruyama discretization of (1) which defines the (possibly inhomogeneous) Markov chain $(X_k)_{k\geq 0}$ given for all $k \geq 0$ by

$$X_{k+1} = X_k - \gamma_{k+1}\nabla U(X_k) + \sqrt{2\gamma_{k+1}}G_{k+1} \ , \tag{2}$$

where $(\gamma_k)_{k\geq 1}$ is a sequence of stepsizes which can be held constant or converges to 0, and $(G_k)_{k\geq 1}$ is a sequence of i.i.d. standard d-dimensional Gaussian random variables. The use of the Euler-Maruyama discretization (2) to approximatively sample from π is referred to as the Unadjusted Langevin Algorithm (ULA) (or the Langevin Monte Carlo algorithm (LMC)), and has already been the matter of many works. For example, weak error estimates have been obtained in Talay and Tubaro (1990); Mattingly et al. (2002) for the constant stepsize setting and in Lamberton and Pagès (2003); Lemaire (2005) when $(\gamma_k)_{k\geq 1}$ is non-increasing and goes to 0. Explicit and non-asymptotic bounds on the total variation (Dalalyan, 2016; Durmus and Moulines, 2017) or the Wasserstein distance (Durmus and Moulines, 2016) between the distribution of X_k and π have been obtained. Roughly, all these results are based on the comparison between the discretization and the diffusion process and the quantification of error accumulation throughout the algorithm. In this paper, we propose another point of view on ULA, which shares nevertheless some relations with the Langevin diffusion (1). Indeed, it has been shown in Jordan et al. (1998) that the family of distributions $(\mu_0 P_t)_{t\geq 0}$, is the solution of a gradient flow equation in the Wasserstein space of order 2 associated with the Kullback-Leibler (KL) divergence, $\mathscr{F}$, where $(P_t)_{t\geq 0}$ is the semi-group associated with (1) and μ_0 is a probability measure on $\mathcal{B}(\mathbb{R}^d)$ admitting a second moment (see Section 2). If π is invariant for $(P_t)_{t\geq 0}$, then it is a stationary solution of this equation, and is the unique minimizer of $\mathscr{F}$ if U is convex. Starting from this observation, we interpret ULA as the first order optimization algorithm on the Wasserstein space of order 2 with objective functional $\mathscr{F}$. Namely, we adapt some proofs of convergence for the gradient descent algorithm from the convex optimization literature to obtain non-asymptotic and explicit bounds between the Kullback-Leibler (KL) divergence from π to distributions of averaged distributions associated with ULA for the constant and non-increasing stepsize setting. Then, these bounds easily imply computable bounds in total variation norm and Wasserstein distance. Note that these two metrics are different in nature since convergence

©2019 Alain Durmus, Szymon Majewski, Błażej Miasojedow.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-173.html.

in one of them does not imply convergence in the other. Convergence in one of these metrics implies a control on the bias of MCMC based estimators of the form $\hat{f}_n = n^{-1}\sum_{k=1}^{n} f(Y_k)$, where $(Y_k)_{k\in\mathbb{N}}$ is Markov chain ergodic with respect to the target density π, for f belonging to a certain class of functions. In the case of the total variation distance, this class is the set of measurable and bounded functions, in the case of the Wasserstein distance, it is the set of Lipschitz functions. If the potential U is strongly convex and gradient Lipschitz, we get back the results of Durmus and Moulines (2017, 2016); Cheng and Bartlett (2017), when the stepsize is held constant in (2). In the case where U is only convex and from a warm start, we get a bound on the complexity for ULA of order $d\bar{\mathcal{O}}(\varepsilon^{-2})$ and $d\bar{\mathcal{O}}(\varepsilon^{-4})$ to get one sample distributed close from π with accuracy $\varepsilon > 0$, in KL divergence and total variation distance respectively. Overview of bounds on computational complexity of ULA is presented in Table 1.

	Strongly convex U		Convex U			
			Warm start		Minimizer of U	
	Best	Ours	Best	Ours	Best	Ours
TV	$d\bar{\mathcal{O}}(\varepsilon^{-2})$	$d\bar{\mathcal{O}}(\varepsilon^{-2})$	$d\bar{\mathcal{O}}(\varepsilon^{-6})$	$d\bar{\mathcal{O}}(\varepsilon^{-4})$	$d^5\bar{\mathcal{O}}(\varepsilon^{-2})$	$d^3\bar{\mathcal{O}}(\varepsilon^{-4})$
Wasserstein	$d\bar{\mathcal{O}}(\varepsilon^{-2})$	$d\bar{\mathcal{O}}(\varepsilon^{-2})$	–	–	–	–
KL	$d\bar{\mathcal{O}}(\varepsilon^{-1})$	$d\bar{\mathcal{O}}(\varepsilon^{-1})$	$d\bar{\mathcal{O}}(\varepsilon^{-3})$	$d\bar{\mathcal{O}}(\varepsilon^{-2})$	–	$d^3\bar{\mathcal{O}}(\varepsilon^{-2})$

Table 1: Overview of bounds on computational complexity of ULA, with constant stepsize. We present complexity to get one sample distributed close from π with accuracy $\varepsilon > 0$ in Wasserstein distance (Wasserstein), total variation distance (TV), and Kullback Leibler divergence (KL). We compare the best results (Best) from literature with ours (Ours) in the strongly convex and convex cases. In the convex case we consider two possible initial measures: a warm start, *i.e.* $W_2^2(\mu_0, \pi) \leq C$, for some absolute constant $C \geq 0$ (Warm start) and starting from Dirac delta at a minimizer of U, $x^\star$.

In addition, we propose two new algorithms to sample from a class of non-smooth log-concave distributions for which we derive computable non-asymptotic bounds as well. The first one can be applied to Lipschitz convex potential for which unbiased estimates of subgradients are available. Remarkably, the bounds we obtain for this algorithm depend on the dimension only through the initial condition and the variance of the stochastic subgradient estimates. Precisely, we get a bound on the complexity to get a sample with distribution close from π with accuracy $\varepsilon > 0$, of order $(M^2 + D^2)\bar{\mathcal{O}}(\varepsilon^{-2})$ in the case of the KL divergence and $(M^2 + D^2)\bar{\mathcal{O}}(\varepsilon^{-4})$ in the case of the total variation distance, where M is Lipschitz constant of the potential U and D^2 is a bound on the variance of the considered stochastic subgradient.

The second method we propose is a generalization of the Stochastic Gradient Langevin Dynamics algorithm (Welling and Teh, 2011), which extends ULA by replacing the gradient

with a sequence of i.i.d. unbiased estimators. In this new scheme, we assume that U can be decomposed as the sum of two functions U_1 and U_2, where U_1 is at least continuously differentiable and U_2 is only convex, and use stochastic gradient estimates for U_1 and the proximal operator associated with U_2. This new method is close to the one proposed in Durmus et al. (2018) but contrary on this work we do not need to approximate U_2 by its Moreau envelope. To get computable bounds from the target distribution π, we interpret this algorithm as a first order optimization algorithm and provide explicit bounds between the KL divergence from π to distributions associated with SGLD. In the case where U is strongly convex and gradient Lipschitz (*i.e.* $U_2 = 0$), we get back the same complexity as Dalalyan and Karagulyan (2017) which is of order $d\bar{\mathcal{O}}(\varepsilon^{-2})$ for the Wasserstein distance. We obtain the same complexity for the total variation distance and a complexity of order $d\bar{\mathcal{O}}(\varepsilon^{-1})$ for the KL divergence . In the case where U is only convex, not necessarily smooth (*i.e.* U_2 could be non-smooth), and from a warm start, we get a complexity of order $d\bar{\mathcal{O}}(\varepsilon^{-2})$ and $d\bar{\mathcal{O}}(\varepsilon^{-4})$ to get one sample distributed close from π with accuracy $\varepsilon > 0$ in KL divergence and total variation distance respectively. Overview of bounds for SGLD is presented in Table 2.

Extensive studies have also analyzed SGLD in a general setting, *i.e.* the potential U is not necessarily convex. In Vollmer et al. (2016) and Nagapetyan et al. (2017), a study of this scheme is done by weak error estimates. Finally, Raginsky et al. (2017) and Xu et al. (2017) gives some results regarding the potential use of SGLD as an optimization algorithm to minimize the potential U by targeting a target density proportional to $x \mapsto \mathrm{e}^{-\beta U(x)}$ for some $\beta > 0$.

	Strongly convex U		Convex U			
			Warm start		Minimizer of U	
	Best	Our	Best	Ours	Best	Ours
TV	–	$d\bar{\mathcal{O}}(\varepsilon^{-2})$	–	$d\bar{\mathcal{O}}(\varepsilon^{-4})$	–	$d^3\bar{\mathcal{O}}(\varepsilon^{-4})$
Wasserstein	$d\bar{\mathcal{O}}(\varepsilon^{-2})$	$d\bar{\mathcal{O}}(\varepsilon^{-2})$	–	–	–	–
KL	–	$d\bar{\mathcal{O}}(\varepsilon^{-1})$	–	$d\bar{\mathcal{O}}(\varepsilon^{-2})$	–	$d^3\bar{\mathcal{O}}(\varepsilon^{-2})$

Table 2: Overview of bounds on computational complexity of SGLD, with constant stepsize. We present complexity to get one sample distributed close from π with accuracy $\varepsilon > 0$ in Wasserstein distance (Wasserstein), total variation distance (TV), and Kullback Leibler divergence (KL). We compare the best results (Best) from literature with ours (Ours) in the strongly convex and convex cases. In the convex case we consider two possible initial measures: warm start, *i.e.* $W_2^2(\mu_0, \pi) \leq C$, for some absolute constant $C \geq 0$ (Warm start) and starting from Dirac delta at a minimizer of U, $x^\star$.

In summary, our contributions are the following:

- We give a new interpretation of ULA and use it to get bounds on the KL divergence from π to the iterates of ULA. We recover the dependence on the dimension of Cheng and Bartlett (2017, Theorem 3) in the strongly convex case and get tighter bounds. Note that this result implies previously known bounds between π and ULA in Wasserstein distance and the total variation distance but with a completely different technique. We also give computable bounds when U is only convex which improves the results of Durmus and Moulines (2017); Dalalyan (2016); Cheng and Bartlett (2017).
- We give two new methodologies to sample from a non-smooth potential U and make a non-asymptotic analysis of them. These two new algorithms are generalizations of SGLD.

The paper is organized as follows. In Section 2, we give some intuition on the strategy we take to analyze ULA and its variants. These ideas come from gradient flow theory in Wasserstein space. In Section 3, we give the main results we obtain on ULA and their proof. In Section 4, two variants of ULA are presented and analyzed. Finally, numerical experiments on logistic regression models are presented in Section 5 to support our theoretical findings regarding our new methodologies.

1.1. Notations and Conventions

Denote by $\mathcal{B}(\mathbb{R}^d)$ the Borel σ-field of $\mathbb{R}^d$, Leb the Lebesgue measure on $\mathcal{B}(\mathbb{R}^d)$, $\mathbb{F}(\mathbb{R}^d)$ the set of all Borel measurable functions on $\mathbb{R}^d$ and for $f \in \mathbb{F}(\mathbb{R}^d)$, $\|f\|_\infty = \sup_{x\in\mathbb{R}^d} |f(x)|$. For μ a probability measure on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$ and $f \in \mathbb{F}(\mathbb{R}^d)$ a μ-integrable function, denote by $\mu(f)$ the integral of f w.r.t. μ. Let μ and ν be two sigma-finite measures on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$. Denote by $\mu \ll \nu$ if μ is absolutely continuous w.r.t. ν and $\mathrm{d}\mu/\mathrm{d}\nu$ the associated density. Let μ, ν be two probability measures on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$. Define the Kullback-Leibler (KL) divergence of μ from ν by

$$\mathrm{KL}\left(\mu|\nu\right) = \begin{cases} \int_{\mathbb{R}^d} \frac{\mathrm{d}\mu}{\mathrm{d}\nu}(x) \log\left(\frac{\mathrm{d}\mu}{\mathrm{d}\nu}(x)\right) \mathrm{d}\nu(x) \;, & \text{if } \mu \ll \nu \\ +\infty & \text{otherwise .} \end{cases}$$

We say that ζ is a transference plan of μ and ν if it is a probability measure on $(\mathbb{R}^d \times \mathbb{R}^d, \mathcal{B}(\mathbb{R}^d \times \mathbb{R}^d))$ such that for all measurable set A of $\mathbb{R}^d$, $\zeta(\mathsf{A} \times \mathbb{R}^d) = \mu(\mathsf{A})$ and $\zeta(\mathbb{R}^d \times \mathsf{A}) = \nu(\mathsf{A})$. We denote by $\Pi(\mu, \nu)$ the set of transference plans of μ and ν. Furthermore, we say that a couple of $\mathbb{R}^d$-random variables (X, Y) is a coupling of μ and ν if there exists $\zeta \in \Pi(\mu, \nu)$ such that (X, Y) are distributed according to ζ. For two probability measures μ and ν, we define the Wasserstein distance of order 2 as

$$W_2(\mu, \nu) = \left(\inf_{\zeta \in \Pi(\mu,\nu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\|^2 \, \mathrm{d}\zeta(x, y) \right)^{1/2} . \tag{3}$$

By (Villani, 2009, Theorem 4.1), for all μ, ν probability measures on $\mathbb{R}^d$, there exists a transference plan $\zeta^\star \in \Pi(\mu, \nu)$ such that for any coupling (X, Y) distributed according to $\zeta^\star$, $W_2(\mu, \nu) = \mathbb{E}[\|X - Y\|^2]^{1/2}$. This kind of transference plan (respectively coupling) will be called an optimal transference plan (respectively optimal coupling) associated with W_2. We denote by $\mathcal{P}_2(\mathbb{R}^d)$ the set of probability measures with finite second moment: for all $\mu \in \mathcal{P}_2(\mathbb{R}^d)$, $\int_{\mathbb{R}^d} \|x\|^2 \, \mathrm{d}\mu(x) < +\infty$. By (Villani, 2009, Theorem 6.16), $\mathcal{P}_2(\mathbb{R}^d)$ equipped with the Wasserstein distance W_2 of order 2 is a complete separable metric space. Denote by $\mathcal{P}^{\mathrm{a}}(\mathbb{R}^d) = \{\mu \in \mathcal{P}_2(\mathbb{R}^d) \,:\, \mu \ll \mathrm{Leb}\}$. For two probability measures μ and ν on $\mathbb{R}^d$, the total variation distance distance between μ and ν is defined by $\|\mu - \nu\|_{\mathrm{TV}} = \sup_{\mathsf{A}\in\mathcal{B}(\mathbb{R}^d)} |\mu(\mathsf{A}) - \nu(\mathsf{A})|$.

Let $n \in \mathbb{N} \cup \{\infty\}$ and $\mathsf{U} \subset \mathbb{R}^d$ be an open set of $\mathbb{R}^d$. Denote by $C^n(\mathsf{U})$ the set of n-th continuously differentiable function from U to $\mathbb{R}$. Denote by $C^n_c(\mathsf{U})$ the set of n-th continuously differentiable function from U to $\mathbb{R}$ with compact support. Let $I \subset \mathbb{R}$ be an interval and $f : I \to \mathbb{R}$. f is absolutely continuous on I if for all $\varepsilon > 0$, there exists $\delta > 0$ such that for all $n \in \mathbb{N}^*$ and $t_1, \ldots, t_{2n} \in I$, $t_1 \le \cdots \le t_{2n}$,

$$\text{if } \sum_{k=1}^{n} \{t_{2k} - t_{2k-1}\} \le \delta \;\text{ then }\; \sum_{k=1}^{n} |f(t_{2k}) - f(t_{2k-1})| \le \varepsilon \;.$$

In the sequel, we take the convention that $\sum_p^n = 0$ and $\prod_p^n = 1$ for $n, p \in \mathbb{N}$, $n < p$.

2. Interpretation of ULA as an Optimization Algorithm

Throughout this paper, we assume that U satisfies the following condition for $m \ge 0$.

A 1 (m) $U : \mathbb{R}^d \to \mathbb{R}$ is m-convex, *i.e.* for all $x, y \in \mathbb{R}^d$,

$$U(tx + (1-t)y) \le tU(x) + (1-t)U(y) - t(1-t)(m/2)\|x - y\|^2$$

Note that $\mathbf{A}\,1(m)$ includes the case where U is only convex when $m = 0$. We consider in this Section the following additional condition on U which will be relaxed in Section 4.

A 2 U is continuously differentiable and L-gradient Lipschitz, *i.e.* there exists $L \ge 0$ such that for all $x, y \in \mathbb{R}^d$, $\|\nabla U(x) - \nabla U(y)\| \le L\|x - y\|$

Under $\mathbf{A}\,1$ and $\mathbf{A}\,2$, the Langevin diffusion (1) has a unique strong solution $(\mathbf{Y}^x_t)_{t\ge0}$ starting at $x \in \mathbb{R}^d$. The Markovian semi-group $(P_t)_{t\ge0}$, given for all $t \ge 0$, $x \in \mathbb{R}^d$ and $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$ by $P_t(x, \mathsf{A}) = \mathbb{P}(\mathbf{Y}^x_t \in \mathsf{A})$, is reversible with respect to π and π is its unique invariant probability measure, see (Ambrosio et al., 2009, Theorem 1.2 and Theorem 1.6). Using this probabilistic framework, (Roberts and Tweedie, 1996, Theorem 1.2) shows that $(P_t)_{t\ge0}$ is irreducible with respect to the Lebesgue measure, strong Feller and $\lim_{t\to+\infty} \|P_t(x, \cdot) - \pi\|_{\mathrm{TV}} = 0$ for all $x \in \mathbb{R}^d$. But to study the properties of the semi-group $(P_t)_{t\ge0}$, an other complementary and significant approach can be used. This dual point of view is based on the adjoint of the infinitesimal generator associated with $(P_t)_{t\ge0}$. The *strong* generator of (1) $(\mathcal{A}, \mathrm{D}(\mathcal{A}))$ is defined for all $f \in \mathrm{D}(\mathcal{A})$ and $x \in \mathbb{R}^d$ by

$$\mathcal{A}f(x) = \lim_{t\to0} t^{-1}(P_t f(x) - f(x)) \;,$$

where $\mathrm{D}(\mathcal{A})$ is the subset of $C_0(\mathbb{R}^d)$ such that for all $f \in \mathrm{D}(\mathcal{A})$, there exists $g \in C_0(\mathbb{R}^d)$ such that $\lim_{t\to0} \left\|t^{-1}(P_t f - f) - g\right\|_\infty = 0$. In particular for $f \in C^2_c(\mathbb{R}^d)$, we get by Itô's formula

$$\mathcal{A}f = \langle \nabla f, \nabla U \rangle + \Delta f \;.$$

In addition, by (Ethier and Kurtz, 1986, Proposition 1.5), for all $f \in C_c^2(\mathbb{R}^d)$, $P_t f(x) \in \mathrm{D}(\mathcal{A})$ and for $x \in \mathbb{R}^d$, $t \mapsto P_t f(x)$ is continuously differentiable,

$$\frac{\mathrm{d}P_t f(x)}{\mathrm{d}t} = \mathcal{A}P_t f(x) = P_t \mathcal{A} f(x) \; . \tag{4}$$

For all $\mu_0 \in \mathcal{P}_2^{\mathrm{a}}(\mathbb{R}^d)$ and $t > 0$, by Girsanov's Theorem (Karatzas and Shreve, 1991, Theorem 5.1, Corollary 5.16, Chapter 3), $\mu_0 P_t(\cdot)$ admits a density with respect to the Lebesgue measure denoted by ρ_t. This density is solution by (4) of the Fokker-Planck equation (in the weak sense):

$$\frac{\partial \rho_t}{\partial t} = \mathrm{div}(\nabla \rho_t + \rho_t \nabla U(x)) \; ,$$

meaning that for all $\phi \in C_c^\infty(\mathbb{R}^d)$ and $t > 0$,

$$\frac{\partial}{\partial t} \int_{\mathbb{R}^d} \phi(y) \rho_t(\mathrm{d}y) = \int_{\mathbb{R}^d} \mathcal{A}\phi(y) \, \rho_t(\mathrm{d}y) \; . \tag{5}$$

In the landmark paper Jordan et al. (1998), the authors shows that if U is infinitely continuously differentiable, $(\rho_t)_{t>0}$ is the limit of the minimization scheme which defines a sequence of probability measures $(\tilde{\rho}_{k,\gamma})_{k \in \mathbb{N}}$ as follows. For $\gamma > 0$ set $\rho_{0,\gamma} = \mathrm{d}\mu_0/\mathrm{d}\,\mathrm{Leb}$ and

$$\tilde{\rho}_{k+1,\gamma} = \frac{\mathrm{d}\tilde{\mu}_{k+1,\gamma}}{\mathrm{d}\,\mathrm{Leb}} \; , \qquad \tilde{\mu}_{k+1,\gamma} = \operatorname*{argmin}_{\mu \in \mathcal{P}_2^{\mathrm{a}}(\mathbb{R}^d)} \; W_2^2(\tilde{\mu}_{k,\gamma}, \mu)/2 + \gamma \mathscr{F}(\mu) \; , \; k \in \mathbb{N} \; , \tag{6}$$

where $\mathscr{F} : \mathcal{P}_2(\mathbb{R}^d) \to (-\infty, +\infty]$ is the free energy functional,

$$\mathscr{F} = \mathscr{H} + \mathscr{E} \; , \tag{7}$$

$\mathscr{H}, \mathscr{E} : \mathcal{P}_2(\mathbb{R}^d) \to (-\infty, +\infty]$ are the Boltzmann H-functional and the potential energy functional, given for all $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ by

$$\mathscr{H}(\mu) = \begin{cases} \int_{\mathbb{R}^d} \frac{\mathrm{d}\mu}{\mathrm{d}\,\mathrm{Leb}}(x) \log\left(\frac{\mathrm{d}\mu}{\mathrm{d}\,\mathrm{Leb}}(x)\right) \mathrm{d}x & \text{if } \mu \ll \mathrm{Leb} \\ +\infty \text{ otherwise} \; , \end{cases} \tag{8}$$

$$\mathscr{E}(\mu) = \int_{\mathbb{R}^d} U(x) \mathrm{d}\mu(x) \; . \tag{9}$$

More precisely, setting $\bar{\rho}_{0,\gamma} = \mathrm{d}\mu_0/\mathrm{d}\,\mathrm{Leb}$ and $\bar{\rho}_{t,\gamma} = \tilde{\rho}_{k,\gamma}$ for $t \in [k\gamma, (k+1)\gamma)$, (Jordan et al., 1998, Theorem 5.1) shows that for all $t > 0$, $\bar{\rho}_{t,\gamma}$ converges to ρ_t weakly in $\mathrm{L}^1(\mathbb{R}^d)$ as γ goes to 0. Note that the minimization scheme in (6) can be seen as a proximal type algorithm (see Martinet (1970) and Rockafeller (1976)) on the Wasserstein space $(\mathcal{P}_2(\mathbb{R}^d), W_2)$ used to minimize the functional $\mathscr{F}$. On $\mathbb{R}^d$, for continuous convex function f the proximal update with step size γ corresponds to one step of backward Euler discretization of the gradient flow ordinary differential equation (ODE) $\mathrm{d}x(t)/\mathrm{d}t = -\nabla f(x(t))$ with parameter γ. Therefore piecewise constant functions $(\bar{\rho}_{t,\gamma})_{\gamma>0}$ can be interpreted as backward Euler discretizations of an informal ODE $\partial \mu_t / \partial t = -\nabla \mathscr{F}(\mu_t)$ and their limit as $\gamma \to 0$, can be interpreted as a solution to this equation. This idea has been formalized and extended

to construct the framework of gradient flows in the Wasserstein space $(\mathcal{P}_2(\mathbb{R}^d), W_2)$, see Ambrosio et al. (2008). We provide a short introduction to this topic in Section A and present useful concepts and results for our proofs.

The following lemma shows that π is the unique minimizer of $\mathscr{F}$. As a result, the distribution of the Langevin diffusion is the steepest descent flow of $\mathscr{F}$ and we get back intuitively that this process converges to the target distribution π.

Lemma 1 *Assume* ***A*** *1(0). The following holds:*

a) $\pi \in \mathcal{P}_2(\mathbb{R}^d)$, $\mathscr{E}(\pi) < +\infty$ *and* $\mathscr{H}(\pi) < +\infty$.

b) For all $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ *satisfying* $\mathscr{E}(\mu) < \infty$

$$\mathscr{F}(\mu) - \mathscr{F}(\pi) = \mathrm{KL}\,(\mu|\pi) \; . \tag{10}$$

Proof The proof is postponed to Section 7.1. ■

Based on this interpretation, we could think about minimizing $\mathscr{F}$ on the Wasserstein space to get close to π using the minimization scheme (6). However, while this scheme is shown in Jordan et al. (1998) to be well-defined, finding explicit recursions $(\tilde{\rho}_{k,\gamma})_{k \in \mathbb{N}}$ is as difficult as minimizing $\mathscr{F}$ and can not be used in practice. In addition, to the authors knowledge, there is no efficient and practical schemes to optimize this functional. On the other hand, discretization schemes have been used to approximate the Langevin diffusion $(\mathbf{Y}_t)_{t \geq 0}$ (1) and its long-time behavior. One of the most popular method is the Euler-Maruyama discretization $(X_k)_{k \in \mathbb{N}}$ given in (2). While most work study the theoretical properties of this discretization to ensure to get samples close to the target distribution π, by comparing the distributions of $(X_k)_{k \in \mathbb{N}}$ and $(\mathbf{Y}_t)_{t \geq 0}$ through couplings or weak error expansions, we interpret this scheme as the first order optimization algorithm for the objective functional $\mathscr{F}$.A similar approach has been recently applied in Wibisono (2018) and in Bernton (2018). We postpone the comparison of our contributions and the results of this two papers in Section 4.2 where it is the most relevant.

3. Main Results for the Unadjusted Langevin Algorithm

Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex continuously differentiable objective function with $x_f \in \arg\min_{\mathbb{R}^d} f \neq \emptyset$. The *inexact* or *stochastic* gradient descent algorithm used to estimate $f(x_f)$ defines the sequence $(x_k)_{k \in \mathbb{N}}$ starting from $x_0 \in \mathbb{R}^d$ by the following recursion for $k \in \mathbb{N}$:

$$x_{k+1} = x_k - \gamma_{k+1} \nabla f(x_k) + \gamma_{k+1} \Xi(x_k) \; ,$$

where $(\gamma_k)_{k \in \mathbb{N}^*}$ is a non-increasing sequence of stepsizes and $\Xi : \mathbb{R}^d \to \mathbb{R}^d$ is a deterministic or/and stochastic perturbation of ∇f. To get explicit bound on the convergence (in expectation) of the sequence $(f(x_k))_{k \in \mathbb{N}}$ to $f(x_f)$, one possibility (see e.g. Beck and Teboulle (2009)) is to show that the following inequality holds: for all $k \in \mathbb{N}$,

$$2\gamma_{k+1}(f(x_{k+1}) - f(x_f)) \leq \|x_k - x_f\|^2 - \|x_{k+1} - x_f\|_2^2 + C\gamma_{k+1}^2 \; , \tag{11}$$

for some constant $C \geq 0$. In a similar manner as for inexact gradient algorithms, in this section we will establish that ULA satisfies an inequality of the form (11) with the objective function $\mathscr{F}$ defined by (7) on $\mathcal{P}_2(\mathbb{R}^d)$, but instead of the Euclidean norm, the Wasserstein distance of order 2 will be used.

Consider the family of Markov kernels $(R_{\gamma_k})_{k\in\mathbb{N}^*}$ associated with the Euler-Maruyama discretization $(X_k)_{k\in\mathbb{N}}$, (2), for a sequence of stepsizes $(\gamma_k)_{k\in\mathbb{N}^*}$, given for all $\gamma > 0, x \in \mathbb{R}^d$ and $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$ by

$$R_\gamma(x, \mathsf{A}) = (4\pi\gamma)^{-d/2} \int_{\mathsf{A}} \exp\left(-\left\|y - x - \gamma\nabla U(x)\right\|^2/(4\gamma)\right) \mathrm{d}y \ . \tag{12}$$

Proposition 2 *Assume* $\boldsymbol{A}\,1(m)$ *for* $m \geq 0$ *and* $\boldsymbol{A}\,2$. *For all* $\gamma \in (0, L^{-1}]$ *and* $\mu \in \mathcal{P}_2(\mathbb{R}^d)$, *we have*

$$2\gamma\left\{\mathscr{F}(\mu R_\gamma) - \mathscr{F}(\pi)\right\} \leq (1 - m\gamma)W_2^2(\mu, \pi) - W_2^2(\mu R_\gamma, \pi) + 2\gamma^2 L d \ , \tag{13}$$

where $\mathscr{F}$ *is defined in* (7).

The main difficulty in establishing Proposition 2 is to deal with the entropy function $\mathscr{H}$ defined by (8) in $\mathscr{F}$. To obtain the desired result, we decompose R_γ for all $\gamma > 0$ in the product of two elementary kernels S_γ and T_γ given for all $x \in \mathbb{R}^d$ and $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$ by

$$S_\gamma(x, \mathsf{A}) = \delta_{x-\gamma\nabla U(x)}(\mathsf{A}) \ , \ T_\gamma(x, \mathsf{A}) = (4\pi\gamma)^{-d/2} \int_{\mathsf{A}} \exp\left(-\left\|y - x\right\|^2/(4\gamma)\right) \mathrm{d}y \ . \tag{14}$$

We take the convention that $S_0 = T_0 = \mathrm{Id}$ is the identity kernel given for all $x \in \mathbb{R}^d$ by $\mathrm{Id}(x, \{x\}) = 1$. S_γ is the deterministic part of the Euler-Maruyama discretization, which corresponds to gradient descent step relative to U for the $\mathscr{E}$ functional, whereas T_γ is the random part, that corresponds to going along the gradient flow of $\mathscr{H}$. Note then $R_\gamma = S_\gamma T_\gamma$ and consider the following decomposition

$$\mathscr{F}(\mu R_\gamma) - \mathscr{F}(\pi) = \mathscr{E}(\mu R_\gamma) - \mathscr{E}(\mu S_\gamma) + \mathscr{E}(\mu S_\gamma) - \mathscr{E}(\pi) + \mathscr{H}(\mu R_\gamma) - \mathscr{H}(\pi) \ . \tag{15}$$

The proof of Proposition 2 then consists in bounding each difference in the decomposition above. This is the matter of the following Lemmas. While the proofs of the bounds for the first two terms are quite elementary, the one for the final term uses results from gradient flow theory which are summarized in Section A. It is worthwhile to observe that we do not apply this theory to the Langevin semi-group but only to the heat semi-group.

Lemma 3 *Assume* $\boldsymbol{A}\,2$. *For all* $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ *and* $\gamma > 0$,

$$\mathscr{E}(\mu T_\gamma) - \mathscr{E}(\mu) \leq L d \gamma \ .$$

Proof First note that by (Nesterov, 2004, Lemma 1.2.3), for all $x, \tilde{x} \in \mathbb{R}^d$, we have

$$\left|U(\tilde{x}) - U(x) - \langle\nabla U(x), \tilde{x} - x\rangle\right| \leq (L/2)\left\|\tilde{x} - x\right\|^2 \ . \tag{16}$$

Therefore, for all $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ and $\gamma > 0$, we get

$$\begin{aligned}\mathscr{E}(\mu T_\gamma) - \mathscr{E}(\mu) &= (4\pi\gamma)^{-d/2} \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \left\{U(x + y) - U(x)\right\} \mathrm{e}^{-\|y\|^2/(4\gamma)} \mathrm{d}y\mathrm{d}\mu(x) \\ &\leq (4\pi\gamma)^{-d/2} \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \left\{\langle\nabla U(x), y\rangle + (L/2)\left\|y\right\|^2\right\} \mathrm{e}^{-\|y\|^2/(4\gamma)} \mathrm{d}y\mathrm{d}\mu(x) \ ,\end{aligned}$$

which concludes the proof. ∎

Lemma 4 *Assume* $\boldsymbol{A}\,1(m)$ *for* $m \geq 0$ *and* $\boldsymbol{A}\,2$. *For all* $\gamma \in (0, L^{-1}]$ *and* $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$,

$$2\gamma\left\{\mathscr{E}(\mu S_\gamma) - \mathscr{E}(\nu)\right\} \leq (1 - m\gamma)W_2^2(\mu, \nu) - W_2^2(\mu S_\gamma, \nu) - \gamma^2(1 - \gamma L)\int_{\mathbb{R}^d} \left\|\nabla U(x)\right\|^2 \mathrm{d}\mu(x) \ ,$$

where $\mathscr{E}$ *and* T_γ *are defined in* (9) *and* (14) *respectively.*

Proof Using (16) and $\mathbf{A}\,1(m)$, for all $x, y \in \mathbb{R}^d$, we get

$$\begin{aligned}U(x - \gamma\nabla U(x)) - U(y) &= U(x - \gamma\nabla U(x)) - U(x) + U(x) - U(y) \\ &\leq -\gamma(1 - \gamma L/2)\left\|\nabla U(x)\right\|^2 + \langle\nabla U(x), x - y\rangle - (m/2)\left\|y - x\right\|^2 \ .\end{aligned}$$

Multiplying both sides by 2γ we obtain:

$$\begin{aligned}2\gamma\left\{U(x - \gamma\nabla U(x)) - U(y)\right\} \leq (1 - m\gamma)\left\|x - y\right\|^2 - \left\|x - \gamma\nabla U(x) - y\right\|^2 \\ - \gamma^2(1 - \gamma L)\left\|\nabla U(x)\right\|^2 \ .\end{aligned} \tag{17}$$

Let now (X, Y) be an optimal coupling between μ and ν. Then by definition and (17), we get

$$\begin{aligned}2\gamma\left\{\mathscr{E}(\mu S_\gamma) - \mathscr{E}(\nu)\right\} \leq (1 - m\gamma)W_2^2(\mu, \nu) - \mathbb{E}\left[\left\|X - \gamma\nabla U(X) - Y\right\|^2\right] \\ - \gamma^2(1 - \gamma L)\mathbb{E}\left[\left\|\nabla U(X)\right\|^2\right] \ .\end{aligned}$$

Using that $W_2^2(\mu S_\gamma, \nu) \leq \mathbb{E}[\left\|X - \gamma\nabla U(X) - Y\right\|^2]$ concludes the proof. ∎

Lemma 5 *Let* $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$, $\mathscr{H}(\nu) < \infty$. *Then for all* $\gamma > 0$,

$$2\gamma\left\{\mathscr{H}(\mu T_\gamma) - \mathscr{H}(\nu)\right\} \leq W_2^2(\mu, \nu) - W_2^2(\mu T_\gamma, \nu) \ ,$$

where T_γ *is given in* (14).

Proof Denote for all $t \geq 0$ by $\mu_t = \mu T_t$. Since $(T_t)_{t\geq 0}$ is the Markov semi-group associated with the Brownian motion, then $(\mu_t)_{t\geq 0}$ is the solution (in the sense of distribution) of the Fokker-Plank equation:

$$\frac{\partial\mu_t}{\partial t} = \Delta\mu_t \ ,$$

and μ_t goes to μ as t goes to 0 in $(\mathcal{P}_2(\mathbb{R}^d), W_2)$. Let $\nu \in \mathcal{P}_2(\mathbb{R}^d)$ and $\gamma > 0$. Then by Theorem 31, for all $\epsilon \in (0, \gamma)$, there exists $(\delta_t) \in \mathrm{L}^1((\epsilon, \gamma))$ such that

$$W_2^2(\mu_\gamma, \nu) - W_2^2(\mu_\epsilon, \nu) = \int_\epsilon^\gamma \delta_s \mathrm{d}s \tag{18}$$

$$\delta_s/2 \leq \mathscr{H}(\nu) - \mathscr{H}(\mu_s) \ , \ \text{for almost all } s \in (\epsilon, \gamma) \ . \tag{19}$$

In addition by (Villani, 2009, Particular case 24.3), $s \mapsto \mathscr{H}(\mu_s)$ is non-increasing on $\mathbb{R}_+^*$ and therefore (19) becomes

$$\delta_s/2 \leq \mathscr{H}(\nu) - \mathscr{H}(\mu_\gamma) \ , \ \text{for almost all } s \in (\epsilon, \gamma) \ .$$

Plugging this bound in (18) yields that for all $\epsilon \in \mathbb{R}_+^*$,

$$W_2^2(\mu_t, \nu) - W_2^2(\mu_\epsilon, \nu) \leq 2(\gamma - \epsilon) \left\{ \mathscr{H}(\nu) - \mathscr{H}(\mu_\gamma) \right\} \ .$$

Taking $\epsilon \to 0$ concludes the proof. ■

We now have all the tools to prove Proposition 2.

Proof [Proof of Proposition 2] Let $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ and $\gamma \in \mathbb{R}_+^*$. By Lemma 3, we get

$$\mathscr{E}(\mu R_\gamma) - \mathscr{E}(\mu S_\gamma) = \mathscr{E}(\mu S_\gamma T_\gamma) - \mathscr{E}(\mu S_\gamma) \leq Ld\gamma \ .$$

By Lemma 4 since $\pi \in \mathcal{P}_2(\mathbb{R}^d)$ (see Lemma 1-a)),

$$2\gamma \left\{ \mathscr{E}(\mu S_\gamma) - \mathscr{E}(\pi) \right\} \leq (1 - m\gamma) W_2^2(\mu, \nu) - W_2^2(\mu S_\gamma, \nu) \ .$$

By Lemma 5 and Lemma 1-a),

$$\begin{aligned} 2\gamma \left\{ \mathscr{H}(\mu R_\gamma) - \mathscr{H}(\pi) \right\} &= 2\gamma \left\{ \mathscr{H}((\mu S_\gamma) T_\gamma) - \mathscr{H}(\pi) \right\} \\ &\leq W_2^2(\mu S_\gamma, \pi) - W_2^2(\mu R_\gamma, \pi) \ . \end{aligned}$$

Plugging these bounds in (15) concludes the proof. ■

Based on inequalities of the form (11) and using the convexity of f, for all $n \in \mathbb{N}$, non-asymptotic bounds (in expectation) between $f(\bar{x}_n)$ and $f(x_f)$ can be derived, where $(\bar{x}_k)_{k \in \mathbb{N}}$ is the sequence of averages of $(x_k)_{k \in \mathbb{N}}$ given for all $n \in \mathbb{N}$ by $\bar{x}_n = n^{-1} \sum_{k=1}^n x_k$. Besides, if f is assumed to be strongly convex, a bound on $\mathbb{E}[\|x_n - x_f\|^2]$ can be established. We will adapt this methodology to get some bounds on the convergence of sequences of averaged measures defined as follows. Let $(\gamma_k)_{k \in \mathbb{N}^*}$ and $(\lambda_k)_{k \in \mathbb{N}^*}$ be two non-increasing sequences of reals numbers referred to as the sequence of stepsizes and weights respectively. Define for all $n, N \in \mathbb{N}$, $n \geq 1$,

$$\Gamma_{N,N+n} = \sum_{k=N+1}^{N+n} \gamma_k \ , \qquad \Lambda_{N,N+n} = \sum_{k=N+1}^{N+n} \lambda_k \ . \tag{20}$$

Let $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$ be an initial distribution. The sequence of probability measures $(\nu_n^N)_{n \in \mathbb{N}^*}$ is defined for all $n, N \in \mathbb{N}$, $n \geq 1$, by

$$\nu_n^N = \Lambda_{N,N+n}^{-1} \sum_{k=N+1}^{N+n} \lambda_k \, \mu_0 Q_\gamma^k \ , \qquad Q_\gamma^k = R_{\gamma_1} \cdots R_{\gamma_k} \ , \ \text{for } k \in \mathbb{N}^* \ , \tag{21}$$

where R_γ is defined by (12) and N is a burn-in time. We take in the following, the convention that Q_γ^0 is the identity operator.

Contrary to most works on ULA, we state our next results in terms of average measures ν_n^N, defined by (21) for $N \in \mathbb{N}$ and $n \in \mathbb{N}^*$ instead of the final iterates $\mu_0 Q_\gamma^n$. Indeed, in the case where $m = 0$, Proposition 2 does not imply very informative bounds for Q_γ^n. However using that $\mathrm{KL}(\cdot|\pi)$ is convex and applying Proposition 2 allow to use averaging trick from optimization to obtain useful bounds on $\mathrm{KL}(\nu_n^N|\pi)$.

Theorem 6 *Assume* ***A*** *1(m) for $m \geq 0$ and* ***A*** *2. Let $(\gamma_k)_{k \in \mathbb{N}^*}$ and $(\lambda_k)_{k \in \mathbb{N}^*}$ be two non-increasing sequences of positive real numbers satisfying $\gamma_1 \leq L^{-1}$, and for all $k \in \mathbb{N}^*$, $\lambda_{k+1}(1 - m\gamma_{k+1})/\gamma_{k+1} \leq \lambda_k/\gamma_k$. Let $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$ and $N \in \mathbb{N}$. Then for all $n \in \mathbb{N}^*$, it holds:*

$$\begin{aligned} &\mathrm{KL}\left(\nu_n^N|\pi\right) + \lambda_{N+n} W_2^2\left(\mu_0 Q_\gamma^{N+n}, \pi\right) / (2\gamma_{N+n} \Lambda_{N,N+n}) \\ &\quad \leq \lambda_{N+1}(1 - m\gamma_{N+1}) W_2^2\left(\mu_0 Q_\gamma^N, \pi\right) / (2\gamma_{N+1} \Lambda_{N,N+n}) + (Ld/\Lambda_{N,N+n}) \sum_{k=N+1}^{N+n} \gamma_k \lambda_k \ , \end{aligned}$$

where ν_n^N and Q_γ^N are defined in (21).

Proof Using the convexity of KL divergence (see (Cover and Thomas, 2006, Theorem 2.7.2) or (van Erven and Harremos, 2014, Theorem 11)) and Proposition 2, we obtain

$$\begin{aligned} \mathrm{KL}\left(\nu_n^N|\pi\right) &\leq \Lambda_{N,N+n}^{-1} \sum_{k=N+1}^{N+n} \lambda_k \, \mathrm{KL}\left(\mu_0 Q_\gamma^k \middle| \pi\right) \\ &\leq (2\Lambda_{N,N+n})^{-1} \left[\frac{(1 - m\gamma_{N+1})\lambda_{N+1}}{\gamma_{N+1}} W_2^2\left(\mu_0 Q_\gamma^N, \pi\right) - \frac{\lambda_{N+n}}{\gamma_{N+n}} W_2^2\left(\mu_0 Q_\gamma^{N+n}, \pi\right) \right. \\ &\quad \left. + \sum_{k=N+1}^{N+n-1} \left\{ \frac{(1 - m\gamma_{k+1})\lambda_{k+1}}{\gamma_{k+1}} - \frac{\lambda_k}{\gamma_k} \right\} W_2^2\left(\mu_0 Q_\gamma^k, \pi\right) + \sum_{k=N+1}^{N+n} Ld\lambda_k\gamma_k \right] \ . \end{aligned}$$

We get the thesis using that $\lambda_{k+1}(1 - m\gamma_{k+1})/\gamma_{k+1} \leq \lambda_k/\gamma_k$ for all $k \in \mathbb{N}^*$. ■

Corollary 7 *Assume* ***A*** *1(0) and* ***A*** *2. Let $\varepsilon > 0$ and $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$. Let*

$$\gamma_\varepsilon \leq \min\left\{\varepsilon/(2Ld), L^{-1}\right\} \ , \qquad n_\varepsilon \geq \lceil W_2^2(\mu_0, \pi) \gamma_\varepsilon^{-1} \varepsilon^{-1} \rceil \ .$$

Then it holds $\mathrm{KL}\left(\nu_{n_\varepsilon}|\pi\right) \leq \varepsilon$ where $\nu_{n_\varepsilon} = n_\varepsilon^{-1} \sum_{k=1}^{n_\varepsilon} \mu_0 R_{\gamma_\varepsilon}^k$.

Proof We apply Theorem 6 with $\gamma_k = \gamma_\varepsilon$ and $\lambda_k = 1$ for all $k \geq 1$. We obtain

$$\mathrm{KL}\left(\nu_{n_\varepsilon}|\pi\right) + W_2^2\left(\mu_0 Q_\gamma^{n_\varepsilon}, \pi\right)/(2\gamma_\varepsilon n_\varepsilon) \leq W_2^2\left(\mu_0, \pi\right)/(2\gamma_\varepsilon n_\varepsilon) + (Ld/n_\varepsilon)\sum_{k=1}^{n_\varepsilon} \gamma_\varepsilon \ ,$$

and the proof is concluded by a straightforward calculation using the definition of γ_ε and n_ε. ■

Corollary 8 *Assume* ***A*** *1(m) for $m \geq 0$ and* ***A*** *2. Let $\alpha \in (0,1)$. Define $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ for all $k \in \mathbb{N}^*$ by $\gamma_k = \gamma_1/k^\alpha$, $\lambda_k = \gamma_1/(k+1)^\alpha$, $\gamma_1 \in \left(0, L^{-1}\right)$. Then, there exists $C \geq 0$ such that for all $n \in \mathbb{N}^*$ we have $\mathrm{KL}\left(\nu_n^0|\pi\right) \leq C\max(n^{\alpha-1}, n^{-\alpha})$, if $\alpha \neq 1/2$, and for $\alpha = 1/2$, we have $\mathrm{KL}\left(\nu_n^0|\pi\right) \leq C(\ln(n)+1)n^{-1/2}$, where ν_n^0 is defined by* (21).

Proof The proof is postponed to Section 7.2. ■

In the case where a warm start is available for the Wasserstein distance, *i.e.* $W_2^2(\mu_0, \pi) \leq C$, for some absolute constant $C \geq 0$, then Corollary 7 implies that the complexity of ULA to obtain a sample close from π in KL with a precision $\varepsilon > 0$ is of order $d\bar{\mathcal{O}}(\varepsilon^{-2})$. In addition, by Pinsker inequality, we have for all probability measure μ on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$, $\|\mu - \pi\|_{\mathrm{TV}} \leq \{2\,\mathrm{KL}(\mu|\pi)\}^{1/2}$, which implies that the complexity of ULA for the total variation distance is of order $d\bar{\mathcal{O}}(\varepsilon^{-4})$.

In addition if we have access to $\eta > 0$ and $M_\eta \geq 0$, independent of the dimension, such that for all $x \in \mathbb{R}^d$, $x \notin \mathrm{B}(x^\star, M_\eta)$, $U(x) - U(x^\star) \geq \eta\|x - x^\star\|$, $x^\star \in \arg\min_{\mathbb{R}^d} U$, Proposition 32 in Appendix B shows that for all $\int_{\mathbb{R}^d}\|x - x^\star\|^2 \,\mathrm{d}\pi(x) \leq 2\eta^{-2}d(1+d) + M_\eta^2$. Therefore, starting at $\delta_{x^\star}$, the overall complexity for the KL is in this case $d^3\bar{\mathcal{O}}(\varepsilon^{-2})$ and $d^3\bar{\mathcal{O}}(\varepsilon^{-4})$ for the total variation distance. This discussion justifies the bound we state in Table 1.

In the case where $m > 0$, based on Proposition 2, we can directly get a bound on the Wasserstein distance between the final iterate of ULA and π.

Theorem 9 *Assume* ***A*** *1(m) for $m > 0$ and* ***A*** *2. Let $(\gamma_k)_{k\in\mathbb{N}^*}$ be a non-increasing sequence of positive real numbers, $\gamma_1 \in \left(0, L^{-1}\right]$, and $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$. Then for all $n \in \mathbb{N}^*$, it holds*

$$W_2^2\left(\mu_0 Q_\gamma^n, \pi\right) \leq \left\{\prod_{k=1}^n (1 - m\gamma_k)\right\} W_2^2\left(\mu_0, \pi\right) + 2Ld\sum_{k=1}^n \gamma_k^2 \prod_{i=k+1}^n (1 - m\gamma_i) \ ,$$

where Q_γ^n is defined in (21).

Proof Using Proposition 2 and since the KL divergence is non-negative, we get for all $k \in \{1, \ldots, n\}$,

$$W_2^2\left(\mu_0 Q_\gamma^k, \pi\right) \leq (1 - m\gamma_k) W_2^2\left(\mu_0 Q_\gamma^{k-1}, \pi\right) + 2Ld\gamma_k^2 \ .$$

The proof then follows from a direct induction. ■

Corollary 10 *Assume* ***A*** *1(m) for $m > 0$ and* ***A*** *2. Let $\varepsilon > 0$ and $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$. Define:*

$$\gamma_\varepsilon \leq \min\left\{m\varepsilon/(4Ld), L^{-1}\right\} \ , \qquad n_\varepsilon \geq \lceil \ln(2W_2^2(\mu_0, \pi)/\varepsilon)\gamma_\varepsilon^{-1}m^{-1}\rceil \ .$$

Then we have $W_2^2\left(\mu_0 R_{\gamma_\varepsilon}^{n_\varepsilon}, \pi\right) \leq \varepsilon$, where R_{γ_ε} is defined by (12).

Proof By Theorem 9, we have

$$W_2^2\left(\mu_0 Q_\gamma^{n_\varepsilon}, \pi\right) \leq (1 - m\gamma_\varepsilon)^{n_\varepsilon} W_2^2\left(\mu_0, \pi\right) + 2Ld\sum_{k=1}^{n_\varepsilon} \gamma_\varepsilon^2 (1 - m\gamma_\varepsilon)^{n_\varepsilon - k} \ .$$

On one hand, by definition of γ_ε, we get $2Ld\sum_{k=1}^{n_\varepsilon}\gamma_\varepsilon^2(1-m\gamma_\varepsilon)^{n_\varepsilon-k} \leq 2Ld\gamma_\varepsilon/m \leq \varepsilon/2$. On the other hand, using that for all $t \in \mathbb{R}_+$, $1 - t \leq \exp(-t)$ and the definition of n_ε, we obtain $(1-m\gamma_\varepsilon)^{n_\varepsilon} W_2^2\left(\mu_0, \pi\right) \leq \exp(-m\gamma_\varepsilon n_\varepsilon) W_2^2\left(\mu_0, \pi\right) \leq \varepsilon/2$. Then the thesis of the corollary follows directly from the above inequalities. ■

Note that the bound in the right hand side of Theorem 9 is tighter than the previous bound given in Dalalyan and Karagulyan (2017, Theorem 1) (for constant stepsize) and in Durmus and Moulines (2016, Theorem 5) (for both constant and non-increasing stepsizes). Indeed Dalalyan and Karagulyan (2017, Theorem 1) shows that, in the constant stepsize setting $\gamma_k = \gamma$, for all $k \in \mathbb{N}$,

$$W_2(\mu_0 Q_\gamma^k, \pi) \leq (1 - m\gamma)^k W_2(\mu_0, \pi) + 1.65(L/m)(\gamma d)^{1/2} \ .$$

On the other hand, the inequality $(t+s)^{1/2} \leq t^{1/2} + s^{1/2}$ for $t, s \geq 0$ and Theorem 9 imply that for all $k \in \mathbb{N}$,

$$W_2(\mu_0 Q_\gamma^k, \pi) \leq (1 - m\gamma)^{k/2} W_2(\mu_0, \pi) + \{2\gamma dL/m\}^{1/2} \ . \tag{22}$$

Thus, the dependency on the condition number L/m is improved. This bound is in agreement for the case where π is the zero-mean d-dimensional Gaussian distribution with covariance matrix Σ. In that case, all the iterates $(X_k)_{k\in\mathbb{N}^*}$ defined by (2) for $\gamma > 0$, starting from $x \in \mathbb{R}^d$, follows the Gaussian distribution with mean $(\mathrm{Id} - \gamma\Sigma)^k x$ and covariance matrix $2\gamma\sum_{i=0}^{k-1}(1-\gamma\Sigma)^{2i}$. Since the Wasserstein distance between d-dimensional Gaussian distributions can be explicitly computed, see Givens and Shortt (1984), denoting by L and m the largest and smallest eigenvalues of Σ respectively, we have by an explicit calculation for $\gamma \in \left(0, L^{-1}\right]$,

$$W_2(\mu_0 Q_\gamma^k, \pi) \leq (1 - m\gamma)^k W_2(\mu_0, \pi) + (d/m)^{1/2}\left\{(1 - \gamma L/2)^{-1/2} - 1\right\} \ .$$

Since for $t \in [0, 1/2]$, $(1-t)^{-1/2} - 1 - t \leq 0$, we get

$$W_2(\mu_0 Q_\gamma^k, \pi) \leq (1 - m\gamma)^k W_2(\mu_0, \pi) + 2^{-1}\gamma(d/m)^{1/2}\left\{(1 - \gamma L)^{-1/2} - 1\right\} \ .$$

Using that $\gamma \leq L^{-1}$, we get that the second term in the right hand side is bounded by $(dL\gamma/m)^{1/2}$, which is precisely the order we get from (22).

Finally, if $(\gamma_k)_{k\in\mathbb{N}^*}$ is given for all $k \in \mathbb{N}^*$, by $\gamma_k = \gamma_1/k^\alpha$, for $\alpha \in (0,1)$, then using (Durmus and Moulines, 2016, Lemma 7) and the same calculation of (Durmus and Moulines, 2015, Section 6.1), we get that there exists $C \geq 0$ such that for all $n \in \mathbb{N}^*$, $W_2(\mu_0 Q_\gamma^n, \pi) \leq Cn^{-\alpha/2}$.

Based on Theorem 9, we can improve Corollary 7 in the case where U is strongly convex using an appropriate burn-in time.

Corollary 11 *Assume* **A** *1(m) for $m > 0$ and* **A** *2. Let $\varepsilon > 0$, $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$ and*

$$\gamma_\varepsilon \leq \min\left\{m\varepsilon/(4Ld), L^{-1}\right\} , \qquad \tilde{\gamma}_\varepsilon \leq \min\left\{\varepsilon/2Ld, L^{-1}\right\} ,$$
$$N_\varepsilon \geq \lceil \ln(2W_2^2(\mu_0,\pi)/\varepsilon)(\gamma_\varepsilon m)^{-1}\rceil , \qquad n_\varepsilon \geq \lceil \tilde{\gamma}_\varepsilon^{-1}\rceil .$$

Let $(\gamma_k)_{k\in\mathbb{N}}$ defined by $\gamma_k = \gamma_\varepsilon$ for $k \in \{1,\dots,N_\varepsilon\}$ and $\gamma_k = \tilde{\gamma}_\varepsilon$ for $k > N_\varepsilon$. Then we have $\mathrm{KL}\left(\nu_{n_\varepsilon}^{N_\varepsilon}|\pi\right) \leq \varepsilon$ where $\nu_{n_\varepsilon}^{N_\varepsilon} = n_\varepsilon^{-1}\sum_{k=1}^{n_\varepsilon}\mu_0 R_{\gamma_\varepsilon}^{N_\varepsilon} R_{\tilde{\gamma}_\varepsilon}^k$.

Proof Using Corollary 10, we have $W_2^2\left(\mu_0 Q_\gamma^{N_\varepsilon}, \pi\right) \leq \varepsilon$. Now applying Theorem 6 we get:

$$\mathrm{KL}\left(\nu_{n_\varepsilon}^{N_\varepsilon}|\pi\right) \leq W_2^2\left(\mu_{N_\varepsilon},\pi\right)/(2\tilde{\gamma}_\varepsilon n_\varepsilon) + (Ld/n_\varepsilon\tilde{\gamma}_\varepsilon)\sum_{k=N_\varepsilon+1}^{N_\varepsilon+n_\varepsilon}(\tilde{\gamma}_\varepsilon)^2 \leq \varepsilon/(2\tilde{\gamma}_\varepsilon n_\varepsilon) + Ld\tilde{\gamma}_\varepsilon \leq \varepsilon$$

■

By (Durmus and Moulines, 2016, Proposition 1), we have $\int_{\mathbb{R}^d}\|x - x^\star\|^2 \mathrm{d}\pi(x) \leq d/m$, where $x^\star = \arg\min_{\mathbb{R}^d} U$. Therefore we have that in the constant stepsize setting, $\gamma_k = \gamma \in (0, L^{-1}]$ for all $k \in \mathbb{N}^*$, Corollary 10 implies that the sufficient number of iterations to have $W_2(\delta_{x^\star}Q_\gamma^n, \pi) \leq \varepsilon$ is of order $d\bar{\mathcal{O}}(\varepsilon^{-2})$. Then Corollary 11 implies that the sufficient number of iterations to get $\mathrm{KL}\left(\nu_n^N|\pi\right) \leq \varepsilon$, for $\varepsilon > 0$, is of order $d\bar{\mathcal{O}}(\varepsilon^{-1})$. By Pinsker inequality, we obtain that a sufficient number of iterations to get $\|\nu_n^N - \pi\|_{\mathrm{TV}} \leq \varepsilon$, for $\varepsilon > 0$, is of order $d\bar{\mathcal{O}}(\varepsilon^{-2})$.

For a sufficiently small constant stepsize γ, ULA produces a Markov Chain with a stationary measure π_γ. In general this measure is different from the measure of interest π. Based on our previous results, we establish computable bounds on the distance between π and π_γ.

Theorem 12 *Assume* **A** *1(m) for $m \geq 0$ and* **A** *2. Let $\gamma \in \left(0, L^{-1}\right]$. Then there exists a measure π_γ, such that $\pi_\gamma R_\gamma = \pi_\gamma$ where R_γ is defined by* (12)*. In addition, we have*

$$\mathrm{KL}\left(\pi_\gamma|\pi\right) \leq Ld\gamma, \qquad \|\pi_\gamma - \pi\|_{\mathrm{TV}} \leq \sqrt{2Ld\gamma}$$

Furthermore, if $m > 0$ we also have $W_2^2\left(\pi_\gamma,\pi\right) \leq 2Ld\gamma/m$.

Proof Under **A** 1 and **A** 2, (Durmus and Moulines, 2017, Proposition 13) shows that R_γ satisfies a geometric Foster-Lyapunov drift condition for $\gamma \leq L^{-1}$. In addition, it is easy to see that R_γ is Leb-irreducible and weak Feller and therefore by (Meyn and Tweedie, 2009, Theorem 6.0.1 together with Theorem 5.5.7), all compact sets are small. Then, by (Meyn and Tweedie, 2009, Theorem 16.0.1), R_γ has a unique invariant distribution π_γ.

Second, taking $\mu = \pi_\gamma$ in Proposition 2 we obtain:

$$2\gamma\,\mathrm{KL}\left(\pi_\gamma R_\gamma|\pi\right) \leq (1-m\gamma)W_2^2(\pi_\gamma,\pi) - W_2^2(\pi_\gamma R_\gamma,\pi) + 2\gamma^2 Ld\,, \tag{23}$$

and because $\pi_\gamma R_\gamma = \pi_\gamma$, the above implies $2\,\mathrm{KL}\left(\pi_\gamma|\pi\right) + mW_2^2(\pi_\gamma,\pi) \leq 2Ld\gamma$. Since both the KL divergence and Wasserstein distance are positive, the desired bounds in KL and W_2^2 follow. The bound in total variation follows from the bound in KL-divergence and Pinsker inequality. ■

4. Extensions of ULA

In this section, two extensions of ULA are presented and analyzed. These two algorithms can be applied to non-continuously differentiable convex potential $U : \mathbb{R}^d \to \mathbb{R}$ and therefore **A** 2 is not assumed anymore. In addition, for the two new algorithms we present, only i.i.d. unbiased estimates of (sub)gradients of U are necessary as in Stochastic Gradient Langevin Dynamics (SGLD) (Welling and Teh, 2011). The main difference in these two approaches is that one relies on the subgradient of U while the other is based on proximal operators which are tools commonly used in non-smooth optimization. However, the theoretical results that we can show for these two algorithms, hold for different sets of conditions.

4.1. Stochastic SubGradient Langevin Dynamics

Note that if U is convex and l.s.c then for any point $x \in \mathbb{R}^d$, its subdifferential $\partial U(x)$ defined by

$$\partial U(x) = \left\{v \in \mathbb{R}^d \,:\, U(y) \geq U(x) + \langle v, y - x\rangle \text{ for all } y \in \mathbb{R}^d\right\} , \tag{24}$$

is non empty, see (Rockafellar and Wets, 1998, Proposition 8.12, Theorem 8.13). For all $x \in \mathbb{R}^d$, any elements of $\partial U(x)$ is referred to as a subgradient of U at x. Consider the following condition on U which assumes that we have access to unbiased estimates of subgradients of U at any point $x \in \mathbb{R}^d$.

A 3 (i) The potential U is M-Lipschitz, *i.e.* for all $x, y \in \mathbb{R}^d$, $|U(x) - U(y)| \leq M\|x - y\|$.

(ii) There exists a measurable space $(\mathsf{Z}, \mathcal{Z})$, a probability measure η on $(\mathsf{Z}, \mathcal{Z})$ and a measurable function $\Theta : \mathbb{R}^d \times \mathsf{Z} \to \mathbb{R}^d$ for all $x \in \mathbb{R}^d$,

$$\int_{\mathsf{Z}} \Theta(x,z)\mathrm{d}\eta(z) \in \partial U(x) .$$

Note that under **A** 3-(i), for all $x \in \mathbb{R}^d$ and $v \in \partial U(x)$,

$$\|v\| \leq M . \tag{25}$$

Assumption **A**3 is satisfied for example in the case where $U = U_1 + U_2$, U_1 is L-gradient Lipschitz and Lipschitz and U_2 is non-smooth but Lipschitz, if there exists a measurable $\tilde{\Theta} : \mathbb{R}^d \times \mathsf{Z} \to \mathbb{R}^d$ such that $\int_{\mathsf{Z}} \tilde{\Theta}(x,z)\mathrm{d}\eta(z) = \nabla U_1(x)$ for any $x \in \mathbb{R}^d$. Then by (Bauschke

and Combettes, 2011, Corollary 16.38), unbiased and i.i.d. estimates of ∇f can be computed setting $\Theta = \partial g + \tilde{\Theta}$.

Let $(Z_k)_{k\in\mathbb{N}^*}$ be a sequence of i.i.d. random variables distributed according to η, $(\gamma_k)_{k\in\mathbb{N}^*}$ be a sequence of non-increasing stepsizes and $\bar{X}_0$ distributed according to $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$. Stochastic SubGradient Langevin Dynamics (SSGLD) defines the sequence of random variables $(\bar{X}_k)_{k\in\mathbb{N}}$ starting at $\bar{X}_0$ for $n \geq 0$ by

$$\bar{X}_{n+1} = \bar{X}_n - \gamma_{n+1}\Theta(\bar{X}_n, Z_{n+1}) + \sqrt{2\gamma_{n+2}}G_{n+1} \;, \tag{26}$$

where $(G_k)_{k\in\mathbb{N}^*}$ is a sequence of i.i.d. d-dimensional standard Gaussian random variables, independent of $(Z_k)_{k\in\mathbb{N}^*}$, see Algorithm 1. Consequently this method defines a new sequence of Markov kernels $(\bar{R}_{\gamma_k,\gamma_{k+1}})_{k\in\mathbb{N}^*}$ given for all $\gamma, \tilde{\gamma} > 0$, $x \in \mathbb{R}^d$ and $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$ by

$$\bar{R}_{\gamma,\tilde{\gamma}}(x, \mathsf{A}) = (4\pi\tilde{\gamma})^{-d/2} \int_{\mathsf{A}\times\mathsf{Z}} \exp\left(-\left\|y - x + \gamma\Theta(x,z)\right\|^2/(4\tilde{\gamma})\right) \mathrm{d}\eta(z)\mathrm{d}y \;. \tag{27}$$

Algorithm 1: SSGLD

Data: initial distribution $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$, non-increasing sequence $(\gamma_k)_{k\geq 1}$, U, Θ, η satisfying **A**3
Result: $(\bar{X}_k)_{k\in\mathbb{N}}$
begin
  Draw $\bar{X}_0 \sim \mu_0$;
  for $k \geq 0$ **do**
    Draw $G_{k+1} \sim \mathcal{N}(0, \mathrm{Id})$ and $Z_{k+1} \sim \eta$;
    Set $\bar{X}_{k+1} = \bar{X}_k - \gamma_{k+1}\Theta(\bar{X}_k, Z_{k+1}) + \sqrt{2\gamma_{k+2}}G_{k+1}$

Let $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ be two non-increasing sequences of reals numbers and $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$ be an initial distribution. The weighted averaged distribution associated with (26) $(\bar{\nu}_n^N)_{n\in\mathbb{N}}$ is defined for all $N, n \in \mathbb{N}$, $n \geq 1$ by

$$\bar{\nu}_n^N = \Lambda_{N,N+n}^{-1} \sum_{k=N+1}^{N+n} \lambda_k \, \mu_0 \bar{Q}_\gamma^k \;, \qquad \bar{Q}_\gamma^k = \bar{R}_{\gamma_1,\gamma_2} \cdots \bar{R}_{\gamma_k,\gamma_{k+1}} \;, \text{ for } k \in \mathbb{N}^* \;, \tag{28}$$

where N is a burn-in time and $\Lambda_{N,N+n}$ is defined in (20). We take in the following the convention that $\bar{Q}_\gamma^0$ is the identity operator.

Under **A**3, define for all $\mu \in \mathcal{P}_2(\mathbb{R}^d)$,

$$\upsilon_\Theta(\mu) = \int_{\mathbb{R}^d\times\mathsf{Z}} \left\| \Theta(x,z) - \int_{\mathsf{Z}} \Theta(x,\tilde{z})\mathrm{d}\eta(\tilde{z}) \right\|^2 \mathrm{d}\eta(z)\mathrm{d}\mu(x) = \mathbb{E}\left[\left\| \Theta(\bar{X}_0, Z_1) - v \right\|^2 \right] \;, \tag{29}$$

where $\bar{X}_0, Z_1$ are independent random variables with distribution μ and η respectively and $v \in \partial U(X_0)$ almost surely. In addition, consider $\bar{S}_\gamma$, the Markov kernel on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$ defined for all $x \in \mathbb{R}^d$ and $\mathsf{A} \in \mathcal{B}(\mathbb{R}^d)$ by

$$\bar{S}_\gamma(x, \mathsf{A}) = \int_{\mathsf{Z}} \mathbb{1}_{\mathsf{A}}\left(x - \gamma\Theta(x,z)\right) \mathrm{d}\eta(z) \;. \tag{30}$$

Theorem 13 *Assume* ***A*** *1(0) and* ***A*** *3. Let $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ be two non-increasing sequences of positive real numbers satisfying for all $k \in \mathbb{N}^*$, $\lambda_{k+1}/\gamma_{k+2} \leq \lambda_k/\gamma_{k+1}$. Let $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$ and $N \in \mathbb{N}$. Then for all $n \in \mathbb{N}^*$, it holds*

$$\mathrm{KL}\left(\bar{\nu}_n^N \middle| \pi\right) \leq \lambda_{N+1} W_2^2\left(\mu_0 \bar{Q}_\gamma^N \bar{S}_{\gamma_{N+1}}, \pi\right)/(2\gamma_{N+2}\Lambda_{N,N+n}) + (2\Lambda_{N,N+n})^{-1} \sum_{k=N+1}^{N+n} \left\{ \gamma_{k+1}\lambda_k \left(M^2 + \upsilon_\Theta(\mu_0 \bar{Q}_\gamma^k) \right) \right\} \;,$$

where $\bar{\nu}_n^N$ and $\bar{Q}_\gamma^N$ are defined in (28).

Proof The proof is postponed to Section 7.3.1.. ■

Note that in the bound given by Theorem 13, we need to control the ergodic average of the variance of the stochastic gradient estimates. When **A** 3 is satisfied, a possible assumption is that $x \mapsto \upsilon(\delta_x)$ is uniformly bounded. This assumption will be satisfied for example when the potential U is a sum of Lipschitz continuous functions.

Corollary 14 *Assume* ***A*** *1(0) and* ***A*** *3. Assume that $\sup_{x\in\mathbb{R}^d} \upsilon_\Theta(\delta_x) \leq D^2 < \infty$. Let $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ given for all $k \in \mathbb{N}^*$ by $\lambda_k = \gamma_k = \gamma > 0$. Let $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$. Then for any $N \in \mathbb{N}, n \in \mathbb{N}^*$ we have*

$$\mathrm{KL}\left(\bar{\nu}_n^N \middle| \pi\right) \leq W_2^2\left(\mu_0 \bar{Q}_\gamma^N \bar{S}_\gamma, \pi\right)/(2n\gamma) + (\gamma/2)\left(M^2 + D^2\right) \;.$$

Furthermore, let $\varepsilon > 0$ and

$$\gamma_\varepsilon \leq \varepsilon/(M^2 + D^2) \;, \qquad n_\varepsilon \geq \lceil W_2^2(\mu_0 \bar{S}_\gamma, \pi)(\gamma_\varepsilon \varepsilon)^{-1} \rceil \;.$$

Then for $\gamma = \gamma_\varepsilon$ we have $\mathrm{KL}\left(\bar{\nu}_{n_\varepsilon}^0 \middle| \pi\right) \leq \varepsilon$.

Proof The first inequality is a direct consequence of Theorem 13. The bound for $\mathrm{KL}\left(\bar{\nu}_{n_\varepsilon}^0 \middle| \pi\right)$ follows directly from this inequality and definitions of γ_ε and n_ε. ■

In the case where a warm start is available for the Wasserstein distance, *i.e.* $W_2^2(\mu_0, \pi) \leq C$, for some absolute constant $C \geq 0$, then Corollary 14 implies that the complexity of SSGLD to obtain a sample close from π in KL with a precision target $\varepsilon > 0$ is of order $(M^2 + D^2)\bar{\mathcal{O}}(\varepsilon^{-2})$. Therefore, this complexity bound depends on the dimension only trough M and D^2 contrary to ULA. In addition, Pinsker inequality implies that the complexity of SSGLD for the total variation distance is of order $(M^2 + D^2)\bar{\mathcal{O}}(\varepsilon^{-4})$.

In addition if we have access to $\eta > 0$ and $M_\eta \geq 0$, independent of the dimension, such that for all $x \in \mathbb{R}^d$, $x \notin \mathrm{B}(x^\star, M_\eta)$, $U(x) - U(x^\star) \geq \eta \left\| x - x^\star \right\|$, where $x^\star \in \arg\min_{\mathbb{R}^d} U$, Proposition 32 and **A**3-(i) imply that starting at $\delta_{x^\star}$, the overall complexity of SSGLD for the KL is in this case $(\eta^{-2}d^2 + M_\eta^2)(M^2 + D^2)\bar{\mathcal{O}}(\varepsilon^{-2})$ and $(\eta^{-2}d^2 + M_\eta^2)(M^2 + D^2)\bar{\mathcal{O}}(\varepsilon^{-4})$ for the total variation distance.

If $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ are given for all $k \in \mathbb{N}^*$ by $\gamma_k = \lambda_k = \gamma_1/k^{-\alpha}$, with $\alpha \in (0,1)$, then by the same reasoning as in the proof of Corollary 8, we obtain that there exists $C \geq 0$

such that for all $n \in \mathbb{N}^*$, we have $\mathrm{KL}\left(\bar{\nu}_n^0 | \pi\right) \leq C \max(n^{\alpha-1}, n^{-\alpha})$, if $\alpha \neq 1/2$, and for $\alpha = 1/2$, we have $\mathrm{KL}\left(\bar{\nu}_n^0 | \pi\right) \leq C(\ln(n)+1)n^{-1/2}$.

We can have a better control on the variance terms using the following conditions on Θ.

A4 There exists $\tilde{L} \geq 0$ such that for η-almost every $z \in \mathsf{Z}$, $x \mapsto \Theta(x,z)$ is $1/\tilde{L}$-cocoercive, *i.e.* for all $x \in \mathbb{R}^d$,

$$\langle \Theta(x,z) - \Theta(y,z), x-y \rangle \geq (1/\tilde{L}) \left\| \Theta(x,z) - \Theta(y,z) \right\|^2 .$$

This assumption is for example satisfied if η-almost every z, $x \mapsto \Theta(x,z)$ is the gradient of a continuously differentiable convex function with Lipschitz gradient, see Nesterov (2004, Thereom 2.1.5) and Zhu and Marcotte (1995). Note that in general **A**4 is not implied by **A**1(0) and **A**2. Indeed, **A**4 is a regularity condition on the stochastic (sub)gradient of U, Θ, while **A**1 and **A**2 depend only on U. However, if U is continuously differentiable, Jensen inequality and **A**4 imply that **A**2 is satisfied with L equals $\tilde{L}$.

Proposition 15 *Assume **A**3 and **A**4. Then we have for all $x \in \mathbb{R}^d$ and $\gamma, \tilde{\gamma} > 0$, $\gamma \leq \tilde{L}^{-1}$*

$$2\gamma(\tilde{L}^{-1} - \gamma) \upsilon_\Theta(\delta_x) \leq \|x - x^\star\|^2 - \int_{\mathbb{R}^d} \|y - x^\star\|^2 \bar{R}_{\gamma,\tilde{\gamma}}(x, \mathrm{d}y) + 2\gamma^2 \upsilon_\Theta(\delta_{x^\star}) + 2\tilde{\gamma} d ,$$

where υ_Θ is defined by (29).

Proof Consider $\bar{X}_1 = x - \gamma\Theta(x, Z_1) + \sqrt{2\tilde{\gamma}} G_1$, where Z_1 and G_1 are two independent random variables, Z_1 has distribution η and G_1 is the standard Gaussian random variables. Then using **A**4, we have

$$\begin{aligned}
\mathbb{E}\left[\left\|\bar{X}_1 - x^\star\right\|^2\right] &= \mathbb{E}\left[\|x - \gamma\Theta(x,Z_1) - x^\star\|\right] + 2\tilde{\gamma} d \\
&= \|x - x^\star\|^2 + \mathbb{E}\left[\gamma^2 \|\Theta(x,Z_1)\|^2 - 2\gamma \langle \Theta(x,Z_1), x - x^\star \rangle\right] + 2\tilde{\gamma} d \\
&\leq \|x - x^\star\|^2 - 2\gamma(\tilde{L}^{-1} - \gamma)\mathbb{E}\left[\|\Theta(x,Z_1) - \Theta(x^\star, Z_1)\|^2\right] \\
&\qquad + 2\gamma^2 \mathbb{E}\left[\|\Theta(x^\star, Z_1)\|^2\right] + 2\tilde{\gamma} d .
\end{aligned}$$

The proof is completed upon noting that $\upsilon_\Theta(\delta_x) \leq \mathbb{E}[\|\Theta(x,Z_1) - \Theta(x^\star, Z_1)\|^2]$ and $\upsilon_\Theta(\delta_{x^\star}) = \mathbb{E}\left[\|\Theta(x^\star, Z_1)\|^2\right]$ ■

Combining Theorem 13 and Proposition 15, we get the following result.

Corollary 16 *Assume **A**1(0)-**A**3 and **A**4. Let $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ defined for all $k \in \mathbb{N}^*$ by $\gamma_k = \lambda_k = \gamma \in (0, \tilde{L}^{-1})$. Let $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$. Then for all $N \in \mathbb{N}$ and $n \in \mathbb{N}^*$, we have*

$$\begin{aligned}
&\mathrm{KL}\left(\bar{\nu}_n^N | \pi\right) \leq W_2^2\left(\mu_0 \bar{R}^N_{\gamma,\gamma} \bar{S}_\gamma, \pi\right) \Big/ (2\gamma n) \\
&\quad + \gamma M^2/2 + (2(\tilde{L}^{-1} - \gamma))^{-1} \left\{ (2n)^{-1} \int_{\mathbb{R}^d} \|x - x^\star\|^2 \, \mathrm{d}\mu_0 \bar{R}^{N+1}_{\gamma,\gamma}(x) + \gamma^2 \upsilon_\Theta(\delta_{x^\star}) + \gamma d \right\} .
\end{aligned}$$

Furthermore, let $\varepsilon > 0$ and

$$\gamma_\varepsilon \leq \min\left[\varepsilon \Big/ \left\{2M^2 + 4\tilde{L}d\right\}, \sqrt{\varepsilon\left(4\tilde{L}\upsilon_\Theta(\delta_{x^\star})\right)^{-1}}, (2\tilde{L})^{-1} \right] ,$$

$$n_\varepsilon \geq 2\max\left\{ \left\lceil W_2^2(\mu_0 \bar{S}_{\gamma_\varepsilon}, \pi)(\gamma_\varepsilon \varepsilon)^{-1} \right\rceil, \left\lceil \tilde{L}\varepsilon^{-1} \int_{\mathbb{R}^d} \|x - x^\star\|^2 \, \mathrm{d}\mu_0 \bar{R}_{\gamma_\varepsilon,\gamma_\varepsilon}(x) \right\rceil \right\} .$$

Then for $\gamma = \gamma_\varepsilon$, then we have $\mathrm{KL}\left(\bar{\nu}^0_{n_\varepsilon} | \pi\right) \leq \varepsilon$.

Proof The proof is postponed to Section 7.3.2. ■

Note that compared to Corollary 14, the dependence on the variance of the stochastic subgradients in the bound on n_ε, given in Corollary 16, is less significant since n_ε scales as $(\upsilon_\Theta(\delta_{x^\star}))^{1/2}$ and not as $\sup_{x\in\mathbb{R}^d} \upsilon_\Theta(\delta_x)$. However, the dependency on the dimension deteriorates a little.

4.2. Stochastic Proximal Gradient Langevin Dynamics

In this section, we propose and analyze an other algorithm to handle non-smooth target distribution using stochastic gradient estimates and proximal operators. For $m \geq 0$, consider the following assumptions on the gradient.

A5 (m) There exists $U_1 : \mathbb{R}^d \to \mathbb{R}$ and $U_2 : \mathbb{R}^d \to \mathbb{R}$ such that $U = U_1 + U_2$ and satisfying the following assumptions:

(i) U_1 satisfies **A**1(m) and **A**2. In addition, there exists a measurable space $(\tilde{\mathsf{Z}}, \tilde{\mathcal{Z}})$, a probability measure $\tilde{\eta}_1$ on $(\tilde{\mathsf{Z}}, \tilde{\mathcal{Z}})$ and a measurable function $\tilde{\Theta}_1 : \mathbb{R}^d \times \tilde{\mathsf{Z}} \to \mathbb{R}^d$ such that for all $x \in \mathbb{R}^d$,

$$\int_{\tilde{\mathsf{Z}}} \tilde{\Theta}_1(x, \tilde{z}) \mathrm{d}\tilde{\eta}_1(\tilde{z}) = \nabla U_1(x) .$$

(ii) U_2 satisfies **A**1(0) and is M_2-Lipschitz.

Under **A**5, consider the proximal operator associated with U_2 with parameter $\gamma > 0$ (see Rockafellar and Wets (1998, Chapter 1 Section G)), defined for all $x \in \mathbb{R}^d$ by

$$\mathrm{prox}^\gamma_{U_2}(x) = \operatorname*{arg\,min}_{y\in\mathbb{R}^d} \left\{ U_2(y) + (2\gamma)^{-1} \|x - y\|^2 \right\} .$$

Note that taking the derivative in the right hand side of this equation, we get that for any $x \in \mathbb{R}^d$ and $\gamma > 0$,

$$\mathrm{prox}^\gamma_{U_2}(x) \in x - \gamma \partial U_2(\mathrm{prox}^\gamma_{U_2}(x)) . \tag{31}$$

Let $(\tilde{Z}_k)_{k\in\mathbb{N}^*}$ be a sequence of i.i.d. random variables distributed according to $\tilde{\eta}_1$, $(\gamma_k)_{k\in\mathbb{N}^*}$ be a sequence of non-increasing stepsizes and $\tilde{X}_0$ distributed according to $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$. Stochastic Proximal Gradient Langevin Dynamics (SPGLD) defines the sequence of random variables $(\tilde{X}_n)_{n\in\mathbb{N}}$ starting at $\tilde{X}_0$ for $n \geq 0$ by

$$\tilde{X}_{n+1} = \mathrm{prox}^{\gamma_{n+1}}_{U_2}(\tilde{X}_n) - \gamma_{n+2} \tilde{\Theta}_1\{\mathrm{prox}^{\gamma_{n+1}}_{U_2}(\tilde{X}_n), \tilde{Z}_{n+1}\} + \sqrt{2\gamma_{n+2}} G_{n+1} , \tag{32}$$

where $(G_k)_{k\in\mathbb{N}^*}$ is a sequence of i.i.d. d-dimensional standard Gaussian random variables, independent of $(\tilde{Z}_k)_{k\in\mathbb{N}^*}$. The recursion (32) is associated with the family of Markov kernels $(\tilde{R}_{\gamma_k,\gamma_{k+1}})_{k\in\mathbb{N}^*}$ given for all $\gamma,\tilde{\gamma}>0$, $x\in\mathbb{R}^d$ and $\mathsf{A}\in\mathcal{B}(\mathbb{R}^d)$ by

$$\begin{aligned}&\tilde{R}_{\gamma,\tilde{\gamma}}(x,\mathsf{A})\\&=(4\pi\tilde{\gamma})^{-d/2}\int_{\mathsf{A}\times\tilde{\mathsf{Z}}}\exp\left(-\left\|y-\mathrm{prox}^{\gamma}_{U_2}(x)+\tilde{\gamma}\tilde{\Theta}_1\{\mathrm{prox}^{\gamma}_{U_2}(x),z\}\right\|^2\Big/(4\tilde{\gamma})\right)\mathrm{d}\tilde{\eta}_1(z)\mathrm{d}y\ .\end{aligned}\tag{33}$$

Note that for all $\gamma,\tilde{\gamma}>0$, $\tilde{R}_{\gamma,\tilde{\gamma}}$ can be decomposed as the product $\tilde{S}^2_{\gamma}\tilde{S}^1_{\tilde{\gamma}}T_{\tilde{\gamma}}$ where $T_{\tilde{\gamma}}$ is defined by (14) and for all $x\in\mathbb{R}^d$ and $\mathsf{A}\in\mathcal{B}(\mathbb{R}^d)$

$$\tilde{S}^1_{\tilde{\gamma}}(x,\mathsf{A})=\int_{\tilde{\mathsf{Z}}}\mathbb{1}_{\mathsf{A}}(x-\tilde{\gamma}\tilde{\Theta}_1(x,z))\mathrm{d}\tilde{\eta}_1(z)\ ,\qquad \tilde{S}^2_{\gamma}(x,\mathsf{A})=\delta_{\mathrm{prox}^{\gamma}_{U_2}(x)}(\mathsf{A})\ .\tag{34}$$

Algorithm 2: SPGLD

Data: initial distribution $\mu_0\in\mathcal{P}_2(\mathbb{R}^d)$, non-increasing sequence $(\gamma_k)_{k\geq1}$, $U=U_1+U_2,\tilde{\Theta}_1,\tilde{\eta}_1$ satisfying **A**5
Result: $(\tilde{X}_k)_{k\in\mathbb{N}}$
begin
  Draw $\tilde{X}_0\sim\mu_0$;
  for $k\geq1$ **do**
    Draw $G_{k+1}\sim\mathcal{N}(0,\mathrm{Id})$ and $\tilde{Z}_{k+1}\sim\tilde{\eta}_1$;
    Set $\tilde{X}_{k+1}=\mathrm{prox}^{\gamma_{k+1}}_{U_2}(\tilde{X}_k)-\gamma_{k+2}\tilde{\Theta}_1(\mathrm{prox}^{\gamma_{k+1}}_{U_2}(\tilde{X}_k),\tilde{Z}_{k+1})+\sqrt{2\gamma_{k+2}}G_k$

Let$(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ be two non-increasing sequences of reals numbers and $\mu_0\in\mathcal{P}_2(\mathbb{R}^d)$ be an initial distribution. The weighted averaged distribution associated with (32) $(\tilde{\nu}^N_n)_{n\in\mathbb{N}}$ is defined for all $N,n\in\mathbb{N}$, $n\geq1$ by

$$\tilde{\nu}^N_n=\Lambda^{-1}_{N,N+n}\sum_{k=N+1}^{N+n}\lambda_k\,\mu_0\tilde{Q}^k_{\gamma}\ ,\qquad \tilde{Q}^k_{\gamma}=\tilde{R}_{\gamma_1,\gamma_2}\cdots\tilde{R}_{\gamma_k,\gamma_{k+1}}\ ,\ \text{for } k\in\mathbb{N}^*\ ,\tag{35}$$

where N is a burn-in time and $\Lambda_{N,N+n}$ is defined in (20). We take in the following the convention that $\tilde{Q}^0_{\gamma}$ is the identity operator.

Under **A**3, define for all $\mu\in\mathcal{P}_2(\mathbb{R}^d)$,

$$\begin{aligned}\upsilon_1(\mu)&=\int_{\mathbb{R}^d\times\tilde{\mathsf{Z}}}\left\|\tilde{\Theta}_1(x,z)-\int_{\tilde{\mathsf{Z}}}\tilde{\Theta}_1(x,\tilde{z})\mathrm{d}\tilde{\eta}_1(\tilde{z})\right\|^2\mathrm{d}\eta(z)\mathrm{d}\mu(x)\\&=\mathbb{E}\left[\left\|\tilde{\Theta}_1(\tilde{X}_0,\tilde{Z}_1)-\nabla U_1(\tilde{X}_0)\right\|^2\right]\ ,\end{aligned}\tag{36}$$

where $\tilde{X}_0,\tilde{Z}_1$ are independent random variables with distribution μ and $\tilde{\eta}_1$ respectively.

Theorem 17 *Assume* ***A****5(m), for $m\geq0$. Let $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ be two non-increasing sequences of positive real numbers satisfying $\gamma_1\in\left(0,L^{-1}\right]$, and for all $k\in\mathbb{N}^*$, $\lambda_{k+1}/\gamma_{k+2}\leq\lambda_k/\gamma_{k+1}$. Let $\mu_0\in\mathcal{P}_2(\mathbb{R}^d)$ and $N\in\mathbb{N}$. Then for all $n\in\mathbb{N}^*$, we have*

$$\begin{aligned}\mathrm{KL}\left(\tilde{\nu}^N_n|\pi\right)\leq\ &\lambda_{N+1}W_2^2\left(\mu_0\tilde{Q}^N_{\gamma}\tilde{S}^2_{\gamma_{N+1}},\pi\right)\Big/(2\gamma_{N+2}\Lambda_{N,N+n})\\&+(2\Lambda_{N,N+n})^{-1}\sum_{k=N+1}^{N+n}\lambda_k\gamma_{k+1}\{2Ld+(1+\gamma_{k+1}L)\upsilon_1(\mu_0Q^{k-1}_{\gamma}\tilde{S}^2_{\gamma_k})+2M_2^2\}\ .\end{aligned}$$

Proof The proof is postponed to Section 7.4.1. ■

Corollary 18 *Assume* ***A*** *5(m), for $m\geq0$. Assume that $\sup_{x\in\mathbb{R}^d}\upsilon_1(\delta_x)\leq D^2<\infty$. Let $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ given for all $k\in\mathbb{N}^*$ by $\lambda_k=\gamma_k=\gamma\in\left(0,L^{-1}\right]$. Let $\mu_0\in\mathcal{P}_2(\mathbb{R}^d)$. Then for any $N\in\mathbb{N},n\in\mathbb{N}^*$ we have*

$$\mathrm{KL}\left(\tilde{\nu}^N_n|\pi\right)\leq W_2^2\left(\mu_0\bar{Q}^N_{\gamma}\bar{S}_{\gamma},\pi\right)/(2n\gamma)+\gamma\left(Ld+M_2^2+D^2\right)\ ,$$

Furthermore, let $\varepsilon>0$ and

$$\gamma_{\varepsilon}\leq\min\left\{\varepsilon/(2(Ld+M_2^2+D^2)),L^{-1}\right\}\ ,\qquad n_{\varepsilon}\geq\left\lceil W_2^2(\mu_0\tilde{S}^2_{\gamma_1},\pi)(\gamma_{\varepsilon}\varepsilon)^{-1}\right\rceil.$$

Then we have $\mathrm{KL}\left(\tilde{\nu}^0_{n_{\varepsilon}}|\pi\right)\leq\varepsilon$.

In the case where a warm start is available for the Wasserstein distance, *i.e.* $W_2^2(\mu_0,\pi)\leq C$, for some absolute constant $C\geq0$, then Corollary 18 implies that the complexity of SPGLD to obtain a sample close from π in KL with a precision target $\varepsilon>0$ is of order $(d+M_2^2+D^2)\bar{\mathcal{O}}(\varepsilon^{-2})$. In addition, Pinsker inequality implies that the complexity of SPGLD for the total variation distance is of order $(d+M_2^2+D^2)\bar{\mathcal{O}}(\varepsilon^{-4})$.

In addition if we have access to $\eta>0$ and $M_{\eta}\geq0$, independent of the dimension, such that for all $x\in\mathbb{R}^d$, $x\notin\mathrm{B}(x^\star,M_{\eta})$, $U(x)-U(x^\star)\geq\eta\left\|x-x^\star\right\|$, Proposition 32, **A**5-(ii), (31) and (25) imply that starting at $\delta_{x^\star}$, the overall complexity of SPGLD for the KL is in this case $(\eta^{-2}d^2+M_{\eta}^2)(d+M_2^2+D^2)\bar{\mathcal{O}}(\varepsilon^{-2})$ and $(\eta^{-2}d^2+M_{\eta}^2)(d+M_2^2+D^2)\bar{\mathcal{O}}(\varepsilon^{-4})$ for the total variation distance If $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ are given for all $k\in\mathbb{N}^*$ by $\gamma_k=\lambda_k=\gamma_1/k^{-\alpha}$, $\gamma_1\in\left(0,L^{-1}\right]$. Then by the same reasoning as in the proof of Corollary 8, we obtain that there exists $C\geq0$ such that for all $n\in\mathbb{N}^*$, we have $\mathrm{KL}\left(\bar{\nu}^0_n|\pi\right)\leq C\max(n^{\alpha-1},n^{-\alpha})$, if $\alpha\neq1/2$, and for $\alpha=1/2$, we have $\mathrm{KL}\left(\bar{\nu}^0_n|\pi\right)\leq C(\ln(n)+1)n^{-1/2}$.

If $\sup_{x\in\mathbb{R}^d}\upsilon_1(\delta_x)<+\infty$ does not hold, we can control the variance of stochastic gradient estimates using **A**4 again based on this following result.

Proposition 19 *Assume* ***A*** *5 and $\tilde{\Theta}_1$ satisfies* ***A*** *4. Then we have for all $x\in\mathbb{R}^d$ and $\gamma\in(0,\tilde{L}^{-1}]$*

$$2\gamma(\tilde{L}^{-1}-\gamma)\upsilon_1(\delta_x)\leq\|x-x^\star\|^2-\int_{\mathbb{R}^d}\|y-x^\star\|^2(\tilde{S}^1_{\gamma}T_{\gamma}\tilde{S}^2_{\gamma})(x,\mathrm{d}y)+2\gamma^2\upsilon_1(\delta_{x^\star})+2\gamma d\ ,$$

where $\tilde{S}^1_{\gamma},\tilde{S}^2_{\gamma}$ and υ_1 are defined by (34)-(36) *respectively.*

Proof Let $\gamma > 0$, $x \in \mathbb{R}^d$ and consider $\tilde{X}_1 = \mathrm{prox}_{U_2}^{\gamma}\left\{x - \gamma\tilde{\Theta}_1(x, \tilde{Z}_1) + \sqrt{2\gamma}G_1\right\}$, where $\tilde{Z}_1$ and G_1 are two independent random variables, $\tilde{Z}_1$ has distribution $\tilde{\eta}_1$ and G_1 is the standard Gaussian random variable, so that $\tilde{X}_1$ has distribution $\tilde{S}_\gamma^1 T_\gamma \tilde{S}_\gamma^2(x, \cdot)$. First by (Bauschke and Combettes, 2011, Theorem 26.2(vii)), we have that $x^\star = \mathrm{prox}_{U_2}^{\gamma}(x^\star - \gamma\nabla U_1(x^\star))$ and by (Bauschke and Combettes, 2011, Proposition 12.27), the proximal is non-expansive, for all $x, y \in \mathbb{R}^d$, $\|\mathrm{prox}_{U_2}^{\gamma}(x) - \mathrm{prox}_{U_2}^{\gamma}(y)\| \leq \|x - y\|$. Using these two results and the fact that $\tilde{\Theta}_1$ satisfies **A** 4, we have

$$
\begin{aligned}
\mathbb{E}\left[\left\|\tilde{X}_1 - x^\star\right\|^2\right] &= \mathbb{E}\left[\left\|\mathrm{prox}_{U_2}^{\gamma}\left\{x - \gamma\tilde{\Theta}_1(x, \tilde{Z}_1) + \sqrt{2\gamma}G_1\right\} - \mathrm{prox}_{U_2}^{\gamma}\{x^\star - \gamma\nabla U_1(x^\star)\}\right\|^2\right] \\
&\leq \mathbb{E}\left[\left\|\left(x - \gamma\tilde{\Theta}_1(x, \tilde{Z}_1) + \sqrt{2\gamma}G_1\right) - (x^\star - \gamma\nabla U_1(x^\star))\right\|^2\right] \\
&\leq \|x - x^\star\|^2 \\
&\quad + \mathbb{E}\left[2\gamma\left\langle x - x^\star, \nabla U_1(x^\star) - \tilde{\Theta}_1(x, \tilde{Z}_1)\right\rangle + \gamma^2\left\|\nabla U_1(x^\star) - \tilde{\Theta}_1(x, \tilde{Z}_1)\right\|^2\right] + 2\gamma d \\
&\leq \|x - x^\star\|^2 - 2\gamma(\tilde{L}^{-1} - \gamma)\mathbb{E}\left[\left\|\tilde{\Theta}_1(x, \tilde{Z}_1) - \tilde{\Theta}_1(x^\star, \tilde{Z}_1)\right\|^2\right] \\
&\qquad + 2\gamma^2\mathbb{E}\left[\left\|\tilde{\Theta}_1(x^\star, \tilde{Z}_1) - \nabla U_1(x^\star)\right\|^2\right] + 2\gamma d\,.
\end{aligned}
$$

The proof is completed upon noting that $\upsilon_1(\delta_x) \leq \mathbb{E}[\left\|\tilde{\Theta}_1(x, \tilde{Z}_1) - \tilde{\Theta}_1(x^\star, \tilde{Z}_1)\right\|^2]$. ■

Combining Theorem 17 and Proposition 19, we get the following result.

Corollary 20 *Assume* ***A*** *5(m) for $m \geq 0$ and that $\tilde{\Theta}_1$ satisfies* ***A*** *4. Let $(\gamma_k)_{k\in\mathbb{N}^*}$ and $(\lambda_k)_{k\in\mathbb{N}^*}$ be two non-increasing sequences of positive real numbers given for all $k \in \mathbb{N}^*$ by $\gamma_k = \lambda_k = \gamma \in (0, L^{-1}]$, $\gamma < \tilde{L}^{-1}$. Let $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$ and $N \in \mathbb{N}$. Then for all $n \in \mathbb{N}^*$, it holds*

$$
\begin{aligned}
\mathrm{KL}\left(\tilde{\nu}_n^N \middle| \pi\right) &\leq W_2^2\left(\mu_0\tilde{Q}_\gamma^N\tilde{S}_{\gamma_{N+1}}^2, \pi\right)\Big/(2\gamma n) + \gamma(Ld + M_2^2) \\
&\quad + (1 + \gamma L)(2(\tilde{L}^{-1} - \gamma))^{-1}\left\{(2n)^{-1}\int_{\mathbb{R}^d}\|x - x^\star\|^2\,\mathrm{d}\mu_0\tilde{Q}_\gamma^N\tilde{S}_\gamma^2(y) + \gamma^2\upsilon_1(\delta_{x^\star}) + \gamma d\right\}\,.
\end{aligned}
$$

Furthermore, for $\varepsilon > 0$, consider stepsize and a number of iterations satisfying:

$$
\begin{aligned}
\gamma_\varepsilon &\leq \min\left[\varepsilon\Big/\left\{4M_2^2 + 4Ld + 8\tilde{L}d\right\}, \sqrt{\varepsilon/\left(8\tilde{L}\upsilon_1(\delta_{x^\star})\right)}, L^{-1}, (2\tilde{L})^{-1}\right]\,, \\
n_\varepsilon &\geq 2\max\left\{\left\lceil W_2^2(\mu_0\tilde{S}_{\gamma_\varepsilon}^2, \pi)(\gamma_\varepsilon\varepsilon)^{-1}\right\rceil, \left\lceil 2\tilde{L}\varepsilon^{-1}\int_{\mathbb{R}^d}\|x - x^\star\|^2\,\mathrm{d}\mu_0\tilde{S}_\gamma^2(y)\right\rceil\right\}\,.
\end{aligned}
$$

Then, we have $\mathrm{KL}\left(\tilde{\nu}_{n_\varepsilon}^0 \middle| \pi\right) \leq \varepsilon$.

Proof The proof of the corollary is a direct consequence of Theorem 17 and Proposition 19, and is postponed to Section 7.4.2. ■

Note that the dependency on the variance of the stochastic gradients is improved compared to the bound given by Corollary 18. We specify once again the result of Theorem 17 for strongly convex potential.

Theorem 21 *Assume* ***A*** *5(m), for $m > 0$. Let $(\gamma_k)_{k\in\mathbb{N}^*}$ be a non-increasing sequences of positive real numbers satisfying for all $k \in \mathbb{N}^*$, $\gamma_k \in \left(0, L^{-1}\right]$. Let $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$. Then for all $n \in \mathbb{N}^*$, it holds*

$$
\begin{aligned}
W_2^2(\mu_0\tilde{Q}_\gamma^n\tilde{S}_{\gamma_{n+1}}^2, \pi) &\leq \left\{\prod_{k=1}^n(1 - m\gamma_{k+1})\right\}W_2^2(\mu_0\tilde{S}_{\gamma_1}^2, \pi) \\
&\quad + \sum_{k=1}^n\gamma_{k+1}^2\left\{\prod_{i=k+2}^{n+1}(1 - m\gamma_i)\right\}\left\{2Ld + (1 + \gamma_{k+1}L)\upsilon_1(\mu_0\tilde{Q}_\gamma^{k-1}\tilde{S}_{\gamma_k}^2) + 2M_2^2\right\}\,.
\end{aligned}
$$

Proof The proof is postponed to Section 7.4.3. ■

Corollary 22 *Assume* ***A*** *5(m), for $m > 0$. Assume that $\sup_{x\in\mathbb{R}^d}\upsilon_1(\delta_x) \leq D^2 < \infty$. Let $\varepsilon > 0$, $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$, and*

$$
\gamma_\varepsilon \leq \min\left\{m\varepsilon/(4(Ld + D^2 + M_2^2)), L^{-1}\right\}\,, \qquad n_\varepsilon \geq \left\lceil \ln(2W_2^2(\mu_0\tilde{S}_{\gamma_\varepsilon}^2, \pi)/(\varepsilon\gamma_\varepsilon m)^{-1}\right\rceil\,.
$$

Then $W_2^2(\mu_0\tilde{R}_{\gamma_\varepsilon,\gamma_\varepsilon}^{n_\varepsilon}\tilde{S}_{\gamma_\varepsilon}^2, \pi) \leq \varepsilon$, where $\tilde{R}_{\gamma,\gamma}$ and $\tilde{S}_\gamma^2$ are defined by (33) and (34) respectively.

Proof Since $\gamma_\varepsilon \leq L^{-1}$, we have $(1 + \gamma_\varepsilon L)\upsilon_1(\mu_0\tilde{R}_{\gamma_\varepsilon}^k\tilde{S}_\gamma^2) \leq 2D^2$ for all $k \geq 1$. Using Theorem 21 then concludes the proof. ■

Note that the bounds given by Theorem 21 are tighter the one given by Dalalyan and Karagulyan (2017, Theorem 3) which shows under **A**5 with $U_2 = 0$ and $\sup_{x\in\mathbb{R}^d}\upsilon_1(\delta_x) \leq D^2$ that

$$
W_2(\mu_0\tilde{R}_{\gamma,\gamma}, \pi) \leq (1 - mh)W_2(\mu_0, \pi) + 1.65(L/m)(\gamma d)^{1/2} + D^2(\gamma d)^{1/2}/(1.65L + Dm)\,.
$$

Indeed, for constant stepsize $\gamma_k = \gamma \in (0, L^{-1}]$ for all $k \in \mathbb{N}^*$, Theorem 21 implies with the same assumptions that

$$
W_2(\mu_0\tilde{R}_{\gamma,\gamma}, \pi) \leq (1 - mh)^{1/2}W_2(\mu_0, \pi) + (2Ld\gamma/m)^{1/2} + ((1 + \gamma)\gamma/m)^{1/2}D\,.
$$

As for ULA, the dependency on the condition number L/m is improved.

In the strongly convex case, we can improve the dependency on the variance of the stochastic gradient under the following condition.

A 6 There exist $\tilde{L}_1, \tilde{m}_1 > 0$ such that for all for $\tilde{\eta}_1$-almost every $z \in \tilde{\mathsf{Z}}$, for all $x, y \in \mathbb{R}^d$, we have

$$
\left\langle\tilde{\Theta}_1(x, z) - \tilde{\Theta}_1(y, z), x - y\right\rangle \geq \tilde{m}_1\|x - y\|^2 + (1/\tilde{L}_1)\left\|\tilde{\Theta}_1(x, z) - \tilde{\Theta}_1(y, z)\right\|^2\,.
$$

The condition **A**6 is for example satisfied if η-almost surely, $x \mapsto \tilde{\Theta}_1(x,z)$ is strongly convex, see (Nesterov, 2004, Theorem 2.1.12).

Proposition 23 *Assume* ***A*** *5(m) for $m>0$ and* ***A*** *6. Then for all $\gamma>0$ we have*

$$2\gamma(\tilde{L}_1^{-1}-\gamma)\upsilon_1(\delta_x) \le (1-\tilde{m}_1\gamma)\left\|x-x^\star\right\|^2 - \int_{\mathbb{R}^d} \|y-x^\star\|^2 \, (\tilde{S}_\gamma^1 T_\gamma \tilde{S}_\gamma^2)(x,\mathrm{d}y) + 2\gamma^2\upsilon_1(\delta_{x^\star}) + 2\gamma d \;,$$

where $\tilde{S}_\gamma^1, \tilde{S}_\gamma^2$ and υ_1 are defined by (34)-(36) *respectively.*

Proof The proof is similar to the proof of Proposition 19 and is postponed to Section 7.4.4. ■

Corollary 24 *Assume* ***A*** *5(m), for $m>0$ and that $\tilde{\Theta}_1$ satisfies* ***A*** *6. Let $(\gamma_k)_{k\in\mathbb{N}^*}$ defined for all $k\in\mathbb{N}^*$ by $\gamma_k=\gamma\in(0, L^{-1}\wedge(2\tilde{L}_1)^{-1}]$. Let $\mu_0\in\mathcal{P}_2(\mathbb{R}^d)$. Define $\tilde{m}=\min(m,\tilde{m}_1)$ and*

$$\begin{aligned}\Delta_1 &= 2(Ld+M_2)/m + \{2\tilde{L}_1(1+\gamma L)/\tilde{m}\}d \\ \Delta_2 &= \{2\tilde{L}_1(1+\gamma L)/\tilde{m}\}\upsilon_1(\delta_{x^\star}) \\ \Delta_3 &= \gamma\tilde{L}_1(1+\gamma L)\left\{\int_{\mathbb{R}^d}\|x-x^\star\|^2\,\mathrm{d}\mu_0\tilde{S}^2_{\gamma_\varepsilon}(x)\right\}\;.\end{aligned} \tag{37}$$

Then for all $n\in\mathbb{N}^*$, it holds

$$W_2^2(\mu_0\tilde{R}^n_{\gamma,\gamma}\tilde{S}^2_\gamma,\pi) \;\le\; (1-m\gamma)^n W_2^2(\mu_0\tilde{S}^2_\gamma,\pi) + (1-\tilde{m}\gamma)^n\Delta_3 + \gamma\Delta_1 + \gamma^2\Delta_2 \;, \tag{38}$$

where $\tilde{R}_{\gamma,\gamma}$ and $\tilde{S}^2_\gamma$ are defined by (33) *and* (34).

Therefore, for $\varepsilon>0$ and

$$\begin{aligned}\gamma_\varepsilon &\le \min\left\{\varepsilon/(4\Delta_1), [\varepsilon/(4\Delta_2)]^{1/2}, L^{-1}, (2\tilde{L}_1)^{-1}\right\} \\ n_\varepsilon &\ge \max\left\{\lceil\ln(4W_2^2(\mu_0\tilde{S}^2_{\gamma_\varepsilon},\pi)/\varepsilon)(\gamma_\varepsilon m)^{-1}\rceil, \lceil\ln(4\Delta_3/\varepsilon)(\gamma_\varepsilon\tilde{m})^{-1}\rceil\right\}\;,\end{aligned}$$

it holds $W_2^2(\mu_0\tilde{R}^{n_\varepsilon}_{\gamma_\varepsilon,\gamma_\varepsilon}\tilde{S}^2_{\gamma_\varepsilon},\pi)\le\varepsilon$.

Proof The proof of the corollary is postponed to Section 7.4.5. ■

Corollary 25 *Assume* ***A*** *5(m), for $m>0$ and that $\tilde{\Theta}_1$ satisfies* ***A*** *6. Define $\tilde{m}=\min(m,\tilde{m}_1)$. Let $\varepsilon>0$, $\mu_0\in\mathcal{P}_2(\mathbb{R}^d)$ and*

$$\begin{aligned}\gamma_\varepsilon &\le \min\left\{\varepsilon/(4\Delta_1), [\varepsilon/(4\Delta_2)]^{1/2}, L^{-1}, (2\tilde{L}_1)^{-1}\right\}\;, \\ N_\varepsilon &\ge \max\left\{\lceil\ln(4W_2^2(\mu_0\tilde{S}^2_{\gamma_\varepsilon},\pi)/\varepsilon)(\gamma_\varepsilon m)^{-1}\rceil, \lceil\ln(4\Delta_3/\varepsilon)(\gamma_\varepsilon\tilde{m})^{-1}\rceil\right\} \\ \tilde{\gamma}_\varepsilon &\le \min\left[\varepsilon\Big/\left\{4M_2^2+4Ld+8\tilde{L}d\right\}, \sqrt{\varepsilon/\left(8\tilde{L}\upsilon_1(\delta_{x^\star})\right)}, L^{-1}, (2\tilde{L})^{-1}\right]\;, \\ n_\varepsilon &\ge 2\max\left\{\lceil\gamma_\varepsilon^{-1}\rceil, \left\lceil 2\tilde{L}\varepsilon^{-1}\int_{\mathbb{R}^d}\|x-x^\star\|^2\,\mathrm{d}\mu_0\tilde{R}^{N_\varepsilon}_{\gamma_\varepsilon,\gamma_\varepsilon}\tilde{S}^2_\gamma(y)\right\rceil\right\}\;,\end{aligned}$$

where $\Delta_1,\Delta_2,\Delta_3$ are defined in (37) *and $\tilde{R}_{\gamma,\gamma}$ and $\tilde{S}^2_\gamma$ are defined by* (33) *and* (34). *Let $(\gamma_k)_{k\in\mathbb{N}}$ defined by $\gamma_k=\gamma_\varepsilon$ for $k\in\{1,\dots,N_\varepsilon\}$ and $\gamma_k=\tilde{\gamma}_\varepsilon$ for $k>N_\varepsilon$. Then we have $\mathrm{KL}\left(\tilde{\nu}^{N_\varepsilon}_{n_\varepsilon}|\pi\right)\le\varepsilon$ where $\tilde{\nu}^{N_\varepsilon}_{n_\varepsilon}=n_\varepsilon^{-1}\sum_{k=1}^{n_\varepsilon}\mu_0\tilde{R}^{N_\varepsilon}_{\gamma_\varepsilon,\gamma_\varepsilon}\tilde{R}^k_{\tilde{\gamma}_\varepsilon,\tilde{\gamma}_\varepsilon}$.*

Proof Corollary 24 implies that after the burn in phase of N_ε steps with stepsize γ_ε, we have $W_2^2(\mu_0\tilde{Q}^{N_\varepsilon}_\gamma\tilde{S}^2_{\gamma_\varepsilon},\pi)\le\varepsilon$. Then, since we can treat $\mu_0\tilde{Q}^{N_\varepsilon}_\gamma$ as a new starting measure, Corollary 20 concludes the proof. ■

4.2.1. Discussion on Related Works

Note that the SPGLD is different from the algorithm MYULA proposed by Durmus et al. (2018) which approximates U_2 by its Moreau envelope, and under **A**5 defines the Markov chain $(X^{\mathrm{M}}_k)_{k\in\mathbb{N}}$ by the recursion:

$$X^{\mathrm{M}}_{k+1} = (1-\gamma/\lambda^{\mathrm{M}}) + (\gamma/\lambda^{\mathrm{M}})\,\mathrm{prox}^{\lambda^{\mathrm{M}}}_{U_2}(X^{\mathrm{M}}_k) - \gamma\nabla U_1(X^{\mathrm{M}}_k) + \sqrt{2\gamma}G_{k+1}\;,$$

for a stepsize $\gamma>0$, $(G_k)_{k\in\mathbb{N}^*}$ a sequence of i.i.d. d-dimensional standard Gaussian random variables and $\lambda^{\mathrm{M}}>0$ a regularization parameter. So taking $\lambda^{\mathrm{M}}=\gamma$, we get that the recursion boils down to

$$X^{\mathrm{M}}_{k+1} = \mathrm{prox}^{\gamma}_{U_2}(X^{\mathrm{M}}_k) - \gamma\nabla U_1(X^{\mathrm{M}}_k) + \sqrt{2\gamma}G_{k+1}\;.$$

Then the main difference with (32) setting $\gamma_k=\gamma$ for all $k\in\mathbb{N}^*$ and $\tilde{\Theta}_1=\nabla U_1$ is that $\nabla U_1(X^{\mathrm{M}}_k)$ is replaced in (32) by $\nabla U_1(\mathrm{prox}^{\gamma}_{U_2}(X^{\mathrm{M}}_k))$.

Recently, two papers have independently interpreted ULA as an algorithm that optimizes $\mathscr{F}$ on the Wasserstein space. They both relies on this interpretation to derive and analyze different algorithms for sampling log-concave target measures, that use the proximal operator associated with U.

First, Wibisono (2018) proposed and analyzed the symmetrized Langevin algorithm (SLA) in order to reduce the discretization bias. SLA combines backward and forward steps, *i.e.* it defines the Markov chain $(X^{\mathrm{SLA}}_k)_{k\in\mathbb{N}}$ by the recursion

$$X^{\mathrm{SLA}}_{k+1} = \mathrm{prox}^{\gamma}_{U}\left\{X^{\mathrm{SLA}}_k - \gamma\nabla U(X^{\mathrm{SLA}}_k) + \sqrt{4\gamma}G_{k+1}\right\}\;, \tag{39}$$

for a stepsize $\gamma>0$ and $(G_k)_{k\in\mathbb{N}^*}$ a sequence of i.i.d. d-dimensional standard Gaussian random variables. Wibisono (2018) shows that in the Gaussian case, *i.e.* for any $x\in\mathbb{R}^d$, $U(x)=\langle(x-\bar{x})\Sigma^{-1},(x-\bar{x})\rangle/2$ for some mean $\bar{x}\in\mathbb{R}^d$ and covariance matrix $\Sigma\in\mathbb{R}^{d\times d}$, SLA is unbiased and converges with an exponential rate to π. For general strongly convex potentials U, exponential convergence in Wasserstein distance of the sequence of distributions associated with $(X^{\mathrm{SLA}}_k)_{k\in\mathbb{N}}$ to a biased limit $\pi^{\mathrm{SLA}}_\gamma$ is established as $k\to+\infty$. However, no explicit bound on $W_2(\pi_\gamma,\pi^{\mathrm{SLA}})$ or nonasymptotic convergence rates are given.

In a parallel work, Bernton (2018) considers the proximal Langevin algorithm (PLA). This algorithm defines the Markov chain $(X^{\mathrm{PLA}}_k)_{k\in\mathbb{N}}$ by the recursion

$$X^{\mathrm{PLA}}_{k+1} = \mathrm{prox}^{\gamma_{k+1}}_{U}(X^{\mathrm{PLA}}_k) - \gamma\sqrt{2\gamma_{k+1}}G_{k+1}\;.$$

Applying techniques from Ambrosio et al. (2008), quantitative results on the Wasserstein distance between the above discretization and gradient flow of KL divergence are obtained. From those results, bounds on the Wasserstein distance between iterates and target distributions are given, in the case where U is strongly convex. The complexity bounds for PLA obtained in Bernton (2018) are of order $d\bar{\mathcal{O}}(\varepsilon^{-2})$ when U is smooth and strongly convex, and are equivalent to our results up to dependence on the starting measure. In the case of $U = U_1 + U_2$ where U_1 is strongly convex and U_2 is Lipschitz, bounds in Bernton (2018) are of order $d^2\bar{\mathcal{O}}(\varepsilon^{-4})$ while our bounds are still of the same order as in the smooth case.

5. Numerical Experiments

In this section, we experiment SPGLD and SSGLD on a Bayesian logistic regression problem, see e.g. (Holmes and Held, 2006; Gramacy and Polson, 2012; Park and Hastie, 2007). Consider i.i.d. observations $(X_i, Y_i)_{i\in\{1,\dots,N\}}$, where $(Y_i)_{i\in\{1,\dots,N\}}$ are binary response variables and $(X_i)_{i\in\{1,\dots,N\}}$ are d-dimensional covariance variables. For all $i \in \{1,\dots,N\}$, Y_i is assumed to be a Bernoulli random variable with parameter $\Phi(\beta^{\mathrm{T}} X_i)$ where β is the parameter of interest and for all $u \in \mathbb{R}$, $\Phi(u) = \mathrm{e}^u/(1+\mathrm{e}^u)$. We choose as prior distributions (see Genkin et al. (2007) and Li and Lin (2010)) a d-dimensional Laplace distribution and a combination of the Laplace distribution and the Gaussian distribution, with density with respect to the Lebesgue measure given respectively for all $\beta \in \mathbb{R}^d$ by

$$\mathrm{p}_1(\beta) \propto \exp\left(-a_1 \sum_{i=1}^{d} |\beta_i|\right) , \qquad \mathrm{p}_{1,2}(\beta) \propto \exp\left(-a_1 \sum_{i=1}^{d} |\beta_i| - a_2 \sum_{i=1}^{d} \beta_i^2\right) ,$$

where a_1 is set to 1 in the case of p_1 and $a_1 = 0.9$, $a_2 = 0.1$ in the case of $\mathrm{p}_{1,2}$. The chosen priors on the one hand reduce impact of the irrelevant features by shrinking them close to zero. On the other hand this choice of priors leads to the log-concave posteriors. Both these property are highly desirable in the high dimensional setting. We obtain then the two different a posteriori distributions $\mathrm{p}_1(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$ and $\mathrm{p}_{1,2}(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$ with potentials given, respectively, by

$$\beta \mapsto \sum_{n=1}^{N} \ell_n(\beta) + a_1 \sum_{i=1}^{d} |\beta_i| , \qquad \beta \mapsto \sum_{n=1}^{N} \ell_n(\beta) + a_2 \sum_{i=1}^{d} \beta_i^2 + a_1 \sum_{i=1}^{d} |\beta_i| .$$

where

$$\ell_n(\beta) = -Y_n \beta^{\mathrm{T}} X_n + \log[1 + \exp(\beta^{\mathrm{T}} X_n)] .$$

We consider the three data sets from UCI repository (Dua and Efi, 2017) Heart disease dataset ($N = 270$, $d = 14$), Australian Credit Approval dataset ($N = 690$, $d = 34$) and Musk dataset ($N = 476$, $d = 166$). We approximate $\mathrm{p}_1(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$ using SPGLD and SSGLD, since the associated potential is Lipschitz, whereas regarding $\mathrm{p}_{1,2}(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$ we only apply SPGLD.

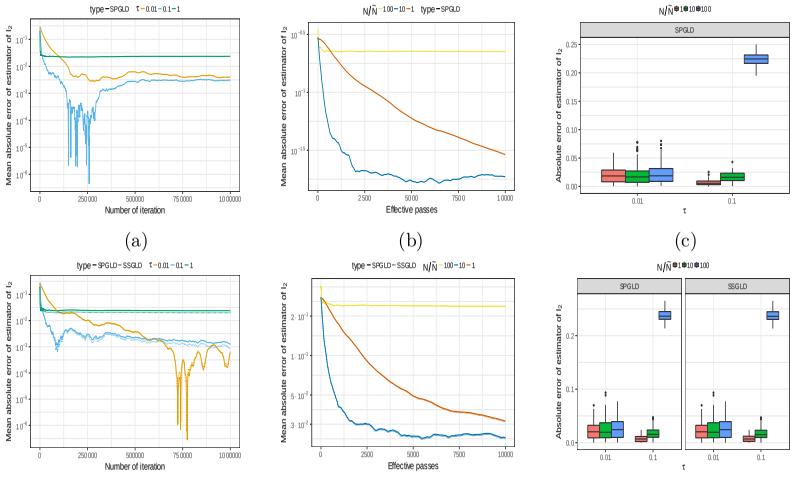


Figure 1: Mean absolute error of estimator of I_2 for Australian Credit Approval dataset: (top row) results for $\mathrm{p}_{1,2}(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$; (a) convergence of SPGLD for $\tilde{N} = N$, (b) convergence of SPGLD in terms of effective passes for $\tau = 0.1$, (c) boxplot of SPGLD for full runs; (bottom row) results for $\mathrm{p}_1(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$; (d) convergence of SPGLD and SSGLD for $\tilde{N} = N$, (e) convergence of SPGLD and SSGLD in terms of effective passes for $\tau = 0.1$, (f) boxplot of SPGLD and SSGLD for full run.

SPGLD is performed using the following stochastic gradient

$$\tilde{\Theta}_1(\beta, Z) = (N/\tilde{N}) \sum_{n\in Z} \nabla \ell_n(\beta) + a_2 \beta ,$$

where a_2 is set to 0 in the case of $\mathrm{p}_1(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$ and Z is a uniformly distributed random subset of $\{1,\dots,N\}$ with cardinal $\tilde{N} \in \{1,\dots,N\}$. In addition, the proximal operator associated with $\beta \mapsto a_1 \sum_{i=1}^{d} |\beta_i|$ is given for all $\beta \in \mathbb{R}^d$ and $\gamma > 0$ by (see e.g. Parikh and Boyd (2013))

$$(\mathrm{prox}^{\gamma}_{a_1,\ell_1}(\beta))_i = \mathrm{sign}(\beta_i) \max(|\beta_i| - a_1\gamma, 0) , \quad \text{for } i \in \{1,\dots,d\} .$$

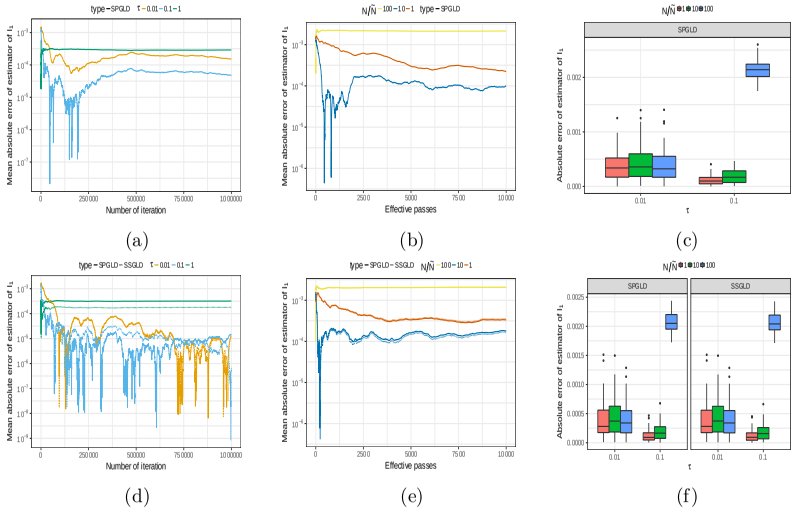


Figure 2: Mean absolute error of estimator of I_1 for Australian Credit Approval dataset: (top row) results for $\mathrm{p}_{1,2}(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$; (a) convergence of SPGLD for $\tilde{N} = N$, (b) convergence of SPGLD in terms of effective passes for $\tau = 0.1$, (c) boxplot of SPGLD for full runs; (bottom row) results for $\mathrm{p}_1(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$; (d) convergence of SPGLD and SSGLD for $\tilde{N} = N$, (e) convergence of SPGLD and SSGLD in terms of effective passes for $\tau = 0.1$, (f) boxplot of SPGLD and SSGLD for full run.

SSGLD is performed using the following stochastic subgradient

$$\Theta(\beta, Z) = (N/\tilde{N}) \sum_{n\in Z} \nabla \ell_n(\beta) + a_1 \sum_{i=1}^{d} \mathrm{sign}(\beta_i)\mathbf{e}_i \ ,$$

where $(\mathbf{e}_i)_{i\in\{1,\dots,d\}}$ denotes the canonical basis and Z is a uniformly distributed random subset of $\{1,\dots,N\}$ with cardinal $\tilde{N} \in \{1,\dots,N\}$.

Based on the results of SPGLD and SSGLD, we estimate the posterior mean I_1 and I_2 of the test functions $\beta \mapsto \beta_1$ and $\beta \mapsto (1/d)\sum_{i=1}^{d} \beta_i^2$. For our experiments, we use constant stepsizes γ of the form $\tau(L+m)^{-1}$ with $\tau = 0.01, 0.1, 1$ and for stochastic (sub) gradient we use $\tilde{N} = N, \lfloor N/10 \rfloor, \lfloor N/100 \rfloor$. For all datasets and all settings of τ, $\tilde{N}$ we run 100 independent runs of SPGLD (SSGLD), where each run was of length 10^6. For each set of parameters we estimate I_1, I_2 and we compute the absolute errors, where the true value were obtained by prox-MALA (see Pereyra (2015)) with 10^7 iterations and stepsize corresponding to optimal acceptance ratio ≈ 0.5, see (Roberts and Rosenthal, 1998). The results for I_2 are presented on Figure 1, Figure 3 and Figure 5 for Australian Credit Approval dataset,

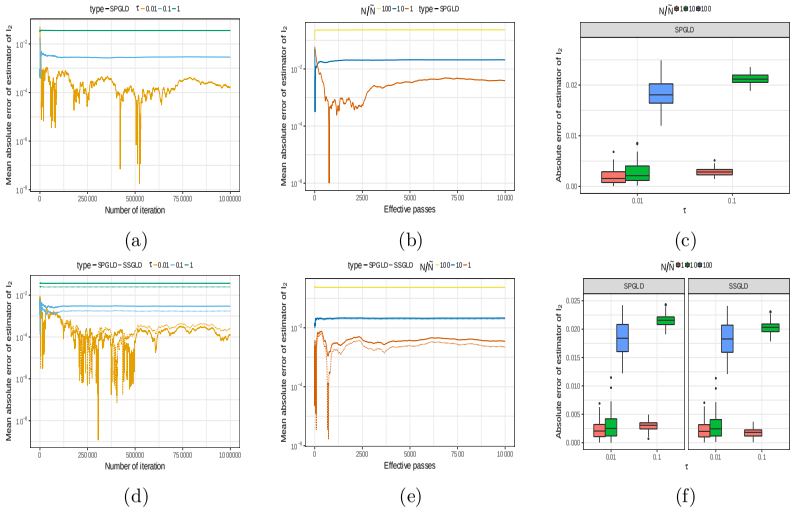


Figure 3: Mean absolute error of estimator of I_2 for Heart disease dataset: (top row) results for $\mathrm{p}_{1,2}(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$; (a) convergence of SPGLD for $\tilde{N} = N$, (b) convergence of SPGLD in terms of effective passes for $\tau = 0.1$, (c) boxplot of SPGLD for full run; (bottom row) results for $\mathrm{p}_1(\cdot|(X,Y)_{i\in\{1,\dots,N\}})$; (d) convergence of SPGLD and SSGLD for $\tilde{N} = N$, (e) convergence of SPGLD and SSGLD in terms of effective passes for $\tau = 0.1$, (f) boxplot of SPGLD and SSGLD for full run.

Heart disease dataset and Musk data respectively. The results for I_1 are presented on Figure 2, Figure 4 and Figure 6 for Australian Credit Approval dataset, Heart disease dataset and Musk data respectively. We note that in the all cases, bias decreases but convergence becomes slower with decreasing γ. When we look for stochastic (sub)gradient then the bias of estimators and also their variance increase when we decrease $\tilde{N}$. However if we look for the effective passes, i.e. the number of iteration is scaled with the cost of computing gradients, we observe that convergence is faster with reasonably small $\tilde{N}$. If we compare SSGLD with SPGLD we see that in the almost all cases, except Musk dataset, SSGLD leads to slightly smaller bias. For the Musk dataset differences between SSGLD and SPGLD are negligible and we do not present the results for SPGLD. In the presented experiments, all results agrees with our theoretical findings and suggest that SPGLD or SSGLD could be an alternative for other MCMC methods.

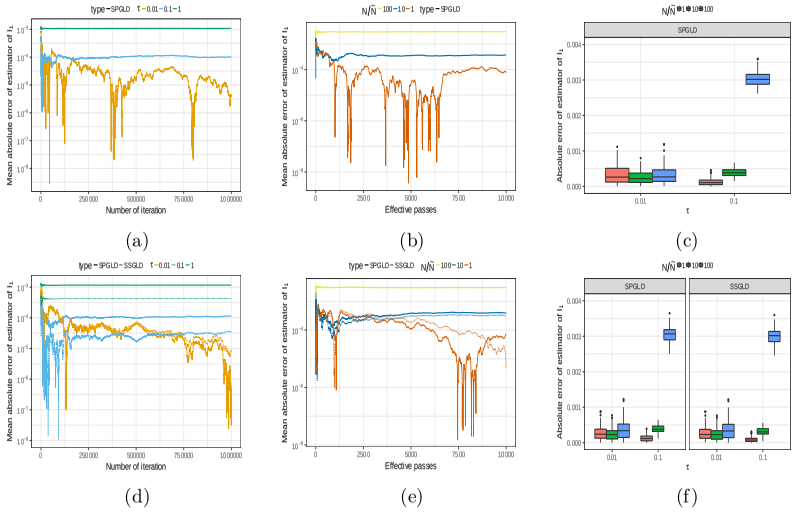


Figure 4: Mean absolute error of estimator of I_1 for Heart disease dataset: (top row) results for $\mathrm{p}_{1,2}(\cdot|(X,Y)_{i\in\{1,\ldots,N\}})$; (a) convergence of SPGLD for $\tilde{N} = N$, (b) convergence of SPGLD in terms of effective passes for $\tau = 0.1$, (c) boxplot of SPGLD for full run; (bottom row) results for $\mathrm{p}_1(\cdot|(X,Y)_{i\in\{1,\ldots,N\}})$; (d) convergence of SPGLD and SSGLD for $\tilde{N} = N$, (e) convergence of SPGLD and SSGLD in terms of effective passes for $\tau = 0.1$, (f) boxplot of SPGLD and SSGLD for full run.

6. Discussion

In this paper, we presented a novel interpretation of the Unadjusted Langevin Algorithm as the first order optimization algorithm, and a new technique of proving non-asymptotic bounds for ULA, based on the proof techniques known from convex optimization. Our proof technique gives simpler proofs of some of the previously known non-asymptotic results for ULA. It can be also used to prove non-asymptotic bound that were previously unknown. Specifically, to the best of the authors knowledge, we provide the first non-asymptotic results for Stochastic Gradient ULA in the non-strongly convex case, as well as the first non-asymptotic results in the non-smooth non-strongly convex case. Furthermore, our technique extends effortlessly to the stochastic non-smooth case, and to the best of the authors knowledge we provide the first non-asymptotic analysis of that case.

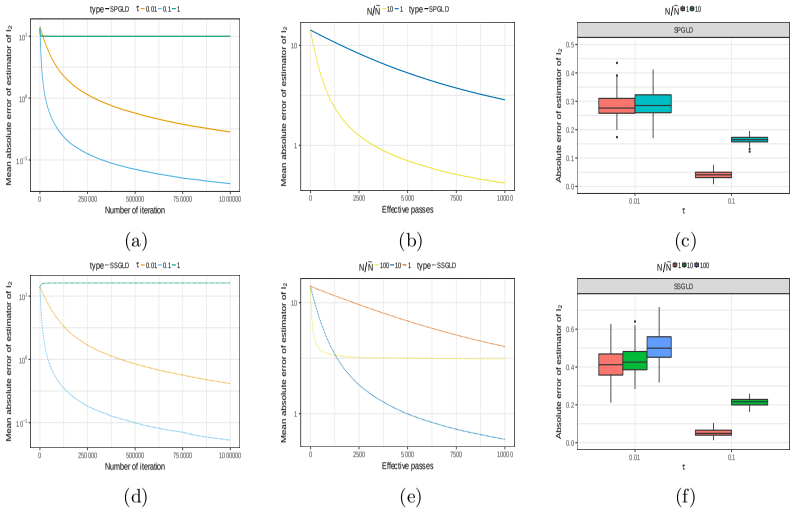


Figure 5: Mean absolute error of estimator of I_2 for Musk dataset: (top row) results for p_{1-2} prior; (a) convergence of SPGLD for $\tilde{N} = N$, (b) convergence of SPGLD in terms of effective passes for $\tau = 0.1$, (c) boxplot of SPGLD for full run; (bottom row) results for p_1 prior; (d) convergence of SSGLD for $\tilde{N} = N$, (e) convergence of SSGLD in terms of effective passes for $\tau = 0.1$, (f) boxplot of SSGLD for full run.

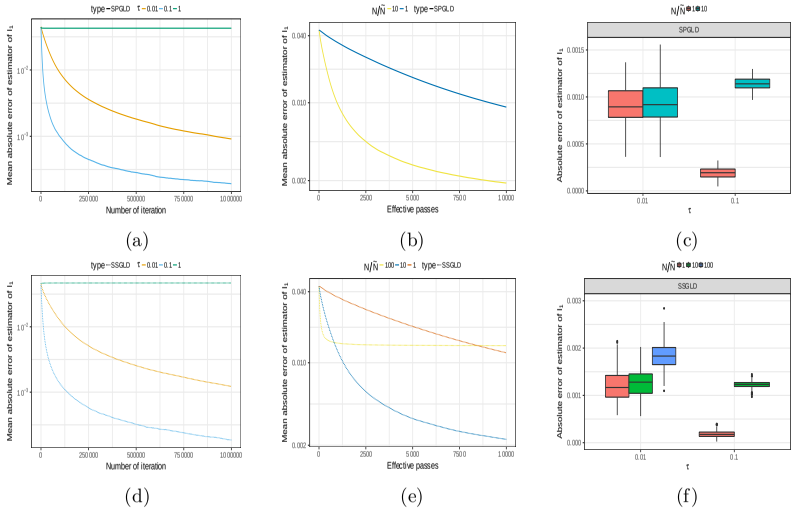
Furthermore our new perspective on the Unadjusted Langevin Algorithm, provides a starting point for the further research into connections between Langevin Monte Carlo and Optimization. In particular, we believe that a very promising direction for future research is to try to modify well-known effective optimization algorithms to minimize the KL divergence with respect to some target density π in Wasserstein space.

7. Postponed Proofs

7.1. Proof of Lemma 1

a) Since e^{-U} is integrable with respect to the Lebesgue measure, under $\mathbf{A}1(m)$ for $m \geq 0$, by (Brazitikos et al., 2014, Lemma 2.2.1), there exists $C_1, C_2 > 0$ such that for all $x \in \mathbb{R}^d$, $U(x) \geq C_1 \|x\| - C_2$. This inequality and $\mathbf{A}2$ implies that $\pi \in \mathcal{P}_2(\mathbb{R}^d)$. In addition, since the function $x \mapsto U(x)\mathrm{e}^{-U(x)/2}$ is bounded on $[-C_2, +\infty)$, we have for all $x \in \mathbb{R}^d$,

$$\left| \left(U(x)\mathrm{e}^{-U(x)/2} \right) \mathrm{e}^{-U(x)/2} \right| \leq C_3 \mathrm{e}^{-U(x)/2}$$



(a) (b) (c)

(d) (e) (f)

Figure 6: Mean absolute error of estimator of I_1 for Musk dataset: (top row) results for p_{1-2} prior; (a) convergence of SPGLD for $\tilde{N} = N$, (b) convergence of SPGLD in terms of effective passes for $\tau = 0.1$, (c) boxplot of SPGLD for full run; (bottom row) results for p_1 prior; (d) convergence of SSGLD for $\tilde{N} = N$, (e) convergence of SSGLD in terms of effective passes for $\tau = 0.1$, (f) boxplot of SSGLD for full run.

for some constant C_3. From this, and $U(x) \geq C_1 \|x\| - C_2$ we conclude that $\mathscr{E}(\pi) < +\infty$. Using the same reasoning, we have $\mathscr{H}(\pi) < +\infty$ which finishes the proof of the first part.
b) First, if μ does not admit a density with respect to Lebesgue measure, then both sides of (10) are $+\infty$. Second, if μ admits a density still denoted by μ with respect to the Lebesgue measure, we have by (7):

$$\mathscr{F}(\mu) - \mathscr{F}(\pi) = \mathrm{KL}\left(\mu|\pi\right) + \int_{\mathbb{R}^d} \{\mu(x) - \pi(x)\} \{U(x) + \log(\pi(x))\} \, \mathrm{d}x = \mathrm{KL}\left(\mu|\pi\right) \ .$$

7.2. Proof of Corollary 8

Using Theorem 6 we first get

$$\mathrm{KL}\left(\nu_n|\pi\right) \leq W_2^2\left(\mu_0, \pi\right)/(2\Gamma_{0,n}) + (Ld/\Gamma_{0,n}) \sum_{k=1}^{n} \gamma_k^2 \ . \tag{40}$$

Note that using a simple integral test, we have $\Gamma_{0,n} \geq C_1 n^{1-\alpha}$ for some constant $C_1 \geq 0$. On the other hand, for some constant $C_2 \geq 0$ we have $\sum_{k=1}^{n} \gamma_k^2 \leq C_2(1 + n^{1-2\alpha})$ if $\alpha \neq 1/2$, and $\sum_{k=1}^{n} \gamma_k^2 \leq C_2(1+\log(n))$ if $\alpha = 1/2$. Combining all these inequalities in (40) concludes the proof.

7.3. Proofs of Section 4.1

Note that for all $\gamma, \tilde{\gamma} > 0$, $\bar{R}_{\gamma,\tilde{\gamma}}$ can be decomposed as $\bar{S}_\gamma T_{\tilde{\gamma}}$ where $T_{\tilde{\gamma}}$ is defined in (14) and $\bar{S}_\gamma$ is given by (30). Then similarly to the proof of Theorem 6, we first give a preliminary bound on $\mathscr{F}(\mu\bar{R}_{\gamma,\tilde{\gamma}}) - \mathscr{F}(\pi)$ for $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ and $\gamma, \tilde{\gamma} > 0$ as in Proposition 2.

Lemma 26 *Assume* ***A****1(0) and* ***A****3. For all $\gamma > 0$ and $\mu \in \mathcal{P}_2(\mathbb{R}^d)$,*

$$2\gamma \left\{\mathscr{E}(\mu) - \mathscr{E}(\pi)\right\} \leq W_2^2(\mu, \pi) - W_2^2(\mu\bar{S}_\gamma, \pi) + \gamma^2 \left\{M^2 + v_\Theta(\mu)\right\} \ ,$$

where $\mathscr{E}$ and T_γ are defined in (9) *and* (14) *respectively, $v_\Theta(\mu)$ in* (29) *and $\bar{S}_\gamma$ in* (30)*.*

Proof Let Z be a random variable with distribution η, $\gamma > 0$ and $\mu \in \mathcal{P}_2(\mathbb{R}^d)$. For all $x, y \in \mathbb{R}^d$, we have using the definition of $\partial U(x)$ (24) and **A**3-(ii)

$$\begin{aligned}
\|y - x + \gamma\Theta(x, Z)\|^2 &= \|y - x\|^2 + 2\gamma \left\langle \Theta(x, Z), y - x \right\rangle + \gamma^2 \|\Theta(x, Z)\|^2 \\
&\leq \|y - x\|^2 - 2\gamma \left\{U(x) - U(y)\right\} + 2\gamma \left\langle \Theta(x, Z) - \mathbb{E}\left[\Theta(x, Z)\right], y - x \right\rangle + \gamma^2 \|\Theta(x, Z)\|^2 \ .
\end{aligned}$$

Let (X, Y) be an optimal coupling between μ and π independent of Z. Then by **A**3-(ii) and rearranging the terms in the previous inequality, we obtain

$$2\gamma \left\{\mathscr{E}(\mu) - \mathscr{E}(\nu)\right\} \leq W_2^2(\mu, \pi) - \mathbb{E}\left[\|Y - X + \gamma\Theta(X, Z)\|^2\right] + \gamma^2 \mathbb{E}\left[\|\Theta(X, Z)\|^2\right] \ .$$

The proof is concluded upon noting that $W_2^2(\mu\bar{S}_\gamma, \pi) \leq \mathbb{E}[\|Y - X + \gamma\Theta(X, Z)\|^2]$ and $\mathbb{E}[\|\Theta(X, Z)\|^2] \leq M^2 + v_\Theta(\mu)$. ■

Proposition 27 *Assume* ***A****1(0) and* ***A****3. For all $\gamma, \tilde{\gamma} > 0$ and $\mu \in \mathcal{P}_2(\mathbb{R}^d)$,*

$$2\tilde{\gamma} \left\{\mathscr{F}(\mu\bar{R}_{\gamma,\tilde{\gamma}}) - \mathscr{F}(\pi)\right\} \leq \left\{W_2^2\left(\mu\bar{S}_\gamma, \pi\right) - W_2^2\left(\mu\bar{R}_{\gamma,\tilde{\gamma}}\bar{S}_{\tilde{\gamma}}, \pi\right)\right\} + \tilde{\gamma}^2 \left\{M^2 + v_\Theta(\mu\bar{R}_{\gamma,\tilde{\gamma}})\right\} \ .$$

where $\mathscr{F}$ is defined in (9)*, $v_\Theta(\mu)$ in* (29)*, $\bar{R}_{\gamma,\tilde{\gamma}}$ and $\bar{S}_\gamma$ in* (27) *in* (30) *respectively.*

Proof Note that by Lemma 26, we have

$$2\tilde{\gamma} \left\{\mathscr{E}(\mu\bar{R}_{\gamma,\tilde{\gamma}}) - \mathscr{E}(\pi)\right\} \leq W_2^2\left(\mu\bar{R}_{\gamma,\tilde{\gamma}}, \pi\right) - W_2^2\left(\mu\bar{R}_{\gamma,\tilde{\gamma}}\bar{S}_{\tilde{\gamma}}, \pi\right) + \tilde{\gamma}^2 \left\{M^2 + v_\Theta(\mu\bar{R}_{\gamma,\tilde{\gamma}})\right\} \ . \tag{41}$$

In addition by Lemma 5, it holds

$$2\tilde{\gamma} \left\{\mathscr{H}(\mu\bar{R}_{\gamma,\tilde{\gamma}}) - \mathscr{H}(\pi)\right\} \leq W_2^2\left(\mu\bar{S}_\gamma, \pi\right) - W_2^2\left(\mu\bar{R}_{\gamma,\tilde{\gamma}}, \pi\right) \ .$$

The proof then follows from combining this inequality with (41). ■

7.3.1. Proof of Theorem 13

By Proposition 27, for all $k \in \mathbb{N}^*$, we have

$$\mathscr{F}(\mu\bar{Q}_\gamma^k) - \mathscr{F}(\pi) \leq (2\gamma_{k+1})^{-1} \left\{ W_2^2 \left(\mu\bar{Q}_\gamma^{k-1} \bar{S}_{\gamma_k}, \pi \right) - W_2^2 \left(\mu\bar{Q}_\gamma^k \bar{S}_{\gamma_{k+1}}, \pi \right) \right\} + (\gamma_{k+1}/2) \left\{ M^2 + \upsilon_\Theta(\mu\bar{Q}_\gamma^k) \right\} .$$

Similarly to the proof of Theorem 6 using the convexity of KL divergence and the condition that $(\lambda_k/\gamma_{k+1})_{k\in\mathbb{N}^*}$ is non-increasing concludes the proof.

7.3.2. Proof of Corollary 16

On the one hand, using Theorem 13, we get:

$$\mathrm{KL}\left(\tilde{\nu}_n^N \middle| \pi\right) \leq (2\gamma n)^{-1} W_2^2 \left(\mu_0 \bar{Q}_\gamma^N \bar{S}_\gamma, \pi\right) + \gamma M^2/2 + (\gamma/(2n)) \sum_{k=N+1}^{N+n} \upsilon_\Theta(\mu_0 \bar{Q}_\gamma^k) \ .$$

On the other hand, using Proposition 15, we obtain:

$$2\gamma(\tilde{L}^{-1} - \gamma) \left(\sum_{k=N+1}^{N+n} \upsilon_\Theta(\mu_0 \bar{Q}_\gamma^k) \right) \leq \int_{\mathbb{R}^d} \|x - x^\star\|^2 \, \mathrm{d}\mu_0 \bar{Q}_\gamma^{N+1}(x) - \int_{\mathbb{R}^d} \|x - x^\star\|^2 \, \mathrm{d}\mu_0 \bar{Q}_\gamma^{N+n+1} + 2n\gamma^2 \upsilon_\Theta(\delta_{x^\star}) + 2n\gamma d \ .$$

Combining the two inequalities above finishes the proof of the first part of Corollary 16. For the second part, first observe that since $\gamma_\varepsilon \leq (2\tilde{L})^{-1}$ we have $(2(\tilde{L}^{-1} - \gamma))^{-1} \leq \tilde{L}$. Furthermore, from the definition of γ_ε we have $\gamma_\varepsilon(\frac{M^2}{2} + \tilde{L}d) \leq \varepsilon/4$, as well as $\gamma_\varepsilon^2 \tilde{L} \upsilon_\Theta(\delta_{x^\star}) \leq \varepsilon/4$. On the other hand, from the definition of n_ε we have $W_2^2(\mu_0 \bar{S}_{\gamma_\varepsilon}, \pi)/(2\gamma\varepsilon n_\varepsilon) \leq \varepsilon/4$ as well as $\tilde{L}(2n_\varepsilon)^{-1} \int_{\mathbb{R}^d} \|x - x^\star\|^2 \, \mathrm{d}\mu_0 \bar{R}_{\gamma_\varepsilon, \gamma_\varepsilon}(x) \leq \varepsilon/4$. Combining those four bounds together finishes the proof.

7.4. Proof of Section 4.2

We proceed for the proof of Theorem 17 similarly to the one of Theorem 6, by decomposing $\mathscr{F}(\mu\tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{F}(\pi) = \mathscr{E}(\mu\tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{E}(\pi) + \mathscr{H}(\mu\tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{H}(\pi)$, for $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ and $\gamma, \tilde{\gamma} > 0$. The main difference is that we now need to handle carefully the proximal step in the first term of the decomposition. To this end, we decompose the potential energy functional according to the decomposition of U, $\mathscr{E} = \mathscr{E}_1 + \mathscr{E}_2$ where for all $\mu \in \mathcal{P}_2(\mathbb{R}^d)$,

$$\mathscr{E}_1(\mu) = \int_{\mathbb{R}^d} U_1 \mathrm{d}\mu(x) \ , \qquad \mathscr{E}_2(\mu) = \int_{\mathbb{R}^d} U_2 \mathrm{d}\mu(x) \ , \tag{42}$$

and consider

$$\mathscr{F}(\mu\tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{F}(\pi) = \mathscr{E}_1(\mu\tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{E}_1(\mu\tilde{S}_\gamma^2\tilde{S}_{\tilde{\gamma}}^1) + \mathscr{E}_1(\mu\tilde{S}_\gamma^2\tilde{S}_{\tilde{\gamma}}^1) - \mathscr{E}_1(\pi) + \mathscr{E}_2(\mu\tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{E}_2(\pi) + \mathscr{H}(\mu\tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{H}(\pi) \ . \tag{43}$$

The first and last terms in the right hand side will be controlled using Lemma 3 and Lemma 5. In the next lemmas, we bound the other terms separately.

Lemma 28 *Assume* $\boldsymbol{A}5(m)$*, for* $m \geq 0$*. For all* $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$ *and* $\gamma \in (0, L^{-1}]$*,*

$$2\gamma\{\mathscr{E}_1(\mu\tilde{S}_\gamma^1) - \mathscr{E}_1(\nu)\} \leq (1 - m\gamma)W_2^2(\mu,\nu) - W_2^2(\mu\tilde{S}_\gamma^1, \nu) - \gamma^2(1 - \gamma L) \int_{\mathbb{R}^d} \|\nabla U_1(x)\|^2 \, \mathrm{d}\mu(x) + \gamma^2(1 + \gamma L)\upsilon_1(\mu) \ ,$$

where $\mathscr{E}_1, \tilde{S}_\gamma^1$ *is defined by* (42)-(34) *and* $\upsilon_1(\mu)$ *by* (36)*.*

Proof Let $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$ and $\gamma > 0$. Since U_1 satisfies $\mathbf{A}2$ by (Nesterov, 2004, Lemma 1.2.3), for all $x, \tilde{x} \in \mathbb{R}^d$, we have $|U_1(\tilde{x}) - U_1(x) - \langle \nabla U_1(x), \tilde{x} - x \rangle| \leq (L/2) \|\tilde{x} - x\|^2$. Using that U_1 is m-strongly convex by $\mathbf{A}5(m)$, for all $x, y \in \mathbb{R}^d$, $z \in \mathsf{Z}$, we get

$$\begin{aligned} U_1(x - \gamma\tilde{\Theta}_1(x,z)) - U_1(y) &= U_1(x - \gamma\tilde{\Theta}_1(x,z)) - U_1(x) + U_1(x) - U_1(y) \\ &\leq -\gamma \left\langle \nabla U_1(x), \tilde{\Theta}_1(x,z) \right\rangle + (L\gamma^2/2) \left\| \tilde{\Theta}_1(x,z) \right\|^2 + \langle \nabla U_1(x), x - y \rangle - (m/2) \|y - x\|^2 \ . \end{aligned}$$

Then multiplying both sides by γ, we obtain

$$\begin{aligned} 2\gamma \left\{ U_1(x - \gamma\tilde{\Theta}_1(x,z)) - U_1(y) \right\} &\leq (1 - m\gamma) \|x - y\|^2 - \left\| x - \gamma\tilde{\Theta}_1(x,z) - y \right\|^2 \\ &- 2\gamma^2 \left\langle \nabla U_1(x), \tilde{\Theta}_1(x,z) \right\rangle + \gamma^2(1 + \gamma L) \left\| \tilde{\Theta}_1(x,z) \right\|^2 + 2\gamma \left\langle \nabla U_1(x) - \tilde{\Theta}_1(x,z), x - y \right\rangle \ . \end{aligned} \tag{44}$$

Let now (X, Y) be an optimal coupling between μ and ν and Z with distribution η independent of (X, Y). Note that $\mathbf{A}5$ implies that $\mathbb{E}[\tilde{\Theta}_1(X, Z)|(X, Y)] = \nabla U_1(X)$. Then by definition and (44), we get

$$\begin{aligned} 2\gamma \left\{ \mathscr{E}(\mu\tilde{S}_\gamma^1) - \mathscr{E}(\nu) \right\} &\leq (1 - m\gamma)W_2^2(\mu,\nu) - \mathbb{E}\left[\left\| X - \gamma\tilde{\Theta}_1(X) - Y \right\|^2 \right] \\ &\quad - 2\gamma^2 \mathbb{E}\left[\|\nabla U_1(X)\|^2 \right] + \gamma^2(1 + \gamma L)\mathbb{E}\left[\left\| \tilde{\Theta}_1(X) \right\|^2 \right] \\ &\leq (1 - m\gamma)W_2^2(\mu,\nu) - \mathbb{E}\left[\left\| X - \gamma\tilde{\Theta}_1(X) - Y \right\|^2 \right] \\ &\quad - \gamma^2(1 - \gamma L)\mathbb{E}\left[\|\nabla U_1(X)\|^2 \right] + \gamma^2(1 + \gamma L)\upsilon_1(\mu) \ . \end{aligned}$$

Using that $W_2^2(\mu\tilde{S}_\gamma^1, \nu) \leq \mathbb{E}[\|X - \gamma\tilde{\Theta}_1(X) - Y\|^2]$ concludes the proof. ∎

Lemma 29 *Assume* $\boldsymbol{A}5(m)$ *for* $m \geq 0$*. For all* $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$ *and* $\gamma > 0$*, we have*

$$2\gamma \{\mathscr{E}_2(\mu) - \mathscr{E}_2(\nu)\} \leq W_2^2(\mu,\nu) - W_2^2(\mu\tilde{S}_\gamma^2, \nu) + 2\gamma^2 M_2^2 \ ,$$

where $\mathscr{E}_2, \tilde{S}_\gamma^2$ *are defined by* (42) *and* (34) *respectively.*

Proof Let $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$ and $\gamma > 0$. First we bound for any $x, y \in \mathbb{R}^d$, $U_2(x) - U_2(y)$ using the decomposition $U_2(x) - U_2(\mathrm{prox}^\gamma_{U_2}(x)) + U_2(\mathrm{prox}^\gamma_{U_2}(x)) - U_2(y)$. For any $x, y \in \mathbb{R}^d$, we have using that $\gamma^{-1}(x - \mathrm{prox}^\gamma_{U_2}(x)) \in \partial U_2(\mathrm{prox}^\gamma_{U_2}(x))$ (see Rockafellar and Wets (1998, Chapter 1 Section G)), where ∂U_2 is the subdifferential of U_2 defined by (24),

$$U_2(\mathrm{prox}^\gamma_{U_2}(x)) - U_2(y) \leq \gamma^{-1} \left\langle x - \mathrm{prox}^\gamma_{U_2}(x), \mathrm{prox}^\gamma_{U_2}(x) - y \right\rangle .$$

Since $\|x - y\|^2 = \|x - \mathrm{prox}^\gamma_{U_2}(x)\|^2 + \|\mathrm{prox}^\gamma_{U_2}(x) - y\|^2 + 2\langle x - \mathrm{prox}^\gamma_{U_2}(x), \mathrm{prox}^\gamma_{U_2}(x) - y\rangle$, we get for all $x, y \in \mathbb{R}^d$,

$$U_2(\mathrm{prox}^\gamma_{U_2}(x)) - U_2(y) \leq (2\gamma)^{-1}(\|x - y\|^2 - \|\mathrm{prox}^\gamma_{U_2}(x) - y\|^2) . \tag{45}$$

Second, since U_2 is M_2-Lipschitz, we get for any $x \in \mathbb{R}^d$, $|U_2(x) - U_2(\mathrm{prox}^\gamma_{U_2}(x))| \leq M_2\|x - \mathrm{prox}^\gamma_{U_2}(x)\|$. Then using that $\gamma^{-1}(x - \mathrm{prox}^\gamma_{U_2}(x)) \in \partial U_2(\mathrm{prox}^\gamma_{U_2}(x))$, and for any $v \in \partial U_2(\mathrm{prox}^\gamma_{U_2}(x))$, since U_2 is M_2-Lipschitz, $\|v\| \leq M_2$, we obtain $|U_2(x) - U_2(\mathrm{prox}^\gamma_{U_2}(x))| \leq \gamma M_2^2$. Combining this result and (45) yields for any $x, y \in \mathbb{R}^d$

$$2\gamma \{U_2(x) - U_2(y)\} \leq \|x - y\|^2 - \|\mathrm{prox}^\gamma_{U_2}(x) - y\|^2 + 2\gamma^2 M_2^2 .$$

Let (X, Y) be an optimal coupling for μ and ν. The proof then follows from using the inequality above for (X, Y), taking the expectation and because $W_2^2(\mu \tilde{S}^1_\gamma, \nu) \leq \|\mathrm{prox}^\gamma_{U_2}(X) - Y\|^2$. ∎

Lemma 30 *Assume* ***A*** *5(m), for $m \geq 0$. For all $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$ and $\gamma, \tilde{\gamma} \in \left(0, L^{-1}\right]$,*

$$2\tilde{\gamma}\{\mathscr{F}(\mu_0 \tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{F}(\pi)\} \leq (1 - m\tilde{\gamma}) W_2^2(\mu_0 \tilde{S}^2_\gamma, \pi) - W_2^2(\mu_0 \tilde{R}_{\gamma,\tilde{\gamma}} \tilde{S}^2_{\tilde{\gamma}}, \pi) + \tilde{\gamma}^2 \{2Ld + (1 + \tilde{\gamma}L) \upsilon_1(\mu_0 \tilde{S}^2_\gamma) + 2M_2^2\} ,$$

where $\mathscr{F}$, $\tilde{R}_{\gamma,\tilde{\gamma}}$ and $\tilde{S}^2_\gamma$ are defined by (7)-(33)-(34) *respectively.*

Proof Let $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$ and $\gamma, \tilde{\gamma} \in \left(0, L^{-1}\right]$. By Lemma 3 and since $\tilde{R}_{\gamma,\tilde{\gamma}} = \tilde{S}^2_\gamma \tilde{S}^1_{\tilde{\gamma}} T_{\tilde{\gamma}}$, we have

$$\mathscr{E}_1(\mu_0 \tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{E}_1(\mu_0 \tilde{S}^2_\gamma \tilde{S}^1_{\tilde{\gamma}}) \leq 2Ld\tilde{\gamma} . \tag{46}$$

By Lemma 28 since $\tilde{\gamma} \leq 1/L$,

$$2\tilde{\gamma}\{\mathscr{E}_1(\mu_0 \tilde{S}^2_\gamma \tilde{S}^1_{\tilde{\gamma}}) - \mathscr{E}_1(\pi)\} \leq (1 - \tilde{\gamma}m) W_2^2(\mu_0 \tilde{S}^2_\gamma, \pi) - W_2^2(\mu_0 \tilde{S}^2_\gamma \tilde{S}^1_{\tilde{\gamma}}, \pi) + \tilde{\gamma}^2(1 + \tilde{\gamma}L) \upsilon_1(\mu_0 \tilde{S}^2_\gamma) . \tag{47}$$

By Lemma 29, we have

$$2\tilde{\gamma}\{\mathscr{E}_2(\mu_0 \tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{E}_2(\pi)\} \leq W_2^2(\mu_0 \tilde{R}_\gamma, \pi) - W_2^2(\mu_0 \tilde{R}_{\gamma,\tilde{\gamma}} \tilde{S}^2_{\tilde{\gamma}}, \pi) + 2\tilde{\gamma}^2 M_2^2 . \tag{48}$$

Finally by Lemma 5, we have

$$2\tilde{\gamma}\{\mathscr{H}(\mu_0 \tilde{R}_{\gamma,\tilde{\gamma}}) - \mathscr{H}(\pi)\} \leq W_2^2(\mu_0 \tilde{S}^2_\gamma \tilde{S}^1_{\tilde{\gamma}}, \pi) - W_2^2(\mu_0 \tilde{R}_{\gamma,\tilde{\gamma}}, \pi) . \tag{49}$$

Combining (46)-(47)-(48)-(49) in (43) concludes the proof. ∎

7.4.1. Proof of Theorem 17

Using the convexity of KL divergence and Lemma 30, we obtain

$$\begin{aligned}
\mathrm{KL}\left(\tilde{\nu}_n^N \middle| \pi\right) &\leq \Lambda_{N,N+n}^{-1} \sum_{k=N+1}^{N+n} \lambda_k \,\mathrm{KL}\left(\mu_0 \tilde{Q}^k_\gamma \middle| \pi\right) \\
&\leq (2\Lambda_{N,N+n})^{-1} \left[\frac{(1 - m\gamma_{N+2})\lambda_{N+1}}{\gamma_{N+2}} W_2^2\left(\mu_0 \tilde{Q}^N_\gamma \tilde{S}^2_{\gamma_{N+1}}, \pi\right) \right. \\
&\quad - \frac{\lambda_{N+n}}{\gamma_{N+n+1}} W_2^2\left(\mu_0 \tilde{Q}^{N+n}_\gamma \tilde{S}^2_{\gamma_{N+n+1}}, \pi\right) \\
&\quad + \sum_{k=N+1}^{N+n-1} \left\{ \frac{(1 - m\gamma_{k+2})\lambda_{k+1}}{\gamma_{k+2}} - \frac{\lambda_k}{\gamma_{k+1}} \right\} W_2^2\left(\mu_0 Q^{k-1}_\gamma \tilde{S}^2_{\gamma_{k+1}}, \pi\right) \\
&\quad \left. + \sum_{k=N+1}^{N+n} \lambda_k \gamma_{k+1} \{2Ld + (1 + \gamma_{k+1}L) \upsilon_1(\mu_0 Q^k_\gamma \tilde{S}^2_{\gamma_k}) + 2M_2^2\} \right] .
\end{aligned}$$

We get the thesis using that $\lambda_{k+1}(1 - m\gamma_{k+2})/\gamma_{k+2} \leq \lambda_k/\gamma_{k+1}$ for all $k \in \mathbb{N}$.

7.4.2. Proof of Corollary 20

Using Theorem 17 we get:

$$\mathrm{KL}\left(\tilde{\nu}_n^N \middle| \pi\right) \leq W_2^2\left(\mu_0 \tilde{Q}^N_\gamma \tilde{S}^2_\gamma, \pi\right) \Big/ (2\gamma n) + \gamma(Ld + M_2^2) + \frac{\gamma}{2n} \sum_{k=N+1}^{N+n} (1 + \gamma L) \upsilon_1(\mu_0 Q^k_\gamma \tilde{S}^2_\gamma)$$

and using Proposition 19 we obtain:

$$\begin{aligned}
2\gamma(\tilde{L}^{-1} - \gamma) \left(\sum_{k=N+1}^{N+n} \upsilon_1(\mu_0 Q^k_\gamma \tilde{S}^2_\gamma) \right) &\leq \int_{\mathbb{R}^d} \|y - x^\star\|^2 \,\mathrm{d}\mu_0 Q^{N+1}_\gamma \tilde{S}^2_\gamma(y) \\
&\quad - \int_{\mathbb{R}^d} \|y - x^\star\|^2 \,\mathrm{d}\mu_0 Q^{N+n+1}_\gamma \tilde{S}^2_\gamma(y) + 2n\gamma^2 \upsilon_1(\delta_{x^\star}) + 2n\gamma d ,
\end{aligned}$$

Combining the two inequalities above finishes the proof of the first part of Corollary 20. For the second part, observe that since $\gamma_\varepsilon \leq L^{-1}$ and $\gamma_\varepsilon \leq (2\tilde{L})^{-1}$ we have $(1 + \gamma L)(2(\tilde{L}^{-1} - \gamma))^{-1} \leq 2\tilde{L}$. Therefore from definition of γ_ε we have $\gamma_\varepsilon(Ld + M_2^2 + 2\tilde{L}d) \leq \varepsilon/4$, as well as $\gamma_\varepsilon^2 2\tilde{L} \upsilon_1(\delta_{x^\star}) \leq \varepsilon/4$. On the other hand, from definition of n_ε we have $W_2^2(\mu_0 \tilde{S}^2_{\gamma_\varepsilon}, \pi)/(2n_\varepsilon\gamma_\varepsilon) \leq \varepsilon/4$ as well as $2\tilde{L}(2n_\varepsilon)^{-1} \int_{\mathbb{R}^d} \|x - x^\star\|^2 \,\mathrm{d}\mu_0 \tilde{S}^2_\gamma(y) \leq \varepsilon/4$. Combining this four bounds we get the thesis.

7.4.3. Proof of Theorem 21

Using Lemma 30 and since the KL divergence is non-negative, we get for all $k \in \{1, \ldots, n\}$,

$$W_2^2\left(\mu_0 \tilde{Q}^k_\gamma \tilde{S}^2_{\gamma_{k+1}}, \pi\right) \leq (1 - m\gamma_{k+1}) W_2^2\left(\mu_0 Q^{k-1}_\gamma \tilde{S}^2_{\gamma_k}, \pi\right) + \gamma^2_{k+1} \{2Ld + (1 + \gamma_{k+1}L) \upsilon_1(\mu_0 Q^{k-1}_\gamma \tilde{S}^2_{\gamma_k}) + 2M_2^2\} .$$

The proof then follows from a direct induction.

7.4.4. Proof of Proposition 23

Let $\gamma > 0$, $x \in \mathbb{R}^d$ and consider $\tilde{X}_1 = \mathrm{prox}^{\gamma}_{U_2}\left\{x - \gamma\tilde{\Theta}_1(x, Z_1) + \sqrt{2\gamma}G_1\right\}$, where Z_1 and G_1 are two independent random variables, Z_1 has distribution $\tilde{\eta}_1$ and G_1 is a standard Gaussian random variable, so that $\tilde{X}_1$ has distribution $\tilde{S}^1_\gamma T_\gamma \tilde{S}^2_\gamma(x, \cdot)$. First by (Bauschke and Combettes, 2011, Theorem 26.2(vii)), we have that $x^\star = \mathrm{prox}^{\gamma}_{U_2}(x^\star - \gamma\nabla U_1(x^\star))$ and by (Bauschke and Combettes, 2011, Proposition 12.27), the proximal is non-expansive, for all $x, y \in \mathbb{R}^d$, $\|\mathrm{prox}^{\gamma}_{U_2}(x) - \mathrm{prox}^{\gamma}_{U_2}(y)\| \le \|x - y\|$. Using these two results and the fact that $\tilde{\Theta}_1$ satisfies **A**4, we have

$$
\begin{aligned}
\mathbb{E}\left[\left\|\tilde{X}_1 - x^\star\right\|^2\right] &= \mathbb{E}\left[\left\|\mathrm{prox}^{\gamma}_{U_2}\left\{x - \gamma\tilde{\Theta}_1(x, Z_1) + \sqrt{2\gamma}G_1\right\} - \mathrm{prox}^{\gamma}_{U_2}\{x^\star - \gamma\nabla U_1(x^\star)\}\right\|^2\right] \\
&\le \mathbb{E}\left[\left\|\left(x - \gamma\tilde{\Theta}_1(x, Z_1) + \sqrt{2\gamma}G_1\right) - (x^\star - \gamma\nabla U_1(x^\star))\right\|^2\right] \\
&\le \|x - x^\star\|^2 \\
&\quad + \mathbb{E}\left[2\gamma\left\langle x - x^\star, \nabla U_1(x^\star) - \tilde{\Theta}_1(x, Z_1)\right\rangle + \gamma^2\left\|\nabla U_1(x^\star) - \tilde{\Theta}_1(x, Z_1)\right\|^2\right] + 2\gamma d \\
&\le (1 - \tilde{m}_1\gamma)\|x - x^\star\|^2 - 2\gamma(\tilde{L}_1^{-1} - \gamma)\mathbb{E}\left[\left\|\tilde{\Theta}_1(x, Z_1) - \tilde{\Theta}_1(x^\star, Z_1)\right\|^2\right] \\
&\qquad + 2\gamma^2\mathbb{E}\left[\left\|\tilde{\Theta}_1(x^\star, Z_1) - \nabla U_1(x^\star)\right\|^2\right] + 2\gamma d\,.
\end{aligned}
$$

The proof is completed upon noting that $\upsilon_1(\delta_x) \le \mathbb{E}[\|\Theta(x, Z_1) - \Theta(x^\star, Z_1)\|^2]$.

7.4.5. Proof of Corollary 24

Using Theorem 21 we get:

$$
\begin{aligned}
W_2^2(\mu_0\tilde{R}^n_{\gamma,\gamma}\tilde{S}^2_\gamma, \pi) &\le (1 - m\gamma)^n W_2^2(\mu_0\tilde{S}^2_\gamma, \pi) \\
&\quad + \gamma^2\sum_{k=1}^{n}(1 - m\gamma)^{n-k}\left(2Ld + (1 + \gamma L)\upsilon_1(\mu_0\tilde{R}^k_{\gamma,\gamma}\tilde{S}^2_\gamma) + 2M_2^2\right) \\
&\le (1 - m\gamma)^n W_2^2(\mu_0\tilde{S}^2_\gamma, \pi) + 2(Ld + M_2)\gamma/m \\
&\quad + \gamma^2\sum_{k=1}^{n}(1 - \tilde{m}\gamma)^{n-k}(1 + \gamma L)\upsilon_1(\mu_0\tilde{R}^k_{\gamma,\gamma}\tilde{S}^2_\gamma)\,. \qquad (50)
\end{aligned}
$$

In addition, using Proposition 23 and $\gamma \le (2\tilde{L}_1)^{-1}$, we have

$$
\begin{aligned}
\gamma\tilde{L}_1^{-1}\sum_{k=1}^{n}(1 - \tilde{m}\gamma)^{n-k}\upsilon_1(\mu_0\tilde{R}^k_{\gamma,\gamma}\tilde{S}^2_\gamma) &\le 2\gamma\sum_{k=1}^{n}(1 - \tilde{m}\gamma)^{n-k}(\gamma\upsilon_1(\delta_{x^\star}) + d) \\
&\quad + \sum_{k=1}^{n}(1 - \tilde{m}\gamma)^{n-k+1}\int_{\mathbb{R}^d}\|x - x^\star\|^2\,\mathrm{d}\mu_0 R^k_{\gamma,\gamma}\tilde{S}^2_\gamma(x) \\
&\quad - \sum_{k=1}^{n}(1 - \tilde{m}\gamma)^{n-k}\int_{\mathbb{R}^d}\|x - x^\star\|^2\,\mathrm{d}\mu_0 R^{k+1}_{\gamma,\gamma}\tilde{S}^2_\gamma(x)\,.
\end{aligned}
$$

Combining this result and (50) concludes the proof of (38).

Now, for $\gamma_\varepsilon, n_\varepsilon$ as defined in the thesis of the corollary we have $\gamma_\varepsilon\Delta_1 \le \varepsilon/4$ and $\Delta_2\gamma_\varepsilon^2 \le \varepsilon/4$. Furthermore, $(1 - m\gamma_\varepsilon)^{n_\varepsilon}W_2^2(\mu_0\tilde{S}^2_{\gamma_\varepsilon}, \pi) \le \exp(-n_\varepsilon m\gamma_\varepsilon)W_2^2(\mu_0\tilde{S}^2_{\gamma_\varepsilon}, \pi) \le \varepsilon/4$, and $(1 - \gamma_\varepsilon\tilde{m})\Delta_3 \le \varepsilon/4$ similarly. Together, the above inequalities conclude the proof.

Acknowledgments

A. D. acknowledges support from Chaire BayeScale "P. Laffitte". B. M. is supported by Polish National Science Center grant no. 2015/17/D/ST1/01198.

Appendix A. Definitions and Useful Results from Theory of Gradient Flows

For a continously differentiable function $f : \mathbb{R}^d \to \mathbb{R}$, the gradient flow associated with f starting at x_0 is the solution $(x(t))_{t\in\mathbb{R}_+}$ of the ordinary differential equation: $\mathrm{d}x(t)/\mathrm{d}t = -\nabla f(x(t))$, with $x(0) = x_0$. Classical theory of gradient flows was developed for functions defined on $\mathbb{R}^d$, and later extended to functionals on Banach spaces. Main motivation for this development were connections between gradient flows in Banach spaces and some partial differential equation - for example the heat equation can be formulated as the gradient flow of $u \mapsto \int_{\mathbb{R}^d}\|\nabla u\|^2\,\mathrm{d}x$ (which is called the Dirichlet energy) on $\mathrm{L}^2(\mathbb{R}^d) = \{u : \mathbb{R}^d \to \mathbb{R} : u \text{ is measurable and } \int_{\mathbb{R}^d}\|u\|^2\,\mathrm{d}x < +\infty\}$. Similarly, widespread interest in the theory of gradient flows in metric spaces started with the work of Jordan et al. (1998), which showed that gradient flow of the free energy functional defined in (7) in the space $(\mathcal{P}_2(\mathbb{R}^d), W_2)$ is a measure valued solution of the Fokker-Planck equation.

For a brief overview of the theory of gradient flows in Euclidean and metric spaces, with a focus on Wasserstein spaces, we refer to (Santambrogio, 2017). For a detailed introduction to the theory of gradient flows in metric spaces we refer the reader to (Ambrosio et al., 2008). Below, we only introduce definitions and results from the theory of gradient flows in the space of probability measures, which are relevant to our work.

Let $I \subset \mathbb{R}$ be an open interval of $\mathbb{R}$ and $(\mu_t)_{t\in I}$ be a curve on $\mathcal{P}_2(\mathbb{R}^d)$, *i.e.* a family of probability measures belonging to $\mathcal{P}_2(\mathbb{R}^d)$. $(\mu_t)_{t\in I}$ is said to be absolutely continuous if there exists $\ell \in \mathrm{L}^1(I)$ such that for all $s, t \in I$, $s \le t$, $W_2(\mu_s, \mu_t) \le \int_s^t |\ell|\,(u)\mathrm{d}u$. Denote by $\mathrm{AC}(I)$ the set of absolutely continuous curves on I and

$$
\mathrm{AC}_{\mathrm{loc}}(\mathbb{R}^*_+) = \left\{(\mu_t)_{t\ge 0} \;:\; (\mu_t)_{t\in I} \in \mathrm{AC}(I) \text{ for any open interval } I \subset \mathbb{R}^*_+\right\}\,.
$$

Note that if $(\mu_t)_{t\in I} \in \mathrm{AC}(I)$, then for any $\nu \in \mathcal{P}_2(\mathbb{R}^d)$, $t \mapsto W_2(\nu, \mu_t)$ is absolutely continuous on I (as a curve from I to $\mathbb{R}_+$). Therefore by (Nielsen, 1997, Theorem 20.8) and (Mitrovic and Zubrinic, 1997, Exercise 4, p.45), $t \mapsto W_2(\nu, \mu_t)$ has derivative for the almost all $t \in I$ and there exists $\delta : I \to \mathbb{R}$ satisfying

$$
\int_I |\delta|\,(u)\mathrm{d}u < +\infty \text{ and } W_2^2(\nu, \mu_t) - W_2^2(\nu, \mu_s) = \int_s^t \delta(u)\mathrm{d}u\,, \text{ for all } s, t \in I \qquad (51)
$$

Let $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$. A constant speed geodesic $(\lambda_t)_{t\in[0,1]}$ between μ and ν is a curve in $\mathcal{P}_2(\mathbb{R}^d)$ such that $\lambda_0 = \mu$, $\lambda_1 = \nu$ and for all for all $s, t \in [0, 1]$, $W_2(\lambda_s, \lambda_t) = |t - s|\,W_2(\mu, \nu)$.

Note that by the triangle inequality, this definition is equivalent to the following: for all $s,t \in [0,1]$, $W_2(\lambda_s,\lambda_t) \le |t-s|\, W_2(\mu,\nu)$. Indeed by the triangle inequality and the assumption $W_2(\lambda_s,\lambda_t) \le |t-s|\, W_2(\mu,\nu)$, we have for all $s,t \in [0,1]$, $s<t$,

$$W_2(\mu,\nu) \le W_2(\mu,\lambda_t) + W_2(\lambda_t,\lambda_s) + W_2(\lambda_s,\nu) \le W_2(\mu,\nu) \; .$$

Therefore the first inequality is in fact an equality, and therefore using again the assumption for $W_2(\mu,\lambda_t)$ and $W_2(\lambda_s,\nu)$ concludes the proof. By definition of the Wasserstein distance of order 2, a constant speed geodesic $(\lambda_t)_{t\in[0,1]}$ between μ and ν is given for all $t \in [0,1]$ by $\lambda_t = (t\,\mathrm{proj}_1 + (1-t)\,\mathrm{proj}_2)_\sharp \zeta$ where ζ is an optimal transport plan between μ and ν and $\mathrm{proj}_1, \mathrm{proj}_2 : \mathbb{R}^{2d} \to \mathbb{R}^d$ are the projections on the first and the last d components respectively.

Let $\mathscr{S} : \mathcal{P}_2(\mathbb{R}^d) \to (-\infty, +\infty]$. The functional $\mathscr{S}$ is said to be lower semi-continuous if for all $M \in \mathbb{R}$, $\{\mathscr{S} \le M\}$ is a closed set of $\mathcal{P}_2(\mathbb{R}^d)$ and m-geodesically convex for $m \ge 0$ if for any $\mu,\nu \in \mathcal{P}_2(\mathbb{R}^d)$ there exists a constant speed geodesic $(\lambda_t)_{t\in[0,1]}$ between μ and ν such that for all $t \in [0,1]$

$$\mathscr{S}(\lambda_t) \le t\mathscr{S}(\mu) + (1-t)\mathscr{S}(\nu) - t(1-t)(m/2)W_2^2(\mu,\nu) \; .$$

If $m = 0$, $\mathscr{S}$ will be simply said geodesically convex.

A curve $(\mu_t)_{t>0} \in \mathrm{AC}_{\mathrm{loc}}(\mathbb{R}_+^*)$ is said to be a gradient flow for the lower semi-continuous and m-geodesically convex function $\mathscr{S} : \mathcal{P}_2(\mathbb{R}^d) \to (-\infty, +\infty]$ if for all $\nu \in \mathcal{P}_2(\mathbb{R}^d)$, $\mathscr{S}(\nu) < +\infty$, and for almost all $t \in \mathbb{R}_+^*$,

$$(1/2)\delta_t + (m/2)W_2^2(\mu_t,\nu) \le \mathscr{S}(\nu) - \mathscr{S}(\mu_t) \; ,$$

where $\delta : \mathbb{R}_+^* \to \mathbb{R}$ satisfies (51) for all open interval of $\mathbb{R}_+^*$. We say that $(\mu_t)_{t\in\mathbb{R}_+^*}$ starts at μ if $\lim_{t\to 0} W_2(\mu_t,\mu) = 0$ and then set $\mu_0 = \mu$. By (Ambrosio et al., 2008, Theorem 11.1.4), there exists at most one gradient flow associated with $\mathscr{S}$.

Consider the functional $\tilde{\mathscr{F}} : \mathcal{P}_2(\mathbb{R}^d) \to (-\infty, +\infty]$ given by $\tilde{\mathscr{F}} = \mathscr{H} + \tilde{\mathscr{E}}$ where $\mathscr{H}$ is defined by (8) and $\tilde{\mathscr{E}}$ for all $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ by

$$\tilde{\mathscr{E}}(\mu) = \int_{\mathbb{R}^d} V(x)\mathrm{d}\mu(x) \; ,$$

where $V : \mathbb{R}^d \to (-\infty, +\infty]$ is a convex lower-semicontinuous function (for all $M \ge 0$, $\{V \le M\}$ is closed subset of $\mathbb{R}^d$) with $\{V < +\infty\} \neq \emptyset$ and the interior of this set is non empty as well. By (Ambrosio et al., 2008, Proposition 9.3.2, Theorem 9.4.12), $\tilde{\mathscr{F}}$ is geodesically convex and (Ambrosio et al., 2008, Theorem 11.2.8,Theorem 11.1.4) shows that there exists the unique gradient flow $(\mu_t)_{t\ge 0}$ starting at $\mu \in \mathcal{P}_2(\mathbb{R}^d)$ and this curve is the unique solution of the Fokker-Plank equation (in the sense of distributions) :

$$\frac{\partial \mu_t}{\partial t} = \mathrm{div}(\nabla \mu_t^x + \mu_t^x \nabla V(x)) \; ,$$

i.e. for all $\phi \in C_c^\infty(\mathbb{R}^d)$ and $t > 0$,

$$\frac{\partial}{\partial t}\int_{\mathbb{R}^d} \phi(y)\mu_t(\mathrm{d}y) = \int_{\mathbb{R}^d} \mathcal{A}\phi(y)\,\mu_t(\mathrm{d}y) \; .$$

In addition for all $t > 0$, μ_t is absolutely continuous with respect to the Lebesgue measure. In particular for $V = 0$, we get the following result.

Theorem 31 *For all $\mu \in \mathcal{P}_2(\mathbb{R}^d)$, there exists a unique solution of the Fokker-Plank equation (in the sense of distributions) :*

$$\frac{\partial \mu_t}{\partial t} = \Delta \mu_t \; .$$

In addition $(\mu_t)_{t\ge 0} \in \mathrm{AC}(\mathbb{R}_+^*)$ and satisfies for almost all $t \in \mathbb{R}_+^*$,

$$\delta_t/2 \le \mathscr{H}(\nu) - \mathscr{H}(\mu_t) \; ,$$

where δ_t is given in (51).

Appendix B. On the Second Order Moment of Log-Concave Measures

A7 *There exist $\eta > 0$, $M_\eta \ge 0$ such that for all $x \in \mathbb{R}^d$, $x \notin \mathrm{B}(0, M_\eta)$,*

$$U(x) - U(x^\star) \ge \eta \left\| x - x^\star \right\| \; .$$

In this section, we give some bounds on to deal with the distance of the initial condition of the algorithms from π in W_2.

Proposition 32 *Assume **A**1(0) and **A**7. Then, we have*

$$\int_{\mathbb{R}^d} \left\| x - x^\star \right\|^2 \mathrm{d}\pi(x) \le 2\eta^{-2} d(1+d) + M_\eta^2 \; .$$

Proof Note that under **A**7, we have

$$\begin{aligned} &\int_{\mathbb{R}^d} \left\| x - x^\star \right\|^2 \mathrm{d}\pi(x) \le \eta^{-2} \int_{\mathbb{R}^d} |U(x) - U(x^\star)|^2 \,\mathrm{d}\pi(x) + M_\eta^2 \\ &\le 2\eta^{-2} \int_{\mathbb{R}^d} |U(x) + \log(\mathrm{Z}) + \mathscr{H}(\pi)|^2 \,\mathrm{d}\pi(x) + 2\eta^{-2} \left| -\mathscr{H}(\pi) - \log(\mathrm{Z}) - U(x^\star) \right|^2 + M_\eta^2 \; . \end{aligned} \tag{52}$$

where $\mathscr{H}$ is defined by (8) and $\mathrm{Z} = \int_{\mathbb{R}^d} \mathrm{e}^{-U(y)}\mathrm{d}y$. Then, by (Bobkov and Madiman, 2011, Proposition I.2), $|-\mathscr{H}(\pi) - \log(\mathrm{Z}) - U(x^\star)| \le d$ and by (Fradelizi et al., 2016, Theorem 2.3), (see also Nguyen (2013) and Wang (2014)), $\int_{\mathbb{R}^d} |U(x) + \log(\mathrm{Z}) + \mathscr{H}(\pi)|^2 \,\mathrm{d}\pi(x) \le d$. Combining these two results in (52) concludes the proof. ■

References

L. Ambrosio, N. Gigli, and G. Savaré. *Gradient Flows: In Metric Spaces and in the Space of Probability Measures (Lectures in Mathematics. ETH Zürich)*. Birkhäuser, 2008. ISBN 978-3-7643-8721-1.

L. Ambrosio, G. Savaré, and L. Zambotti. Existence and stability for Fokker–Planck equations with log-concave reference measure. *Probability Theory and Related Fields*, 145(3): 517–564, 2009. doi: 10.1007/s00440-008-0177-3.

C. Andrieu, N. De Freitas, A. Doucet, and M. I Jordan. An Introduction to MCMC for Machine Learning. *Machine Learning*, 50(1/2):5–43, 2003. doi: 10.1023/a:1020281327116.

H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer Publishing Company, Incorporated, 1st edition, 2011. ISBN 1441994661, 9781441994660.

A. Beck and M. Teboulle. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. doi: 10.1137/080716542.

E. Bernton. Langevin Monte Carlo and JKO splitting. *arXiv preprint arXiv:1802.08671*, 2018.

S. Bobkov and M. Madiman. The Entropy Per Coordinate of a Random Vector is Highly Constrained Under Convexity Conditions. *IEEE Transactions on Information Theory*, 57(8):4940–4954, 2011.

F. Bolley, I. Gentil, and A. Guillin. Convergence to equilibrium in Wasserstein distance for Fokker-Planck equations. *Journal of Functional Analysis*, 263(8):2430–2457, 2012. doi: 10.1016/j.jfa.2012.07.007.

S. Brazitikos, A. Giannopoulos, P. Valettas, and B.-H. Vritsiou. *Geometry of isotropic convex bodies*, volume 196. American Mathematical Society Providence, 2014.

S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng. Handbook of Markov chain Monte Carlo, 2011.

X. Cheng and P. Bartlett. Convergence of Langevin MCMC in KL-divergence. *arXiv preprint arXiv:1705.09048*, 2017.

T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, second edition, 2006. ISBN 978-0-471-24195-9; 0-471-24195-4.

A. S. Dalalyan. Theoretical guarantees for approximate sampling from smooth and log-concave densities. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2016. doi: 10.1111/rssb.12183.

A. S Dalalyan and A. G. Karagulyan. User-friendly guarantees for the Langevin Monte Carlo with inaccurate gradient. *arXiv preprint arXiv:1710.00095*, 2017.

D. Dua and K.T. Efi. UCI Machine Learning Repository, 2017. URL `http://archive.ics.uci.edu/ml`.

A. Durmus and É. Moulines. Supplement to "High-dimensional Bayesian inference via the Unadjusted Langevin Algorithm", 2015. https://hal.inria.fr/hal-01176084/.

A. Durmus and É. Moulines. High-dimensional Bayesian inference via the Unadjusted Langevin Algorithm. 2016.

A. Durmus and É. Moulines. Nonasymptotic convergence analysis for the unadjusted Langevin algorithm. *The Annals of Applied Probability*, 27(3):1551–1587, 06 2017. doi: 10.1214/16-AAP1238.

A. Durmus, É. Moulines, and M. Pereyra. Efficient Bayesian Computation by Proximal Markov Chain Monte Carlo: When Langevin Meets Moreau. *SIAM Journal on Imaging Sciences*, 11(1):473–506, 2018. doi: 10.1137/16m1108340.

S.N. Ethier and T.G. Kurtz. *Markov processes: characterization and convergence*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley, 1986. URL `http://books.google.fr/books?id=BAWnAAAAIAAJ`.

M. Fradelizi, M. Madiman, and L. Wang. Optimal concentration of information content for log-concave densities. In *High dimensional probability VII*, pages 45–60. Springer, 2016.

A. Gelman, J. B Carlin, H. S. Stern, and D. B. Rubin. *Bayesian data analysis*, volume 2. Chapman & Hall/CRC Boca Raton, FL, USA, 2014.

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

A. Genkin, D. D Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304, 2007.

C. R. Givens and R. M. Shortt. A class of Wasserstein metrics for probability distributions. *Michigan Mathematical Journal*, 31(2):231–240, 1984.

R. B. Gramacy and N. G. Polson. Simulation-based Regularized Logistic Regression. *Bayesian Analysis*, 7(3):567–590, 09 2012. doi: 10.1214/12-BA719.

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. doi: 10.1093/biomet/57.1.97.

C. C. Holmes and L. Held. Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian Analysis*, 1(1):145–168, 03 2006. doi: 10.1214/06-BA105.

R. Jordan, D. Kinderlehrer, and F. Otto. The variational formulation of the Fokker-Planck equation. *SIAM Journal on Mathematical Analysis*, 29(1):1–17, 1998.

I. Karatzas and S.E. Shreve. *Brownian Motion and Stochastic Calculus*. Graduate Texts in Mathematics. Springer New York, 1991. ISBN 9780387976556.

W. Krauth. *Statistical mechanics: algorithms and computations*, volume 13. OUP Oxford, 2006.

D. Lamberton and G. Pagès. Recursive computation of the invariant distribution of a diffusion: the case of a weakly mean reverting drift. *Stochastics and Dynamics*, 3(4):435–451, 2003. doi: 10.1142/S0219493703000838.

V. Lemaire. *Estimation de la mesure invariante d'un processus de diffusion*. PhD thesis, Université Paris-Est, 2005.

Q. Li and N. Lin. The Bayesian elastic net. *Bayesian Analysis*, 5(1):151–170, 2010. doi: 10.1214/10-BA506.

B. Martinet. Régularisation d'inéquations variationnelles par approximations successives. *Rev. Française Informat. Recherche Opérationnelle*, 4:154–158, 1970.

J. C. Mattingly, A. M. Stuart, and D. J. Higham. Ergodicity for SDEs and approximations: locally Lipschitz vector fields and degenerate noise. *Stochastic Processes and their Applications*, 101(2):185–232, 2002. doi: 10.1016/S0304-4149(02)00150-3.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 23(2): 1087–1092, 1953.

S. Meyn and R. Tweedie. *Markov Chains and Stochastic Stability*. Cambridge University Press, New York, NY, USA, 2nd edition, 2009. ISBN 0521731828, 9780521731829.

D. Mitrovic and D. Zubrinic. *Fundamentals of applied functional analysis*, volume 91. CRC Press, 1997.

T. Nagapetyan, A. B Duncan, L. Hasenclever, S. J. Vollmer, L. Szpruch, and K. C. Zygalakis. The true cost of stochastic gradient langevin dynamics. *arXiv preprint arXiv:1706.02692*, 2017.

Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Applied Optimization. Springer, 2004. ISBN 9781402075537.

V.H. Nguyen. *Inégalités Fonctionnelles Et Convexité*. 2013.

O.A. Nielsen. *An Introduction to Integration and Measure Theory*. Wiley-Interscience and Canadian Mathematics Series of Monographs and Texts. Wiley, 1997. ISBN 9780471595182.

N. Parikh and S. Boyd. *Proximal Algorithms*. Foundations and Trends(r) in Optimization. Now Publishers, 2013. ISBN 9781601987167.

G. Parisi. Correlation functions and computer simulations. *Nuclear Physics B*, 180:378–384, 1981.

M. Y. Park and T. Hastie. L_1-regularization path algorithm for generalized linear models. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, 69(4):659–677, 2007. doi: 10.1111/j.1467-9868.2007.00607.x.

M. Pereyra. Proximal Markov chain Monte Carlo algorithms. *Statistics and Computing*, pages 1–16, 2015. doi: 10.1007/s11222-015-9567-4.

M. Raginsky, A. Rakhlin, and M. Telgarsky. Non-convex learning via stochastic gradient langevin dynamics: a nonasymptotic analysis. *arXiv preprint arXiv:1702.03849*, 2017.

G. O. Roberts and J. S. Rosenthal. Optimal scaling of discrete approximations to langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60 (1):255–268, feb 1998. doi: 10.1111/1467-9868.00123.

G. O. Roberts and R. L. Tweedie. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996. doi: 10.2307/3318418.

R. T. Rockafellar and R. J.-B. Wets. *Variational analysis*, volume 317 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1998. ISBN 3-540-62772-3. doi: 10.1007/978-3-642-02431-3.

T. Rockafeller. Monotone operators and the proximal point algorithm. *SIAM J. Control Optimization*, 14:877–898, 1976.

P. J. Rossky, J. D. Doll, and H. L. Friedman. Brownian dynamics as smart monte carlo simulation. *The Journal of Chemical Physics*, 69(10):4628–4633, nov 1978. doi: 10.1063/1.436415.

F. Santambrogio. {Euclidean, metric, and Wasserstein} gradient flows: an overview. *Bulletin of Mathematical Sciences*, 7(1):87–154, mar 2017. doi: 10.1007/s13373-017-0101-1.

A. M. Stuart. Inverse problems: a Bayesian perspective. *Acta Numerica*, 19:451–559, 2010. ISSN 0962-4929. doi: 10.1017/S0962492910000061.

D. Talay and L. Tubaro. Expansion of the global error for numerical schemes solving stochastic differential equations. *Stochastic Analysis and Applications*, 8(4):483–509 (1991), 1990. doi: 10.1080/07362999008809220.

T. van Erven and P. Harremos. Rényi Divergence and Kullback-Leibler Divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014. doi: 10.1109/TIT.2014.2320500.

C. Villani. *Optimal transport : old and new*. Grundlehren der mathematischen Wissenschaften. Springer, Berlin, 2009. ISBN 978-3-540-71049-3.

S. J. Vollmer, K. C. Zygalakis, and Y. W. Teh. Exploration of the (non-) asymptotic bias and variance of stochastic gradient langevin dynamics. *The Journal of Machine Learning Research*, 17(1):5504–5548, 2016.

L. Wang. *Heat capacity bound, energy fluctuations and convexity*. Yale University, 2014.

M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the International Conference on Machine Learning*, pages 681–688, 2011.

A. Wibisono. Sampling as optimization in the space of measures: The Langevin dynamics as a composite optimization problem. *arXiv preprint arXiv:1802.08089*, 2018.

P. Xu, J. Chen, D. Zou, and Q. Gu. Global convergence of langevin dynamics based algorithms for nonconvex optimization. *arXiv preprint arXiv:1707.06618*, 2017.

D. Zhu and P. Marcotte. New classes of generalized monotonicity. *Journal of Optimization Theory and Applications*, 87(2):457–471, nov 1995. doi: 10.1007/bf02192574.

Journal of Machine Learning Research 20 (2019) 1-25 Submitted 4/18; Revised 1/19; Published 4/19

Deep Optimal Stopping

Sebastian Becker sebastian.becker@zenai.ch
Zenai AG, 8045 Zurich, Switzerland

Patrick Cheridito patrick.cheridito@math.ethz.ch
RiskLab, Department of Mathematics
ETH Zurich, 8092 Zurich, Switzerland

Arnulf Jentzen arnulf.jentzen@sam.math.ethz.ch
SAM, Department of Mathematics
ETH Zurich, 8092 Zurich, Switzerland

Editor: Jon McAuliffe

Abstract

In this paper we develop a deep learning method for optimal stopping problems which directly learns the optimal stopping rule from Monte Carlo samples. As such, it is broadly applicable in situations where the underlying randomness can efficiently be simulated. We test the approach on three problems: the pricing of a Bermudan max-call option, the pricing of a callable multi barrier reverse convertible and the problem of optimally stopping a fractional Brownian motion. In all three cases it produces very accurate results in high-dimensional situations with short computing times.

Keywords: optimal stopping, deep learning, Bermudan option, callable multi barrier reverse convertible, fractional Brownian motion

1. Introduction

We consider optimal stopping problems of the form $\sup_\tau \mathbb{E}\, g(\tau, X_\tau)$, where $X = (X_n)_{n=0}^N$ is an $\mathbb{R}^d$-valued discrete-time Markov process and the supremum is over all stopping times τ based on observations of X. Formally, this just covers situations where the stopping decision can only be made at finitely many times. But practically all relevant continuous-time stopping problems can be approximated with time-discretized versions. The Markov assumption means no loss of generality. We make it because it simplifies the presentation and many important problems already are in Markovian form. But every optimal stopping problem can be made Markov by including all relevant information from the past in the current state of X (albeit at the cost of increasing the dimension of the problem).

In theory, optimal stopping problems with finitely many stopping opportunities can be solved exactly. The optimal value is given by the smallest supermartingale that dominates the reward process – the so-called Snell envelope – and the smallest (largest) optimal stopping time is the first time the immediate reward dominates (exceeds) the continuation value; see, e.g., Peskir and Shiryaev (2006) or Lamberton and Lapeyre (2008). However, traditional numerical methods suffer from the curse of dimensionality. For instance, the complexity of standard tree- or lattice-based methods increases exponentially in the dimension. For typical problems they yield good results for up to three dimensions. To treat higher-dimensional problems, various Monte Carlo based have been developed over the last years. A common approach consists in estimating continuation values to either derive stopping rules or recursively approximate the Snell envelope; see, e.g., Tilley (1993), Barraquand and Martineau (1995), Carriere (1996), Longstaff and Schwartz (2001), Tsitsiklis and Van Roy (2001), Boyle et al. (2003), Broadie and Glasserman (2004), Bally et al. (2005), Kolodko and Schoenmakers (2006), Egloff et al. (2007), Berridge and Schumacher (2008), Jain and Oosterlee (2015), Belomestny et al. (2018) or Haugh and Kogan (2004) and Kohler et al. (2010), which use neural networks with one hidden layer to do this. A different strand of the literature has focused on approximating optimal exercise boundaries; see, e.g., Andersen (2000), García (2003) and Belomestny (2011). Based on an idea of Davis and Karatzas (1994), a dual approach was developed by Rogers (2002) and Haugh and Kogan (2004); see Jamshidian (2007) and Chen and Glasserman (2007) for a multiplicative version and Andersen and Broadie (2004), Broadie and Cao (2008), Belomestny et al. (2009), Rogers (2010), Desai et al. (2012), Belomestny (2013), Belomestny et al. (2013) and Lelong (2016) for extensions and primal-dual methods. In Sirignano and Spiliopoulos (2018) optimal stopping problems in continuous time are treated by approximating the solutions of the corresponding free boundary PDEs with deep neural networks.

In this paper we use deep learning to approximate an optimal stopping time. Our approach is related to policy optimization methods used in reinforcement learning (Sutton and Barto, 1998), deep reinforcement learning (Schulman et al., 2015; Mnih et al., 2015; Silver et al., 2016; Lillicrap et al., 2016) and the deep learning method for stochastic control problems proposed by Han and E (2016). However, optimal stopping differs from the typical control problems studied in this literature. The challenge of our approach lies in the implementation of a deep learning method that can efficiently learn optimal stopping times. We do this by decomposing an optimal stopping time into a sequence of 0-1 stopping decisions and approximating them recursively with a sequence of multilayer feedforward neural networks. We show that our neural network policies can approximate optimal stopping times to any degree of desired accuracy. A candidate optimal stopping time $\hat{\tau}$ can be obtained by running a stochastic gradient ascent. The corresponding expectation $\mathbb{E}\, g(\hat{\tau}, X_{\hat{\tau}})$ provides a lower bound for the optimal value $\sup_\tau \mathbb{E}\, g(\tau, X_\tau)$. Using a version of the dual method of Rogers (2002) and Haugh and Kogan (2004), we also derive an upper bound. In all our examples, both bounds can be computed with short run times and lie close together.

The rest of the paper is organized as follows: In Section 2 we introduce the setup and explain our method of approximating optimal stopping times with neural networks. In Section 3 we construct lower bounds, upper bounds, point estimates and confidence intervals for the optimal value. In Section 4 we test the approach on three examples: the pricing of a Bermudan max-call option on different underlying assets, the pricing of a callable multi barrier reverse convertible and the problem of optimally stopping a fractional Brownian motion. In the first two examples, we use a multi-dimensional Black–Scholes model to describe the dynamics of the underlying assets. Then the pricing of a Bermudan max-call option amounts to solving a d-dimensional optimal stopping problem, where d is the number of assets. We provide numerical results for $d = 2, 3, 5, 10, 20, 30, 50, 100, 200$ and 500. In the case of a callable MBRC, it becomes a $d+1$-dimensional stopping problem since one also needs to keep track of the barrier event. We present results for $d = 2, 3, 5, 10, 15$ and 30. In the third example we only consider a one-dimensional fractional Brownian

©2019 Sebastian Becker, Patrick Cheridito and Arnulf Jentzen.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-232.html.

motion. But fractional Brownian motion is not Markov. In fact, all of its increments are correlated. So, to optimally stop it, one has to keep track of all past movements. To make it tractable, we approximate the continuous-time problem with a time-discretized version, which if formulated as a Markovian problem, has as many dimensions as there are time-steps. We compute a solution for 100 time-steps.

2. Deep Learning Optimal Stopping Rules

Let $X = (X_n)_{n=0}^N$ be an $\mathbb{R}^d$-valued discrete-time Markov process on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, where N and d are positive integers. We denote by $\mathcal{F}_n$ the σ-algebra generated by $X_0, X_1, \ldots, X_n$ and call a random variable $\tau\colon \Omega \to \{0, 1, \ldots, N\}$ an X-stopping time if the event $\{\tau = n\}$ belongs to $\mathcal{F}_n$ for all $n \in \{0, 1, \ldots, N\}$.

Our aim is to develop a deep learning method that can efficiently learn an optimal policy for stopping problems of the form

$$\sup_{\tau \in \mathcal{T}} \mathbb{E}\, g(\tau, X_\tau), \tag{1}$$

where $g\colon \{0, 1, \ldots, N\} \times \mathbb{R}^d \to \mathbb{R}$ is a measurable function and $\mathcal{T}$ denotes the set of all X-stopping times. To make sure that problem (1) is well-defined and admits an optimal solution, we assume that g satisfies the integrability condition

$$\mathbb{E}\,|g(n, X_n)| < \infty \quad \text{for all } n \in \{0, 1, \ldots, N\}; \tag{2}$$

see, e.g., Peskir and Shiryaev (2006) or Lamberton and Lapeyre (2008). To be able to derive confidence intervals for the optimal value (1), we will have to make the slightly stronger assumption

$$\mathbb{E}\big[g(n, X_n)^2\big] < \infty \quad \text{for all } n \in \{0, 1, \ldots, N\} \tag{3}$$

in Subsection 3.3 below. This is satisfied in all our examples in Section 4.

2.1. Expressing Stopping Times in Terms of Stopping Decisions

Any X-stopping time can be decomposed into a sequence of 0-1 stopping decisions. In principle, the decision whether to stop the process at time n if it has not been stopped before, can be made based on the whole evolution of X from time 0 until n. But to optimally stop the Markov process X, it is enough to make stopping decisions according to $f_n(X_n)$ for measurable functions $f_n\colon \mathbb{R}^d \to \{0, 1\}$, $n = 0, 1, \ldots, N$. Theorem 1 below extends this well-known fact and serves as the theoretical basis of our method.

Consider the auxiliary stopping problems

$$V_n = \sup_{\tau \in \mathcal{T}_n} \mathbb{E}\, g(\tau, X_\tau) \tag{4}$$

for $n = 0, 1, \ldots, N$, where $\mathcal{T}_n$ is the set of all X-stopping times satisfying $n \le \tau \le N$. Obviously, $\mathcal{T}_N$ consists of the unique element $\tau_N \equiv N$, and one can write $\tau_N = N f_N(X_N)$ for the constant function $f_N \equiv 1$. Moreover, for given $n \in \{0, 1, \ldots, N\}$ and a sequence of measurable functions $f_n, f_{n+1}, \ldots, f_N\colon \mathbb{R}^d \to \{0, 1\}$ with $f_N \equiv 1$,

$$\tau_n = \sum_{m=n}^{N} m f_m(X_m) \prod_{j=n}^{m-1} (1 - f_j(X_j)) \tag{5}$$

defines[1] a stopping time in $\mathcal{T}_n$. The following result shows that, for our method of recursively computing an approximate solution to the optimal stopping problem (1), it will be sufficient to consider stopping times of the form (5).

Theorem 1 *For a given $n \in \{0, 1, \ldots, N-1\}$, let τ_{n+1} be a stopping time in $\mathcal{T}_{n+1}$ of the form*

$$\tau_{n+1} = \sum_{m=n+1}^{N} m f_m(X_m) \prod_{j=n+1}^{m-1} (1 - f_j(X_j)) \tag{6}$$

for measurable functions $f_{n+1}, \ldots, f_N\colon \mathbb{R}^d \to \{0, 1\}$ with $f_N \equiv 1$. Then there exists a measurable function $f_n\colon \mathbb{R}^d \to \{0, 1\}$ such that the stopping time $\tau_n \in \mathcal{T}_n$ given by (5) satisfies

$$\mathbb{E}\, g(\tau_n, X_{\tau_n}) \ge V_n - \big(V_{n+1} - \mathbb{E}\, g(\tau_{n+1}, X_{\tau_{n+1}})\big)\,,$$

where V_n and V_{n+1} are the optimal values defined in (4).

Proof Denote $\varepsilon = V_{n+1} - \mathbb{E}\, g(\tau_{n+1}, X_{\tau_{n+1}})$, and consider a stopping time $\tau \in \mathcal{T}_n$. By the Doob–Dynkin lemma (see, e.g., Aliprantis and Border, 2006, Theorem 4.41), there exists a measurable function $h_n\colon \mathbb{R}^d \to \mathbb{R}$ such that $h_n(X_n)$ is a version of the conditional expectation $\mathbb{E}\big[g(\tau_{n+1}, X_{\tau_{n+1}}) \mid X_n\big]$. Moreover, due to the special form (6) of τ_{n+1},

$$g(\tau_{n+1}, X_{\tau_{n+1}}) = \sum_{m=n+1}^{N} g(m, X_m) 1_{\{\tau_{n+1} = m\}} = \sum_{m=n+1}^{N} g(m, X_m) 1_{\{f_m(X_m) \prod_{j=n+1}^{m-1} (1 - f_j(X_j)) = 1\}}$$

is a measurable function of $X_{n+1}, \ldots, X_N$. So it follows from the Markov property of X that $h_n(X_n)$ is also a version of the conditional expectation $\mathbb{E}\big[g(\tau_{n+1}, X_{\tau_{n+1}}) \mid \mathcal{F}_n\big]$. Since the events

$$D = \{g(n, X_n) \ge h_n(X_n)\} \quad \text{and} \quad E = \{\tau = n\}$$

are in $\mathcal{F}_n$, $\tau_n = n 1_D + \tau_{n+1} 1_{D^c}$ belongs to $\mathcal{T}_n$ and $\tilde{\tau} = \tau_{n+1} 1_E + \tau 1_{E^c}$ to $\mathcal{T}_{n+1}$. It follows from the definitions of V_{n+1} and ε that $\mathbb{E}\, g(\tau_{n+1}, X_{\tau_{n+1}}) = V_{n+1} - \varepsilon \ge \mathbb{E}\, g(\tilde{\tau}, X_{\tilde{\tau}}) - \varepsilon$. Hence,

$$\mathbb{E}\big[g(\tau_{n+1}, X_{\tau_{n+1}}) 1_{E^c}\big] \ge \mathbb{E}[g(\tilde{\tau}, X_{\tilde{\tau}}) 1_{E^c}] - \varepsilon = \mathbb{E}[g(\tau, X_\tau) 1_{E^c}] - \varepsilon,$$

from which one obtains

$$\begin{aligned}
&\mathbb{E}\, g(\tau_n, X_{\tau_n}) = \mathbb{E}\big[g(n, X_n) I_D + g(\tau_{n+1}, X_{\tau_{n+1}}) I_{D^c}\big] = \mathbb{E}[g(n, X_n) I_D + h_n(X_n) I_{D^c}] \\
&\ge \mathbb{E}[g(n, X_n) I_E + h_n(X_n) I_{E^c}] = \mathbb{E}\big[g(n, X_n) I_E + g(\tau_{n+1}, X_{\tau_{n+1}}) I_{E^c}\big] \\
&\ge \mathbb{E}[g(n, X_n) I_E + g(\tau, X_\tau) I_{E^c}] - \varepsilon = \mathbb{E}\, g(\tau, X_\tau) - \varepsilon.
\end{aligned}$$

Since $\tau \in \mathcal{T}_n$ was arbitrary, this shows that $\mathbb{E}\, g(\tau_n, X_{\tau_n}) \ge V_n - \varepsilon$. Moreover, one has $1_D = f_n(X_n)$ for the function $f_n\colon \mathbb{R}^d \to \{0, 1\}$ given by

$$f_n(x) = \begin{cases} 1 & \text{if } g(n, x) \ge h_n(x) \\ 0 & \text{if } g(n, x) < h_n(x) \end{cases}.$$

1. In expressions of the form (5), we understand the empty product $\prod_{j=n}^{n-1} (1 - f_j(X_j))$ as 1.

Therefore,

$$\tau_n = n f_n(X_n) + \tau_{n+1}(1 - f_n(X_n)) = \sum_{m=n}^{N} m f_m(X_m) \prod_{j=n}^{m-1} (1 - f_j(X_j)),$$

which concludes the proof. ∎

Remark 2 *Since for $f_N \equiv 1$, the stopping time $\tau_N = f_N(X_N)$ is optimal in $\mathcal{T}_N$, Theorem 1 inductively yields measurable functions $f_n \colon \mathbb{R}^d \to \{0,1\}$ such that for all $n \in \{0,1,\dots,N-1\}$, the stopping time τ_n given by (5) is optimal among $\mathcal{T}_n$. In particular,*

$$\tau = \sum_{n=1}^{N} n f_n(X_n) \prod_{j=0}^{n-1} (1 - f_j(X_j)) \tag{7}$$

is an optimal stopping time for problem (1).

Remark 3 *In many applications, the Markov process X starts from a deterministic initial value $x_0 \in \mathbb{R}^d$. Then the function f_0 enters the representation (7) only through the value $f_0(x_0) \in \{0,1\}$; that is, at time 0, only a constant and not a whole function has to be learned.*

2.2. Neural Network Approximation

Our numerical method for problem (1) consists in iteratively approximating optimal stopping decisions $f_n \colon \mathbb{R}^d \to \{0,1\}$, $n = 0,1,\dots,N-1$, by a neural network $f^\theta \colon \mathbb{R}^d \to \{0,1\}$ with parameter $\theta \in \mathbb{R}^q$. We do this by starting with the terminal stopping decision $f_N \equiv 1$ and proceeding by backward induction. More precisely, let $n \in \{0,1,\dots,N-1\}$, and assume parameter values $\theta_{n+1}, \theta_{n+2}, \dots, \theta_N \in \mathbb{R}^q$ have been found such that $f^{\theta_N} \equiv 1$ and the stopping time

$$\tau_{n+1} = \sum_{m=n+1}^{N} m f^{\theta_m}(X_m) \prod_{j=n+1}^{m-1} (1 - f^{\theta_j}(X_j))$$

produces an expected value $\mathbb{E}\, g(\tau_{n+1}, X_{\tau_{n+1}})$ close to the optimum V_{n+1}. Since f^θ takes values in $\{0,1\}$, it does not directly lend itself to a gradient-based optimization method. So, as an intermediate step, we introduce a feedforward neural network $F^\theta \colon \mathbb{R}^d \to (0,1)$ of the form

$$F^\theta = \psi \circ a_I^\theta \circ \varphi_{q_{I-1}} \circ a_{I-1}^\theta \circ \cdots \circ \varphi_{q_1} \circ a_1^\theta,$$

where

- $I, q_1, q_2, \dots, q_{I-1}$ are positive integers specifying the depth of the network and the number of nodes in the hidden layers (if there are any),
- $a_1^\theta \colon \mathbb{R}^d \to \mathbb{R}^{q_1}, \dots, a_{I-1}^\theta \colon \mathbb{R}^{q_{I-2}} \to \mathbb{R}^{q_{I-1}}$ and $a_I^\theta \colon \mathbb{R}^{q_{I-1}} \to \mathbb{R}$ are affine functions,
- for $j \in \mathbb{N}$, $\varphi_j \colon \mathbb{R}^j \to \mathbb{R}^j$ is the component-wise ReLU activation function given by $\varphi_j(x_1,\dots,x_j) = (x_1^+,\dots,x_j^+)$

- $\psi \colon \mathbb{R} \to (0,1)$ is the standard logistic function $\psi(x) = e^x/(1+e^x) = 1/(1+e^{-x})$.

The components of the parameter $\theta \in \mathbb{R}^q$ of F^θ consist of the entries of the matrices $A_1 \in \mathbb{R}^{q_1 \times d}, \dots, A_{I-1} \in \mathbb{R}^{q_{I-1} \times q_{I-2}}, A_I \in \mathbb{R}^{1 \times q_{I-1}}$ and the vectors $b_1 \in \mathbb{R}^{q_1}, \dots, b_{I-1} \in \mathbb{R}^{q_{I-1}}, b_I \in \mathbb{R}$ given by the representation of the affine functions

$$a_i^\theta(x) = A_i x + b_i, \quad i = 1,\dots,I.$$

So the dimension of the parameter space is

$$q = \begin{cases} d+1 & \text{if } I = 1 \\ 1 + q_1 + \cdots + q_{I-1} + d q_1 + \cdots + q_{I-2} q_{I-1} + q_{I-1} & \text{if } I \geq 2, \end{cases}$$

and for given $x \in \mathbb{R}^d$, $F^\theta(x)$ is continuous as well as almost everywhere smooth in θ. Our aim is to determine $\theta_n \in \mathbb{R}^q$ so that

$$\mathbb{E}\Big[g(n, X_n) F^{\theta_n}(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - F^{\theta_n}(X_n)) \Big]$$

is close to the supremum $\sup_{\theta \in \mathbb{R}^q} \mathbb{E}\big[g(n, X_n) F^\theta(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - F^\theta(X_n)) \big]$. Once this has been achieved, we define the function $f^{\theta_n} \colon \mathbb{R}^d \to \{0,1\}$ by

$$f^{\theta_n} = 1_{[0,\infty)} \circ a_I^{\theta_n} \circ \varphi_{q_{I-1}} \circ a_{I-1}^{\theta_n} \circ \cdots \circ \varphi_{q_1} \circ a_1^{\theta_n}, \tag{8}$$

where $1_{[0,\infty)} \colon \mathbb{R} \to \{0,1\}$ is the indicator function of $[0,\infty)$. The only difference between F^{θ_n} and f^{θ_n} is the final nonlinearity. While F^{θ_n} produces a stopping probability in $(0,1)$, the output of f^{θ_n} is a hard stopping decision given by 0 or 1, depending on whether F^{θ_n} takes a value below or above $1/2$.

The following result shows that for any depth $I \geq 2$, a neural network of the form (8) is flexible enough to make almost optimal stopping decisions provided it has sufficiently many nodes.

Proposition 4 *Let $n \in \{0,1,\dots,N-1\}$ and fix a stopping time $\tau_{n+1} \in \mathcal{T}_{n+1}$. Then, for every depth $I \geq 2$ and constant $\varepsilon > 0$, there exist positive integers $q_1, \dots, q_{I-1}$ such that*

$$\begin{aligned} &\sup_{\theta \in \mathbb{R}^q} \mathbb{E}\Big[g(n, X_n) f^\theta(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - f^\theta(X_n)) \Big] \\ &\geq \sup_{f \in \mathcal{D}} \mathbb{E}\big[g(n, X_n) f(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - f(X_n)) \big] - \varepsilon, \end{aligned}$$

where $\mathcal{D}$ is the set of all measurable functions $f \colon \mathbb{R}^d \to \{0,1\}$.

Proof Fix $\varepsilon > 0$. It follows from the integrability condition (2) that there exists a measurable function $\tilde{f} \colon \mathbb{R}^d \to \{0,1\}$ such that

$$\begin{aligned} &\mathbb{E}\Big[g(n, X_n) \tilde{f}(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - \tilde{f}(X_n)) \Big] \\ &\geq \sup_{f \in \mathcal{D}} \mathbb{E}\big[g(n, X_n) f(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - f(X_n)) \big] - \varepsilon/4. \end{aligned} \tag{9}$$

$\tilde{f}$ can be written as $\tilde{f} = 1_A$ for the Borel set $A = \{x \in \mathbb{R}^d : \tilde{f}(x) = 1\}$. Moreover, by (2),

$$B \mapsto \mathbb{E}[|g(n, X_n)|1_B(X_n)] \quad \text{and} \quad B \mapsto \mathbb{E}\big[|g(\tau_{n+1}, X_{\tau_{n+1}})|1_B(X_n)\big]$$

define finite Borel measures on $\mathbb{R}^d$. Since every finite Borel measure on $\mathbb{R}^d$ is tight (see, e.g., Aliprantis and Border, 2006), there exists a compact (possibly empty) subset $K \subseteq A$ such that

$$\begin{aligned} &\mathbb{E}\big[g(n, X_n)1_K(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - 1_K(X_n))\big] \\ &\geq \mathbb{E}\Big[g(n, X_n)\tilde{f}(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - \tilde{f}(X_n))\Big] - \varepsilon/4. \end{aligned} \tag{10}$$

Let $\rho_K \colon \mathbb{R}^d \to [0, \infty]$ be the distance function given by $\rho_K(x) = \inf_{y \in K} \|x - y\|_2$. Then

$$k_j(x) = \max\{1 - j\rho_K(x), -1\}, \quad j \in \mathbb{N},$$

defines a sequence of continuous functions $k_j \colon \mathbb{R}^d \to [-1, 1]$ that converge pointwise to $1_K - 1_{K^c}$. So it follows from Lebesgue's dominated convergence theorem that there exists a $j \in \mathbb{N}$ such that

$$\begin{aligned} &\mathbb{E}\Big[g(n, X_n)\, 1_{\{k_j(X_n) \geq 0\}} + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - 1_{\{k_j(X_n) \geq 0\}})\Big] \\ &\geq \mathbb{E}\big[g(n, X_n)1_K(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - 1_K(X_n))\big] - \varepsilon/4. \end{aligned} \tag{11}$$

By Theorem 1 of Leshno et al. (1993), k_j can be approximated uniformly on compacts by functions of the form

$$\sum_{i=1}^{r} (v_i^T x + c_i)^+ - \sum_{i=1}^{s} (w_i^T x + d_i)^+ \tag{12}$$

for $r, s \in \mathbb{N}$, $v_1, \ldots, v_r, w_1, \ldots, w_s \in \mathbb{R}^d$ and $c_1, \ldots, c_r, d_1, \ldots, d_s \in \mathbb{R}$. So there exists a function $h \colon \mathbb{R}^d \to \mathbb{R}$ expressible as in (12) such that

$$\begin{aligned} &\mathbb{E}\big[g(n, X_n)\, 1_{\{h(X_n) \geq 0\}} + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - 1_{\{h(X_n) \geq 0\}})\big] \\ &\geq \mathbb{E}\Big[g(n, X_n)\, 1_{\{k_j(X_n) \geq 0\}} + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - 1_{\{k_j(X_n) \geq 0\}})\Big] - \varepsilon/4. \end{aligned} \tag{13}$$

Now note that for any integer $I \geq 2$, the composite mapping $1_{[0,\infty)} \circ h$ can be written as a neural net f^θ of the form (8) with depth I for suitable integers $q_1, \ldots, q_{I-1}$ and parameter value $\theta \in \mathbb{R}^q$. Hence, one obtains from (9), (10), (11) and (13) that

$$\begin{aligned} &\mathbb{E}\Big[g(n, X_n)\, f^\theta(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - f^\theta(X_n))\Big] \\ &\geq \sup_{f \in \mathcal{D}} \mathbb{E}\big[g(n, X_n)f(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - f(X_n))\big] - \varepsilon, \end{aligned}$$

and the proof is complete. ■

We always choose $\theta_N \in \mathbb{R}^q$ such that[2] $f^{\theta_N} \equiv 1$. Then our candidate optimal stopping time

$$\tau^\Theta = \sum_{n=1}^{N} n f^{\theta_n}(X_n) \prod_{j=0}^{n-1} (1 - f^{\theta_j}(X_j)) \tag{14}$$

2. It is easy to see that this is possible.

is specified by the vector $\Theta = (\theta_0, \theta_1, \ldots, \theta_{N-1}) \in \mathbb{R}^{Nq}$. The following is an immediate consequence of Theorem 1 and Proposition 4:

Corollary 5 *For a given optimal stopping problem of the form* (1)*, a depth $I \geq 2$ and a constant $\varepsilon > 0$, there exist positive integers $q_1, \ldots, q_{I-1}$ and a vector $\Theta \in \mathbb{R}^{Nq}$ such that the corresponding stopping time* (14) *satisfies $\mathbb{E}\, g(\tau^\Theta, X_{\tau^\Theta}) \geq \sup_{\tau \in \mathcal{T}} \mathbb{E}\, g(\tau, X_\tau) - \varepsilon$.*

2.3. Parameter Optimization

We train neural networks of the form (8) with fixed depth $I \geq 2$ and given numbers $q_1, \ldots, q_{I-1}$ of nodes in the hidden layers[3]. To numerically find parameters $\theta_n \in \mathbb{R}^q$ yielding good stopping decisions f^{θ_n} for all times $n \in \{0, 1, \ldots, N-1\}$, we approximate expected values with averages of Monte Carlo samples calculated from simulated paths of the process $(X_n)_{n=0}^N$.

Let $(x_n^k)_{n=0}^N$, $k = 1, 2, \ldots$ be independent realizations of such paths. We choose $\theta_N \in \mathbb{R}^q$ such that $f^{\theta_N} \equiv 1$ and determine determine $\theta_n \in \mathbb{R}^q$ for $n \leq N-1$ recursively. So, suppose that for a given $n \in \{0, 1, \ldots, N-1\}$, parameters $\theta_{n+1}, \ldots, \theta_N \in \mathbb{R}^q$, have been found so that the stopping decisions $f^{\theta_{n+1}}, \ldots, f^{\theta_N}$ generate a stopping time

$$\tau_{n+1} = \sum_{m=n+1}^{N} m f^{\theta_m}(X_m) \prod_{j=n+1}^{m-1} (1 - f^{\theta_j}(X_j))$$

with corresponding expectation $\mathbb{E}\, g(\tau_{n+1}, X_{\tau_{n+1}})$ close to the optimal value V_{n+1}. If $n = N-1$, one has $\tau_{n+1} \equiv N$, and if $n \leq N-2$, τ_{n+1} can be written as

$$\tau_{n+1} = l_{n+1}(X_{n+1}, \ldots, X_{N-1})$$

for a measurable function $l_{n+1} \colon \mathbb{R}^{d(N-n-1)} \to \{n+1, n+2, \ldots, N\}$. Accordingly, denote

$$l_{n+1}^k = \begin{cases} N & \text{if } n = N-1 \\ l_{n+1}(x_{n+1}^k, \ldots, x_{N-1}^k) & \text{if } n \leq N-2 \end{cases}.$$

If at time n, one applies the soft stopping decision F^θ and afterward behaves according to $f^{\theta_{n+1}}, \ldots, f^{\theta_N}$, the realized reward along the k-th simulated path of X is

$$r_n^k(\theta) = g(n, x_n^k)F^\theta(x_n^k) + g(l_{n+1}^k, x_{l_{n+1}^k}^k)(1 - F^\theta(x_n^k)).$$

For large $K \in \mathbb{N}$,

$$\frac{1}{K} \sum_{k=1}^{K} r_n^k(\theta) \tag{15}$$

approximates the expected value

$$\mathbb{E}\Big[g(n, X_n)F^\theta(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - F^\theta(X_n))\Big].$$

3. For a given application, one can try out different choices of I and $q_1, \ldots, q_{I-1}$ to find a suitable trade-off between accuracy and efficiency. Alternatively, the determination of I and $q_1, \ldots, q_{I-1}$ could be built into the training algorithm.

Since $r_n^k(\theta)$ is almost everywhere differentiable in θ, a stochastic gradient ascent method can be applied to find an approximate optimizer $\theta_n \in \mathbb{R}^q$ of (15). The same simulations $(x_n^k)_{n=0}^N$, $k = 1, 2, \ldots$ can be used to train the stopping decisions f^{θ_n} at all times $n \in \{0, 1, \ldots, N-1\}$. In the numerical examples in Section 4 below, we employed mini-batch gradient ascent with Xavier initialization (Glorot and Bengio, 2010), batch normalization (Ioffe and Szegedy, 2015) and Adam updating (Kingma and Ba, 2015).

Remark 6 *If the Markov process X starts from a deterministic initial value $x_0 \in \mathbb{R}^d$, the initial stopping decision is given by a constant $f_0 \in \{0, 1\}$. To learn f_0 from simulated paths of X, it is enough to compare the initial reward $g(0, x_0)$ to a Monte Carlo estimate $\hat{C}$ of $\mathbb{E}\, g(\tau_1, X_{\tau_1})$, where $\tau_1 \in \mathcal{T}_1$ is of the form*

$$\tau_1 = \sum_{n=1}^{N} n f^{\theta_n}(X_n) \prod_{j=1}^{n-1} (1 - f^{\theta_j}(X_j))$$

for $f^{\theta_N} \equiv 1$ and trained parameters $\theta_1, \ldots, \theta_{N-1} \in \mathbb{R}^q$. Then one sets $f_0 = 1$ (that is, stop immediately) if $g(0, x_0) \geq \hat{C}$ and $f_0 = 0$ (continue) otherwise. The resulting stopping time is of the form

$$\tau^\Theta = \begin{cases} 0 & \text{if } f_0 = 1 \\ \tau_1 & \text{if } f_0 = 0. \end{cases}$$

3. Bounds, Point Estimates and Confidence Intervals

In this section we derive lower and upper bounds as well as point estimates and confidence intervals for the optimal value $V_0 = \sup_{\tau \in \mathcal{T}} \mathbb{E}\, g(\tau, X_\tau)$.

3.1. Lower Bound

Once the stopping decisions f^{θ_n} have been trained, the stopping time τ^Θ given by (14) yields a lower bound $L = \mathbb{E}\, g(\tau^\Theta, X_{\tau^\Theta})$ for the optimal value $V_0 = \sup_{\tau \in \mathcal{T}} \mathbb{E}\, g(\tau, X_\tau)$. To estimate it, we simulate a new set[4] of independent realizations $(y_n^k)_{n=0}^N$, $k = 1, 2, \ldots, K_L$, of $(X_n)_{n=0}^N$. τ^Θ is of the form $\tau^\Theta = l(X_0, \ldots, X_{N-1})$ for a measurable function $l \colon \mathbb{R}^{dN} \to \{0, 1, \ldots, N\}$. Denote $l^k = l(y_0^k, \ldots, y_{N-1}^k)$. The Monte Carlo approximation

$$\hat{L} = \frac{1}{K_L} \sum_{k=1}^{K_L} g(l^k, y_{l^k}^k)$$

gives an unbiased estimate of the lower bound L, and by the law of large numbers, $\hat{L}$ converges to L for $K_L \to \infty$.

4. In particular, we assume that the samples $(y_n^k)_{n=0}^N$, $k = 1, \ldots, K_L$, are drawn independently from the realizations $(x_n^k)_{n=0}^N$, $k = 1, \ldots, K$, used in the training of the stopping decisions.

3.2. Upper Bound

The Snell envelope of the reward process $(g(n, X_n))_{n=0}^N$ is the smallest[5] supermartingale with respect to $(\mathcal{F}_n)_{n=0}^N$ that dominates $(g(n, X_n))_{n=0}^N$. It is given[6] by

$$H_n = \operatorname{ess\,sup}_{\tau \in \mathcal{T}_n} \mathbb{E}[g(\tau) \mid \mathcal{F}_n], \quad n = 0, 1, \ldots, N;$$

see, e.g., Peskir and Shiryaev (2006) or Lamberton and Lapeyre (2008). Its Doob–Meyer decomposition is

$$H_n = H_0 + M_n^H - A_n^H,$$

where M^H is the $(\mathcal{F}_n)$-martingale given[6] by

$$M_0^H = 0 \quad \text{and} \quad M_n^H - M_{n-1}^H = H_n - \mathbb{E}[H_n \mid \mathcal{F}_{n-1}], \quad n = 1, \ldots, N,$$

and A^H is the nondecreasing $(\mathcal{F}_n)$-predictable process given[6] by

$$A_0^H = 0 \quad \text{and} \quad A_n^H - A_{n-1}^H = H_{n-1} - \mathbb{E}[H_n \mid \mathcal{F}_{n-1}], \quad n = 1, \ldots, N.$$

Our estimate of an upper bound for the optimal value V_0 is based on the following variant[7] of the dual formulation of optimal stopping problems introduced by Rogers (2002) and Haugh and Kogan (2004).

Proposition 7 *Let $(\varepsilon_n)_{n=0}^N$ be a sequence of integrable random variables on $(\Omega, \mathcal{F}, \mathbb{P})$. Then*

$$V_0 \geq \mathbb{E}\left[\max_{0 \leq n \leq N} \left(g(n, X_n) - M_n^H - \varepsilon_n\right)\right] + \mathbb{E}\left[\min_{0 \leq n \leq N} \left(A_n^H + \varepsilon_n\right)\right]. \tag{16}$$

Moreover, if $\mathbb{E}[\varepsilon_n \mid \mathcal{F}_n] = 0$ for all $n \in \{0, 1, \ldots, N\}$, one has

$$V_0 \leq \mathbb{E}\left[\max_{0 \leq n \leq N} \left(g(n, X_n) - M_n - \varepsilon_n\right)\right] \tag{17}$$

for every $(\mathcal{F}_n)$-martingale $(M_n)_{n=0}^N$ starting from 0.

Proof First, note that

$$\begin{aligned} &\mathbb{E}\left[\max_{0 \leq n \leq N} \left(g(n, X_n) - M_n^H - \varepsilon_n\right)\right] \leq \mathbb{E}\left[\max_{0 \leq n \leq N} \left(H_n - M_n^H - \varepsilon_n\right)\right] \\ &= \mathbb{E}\left[\max_{0 \leq n \leq N} \left(H_0 - A_n^H - \varepsilon_n\right)\right] = V_0 - \mathbb{E}\left[\min_{0 \leq n \leq N} \left(A_n^H + \varepsilon_n\right)\right], \end{aligned}$$

which shows (16).

Now, assume that $\mathbb{E}[\varepsilon_n \mid \mathcal{F}_n] = 0$ for all $n \in \{0, 1, \ldots, N\}$, and let τ be an X-stopping time. Then

$$\mathbb{E}\, \varepsilon_\tau = \mathbb{E}\left[\sum_{n=0}^{N} 1_{\{\tau = n\}} \varepsilon_n\right] = \mathbb{E}\left[\sum_{n=0}^{N} 1_{\{\tau = n\}} \mathbb{E}[\varepsilon_n \mid \mathcal{F}_n]\right] = 0.$$

5. in the $\mathbb{P}$-almost sure order
6. up to $\mathbb{P}$-almost sure equality
7. See also the discussion on noisy estimates in Andersen and Broadie (2004).

So one obtains from the optional stopping theorem (see, e.g., Grimmett and Stirzaker, 2001),

$$\mathbb{E}\, g(\tau, X_\tau) = \mathbb{E}[g(\tau, X_\tau) - M_\tau - \varepsilon_\tau] \le \mathbb{E}\left[\max_{0\le n\le N} (g(n, X_n) - M_n - \varepsilon_n)\right]$$

for every $(\mathcal{F}_n)$-martingale $(M_n)_{n=0}^N$ starting from 0. Since $V_0 = \sup_{\tau\in\mathcal{T}} \mathbb{E}\, g(\tau, X_\tau)$, this implies (17). ■

For every $(\mathcal{F}_n)$-martingale $(M_n)_{n=0}^N$ starting from 0 and each sequence of integrable error terms $(\varepsilon_n)_{n=0}^N$ satisfying $\mathbb{E}[\varepsilon_n \mid \mathcal{F}_n] = 0$ for all n, the right side of (17) provides an upper bound[8] for V_0, and by (16), this upper bound is tight if $M = M^H$ and $\varepsilon \equiv 0$. So we try to use our candidate optimal stopping time τ^Θ to construct a martingale close to M^H. The closer τ^Θ is to an optimal stopping time, the better the value process[9]

$$H_n^\Theta = \mathbb{E}\Big[g(\tau_n^\Theta, X_{\tau_n^\Theta}) \mid \mathcal{F}_n\Big], \quad n = 0, 1, \dots, N,$$

corresponding to

$$\tau_n^\Theta = \sum_{m=n}^{N} m f^{\theta_m}(X_m) \prod_{j=n}^{m-1} (1 - f^{\theta_j}(X_j)), \quad n = 0, 1, \dots, N,$$

approximates the Snell envelope $(H_n)_{n=0}^N$. The martingale part of $(H_n^\Theta)_{n=0}^N$ is given by $M_0^\Theta = 0$ and

$$M_n^\Theta - M_{n-1}^\Theta = H_n^\Theta - \mathbb{E}\big[H_n^\Theta \mid \mathcal{F}_{n-1}\big] = f^{\theta_n}(X_n) g(n, X_n) + (1 - f^{\theta_n}(X_n)) C_n^\Theta - C_{n-1}^\Theta, \; n \ge 1, \tag{18}$$

for the continuation values[10]

$$C_n^\Theta = \mathbb{E}[g(\tau_{n+1}^\Theta, X_{\tau_{n+1}^\Theta}) \mid \mathcal{F}_n] = \mathbb{E}[g(\tau_{n+1}^\Theta, X_{\tau_{n+1}^\Theta}) \mid X_n], \quad n = 0, 1, \dots, N-1.$$

Note that C_N^Θ does not have to be specified. It formally appears in (18) for $n = N$. But $(1 - f^{\theta_N}(X_N))$ is always 0. To estimate M^Θ, we generate a third set[11] of independent realizations $(z_n^k)_{n=0}^N$, $k = 1, 2, \dots, K_U$, of $(X_n)_{n=0}^N$. In addition, for every z_n^k, we simulate J continuation paths $\tilde{z}_{n+1}^{k,j}, \dots, \tilde{z}_N^{k,j}$, $j = 1, \dots, J$, that are conditionally independent[12] of each other and of $z_{n+1}^k, \dots, z_N^k$. Let us denote by $\tau_{n+1}^{k,j}$ the value of τ_{n+1}^Θ along $\tilde{z}_{n+1}^{k,j}, \dots, \tilde{z}_N^{k,j}$. Estimating the continuation values as

$$C_n^k = \frac{1}{J} \sum_{j=1}^{J} g\left(\tau_{n+1}^{k,j}, \tilde{z}_{\tau_{n+1}^{k,j}}^{k,j}\right), \quad n = 0, 1, \dots, N-1,$$

yields the noisy estimates

$$\Delta M_n^k = f^{\theta_n}(z_n^k) g(n, z_n^k) + (1 - f^{\theta_n}(z_n^k)) C_n^k - C_{n-1}^k$$

of the increments $M_n^\Theta - M_{n-1}^\Theta$ along the k-th simulated path $z_0^k, \dots, z_N^k$. So

$$M_n^k = \begin{cases} 0 & \text{if } n = 0 \\ \sum_{m=1}^n \Delta M_m^k & \text{if } n \ge 1 \end{cases}$$

can be viewed as realizations of $M_n^\Theta + \varepsilon_n$ for estimation errors ε_n with standard deviations proportional to $1/\sqrt{J}$ such that $\mathbb{E}[\varepsilon_n \mid \mathcal{F}_n] = 0$ for all n. Accordingly,

$$\hat{U} = \frac{1}{K_U} \sum_{k=1}^{K_U} \max_{0\le n\le N} \left(g\left(n, z_n^k\right) - M_n^k\right),$$

is an unbiased estimate of the upper bound

$$U = \mathbb{E}\left[\max_{0\le n\le N} \left(g(n, X_n) - M_n^\Theta - \varepsilon_n\right)\right],$$

which, by the law of large numbers, converges to U for $K_U \to \infty$.

3.3. Point Estimate and Confidence Intervals

Our point estimate of V_0 is the average

$$\frac{\hat{L} + \hat{U}}{2}.$$

To derive confidence intervals, we assume that $g(n, X_n)$ is square-integrable[13] for all n. Then

$$g(\tau^\theta, X_{\tau^\Theta}) \quad \text{and} \quad \max_{0\le n\le N} \left(g(n, X_n) - M_n^\Theta - \varepsilon_n\right)$$

are square-integrable too. Hence, one obtains from the central limit theorem that for large K_L, $\hat{L}$ is approximately normally distributed with mean L and variance $\hat{\sigma}_L^2/K_L$ for

$$\hat{\sigma}_L^2 = \frac{1}{K_L - 1} \sum_{k=1}^{K_L} \left(g(l^k, y_{l^k}^k) - \hat{L}\right)^2.$$

8. Note that for the right side of (17) to be a valid upper bound, it is sufficient that $\mathbb{E}[\varepsilon_n \mid \mathcal{F}_n] = 0$ for all n. In particular, $\varepsilon_0, \varepsilon_1, \dots, \varepsilon_N$ can have any arbitrary dependence structure.
9. Again, since H_n^Θ, M_n^Θ and C_n^Θ are given by conditional expectations, they are only specified up to $\mathbb{P}$-almost sure equality.
10. The two conditional expectations are equal since $(X_n)_{n=0}^N$ is Markov and τ_{n+1}^Θ only depends on $(X_{n+1}, \dots, X_{N-1})$.
11. The realizations $(z_n^k)_{n=0}^N$, $k = 1, \dots, K_U$, must be drawn independently of $(x_n^k)_{n=0}^N$, $k = 1, \dots, K$, so that our estimate of the upper bound does not depend on the samples used to train the stopping decisions. But theoretically, they can depend on $(y_n^k)_{n=0}^N$, $k = 1, \dots, K_L$, without affecting the unbiasedness of the estimate $\hat{U}$ or the validity of the confidence interval derived in Subsection 3.3 below.
12. More precisely, the tuples $(\tilde{z}_{n+1}^{k,j}, \dots, \tilde{z}_N^{k,j})$, $j = 1, \dots, J$, are simulated according to $p_n(z_n^k, \cdot)$, where p_n is a transition kernel from $\mathbb{R}^d$ to $\mathbb{R}^{(N-n)d}$ such that $p_n(X_n, B) = \mathbb{P}[(X_{n+1}, \dots, X_N) \in B \mid X_n]$ $\mathbb{P}$-almost surely for all Borel sets $B \subseteq \mathbb{R}^{(N-n)d}$. We generate them independently of each other across j and k. On the other hand, the continuation paths starting from z_n^k do not have to be drawn independently of those starting from $z_{n'}^k$ for $n \ne n'$.
13. See condition (3).

So, for every $\alpha \in (0,1]$,

$$\left[\hat{L} - z_{\alpha/2}\frac{\hat{\sigma}_L}{\sqrt{K_L}}\,,\,\infty\right)$$

is an asymptotically valid $1-\alpha/2$ confidence interval for L, where $z_{\alpha/2}$ is the $1-\alpha/2$ quantile of the standard normal distribution. Similarly,

$$\left(-\infty\,,\,\hat{U} + z_{\alpha/2}\frac{\hat{\sigma}_U}{\sqrt{K_U}}\right] \quad \text{with} \quad \hat{\sigma}_U^2 = \frac{1}{K_U - 1}\sum_{k=1}^{K_U}\left(\max_{0\le n\le N}\left(g\left(n, z_n^k\right) - M_n^k\right) - \hat{U}\right)^2,$$

is an asymptotically valid $1-\alpha/2$ confidence interval for U. It follows that for every constant $\varepsilon > 0$, one has

$$\begin{aligned}&\mathbb{P}\left[V_0 < \hat{L} - z_{\alpha/2}\frac{\hat{\sigma}_L}{\sqrt{K_L}} \quad \text{or} \quad V_0 > \hat{U} + z_{\alpha/2}\frac{\hat{\sigma}_U}{\sqrt{K_U}}\right]\\ &\le \mathbb{P}\left[L < \hat{L} - z_{\alpha/2}\frac{\hat{\sigma}_L}{\sqrt{K_L}}\right] + \mathbb{P}\left[U > \hat{U} + z_{\alpha/2}\frac{\hat{\sigma}_U}{\sqrt{K_U}}\right] \le \alpha + \varepsilon\end{aligned}$$

as soon as K_L and K_U are large enough. In particular,

$$\left[\hat{L} - z_{\alpha/2}\frac{\hat{\sigma}_L}{\sqrt{K_L}}\,,\,\hat{U} + z_{\alpha/2}\frac{\hat{\sigma}_U}{\sqrt{K_U}}\right] \tag{19}$$

is an asymptotically valid $1-\alpha$ confidence interval for V_0.

4. Examples

In this section we test[14] our method on three examples: the pricing of a Bermudan max-call option, the pricing of a callable multi barrier reverse convertible and the problem of optimally stopping a fractional Brownian motion.

4.1. Bermudan Max-Call Options

Bermudan max-call options are one of the most studied examples in the numerics literature on optimal stopping problems (see, e.g., Longstaff and Schwartz, 2001; Rogers, 2002; García, 2003; Boyle et al., 2003; Haugh and Kogan, 2004; Broadie and Glasserman, 2004; Andersen and Broadie, 2004; Broadie and Cao, 2008; Berridge and Schumacher, 2008; Belomestny, 2011, 2013; Jain and Oosterlee, 2015; Lelong, 2016). Their payoff depends on the maximum of d underlying assets.

Assume the risk-neutral dynamics of the assets are given by a multi-dimensional Black–Scholes model[15]

$$S_t^i = s_0^i \exp\left(\left[r - \delta_i - \sigma_i^2/2\right]t + \sigma_i W_t^i\right), \quad i = 1, 2, \ldots, d, \tag{20}$$

14. All computations were performed in single precision (float32) on a NVIDIA GeForce GTX 1080 GPU with 1974 MHz core clock and 8 GB GDDR5X memory with 1809.5 MHz clock rate. The underlying system consisted of an Intel Core i7-6800K 3.4 GHz CPU with 64 GB DDR4-2133 memory running Tensorflow 1.11 on Ubuntu 16.04.

15. We make this assumption so that we can compare our results to those obtained with different methods in the literature. But our approach works for any asset dynamics as long as it can efficiently be simulated.

for initial values $s_0^i \in (0,\infty)$, a risk-free interest rate $r \in \mathbb{R}$, dividend yields $\delta_i \in [0,\infty)$, volatilities $\sigma_i \in (0,\infty)$ and a d-dimensional Brownian motion W with constant instantaneous correlations[16] $\rho_{ij} \in \mathbb{R}$ between different components W^i and W^j. A Bermudan max-call option on $S^1, S^2, \ldots, S^d$ has payoff $\left(\max_{1\le i\le d} S_t^i - K\right)^+$ and can be exercised at any point of a time grid $0 = t_0 < t_1 < \cdots < t_N$. Its price is given by

$$\sup_\tau \mathbb{E}\left[e^{-r\tau}\left(\max_{1\le i\le d} S_\tau^i - K\right)^+\right],$$

where the supremum is over all S-stopping times taking values in $\{t_0, t_1, \ldots, t_N\}$ (see, e.g., Schweizer, 2002). Denote $X_n^i = S_{t_n}^i$, $n = 0, 1, \ldots, N$, and let $\mathcal{T}$ be the set of X-stopping times. Then the price can be written as $\sup_{\tau\in\mathcal{T}} \mathbb{E}\, g(\tau, X_\tau)$ for

$$g(n,x) = e^{-rt_n}\left(\max_{1\le i\le d} x^i - K\right)^+,$$

and it is straight-forward to simulate $(X_n)_{n=0}^N$.

In the following we assume the time grid to be of the form $t_n = nT/N$, $n = 0, 1, \ldots, N$, for a maturity $T > 0$ and $N+1$ equidistant exercise dates. Even though $g(n, X_n)$ does not carry any information that is not already contained in X_n, our method worked more efficiently when we trained the optimal stopping decisions on Monte Carlo simulations of the $d+1$-dimensional Markov process $(Y_n)_{n=0}^N = (X_n, g(n, X_n))_{n=0}^N$ instead of $(X_n)_{n=0}^N$. Since Y_0 is deterministic, we first trained stopping times $\tau_1 \in \mathcal{T}_1$ of the form

$$\tau_1 = \sum_{n=1}^N n f^{\theta_n}(Y_n)\prod_{j=1}^{n-1}(1 - f^{\theta_j}(Y_k))$$

for $f^{\theta_N} \equiv 1$ and $f^{\theta_1}, \ldots, f^{\theta_{N-1}} \colon \mathbb{R}^{d+1} \to \{0,1\}$ given by (8) with $I = 2$ and $q_1 = q_2 = d+40$. Then we determined our candidate optimal stopping times as

$$\tau^\Theta = \begin{cases} 0 & \text{if } f_0 = 1\\ \tau_1 & \text{if } f_0 = 0\end{cases}$$

for a constant $f_0 \in \{0,1\}$ depending[17] on whether it was optimal to stop immediately at time 0 or not (see Remark 6 above).

It is straight-forward to simulate from model (20). We conducted 3,000$+d$ training steps, in each of which we generated a batch of 8,192 paths of $(X_n)_{n=0}^N$. To estimate the lower bound L we simulated $K_L = 4{,}096{,}000$ trial paths. For our estimate of the upper bound U, we produced $K_U = 1{,}024$ paths $(z_n^k)_{n=0}^N$, $k = 1, \ldots, K_U$, of $(X_n)_{n=0}^N$ and $K_U \times J$ realizations $(v_n^{k,j})_{n=1}^N$, $k = 1, \ldots, K_U$, $j = 1, \ldots, J$, of $(W_{t_n} - W_{t_{n-1}})_{n=1}^N$ with $J = 16{,}384$. Then for all n and k, we generated the i-th component of the j-th continuation path departing from z_n^k according to

$$z_m^{i,k,j} = z_n^{i,k}\exp\left(\left[r - \delta_i - \sigma_i^2/2\right](m-n)\Delta t + \sigma_i[v_{n+1}^{i,k,j} + \cdots + v_m^{i,k,j}]\right), \quad m = n+1, \ldots, N.$$

16. That is, $\mathbb{E}[(W_t^i - W_s^i)(W_t^j - W_s^i)] = \rho_{ij}(t-s)$ for all $i \ne j$ and $s < t$.

17. In fact, in none of the examples in this paper it is optimal to stop at time 0. So $\tau^\Theta = \tau_1$ in all these cases.

Symmetric case
We first considered the special case, where $s_0^i = s_0$, $\delta_i = \delta$, $\sigma_i = \sigma$ for all $i = 1, \dots, d$, and $\rho_{ij} = \rho$ for all $i \neq j$. Our results are reported in Table 1.

Asymmetric case
As a second example, we studied model (20) with $s_0^i = s_0$, $\delta_i = \delta$ for all $i = 1, 2, \dots, d$, and $\rho_{ij} = \rho$ for all $i \neq j$, but different volatilities $\sigma_1 < \sigma_2 < \cdots < \sigma_d$. For $d \leq 5$, we chose the specification $\sigma_i = 0.08 + 0.32 \times (i-1)/(d-1)$, $i = 1, 2, \dots, d$. For $d > 5$, we set $\sigma_i = 0.1 + i/(2d)$, $i = 1, 2, \dots, d$. The results are given in Table 2.

4.2. Callable Multi Barrier Reverse Convertibles

A MBRC is a coupon paying security that converts into shares of the worst-performing of d underlying assets if a prespecified trigger event occurs. Let us assume that the price of the i-th underlying asset in percent of its starting value follows the risk-neutral dynamics

$$S_t^i = \begin{cases} 100 \exp\big([r - \sigma_i^2/2]t + \sigma_i W_t^i\big) & \text{for } t \in [0, T_i) \\ 100(1-\delta_i) \exp\big([r - \sigma_i^2/2]t + \sigma_i W_t^i\big) & \text{for } t \in [T_i, T] \end{cases} \tag{21}$$

for a risk-free interest rate $r \in \mathbb{R}$, volatility $\sigma_i \in (0, \infty)$, maturity $T \in (0, \infty)$, dividend payment time $T_i \in (0, T)$, dividend rate $\delta_i \in [0, \infty)$ and a d-dimensional Brownian motion W with constant instantaneous correlations $\rho_{ij} \in \mathbb{R}$ between different components W^i and W^j.

Let us consider a MBRC that pays a coupon c at each of N time points $t_n = nT/N$, $n = 1, 2, \dots, N$, and makes a time-T payment of

$$G = \begin{cases} F & \text{if } \min_{1 \leq i \leq d} \min_{1 \leq m \leq M} S_{u_m}^i > B \text{ or } \min_{1 \leq i \leq d} S_T^i > K \\ \min_{1 \leq i \leq d} S_T^i & \text{if } \min_{1 \leq i \leq d} \min_{1 \leq m \leq M} S_{u_m}^i \leq B \text{ and } \min_{1 \leq i \leq d} S_T^i \leq K, \end{cases}$$

where $F \in [0, \infty)$ is the nominal amount, $B \in [0, \infty)$ a barrier, $K \in [0, \infty)$ a strike price and u_m the end of the m-th trading day. Its value is

$$\sum_{n=1}^{N} e^{-rt_n} c + e^{-rT} \mathbb{E}\, G \tag{22}$$

and can easily be estimated with a standard Monte Carlo approximation.

A callable MBRC can be redeemed by the issuer at any of the times $t_1, t_2, \dots, t_{N-1}$ by paying back the notional. To minimize costs, the issuer will try to find a $\{t_1, t_2, \dots, T\}$-valued stopping time such that

$$\mathbb{E}\left[\sum_{n=1}^{\tau} e^{-rt_n} c + 1_{\{\tau < T\}} e^{-r\tau} F + 1_{\{\tau = T\}} e^{-rT} G\right]$$

is minimal.

Let $(X_n)_{n=1}^N$ be the $d+1$-dimensional Markov process given by $X_n^i = S_{t_n}^i$ for $i = 1, \dots, d$, and

$$X_n^{d+1} := \begin{cases} 1 & \text{if the barrier has been breached before or at time } t_n \\ 0 & \text{else.} \end{cases}$$

d	s_0	$\hat{L}$	t_L	$\hat{U}$	t_U	Point est.	95% CI	Literature
2	90	8.072	28.7	8.075	25.4	8.074	[8.060, 8.081]	8.075
2	100	13.895	28.7	13.903	25.3	13.899	[13.880, 13.910]	13.902
2	110	21.353	28.4	21.346	25.3	21.349	[21.336, 21.354]	21.345
3	90	11.290	28.8	11.283	26.3	11.287	[11.276, 11.290]	11.29
3	100	18.690	28.9	18.691	26.4	18.690	[18.673, 18.699]	18.69
3	110	27.564	27.6	27.581	26.3	27.573	[27.545, 27.591]	27.58
5	90	16.648	27.6	16.640	28.4	16.644	[16.633, 16.648]	[16.620, 16.653]
5	100	26.156	28.1	26.162	28.3	26.159	[26.138, 26.174]	[26.115, 26.164]
5	110	36.766	27.7	36.777	28.4	36.772	[36.745, 36.789]	[36.710, 36.798]
10	90	26.208	30.4	26.272	33.9	26.240	[26.189, 26.289]	
10	100	38.321	30.5	38.353	34.0	38.337	[38.300, 38.367]	
10	110	50.857	30.8	50.914	34.0	50.886	[50.834, 50.937]	
20	90	37.701	37.2	37.903	44.5	37.802	[37.681, 37.942]	
20	100	51.571	37.5	51.765	44.3	51.668	[51.549, 51.803]	
20	110	65.494	37.3	65.762	44.4	65.628	[65.470, 65.812]	
30	90	44.797	45.1	45.110	56.2	44.953	[44.777, 45.161]	
30	100	59.498	45.5	59.820	56.3	59.659	[59.476, 59.872]	
30	110	74.221	45.3	74.515	56.2	74.368	[74.196, 74.566]	
50	90	53.903	58.7	54.211	79.3	54.057	[53.883, 54.266]	
50	100	69.582	59.1	69.889	79.3	69.736	[69.560, 69.945]	
50	110	85.229	59.0	85.697	79.3	85.463	[85.204, 85.763]	
100	90	66.342	95.5	66.771	147.7	66.556	[66.321, 66.842]	
100	100	83.380	95.9	83.787	147.7	83.584	[83.357, 83.862]	
100	110	100.420	95.4	100.906	147.7	100.663	[100.394, 100.989]	
200	90	78.993	170.9	79.355	274.6	79.174	[78.971, 79.416]	
200	100	97.405	170.1	97.819	274.3	97.612	[97.381, 97.889]	
200	110	115.800	170.6	116.377	274.5	116.088	[115.774, 116.472]	
500	90	95.956	493.4	96.337	761.2	96.147	[95.934, 96.407]	
500	100	116.235	493.5	116.616	761.7	116.425	[116.210, 116.685]	
500	110	136.547	493.7	136.983	761.4	136.765	[136.521, 137.064]	

Table 1: Summary results for max-call options on d symmetric assets for parameter values of $r = 5\%$, $\delta = 10\%$, $\sigma = 20\%$, $\rho = 0$, $K = 100$, $T = 3$, $N = 9$. t_L is the number of seconds it took to train τ^Θ and compute $\hat{L}$. t_U is the computation time for $\hat{U}$ in seconds. 95% CI is the 95% confidence interval (19). The last column lists values calculated with a binomial lattice method by Andersen and Broadie (2004) for $d = 2$–3 and the 95% confidence intervals of Broadie and Cao (2008) for $d = 5$.

d	s_0	$\hat{L}$	t_L	$\hat{U}$	t_U	Point est.	95% CI	Literature
2	90	14.325	26.8	14.352	25.4	14.339	[14.299, 14.367]	
2	100	19.802	27.0	19.813	25.5	19.808	[19.772, 19.829]	
2	110	27.170	26.5	27.147	25.4	27.158	[27.138, 27.163]	
3	90	19.093	26.8	19.089	26.5	19.091	[19.065, 19.104]	
3	100	26.680	27.5	26.684	26.4	26.682	[26.648, 26.701]	
3	110	35.842	26.5	35.817	26.5	35.829	[35.806, 35.835]	
5	90	27.662	28.0	27.662	28.6	27.662	[27.630, 27.680]	[27.468, 27.686]
5	100	37.976	27.5	37.995	28.6	37.985	[37.940, 38.014]	[37.730, 38.020]
5	110	49.485	28.2	49.513	28.5	49.499	[49.445, 49.533]	[49.155, 49.531]
10	90	85.937	31.8	86.037	34.4	85.987	[85.857, 86.087]	
10	100	104.692	30.9	104.791	34.2	104.741	[104.603, 104.864]	
10	110	123.668	31.0	123.823	34.4	123.745	[123.570, 123.904]	
20	90	125.916	38.4	126.275	45.6	126.095	[125.819, 126.383]	
20	100	149.587	38.2	149.970	45.2	149.779	[149.480, 150.053]	
20	110	173.262	38.4	173.809	45.3	173.536	[173.144, 173.937]	
30	90	154.486	46.5	154.913	57.5	154.699	[154.378, 155.039]	
30	100	181.275	46.4	181.898	57.5	181.586	[181.155, 182.033]	
30	110	208.223	46.4	208.891	57.4	208.557	[208.091, 209.086]	
50	90	195.918	60.7	196.724	81.1	196.321	[195.793, 196.963]	
50	100	227.386	60.7	228.386	81.0	227.886	[227.247, 228.605]	
50	110	258.813	60.7	259.830	81.1	259.321	[258.661, 260.092]	
100	90	263.193	98.5	264.164	151.2	263.679	[263.043, 264.425]	
100	100	302.090	98.2	303.441	151.2	302.765	[301.924, 303.843]	
100	110	340.763	97.8	342.387	151.1	341.575	[340.580, 342.781]	
200	90	344.575	175.4	345.717	281.0	345.146	[344.397, 346.134]	
200	100	392.193	175.1	393.723	280.7	392.958	[391.996, 394.052]	
200	110	440.037	175.1	441.594	280.8	440.815	[439.819, 441.990]	
500	90	476.293	504.5	477.911	760.7	477.102	[476.069, 478.481]	
500	100	538.748	504.6	540.407	761.6	539.577	[538.499, 540.817]	
500	110	601.261	504.9	603.243	760.8	602.252	[600.988, 603.707]	

Table 2: Summary results for max-call options on d asymmetric assets for parameter values of $r = 5\%$, $\delta = 10\%$, $\rho = 0$, $K = 100$, $T = 3$, $N = 9$. t_L is the number of seconds it took to train τ^Θ and compute $\hat{L}$. t_U is the computation time for $\hat{U}$ in seconds. 95% CI is the 95% confidence interval (19). The last column reports the 95% confidence intervals of Broadie and Cao (2008).

Then the issuer's minimization problem can be written as

$$\inf_{\tau \in \mathcal{T}} \mathbb{E}\, g(\tau, X_\tau), \tag{23}$$

where $\mathcal{T}$ is the set of all X-stopping times and

$$g(n,x) = \begin{cases} \sum_{m=1}^{n} e^{-rt_m} c + e^{-rt_n} F & \text{if } 1 \leq n \leq N-1 \text{ or } x^{d+1} = 0 \\ \sum_{m=1}^{N} e^{-rt_m} c + e^{-rt_N} h(x) & \text{if } n = N \text{ and } x^{d+1} = 1, \end{cases}$$

where

$$h(x) = \begin{cases} F & \text{if } \min_{1 \leq i \leq d} x^i > K \\ \min_{1 \leq i \leq d} x^i & \text{if } \min_{1 \leq i \leq d} x^i \leq K. \end{cases}$$

Since the issuer cannot redeem at time 0, we trained stopping times of the form

$$\tau^\Theta = \sum_{n=1}^{N} n f^{\theta_n}(Y_n) \prod_{j=1}^{n-1} (1 - f^{\theta_j}(Y_k)) \in \mathcal{T}_1$$

for $f^{\theta_N} \equiv 1$ and $f^{\theta_1}, \ldots, f^{\theta_{N-1}} \colon \mathbb{R}^{d+1} \to \{0,1\}$ given by (8) with $I = 2$ and $q_1 = q_2 = d+40$. Since (23) is a minimization problem, τ^Θ yields an upper bound and the dual method a lower bound.

We simulated the model (21) like (20) in Subsection 4.1 with the same number of trials except that here we used the lower number $J = 1{,}024$ to estimate the dual bound. Numerical results are reported in Table 3.

4.3. Optimally Stopping a Fractional Brownian Motion

A fractional Brownian motion with Hurst parameter $H \in (0,1]$ is a continuous centered Gaussian process $(W_t^H)_{t \geq 0}$ with covariance structure

$$\mathbb{E}[W_t^H W_s^H] = \frac{1}{2}\left(t^{2H} + s^{2H} - |t-s|^{2H}\right);$$

see, e.g., Mandelbrot and Van Ness (1968) or Samoradnitsky and Taqqu (1994). For $H = 1/2$, W^H is a standard Brownian motion. So, by the optional stopping theorem, one has $\mathbb{E}\, W_\tau^{1/2} = 0$ for every $W^{1/2}$-stopping time τ bounded above by a constant; see, e.g., Grimmett and Stirzaker (2001). However, for $H \neq 1/2$, the increments of W^H are correlated – positively for $H \in (1/2, 1]$ and negatively for $H \in (0, 1/2)$. In both cases, W^H is neither a martingale nor a Markov process, and there exist bounded W^H-stopping times τ such that $\mathbb{E}\, W_\tau^H > 0$; see, e.g., Kulikov and Gusyatnikov (2016) for two classes of simple stopping rules $0 \leq \tau \leq 1$ and estimates of the corresponding expected values $\mathbb{E}\, W_\tau^H$.

To approximate the supremum

$$\sup_{0 \leq \tau \leq 1} \mathbb{E}\, W_\tau^H \tag{24}$$

d	ρ	$\hat{L}$	t_L	$\hat{U}$	t_U	Point est.	95% CI	Non-callable
2	0.6	98.235	24.9	98.252	204.1	98.243	[98.213, 98.263]	106.285
2	0.1	97.634	24.9	97.634	198.8	97.634	[97.609, 97.646]	106.112
3	0.6	96.930	26.0	96.936	212.9	96.933	[96.906, 96.948]	105.994
3	0.1	95.244	26.2	95.244	211.4	95.244	[95.216, 95.258]	105.553
5	0.6	94.865	41.0	94.880	239.2	94.872	[94.837, 94.894]	105.530
5	0.1	90.807	41.1	90.812	238.4	90.810	[90.775, 90.828]	104.496
10	0.6	91.568	71.3	91.629	300.9	91.599	[91.536, 91.645]	104.772
10	0.1	83.110	71.7	83.137	301.8	83.123	[83.078, 83.153]	102.495
15	0.6	89.558	94.9	89.653	359.8	89.606	[89.521, 89.670]	104.279
15	0.1	78.495	94.7	78.557	360.5	78.526	[78.459, 78.571]	101.209
30	0.6	86.089	158.5	86.163	534.1	86.126	[86.041, 86.180]	103.385
30	0.1	72.037	159.3	72.749	535.6	72.393	[71.830, 72.760]	99.279

Table 3: Summary results for callable MBRCs with d underlying assets for $F = K = 100$, $B = 70$, $T = 1$ year (= 252 trading days), $N = 12$, $c = 7/12$, $\delta_i = 5\%$, $T_i = 1/2$, $r = 0$, $\sigma_i = 0.2$ and $\rho_{ij} = \rho$ for $i \neq j$. t_U is the number of seconds it took to train τ^Θ and compute $\hat{U}$. t_L is the number of seconds it took to compute $\hat{L}$. The last column lists fair values of the same MBRCs without the callable feature. We estimated them by averaging 4,096,000 Monte Carlo samples of the payoff. This took between 5 (for $d = 2$) and 44 (for $d = 30$) seconds.

over all W^H-stopping times $0 \leq \tau \leq 1$, we denote $t_n = n/100$, $n = 0, 1, 2, \ldots, 100$, and introduce the 100-dimensional Markov process $(X_n)_{n=0}^{100}$ given by

$$\begin{aligned} X_0 &= (0, 0, \ldots, 0) \\ X_1 &= (W^H_{t_1}, 0, \ldots, 0) \\ X_2 &= (W^H_{t_2}, W^H_{t_1}, 0, \ldots, 0) \\ &\vdots \\ X_{100} &= (W^H_{t_{100}}, W^H_{t_{99}}, \ldots, W^H_{t_1}). \end{aligned}$$

The discretized stopping problem

$$\sup_{\tau \in \mathcal{T}} \mathbb{E}\, g(X_\tau), \tag{25}$$

where $\mathcal{T}$ is the set of all X-stopping times and $g\colon \mathbb{R}^{100} \to \mathbb{R}$ the projection $(x^1, \ldots, x^{100}) \mapsto x^1$, approximates (24) from below.

We computed estimates of (25) for $H \in \{0.01, 0.05, 0.1, 0.15, \ldots, 1\}$ by training networks of the form (8) with depth $I = 2$, $d = 100$ and $q_1 = q_2 = 140$. To simulate the vector $Y = (W^H_{t_n})_{n=0}^{100}$, we used the representation $Y = BZ$, where BB^T is the Cholesky decomposition of the covariance matrix of Y and Z a 100-dimensional random vector with independent standard normal components. We carried out 6,000 training steps with a batch size of 2,048. To estimate the lower bound L we generated $K_L = 4{,}096{,}000$ simulations of Z. For our estimate of the upper bound U, we first simulated $K_U = 1{,}024$ realizations v^k,

$k = 1, \ldots, K_U$ of Z and set $w^k = Bv^k$. Then we produced another $K_U \times J$ simulations $\tilde{v}^{k,j}$, $k = 1, \ldots, K_U$, $j = 1, \ldots, J$, of Z, and generated for all n and k, continuation paths starting from

$$z^k_n = (w^k_n, \ldots, w^k_1, 0, \ldots, 0)$$

according to

$$\tilde{z}^{k,j}_m = (\tilde{w}^{k,j}_m, \ldots, \tilde{w}^{k,j}_{n+1}, w^k_n, \ldots, w^k_1, 0 \ldots, 0), \quad m = n+1, \ldots, 100,$$

with

$$\tilde{w}^{k,j}_l = \sum_{i=1}^{n} B_{li} v^k_i + \sum_{i=n+1}^{l} B_{li} \tilde{v}^{k,j}_i, \quad l = n+1, \ldots, m.$$

For $H \in \{0.01, ..., 0.4\} \cup \{0.6, ..., 1.0\}$, we chose $J = 16{,}384$, and for $H \in \{0.45, 0.5, 0.55\}$, $J = 32{,}768$. The results are listed in Table 4 and depicted in graphical form in Figure 1. Note that for $H = 1/2$ and $H = 1$, our 95% confidence intervals contain the true values, which in these two cases, can be calculated exactly. As mentioned above, $W^{1/2}$ is a Brownian motion, and therefore, $\mathbb{E}\, W^{1/2}_\tau = 0$ for every $(W^{1/2}_{t_n})_{n=0}^{100}$-stopping time τ. On the other hand, one has[18] $W^1_t = tW^1_1$, $t \geq 0$. So, in this case, the optimal stopping time is given[18] by

$$\tau = \begin{cases} 1 & \text{if } W^1_{t_1} > 0 \\ t_1 & \text{if } W^1_{t_1} \leq 0, \end{cases}$$

and the corresponding expectation by

$$\mathbb{E}\, W^1_\tau = \mathbb{E}\Big[W^1_1 1_{\{W^1_{t_1} > 0\}} - W^1_{t_1} 1_{\{W^1_{t_1} \leq 0\}}\Big] = 0.99\, \mathbb{E}\Big[W^1_1 1_{\{W^1_1 > 0\}}\Big] = 0.99/\sqrt{2\pi} = 0.39495...$$

Moreover, it can be seen that for $H \in (1/2, 1)$, our estimates are up to three times higher than the expected payoffs generated by the heuristic stopping rules of Kulikov and Gusyatnikov (2016). For $H \in (0, 1/2)$, they are up to five times higher.

Acknowledgments

We thank Philippe Ehlers, Ariel Neufeld, Martin Stefanik, the action editor and the referees for fruitful discussions and helpful comments.

References

Charalambos D. Aliprantis and Kim C. Border. *Infinite Dimensional Analysis*. Springer, Berlin, 3rd Edition, 2006.

Leif Andersen. A simple approach to the pricing of Bermudan swaptions in the multifactor LIBOR market model. *The Journal of Computational Finance*, 3(2):5–32, 2000.

18. up to $\mathbb{P}$-almost sure equality

H	$\hat{L}$	$\hat{U}$	Point est.	95% CI
0.01	1.518	1.519	1.519	$[1.517, 1.520]$
0.05	1.293	1.293	1.293	$[1.292, 1.294]$
0.10	1.048	1.049	1.049	$[1.048, 1.050]$
0.15	0.838	0.839	0.839	$[0.838, 0.840]$
0.20	0.658	0.659	0.658	$[0.657, 0.659]$
0.25	0.501	0.504	0.503	$[0.501, 0.505]$
0.30	0.369	0.370	0.370	$[0.368, 0.371]$
0.35	0.255	0.256	0.255	$[0.254, 0.257]$
0.40	0.155	0.158	0.156	$[0.154, 0.158]$
0.45	0.067	0.075	0.071	$[0.066, 0.075]$
0.50	0.000	0.005	0.002	$[0.000, 0.005]$
0.55	0.057	0.065	0.061	$[0.057, 0.065]$
0.60	0.115	0.118	0.117	$[0.115, 0.119]$
0.65	0.163	0.165	0.164	$[0.163, 0.166]$
0.70	0.206	0.207	0.207	$[0.205, 0.208]$
0.75	0.242	0.245	0.244	$[0.242, 0.245]$
0.80	0.276	0.278	0.277	$[0.276, 0.279]$
0.85	0.308	0.309	0.308	$[0.307, 0.310]$
0.90	0.336	0.339	0.337	$[0.335, 0.339]$
0.95	0.365	0.367	0.366	$[0.365, 0.367]$
1.00	0.395	0.395	0.395	$[0.394, 0.395]$

Table 4: Estimates of $\sup_{\tau \in \{0, t_1, \ldots, 1\}} \mathbb{E}\, W_\tau^H$. For all $H \in \{0.01, 0.05, \ldots, 1\}$, it took about 430 seconds to train τ^Θ and compute $\hat{L}$. The computation of $\hat{U}$ took about 17,000 seconds for $H \in \{0.01, \ldots, 0.4\} \cup \{0.6, \ldots, 1\}$ and about 34,000 seconds for $H \in \{0.45, 0.5, 0.55\}$.

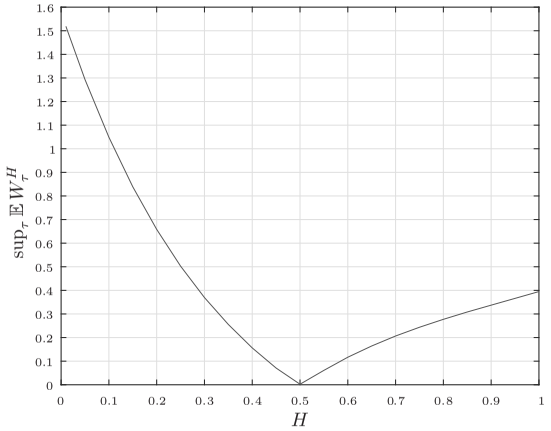


Figure 1: Estimates of $\sup_{\tau \in \{0, t_1, \ldots, 1\}} \mathbb{E}\, W_\tau^H$ for different values of H.

Leif Andersen and Mark Broadie. Primal-dual simulation algorithm for pricing multidimensional American options. *Management Science*, 50(9):1222–1234, 2004.

Vlad Bally, Gilles Pagès, and Jacques Printems. A quantization tree method for pricing and hedging multidimensional American options. *Mathematical Finance*, 15(1):119–168, 2005.

Jérôme Barraquand and Didier Martineau. Numerical valuation of high dimensional multivariate American securities. *The Journal of Financial and Quantitative Analysis*, 30(3):383–405, 1995.

Denis Belomestny. On the rates of convergence of simulation-based optimization algorithms for optimal stopping problems. *The Annals of Applied Probability*, 21(1):215–239, 2011.

Denis Belomestny. Solving optimal stopping problems via empirical dual optimization. *The Annals of Applied Probability*, 23(5):1988–2019, 2013.

Denis Belomestny, Christian Bender, and John Schoenmakers. True upper bounds for Bermudan products via non-nested Monte Carlo. *Mathematical Finance*, 19(1):53–71, 2009.

Denis Belomestny, Mark Joshi, and John Schoenmakers. Multilevel dual approach for pricing American style derivatives. *Finance and Stochastics*, 17(4):717–742, 2013.

Denis Belomestny, John Schoenmakers, Vladimir Spokoiny, and Yuri Tavyrikov. Optimal stopping via reinforced regression. *arXiv:1808.02341*, 2018.

Steffan J. Berridge and Johannes M. Schumacher. An irregular grid approach for pricing high-dimensional American options. *Journal of Computational and Applied Mathematics*, 222(1):94–111, 2008.

Phelim P. Boyle, Adam W. Kolkiewicz, and Ken Seng Tan. An improved simulation method for pricing high-dimensional American derivatives. *Mathematics and Computers in Simulation*, 62:315–322, 2003.

Mark Broadie and Menghui Cao. Improved lower and upper bound algorithms for pricing American options by simulation. *Quantitative Finance*, 8(8):845–861, 2008.

Mark Broadie and Paul Glasserman. A stochastic mesh method for pricing high-dimensional American options. *The Journal of Computational Finance*, 7(4):35–72, 2004.

Jacques F. Carriere. Valuation of the early-exercise price for options using simulations and nonparametric regression. *Insurance: Mathematics and Economics*, 19(1):19–30, 1996.

Nan Chen and Paul Glasserman. Additive and multiplicative duals for American option pricing. *Finance and Stochastics*, 11(2):153–179, 2007.

Mark H. A. Davis and Ioannis Karatzas. A deterministic approach to optimal stopping. In *Probability, Statistics and Optimisation: A Tribute to Peter Whittle (ed. Frank P. Kelly)*, pages 455–466. John Wiley & Sons, NewYork, 1994.

Vijay V. Desai, Vivek F. Farias, and Ciamac C. Moallemi. Pathwise optimization for optimal stopping problems. *Management Science*, 58(12):2292–2308, 2012.

Daniel Egloff, Michael Kohler, and Nebojsa Todorovic. A dynamic look-ahead Monte Carlo algorithm for pricing Bermudan options. *The Annals of Applied Probability*, 17(4):1138–1171, 2007.

Diego García. Convergence and biases of Monte Carlo estimates of American option prices using a parametric exercise rule. *Journal of Economic Dynamics and Control*, 27(10):1855–1879, 2003.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, *PMLR*, 9:249–256, 2010.

Geoffrey Grimmett and David Stirzaker. *Probability and Random Processes*. Oxford University Press, 3rd Edition, 2001.

Jiequn Han and Weinan E. Deep learning approximation for stochastic control problems. *Deep Reinforcement Learning Workshop, NIPS*, 2016.

Martin B. Haugh and Leonid Kogan. Pricing American options: a duality approach. *Operations Research*, 52(2):258–270, 2004.

Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, *PMLR*, 37:448–456, 2015.

Shashi Jain and Cornelis W. Oosterlee. The stochastic grid bundling method: efficient pricing of Bermudan options and their Greeks. *Applied Mathematics and Computation*, 269:412–431, 2015.

Farshid Jamshidian. The duality of optimal exercise and domineering claims: a Doob–Meyer decomposition approach to the Snell envelope. *Stochastics*, 79:27–60, 2007.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

Michael Kohler, Adam Krzyżak, and Nebojsa Todorovic. Pricing of high-dimensional American options by neural networks. *Mathematical Finance*, 20(3):383–410, 2010.

Anastasia Kolodko and John Schoenmakers. Iterative construction of the optimal Bermudan stopping time. *Finance and Stochastics*, 10(1):27–49, 2006.

Alexander V. Kulikov and Pavel P. Gusyatnikov. Stopping times for fractional Brownian motion. In *Computational Management Science*, Volume 682 of *Lecture Notes in Economics and Mathematical Systems*, pages 195–200. Springer International Publishing, 2016.

Damien Lamberton and Bernard Lapeyre. *Introduction to Stochastic Calculus Applied to Finance*. Chapman & Hall/CRC, 2nd Edition, 2008.

Jérôme Lelong. Pricing American options using martingale bases. *arXiv:1604.03317*, 2016.

Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *International Conference on Learning Representations*, 2016.

Francis A. Longstaff and Eduardo S. Schwartz. Valuing American options by simulation: a simple least-squares approach. *The Review of Financial Studies*, 14(1):113–147, 2001.

Benoit B. Mandelbrot and John W. Van Ness. Fractional Brownian motions, fractional noises and applications. *SIAM Review*, 10(4):422–437, 1968.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu et al. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Goran Peskir and Albert N. Shiryaev. *Optimal Stopping and Free-Boundary Problems*. Lectures in Mathematics. Birkhäuser Basel, 2006.

Chris Rogers. Monte Carlo valuation of American options. *Mathematical Finance*, 12(3):271–286, 2002.

Chris Rogers. Dual valuation and hedging of Bermudan options. *SIAM Journal on Financial Mathematics*, 1(1):604–608, 2010.

Gennady Samoradnitsky and Murad S. Taqqu. *Stable Non-Gaussian Random Processes*. Chapman and Hall/CRC, 1994.

John Schulman, Sergey Levine, Philipp Moritz, Michael Jordan, and Pieter Abbeel. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning, PMLR*, 37:1889–1897, 2015.

Martin Schweizer. On Bermudan options. In *Advances in Finance and Stochastics*, pages 257–270. Springer Berlin Heidelberg, 2002.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

Justin Sirignano and Konstantinos Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, 2018.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning*. The MIT Press, 1998.

James A. Tilley. Valuing American options in a path simulation model. *Transactions of the Society of Actuaries*, 45:83–104, 1993.

John N. Tsitsiklis and Benjamin Van Roy. Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks*, 12(4):694–703, 2001.

Journal of Machine Learning Research 20 (2019) 1-42 Submitted 5/18; Revised 12/18; Published 3/19

Fairness Constraints: A Flexible Approach for Fair Classification

Muhammad Bilal Zafar mzafar@mpi-sws.org
Bosch Center for Artificial Intelligence and Max Planck Institute for Software Systems
Saarbrücken, Germany

Isabel Valera isabel.valera@tue.mpg.de
Max Planck Institute for Intelligent Systems
Tübingen, Germany

Manuel Gomez-Rodriguez manuelgr@mpi-sws.org
Max Planck Institute for Software Systems
Kaiserslautern, Germany

Krishna P. Gummadi gummadi@mpi-sws.org
Max Planck Institute for Software Systems
Saarbrücken, Germany

Editor: Moritz Hardt

Abstract

Algorithmic decision making is employed in an increasing number of real-world applications to aid human decision making. While it has shown considerable promise in terms of improved decision accuracy, in some scenarios, its outcomes have been also shown to impose an unfair (dis)advantage on people from certain social groups (*e.g.*, women, blacks). In this context, there is a need for computational techniques to limit unfairness in algorithmic decision making. In this work, we take a step forward to fulfill that need and introduce a flexible constraint-based framework to enable the design of fair margin-based classifiers. The main technical innovation of our framework is a general and intuitive measure of decision boundary unfairness, which serves as a tractable proxy to several of the most popular computational definitions of unfairness from the literature. Leveraging our measure, we can reduce the design of fair margin-based classifiers to adding tractable constraints on their decision boundaries. Experiments on multiple synthetic and real-world datasets show that our framework is able to successfully limit unfairness, often at a small cost in terms of accuracy.

Keywords: Supervised learning, margin-based classifiers, fairness, discrimination, disparate impact.

1. Introduction

Algorithmic decision making systems are assisting, or in some cases even replacing, human decision making in an increasing number of application domains. Examples of such domains include credit approval (FICO, 2017), criminal risk assessment (Perry, 2013), hiring (Posse, 2016; Taylor, 2016) and education (Romero and Ventura, 2011). By leveraging vast amounts of (training) data and modern machine learning methods, algorithmic decision systems are able to provide highly accurate predictions, often surpassing even human accuracy (Kleinberg et al., 2017a; Liu et al., 2017). However, recent studies by civil organizations (Bhandari, 2016), governments (Podesta et al., 2014; Muñoz et al., 2016), and researchers (Sweeney, 2013) alike have raised concerns about the unfairness of these algorithmic decision systems towards people from certain social groups (*e.g.*, women, blacks). Importantly, in various countries, these concerns are often grounded on anti-discrimination laws which prohibit unfair treatment of people based on one or more *sensitive features*, such as gender and race (Civil Rights Act, 1964; Barocas and Selbst, 2016).

To overcome the above concerns, a number of recent studies in the emerging field of ethical machine learning have proposed mechanisms to ensure that algorithmic decision systems do not lead to unfair outcomes (Kamiran and Calders, 2009; Corbett-Davies et al., 2017; Hardt et al., 2016; Feldman et al., 2015; Zafar et al., 2017b,a; Zemel et al., 2013; Dwork et al., 2012). In doing so, they have have measured the (un)fairness of a decision making process through the distribution of its outcomes among people from different sensitive feature groups (*e.g.*, men, women). More specifically, they have typically adopted one of the following three notions of (un)fairness: [1]

— ***Disparate treatment***: a decision making process suffers from disparate treatment if its outcomes change based on a change in the sensitive feature value with all other features being the same.

— ***Disparate impact***: a decision making process suffers from disparate impact if it grants disproportionately large fraction of beneficial (or positive classification) outcomes to certain sensitive feature groups (*e.g.*, men, women).

— ***Disparate mistreatment***: a decision making process suffers from disparate mistreatment if its accuracy (or error rate) is different for different sensitive feature groups.

However, the mechanisms proposed in prior studies typically lack flexibility with respect to one or more of the following aspects:

(i) They are specifically designed for only one of the above notions. As a consequence, they cannot accommodate more than one of them *simultaneously* (*e.g.*, disparate treatment and disparate impact).

(ii) They cannot ensure fairness with respect to multiple sensitive features *simultaneously* (*e.g.*, gender and race).

(iii) They are only limited to a narrow range of classification models (*e.g.*, logistic regression).

In this work, we propose a flexible framework to design a variety of fair classifiers that do not suffer from the above limitations. More specifically, for any convex boundary-based classifier, our framework defines an intuitive measure of decision boundary unfairness: the covariance between the sensitive features and the signed distance between the (non sensitive) feature vectors and the decision boundary of the classifier for a subset the subjects which depends on the fairness notion of interest. This measure can be readily incorporated into the classifier formulation in the form of convex of convex-concave constraints, one per sensitive feature or fairness notion, which can be efficiently solved using well-known meth-

. A preliminary version of this work appeared in Zafar et al. (2017a,b).

. Open-source code implementation is available at: http://fate-computing.mpi-sws.org/.

1. As we later discuss in Section 2, the same notions are often referred to by different names by different studies.

©2019 Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, Krishna P. Gummadi.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-262.html.

ods (Boyd and Vandenberghe, 2004; Shen et al., 2016). Interestingly, our framework also allows for a dual formulation which maximizes fairness under accuracy constraints and, as a consequence, it ensures compliance with the business necessity clause of anti-discrimination doctrines (Barocas and Selbst, 2016), an aspect not considered by prior studies. Experiments on multiple synthetic and real-world datasets show that our framework is able to successfully limit disparate treatment, disparate impact and disparate mistreatment, often at a small cost in terms of accuracy, and it provides more flexibility than state-of-the-art methods (see Table 3).

The rest of the paper is organized as follows. First, we first revisit several well-known fairness notions from the literature and discuss the type of scenarios each notion is most suitable for (Section 2). Then, we formally state the problem of fairness-aware classification (Section 3) and describe our framework (Section 4). Later, we experiment with several datasets, including comparisons with related methodologies to highlight the effectiveness of our mechanism in controlling unfairness (Section 5). Finally, we conclude with a review of the related work on unfairness in algorithmic decision making and strategies proposed to mitigate unfairness (Section 6) as well as a discussion of future work (Section 7).

2. Background on different notions of (un)fairness

In this section, we revisit three of the most popular notions of fairness used in the machine learning literature: disparate treatment, disparate impact, and disparate mistreatment. More specifically, we first elaborate on each of these notions separately in the context of automated decision making systems and then highlight the differences between them.

— **Disparate treatment.** A decision making system suffers from disparate treatment if it provides different outputs for groups of people with the same (or similar) values of non-sensitive features but different values of sensitive features (Barocas and Selbst, 2016). In other words, (partly) basing the decision outcomes on the sensitive feature value amounts to disparate treatment. [2] This notion has been also referred to as *direct discrimination* (Pedreschi et al., 2008).

Figure 1 provides examples of binary classifiers with and without disparate treatment in a *stop-and-frisk* (Gelman et al., 2007) application. In all cases, the classifiers need to decide whether to stop a pedestrian on the suspicion of possessing an illegal weapon based on a set of features such as bulge in clothing and proximity to a crime scene. The "ground truth" on whether a pedestrian actually possesses an illegal weapon is also shown. We deem classifiers $\mathbf{C_2}$ and $\mathbf{C_3}$ to be unfair due to disparate treatment since $\mathbf{C_2}$'s ($\mathbf{C_3}$'s) decisions for *Male* 1 and *Female* 1 (*Male* 2 and *Female* 2) are different even though they have the

2. Technically, the disparate treatment doctrine tries to counter *explicit* as well as *intentional* discrimination (Barocas and Selbst, 2016). It follows from the specification of disparate treatment that a decision maker with an intent to discriminate could try to disadvantage a group with a certain sensitive feature value (*e.g.*, a specific race group) not by *explicitly using the sensitive feature* itself, but by *intentionally basing decisions on a correlated feature* (*e.g.*, the non-sensitive feature location might be correlated with the sensitive feature race). This practice is often referred to as *redlining* in the US anti-discrimination law and also qualifies as disparate treatment (Gano, 2017). However, such hidden intentional disparate treatment maybe be hard to detect, and some authors argue that disparate impact might be a more suitable framework for detecting such covert discrimination (Siegel, 2014). Hence, in this paper, when discussing disparate treatment, we will focus only on *explicit* disparate treatment.

User Attributes			Ground Truth (Has Weapon)	Classifier's Decision to Stop		
Sensitive	Non-sensitive					
Gender	Clothing Bulge	Prox. Crime		C_1	C_2	C_3
Male 1	1	1	✓	1	1	1
Male 2	1	0	✓	1	1	0
Male 3	0	1	✗	1	0	1
Female 1	1	1	✓	1	0	1
Female 2	1	0	✗	1	1	1
Female 3	0	0	✓	0	1	0

	Disp. Treat.	Disp. Imp.	Disp. Mist.
$\mathbf{C_1}$	✗	✓	✓
$\mathbf{C_2}$	✓	✗	✓
$\mathbf{C_3}$	✓	✗	✗

Figure 1: Decisions of three fictitious classifiers ($\mathbf{C_1}$, $\mathbf{C_2}$ and $\mathbf{C_3}$) on whether (1) or not (0) to stop a pedestrian on the suspicion of possessing an illegal weapon. Gender is a sensitive attribute, whereas the other two attributes (suspicious bulge in clothing and proximity to a crime scene) are non-sensitive. Ground truth on whether the person is actually in possession of an illegal weapon is also shown. $\mathbf{C_1}$ has no disparate treatment. It has disparate impact because it stops disproportional fractions of males and females (1.0 and 0.66, respectively). It has disparate mistreatment because it has different false negative rates (0.0 and 0.5, respectively) for males and females. $\mathbf{C_2}$ has disparate treatment since, of a male and a female having same non-sensitive attributes (Male 1 and Female 1), it stops only one of them. It has no disparate impact, since it stops equal fractions (0.66) of males and females. It has disparate mistreatment because of different false positive (0.0 and 1.0, respectively) and false negative rates (0.0 and 0.5, respectively) for males and females. $\mathbf{C_3}$ has disparate treatment and no disparate impact. It has no disparate mistreatment because both false positive rates (1.0) and false negative rates (0.5) for males and females are equal.

same values of non-sensitive features. On the other hand, classifier $\mathbf{C_1}$ does not suffer from disparate treatment since, with all feature values except gender being equal, its decisions are identical.

— **Disparate impact.** A decision making system suffers from disparate impact if it provides outputs that benefit (hurt) a group of people sharing a value of a sensitive feature more frequently than other groups of people (Barocas and Selbst, 2016). This notion has been also referred to as *statistical parity* (Corbett-Davies et al., 2017) or *demographic parity* (Dwork et al., 2012).

Similarly as with disparate treatment, Figure 1 provides examples of binary classifiers with and without disparate impact in a stop-and-frisk application. Under the assumption that a pedestrian benefits from a decision of not being stopped, we deem classifier $\mathbf{C_1}$ to be unfair due to disparate impact because the fraction of males and females that were stopped are different (1.0 and 0.66, respectively), where the latter benefit from a decision of not being stopped. On the other hand, classifiers $\mathbf{C_2}$ and $\mathbf{C_3}$ do not suffer from disparate impact because the fractions of males and females that were stopped are the same (0.66).

— **Disparate mistreatment.** A decision making system suffers from disparate mistreatment if it achieves different classification accuracy (or conversely, error rate) for groups of people sharing different values of a sensitive feature (Zafar et al., 2017a). This notion has been also referred to as *equality of opportunity* (Hardt et al., 2016) and *predictive equality* (Corbett-Davies et al., 2017). In addition to overall classification accuracy, this notion has been also particularized to *different types of misclassifications*, *i.e.*, false positives and

false negatives. In that context, a decision making system suffers from disparate mistreatment if individual misclassification rates (*e.g.*, false positive rate, false negative rate) are different for groups of people sharing different values of a sensitive feature.

In Figure 1, we deem classifiers $\mathbf{C_1}$ and $\mathbf{C_2}$ to be unfair due to disparate mistreatment since their rate of erroneous decisions for males and females are different: $\mathbf{C_1}$ has different false negative rates for males and females (0.0 and 0.5, respectively), whereas $\mathbf{C_2}$ has different false positive rates (0.0 and 1.0) as well as different false negative rates (0.0 and 0.5) for males and females. Finally, classifier $\mathbf{C_3}$ does not suffer from disparate mistreatment because it has the same false negative and false positive rates for males and females.

Differences among (un)fairness notions. The above (un)fairness notions account for either *direct* (or *intentional*) and *indirect* (or *unintentional*) unfairness (Altman, 2016). More specifically, *disparate treatment accounts for direct unfairness*, *i.e.*, a situation where a decision making process directly (or intentionally) uses the sensitive feature information to put a group of people sharing a value of a sensitive feature on relative disadvantage. In this way, removing disparate treatment corresponds to a very intuitive notion of fairness: two otherwise similar persons should not be treated differently solely because of the difference in gender. On the other hand, *disparate impact and disparate mistreatment account for indirect unfairness*, *i.e.*, a situation where the decision making process can indirectly or unintentionally leverage the correlation between sensitive features and class labels to put a sensitive feature group at relative disadvantage (through low beneficial outcome rate under disparate impact and through high misclassification rate under disparate mistreatment).

Moreover, while disparate impact and disparate mistreatment both account for indirect unfairness, their application scenarios strongly differ. Unlike in the case of disparate mistreatment, the notion of disparate impact is independent of the "ground truth" information about the decisions, *i.e.*, whether or not the decisions are correct or valid. Thus, the notion of disparate impact is particularly appealing in application scenarios where ground truth information for decisions does not exist and the historical decisions used during training are not reliable and thus cannot be trusted (Barocas and Selbst, 2016). Unreliability of historical decisions for automated decision making systems is particularly concerning in scenarios like recruiting or loan approvals, where biased judgments by humans in the past may be used when training classifiers for the future. In such application scenarios, it is hard to distinguish correct and incorrect decisions, making it hard to assess or use disparate mistreatment as a notion of fairness.

However, in scenarios where ground truth information for decisions can be obtained, disparate impact can be quite misleading as a notion of fairness. That is, in scenarios where the validity of decisions can be reliably ascertained, it would be possible to distinguish disproportionality in decision outcomes for sensitive groups that arises from justifiable reasons (*e.g.*, qualification of the candidates) and disproportionality that arises for non-justifiable reasons (*i.e.*, discrimination against certain groups). By requiring decision outcomes to be proportional, disparate impact risks introducing reverse-discrimination against qualified candidates. Such practices have previously been deemed unlawful (*e.g.*, *Ricci vs. DeStefano, 2009*). In contrast, when the correctness of decisions can be determined, disparate mistreatment can not only be accurately assessed, but also avoids reverse-discrimination, making it a more appealing notion of fairness (Hardt et al., 2016; Zafar et al., 2017a).

3. Fairness in classification

In a binary classification task, [3] one aims to find a mapping function $f(\boldsymbol{x})$ between user feature vectors $\boldsymbol{x} \in \mathbb{R}^d$ and class labels $y \in \{-1, 1\}$. This task is achieved by utilizing a training set, $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, to construct a mapping that works *well* on an *unseen* test set. For decision boundary-based classifiers, finding this mapping usually reduces to building a decision boundary in feature space that separates users in the training set according to their class labels. One typically looks for a decision boundary, defined by a set of parameters $\boldsymbol{\theta}^*$, that achieves the greatest classification accuracy in a test set, by minimizing a loss function over a training set $L(\boldsymbol{\theta})$, *i.e.*, $\boldsymbol{\theta}^* = \text{argmin}_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$. Then, given an *unseen* feature vector $\boldsymbol{x}_i$ from the test set, the classifier predicts the label $\hat{y}_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) = 1$ if $d_{\boldsymbol{\theta}^*}(\boldsymbol{x}_i) \geq 0$ and $\hat{y}_i = -1$ otherwise, where $d_{\boldsymbol{\theta}^*}(\boldsymbol{x})$ denotes the signed distance from the feature vector $\boldsymbol{x}$ to the decision boundary.

In the context of fairness in binary classification, each user also has an associated sensitive feature $z \in \{0, 1\}$[4] and the goal is finding a mapping (or decision boundary) that provides both accurate predictions and fairness guarantees. More formally, we can express the absence of disparate treatment, disparate impact and disparate mistreatment in a binary classifier as follows:

1. No disparate treatment. A binary classifier does not suffer from disparate treatment if the probability that the classifier outputs a specific value of $\hat{y}$ given a feature vector $\boldsymbol{x}$ does not change after observing the sensitive feature z, *i.e.*,

$$P(\hat{y}|\boldsymbol{x}, z) = P(\hat{y}|\boldsymbol{x}). \tag{3.1}$$

2. No disparate impact. A binary classifier does not suffer from disparate impact if the probability that a classifier assigns a user to the positive class, $\hat{y} = 1$, is the same for both values of the sensitive feature z, *i.e.*,

$$P(\hat{y} = 1|z = 0) = P(\hat{y} = 1|z = 1). \tag{3.2}$$

Notice that neither disparate treatment nor disparate impact depend on the subjects' ground truth label (y).

3. No disparate mistreatment. A binary classifier does not suffer from disparate mistreatment if the misclassification rates for different groups of people having different values of the sensitive feature z are the same. Table 1 describes various ways of measuring misclassification rates. Specifically, misclassification rates can be measured as fractions over the *class distribution in the ground truth labels*, *i.e.*, as false positive and false negative rates, or over the *class distribution in the predicted labels*, *i.e.*, as false omission and false discovery rates. [5] Consequently, the absence of disparate mistreatment in a binary classification task

3. For simplicity, we consider binary classification tasks in this work. However, our ideas can be easily extended to m-ary classification.
4. For ease of exposition, we assume z to be unidimensional and binary, however, our setup can be easily generalized to categorical as well as multiple sensitive features.
5. In prediction tasks where a positive prediction entails a large cost (*e.g.*, cost involved in the treatment of a disease), one might be more interested in measuring error rates as fractions over the class distribution

		Predicted Label		
		$\hat{y}=1$	$\hat{y}=-1$	
True Label	$y=1$	True positive	False negative	$P(\hat{y}\neq y \vert y=1)$ False Negative Rate
	$y=-1$	False positive	True negative	$P(\hat{y}\neq y \vert y=-1)$ False Positive Rate
		$P(\hat{y}\neq y \vert \hat{y}=1)$ False Discovery Rate	$P(\hat{y}\neq y \vert \hat{y}=-1)$ False Omission Rate	$P(\hat{y}\neq y)$ Overall Misclass. Rate

Table 1: In addition to the overall misclassification rate, error rates can be measured in two different ways: false negative rate and false positive rate are defined as fractions over the *class distribution in the ground truth labels*, or true labels. On the other hand, false discovery rate and false omission rate are defined as fractions over the *class distribution in the predicted labels*.

can be specified with respect to the different misclassification measures as follows:

— ***Overall misclassification rate (OMR)***:

$$P(\hat{y}\neq y|z=0) = P(\hat{y}\neq y|z=1), \tag{3.3}$$

— ***False positive rate (FPR)***:

$$P(\hat{y}\neq y|y=-1,z=0) = P(\hat{y}\neq y|y=-1,z=1), \tag{3.4}$$

— ***False negative rate (FNR)***:

$$P(\hat{y}\neq y|y=1,z=0) = P(\hat{y}\neq y|y=1,z=1), \tag{3.5}$$

— ***False omission rate (FOR)***:

$$P(\hat{y}\neq y|\hat{y}=-1,z=0) = P(\hat{y}\neq y|\hat{y}=-1,z=1), \tag{3.6}$$

— ***False discovery rates (FDR)***:

$$P(\hat{y}\neq y|\hat{y}=1,z=0) = P(\hat{y}\neq y|\hat{y}=1,z=1). \tag{3.7}$$

Remarks. We would like to note that the formal criteria to avoid disparate impact and disparate mistreatment, as given by Eqs.(3.2–3.7), represent equality in certain *group-conditional probabilities* for the two sensitive feature groups, *i.e.*, $P(\cdot|z=0)$ and $P(\cdot|z=1)$. For example, the disparate impact criterion in Eq. (3.2) requires the group-conditional probabilities of positive class outcomes to be the same for users with $z=0$ and $z=1$, whereas the disparate mistreatment criterion in Eq. (3.3) requires the group-conditional probabilities of users being misclassified to be the same. In Section 4, we will show that these group-conditional equalities can be relaxed by various decision boundary covariances to facilitate efficient training of fair classifiers.

Finally, we would also like to highlight that some recent works have explored the cost of achieving fairness in a decision making process and pointed out some inherent tensions between fairness and accuracy (Chouldechova, 2016; Corbett-Davies et al., 2017; Kleinberg et al., 2017b; Menon and Williamson, 2017). In fact, this cost can become prohibitively high if one aims to achieve multiple fairness criteria *simultaneously*. For example, Chouldechova (2016) and Kleinberg et al. (2017b) have recently shown that, when the fraction of users with positive class labels differ between members of different sensitive attribute value groups, it is impossible to construct classifiers that are equally *well-calibrated* (where well-calibration essentially measures the false discovery and false omission rates of a classifier) and also satisfy the equal false positive and false negative rate criterion (except for a "dumb", or null, classifier that assign all examples to a single class). Pleiss et al. (2017) further expand on this impossibility result. These results suggest that satisfying all five criteria of disparate mistreatment (Table 1) simultaneously is impossible when the underlying distribution of data is different for different groups. Similarly, Kleinberg et al. (2017b) show that when the fraction of users with positive class labels differ between members of different sensitive attribute value groups, it is impossible to satisfy disparate impact and disparate mistreatment simultaneously—where disparate mistreatment is defined in terms of false positive and false negative rates, in terms of false discovery and false omission rates, or both. However, in practice, it may still be interesting to explore the best, even if imperfect, extent of fairness a classifier can achieve.

4. Our fair classification framework

In this section, we present our framework to design boundary-based classifiers which are free of disparate treatment, disparate impact and disparate mistreatment, as defined in Section 3.

4.1. Fairness criteria as constraints during training

To design a *fair* convex boundary-based classifier, one can think of including fairness constraints during training. More specifically, minimizing the corresponding loss function under fairness constraints, *i.e.*,

$$\begin{aligned} &\text{minimize} && L(\boldsymbol{\theta}) && \} \text{ Classifier loss function} \\ &\text{subject to} && P(.|z=0) = P(.|z=1) && \} \text{ Fairness constraints,} \end{aligned} \tag{4.1}$$

where the probabilities in the constraint(s) can be replaced with the respective disparate impact and disparate mistreatment criteria in Eqs. (3.2–3.7). Here, note that, if $z \notin \boldsymbol{x}$ (*i.e.*, $\boldsymbol{x}$ and z consist of disjoint feature sets), the resulting classifier does not suffer from disparate treatment since z is not used during test (*i.e.*, at decision time).

Due to its flexibility, the above formulation exhibits several advantages:

(i) it can satisfy both disparate impact and (any version of) disparate mistreatment by including the corresponding constraints. Disparate treatment can be achieved by excluding the sensitive features z from $\boldsymbol{x}$ so that they are not used during test.

in the *predicted labels*, rather than over the class distribution in the *ground truth labels*, *e.g.*, to ensure that the false discovery rates, instead of false positive rates, for all groups are the same.

(ii) it can accommodate any convex decision boundary based classifier.
(iii) it can ensure fairness with respect to multiple sensitive features (*e.g.*, gender, race) by including constraints for each sensitive feature separately.

Unfortunately, solving the formulation given by Eq. (4.1) is very challenging. First, for many such classifiers (*e.g.*, SVM), the probabilities in Eqs. (3.2–3.7) are a non-convex function of the classifier parameters $\boldsymbol{\theta}$, therefore leading to non-convex formulations, which are difficult to solve efficiently. Second, as long as the user feature vectors lie on the same side of the decision boundary, the probabilities are invariant to changes in the decision boundary. In other words, the probabilities are functions having saddle points. The presence of saddle points furthers complicate the procedure for solving non-convex optimization problems (Dauphin et al., 2014).

To overcome these challenges, we next introduce a relaxation of the group-conditional probability constraints given by Eqs. (3.2–3.7) using a novel covariance measure of decision boundary unfairness.

4.2. Designing fair classifiers using decision boundary covariances

In this section, we first introduce our covariance measure of decision boundary unfairness in the context of disparate impact, use this measure to design classifiers free of disparate impact, and then generalize our measure to disparate mistreatment.

4.2.1. Disparate impact

We measure the decision boundary (un)fairness due to disparate impact by means of the covariance between the users' sensitive attribute z and the signed distance from the users' feature vectors to the decision boundary $d_{\boldsymbol{\theta}}(\boldsymbol{x})$, *i.e.*,

$$\text{Cov}_{DI}(z, d_{\boldsymbol{\theta}}(\boldsymbol{x})) = \mathbb{E}[(z-\bar{z})d_{\boldsymbol{\theta}}(\boldsymbol{x})] - \mathbb{E}[(z-\bar{z})]\bar{d}_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx \frac{1}{N}\sum_{(\boldsymbol{x},z)\in\mathcal{D}} (z-\bar{z})\, d_{\boldsymbol{\theta}}(\boldsymbol{x}), \quad (4.2)$$

where $\mathbb{E}[(z-\bar{z})]\bar{d}_{\boldsymbol{\theta}}(\boldsymbol{x})$ cancels out since $\mathbb{E}[(z-\bar{z})] = 0$. Note that, if a decision boundary satisfies Eq. (3.2), *i.e.*, $P(d_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq 0|z=0) = P(d_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq 0|z=1)$, then the (empirical) covariance defined above will be (approximately) zero (for a sufficiently large training set)[6]. Moreover, in contrast to the group-conditional probabilities given by Eq. (3.2) the decision boundary covariance in Eq. (4.2) is a convex function with respect to the decision boundary parameters $\boldsymbol{\theta}$ because $d_{\boldsymbol{\theta}}(\boldsymbol{x})$ is convex with respect to $\boldsymbol{\theta}$ for all linear, convex boundary-based classifiers. [7] Hence, it can be easily included in the formulation of these classifiers without increasing the complexity of their training.

6. Note that the converse is not true, that is why we call our covariance measure a proxy.
7. For non-linear convex boundary-based classifiers like non-linear SVM, the equivalent of $d_{\boldsymbol{\theta}}(\boldsymbol{x})$ (via Representer Theorem) is still convex in the corresponding reproducing kernel Hilbert space as we will discuss shortly.

More specifically, to train a classifier free of disparate impact, one can replace the (intractable) constraint in Eq. (4.1) by an alternative constraint including the decision boundary covariance constraint as follows:

$$\begin{aligned} &\text{minimize} && L(\boldsymbol{\theta}) \\ &\text{subject to} && \tfrac{1}{N}\textstyle\sum_{(\boldsymbol{x},z)\in\mathcal{D}} (z-\bar{z})\, d_{\boldsymbol{\theta}}(\boldsymbol{x}) \leq c, \\ &&& \tfrac{1}{N}\textstyle\sum_{(\boldsymbol{x},z)\in\mathcal{D}} (z-\bar{z})\, d_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq -c, \end{aligned} \quad (4.3)$$

where $c \in \mathbb{R}^+$ is a given threshold, which trades off accuracy and unfairness due to disparate impact. Furthermore, note that since the above optimization problem is convex, our scheme ensures that the trade-off between the classifier loss function and decision boundary covariance is Pareto optimal.

When considering *multiple sensitive features* (*e.g.*, gender, race), one can include constraints for each sensitive feature separately. For *polyvalent sensitive features* having $k \geq 2$ values, one can first convert the sensitive feature into k binary sensitive features using one hot encoding, and then add constraints for each of the k sensitive features. To avoid "fairness-gerrymandering" (Kearns et al., 2018) when considering multiple sensitive features, one could construct all possible combinations of the sensitive feature values (*e.g.*, white man, black women) and add constraints for each combination separately.

4.2.2. Disparate Mistreatment

We can naturally extend our covariance measure to (un)fairness due to disparate mistreatment. More specifically, for the case disparate mistreatment with respect to the overall misclassification rate, we compute the covariance between the users' sensitive attributes and the signed distance between the feature vectors of *misclassified* users and the classifier decision boundary, *i.e.*,

$$\text{Cov}_{OMR}(z, g_{\boldsymbol{\theta}}(y,\boldsymbol{x})) = \mathbb{E}[(z-\bar{z})(g_{\boldsymbol{\theta}}(y,\boldsymbol{x}) - \bar{g}_{\boldsymbol{\theta}}(y,\boldsymbol{x}))] \approx \frac{1}{N}\sum_{(\boldsymbol{x},y,z)\in\mathcal{D}} (z-\bar{z})\, g_{\boldsymbol{\theta}}(y,\boldsymbol{x}), \quad (4.4)$$

where $g_{\boldsymbol{\theta}}(y,\boldsymbol{x}) = \min(0, y d_{\boldsymbol{\theta}}(\boldsymbol{x}))$ and the term $\mathbb{E}[(z-\bar{z})]\bar{g}_{\boldsymbol{\theta}}(\boldsymbol{x})$ cancels out since $\mathbb{E}[(z-\bar{z})] = 0$. As in the case of disparate impact, if a decision boundary satisfies Eq. (3.3), then the (empirical) covariance defined above will be (approximately) zero (for a sufficiently large training set) and we can train a classifier free of disparate mistreatment with respect to overall misclassification rate by replacing the (intractable) constraint in Eq. (4.1) by an alternative constraint as follows:

$$\begin{aligned} &\text{minimize} && L(\boldsymbol{\theta}) \\ &\text{subject to} && \tfrac{1}{N}\textstyle\sum_{(\boldsymbol{x},y,z)\in\mathcal{D}} (z-\bar{z})\, g_{\boldsymbol{\theta}}(y,\boldsymbol{x}) \leq c, \\ &&& \tfrac{1}{N}\textstyle\sum_{(\boldsymbol{x},y,z)\in\mathcal{D}} (z-\bar{z})\, g_{\boldsymbol{\theta}}(y,\boldsymbol{x}) \geq -c, \end{aligned} \quad (4.5)$$

where $c \in \mathbb{R}^+$ is a given threshold, which trades off accuracy and unfairness due to disparate mistreatment. Similarly, we can define the above covariance measure for disparate mistreatment with respect to false positive rates, false negative rates, false omission rates

or false discovery rates. For example, for false positive rates, one needs to consider the set of *misclassified* users with (ground-truth) negative labels ($\mathcal{D}^-$), *i.e.*,

$$\text{Cov}_{FPR}(z, g_{\boldsymbol{\theta}}(y, \boldsymbol{x})) \approx \frac{1}{N^-} \sum_{(\boldsymbol{x},y,z)\in\mathcal{D}^-} (z - \bar{z})\, g_{\boldsymbol{\theta}}(y, \boldsymbol{x}), \tag{4.6}$$

where N^- represents the size of $\mathcal{D}^-$.

In contrast with the covariance measure in the case of disparate impact, defined by Eq. (4.2), the above covariance measures are not convex. Fortunately, the covariance constraints for disparate mistreatment with respect to overall misclassification rates, false positive rates and false negative rates can be easily converted into convex-concave constraints, which can be handled efficiently (Shen et al., 2016), as follows. Consider the constraints in Eq. (4.5), *i.e.*,

$$\sum_{(\boldsymbol{x},y,z)\in\mathcal{D}} (z - \bar{z})\, g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) \sim c,$$

where $\sim$ denotes '$\geq$' and '$\leq$' and, without loss of generality, we just left out the constant term $\frac{1}{N}$. Then, we can split the sum in the above expression into two terms:

$$\sum_{(\boldsymbol{x},y)\in\mathcal{D}_0} (0 - \bar{z})\, g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) + \sum_{(\boldsymbol{x},y)\in\mathcal{D}_1} (1 - \bar{z})\, g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) \sim c, \tag{4.7}$$

where $\mathcal{D}_0$ and $\mathcal{D}_1$ are the subsets of the training dataset $\mathcal{D}$ taking values $z = 0$ and $z = 1$, respectively. Define $N_0 = |\mathcal{D}_0|$ and $N_1 = |\mathcal{D}_1|$, then one can write $\bar{z} = \frac{(0\times N_0)+(1\times N_1)}{N} = \frac{N_1}{N}$ and rewrite Eq. (4.7) as:

$$\frac{-N_1}{N} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_0} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) + \frac{N_0}{N} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_1} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) \sim c,$$

which, given that $g_{\boldsymbol{\theta}}(y, \boldsymbol{x})$ is convex in $\boldsymbol{\theta}$, results into a convex-concave (or, difference of convex) function. Finally, we can rewrite the problem defined by (4.5) as:

$$\begin{aligned} &\text{minimize} && L(\boldsymbol{\theta}) \\ &\text{subject to} && \tfrac{-N_1}{N} \textstyle\sum_{(\boldsymbol{x},y)\in\mathcal{D}_0} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) + \tfrac{N_0}{N} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_1} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) \leq c \\ &&& \tfrac{-N_1}{N} \textstyle\sum_{(\boldsymbol{x},y)\in\mathcal{D}_0} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) + \tfrac{N_0}{N} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_1} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) \geq -c, \end{aligned} \tag{4.8}$$

which is a Disciplined Convex-Concave Program (DCCP) for any convex loss $L(\boldsymbol{\theta})$, and can be efficiently solved using well-known heuristics (Shen et al., 2016). Note that the non-convexity of the formulation in Eq. (4.8) implies that the resulting tradeoff between the classifier loss function and decision boundary covariance is not guarenteed to be Pareto optimal. This is in contrast with the convex formulation in Eq. (4.3). However, as we show via comparisons with related methods in Section 5.2.2, Eq. (4.8) can still lead to competitive results in comparison with the state-of-the-art.

Proceeding similarly, we can convert the covariance constraints for disparate mistreatment with respect to false positive rates and false negative rates to convex-concave constraints. For example, Eq. (4.1) can be rewritten to impose equality in false positive rates

as:

$$\begin{aligned} &\text{minimize} && L(\boldsymbol{\theta}) \\ &\text{subject to} && \tfrac{-N_1^-}{N^-} \textstyle\sum_{(\boldsymbol{x},y)\in\mathcal{D}_0^-} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) + \tfrac{N_0^-}{N^-} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_1^-} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) \leq c \\ &&& \tfrac{-N_1^-}{N^-} \textstyle\sum_{(\boldsymbol{x},y)\in\mathcal{D}_0^-} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) + \tfrac{N_0^-}{N^-} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_1^-} g_{\boldsymbol{\theta}}(y, \boldsymbol{x}) \geq -c, \end{aligned} \tag{4.9}$$

where $\mathcal{D}_i^-$ is the subset of the training data with $z = i$ and $y = -1$, and $N_i^- = |\mathcal{D}_i^-|$. Note that unlike in Zafar et al. (2017a), we define the false positive rate covariance (Eq. (4.6)) only over the ground truth negative dataset instead of the whole dataset. In cases where $\frac{N_0^-}{N_0} \neq \frac{N_1^-}{N_1}$ (or in other words, the base-rates are different for the two sensitive feature groups), the false positive rate covariance as defined by Zafar et al. (2017a) would not fully remove disparate mistreatment.

Finally, while the covariance constraints for disparate mistreatment with respect to false omission and false discovery rates can be readily defined, the corresponding constraints cannot be easily converted into convex-concave constraints. Handling such constraints efficiently is left as an interesting venue for future work.

4.3. Accounting for the business necessity clause

In the previous section, we have used covariance constraints to design classifiers that maximize accuracy under fairness constraints. However, if the underlying correlation between the class labels and the sensitive attributes in the training set is very high, enforcing these constraints may results in underwhelming performance (accuracy) and thus be unacceptable in terms of business objectives. This is particularly concerning in the case of disparate impact, where a "business necessity" clause has been argued for—an employer would need to ensure that the decision making causes *least possible* disparate impact under the given performance (accuracy) constraints (Barocas and Selbst, 2016).

Fortunately we can account for the above mentioned "business necessity" clause in disparate impact using an alternative formulation that maximizes fairness (minimizes disparate impact) subject to accuracy constraints. More specifically, we can find the decision boundary parameters $\boldsymbol{\theta}$ by minimizing the corresponding (absolute) decision boundary covariance over the training set under constraints on the classifier loss function, *i.e.*:

$$\begin{aligned} &\text{minimize} && |\tfrac{1}{N} \textstyle\sum_{i=1}^{N} (z_i - \bar{z})\, d_{\boldsymbol{\theta}}(\boldsymbol{x}_i)| \\ &\text{subject to} && L(\boldsymbol{\theta}) \leq (1+\gamma) L(\boldsymbol{\theta}^*), \end{aligned} \tag{4.10}$$

where $L(\boldsymbol{\theta}^*)$ denotes the optimal loss over the training set provided by the unconstrained classifier and $\gamma \geq 0$ specifies the maximum additional loss with respect to the loss provided by the unconstrained classifier. Here, we can ensure maximum fairness with no loss in accuracy by setting $\gamma = 0$.

Remarkably, in many classifiers, including logistic regression and SVMs, the loss function (or the dual of the loss function) is additive over the points in the training set, *i.e.*, $L(\boldsymbol{\theta}) = \sum_{i=1}^{N} L_i(\boldsymbol{\theta})$, where $L_i(\boldsymbol{\theta})$ is the individual loss associated with the i-th point in the training set. Moreover, the individual loss $L_i(\boldsymbol{\theta})$ typically tells us how *close* the predicted label $f(\boldsymbol{x}_i)$ is to the true label y_i, by means of the signed distance to the decision boundary. Therefore, one may think of incorporating loss constraints for a certain set of users, and consequently, prevent individual users originally classified as positive (by the unconstrained

classifier) from being classified as negative by the constrained classifier. To do so, we find the decision boundary parameters $\boldsymbol{\theta}$ as:

$$\begin{array}{ll} \text{minimize} & |\frac{1}{N}\sum_{i=1}^{N}(z_i - \bar{z})\, d_{\boldsymbol{\theta}}(\boldsymbol{x}_i)| \\ \text{subject to} & L_i(\boldsymbol{\theta}) \leq (1+\gamma_i) L_i(\boldsymbol{\theta}^*) \quad \forall i \in \{1,\ldots,N\}, \end{array} \tag{4.11}$$

where $L_i(\boldsymbol{\theta}^*)$ is the individual loss associated to the i-th user in the training set provided by the unconstrained classifier and $\gamma_i \geq 0$ is her allowed additional loss.

The problem formulation in Eq. (4.11) could be used to ensure that certain set of users who are correctly classified by the unconstrained classifier are not misclassified by the fairness-constrained classifier (*e.g.*, to make sure that applying fairness constraints does not lead to egregious misclassification of certain users). However, note that, in comparison with Eq. (4.10), Eq. (4.11) involves a tighter set of constraints—it aims to bound *individual* users' losses as opposed to the *aggregate* loss over all the users. As a result, Eq. (4.11) could lead to larger drops in accuracy for the same level of fairness (refer to Figure 10 for an example).

One could also think of extending the formulation of disparate mistreatment-free classification given by Eq. (4.8) to include a similar business necessity clause. However, such a formulation would result in an optimization problem—with a convex-concave objective and convex constraints—that is currently not supported by the standard convex-concave solvers (Shen et al., 2016). Extending these solvers to cater to such problems, or reformulating the optimization problem and solving it with alternative optimizers (*e.g.*, using evolutionary multi-objective optimization as in Quadrianto and Sharmanska (2017)) would be an interesting direction for future work.

4.4. Examples

In this section, we illustrate how to design fair logistic regression classifiers and linear and nonlinear SVMs using our covariance measures.

Logistic regression free of disparate impact. In logistic regression classifiers, one maps the feature vectors $\boldsymbol{x}_i$ to the class labels y_i by means of a probability distribution:

$$p(y=1|\boldsymbol{x},\boldsymbol{\theta}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}, \tag{4.12}$$

where $\boldsymbol{\theta}$ is obtained by solving a maximum likelihood problem over the training set, *i.e.*, $\boldsymbol{\theta}^* = \text{argmin}_{\boldsymbol{\theta}} -\sum_{(\boldsymbol{x},y)\in\mathcal{D}} \log p(y|\boldsymbol{x},\boldsymbol{\theta})$. Thus, the corresponding loss function is given by $-\sum_{(\boldsymbol{x},y)\in\mathcal{D}} \log p(y|\boldsymbol{x},\boldsymbol{\theta})$, and the problem defined by Eq. (4.3) adopts the following form:

$$\left.\begin{array}{ll} \text{minimize} & -\sum_{(\boldsymbol{x},y)\in\mathcal{D}} \log p(y|\boldsymbol{x},\boldsymbol{\theta}) \end{array}\right\} \text{Logistic regression formulation}$$
$$\left.\begin{array}{ll} \text{subject to} & \frac{1}{N}\sum_{(\boldsymbol{x},z)\in\mathcal{D}}(z-\bar{z})\,\boldsymbol{\theta}^T\boldsymbol{x} \leq c, \\ & \frac{1}{N}\sum_{(\boldsymbol{x},z)\in\mathcal{D}}(z-\bar{z})\,\boldsymbol{\theta}^T\boldsymbol{x} \geq -c. \end{array}\right\} \text{Disparate impact constraints} \tag{4.13}$$

Linear SVM free of disparate mistreatment. A linear SVM distinguishes among classes using a linear hyperplane $\boldsymbol{\theta}^T\boldsymbol{x} = 0$. In this case, the parameter vector $\boldsymbol{\theta}$ of the *fair* linear SVM can be found by solving the problem defined by Eq. (4.8), which becomes the following quadratic program with convex-concave constraints:

$$\left.\begin{array}{ll} \text{minimize} & \|\boldsymbol{\theta}\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{subject to} & y_i\boldsymbol{\theta}^T\boldsymbol{x}_i \geq 1-\xi_i, \forall i \in \{1,\ldots,N\} \\ & \xi_i \geq 0, \forall i \in \{1,\ldots,N\}, \end{array}\right\} \text{SVM formulation}$$
$$\left.\begin{array}{l} \frac{-N_1}{N}\sum_{(\boldsymbol{x},y)\in\mathcal{D}_0}\min(0, y\boldsymbol{\theta}^T\boldsymbol{x}) + \frac{N_0}{N}\sum_{(\boldsymbol{x},y)\in\mathcal{D}_1}\min(0, y\boldsymbol{\theta}^T\boldsymbol{x}) \leq c, \\ \frac{-N_1}{N}\sum_{(\boldsymbol{x},y)\in\mathcal{D}_0}\min(0, y\boldsymbol{\theta}^T\boldsymbol{x}) + \frac{N_0}{N}\sum_{(\boldsymbol{x},y)\in\mathcal{D}_1}\min(0, y\boldsymbol{\theta}^T\boldsymbol{x}) \geq -c, \end{array}\right\} \begin{array}{c}\text{Disparate} \\ \text{mistreatment} \\ \text{constraints}\end{array} \tag{4.14}$$

where $\boldsymbol{\theta}$ and ξ are the variables, $\|\boldsymbol{\theta}\|^2$ corresponds to the boundary between the *support vectors* assigned to different classes, and $C\sum_{i=1}^{n}\xi_i$ penalizes the number of data points falling inside the boundary.

Nonlinear SVM free of disparate impact. In a nonlinear SVM, the decision boundary takes the form $\boldsymbol{\theta}^T\Phi(\boldsymbol{x}) = 0$, where $\Phi(\cdot)$ is a nonlinear transformation that maps every feature vector $\boldsymbol{x}$ into a higher dimensional transformed feature space. Similarly as in the case of a linear SVM, one may think of finding the parameter vector $\boldsymbol{\theta}$ by solving a constrained quadratic program, similar to the one defined by Eq. (4.14). However, the dimensionality of the transformed feature space can be large, or even infinite, making the corresponding optimization problem difficult to solve. Fortunately, we can leverage the *kernel trick* (Schölkopf and Smola, 2002) both in the original optimization problem and the fairness inequalities, and resort instead to the dual form of the problem, which can be solved efficiently. In particular, the dual form is given by (for conciseness, we use the dual form notation of Gentle et al. (2012)):

$$\left.\begin{array}{ll} \text{minimize} & \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{G}\boldsymbol{\alpha} - \mathbf{1}^T\boldsymbol{\alpha} \\ \text{subject to} & 0 \leq \boldsymbol{\alpha} \leq C, \\ & \boldsymbol{y}^T\boldsymbol{\alpha} = 0, \end{array}\right\} \text{SVM formulation}$$
$$\left.\begin{array}{l} \frac{1}{N}\sum_{(\boldsymbol{x},z)\in\mathcal{D}}(z-\bar{z})\, d_{\boldsymbol{\alpha}}(\boldsymbol{x}) \leq c, \\ \frac{1}{N}\sum_{(\boldsymbol{x},z)\in\mathcal{D}}(z-\bar{z})\, d_{\boldsymbol{\alpha}}(\boldsymbol{x}) \geq -c, \end{array}\right\} \begin{array}{c}\text{Disparate impact} \\ \text{constraints}\end{array} \tag{4.15}$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_N]^T$ are the dual variables, $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]^T$ are the class labels, $\boldsymbol{G}$ is the $N \times N$ Gram matrix with $\boldsymbol{G}_{i,j} = y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, and the kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)\rangle$ denotes the inner product between a pair of transformed feature vectors. Here, $d_{\boldsymbol{\alpha}}(\boldsymbol{x}_i) = \sum_{j=1}^{N}\alpha_j y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ can still be interpreted as a signed distance to the decision boundary in the transformed feature space.

5. Evaluation

In this section, we first experiment with synthetic datasets in which we simulate various kind of unfairness and analyze both quantitatively and qualitatively the effectiveness of our framework at designing fair classifiers. We then evaluate the performance of our framework on several real-world datasets in comparison with multiple baselines.

Across this section, we quantify disparate impact (Eq. 3.2) as the absolute difference between the positive class probability for the sensitive feature groups with $z = 0$ and $z = 1$, as various prior studies (Corbett-Davies et al., 2017; Calders and Verwer, 2010; Menon and Williamson, 2017), *i.e.*,

$$DI = \Big|P(\hat{y} = 1|z = 0) - P(\hat{y} = 1|z = 1)\Big|, \tag{5.1}$$

where a value of DI closer to zero denotes a smaller degree of disparate impact. Similarly, we quantify disparate mistreatment with respect to false positive rates and false negative rates (Eqs. (3.4-3.5)) as the difference between false positive (negative) rate probabilities, *i.e.*,

$$DM_{FPR} = P(\hat{y} \neq y|z = 0, y = -1) - P(\hat{y} \neq y|z = 1, y = -1), \tag{5.2}$$
$$DM_{FNR} = P(\hat{y} \neq y|z = 0, y = 1) - P(\hat{y} \neq y|z = 1, y = 1), \tag{5.3}$$

where the closer the values of DM_{FPR} and DM_{FNR} to 0, the lower the degree of disparate mistreatment. Note that unlike in the case of disparate impact in Eq. (5.1), we do not use the absolute difference while quantifying disparate mistreatment. As we later show in Section 5.1.2, the (in)equality in the signs of DM_{FPR} and DM_{FNR} carries significant consequences when considering disparate mistreatment w.r.t. false positive rate and false negative rate simultaneously. In such cases, the sign of the differences should also be taken into account.

5.1. Experiments on synthetic data

In this section, we first generate synthetic data where a classifier optimizing for accuracy would lead to disparate impact and then generate data where the accuracy-optimizing classifier would lead to disparate mistreatment. In both the cases, we *simultaneously* control for disparate treatment as well, that is, the classifiers do not leverage sensitive feature during decision time in either of the cases.

5.1.1. Mitigating disparate impact

To simulate different degrees of disparate impact in classification outcomes, we generate two synthetic datasets with different levels of correlation between a single, binary sensitive attribute and class labels. Specifically, we generate 4,000 binary class labels uniformly at random and assign a 2-dimensional user feature vector per label by drawing samples from two different Gaussian distributions:

$$p(\boldsymbol{x}|y = 1) = \mathcal{N}([2; 2], [5,\ 1; 1,\ 5])$$
$$p(\boldsymbol{x}|y = -1) = \mathcal{N}([-2; -2], [10,\ 1; 1,\ 3]).$$

Then, we draw each user's sensitive attribute z from a Bernoulli distribution: $p(z = 1) = p(\boldsymbol{x}'|y = 1)/(p(\boldsymbol{x}'|y = 1) + p(\boldsymbol{x}'|y = -1))$, where $\boldsymbol{x}' = [\cos(\phi),\ -\sin(\phi); \sin(\phi),\ \cos(\phi)]\boldsymbol{x}$ is simply a rotated version of the feature vector, $\boldsymbol{x}$. We generate two datasets with different values for the parameter ϕ ($\pi/4$ and $\pi/8$), which controls the correlation between the sensitive attribute, z, and the class labels, y (and hence, the resulting degree of disparate impact). Here, the closer ϕ is to zero, the higher the correlation between z and y.

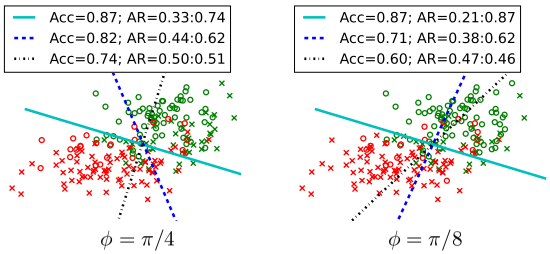


Figure 2: [Disparate impact constraints] Performance of different (unconstrained and constrained) classifiers along with their accuracy (Acc) and positive class acceptance rates (AR) for groups $z = 0$ (crosses) and $z = 1$ (circles). Green points represent examples with $y = 1$ and red points represent example with $y = -1$. The solid lines show the decision boundaries for logistic regression classifiers without fairness constraints. The dashed lines show the decision boundaries for logistic regression classifiers trained to maximize accuracy under fairness constraints (Eq. (4.13)). Each column corresponds to a dataset with different correlation value between sensitive attribute values and class labels.

Next, we train logistic regression classifiers optimizing for accuracy on both the datasets. The accuracy of the classifiers in both cases is 0.87 (note that the datasets only differ in terms of the correlation between z and y). However, the classifiers lead to $DI = |0.33 - 0.74| = 0.41$ and $DI = |0.21 - 0.87| = 0.66$ on datasets with $\phi = \pi/4$ and $\phi = \pi/8$, respectively. To overcome the unfairness, we train logistic regression classifiers with disparate impact constraints (Eq. 4.13) on both datasets.

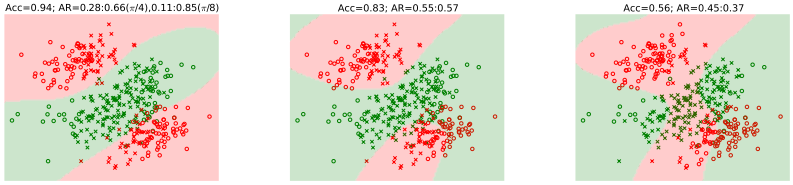
Figure 2 shows the decision boundaries provided by the classifiers for two (successively decreasing) covariance thresholds, c. We compare these boundaries against the unconstrained decision boundary (solid line). As expected, given the data generation process, fairness constraints map into a rotation of the decision boundary (dashed lines), which is greater as we decrease threshold value c or increase the correlation in the original data (from $\phi = \pi/4$ to $\phi = \pi/8$). This movement of the decision boundaries shows that our fairness constraints are successfully undoing (albeit in a highly controlled setting) the rotations we used to induce disparate impact in the dataset. Moreover, a smaller covariance threshold (a larger rotation) leads to a more fair solution, although, it comes at a larger cost in accuracy.

Next, we illustrate how the decision boundary of an non-linear classifier, a SVM with radial basis function (RBF) kernel, changes under disparate impact constraints (Eq. (4.15)). To this end, we generate 4,000 user binary class labels uniformly at random and assign a 2-dimensional user feature vector per label by drawing samples from

$$p(\boldsymbol{x}|y = 1, \beta) = \beta\mathcal{N}([2; 2], [5\ 1; 1\ 5]) + (1 - \beta)\mathcal{N}([-2; -2], [10\ 1; 1\ 3])$$
$$p(\boldsymbol{x}|y = -1, \beta) = \beta\mathcal{N}([4; -4], [4\ 4; 2\ 5]) + (1 - \beta)\mathcal{N}([-4; 6], [6\ 2; 2\ 3])$$

where $\beta \in \{0, 1\}$ is sampled from Bernoulli(0.5). Then, we generate each user's sensitive attribute z by applying the same rotation as described earlier.

Figure 3 shows the decision boundaries provided by the SVM that maximizes accuracy under fairness constraints with $c = 0$ for two different correlation values: $\phi = \pi/4$ and



(a) Unconstrained (b) $\phi = \pi/4$ (c) $\phi = \pi/8$

Figure 3: [Disparate impact constraints] Decision boundaries for SVM classifier with RBF Kernel trained without fairness constraints (left) and with fairness constraints (middle and right) on two synthetic datasets. Also shown are the classification accuracy (Acc) and acceptance rate (AR) for each group.

$\phi = \pi/8$, in comparison with the unconstrained SVM. We observe that, in this case, the decision boundaries provided by the constrained SVMs are very different to the decision boundary provided by the unconstrained SVM, and are not just simple shifts or rotations of the latter.

5.1.2. Mitigating disparate mistreatment

In a manner similar to the previous section, we now experiment with synthetic datasets where training classifiers optimizing for accuracy would lead to disparate mistreatment. However, disparate mistreatment can arise in multiple different ways, as detailed below. To study these different situations, we first start with a simple scenario in which the classifier is unfair in terms of *only* false positive rate *or* false negative rate. Then, we focus on a more complex scenario in which the classifier is unfair in terms of *both*.

Disparate mistreatment on *only* false positive rate *or* false negative rate. The first scenario considers a case where a classifier maximizing accuracy leads to disparate mistreatment in terms of only the false positive rate (false negative rate), while being fair with respect to false negative rate (false positive rate), *i.e.*, $DM_{FPR} \neq 0$ and $DM_{FNR} = 0$ (or, alternatively, $DM_{FPR} = 0$ and $DM_{FNR} \neq 0$).

To simulate this scenario, we generate 10,000 binary class labels ($y \in \{-1, 1\}$) and corresponding sensitive attribute values ($z \in \{0, 1\}$), both uniformly at random, and assign a two-dimensional user feature vector ($\boldsymbol{x}$) to each of the points. To ensure different distributions for negative classes of the two sensitive attribute value groups (so that the two groups have different false positive rates), the user feature vectors are sampled from the following distributions (we sample 2500 points from each distribution):

$$
\begin{aligned}
p(\boldsymbol{x}|z=0, y=1) &= \mathcal{N}([2,2],[3,1;1,3]) \\
p(\boldsymbol{x}|z=1, y=1) &= \mathcal{N}([2,2],[3,1;1,3]) \\
p(\boldsymbol{x}|z=0, y=-1) &= \mathcal{N}([1,1],[3,3;1,3]) \\
p(\boldsymbol{x}|z=1, y=-1) &= \mathcal{N}([-2,-2],[3,1;1,3]).
\end{aligned}
$$

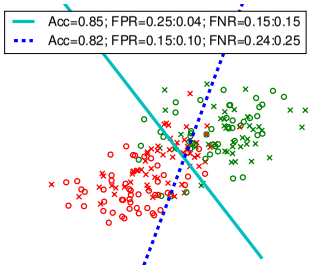


Figure 4: [Disparate mistreatment constraints] The figure shows the original decision boundary (solid line) and fair decision boundary (dashed line), along with corresponding accuracy and false positive rates for groups $z = 0$ (crosses) and $z = 1$ (circles). Fairness constraints cause the original decision boundary to rotate such that previously misclassified subjects with $z = 0$ are moved into the negative class (decreasing false positives), while well-classified subjects with $z = 1$ are moved into the positive class (increasing false positives), leading to equal false positive rates for both groups.

Next, we train a logistic regression classifier optimizing for accuracy on this data. The classifier is able to achieve an accuracy of 0.85. However, due to difference in feature distributions for the two sensitive attribute groups, it achieves $DM_{FNR} = 0.15 - 0.15 = 0$ and $DM_{FPR} = 0.25 - 0.04 = 0.21$, which constitutes a clear case of disparate mistreatment in terms of false positive rate. We then train a logistic regression classifier subject to fairness constraints on false positive rate, with a covariance threshold $c = 0$.

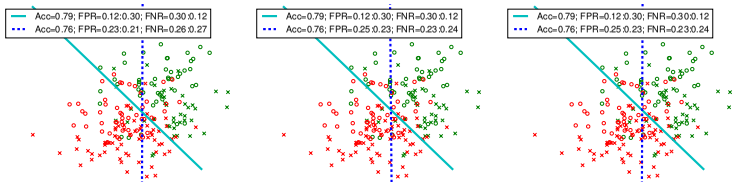
Figure 4 shows the decision boundaries for both the unconstrained classifier (solid) and the fairness-constrained classifier (dashed). We observe that applying the fairness constraint successfully causes the false positive rates for both groups ($z = 0$ and $z = 1$) to converge, and hence, the outcomes of the classifier become more fair, *i.e.*, $DM_{FPR} \to 0$, while DM_{FNR} remains close to zero. We note that the invariance of DM_{FNR} may however change depending on the underlying distribution of the data.

Disparate mistreatment on *both* false positive rate and false negative rate. In this part, we consider a more complex scenario, where the outcomes of the classifier suffer from disparate mistreatment with respect to *both* false positive rate and false negative rate, *i.e.*, both DM_{FPR} and DM_{FNR} are non-zero. This scenario can in turn be split into two cases:

I. DM_{FPR} and DM_{FNR} have *opposite signs*, *i.e.*, the decision boundary disproportionately *favors* subjects from a certain sensitive attribute value group to be in the positive class (even when such assignments are misclassifications) while disproportionately assigning the subjects from the other group to the negative class. As a result, false positive rate for one group is higher than the other, while the false negative rate for the same group is lower.

II. DM_{FPR} and DM_{FNR} have the *same sign*, *i.e.*, both false positive as well as false negative rate are higher for a certain sensitive attribute value group. These cases might arise in scenarios when a certain group is harder to classify than the other.

Next, we experiment with each of the above cases separately.



(a) FPR constraints (b) FNR constraints (c) Both constraints

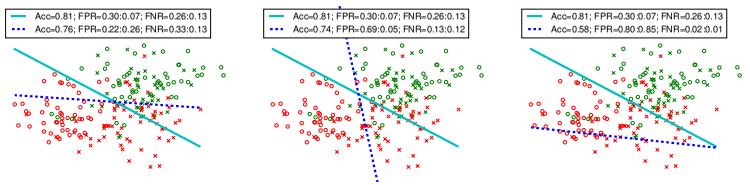
Figure 5: [Disparate mistreatment constraints] DM_{FPR} and DM_{FNR} have opposite signs. Removing disparate mistreatment on FPR can potentially help remove disparate mistreatment on FNR. Removing disparate mistreatment on both at the same time leads to very similar results.

— **Case I:** To simulate this scenario, we first generate 2,500 samples from each of the following distributions:

$$\begin{aligned} p(\boldsymbol{x}|z=0,y=1) &= \mathcal{N}([2,0],[5,1;1,5]) \\ p(\boldsymbol{x}|z=1,y=1) &= \mathcal{N}([2,3],[5,1;1,5]) \\ p(\boldsymbol{x}|z=0,y=-1) &= \mathcal{N}([-1,-3],[5,1;1,5]) \\ p(\boldsymbol{x}|z=1,y=-1) &= \mathcal{N}([-1,0],[5,1;1,5]) \end{aligned}$$

An accuracy-maximizing logistic regression classifier on this dataset attains an overall accuracy of 0.79 but leads to a false positive rate of 0.12 and 0.30 (*i.e.*, $DM_{FPR} = 0.12 - 0.30 = -0.18$) for the sensitive attribute groups $z = 0$ and $z = 1$, respectively; and false negative rates of 0.30 and 0.12 (*i.e.*, $DM_{FNR} = 0.30 - 0.12 = 0.18$). To remove this disparate mistreatment, we train three different classifiers, with fairness constraints on (i) false positive rates (ii) false negative rates and (iii) on both false positive and false negative rates.

Figure 5 summarizes the results for this scenario by showing the decision boundaries for the unconstrained classifier (solid) and the constrained fair classifiers. Here, we can observe several interesting patterns. First, removing disparate mistreatment on only false positive rate causes a rotation in the decision boundary to move previously *misclassified* subjects with $z = 1$ into the negative class, *decreasing* their false positive rate. However, in the process, it also moves previously *well-classified* subjects with $z = 1$ into the negative class, *increasing* their false negative rate. As a consequence, controlling disparate mistreatment on false positive rate (Figure 5(a)), also removes disparate mistreatment on false negative rate. A similar effect occurs when we control disparate mistreatment only with respect to the false negative rate (Figure 5(b)), and therefore, provides similar results as the constrained classifier for both false positive and false negative rates (Figure 5(c)). This effect is explained by the distribution of the data, where the centroids of the clusters for the group with $z = 0$ are shifted with respect to the ones for the group $z = 1$.



(a) FPR constraints (b) FNR constraints (c) Both constraints

Figure 6: [Disparate mistreatment constraints] DM_{FPR} and DM_{FNR} have the same sign. Removing disparate mistreatment on FPR can potentially increase disparate mistreatment on FNR. Removing disparate mistreatment on both at the same time causes a larger drop in accuracy.

— **Case II:** To simulate the scenario where both DM_{FPR} and DM_{FNR} have the same sign, we generate 2,500 samples from each of the following distributions:

$$\begin{aligned} p(\boldsymbol{x}|z=0,y=1) &= \mathcal{N}([1,2],[5,2;2,5]) \\ p(\boldsymbol{x}|z=1,y=1) &= \mathcal{N}([2,3],[10,1;1,4]) \\ p(\boldsymbol{x}|z=0,y=-1) &= \mathcal{N}([0,-1],[7,1;1,7]) \\ p(\boldsymbol{x}|z=1,y=-1) &= \mathcal{N}([-5,0],[5,1;1,5]) \end{aligned}$$

We then train a accuracy-optimizing logistic regression classifier on this dataset. It attains an accuracy of 0.81 but leads to $DM_{FPR} = 0.30 - 0.07 = 0.23$ and $DM_{FNR} = 0.26 - 0.13 = 0.13$, resulting in disparate mistreatment in terms of both false positive and negative rates. Then, similarly to the previous scenario, we train three different kind of constrained classifiers to remove disparate mistreatment on (i) false positive rate, (ii) false negatives rate, and (iii) both.

Figure 6 summarizes the results by showing the decision boundaries for both the unconstrained classifiers (solid) and the fair constrained classifier (dashed) when controlling for disparate mistreatment with respect to false positive rate, false negative rate and both, respectively. We observe following noticeable patterns. First, controlling disparate mistreatment for only false positive rate (false negative rate), leads to a minor drop in accuracy, but in contrast to Case I, can exacerbate the disparate mistreatment on false negative rate (false positive rate). For example, while the decision boundary is moved to control for disparate mistreatment on false negative rate, that is, to ensure that more subjects with $z = 0$ are well-classified in the positive class (reducing false negative rate), it also moves previously well-classified negative subjects into the positive class, hence increasing the false positive rate. A similar phenomenon occur when controlling disparate mistreatment with respect to only false positive rate. As a consequence, controlling for both types of disparate mistreatment simultaneously brings DM_{FPR} and DM_{FNR} close to zero, but causes a large drop in accuracy.

5.2. Real-world datasets

We now evaluate the effectiveness of our covariance framework in removing disparate impact and disparate mistreatment on several real-world datasets. In doing so, we also compare the performance of our framework to several methods from the fair machine learning literature.

In all the experiments, to obtain more reliable estimates of accuracy and fairness, we repeatedly split each dataset into a train (70%) and test (30%) set 5 times and report the average statistics for accuracy and fairness.

5.2.1. Mitigating disparate impact

Datasets and experimental setup. Here, we experiment with two real-world datasets: The Adult income dataset (Adult, 1996) and the Bank marketing dataset (Bank, 2014).

The Adult dataset contains a total of 45,222 subjects, each with 14 features (*e.g.*, age, educational level) and a binary label, which indicates whether a subject's incomes is above (positive class) or below (negative class) 50K USD. With the aim of experimenting with binary as well as non-binary (polyvalent) sensitive attributes, we consider the features gender and race to be sensitive. Here, gender (with feature values: men and women) serves as an example of binary sensitive attribute and race (with feature values: American-Indian, Asian, Black, White and Other) serves as an example of a non-binary sensitive attribute.

The Bank dataset contains a total of 41,188 subjects, each with 20 attributes (*e.g.*, marital status) and a binary label, which indicates whether the client has subscribed (positive class) or not (negative class) to a term deposit. In this case, we consider age as (binary) sensitive attribute, which is discretized to indicate whether the client's age is between 25 and 60 years. For detailed statistics about the distribution of different sensitive attributes in positive class in these datasets, we refer the reader to Appendix A.

For the sake of conciseness, while presenting the results for binary sensitive attributes, we refer to women and men, respectively, as protected and non-protected groups in Adult data. Similarly, in Bank data, we refer to users between age 25 and 60 as protected and rest of the users as non-protected group.

Methods. In our experiments, we also compare our approach to well-known competing method from fairness-aware machine learning literature (detailed in Section 6). More specifically, we consider the following methods:

- ***Our method (C-LR and C-SVM)*:** Implements our covariance constraints-based methods for controlling disparate impact with a logistic regression classifier (Eq. (4.13)) and a dual-form SVM classifier with a linear kernel (Eq. (4.15)). [8] On the datasets considered here, different choices of kernel (linear vs. RBF) lead to a very similar performance in terms of accuracy and disparate impact. This method does not use the sensitive feature information at decision time.
- ***Preferential sampling (PS-LR and PS-SVM)*:** Implements the data pre-processing technique of Kamiran and Calders (2010) on a logistic regression and a SVM classifier. Specifically, this method operates as follows: (i) We first train a standard (potentially unfair) classifier on the given dataset. (ii) Next, we move / replicate the protected group data points to / on the positive side of the decision boundary (and vice versa for the non-protected group) until the decision boundary leads to zero disparate impact, *i.e.*, until it satisfies Eq. (3.2). (iii) We then train the final (fair) classifier on the perturbed dataset. This method does not use the sensitive feature information at decision time.
- ***Fairness-regularized logistic regression (FR-LR)*:** The in-processing technique of Kamishima et al. (2011). This technique is only limited to the logistic regression classification model. This technique works by adding a fairness regularization term in the objective function that penalizes the mutual information between the sensitive feature and the classifier decisions. In this way, this method treats the mutual information as the unfairness proxy, as opposed to covariance in our case. This technique needs the sensitive feature information at decision time, hence cannot remove disparate treatment.
- ***Post-Processing (PP-LR and PP-SVM)*:** The post-processing technique discussed in Corbett-Davies et al. (2017). This method works by first training a standard logistic regression or SVM classifier on the given dataset and then finding a pair of positive class acceptance thresholds [9] such that the decisions based on those thresholds lead to maximum accuracy while having no disparate impact. This technique also requires the sensitive feature information at decision time so it cannot avoid disparate treatment.

Results. First, we experiment with two standard (unconstrained) logistic regression and SVM classifiers. In the Adult dataset, the logistic regression classifier leads to an accuracy of 0.846. However, the classifier results in highly disparate positive class acceptance rates for protected and non-protected groups: 0.08 and 0.26. The SVM classifier leads to a similar accuracy (0.847) and disparity in positive class acceptance rates (0.08 vs 0.25). In the Bank dataset, the two classifiers lead to accuracies of 0.911 and 0.910, respectively, and acceptance rates of 0.06 vs. 0.25, and 0.05 vs. 0.23 respectively. The high disparity in acceptance rates over the two datasets clearly constitutes a case of disparate impact.

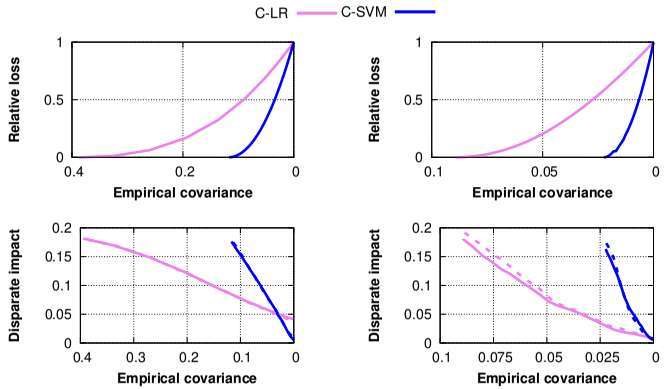
We then apply our framework to limit disparate impact with respect to a single binary sensitive attribute, gender and age, for respectively, the Adult and Bank datasets. For each dataset, we train several logistic regression and SVM classifiers (denoted by 'C-LR' and 'C-SVM', respectively), each subject to fairness constraints with different values of covariance threshold, c (Eqs.(4.13, 4.15)). Next, we study the effect of covariance constraints on the loss function value, level of disparate impact and accuracy of the classifier.

Figure 7 (top row) shows the empirical decision boundary covariance against the relative loss incurred by the classifier. The 'relative loss' is normalized between the loss incurred by an unconstrained classifier and by the classifier with a covariance threshold of 0. We notice that as expected, a decreasing value of empirical covariance results in an increasing loss. However, each pair of (covariance, loss) values is guaranteed to be Pareto optimal, since our problem formulation is convex.

The bottom row in Figure 7 investigates the correspondence between decision boundary covariance and disparate impact, as defined in Eq. (5.1), computed on the training set (solid

8. For the SVM classifier, the hyperparameter C (in Eq. (4.15)) is only cross-validated for the unconstrained classifier, and the same hyperparameter is used for the fairness-constrained classifiers. One could further optimize the classifier performance by cross-validating the value of C for each value of the covariance threshold separately.

9. The acceptance threshold is zero for a standard logistic regression or SVM classifier.

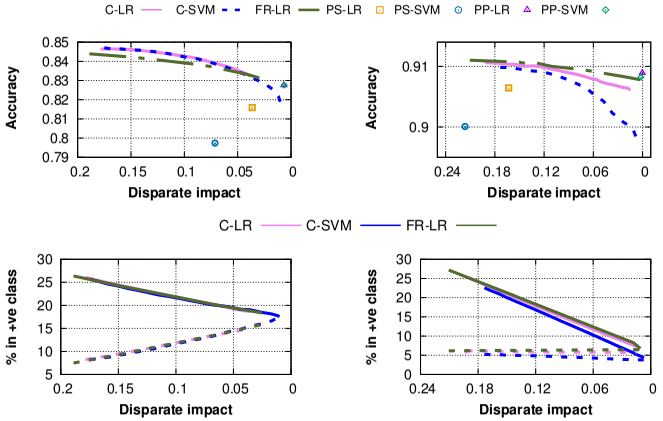


Adult data Bank data

Figure 7: [Disparate impact constraints: Single, binary sensitive attribute] Panels in the top row show the trade-off between the empirical covariance in Eq. (4.2) and the relative loss with respect to the unconstrained classifier, for the Adult and Bank (training) datasets. Here each pair of (covariance, loss) values is guaranteed to be Pareto optimal by construction. Panels in the bottom row show the correspondence between the empirical covariance and disparate impact in Eq. (5.1) for classifiers trained under fairness constraints. Solid lines correspond to the training data whereas dashed lines correspond to the test data. The figure shows that a decreasing empirical covariance leads to higher loss but lower disparate impact. Moreover, the classifiers do not lead to "fairness overfitting"—in fact, the dashed lines for the Adult data are barely visible due to strong correspondance between the performance on the training and test sets.

lines) as well as the test set (dashed lines). The figure shows that, as desired: (i) the lower the covariance, the lower the disparate impact of the classifier; (ii) zero disparate impact maps to roughly zero covariance; and, (iii) there is a strong correspondence between the covariance-fairness relationship across the training and the test set, that is, a fair classifier on the training data also leads to fair outcomes on the test data.

We next compare the performance of our constrained classifiers in terms of disparate impact–accuracy tradeoffs with the baselines methods mentioned above. The results presented in Figure 8, top row, show that: (i) the performance of our classifiers (C-LR, C-SVM) and fairness-regularized logistic regression (FR-LR) is comparable, ours are slightly better for Adult data (left column) while slightly worse for Bank data (right column); (ii) the preferential sampling presents the worst performance and results in high disparate impact; and, (iii) the post-processing technique leads to the best performance among all methods. However, we note that both FR-LR and PP-LR / PP-SVM use the sensitive feature information at decision time while the other two techniques do not use it.



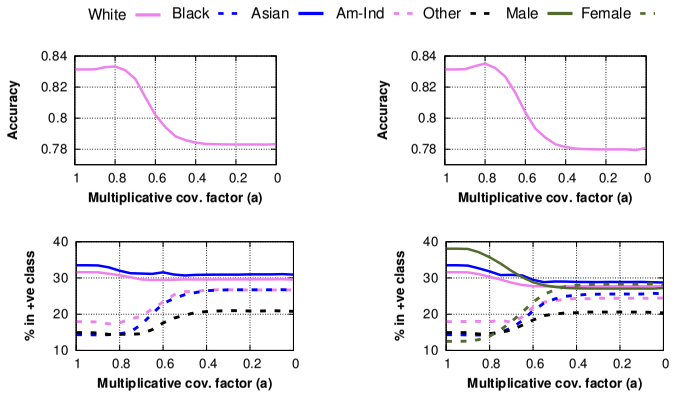
Adult data Bank data

Figure 8: [Disparate impact constraints: Single, binary sensitive attribute] The figure shows the accuracy against disparate impact in Eq. 5.1 (top) and the percentage of protected (dashed) and non-protected (solid) users in the positive class against the disparate impact value (bottom). For all methods, a decreasing degree of disparate impact also leads to a decreasing accuracy. The post-processing technique (PP-LR and PP-SVM) achieves the best disparate impact-accuracy tradeoff. However, this technique as well as FR-LR use the sensitive feature information at decision time (as opposed to C-LR, C-SVM, PS-LR and PS-SVM), and would hence result in disparate treatment.

For a more fair comparison, we also train our method with access to sensitive features at decision time. Specifically, we train constrained logistic regression classifiers (C-LR) under the same setup as above, with the exception that the non-sensitive ($\boldsymbol{x}$) and sensitive features (z) are not disjoint feature sets—that is, the classifier learns a non-zero weight for the sensitive feature z.

Under this setup, on the Adult dataset, our constrained logistic regression classifier (C-LR) achieves an accuracy of 0.839 and DI of 0.09, as compared to 0.828 accuracy and 0.01 DI achieved by the PP-LR classifier. In this case C-LR achieves a better accuracy than PP-LR, but does not remove DI as well as PP-LR. Next, we adjust the thresholds of PP-LR in a way that the resulting classifier has DI ≤ 0.9 (*i.e.*, it tries to match the DI of C-LR) while maximizing accuracy. Under these thresholds, PP-LR achieves an accuracy of 0.840 and DI of 0.07. On the Bank dataset, C-LR achieves an accuracy of 0.908 (0.909 for PP-LR) and DI of 0.01 (0.0 for PP-LR). On both Bank and Adult datasets, both methods achieve similar accuracy for a similar level of DI (with PP-LR performing marginally better).

The bottom row of Figure 8 shows the percentage of users from protected and non-protected groups in the positive class along with the degree of disparate impact. We note



(a) Non-binary (polyvalent) sensitive attribute (b) Multiple sensitive attributes

Figure 9: [Disparate impact constraints: Non-binary and several sensitive attributes] The figure shows accuracy (top) and percentage of users in positive class (bottom) against a multiplicative factor $a \in [0, 1]$ such that $c = ac^*$, where c^* denotes the unconstrained classifier covariance.

that in the Adult data, all classifiers move non-protected users (men) to the negative class and protected users (women) to the positive class to remove disparate impact. In contrast, in the Bank data, they only move non-protected (young and old) users originally labeled as positive to the negative class since it provides a smaller accuracy loss. However, the latter can be problematic: from a business perspective, a bank may be interested in finding potential subscribers rather than losing existing customers. This observation could motivate the business necessity clause of the disparate impact doctrine. To counter such situations, one can use our alternative formulation in Section 4.3. We experiment with this formulation later in this section.

Finally, we apply our framework to control disparate impact with respect to non-binary (race) and several (gender and race) sensitive attributes in the Adult dataset. We do not compare with competing methods since they cannot handle non-binary or several sensitive attributes. Figure 9 summarizes the results by showing the accuracy and the percentage of subjects sharing each sensitive attribute value classified as positive against a multiplicative covariance factor $a \in [0, 1]$ such that $c = ac^*$, where c^* is the unconstrained classifier covariance[10] (note that disparate impact in Eq. (5.1) is only defined for a binary sensitive feature). As expected, as the value of c decreases, the percentage of subjects in the positive

10. For several sensitive features, we compute the initial covariance c_k^* for each of the sensitive feature k, and then compute the covariance threshold separately for each sensitive feature as ac_k^*.

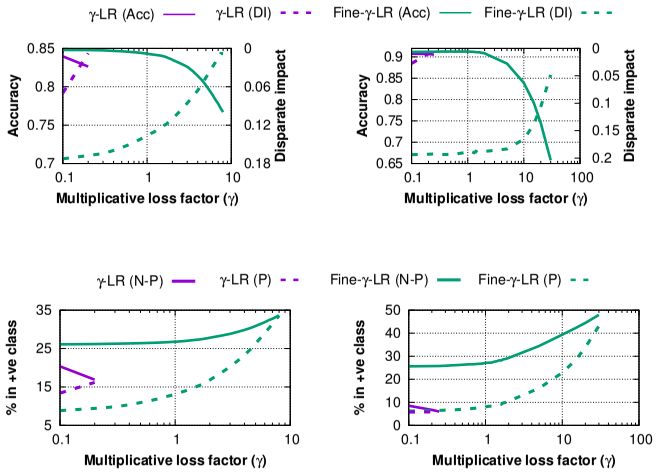


Figure 10: [Business necessity clause] Panels in top row show the accuracy (solid) and disparate impact (dashed) against γ. Panels in the bottom row show the percentage of protected (P, dashed) and non-protected (N-P, solid) users in the positive class against γ.

class from sensitive attribute value groups become nearly equal [11] while the loss in accuracy is modest.

Disparate impact's business necessity clause. We now experiment with our formulation for handling the business necessity clause (Section 4.3) to avoid scenarios where removing disparate impact leads to almost all the users being assigned the negative class label (Figure 8). Specifically, we demonstrate that our formulation in Section 4.3 can minimize disparate impact while precisely controlling loss in accuracy. We also demonstrate that our formulation can additionally provide guarantees for classifying certain users in the positive class while minimizing disparate impact.

To this end, we first train several logistic regression classifiers (denoted by 'γ-LR'), which minimize the decision boundary covariance subject to accuracy constraints over the entire dataset by solving problem (4.10) with increasing values of γ. Then, we train logistic regression classifiers (denoted by 'Fine-γ-LR') that minimize the decision boundary covariance subject to *fine-grained* accuracy constraints by solving problem (4.11). Here, we prevent the non-protected users that were classified as positive by the unconstrained logistic regression

11. The scarce representation of the race value 'Other' (only 0.8% of the data) hinders an accurate estimation of the decision boundary covariance and, as a result, the classifier does not reach perfect fairness with respect to this sensitive attribute value.

classifier from being classified as negative by constraining that their distance from decision boundary stays positive while learning the fair boundary. We then increase $\gamma_i = \gamma$ for the remaining users. In both cases, we increased the value of γ until we reach 0 disparate impact during training. Figure 10 summarizes the results for both datasets, by showing (a) the average accuracy (solid curves) and disparate impact (dashed curves) against γ, and (b) the percentage of non-protected (N-P, solid curves) and protected (P, dashed curves) users in the positive class against γ. We observe that, as we increase γ, the classifiers that constrain the overall training loss (γ-LR) remove non-protected users from the positive class and add protected users to the positive class, in contrast, the classifiers that prevent the non-protected users that were classified as positive in the unconstrained classifier from being classified as negative (Fine-γ-LR) add both protected and non-protected users to the positive class. As a consequence, the latter achieves lower accuracy for the same value of disparate impact.

5.2.2. Mitigating disparate mistreatment

Datasets and experimental setup. In this section, we experiment with two real-world datasets: COMPAS risk assessment dataset (Larson et al., 2016a) and the NYPD stop-question-and-frisk (SQF) dataset (Stop, Question and Frisk Data, 2017).

The ProPublica COMPAS dataset consists of data about $7,215$ pretrial criminal defendants, and contains a number of features such as age of the criminal defendant, number of prior criminal offenses *etc.*, and a class label indicating whether a person would recidivate within two years (positive class) or not (negative class). For more information about the data collection, we point the reader to a detailed description (Larson et al., 2016b) and some of the follow-up discussion on this dataset (Angwin and Larson; Flores et al., 2016). We designate race as the sensitive feature. Following ProPublica's analysis (Larson et al., 2016b), we only consider a subset of offenders whose race (the sensitive feature) is either black or white. Recidivism rates for the two groups are shown in Table 6 in Appendix A. For modeling the classification task, we use the same set of features as used by ProPublica (Larson et al., 2016b). [12] After performing the filtering described above, we obtain $5,287$ subjects and 5 features.

The NYPD SQF dataset consists of $84,868$ pedestrians who were stopped in the year 2012 on the suspicion of having a weapon. The dataset also contains over 100 features (*e.g.*, gender, height, reason for stop) and a binary label which indicates whether (negative class) or not (positive class) a weapon was discovered. For our analysis, we consider the race to be the sensitive feature with values blacks and whites. The classes in this dataset are highly imbalanced (97% of subjects in positive class), and as a result, a logistic regression classifier classifies almost all data points into the positive class. To counter this imbalance, we subsample the dataset to have equal number of subjects from each class. Information about weapon discovery rate for both races in included in Tables 7 and 8 in Appendix A.

12. Notice that goal of this section is not to analyze the best set of features for recidivism prediction, rather, we focus on showing that our method can effectively remove disparate mistreatment in a given dataset. Hence, we chose to use the same set of features as used by ProPublica for their analysis. Moreover, since race is one of the features in the learnable set, we additionally assume that *all* the methods have access to the sensitive attributes while making decisions.

Algorithm 1: Baseline method for removing disparate mistreatment w.r.t. FPR.

Input: Training set $\mathcal{D} = \{(\boldsymbol{x}_i, y_i, z_i)\}_{i=1}^N$, $\Delta > 0$ $\epsilon > 0$
Output: Fair baseline decision boundary $\boldsymbol{\theta}$
Initialize: Penalty $C = 1$
Train (unfair) classifier $\boldsymbol{\theta} = \text{argmin}_{\boldsymbol{\theta}} \sum_{\mathbf{d} \in \mathcal{D}} L(\boldsymbol{\theta}, \mathbf{d})$
Compute $\hat{y}_i = \text{sign}(d_{\boldsymbol{\theta}}(\boldsymbol{x}_i))$ and D_{FP} on $\mathcal{D}$.
if $D_{FP} > 0$ **then** $s = 0$
else $s = 1$
$\mathcal{P} = \{\boldsymbol{x}_i, y_i, z_i | \hat{y} \neq y_i, z_i = s\}$, $\bar{\mathcal{P}} = \mathcal{D} \setminus \mathcal{P}$.
while $D_{FP} > \epsilon$ **do**
  Increase penalty: $C = C + \Delta$.
  $\boldsymbol{\theta} = \text{argmin}_{\boldsymbol{\theta}} \quad C \sum_{\mathbf{d} \in \mathcal{P}} L(\boldsymbol{\theta}, \mathbf{d}) + \sum_{\mathbf{d} \in \bar{\mathcal{P}}} L(\boldsymbol{\theta}, \mathbf{d})$
end

Furthermore, for training the classifiers, we consider a similar set of features as Goel et al. (2015). After performing these two filtering steps, we obtain $5,832$ subjects and 19 features.

Methods. In our experiments, we compare our approach to two baseline methods. More specifically, we consider the following methods:

- ***Our method***: Implements our scheme to avoid disparate treatment and disparate mistreatment *simultaneously*. Disparate mistreatment is avoided by using fairness constraints on false positive and / or false negative rates. Disparate treatment is avoided by ensuring that sensitive attribute information is not used while making decisions, *i.e.*, by keeping user feature vectors ($\boldsymbol{x}$) and the sensitive features (z) disjoint.
- ***Our method$_{sen}$***: Implements our scheme to avoid disparate mistreatment only. The user feature vectors ($\boldsymbol{x}$) and the sensitive features (z) are not disjoint, that is, the classifier learns a non-zero weight for z. Therefore, the sensitive feature information is used for decision making, resulting in disparate treatment.
- ***Hardt et al.*** (Hardt et al., 2016): Operates by post-processing the outcomes of a possibly discriminatory classifier (logistic regression in this case) and using different decision thresholds for different sensitive feature value groups to remove disparate mistreatment. By construction, it needs the sensitive feature information while making decisions, and hence cannot avoid disparate treatment. This method is similar to the post-processing scheme discussed in Corbett-Davies et al. (2017).
- ***Donini et al.*** (Donini et al., 2018): This pre- / in-processing method aims to remove the discrepancy in the false negative rate of the two groups by adding constraints to a SVM classifier. We use the implementation provided by the authors[13] and train the method such that it has access to the sensitive feature information at decision time.
- ***Baseline***: Baseline introduced by us to facilitate a third comparison method. Tries to remove disparate mistreatment by introducing different penalties for misclassified

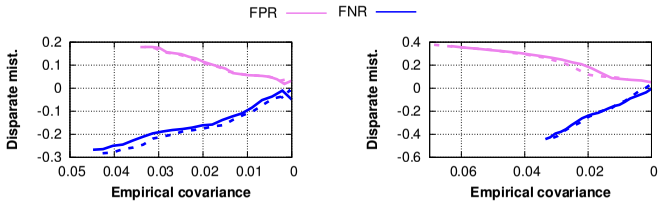
13. https://github.com/jmikko/fair_ERM

data points with different sensitive attribute values during training phase. Specifically, it proceeds in two steps. First, it trains an (unfair) classifier minimizing a loss function (*e.g.*, logistic loss) over the training data. Next, it selects the set of misclassified data points from the sensitive attribute value group that presents the higher error rate. For example, if one wants to remove disparate mistreatment with respect to false positive rate and $DM_{FPR} > 0$ (which means the false positive rate for points with $z = 0$ is higher than that of $z = 1$), it selects the set of misclassified data points in the training set having $z = 0$ and $y = -1$. Next, it iteratively re-trains the classifier with increasingly higher penalties on this set of data points until a certain fairness level is achieved in the training set (until $DM_{FPR} \leq \epsilon$). The algorithm is summarized in Figure 1, particularized to ensure fairness in terms of false positive rate. This process can be intuitively extended to account for fairness in terms of false negative rate or for *both* false positive rate and false negative rate. This method can be trained with or without using sensitive feature information while making decisions. We opt for the latter option.

Results. First, we experiment with a standard logistic regression classifier optimizing for accuracy on both datasets. For the COMPAS dataset, the (unconstrained) logistic regression classifier leads to an accuracy of 0.664. However, the classifier yields false positive rates of 0.35 and 0.17, respectively, for blacks and whites (*i.e.*, $DM_{FPR} = 0.18$), and false negative rates of 0.32 and 0.61 (*i.e.*, $DM_{FNR} = -0.29$). These results constitute a clear case of disparate mistreatment in terms of both false positive rate and false negative rate. The classifier puts one group (blacks) at relative disadvantage by disproportionately misclassifying negative (did not recidivate) subjects from this group into positive (did recidivate) class. This disproportional assignment results in a significantly higher false positive rate for blacks as compared to whites. On the other hand, the classifier puts the other group (whites) on a relative advantage by disproportionately misclassifying positive (did recidivate) subjects from this group into negative (did not recidivate) class (resulting in a higher false negative rate). Note that this scenario resembles our synthetic example Case I in Section 5.1.2.

For the SQF data, the (unconstrained) logistic regression classifier leads to an accuracy of 0.751. However, the classifier yields false positive rates of 0.38 and 0.11, respectively, for blacks and whites (*i.e.*, $DM_{FPR} = 0.27$), and false negative rates of 0.19 and 0.31 (*i.e.*, $DM_{FNR} = -0.12$). Notice that unlike the COMPAS dataset, being classified positive in here is an advantageous outcome—positive class in this case is not being stopped whereas the positive class in the COMPAS dataset is being classified as being a recidivist. This scenario also resembles our synthetic example Case I in Section 5.1.2.

Next, we apply our framework on a logistic regression classifier to mitigate disparate mistreatment with respect to false positive rate, false negative rate, and on both. Figure 11 shows the link between the empirical decision boundary covariance when imposing the constraints to remove disparate mistreatment based on false positive rate or false negative rate on the ProPublica COMPAS and NYPD SQF datasets. The figure shows that: (i) a lower value of false positive or false negative rate covariance corresponds to a lower degree of disparate mistreatment with respect to the corresponding error rate, *i.e.*, Eqs. (5.2) and (5.3) respectively; and, (ii) a zero disparate mistreatment roughly corresponds to zero



ProPublica COMPAS data NYPD SQF data

Figure 11: [Disparate mistreatment constraints] The figure shows the effect of applying Our method$_{\text{sen}}$ to limit disparate mistreatment with respect to false positive rate and false negative rate, as defined by Eqs. (5.2) and (5.3)), respectively. Each type of constraint is applied separately. The plots show the correspondence between the empirical covariance (for each constraint type) and disparate mistreatment caused by the classifiers trained under fairness constraints on ProPublica COMPAS (left) and NYPD SQF (right) datasets. Solid lines correspond to the training data whereas dashed lines correspond to the test data. The figure shows that a decreasing empirical covariance leads to lower disparate mistreatment. Moreover, the statistics for training and test sets show a close correspondence, meaning that the constraints do not seem to be overfitting on the training data.

		FPR constraints			FNR constraints			Both constraints		
		Acc	D_{FPR}	D_{FNR}	Acc	D_{FPR}	D_{FNR}	Acc	D_{FPR}	D_{FNR}
ProPuclica COMPAS	**Our method$_{\text{sen}}$**	0.653	0.03	−0.10	0.656	−0.05	−0.01	0.654	−0.02	−0.03
	Baseline	0.631	0.01	−0.18	0.656	−0.03	−0.03	0.615	−0.19	0.13
	Hardt et al.	0.661	0.01	−0.08	0.654	−0.06	0.01	0.632	0.02	0.01
	Donini et al.	–	–	–	0.649	0.03	−0.09	–	–	–
NYPD SQF	**Our method**	0.633	0.06	−0.01	0.705	0.22	−0.07	0.642	0.05	0.04
	Our method$_{\text{sen}}$	0.727	0.08	0.07	0.743	0.18	0.00	0.726	0.07	0.07
	Baseline	0.527	0.02	−0.08	0.734	0.14	0.01	0.435	−0.71	0.95
	Hardt et al.	0.725	0.03	0.12	0.734	0.14	0.04	0.722	0.02	0.06
	Donini et al.	–	–	–	0.737	0.21	−0.04	–	–	–

Table 2: Performance of different methods while removing disparate mistreatment with respect to false positive rate, false negative rate and both.

covariance (though the correspondence is not completely strict on the ProPublica COMPAS data).

Next, we compare the performance of our scheme with the three alternative methods. While controlling for disparate mistreatment with respect to FPR and FNR simultaneously, the method of Hardt et al. can be interpreted as finding the optimal point that minimizes the loss on the average of the two group-conditional ROC curves (one curve for each sensitive feature group), or the one that minimizes the loss on the point-wise minimum of the two

curves. The optimal point in both cases lies on the point-wise minimum of the two curves. Both variants lead to similar performance, hence we report the results for the former.

Table 2 summarizes the results by showing the trade-off between fairness and accuracy achieved by our method, the methods by Hardt et al. and Donini et al., and the baseline. Similarly to the results in Section 5.1.2, we observe that for all four methods, controlling for disparate mistreatment on false positive rate (false negative rate) also helps decrease disparate mistreatment on false negative rate (false positive rate), at least to a limited extent. Moreover, our method and the methods by Hardt et al. and Donini et al. achieve similar accuracy for a given level of fairness when provided with the same amount of information (sensitive attribute information). The baseline on the other hand leads to the worst performance (especially on the NYPD SQF data.) We also note that the baseline tends to be somewhat unstable and fails to converge to a fair solution in some cases (*e.g.*, both FPR and FNR constraints on COMPAS and SQF datasets).

6. Related Work

6.1. Algorithmic decision making and evidence of unfairness

Algorithmic (or, automated) decision making has been used in applications involving human subjects since several decades. For example, previous works have studied the factors determining whether a criminal defendant would recidivate if granted bail (Burgess, 1928), or whether a loan applicant is likely to return their loan or not (Furletti, 2002). However, with the advent of complex learning methods, and convenient accessibility of 'big data' in several domains, automated decision making has permeated into a large number of human-centric applications, *e.g.*, job screening (Posse, 2016; Taylor, 2016), community safety (Perry, 2013; Lowenkamp, 2009; Kleinberg et al., 2017a), product personalization (Covington et al., 2016; Gomez-Uribe and Hunt, 2015) and online ad delivery (Google AdSense).

Several recent studies have shown that algorithmic decision making systems can potentially lead to (unintentional) unfairness against certain legally "protected" groups (Civil Rights Act, 1964). For example, Sweeney (2013) showed that Google ad platform is more likely to link typically African-American names with having criminal records as compared to typically white names. A recent study by ProPublica (Larson et al., 2016a) found that a criminal risk assessment tool used in Broward County, Florida was disproportionately marking African-American defendants as high risk (as compared to whites), even when they did not recidivate. Since the fairness of a decision making process is often a legally mandated criterion (Barocas and Selbst, 2016; Civil Rights Act, 1964), in the face of potential unfairness by automated decision making systems, a number of studies by regulatory authorities have called for taking fairness into account while designing algorithmic decision systems. (Podesta et al., 2014; Muñoz et al., 2016; Ramirez et al., 2016).

While automated decision making spans a variety of tasks including classification, ranking and recommendations, in this paper, we only focus on fairness in the context of classification. Next, we discuss the state-of-the-art in the context of the notions of unfairness discussed in Section 1.

6.2. Avoiding unfairness in classification

In this section, we will discuss techniques that aim to remove disparate treatment, disparate impact or disparate mistreatment from classification outcomes.

The first study on fair classification dates back to 2008 when Pedreschi et al. (2008) proposed techniques to avoid unfairness in classification rule mining. In the years that followed, a number of studies proposed techniques to remove unfairness from classification outcomes. Especially, last year or so has seen a flurry of methods proposed to control unfairness in classification. These studies operate by first specifying one or more measures of unfairness that they aim to control, *i.e.*, disparate treatment, disparate impact or disparate mistreatment, and then propose techniques to control for the selected measure(s).

These techniques can be divided into three different categories: *pre-processing*, *in-processing* and *post-processing*. Below, we discuss each of these categories separately.

6.2.1. Pre-processing

This technique consists of pre-processing the training data that would later be fed to a training algorithm Kamiran and Calders (2010); Luong et al. (2011); Feldman et al. (2015); Calmon et al. (2017). The goal is to pre-process the training data such that *any* classification algorithm trained on this data would generate unfairness-free outcomes. This strategy can be roughly divided into two different sub-categories. Below, we briefly discuss these subcategories:

The first sub-category involves changing the values of class labels for certain data points Kamiran and Calders (2010); Luong et al. (2011). For example, Kamiran and Calders (2010) propose a pre-processing technique that operates by first training an unconstrained classifier, and then moving / duplicating the data points from the group with lower acceptance rate (as compared to the other group) until the classification outcomes are free of disparate impact.

The second sub-category involves perturbing the non-sensitive features Feldman et al. (2015), or mapping the data to a transformed space Calmon et al. (2017). For example, building on ideas in the area of privacy-preserving data analysis (specifically t-closeness), Feldman et al. (2015) "repair" the non-sensitive features such that it is impossible to predict the sensitive features from non-sensitive features (which in turn means that the classifier trained on this data will not incur disparate impact), while ensuring that the resulting distribution is close to the original data distribution.

On the plus side, the pre-processing techniques have an advantage that the transformed dataset can be used to train any downstream algorithm.

However, these techniques also suffer from some disadvantages. First, since these techniques are not optimized for any specific classification model, and treat the learning algorithm as a black box, as a consequence, the pre-processing can lead to unpredictable loss in accuracy or may not remove unfairness on the test data (as we saw in Section 5.2.1). Furthermore, transforming the dataset might also affect the explainability of the classifier—*e.g.*, since the feature values were transformed during pre-processing, the feature weights of a linear classifiers might not be interpretable anymore.

Method	Type	DT	DI	DM	BN	Polyvalent sens.	Multiple sens.	Range of classifiers
Our framework	In	✓	✓	✓	✓	✓	✓	Any convex margin-based
Kamiran and Calders (2010)	Pre	✓	✓	✗	✗	✗	✗	Any score-based
Calders and Verwer (2010)	In/Post	✓	✓	✗	✗	✗	✗	Naive Bayes
Kamiran et al. (2010)	In	✓	✓	✗	✗	✗	✗	Decision tree
Luong et al. (2011)	Pre	✓	✗	✗	✗	✗	✗	Any
Kamishima et al. (2011)	In	✗	✓	✗	✗	✗	✗	Logistic regression
Zemel et al. (2013)	Pre/In	✓	✓	✗	✗	✗	✗	Log loss
Feldman et al. (2015)	Pre	✓	✓	✗	✗	✓	✓	Any (only numerical features)
Edwards and Storkey (2016)	Pre/In	✓	✓	✗	✗	✓	✓	MLPs
Goh et al. (2016)	In	✓	✓	✓	✗	✓	✓	Ramp loss
Hardt et al. (2016)	Post	✗	✗	✓	✗	✓	✓	Any score-based
Corbett-Davies et al. (2017)	Post	✗	✓	✓	✗	✓	✓	Any score-based
Woodworth et al. (2017)	In	✗	✗	✓	✗	✗	✗	Any convex linear
Quadrianto and Sharmanska (2017)	In	✓	✓	✓	✗	✗	✗	Hinge loss
Calmon et al. (2017)	Pre	✓	✓	✗	✗	✓	✓	Any
Dwork et al. (2018)	In/Post	✗	✓	✓	✗	✓	✓	Any score-based
Menon and Williamson (2018)	Post	✗	✓	✓	✗	✗	✗	Any score-based
Madras et al. (2018)	Pre/In	✓	✓	✓	✗	✗	✗	MLPs
Agarwal et al. (2018)	In/Post	✓	✓	✓	✗	✓	✓	Any score-based
Donini et al. (2018)	Pre/In	✓	✗	✓	✗	✓	✓	SVM

Table 3: Capabilities of different methods in mitigating disparate treatment (DT), disparate impact (DI) and disparate mistreatment (DM). We also show the type of each method: pre-processing (pre), in-processing (in) and post-processing (post). None of the prior methods addresses disparate impact's business necessity (BN) clause. Many of the methods do not generalize to multiple (*e.g.*, gender and race) or polyvalent sensitive features (*e.g.*, race, that has more than two values). The strategy by Feldman et al. (2015) is limited to only numerical non-sensitive features.

6.2.2. In-processing

The second strategy consists of modifying the training procedure of the classifier. Examples of this scheme include Calders and Verwer (2010); Kamishima et al. (2011); Goh et al. (2016); Woodworth et al. (2017); Kamiran et al. (2010); Quadrianto and Sharmanska (2017). Our proposed covariance constraints also fall under this category.

For example, the technique by Kamishima et al. (2011)—which is only limited to a logistic regression classifier—works by adding a regularization term in the objective that penalizes the mutual information between the sensitive feature and the classifier decisions. The method of Kamiran et al. (2010), which is limited to a decision tree classifier, operates by changing the splitting or the leaf node labeling criterion of the tree learning phase to remove disparate impact.

Goh et al. (2016), Woodworth et al. (2017) and Quadrianto and Sharmanska (2017) on the other hand suggest adding constraints similar to ours to the classification model. However, their works are only limited to a single specific loss function (Goh et al., 2016; Quadrianto and Sharmanska, 2017) or to a single notion of unfairness (Woodworth et al., 2017).

Zemel et al. (2013), building on Dwork et al. (2012), combined pre-processing and in-processing by jointly learning a 'fair' representation of the data and the classifier parameters. The joint representation is learnt using a multi-objective loss function that ensures that (i) the resulting representations do not lead to disparate impact, (ii) the reconstruction loss from the original data and intermediate representations is small and (iii) the class label can be predicted with high accuracy. This approach has two main limitations: i) it leads to a non-convex optimization problem and does not guarantee optimality, and ii) the accuracy of the classifier depends on the dimension of the fair representation, which needs to be chosen rather arbitrarily. Inspired by Zemel et al. (2013), the methods of Edwards and Storkey (2016) and Madras et al. (2018) also aim at learning fair representations of the data.

6.2.3. Post-processing

The third and final strategy consists of post-processing the classifier scores such that the new outcomes contain no disparate impact or disparate mistreatment (Hardt et al., 2016; Corbett-Davies et al., 2017; Dwork et al., 2018; Menon and Williamson, 2018).

This approach usually involves learning different decision thresholds for a given score function to remove unfairness (specifically, disparate impact or disparate mistreatment). However, since these strategies require the sensitive feature information at the decision time, they cannot be used in cases where sensitive feature information is unavailable (*e.g.*, due to privacy reasons) or prohibited from being used due to disparate treatment laws (Barocas and Selbst, 2016).

Dwork et al. (2018) combine the in-processing and post-processing scheme by first training a number of classifiers for each group (with each classifier having different acceptance rate for the given group), and then selecting the group-conditional classifiers that minimize a certain loss function. The loss function is formulated as a combination of the loss in accuracy and a penalty term penalizing the deviation from the fairness criterion. Like Hardt et al. (2016) and Corbett-Davies et al. (2017), this method too requires access to the sensitive feature information at the decision time.

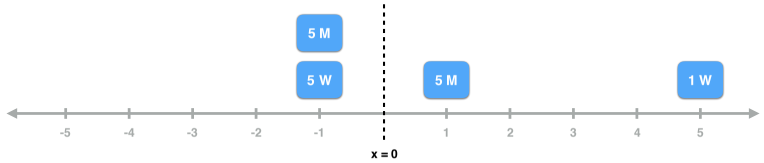


Figure 12: Covariance constraints may perform unfavorably in the presence of outliers. The figure shows a hypothetical dataset with just one feature (x) with values ranging from -5 to 5. Data points belong to two groups: men (M) or women (W). Each box shows the number of subjects of from a certain group (M or W) with that feature value. The decision boundary is at $x = 0$. The decision boundary covariance in this case is 0, yet the disparity in positive class outcome rates between men and women (0.5 for men and 0.17 for women) is very high. This situation is caused by one woman with feature value 5—this outlier point cancels out the effect of five normal examples (W with feature value -1) while computing the covariance.

In addition to the issues discussed above, most prior studies suffer from one or more of the following limitations: (i) they only accommodate a single, binary sensitive feature, (ii) they are restricted to a narrow range of classifiers, and, (iii) they cannot accommodate multiple unfairness notions *simultaneously*. Table 3 compares the capabilities of different methods in meeting different fairness criteria.

Finally, some recent studies (Kusner et al., 2017; Kilbertus et al., 2017) focus on detecting and removing unfairness by leveraging causal inference techniques. However, these studies often require access to causal graphs specifying causal relationships between different features, which can be quite challenging to obtain in practice.

7. Discussion, limitations and future work

In this work, we introduced a constraint-based framework to design fair margin-based classifiers that do not suffer from disparate treatment, disparate impact and disparate mistreatment. The key technical innovation of our framework is a general and intuitive measure of decision boundary unfairness, which serves as a tractable proxy for the above mentioned unfairness notions. Evaluation on various real-world datasets shows that our constrained-based framework is effective in limiting various forms of unfairness from classification outcomes, often at a small cost in terms of accuracy.

However, we also note that covariance mechanism only serves as a *proxy* for the unfairness measure under consideration (*i.e.*, disparate impact or disparate mistreatment). As a result, one may encounter cases where a zero value of the covariance proxy still results in a non-zero value of the unfairness measure. Such situations can arise due to various reasons. Below, we (non-exhaustively) list some of the reasons and leave a detailed formal analysis of such reasons for future work.

First, since our mechanism relies on empirically estimating the decision boundary covariance, very small presence of a certain group in the dataset can lead to poor estimate of the covariance and might not fully remove unfairness. Moreover, while the post-processing

schemes to remove unfairness (Hardt et al., 2016; Corbett-Davies et al., 2017) operate on the data of dimensionality 1 (that is, the scalar score assigned to each item by the classifier), our method operates by using all the features used in classification in order to compute the decision boundary covariance. As a result, our method is expected to suffer more from the data sparsity problem.

Second, we also notice that our method might not perform well in the presence of outliers. Consider for instance the example shown in Figure 12, where an outlier point causes the decision boundary covariance with respect to disparate impact (Eq. (4.2)) to be zero, even when the disparity in positive class outcomes caused by the corresponding decision boundary is very high. However, such outliers can in fact deteriorate the performance of any learning task (Bishop, 2006), even when no other constraints are applied, and one might wish to remove such outliers before training any classification model.

Third, while we observed that a decreasing covariance threshold corresponds to a more fair classifier (with respect to disparate impact or disparate mistreatment), the relation between the two is only empirically observed. A precise mapping between covariance and the precise value of the fairness notion under consideration is quite challenging to derive analytically, since it depends on the specific classifier and the dataset being used. Such a theoretical analysis would be an interesting future direction.

Our framework also opens many avenues for future work. For example, one could include fairness constraints in other supervised (*e.g.*, regression, recommendation) as well as unsupervised (*e.g.*, set selection, ranking) learning tasks. Our fairness constraints can be solved using standard convex or convex-concave optimizers (*e.g.*, SLSQP and DCCP, respectively) and we faced no scalability issues during the experiments conducted in this paper. However, extending these solvers to scale to cases when datasets or resulting optimization problems are too large to fit into memory would be an interesting future direction.

Appendix A. Additional dataset details

In this section, we show the distribution of sensitive features and class labels in the real-world datasets used in the evaluation (Section 5).

Gender	Low income (-ve)	High income (+ve)	Total
Males	20,988(69%)	9,539(21%)	30,527(100%)
Females	13,026(88%)	1,669(12%)	14,695(100%)
Total	34,014(75%)	11,208(25%)	45,222(100%)

Race	Low income (-ve)	High income (+ve)	Total
American-Indian/Eskimo	382(88%)	53(12%)	435(100%)
Asian/Pacific-Islander	934(72%)	369(28%)	1,303(100%)
White	28,696(74%)	10,207(26%)	38,903(100%)
Black	3,694(87%)	534(13%)	4,228(100%)
Other	308(87%)	45(13%)	353(100%)
Total	34,014(75%)	11,208(25%)	45,222(100%)

Table 4: [Adult dataset] High ($>$ 50K USD) and low income ($\leq$ 50K USD) rates in for gender and race groups.

Age	Yes (+ve)	No (-ve)	Total
$25 \leq$ age ≤ 60	3,970(10%)	35,240(90%)	39,210(100%)
age < 25 or age > 60	670(34%)	1,308(66%)	1,978(100%)
Total	4,640(11%)	36,548(89%)	41,188(100%)

Table 5: [Bank dataset] Term deposit subscription rates for the two race groups.

Race	Yes (+ve)	No (-ve)	Total
Black	1,661(52%)	1,514(48%)	3,175(100%)
White	8,22(39%)	1,281(61%)	2,103(100%)
Total	2,483(47%)	2,795(53%)	5,278(100%)

Table 6: [ProPublica COMPAS dataset] Recidivism rates both races.

Race	Yes (-ve)	No (+ve)	Total
Black	2,113(3%)	77,337(97%)	79,450
White	803(15%)	4,616(85%)	5,419
Total	2,916(3%)	81,953(97%)	84,869

Table 7: Persons found to be in possession of a weapon in 2012 NYPD SQF dataset (original).

Race	Yes (-ve)	No (+ve)	Total
Black	2,113(43%)	2,756(57%)	4,869
White	803(83%)	160(17%)	963
Total	2,916(50%)	2,916(50%)	5,832

Table 8: Persons found to be in possession of a weapon in 2012 NYPD SQF dataset (class-balanced).

References

Adult. http://tinyurl.com/UCI-Adult, 1996.

Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. A Reductions Approach to Fair Classification. In *ICML*, 2018.

Andrew Altman. Discrimination. In *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, 2016. https://plato.stanford.edu/archives/win2016/entries/discrimination/.

Julia Angwin and Jeff Larson. Bias in Criminal Risk Scores Is Mathematically Inevitable, Researchers Say. https://www.propublica.org/article/bias-in-criminal-risk-scores-is-mathematically-inevitable-researchers-say.

Bank. http://tinyurl.com/UCI-Bank, 2014.

Solon Barocas and Andrew D. Selbst. Big Data's Disparate Impact. *California Law Review*, 2016.

Esha Bhandari. Big Data Can Be Used To Violate Civil Rights Laws, and the FTC Agrees, 2016. https://www.aclu.org/blog/free-future/big-data-can-be-used-violate-civil-rights-laws-and-ftc-agrees.

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

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. 2004.

Ernest W. Burgess. Factors Determining Success or Failure on Parole. *The Workings of the Indeterminate Sentence Law and the Parole System in Illinois*, 1928.

Toon Calders and Sicco Verwer. Three Naive Bayes Approaches for Discrimination-Free Classification. *Data Mining and Knowledge Discovery*, 2010.

Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized Pre-Processing for Discrimination Prevention. In *NIPS*, 2017.

Alexandra Chouldechova. Fair Prediction with Disparate Impact:A Study of Bias in Recidivism Prediction Instruments. *arXiv preprint, arXiv:1610.07524*, 2016.

Civil Rights Act. Civil Rights Act of 1964, Title VII, Equal Employment Opportunities, 1964.

Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. Algorithmic Decision Making and the Cost of Fairness. In *KDD*, 2017.

Paul Covington, Jay Adams, and Emre Sargin. Deep Neural Networks for YouTube Recommendations. In *RecSys*, 2016.

Yann N. Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and Attacking the Saddle Point Problem in High-dimensional Non-convex Optimization. In *NIPS*, 2014.

Michele Donini, Luca Oneto, Shai Ben-David, John Shawe-Taylor, and Massimiliano Pontil. Empirical Risk Minimization under Fairness Constraints. In *NIPS*. 2018.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, and Omer Reingold. Fairness Through Awareness. In *ITCSC*, 2012.

Cynthia Dwork, Nicole Immorlica, Adam Tauman Kalai, and Mark DM Leiserson. Decoupled Classifiers for Group-fair and Efficient Machine Learning. In *Conference on Fairness, Accountability and Transparency*, 2018.

Harrison Edwards and Amos Storkey. Censoring Representations with an Adversary. In *ICLR*, 2016.

Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and Removing Disparate Impact. In *KDD*, 2015.

FICO. FICO Score, 2017. https://en.wikipedia.org/wiki/Credit_score_in_the_United_States.

Anthony W. Flores, Christopher T. Lowenkamp, and Kristin Bechtel. False Positives, False Negatives, and False Analyses: A Rejoinder to "Machine Bias: There's Software Used Across the Country to Predict Future Criminals. And it's Biased Against Blacks.". 2016.

Mark J. Furletti. An Overview and History of Credit Reporting, 2002. http://dx.doi.org/10.2139/ssrn.927487.

Alex Gano. Disparate impact and mortgage lending: A beginner's guide. *U. Colo. L. Rev.*, 88:1109, 2017.

Andrew Gelman, Jeffrey Fagan, and Alex Kiss. An analysis of the new york city police department's stop-and-frisk policy in the context of claims of racial bias. *Journal of the American Statistical Association*, 102(479):813–823, 2007.

James E Gentle, Wolfgang Karl Härdle, and Yuichi Mori. *Handbook of Computational Statistics: Concepts and Methods*. Springer Science & Business Media, 2012.

Sharad Goel, Justin M. Rao, and Ravi Shroff. Precinct or Prejudice? Understanding Racial Disparities in New York City's Stop-and-Frisk Policy. *Annals of Applied Statistics*, 2015.

Gabriel Goh, Andrew Cotter, Maya Gupta, and Michael Friedlander. Satisfying Real-world Goals with Dataset Constraints. In *NIPS*, 2016.

Carlos A. Gomez-Uribe and Neil Hunt. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Trans. Manage. Inf. Syst.*, 2015.

Google AdSense. https://www.google.com/adsense.

Moritz Hardt, Eric Price, and Nathan Srebro. Equality of Opportunity in Supervised Learning. In *NIPS*, 2016.

Faisal Kamiran and Toon Calders. Classifying without Discriminating. In *IC4*, 2009.

Faisal Kamiran and Toon Calders. Classification with No Discrimination by Preferential Sampling. In *BENELEARN*, 2010.

Faisal Kamiran, Toon Calders, and Mykola Pechenizkiy. Discrimination aware decision tree learning. In *ICDM*, 2010.

Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware Classifier with Prejudice Remover Regularizer. In *PADM*, 2011.

Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing Fairness Gerrymandering: Auditing and Learning for Subgroup Fairness. In *ICML*, 2018.

Niki Kilbertus, Mateo Rojas Carulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. Avoiding Discrimination through Causal Reasoning. In *NIPS*. 2017.

Jon Kleinberg, Himabindu Lakkaraju, Jure Leskovec, Jens Ludwig, and Sendhil Mullainathan. Human Decisions and Machine Predictions. 2017a. Working paper.

Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent Trade-Offs in the Fair Determination of Risk Scores. In *ITCS*, 2017b.

Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual Fairness. In *NIPS*. 2017.

Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. https://github.com/propublica/compas-analysis, 2016a.

Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. How We Analyzed the COMPAS Recidivism Algorithm. https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm, 2016b.

Yun Liu, Krishna Gadepalli, Mohammad Norouzi, George E. Dahl, Timo Kohlberger, Aleksey Boyko, Subhashini Venugopalan, Aleksei Timofeev, Philip Q. Nelson, Greg S. Corrado, Jason D. Hipp, Lily Peng, and Martin C. Stumpe. Detecting Cancer Metastases on Gigapixel Pathology Images. *arXiv preprint arXiv:1703.02442*, 2017.

Christopher T Lowenkamp. The Development of an Actuarial Risk Assessment Instrument for US Pretrial Services. *Fed. Probation*, 2009.

Binh Thanh Luong, Salvatore Ruggieri, and Franco Turini. kNN as an Implementation of Situation Testing for Discrimination Discovery and Prevention. In *KDD*, 2011.

David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning Adversarially Fair and Transferable Representations. In *ICML*, 2018.

Aditya Krishna Menon and Robert C. Williamson. The Cost of Fairness in Classification. *arXiv:1705.09055*, 2017.

Aditya Krishna Menon and Robert C Williamson. The Cost of Fairness in Binary Classification. In *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, 2018.

Cecilia Muñoz, Megan Smith, and DJ Patil. Big Data: A Report on Algorithmic Systems, Opportunity, and Civil Rights. *Executive Office of the President. The White House.*, 2016.

Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. Discrimination-aware Data Mining. In *KDD*, 2008.

Walt L Perry. *Predictive Policing: The Role of Crime Forecasting in Law Enforcement Operations*. Rand Corporation, 2013.

Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On Fairness and Calibration. In *NIPS*. 2017.

John Podesta, Penny Pritzker, Ernest Moniz, John Holdren, and Jeffrey Zients. Big Data: Seizing Opportunities, Preserving Values. *Executive Office of the President. The White House.*, 2014.

Christian Posse. Cloud Jobs API: Machine Learning Goes to Work on Job Search and Discovery, 2016. https://cloud.google.com/blog/big-data/2016/11/cloud-jobs-api-machine-learning-goes-to-work-on-job-search-and-discovery.

Novi Quadrianto and Viktoriia Sharmanska. Recycling Privileged Learning and Distribution Matching for Fairness. In *NIPS*, 2017.

Edith Ramirez, Julie Brill, Maureen K. Ohlhausen, and Terrell McSweeny. Big Data: A Tool for Inclusion or Exclusion? Understanding the Issues. *Federal Trade Commission (FTC) Report*, 2016.

Cristóbal Romero and Sebastián Ventura. Preface to the Special Issue on Data Mining for Personalised Educational Systems. *User Modeling and User-Adapted Interaction*, 2011.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, 2002.

Xinyue Shen, Steven Diamond, Yuantao Gu, and Stephen Boyd. Disciplined Convex-Concave Programming. *arXiv:1604.02639*, 2016.

Reva B Siegel. Race-Conscious but Race-Neutral: The Constitutionality of Disparate Impact in the Roberts Court. *Ala. L. Rev.*, 66:653, 2014.

Stop, Question and Frisk Data. http://www1.nyc.gov/site/nypd/stats/reports-analysis/stopfrisk.page, 2017.

Latanya Sweeney. Discrimination in Online Ad Delivery. *ACM Queue*, 2013.

Tess Taylor. Recruiting will be an elementary task for Watson, says IBM, 2016. http://www.hrdive.com/news/recruiting-will-be-an-elementary-task-for-watson-says-ibm/427692/.

Blake Woodworth, Suriya Gunasekar, Mesrob I. Ohannessian, and Nathan Srebro. Learning Non-Discriminatory Predictors. In *COLT*, 2017.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. Fairness Beyond Disparate Treatment & Disparate Impact: Learning Classification without Disparate Mistreatment. In *WWW*, 2017a.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. Fairness Constraints: Mechanisms for Fair Classification. In *AISTATS*, 2017b.

R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning Fair Representations. In *ICML*, 2013.

Journal of Machine Learning Research 20 (2019) 1-70 Submitted 5/18; Revised 4/19; Published 4/19

Generalized Score Matching for Non-Negative Data

Shiqing Yu SQYU@UW.EDU
Department of Statistics
University of Washington, Seattle, WA, U.S.A.

Mathias Drton MD5@UW.EDU
Department of Mathematical Sciences
University of Copenhagen, Copenhagen, Denmark
and
Department of Statistics
University of Washington, Seattle, WA, U.S.A.

Ali Shojaie ASHOJAIE@UW.EDU
Department of Biostatistics
University of Washington, Seattle, WA, U.S.A.

Editor: Aapo Hyvarinen

Abstract

A common challenge in estimating parameters of probability density functions is the intractability of the normalizing constant. While in such cases maximum likelihood estimation may be implemented using numerical integration, the approach becomes computationally intensive. The score matching method of Hyvärinen (2005) avoids direct calculation of the normalizing constant and yields closed-form estimates for exponential families of continuous distributions over $\mathbb{R}^m$. Hyvärinen (2007) extended the approach to distributions supported on the non-negative orthant, $\mathbb{R}^m_+$. In this paper, we give a generalized form of score matching for non-negative data that improves estimation efficiency. As an example, we consider a general class of pairwise interaction models. Addressing an overlooked inexistence problem, we generalize the regularized score matching method of Lin et al. (2016) and improve its theoretical guarantees for non-negative Gaussian graphical models.

Keywords: exponential family, graphical model, positive data, score matching, sparsity

1. Introduction

Score matching was first developed in Hyvärinen (2005) for continuous distributions supported on all of $\mathbb{R}^m$. Consider such a distribution P_0, with density p_0 and support equal to $\mathbb{R}^m$. Let $\mathcal{P}$ be a family of distributions with twice continuously differentiable densities. The score matching estimator of p_0 using $\mathcal{P}$ as a model is the minimizer of the expected squared ℓ_2 distance between the gradients of $\log p_0$ and a log-density from $\mathcal{P}$. So we minimize the loss $\int_{\mathbb{R}^m} p_0(\boldsymbol{x}) \|\nabla \log p(\boldsymbol{x}) - \nabla \log p_0(\boldsymbol{x})\|_2^2 \, \mathrm{d}\boldsymbol{x}$ with respect to densities p from $\mathcal{P}$. The loss depends on p_0, but integration by parts can be used to rewrite it in a form that can be approximated by averaging over the sample without knowing p_0. A key feature of score matching is that normalizing constants cancel in gradients of log-densities, allowing for simple treatment of models with intractable normalizing constants. For exponential families, the loss is quadratic in the canonical parameter, making optimization straightforward.

©2019 Shiqing Yu, Mathias Drton, Ali Shojaie.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-278.html.

If the considered distributions are supported on a proper subset of $\mathbb{R}^m$, then the integration by parts arguments underlying the score matching estimator may fail due to discontinuities at the boundary of the support. For data supported on the non-negative orthant $\mathbb{R}^m_+$, Hyvärinen (2007) addresses this problem by modifying the loss to $\int_{\mathbb{R}^m} p_0(\boldsymbol{x}) \|\nabla \log p(\boldsymbol{x}) \circ \boldsymbol{x} - \nabla \log p_0(\boldsymbol{x}) \circ \boldsymbol{x}\|_2^2 \, \mathrm{d}\boldsymbol{x}$, where $\circ$ denotes entrywise multiplication. In this loss, boundary effects are dampened by multiplying gradients elementwise with the identity functions x_j.

In this paper, we propose *generalized score matching* methods that are based on elementwise multiplication with functions other than x_j. As we show, this can lead to drastically improved estimation accuracy, both theoretically and empirically. To demonstrate these advantages, we consider a family of graphical models on $\mathbb{R}^m_+$, which does not have tractable normalizing constants and hence serves as a practical example.

Graphical models specify conditional independence relations for a random vector $\boldsymbol{X} = (X_i)_{i \in V}$ indexed by the nodes of a graph (Lauritzen, 1996). For undirected graphs, variables X_i and X_j are required to be conditionally independent given $(X_k)_{k \neq i,j}$ if there is no edge between i and j. The smallest undirected graph with this property is the *conditional independence graph* of $\boldsymbol{X}$. Estimation of this graph and associated interaction parameters has been a topic of continued research as reviewed by Drton and Maathuis (2017).

Largely due to their tractability, Gaussian graphical models (GGMs) have gained great popularity. The conditional independence graph of a multivariate normal vector $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is determined by the *inverse covariance matrix* $\mathbf{K} \equiv \boldsymbol{\Sigma}^{-1}$, also termed *concentration* or *precision matrix*. Specifically, X_i and X_j are conditionally independent given all other variables if and only if the (i,j)-th and the (j,i)-th entries of $\mathbf{K}$ are both zero. This simple relation underlies a rich literature including Drton and Perlman (2004), Meinshausen and Bühlmann (2006), Yuan and Lin (2007) and Friedman et al. (2008), among others.

More recent work has provided tractable procedures also for non-Gaussian graphical models. This includes Gaussian copula models (Liu et al., 2009; Dobra and Lenkoski, 2011; Liu et al., 2012), Ising models (Ravikumar et al., 2010), other exponential family models (Chen et al., 2015; Yang et al., 2015), as well as semi- or non-parametric estimation techniques (Fellinghauer et al., 2013; Voorman et al., 2014). In this paper, we apply our method to a class of pairwise interaction models that generalizes non-negative Gaussian random variables, as recently considered by Lin et al. (2016) and Yu et al. (2016), as well as square root graphical models proposed by Inouye et al. (2016) when the sufficient statistic function is a pure power. However, our main ideas can also be applied for other classes of exponential families whose support is restricted to a rectangular set.

Our focus will be on *pairwise interaction power models* with probability distributions having (Lebesgue) densities proportional to

$$\exp\left\{-\frac{1}{2a}\boldsymbol{x}^{a\top}\mathbf{K}\boldsymbol{x}^a + \boldsymbol{\eta}^\top \frac{\boldsymbol{x}^b - \mathbf{1}_m}{b}\right\} \tag{1}$$

on $\mathbb{R}^m_+ \equiv [0,\infty)^m$. Here $a > 0$ and $b \geq 0$ are known constants, and $\mathbf{K} \in \mathbb{R}^{m \times m}$ and $\boldsymbol{\eta} \in \mathbb{R}^m$ are unknown parameters of interest. When $b = 0$ we define $(x^b - 1)/b \equiv \log x$ and $\mathbb{R}^m_+ \equiv (0,\infty)^m$. This class of models is motivated by the form of important univariate distributions for non-negative data, including gamma and truncated normal distributions. It provides a framework for pairwise interaction that is concrete yet rich enough to capture key differences in how densities may behave at the boundary of the non-negative orthant,

$\mathbb{R}^m_+$. Moreover, the conditional independence graph of a random vector $\boldsymbol{X}$ with distribution as in (1) is determined just as in the Gaussian case: X_i and X_j are conditionally independent given all other variables if and only if $\kappa_{ij} = \kappa_{ji} = 0$ in the interaction matrix $\mathbf{K}$. Section 5.1 gives further details on these models. We will develop estimators of $(\boldsymbol{\eta}, \mathbf{K})$ in (1) and the associated conditional independence graph using the proposed *generalized score matching*.

A special case of (1) are truncated Gaussian graphical models, with $a = b = 1$. Let $\boldsymbol{\mu} \in \mathbb{R}^m$, and let $\mathbf{K}$ be a positive definite matrix. Then a non-negative random vector $\boldsymbol{X}$ follows a truncated normal distribution for mean parameter $\boldsymbol{\mu}$ and inverse covariance parameter $\mathbf{K}$, in symbols $\boldsymbol{X} \sim \mathrm{TN}(\boldsymbol{\mu}, \mathbf{K})$, if it has density proportional to

$$\exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \mathbf{K}(\boldsymbol{x}-\boldsymbol{\mu})\right\} \tag{2}$$

on $\mathbb{R}^m_+$. We refer to $\boldsymbol{\Sigma} = \mathbf{K}^{-1}$ as the covariance parameter of the distribution, and note that the $\boldsymbol{\eta}$ parameter in (1) is $\mathbf{K}\boldsymbol{\mu}$. Another special case of (1) is the exponential square root graphical models in Inouye et al. (2016), where $a = b = 1/2$.

Lin et al. (2016) estimate truncated GGMs based on Hyvärinen's modification, with an ℓ_1 penalty on the entries of $\mathbf{K}$ added to the loss. However, the paper overlooks the fact that the loss can be unbounded from below in the high-dimensional setting even with an ℓ_1 penalty, such that no minimizer may exist. Since the unpenalized loss is quadratic in the parameter to be estimated, we propose modifying it by adding small positive values to the diagonals of the positive semi-definite matrix that defines the quadratic part, in order to ensure that the loss is bounded and strongly convex and admits a unique minimizer. We apply this to the estimator for GGMs considered in Lin et al. (2016), which uses score-matching on $\mathbb{R}^m$, and to the *generalized score matching* estimator for pairwise interaction power models on $\mathbb{R}^m_+$ proposed in this paper. In these cases, we show, both empirically and theoretically, that the consistency results still hold (or even improve) if the positive values added are smaller than a threshold that is readily computable.

The rest of the paper is organized as follows. Section 2 introduces score matching and our proposed *generalized score matching*. In Section 3, we apply generalized score matching to exponential families, with univariate truncated normal distributions as an example. *Regularized generalized score matching* for graphical models is formulated in Section 4. The estimators for pairwise interaction power models are shown in Section 5, while theoretical consistency results are presented in Section 6, where we treat the probabilistically most tractable case of truncated GGMs. Simulation results and applications to RNAseq data are given in Section 7. Proofs for theorems in Sections 2–6 are presented in Appendices A and B. Additional experimental results are presented in Appendix C.

1.1. Notation

Constant scalars, vectors, and functions are written in lower-case (e.g., a, $\boldsymbol{a}$), random scalars and vectors in upper-case (e.g., X, $\boldsymbol{X}$). Regular font is used for scalars (e.g. a, X), and boldface for vectors (e.g. $\boldsymbol{a}$, $\boldsymbol{X}$). Matrices are in upright bold, with constant matrices in upper-case ($\mathbf{K}$, $\mathbf{M}$) and random matrices holding observations in lower-case ($\mathbf{x}$, $\mathbf{y}$). Subscripts refer to entries in vectors and columns in matrices. Superscripts refer to rows in matrices. So X_j is the j-th component of a random vector $\boldsymbol{X}$. For a data matrix

$\mathbf{x} \in \mathbb{R}^{n\times m}$, each row comprising one observation of m variables/features, $X_j^{(i)}$ is the j-th feature for the i-th observation. Stacking the columns of a matrix $\mathbf{K} = [\kappa_{ij}]_{i,j} \in \mathbb{R}^{q\times r}$ gives its vectorization $\mathrm{vec}(\mathbf{K}) = (\kappa_{11}, \ldots, \kappa_{q1}, \kappa_{12}, \ldots, \kappa_{q2}, \ldots, \kappa_{1r}, \ldots, \kappa_{qr})^\top$. For a matrix $\mathbf{K} \in \mathbb{R}^{q\times q}$, $\mathrm{diag}(\mathbf{K}) \in \mathbb{R}^q$ denotes its diagonal, and for a vector $\boldsymbol{v} \in \mathbb{R}^q$, $\mathrm{diag}(\boldsymbol{v})$ is the $q \times q$ diagonal matrix with diagonals $v_1, \ldots, v_q$.

For $a \geq 1$, the ℓ_a-norm of a vector $\boldsymbol{v} \in \mathbb{R}^q$ is denoted

$$\|\boldsymbol{v}\|_a = \left(\sum_{j=1}^q |v_j|^a\right)^{1/a},$$

with $\|\boldsymbol{v}\|_\infty = \max_{j=1,\ldots,q} |v_j|$. A matrix $\mathbf{K} = [\kappa_{ij}]_{i,j} \in \mathbb{R}^{q\times r}$ has Frobenius norm

$$|||\mathbf{K}|||_F \equiv \|\mathrm{vec}(\mathbf{K})\|_2 \equiv \sqrt{\sum_{i=1}^q \sum_{j=1}^r \kappa_{ij}^2},$$

and max norm $\|\mathbf{K}\|_\infty \equiv \|\mathrm{vec}(\mathbf{K})\|_\infty \equiv \max_{i,j} |\kappa_{ij}|$. Its ℓ_a-ℓ_b operator norm is

$$|||\mathbf{K}|||_{a,b} \equiv \max_{\boldsymbol{x}\neq\mathbf{0}} \frac{\|\mathbf{K}\boldsymbol{x}\|_b}{\|\boldsymbol{x}\|_a}$$

with shorthand notation $|||\mathbf{K}|||_a \equiv |||\mathbf{K}|||_{a,a}$; for instance, $|||\mathbf{K}|||_\infty \equiv \max_{i=1\ldots,q} \sum_{j=1}^r |\kappa_{ij}|$.

For a function $f : \mathbb{R}^m \to \mathbb{R}$, we define $\partial_j f(\boldsymbol{x})$ as the partial derivative with respect to x_j, and $\partial_{jj} f(\boldsymbol{x}) = \partial_j \partial_j f(\boldsymbol{x})$. For $\boldsymbol{f} : \mathbb{R} \to \mathbb{R}^m$, $\boldsymbol{f}(x) = (f_1(x), \ldots, f_m(x))^\top$, we let $\boldsymbol{f}'(x) = (f_1'(x), \ldots, f_m'(x))^\top$ be the vector of derivatives. Likewise $\boldsymbol{f}''(x)$ is used for second derivatives. The symbol $\mathbb{1}_A(\cdot)$ denotes the indicator function of the set A, while $\mathbf{1}_n \in \mathbb{R}^n$ is the vector of all 1's. For $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^m$, $\boldsymbol{a} \circ \boldsymbol{b} \equiv (a_1 b_1, \ldots, a_m b_m)^\top$. A density of a distribution is always a probability density function with respect to Lebesgue measure. When it is clear from the context, $\mathbb{E}_0$ denotes the expectation under a true distribution P_0.

2. Score Matching

In this section, we review the original score matching and develop our generalized score matching estimators.

2.1. Original Score Matching

Let $\boldsymbol{X}$ be a random vector taking values in $\mathbb{R}^m$ with distribution P_0 and density p_0. Let $\mathcal{P}$ be a family of distributions of interest with twice continuously differentiable densities supported on $\mathbb{R}^m$. Suppose $P_0 \in \mathcal{P}$. The *score matching loss* for $P \in \mathcal{P}$, with density p, is given by

$$J(P) = \int_{\mathbb{R}^m} p_0(\boldsymbol{x}) \|\nabla \log p(\boldsymbol{x}) - \nabla \log p_0(\boldsymbol{x})\|_2^2 \, \mathrm{d}\boldsymbol{x}. \tag{3}$$

The gradients in (3) can be thought of as gradients with respect to a hypothetical location parameter, evaluated at the origin (Hyvärinen, 2005). The loss $J(P)$ is minimized if and only

if $P = P_0$, which forms the basis for estimation of P_0. Importantly, since the loss depends on p only through its log-gradient, it suffices to know p up to a normalizing constant. Under mild conditions, (3) can be rewritten as

$$J(P) = \int_{\mathbb{R}^m} p_0(\boldsymbol{x}) \sum_{j=1}^{m} \left[\partial_{jj} \log p(\boldsymbol{x}) + \frac{(\partial_j \log p(\boldsymbol{x}))^2}{2} \right] \mathrm{d}\boldsymbol{x}, \tag{4}$$

plus a constant independent of p. The integral in (4) can be approximated by a sample average; this alleviates the need for knowing the true density p_0, and provides a way to estimate p_0.

2.2. Generalized Score Matching for Non-Negative Data

When the true density p_0 is supported on a proper subset of $\mathbb{R}^m$, the integration by parts underlying the equivalence of (3) and (4) may fail due to discontinuity at the boundary. For distributions supported on the non-negative orthant, $\mathbb{R}^m_+$, Hyvärinen (2007) addressed this issue by instead minimizing the *non-negative score matching loss*

$$J_+(P) = \int_{\mathbb{R}^m_+} p_0(\boldsymbol{x}) \|\nabla \log p(\boldsymbol{x}) \circ \boldsymbol{x} - \nabla \log p_0(\boldsymbol{x}) \circ \boldsymbol{x}\|_2^2 \, \mathrm{d}\boldsymbol{x}. \tag{5}$$

This loss can be motivated via gradients with respect to a hypothetical scale parameter (Hyvärinen, 2007). Under mild conditions, $J_+(P)$ can again be rewritten in terms of an expectation of a function independent of p_0, thus allowing one to form a sample loss.

In this work, we consider generalizing the non-negative score matching loss as follows.

Definition 1 *Let $\mathcal{P}_+$ be the family of distributions of interest, and assume every $P \in \mathcal{P}_+$ has a twice continuously differentiable density supported on $\mathbb{R}^m_+$. Suppose the m-variate random vector $\boldsymbol{X}$ has true distribution $P_0 \in \mathcal{P}_+$, and let p_0 be its twice continuously differentiable density. Let $h_1, \ldots, h_m : \mathbb{R}_+ \to \mathbb{R}_+$ be a.s. positive functions that are absolutely continuous in every bounded sub-interval of $\mathbb{R}_+$, and set $\boldsymbol{h}(\boldsymbol{x}) = (h_1(x_1), \ldots, h_m(x_m))^\top$. For $P \in \mathcal{P}_+$ with density p, the* generalized $\boldsymbol{h}$-score matching loss *is*

$$J_{\boldsymbol{h}}(P) = \int_{\mathbb{R}^m_+} \frac{1}{2} p_0(\boldsymbol{x}) \|\nabla \log p(\boldsymbol{x}) \circ \boldsymbol{h}(\boldsymbol{x})^{1/2} - \nabla \log p_0(\boldsymbol{x}) \circ \boldsymbol{h}(\boldsymbol{x})^{1/2}\|_2^2 \, \mathrm{d}\boldsymbol{x}, \tag{6}$$

where $\boldsymbol{h}^{1/2}(\boldsymbol{x}) \equiv (h_1^{1/2}(x_1), \ldots, h_m^{1/2}(x_m))^\top$.

Proposition 2 *The distribution P_0 is the unique minimizer of $J_{\boldsymbol{h}}(P)$ for $P \in \mathcal{P}_+$.*

Proof First, observe that $J_{\boldsymbol{h}}(P) \geq 0$ and $J_{\boldsymbol{h}}(P_0) = 0$. For uniqueness, suppose $J_{\boldsymbol{h}}(P_1) = 0$ for some $P_1 \in \mathcal{P}_+$. Let p_0 and p_1 be the respective densities. By assumption $p_0(\boldsymbol{x}) > 0$ a.s. and $h_j^{1/2}(\boldsymbol{x}) > 0$ a.s. for all $j = 1, \ldots, m$. Therefore, we must have $\nabla \log p_1(\boldsymbol{x}) = \nabla \log p_0(\boldsymbol{x})$ a.s., or equivalently, $p_1(\boldsymbol{x}) = \text{const} \times p_0(\boldsymbol{x})$ almost surely in $\mathbb{R}^m_+$. Since p_1 and p_0 are continuous densities supported on $\mathbb{R}^m_+$, it follows that $p_1(\boldsymbol{x}) = p_0(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^m_+$. ■

Choosing all $h_j(x) = x^2$ recovers the loss from (5). In our generalization, we will focus on using functions h_j that are increasing but are bounded or grow rather slowly. This will alleviate the need to estimate higher moments, leading to better practical performance and improved theoretical guarantees.

We will consider the following assumptions:

(A1) $p_0(\boldsymbol{x}) h_j(x_j) \partial_j \log p(\boldsymbol{x}) \Big|_{x_j \searrow 0^+}^{x_j \nearrow +\infty} = 0$, $\quad \forall \boldsymbol{x}_{-j} \in \mathbb{R}_+^{m-1}, \quad \forall p \in \mathcal{P}_+$;

(A2) $\mathbb{E}_{p_0} \|\nabla \log p(\boldsymbol{X}) \circ \boldsymbol{h}^{1/2}(\boldsymbol{X})\|_2^2 < +\infty$, $\mathbb{E}_{p_0} \|(\nabla \log p(\boldsymbol{X}) \circ \boldsymbol{h}(\boldsymbol{X}))'\|_1 < +\infty$, $\forall p \in \mathcal{P}_+$,

where $\partial_j \log p(\boldsymbol{x}) \equiv \frac{\partial \log p(\boldsymbol{y})}{\partial y_j}\Big|_{\boldsymbol{y}=\boldsymbol{x}}$, $f(\boldsymbol{x}) \Big|_{x_j \searrow 0^+}^{x_j \nearrow +\infty} \equiv \lim_{x_j \nearrow +\infty} f(\boldsymbol{x}) - \lim_{x_j \searrow 0} f(\boldsymbol{x})$, "$\forall p \in \mathcal{P}_+$" is a shorthand for "for all p being the density of some $P \in \mathcal{P}_+$", and the prime symbol denotes component-wise differentiation. While the second half of (A2) was not made explicit in Hyvärinen (2005, 2007), (A1)-(A2) were both required for integration by parts and Fubini-Tonelli to apply.

Once the forms of p_0 and p are given, sufficient conditions for $\boldsymbol{h}$ for Assumptions (A1)-(A2) to hold are easy to find. In particular, (A1) and (A2) are easily satisfied and verified for exponential families.

Integration by parts yields the following theorem which shows that $J_{\boldsymbol{h}}$ from (6) is an expectation (under P_0) of a function that does not depend on p_0, similar to (4). The proof is given in Appendix A.1.

Theorem 3 *Under (A1) and (A2), the loss from (6) equals*

$$J_{\boldsymbol{h}}(P) = \int_{\mathbb{R}^m_+} p_0(\boldsymbol{x}) \sum_{j=1}^{m} \Big[h_j'(x_j) \partial_j (\log p(\boldsymbol{x})) + h_j(x_j) \partial_{jj} (\log p(\boldsymbol{x})) + \frac{1}{2} h_j(x_j) \left(\partial_j (\log p(\boldsymbol{x}))\right)^2 \Big] \mathrm{d}\boldsymbol{x} \tag{7}$$

plus a constant independent of p.

Given a data matrix $\mathbf{x} \in \mathbb{R}^{n \times m}$ with rows $\boldsymbol{X}^{(i)}$, we define the sample version of (7) as

$$\hat{J}_{\boldsymbol{h}}(P) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{m} \left\{ h_j'(X_j^{(i)}) \partial_j (\log p(\boldsymbol{X}^{(i)})) + h_j(X_j^{(i)}) \left[\partial_{jj} (\log p(\boldsymbol{X}^{(i)})) + \frac{1}{2} \left(\partial_j (\log p(\boldsymbol{X}^{(i)})) \right)^2 \right] \right\}. \tag{8}$$

Subsequently, for a distribution P with density p, we let $J_{\boldsymbol{h}}(p) \equiv J_{\boldsymbol{h}}(P)$. Similarly, when a distribution $P_{\boldsymbol{\theta}}$ with density $p_{\boldsymbol{\theta}}$ is associated to a parameter vector $\boldsymbol{\theta}$, we write $J_{\boldsymbol{h}}(\boldsymbol{\theta}) \equiv J_{\boldsymbol{h}}(p_{\boldsymbol{\theta}}) \equiv J_{\boldsymbol{h}}(P_{\boldsymbol{\theta}})$. We apply similar conventions to the sample version $\hat{J}_{\boldsymbol{h}}(P)$. We note that this type of loss is also treated in slightly different settings in Parry (2016) and Almeida and Gidas (1993).

Remark 4 In the one-dimensional case, using the notation in Parry et al. (2012), $J_{\boldsymbol{h}}(P)$ and $\hat{J}_{\boldsymbol{h}}(P)$ correspond to $d(P_0, P)$ and $S(x, P)$, respectively, and can be generated by $\phi(x, p, p_1) \equiv -h(x)p_1^2/(2p)$ (c.f. Equations (39), (51), (53) and Section 10.1 therein). Thus Theorem 3 follows from this correspondence. While (A1) is equivalent to the condition implied by the boundary divergence $d_b = 0$ in that paper, (A2), which we assume for invoking Fubini-Tonelli due to multi-dimensionality, is not present. On the other hand, while Parry (2016) treats the multivariate case, it does not cover the connection between our $J_{\boldsymbol{h}}$ and $\hat{J}_{\boldsymbol{h}}$. Since ϕ is concave but not strictly concave in (p, p_1), the results in Parry (2016) only imply that P_0 is *a* minimizer, a weaker conclusion than Proposition 2.

3. Exponential Families

In this section, we study the case where $\mathcal{P}_+ \equiv \{p_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$ is an exponential family comprising continuous distributions with support $\mathbb{R}_+^m$. More specifically, we consider densities that are indexed by the canonical parameter $\boldsymbol{\theta} \in \mathbb{R}^r$ and have the form

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{t}(\boldsymbol{x}) - \psi(\boldsymbol{\theta}) + b(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}_+^m, \tag{9}$$

where $\boldsymbol{t}(\boldsymbol{x}) \in \mathbb{R}_+^r$ comprises the sufficient statistics, $\psi(\boldsymbol{\theta})$ is a normalizing constant depending on $\boldsymbol{\theta}$ only, and $b(\boldsymbol{x})$ is the base measure, with $\boldsymbol{t}$ and b a.s. differentiable with respect to each component. Define $\boldsymbol{t}_{\boldsymbol{j}}'(\boldsymbol{x}) \equiv (\partial_j t_1(\boldsymbol{x}), \ldots, \partial_j t_r(\boldsymbol{x}))^\top$ and $b_j'(\boldsymbol{x}) \equiv \partial_j b(\boldsymbol{x})$.

Theorem 5 *Under Assumptions (A1)-(A2) from Section 2.2, the empirical generalized $\boldsymbol{h}$-score matching loss (8) can be rewritten as a quadratic function in $\boldsymbol{\theta} \in \mathbb{R}^r$:*

$$\hat{J}_{\boldsymbol{h}}(p_{\boldsymbol{\theta}}) = \frac{1}{2}\boldsymbol{\theta}^\top \boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta} - \boldsymbol{g}(\mathbf{x})^\top \boldsymbol{\theta} + \text{const}, \quad \textit{where} \tag{10}$$

$$\boldsymbol{\Gamma}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^n \sum_{j=1}^m h_j(X_j^{(i)}) \boldsymbol{t}_j'(\boldsymbol{X}^{(i)}) \boldsymbol{t}_j'(\boldsymbol{X}^{(i)})^\top \quad \textit{and} \tag{11}$$

$$\boldsymbol{g}(\mathbf{x}) = -\frac{1}{n}\sum_{i=1}^n \sum_{j=1}^m \left[h_j(X_j^{(i)}) b_j'(\boldsymbol{X}^{(i)}) \boldsymbol{t}_j'(\boldsymbol{X}^{(i)}) + h_j(X_j^{(i)}) \boldsymbol{t}_j''(\boldsymbol{X}^{(i)}) + h_j'(X_j^{(i)}) \boldsymbol{t}_j'(\boldsymbol{X}_i) \right] \tag{12}$$

are sample averages of functions of the data matrix $\mathbf{x}$ only.

Define $\boldsymbol{\Gamma}_0 \equiv \mathbb{E}_{p_0}\boldsymbol{\Gamma}(\mathbf{x})$, $\boldsymbol{g}_0 \equiv \mathbb{E}_{p_0}\boldsymbol{g}(\mathbf{x})$, and $\boldsymbol{\Sigma}_0 \equiv \mathbb{E}_{p_0}[(\boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta}_0 - g(\mathbf{x}))(\boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta}_0 - g(\mathbf{x}))^\top]$.

Theorem 6 *Suppose that*

(C1) $\boldsymbol{\Gamma}$ is a.s. invertible, and

(C2) $\boldsymbol{\Gamma}_0$, $\boldsymbol{\Gamma}_0^{-1}$, $\boldsymbol{g}_0$ and $\boldsymbol{\Sigma}_0$ exist and are entry-wise finite.

Then the minimizer of (10) is a.s. unique with closed-form solution $\hat{\boldsymbol{\theta}} \equiv \boldsymbol{\Gamma}(\mathbf{x})^{-1}\boldsymbol{g}(\mathbf{x})$. Moreover,

$$\hat{\boldsymbol{\theta}} \to_{a.s.} \boldsymbol{\theta}_0 \quad \textit{and} \quad \sqrt{n}(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0) \to_d \mathcal{N}_r\left(\mathbf{0}, \boldsymbol{\Gamma}_0^{-1}\boldsymbol{\Sigma}_0\boldsymbol{\Gamma}_0^{-1}\right) \quad \textit{as } n \to \infty.$$

Theorems 5 and 6 are proved in Appendix A.2. Theorem 5 clarifies the quadratic nature of the loss, and Theorem 6 provides a basis for asymptotically valid tests and confidence intervals for the parameter $\boldsymbol{\theta}$. Note that Condition (C1) holds if and only if $h_j(X_j) > 0$ a.s. and $[\boldsymbol{t}_j'(\boldsymbol{X}^{(1)}), \ldots, \boldsymbol{t}_j'(\boldsymbol{X}^{(n)})] \in \mathbb{R}^{r\times n}$ has rank r a.s. for some $j = 1, \ldots, m$.

The conclusion in Theorem 6 indicates that, similar to the estimator in Hyvärinen (2007) with $h_j(x) = x^2$, the closed-form solution for our generalized $\hat{\boldsymbol{\theta}}$ allows one to consistently estimate the canonical parameter in an exponential family distribution without needing to calculate the often complicated normalizing constant $\psi(\boldsymbol{\theta})$ or resort to numerical methods. Computational details are explicated in Section 5.3.

Below we illustrate the estimator $\hat{\boldsymbol{\theta}}$ in the case of univariate truncated normal distributions. We assume (A1)-(A2) and (C1)-(C2) throughout.

Example 3.1 *Univariate ($m = r = 1$) truncated normal distributions for mean parameter μ and variance parameter σ^2 have density*

$$p_{\mu,\sigma^2}(x) \propto \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}, \quad x \in \mathbb{R}_+. \tag{13}$$

If σ^2 is known but μ unknown, then writing the density in canonical form as in (9) yields

$$p_\theta(x) \propto \exp\{\theta t(x) + b(x)\}, \quad \theta \equiv \frac{\mu}{\sigma^2}, \quad t(x) \equiv x, \quad b(x) = -\frac{x^2}{2\sigma^2}.$$

Given an i.i.d. sample $X_1, \ldots, X_n \sim p_{\mu_0,\sigma^2}$, the generalized h-score matching estimator of μ is

$$\hat{\mu}_h \equiv \frac{\sum_{i=1}^n h(X_i)X_i - \sigma^2 h'(X_i)}{\sum_{i=1}^n h(X_i)}.$$

If $\lim_{x\searrow 0^+} h(x) = 0$, $\lim_{x\nearrow+\infty} h^2(x)(x-\mu_0)p_{\mu_0,\sigma^2}(x) = 0$ and the expectations are finite (for example, when $h(x) = o(\exp(Mx^2))$ for $M < \frac{1}{4\sigma^2}$), then

$$\sqrt{n}(\hat{\mu}_h - \mu_0) \to_d \mathcal{N}\left(0, \frac{\mathbb{E}_0[\sigma^2 h^2(X) + \sigma^4 h'^2(X)]}{\mathbb{E}_0^2[h(X)]}\right).$$

We recall that the Cramér-Rao lower bound *(i.e. the lower bound on the variance of any unbiased estimator) for estimating μ is*

$$\frac{\sigma^4}{\text{var}(X - \mu_0)}.$$

Example 3.2 *Consider the univariate truncated normal distributions from (13) in the setting where the mean parameter μ is known but the variance parameter $\sigma^2 > 0$ is unknown. In canonical form as in (9), we write*

$$p_\theta(x) \propto \exp\{\theta t(x) + b(x)\}, \quad \theta \equiv \frac{1}{\sigma^2}, \quad t(x) \equiv -(x-\mu)^2/2, \quad b(x) = 0.$$

Given an i.i.d. sample $X_1, \ldots, X_n \sim p_{\mu,\sigma_0^2}$, the generalized h-score matching estimator of σ^2 is

$$\hat{\sigma}_h^2 \equiv \frac{\sum_{i=1}^n h(X_i)(X_i-\mu)^2}{\sum_{i=1}^n h(X_i) + h'(X_i)(X_i-\mu)}.$$

If, in addition to the assumptions in Example 3.1, $\lim_{x\nearrow+\infty} h^2(x)(x-\mu)^3 p_{\mu,\sigma_0^2}(x) = 0$, *then*

$$\sqrt{n}(\hat{\sigma}_h^2 - \sigma_0^2) \to_d \mathcal{N}\left(0, \frac{2\sigma_0^6 \mathbb{E}_0[h^2(X)(X-\mu)^2] + \sigma_0^8 \mathbb{E}_0[h'^2(X)(X-\mu)^2]}{\mathbb{E}_0^2[h(X)(X-\mu)^2]}\right).$$

Moreover, the Cramér-Rao lower bound *for estimating* σ^2 *is*

$$\frac{4\sigma_0^8}{\mathrm{var}(X-\mu)^2}.$$

Remark 7 In Example 3.2, if $\mu_0 = 0$, then $h(x) \equiv 1$ also satisfies (A1)-(A2) and (C1)-(C2) and one recovers the sample variance $\frac{1}{n}\sum_i X_i^2$, which obtains the Cramér-Rao lower bound.

In these examples, there is a benefit in using a bounded function h, which can be explained as follows. When $\mu \gg \sigma$, there is effectively no truncation to the Gaussian distribution, and our method adapts to using low moments in (6), since a bounded and increasing $h(x)$ becomes almost constant as it reaches its asymptote for x large. Hence, we effectively revert to the original score matching (recall Section 2.1). In the other cases, the truncation effect is significant and our estimator uses higher moments accordingly.

Figure 1 plots the asymptotic variance of $\hat{\mu}_h$ from Example 3.1, with $\sigma = 1$ known. Efficiency as measured by the Cramér-Rao lower bound divided by the asymptotic variance is also shown. We see that two truncated versions of $\log(1+x)$ have asymptotic variance close to the Cramér-Rao bound. This asymptotic variance is also reflective of the variance for smaller finite samples.

Figure 2 is the analog of Figure 1 for $\hat{\sigma}_h^2$ from Example 3.2 with $\mu = 0.5$ known. While the specifics are a bit different the benefits of using bounded or slowly growing h are again clear. We note that when σ is small, the effect of truncation to the positive part of the real line is small.

In both plots we order/color the curves based on their overall efficiency, so they have different colors in one from the other, although the same functions are presented. For all functions presented here (A1)–(A2) and (C1)–(C2) are satisfied.

4. Regularized Generalized Score Matching

In high-dimensional settings, when the number r of parameters to estimate may be larger than the sample size n, it is hard, if not impossible, to estimate the parameters consistently without turning to some form of regularization. More specifically, for exponential families, condition (C1) in Section 3 fails when $r > n$. A popular approach is then the use of ℓ_1 regularization to exploit possible sparsity.

Let the data matrix $\mathbf{x} \in \mathbb{R}^{n\times m}$ comprise n i.i.d. samples from distribution P_0. Assume P_0 has density p_0 belonging to an exponential family $\mathcal{P}_+ \equiv \{p_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$, where $\boldsymbol{\Theta} \subseteq \mathbb{R}^r$. Adding an ℓ_1 penalty to (10), we obtain the regularized generalized score matching loss

$$\frac{1}{2}\boldsymbol{\theta}^\top \boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta} - \boldsymbol{g}(\mathbf{x})^\top \boldsymbol{\theta} + \lambda\|\boldsymbol{\theta}\|_1 \tag{14}$$

as in Lin et al. (2016). The loss in (14) involves a quadratic smooth part as in the familiar lasso loss for linear regression. However, although the matrix $\boldsymbol{\Gamma}$ is positive semidefinite,

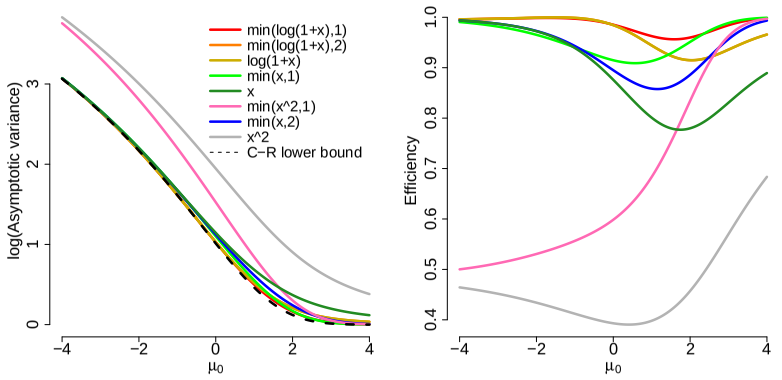


Figure 1: Log of asymptotic variance and efficiency with respect to the Cramér-Rao bound for $\hat{\mu}_h$ ($\sigma^2 = 1$ known).

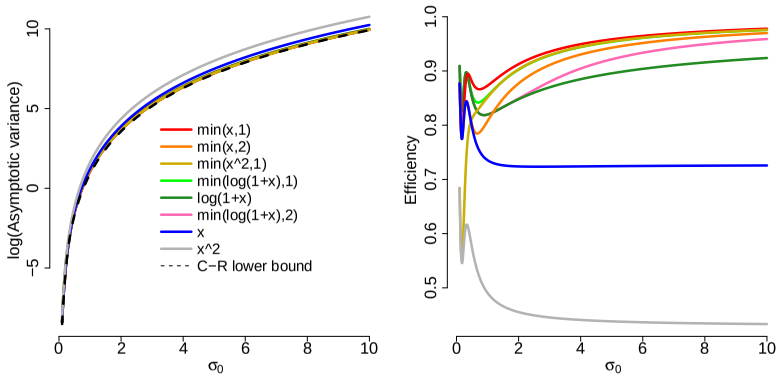


Figure 2: Log of asymptotic variance and efficiency with respect to the Cramér-Rao bound for $\hat{\sigma}_h^2$ ($\mu = 0.5$ known).

the regularized loss in (14) is not guaranteed to be bounded unless the tuning parameter λ is sufficiently large—a problem that does not occur in lasso. We note that here, and throughout, we suppress the dependence on the data $\mathbf{x}$ for $\boldsymbol{\Gamma}(\mathbf{x})$, $\boldsymbol{g}(\mathbf{x})$ and derived quantities.

For a more detailed explanation, note that that by (11), $\boldsymbol{\Gamma} = \mathbf{H}^\top\mathbf{H}$ for some $\mathbf{H} \in \mathbb{R}^{nm\times r}$. In the high-dimensional case, the rank of $\boldsymbol{\Gamma}$, or equivalently $\mathbf{H}$, is at most $nm < r$. Hence, $\boldsymbol{\Gamma}$ is not invertible and $\boldsymbol{g}$ does not necessarily lie in the column span of $\boldsymbol{\Gamma}$. Let $\mathrm{Ker}(\boldsymbol{\Gamma})$ be the kernel of $\boldsymbol{\Gamma}$. Then there may exist $\boldsymbol{\nu} \in \mathrm{Ker}(\boldsymbol{\Gamma})$ with $\boldsymbol{g}^\top\boldsymbol{\nu} \neq 0$. In this case, if

$$0 \leq \lambda < \sup_{\boldsymbol{\nu}\in\mathrm{Ker}(\boldsymbol{\Gamma})} |\boldsymbol{g}^\top\boldsymbol{\nu}|/\|\boldsymbol{\nu}\|_1,$$

there exists $\boldsymbol{\nu} \in \mathrm{Ker}(\boldsymbol{\Gamma})$ with $\frac{1}{2}\boldsymbol{\nu}^\top\boldsymbol{\Gamma}\boldsymbol{\nu} = 0$ and $-\boldsymbol{g}^\top\boldsymbol{\nu} + \lambda\|\boldsymbol{\nu}\|_1 < 0$. Evaluating at $\boldsymbol{\theta}(a) = a\cdot\boldsymbol{\nu}$ for scalar $a > 0$, the loss becomes $a\left(-\boldsymbol{g}^\top\boldsymbol{\nu} + \lambda\|\boldsymbol{\nu}\|_1\right)$, which is negative and linear in a, and thus unbounded below. In this case no minimizer of (14) exists for small values of λ. This issue also exists for the estimators from Zhang and Zou (2014) and Liu and Luo (2015), which correspond to score matching for GGMs. We note that in the context of estimating the interaction matrix in pairwise models, $r = m^2$; thus, the condition $nm < r$ reduces to $n < m$, or $n < m + 1$ when both $\mathbf{K}$ and $\boldsymbol{\eta}$ are estimated.

To circumvent the unboundedness problem, we add small values $\gamma_\ell > 0$ to the diagonal entries of $\boldsymbol{\Gamma}$, which become $\boldsymbol{\Gamma}_{\ell,\ell} + \gamma_\ell$, $\ell = 1,\ldots,r$. This is in the spirit of work such as Ledoit and Wolf (2004) and corresponds to an elastic net-type penalty (Zou and Hastie, 2005) with weighted ℓ_2 penalty $\sum_{\ell=1}^r \gamma_\ell\theta_\ell^2$. After this modification, $\boldsymbol{\Gamma}$ is positive definite, our regularized loss is strongly convex in $\boldsymbol{\theta}$, and a unique minimizer exists for all $\lambda \geq 0$. For the special case of truncated GGMs, we will show that a result on consistent estimation holds if we choose $\gamma_\ell = \delta_0\boldsymbol{\Gamma}_{\ell,\ell}$ for a suitably small constant $\delta_0 > 0$, for which we propose a particular choice to avoid tuning. This choice of γ_ℓ depends on the data through $\boldsymbol{\Gamma}_{\ell,\ell}$.

Definition 8 *For $\boldsymbol{\gamma} \in \mathbb{R}_+^r\backslash\{\mathbf{0}\}$, let $\boldsymbol{\Gamma}_{\boldsymbol{\gamma}} \equiv \boldsymbol{\Gamma} + \mathrm{diag}(\boldsymbol{\gamma})$. The* regularized generalized $\boldsymbol{h}$-score matching estimator *with tuning parameter $\lambda \geq 0$ and* amplifier $\boldsymbol{\gamma}$ *is the estimator*

$$\hat{\boldsymbol{\theta}} \in \operatorname*{argmin}_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \hat{J}_{\boldsymbol{h},\lambda,\boldsymbol{\gamma}}(\boldsymbol{\theta}) \equiv \operatorname*{argmin}_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \frac{1}{2}\boldsymbol{\theta}^\top\boldsymbol{\Gamma}_{\boldsymbol{\gamma}}(\mathbf{x})\boldsymbol{\theta} - \boldsymbol{g}(\mathbf{x})^\top\boldsymbol{\theta} + \lambda\|\boldsymbol{\theta}\|_1. \tag{15}$$

In the case where $\boldsymbol{\gamma} = (\delta - 1)\mathrm{diag}(\boldsymbol{\Gamma})$ for some $\delta > 1$, we also call δ the *multiplier*. We note that $\hat{\boldsymbol{\theta}}$ from (15) is a *piecewise linear* function of λ (Lin et al., 2016).

5. Score Matching for Graphical Models for Non-negative Data

In this section we apply our generalized score matching estimator to a general class of graphical models for non-negative data.

5.1. A General Framework of Pairwise Interaction Models

We consider the class of pairwise interaction power models with density introduced in (1). We recall the form of the density:

$$p_{\boldsymbol{\eta},\mathbf{K}}(\boldsymbol{x}) \propto \exp\left(-\frac{1}{2a}\boldsymbol{x}^{a\top}\mathbf{K}\boldsymbol{x}^a + \boldsymbol{\eta}^\top\frac{\boldsymbol{x}^b - \mathbf{1}_m}{b}\right)\mathbb{1}_{\mathbb{R}_+^m}(\boldsymbol{x}), \tag{16}$$

where a and b are known constants, and the interaction matrix $\mathbf{K}$ and the vector $\boldsymbol{\eta}$ are parameters. When $b = 0$, we use the convention that $\frac{x^0-1}{0} \equiv \log x$ and apply the logarithm element-wise. Our focus will be on the interaction matrix $\mathbf{K}$ that determines the conditional independence graph through its support $S(\mathbf{K}) \equiv \{(i,j) : \kappa_{ij} \neq 0\}$. However, unless $\boldsymbol{\eta}$ is known or assumed to be zero, we also need to estimate $\boldsymbol{\eta}$ as a nuisance parameter. In the case where we assume $\boldsymbol{\eta} \equiv \mathbf{0}$ is known (i.e. the linear part $(\boldsymbol{x}^b - \mathbf{1}_m)/b$ is not present), we call the distribution (and the corresponding estimator) a *centered* distribution (estimator), in contrast to the general case termed *non-centered* when we assume $\boldsymbol{\eta} \neq \mathbf{0}$ or unknown.

We first give a set of sufficient conditions for the density to be valid, i.e., the right-hand side of (16) to be integrable. The proof is given in Appendix A.3.

Theorem 9 *Define conditions*

(CC1) $\mathbf{K}$ *is* strictly co-positive, *i.e.,* $\boldsymbol{v}^\top\mathbf{K}\boldsymbol{v} > 0$ *for all* $\boldsymbol{v} \in \mathbb{R}_+^m\backslash\{\mathbf{0}\}$*;*

(CC2) $2a > b > 0$*;*

(CC3) $a > 0$, $b = 0$, *and* $\eta_j > -1$ *for* $j = 1,\ldots,m$ *($\boldsymbol{\eta} \succ -\mathbf{1}_m$).*

In the non-centered case, if (CC1) and one of (CC2) and (CC3) holds, then the function on the right-hand side of (16) is integrable over $\mathbb{R}_+^m$. In the centered case, (CC1) and $a > 0$ are sufficient.

We emphasize that (CC1) is a weaker condition than positive definiteness. Criteria for strict co-positivity are discussed in Väliaho (1986).

5.2. Implementation for Different Models

In this section we give some implementation details for the regularized generalized $\boldsymbol{h}$-score matching estimator defined in (15) applied to the pairwise interaction models from (16). We again let $\boldsymbol{\Psi} \equiv \left(\mathbf{K}^\top, \boldsymbol{\eta}\right)^\top \in \mathbb{R}^{(m+1)\times m}$. The unregularized loss is then

$$\hat{J}_{\boldsymbol{h}}(P) = \frac{1}{2}\mathrm{vec}(\boldsymbol{\Psi})^\top\boldsymbol{\Gamma}(\boldsymbol{x})\mathrm{vec}(\boldsymbol{\Psi}) - \boldsymbol{g}(\boldsymbol{x})^\top\mathrm{vec}(\boldsymbol{\Psi}).$$

The general form of the matrix $\boldsymbol{\Gamma}$ and the vector $\boldsymbol{g}$ in the loss were given in equations (10)–(12). Here $\boldsymbol{\Gamma} \in \mathbb{R}^{(m+1)m\times(m+1)m}$ is block-diagonal, with the j-th $\mathbb{R}^{(m+1)\times(m+1)}$ block

$$\begin{aligned}\boldsymbol{\Gamma}_j(\mathbf{x}) &\equiv \begin{bmatrix}\boldsymbol{\Gamma}_{11,j} & \boldsymbol{\Gamma}_{12,j}\\ \boldsymbol{\Gamma}_{12,j}^\top & \Gamma_{22,j}\end{bmatrix} \qquad (17)\\ &\equiv \frac{1}{n}\sum_{i=1}^n \begin{bmatrix} h_j\left(X_j^{(i)}\right){X_j^{(i)}}^{2a-2}\boldsymbol{X}^{(i)^a}\boldsymbol{X}^{(i)^{a\top}} & -h_j\left(X_j^{(i)}\right){X_j^{(i)}}^{a+b-2}\boldsymbol{X}^{(i)^a}\\ -h_j\left(X_j^{(i)}\right){X_j^{(i)}}^{a+b-2}\boldsymbol{X}^{(i)^{a\top}} & h_j\left(X_j^{(i)}\right){X_j^{(i)}}^{2b-2}\end{bmatrix}\\ &= \frac{1}{n}\mathbf{y}^\top\mathbf{y}, \quad \mathbf{y} \equiv \left[-(\sqrt{\boldsymbol{h}_j(\boldsymbol{X}_j)}\circ\boldsymbol{X}_j^{a-1})\circ\mathbf{x}^a \quad \sqrt{\boldsymbol{h}_j(\boldsymbol{X}_j)}\circ\boldsymbol{X}_j^{b-1}\right] \in \mathbb{R}^{n,m+1}, \qquad (18)\end{aligned}$$

where the $\circ$ product between a vector and a matrix means an elementwise multiplication of the vector with each *column* of the matrix, and $\boldsymbol{h}_j(\boldsymbol{X}_j) \equiv [h_j(X_j^{(1)}),\ldots,h_j(X_j^{(n)})^\top] \in \mathbb{R}^m$.

Furthermore, $\boldsymbol{g} \equiv \begin{bmatrix} \mathrm{vec}(\mathbf{g}_1) \\ \boldsymbol{g}_2 \end{bmatrix} \in \mathbb{R}^{(m+1)m}$, where $\mathbf{g}_1$ and $\boldsymbol{g}_2$ correspond to each entry of $\mathbf{K}$ and $\boldsymbol{\eta}$, respectively. The j-th column of $\mathbf{g}_1 \in \mathbb{R}^{m\times m}$, written as $\boldsymbol{g}_{1,j}(\mathbf{x})$, is

$$\frac{1}{n}\sum_{i=1}^{n}\left(h_j'\left(X_j^{(i)}\right){X_j^{(i)}}^{a-1} + (a-1)h_j\left(X_j^{(i)}\right){X_j^{(i)}}^{a-2}\right)\boldsymbol{X}^{(i)a} + a h_j\left(X_j^{(i)}\right){X_j^{(i)}}^{2a-2}\boldsymbol{e}_{j,m},$$

where $\boldsymbol{e}_{j,m}$ is the m-vector with 1 at the j-th position and 0 elsewhere, and the j-th entry of $\boldsymbol{g}_2 \in \mathbb{R}^m$ is

$$g_{2,j} = \frac{1}{n}\sum_{i=1}^{n} -h_j'\left(X_j^{(i)}\right){X_j^{(i)}}^{b-1} - (b-1)h_j\left(X_j^{(i)}\right){X_j^{(i)}}^{b-2}.$$

These formulae also hold for $b = 0$ since $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$ only depend on the gradient of the log density, and $\frac{\mathrm{d}(x^b-1)/b}{\mathrm{d}x} = x^{b-1}$ also holds for $b = 0$. In the centered case where we know $\boldsymbol{\eta}_0 \equiv \mathbf{0}$, we only estimate $\mathbf{K} \in \mathbb{R}^{m\times m}$, and $\boldsymbol{\Gamma} \in \mathbb{R}^{m^2\times m^2}$ is still block-diagonal, with the j-th block being the $\boldsymbol{\Gamma}_{11,j}$ submatrix in (17), while $\boldsymbol{g}$ is just $\mathrm{vec}(\mathbf{g}_1)$. Since b only appears in the $\boldsymbol{\eta}$ part of the density, the formulae only depend on a in the centered case.

We emphasize that it is indeed necessary to introduce amplifiers $\boldsymbol{\gamma} \succ \mathbf{0}$ or a multiplier $\delta > 1$ in addition to the ℓ_1 penalty. It is clear from (18) that $\mathrm{rank}(\boldsymbol{\Gamma}_j) \le \min\{n, m+1\}$ (or $\min\{n, m\}$ if centered). Thus, $\boldsymbol{\Gamma}$ is non-invertible when $n \le m$ (or $n < m$ if centered) and $\boldsymbol{g}$ need not lie in its column span.

We claim that including amplifiers/multipliers for the submatrices $\boldsymbol{\Gamma}_{11,j}$ only is sufficient for unique existence of a solution for all penalty parameters $\lambda \ge 0$. To see this, consider any nonzero vector $\boldsymbol{\nu} \in \mathbb{R}^{m+1}$. Partition it as $\boldsymbol{\nu} \equiv (\boldsymbol{\nu}_1, \boldsymbol{\nu}_2)$ with $\boldsymbol{\nu}_1 \in \mathbb{R}^m$. Let $\boldsymbol{\Gamma}_{j,\boldsymbol{\gamma}}$ be our amplified version of the matrix $\boldsymbol{\Gamma}_j$ from (21), so

$$\boldsymbol{\Gamma}_{j,\boldsymbol{\gamma}} \;=\; \begin{pmatrix} \boldsymbol{\Gamma}_{11,j} + \mathrm{diag}(\gamma_1, \ldots, \gamma_m) & \boldsymbol{\Gamma}_{12,j} \\ \boldsymbol{\Gamma}_{12,j}^\top & \Gamma_{22,j} \end{pmatrix}.$$

As $\boldsymbol{\Gamma}_j$ itself is positive semidefinite, we find that if at least one of the first m entries of $\boldsymbol{\nu}$ is nonzero then

$$\boldsymbol{\nu}^\top\boldsymbol{\Gamma}_{j,\boldsymbol{\gamma}}\boldsymbol{\nu} \;\ge\; \boldsymbol{\nu}^\top\boldsymbol{\Gamma}_j\boldsymbol{\nu} + \sum_{k=1}^{m}\nu_k^2\boldsymbol{\gamma}_k \;\ge\; \sum_{k=1}^{m}\nu_k^2\boldsymbol{\gamma}_k \;>\; 0.$$

If only the last entry of $\boldsymbol{\nu}$ is nonzero then

$$\boldsymbol{\nu}^\top\Gamma_{j,\boldsymbol{\gamma}}\boldsymbol{\nu} \;=\; \nu_{m+1}^2\Gamma_{22,j} \;>\; 0$$

almost surely; recall that $\Gamma_{22,j} = \frac{1}{n}\sum_{i=1}^{n} h_j\left(X_j^{(i)}\right)X_j^{2b-2}$. We conclude that $\boldsymbol{\Gamma}_{j,\boldsymbol{\gamma}}$ (and thus the entire amplified $\boldsymbol{\Gamma}$) is a.s. positive definite, which ensures unique existence of the loss minimizer.

Given the formulae for $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$, one adds the ℓ_1 penalty on $\boldsymbol{\Psi}$ to get the regularized loss (24). Our methodology readily accommodates two different choices of the penalty parameter λ for $\mathbf{K}$ and $\boldsymbol{\eta}$. This is also theoretically supported for truncated GGMs, since if the ratio of the respective values $\lambda_{\mathbf{K}}$ and $\lambda_{\boldsymbol{\eta}}$ is fixed, the proof of the theorems in Section 6 can be

easily modified by replacing $\boldsymbol{\eta}$ by $(\lambda_{\boldsymbol{\eta}}/\lambda_{\mathbf{K}})\boldsymbol{\eta}$. To avoid picking two tuning parameters, one may also choose to remove the penalty on $\boldsymbol{\eta}$ altogether by profiling out $\boldsymbol{\eta}$ and solve for $\hat{\boldsymbol{\eta}} \equiv \boldsymbol{\Gamma}_{22}^{-1}\left(\boldsymbol{g}_2 - \boldsymbol{\Gamma}_{12}^\top\mathrm{vec}(\hat{\mathbf{K}})\right)$, with $\hat{\mathbf{K}}$ the minimizer of the profiled loss

$$\hat{J}_{\boldsymbol{h},\lambda,\boldsymbol{\gamma},\mathrm{profile}}(\mathbf{K}) \equiv \frac{1}{2}\mathrm{vec}(\mathbf{K})^\top\boldsymbol{\Gamma}_{\boldsymbol{\gamma},11.2}\mathrm{vec}(\mathbf{K}) - (\boldsymbol{g}_1 - \boldsymbol{\Gamma}_{12}\boldsymbol{\Gamma}_{22}^{-1}\boldsymbol{g}_2)^\top\mathrm{vec}(\mathbf{K}) + \lambda\|\mathbf{K}\|_1, \qquad (19)$$

where the Schur complement $\boldsymbol{\Gamma}_{\boldsymbol{\gamma},11.2} \equiv \boldsymbol{\Gamma}_{\boldsymbol{\gamma},11} - \boldsymbol{\Gamma}_{12}\boldsymbol{\Gamma}_{22}^{-1}\boldsymbol{\Gamma}_{12}^\top$ is a.s. positive definite such that the profiled estimator exists a.s. for all $\lambda \ge 0$. This profiled approach corresponds to choosing $\lambda_{\boldsymbol{\eta}}/\lambda_{\mathbf{K}} = 0$. A detailed theoretical analysis of the profiled estimator is beyond the scope of this paper, however. We note that in the other extreme, with $\lambda_{\boldsymbol{\eta}}/\lambda_{\mathbf{K}} = +\infty$, the non-centered estimator reduces to the estimator from the centered case.

Example 5.3 *The truncated normal model comprises the density*

$$p_{\boldsymbol{\mu},\mathbf{K}}(\boldsymbol{x}) \;\propto\; \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top\mathbf{K}(\boldsymbol{x}-\boldsymbol{\mu})\right\}\mathbb{1}_{[0,\infty)^m}(\boldsymbol{x}). \qquad (20)$$

This corresponds to (16) with $a = b = 1$, and $\boldsymbol{\eta} = \mathbf{K}\boldsymbol{\mu}$. The j-th $(m+1)\times(m+1)$ block of $\boldsymbol{\Gamma}(\mathbf{x})$ is

$$\frac{1}{n}\begin{bmatrix} \mathbf{x}^\top\mathrm{diag}(\boldsymbol{h}_j(\boldsymbol{X}_j))\mathbf{x} & -\mathbf{x}^\top\boldsymbol{h}_j(\boldsymbol{X}_j) \\ -\boldsymbol{h}_j(\boldsymbol{X}_j)^\top\mathbf{x} & \boldsymbol{h}_j(\boldsymbol{X}_j)^\top\mathbf{1}_n \end{bmatrix}. \qquad (21)$$

Partitioning the vector $\boldsymbol{g}(\mathbf{x})$ into m subvectors $\boldsymbol{g}_j(\mathbf{x}) \in \mathbb{R}^{m+1}$, where the entries of $\boldsymbol{g}_j(\mathbf{x})$ correspond to column $\boldsymbol{\Psi}_j$, the k-th entry of $\boldsymbol{g}_j(\mathbf{x})$ is

$$g_{jk}(\mathbf{x}) \equiv \begin{cases} \frac{1}{n}\sum_{i=1}^{n} h_j'\left(X_j^{(i)}\right)X_k^{(i)} & \text{if } k \le m,\ k \ne j, \\ \frac{1}{n}\sum_{i=1}^{n} h_j'\left(X_j^{(i)}\right)X_k^{(i)} + h_j\left(X_j^{(i)}\right) & \text{if } k = j, \\ -\frac{1}{n}\sum_{i=1}^{n} h_j'\left(X_j^{(i)}\right) & \text{if } k = m+1. \end{cases} \qquad (22)$$

Example 5.4 *The exponential square-root graphical model in Inouye et al. (2016) has*

$$p_{\boldsymbol{\eta},\mathbf{K}}(\boldsymbol{x}) \propto \exp\left(-\sqrt{\boldsymbol{x}}^\top\mathbf{K}\sqrt{\boldsymbol{x}} + 2\boldsymbol{\eta}^\top\sqrt{\boldsymbol{x}}\right)\mathbb{1}_{[0,\infty)^m}(\boldsymbol{x}),$$

which corresponds to (16) with $a = b = 1/2$. We refer to this as the exponential *model. In this case, the j-th $\mathbb{R}^{(m+1)\times(m+1)}$ block of $\boldsymbol{\Gamma}$ is*

$$\boldsymbol{\Gamma}_j(\mathbf{x}) \equiv \frac{1}{n}\sum_{i=1}^{n}\frac{h_j\left(X_j^{(i)}\right)}{X_j^{(i)}}\begin{pmatrix} -\sqrt{\boldsymbol{X}^{(i)}} \\ 1 \end{pmatrix}\left(-\sqrt{\boldsymbol{X}^{(i)}}^\top, 1\right)$$

and $\boldsymbol{g} = \mathrm{vec}(\mathbf{g}_0)$, where the j-th column of $\mathbf{g}_0 \in \mathbb{R}^{(m+1)\times m}$ is

$$\boldsymbol{g}_j(\mathbf{x}) \equiv \frac{1}{n}\sum_{i=1}^{n}\frac{2h_j'\left(X_j^{(i)}\right)X_j^{(i)} - h_j\left(X_j^{(i)}\right)}{2{X_j^{(i)}}^{3/2}}\begin{pmatrix} \sqrt{\boldsymbol{X}^{(i)}} \\ -1 \end{pmatrix} + \frac{h_j\left(X_j^{(i)}\right)}{2X_j^{(i)}}\boldsymbol{e}_{j,m+1}.$$

Example 5.5 *If $a = 1/2$ and $b = 0$, then (16) becomes*

$$p_{\boldsymbol{\eta},\mathbf{K}}(\boldsymbol{x}) \propto \exp\left(-\sqrt{\boldsymbol{x}}^\top \mathbf{K}\sqrt{\boldsymbol{x}} + \boldsymbol{\eta}^\top \log(\boldsymbol{x})\right) \mathbb{1}_{(0,\infty)^m}(\boldsymbol{x}). \tag{23}$$

If $\mathbf{K}$ is diagonal in this case, then $\boldsymbol{X} \sim p_{\boldsymbol{\eta},\mathbf{K}}$ has independent entries with X_j following the gamma distribution with rate κ_{jj} and shape $\eta_j + 1$, which gives an intuition for condition (CC3) $\eta_j > -1$ in Theorem 9. We can thus view (23) as a multivariate gamma distribution with pairwise interactions among the covariates, and call this the gamma *model. For this model, the j-th block of $\boldsymbol{\Gamma}$ is*

$$\boldsymbol{\Gamma}_j(\mathbf{x}) \equiv \frac{1}{n}\sum_{i=1}^n \frac{h_j\left(X_j^{(i)}\right)}{{X_j^{(i)}}^2} \begin{pmatrix} -\sqrt{X_j^{(i)}\boldsymbol{X}^{(i)}} \\ 1 \end{pmatrix} \left(-\sqrt{X_j^{(i)}\boldsymbol{X}^{(i)}}^\top, 1\right)$$

and the part of $\boldsymbol{g}$ corresponding to $\mathbf{K}_j$ is

$$\boldsymbol{g}_{1,j}(\mathbf{x}) \equiv \frac{1}{n}\sum_{i=1}^n \frac{2h_j'\left(X_j^{(i)}\right)X_j^{(i)} - h_j\left(X_j^{(i)}\right)}{2{X_j^{(i)}}^{3/2}}\sqrt{\boldsymbol{X}^{(i)}} + \frac{h_j\left(X_j^{(i)}\right)}{2X_j^{(i)}}\boldsymbol{e}_{j,m},$$

while the part for η_j is

$$g_{2,j}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^n \frac{h_j\left(X_j^{(i)}\right)}{{X_j^{(i)}}^2} - \frac{h_j'\left(X_j^{(i)}\right)}{X_j^{(i)}}.$$

We note that the $\boldsymbol{\Gamma}_{11,j}$ sub-matrix of $\boldsymbol{\Gamma}_j$ and the $\boldsymbol{g}_{1,j}$ sub-vector of $\boldsymbol{g}_j$ for the gamma model are the same as those for the exponential model, since $a = 1/2$ in both cases and the parts involving $\mathbf{K}$ in the densities are the same.

5.3. Computational Details

In the most general exponential family setting, as in Eq. (10)–(12) in Theorem 5, the time complexity for forming $\boldsymbol{\Gamma} \in \mathbb{R}^{r\times r}$ and $\boldsymbol{g} \in \mathbb{R}^r$ is $\mathcal{O}\left(nm(f_{b'}(m) + r^2 + r(f_{t'}(m) + f_{t''}(m)))\right)$. Here $f_{b'}(m)$ is the average time complexity for calculating $\partial_j b(\boldsymbol{x})$ over $j = 1,\dots,m$, and similarly $f_{t'}(m)$ for $\partial_j t_\ell(\boldsymbol{x})$ and $f_{t''}(m)$ for $\partial_{jj} t_\ell(\boldsymbol{x})$ over $j = 1,\dots,m$ and $\ell = 1\dots,r$. In many applications, however, these three functions would be constant in m, thus giving an $\mathcal{O}(nmr^2)$ computational complexity, with the dominating term coming from the operations for $\boldsymbol{t}_j'\boldsymbol{t}_j'^\top$ in $\boldsymbol{\Gamma}$ since $\boldsymbol{\Gamma}$ is of dimension $r \times r$.

For pairwise interaction power models, $r = m^2$ and the formula above becomes $\mathcal{O}(nm^5)$. However, since $\boldsymbol{\Gamma}$ is block-diagonal with only m^3 nonzero entries and by the special form of $\boldsymbol{t}(\boldsymbol{x}) = \boldsymbol{x}^a\boldsymbol{x}^{a\top}$, the true complexity is in fact $\mathcal{O}(nm^3)$.

While the introduction of the ℓ_1 penalty inevitably precludes the estimator from having a closed-form solution and introduces non-differentiability, state-of-art numerical optimization algorithms, such as coordinate-descent (Friedman et al., 2007), can be applied for fast estimation. To speed up estimation, one can usually use warm starts using the solution from the previous λ's, as well as lasso-type strong screening rules (Tibshirani et al., 2012) to eliminate components of $\hat{\boldsymbol{\theta}}$ that are known a priori to have zero estimates.

In our implementation for pairwise interaction models of Section 5.1 (that will become available in an R package), we optimize our loss functions with respect to a symmetric matrix $\hat{\mathbf{K}}$; in the non-centered case the vector $\hat{\boldsymbol{\eta}}$ is also included. We use a coordinate-descent method analogous to Algorithm 2 in Lin et al. (2016), where in each step we update each element of $\hat{\mathbf{K}}$ and $\hat{\boldsymbol{\eta}}$ based on the other entries from the previous steps, while maintaining symmetry. In our simulations in Section 7 we always scale the data matrix by column ℓ_2 norms before proceeding to estimation. Note that estimation of $\hat{\mathbf{K}}$ without symmetry can be parallelized as the loss can be decomposed into a sum over the columns.

5.4. Choice of the Function $\boldsymbol{h}$

In this subsection we discuss the requirements on the function $\boldsymbol{h}$ as well as some reasonable choices of $\boldsymbol{h}$.

5.4.1. Requirements on $\boldsymbol{h}$

In Section 2.2, we presented two assumptions (A1) and (A2) under which the generalized score-matching loss is valid, i.e., the integration by parts is justified and Theorem 3 holds. In this section, we present some sufficient (and nearly necessary) requirements on $\boldsymbol{h}$ such that (A1) and (A2) are satisfied.

Definition 10 *Suppose $\boldsymbol{h} : \mathbb{R}_+^m \to \mathbb{R}_+^m$ with $\boldsymbol{h}(\boldsymbol{x}) = (h_1(x_1),\dots,h_m(x_m))^\top$. We write that $\boldsymbol{h} \in \mathcal{H}_{a,b}$ (for simplicity we omit the dependency on m) if for all $j = 1,\dots,m$:*

i) h_j is absolutely continuous in every bounded sub-interval of $\mathbb{R}_+$, and thus has derivative h_j' a.s.;

ii) $h_j(x) > 0$ a.s. on $\mathbb{R}_+$;

iii) h_j and h_j' are both bounded by some piecewise powers of x a.s. on $\mathbb{R}_+$;

iv) $\lim_{x\searrow 0+} h_j(x)/x_j^q = 0$, where $q \equiv \begin{cases} \max\{1-a, 1-b\} & \text{if } b > 0, \\ 1 - \eta_{0,j} & \text{if } b = 0. \end{cases}$

Theorem 11 *Assume every P in the family of distribution $\mathcal{P}_+$ satisfies (CC1)–(CC3) and thus has finite normalizing constants. If $\boldsymbol{h} \in \mathcal{H}_{a,b}$, then (A1) and (A2) are satisfied.*

In centered models, where $\boldsymbol{\eta} \equiv \mathbf{0}$, we can assume $b = 2a$ and iv) in the definition of $\mathcal{H}_{a,2a}$ has $q = 1 - a$. For truncated GGMs, $a = b = 1$, so iv) in Definition 10 is simply $\lim_{x_j\searrow 0^+} h_j(x_j) = 0$.

In the case of $b = 0$, $\boldsymbol{\eta}$ is an unknown parameter, and (CC3) requires each of its component to be greater than -1. If one has prior information on $\boldsymbol{\eta}$ or restricts the parameter space for $\boldsymbol{\eta}$, the requirement reduces to $h_j(x_j) = o(x_j^{1-\eta_{0,j}})$ as $x_j \searrow 0^+$. Otherwise, it suffices to require $h_j(x_j) = o(x_j^2)$. Note that this is only a condition for $x_j \searrow 0^+$, and the globally quadratic behavior of $h_j(x_j) = x_j^2$ from the original score matching is not needed on the entire $\mathbb{R}_+$, leaving opportunities for improvements.

5.4.2. Reasonable Choices of $\boldsymbol{h}$

Assume a common univariate h for all components in $\boldsymbol{h}$. Inspired by Theorem 11, we consider h that behaves like a power of x both as $x \nearrow +\infty$ and as $x \searrow 0^+$. Since the requirements on the two tails are separate, we can choose h to be a piecewise defined function that joins two powers with possibly different degrees. In other words, $h(x) = \min(x^{p_1}, cx^{p_2})$ for some powers $p_1 \geq p_2 \geq 0$ and constant $c > 0$. Only one constant c is required since generalized score matching is invariant to scaling of h. In determining the exact power of p_1 we have the following considerations:

a) In the centered case:

   (i) (A1) and (A2): Theorem 11 requires that $p_1 \geq 1 - a$.

   (ii) "Controlled $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$ for $\boldsymbol{x}^{a}$": We propose avoiding poles at the origin for the entries of $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$. The formula for $\boldsymbol{\Gamma}_{11}$ in (18) shows that to this end $\sqrt{h(x)}x^{a-1}$ needs to have a non-negative degree . This requires $p_1 \geq 2 - 2a$. The formula for $\boldsymbol{g}_1$ similarly shows that $h'(x)x^{a-1}$, $h(x)x^{a-2}$ and $h(x)x^{2a-2}$ all need to have a non-negative degree for small x. This requires $p_1 \geq 2 - a$.

b) In the non-centered case, in addition to (i) and (ii),

   (iii) (A1) and (A2): Theorem 11 requires $p_1 \geq \max\{1 - a, 1 - b\}$ for $b > 0$, or $1 - \min_j \eta_{0,j}$ for $b = 0$.

   (iv) "Controlled $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$ for $\boldsymbol{x}^{b}$": From the definition of $\boldsymbol{\Gamma}_{22}$ and $\boldsymbol{g}_2$ and by the same reasoning as above, $\sqrt{h(x)}x^{b-1}$, $h'(x)x^{b-1}$ and $h(x)x^{b-2}$ need to be non-negative powers of x, thus requiring $p_1 \geq \max\{2 - b, 2 - 2b\} = 2 - b$.

The choice of p_2, is only relevant for large data points. Our main consideration is then merely how well $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$ concentrate on their true population values (Theorem 13). From this perspective, our intuition is that p_2 should be chosen small so that the tails of the distributions of the entries of $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$ are well-behaved. Thus, we can choose $p_2 = 0$, in which case $h(x) = \min(x^{p_1}, c)$ is a truncated power.

5.5. Tuning Parameter Selection

By treating the unpenalized loss (i.e., $\lambda = 0$, $\boldsymbol{\gamma} = 0$) as a negative log-likelihood, we may use the extended Bayesian Information Criterion (eBIC) to choose the tuning parameter (Chen and Chen, 2008; Foygel and Drton, 2010). Consider the centered case as an example. Let $\hat{S}^\lambda \equiv \{(i,j) : \hat{\kappa}_{ij}^\lambda \neq 0, i < j\}$, where $\hat{\mathbf{K}}^\lambda$ be the estimate associated with tuning parameter λ. The eBIC is then

$$\text{eBIC}(\lambda) = -n\text{vec}(\hat{\mathbf{K}})^\top \boldsymbol{\Gamma}(\mathbf{x})\text{vec}(\hat{\mathbf{K}}) + 2n\boldsymbol{g}(\mathbf{x})^\top \text{vec}(\hat{\mathbf{K}}) + |\hat{S}^\lambda| \log n + 2\log \binom{p(p-1)/2}{|\hat{S}^\lambda|},$$

where $\hat{\mathbf{K}}$ can be either the original estimate associated with λ, or a refitted solution obtained by restricting the support to $\hat{S}^\lambda$.

We use the eBIC instead of the ordinary BIC (Bayesian Information Criterion) since the BIC tends to choose an overly complex model when the model space is large, as encountered in the high-dimensional setting. The extension in eBIC comes from the last term in the above display which can be motivated by a prior distribution under which the number of edges in the conditional independence graph is uniformly distributed; see also Żak-Szatkowska and Bogdan (2011) and Barber and Drton (2015).

6. Theory for Graphical Models

In our regularized generalized score matching framework, we introduced the amplifiers/ multipliers to address the inexistence problem. We also proposed using a general function $\boldsymbol{h}$ in place of $\boldsymbol{x}^2$ as a means to improve estimation accuracy. This section provides a theoretical analysis of these two aspects.

In Section 6.1, we present the theory for our regularized generalized score matching estimators for general pairwise interaction models before going into the details for the special cases of (truncated) GGMs. Next, we show that a specific choice of amplifiers/multipliers yields consistent estimation without the need for tuning. This point is important even in the case of Gaussian models on all of $\mathbb{R}^m$. Therefore, in Section 6.2 we digress from non-negative data and consider the original score matching of Hyvärinen (2005) for centered Gaussian distributions. Finally, in Section 6.3, we derive probabilistic results for $\hat{\boldsymbol{\Psi}}$ based on Theorem 13, justifying the benefits of using a general bounded $\boldsymbol{h}$ over $\boldsymbol{x}^2$ in the non-negative setting. As the most important models from the class of pairwise interaction power models over $\mathbb{R}_+^m$, we only treat truncated GGMs since they have the most tractable concentration bounds; this case also provides a comparison to Corollary 2 in Lin et al. (2016), which uses $\boldsymbol{x}^2$.

6.1. Theory for Pairwise Interaction Models

The graphical models we treat are parametrized by the interaction matrix $\mathbf{K}$ and the coefficients $\boldsymbol{\eta}$ on $(\boldsymbol{x}^b - \mathbf{1}_m)/b$. It is convenient to accommodate this setting with a matrix-valued parameter $\boldsymbol{\Psi} \in \mathbb{R}^{r_1 \times r_2}$ (in place of $\boldsymbol{\theta}$) and specify our regularized $\boldsymbol{h}$-score matching loss as

$$\hat{J}_{\boldsymbol{h},\lambda,\boldsymbol{\gamma}}(\boldsymbol{\Psi}) \equiv \operatorname*{argmin}_{\boldsymbol{\Psi} \in \mathbb{R}^{r_1 \times r_2}} \frac{1}{2}\text{vec}(\boldsymbol{\Psi})^\top \boldsymbol{\Gamma}_{\boldsymbol{\gamma}}(\mathbf{x})\text{vec}(\boldsymbol{\Psi}) - \boldsymbol{g}(\mathbf{x})^\top \text{vec}(\boldsymbol{\Psi}) + \lambda\|\boldsymbol{\Psi}\|_1. \tag{24}$$

In the non-centered case we thus take $\boldsymbol{\Psi} = [\mathbf{K}, \boldsymbol{\eta}]^\top \in \mathbb{R}^{m(m+1)\times m}$. In the centered case, $\boldsymbol{\Psi}$ is simply the $m \times m$ interaction matrix $\mathbf{K}$. Following related prior work such as Lin et al. (2016), for ease of proof we allow the matrix $\mathbf{K}$ to be nonsymmetric, which allows us to decouple optimization over the different columns of $\mathbf{K}$ or $\boldsymbol{\Psi}$, while in our implementations we ensure that $\mathbf{K}$ is symmetric.

Definition 12 *Let $\boldsymbol{\Gamma}_0 \equiv \mathbb{E}_0\boldsymbol{\Gamma}(\mathbf{x})$ and $\boldsymbol{g}_0 \equiv \mathbb{E}_0\boldsymbol{g}(\mathbf{x})$ be the population versions of $\boldsymbol{\Gamma}(\mathbf{x})$ and $\boldsymbol{g}(\mathbf{x})$ under the distribution given by a true parameter matrix $\boldsymbol{\Psi}_0$. The support of a matrix $\boldsymbol{\Psi}$ is $S(\boldsymbol{\Psi}) \equiv \{(i,j) : \psi_{ij} \neq 0\}$, and we let $S_0 = S(\boldsymbol{\Psi}_0)$. For a matrix $\boldsymbol{\Psi}_0$, we define $d_{\boldsymbol{\Psi}_0}$ to be the maximum number of non-zero entries in any column, and $c_{\boldsymbol{\Psi}_0} \equiv |||\boldsymbol{\Psi}_0|||_{\infty,\infty}$. Writing $\boldsymbol{\Gamma}_{0,AB}$ for the $A \times B$ submatrix of $\boldsymbol{\Gamma}_0$, we define*

$$c_{\boldsymbol{\Gamma}_0} \equiv |||(\boldsymbol{\Gamma}_{0,S_0S_0})^{-1}|||_{\infty,\infty}. \tag{25}$$

Finally, $\boldsymbol{\Gamma}_0$ satisfies the irrepresentability condition with incoherence parameter $\alpha \in (0,1]$ and edge set S_0 *if*

$$|||\boldsymbol{\Gamma}_{0,S_0^cS_0}(\boldsymbol{\Gamma}_{0,S_0S_0})^{-1}|||_{\infty,\infty} \leq (1 - \alpha). \tag{26}$$

Our analysis of the regularized generalized $\boldsymbol{h}$-score matching estimator builds on the following theorem taken from Lin et al. (2016, Theorem 1).

Theorem 13 *Suppose $\boldsymbol{\Gamma}_0$ has $\boldsymbol{\Gamma}_{0,S_0S_0}$ invertible and satisfies the irrepresentability condition (26) with incoherence parameter $\alpha \in (0,1]$. Assume that*

$$\|\boldsymbol{\Gamma}_{\boldsymbol{\gamma}}(\mathbf{x}) - \boldsymbol{\Gamma}_0\|_\infty < \epsilon_1, \qquad \|\boldsymbol{g}(\mathbf{x}) - \boldsymbol{g}_0\|_\infty < \epsilon_2, \tag{27}$$

with $d_{\boldsymbol{\Psi}_0}\epsilon_1 \le \alpha/(6c_{\boldsymbol{\Gamma}_0})$. If

$$\lambda > \frac{3(2-\alpha)}{\alpha}\max\{c_{\boldsymbol{\Psi}_0}\epsilon_1, \epsilon_2\},$$

then the following holds:

(a) The regularized generalized $\boldsymbol{h}$-score matching estimator $\hat{\boldsymbol{\Psi}}$ minimizing (24) is unique, with support $\hat{S} \equiv S(\hat{\boldsymbol{\Psi}}) \subseteq S_0$, and satisfies

$$\|\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0\|_\infty \le \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda.$$

(b) If

$$\min_{1\le j<k\le m} |\boldsymbol{\Psi}_{0,jk}| > \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda,$$

then $\hat{S} = S_0$ and $\operatorname{sign}(\hat{\boldsymbol{\Psi}}_{jk}) = \operatorname{sign}(\boldsymbol{\Psi}_{0,jk})$ for all $(j,k) \in S_0$.

This result is deterministic, and the improvement of our generalized estimator over the one in Lin et al. (2016) is in its probabilistic guarantees, as shown for truncated GGMs in Theorems 16 and 17 in Section 6.3. Before going into these examples, we state a general corollary.

Corollary 14 *Under the assumptions of Theorem 13, the matrix $\hat{\boldsymbol{\Psi}}$ minimizing (24) satisfies*

$$\begin{aligned}\|\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0\|_F &\le \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda\sqrt{|S_0|} \le \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda\sqrt{d_{\boldsymbol{\Psi}_0}m},\\ \|\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0\|_2 &\le \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda\min(\sqrt{|S_0|}, d_{\boldsymbol{\Psi}_0}).\end{aligned}$$

6.2. Revisiting Gaussian Score Matching

In this section we consider estimating the inverse covariance matrix $\mathbf{K}$ of a centered Gaussian distribution $\mathrm{N}(\mathbf{0},\mathbf{K})$, which of course has density proportional to (2) on all of $\mathbb{R}^m$. As shown, e.g., in Example 1 of Lin et al. (2016), the ℓ_1-regularized score matching loss then takes the form

$$\frac{1}{2}\operatorname{tr}(\mathbf{K}\mathbf{K}\mathbf{x}\mathbf{x}^\top) - \operatorname{tr}(\mathbf{K}) + \lambda\|\mathbf{K}\|_1, \tag{28}$$

which can be written as (14) with $\boldsymbol{\theta} = \operatorname{vec}(\mathbf{K})$, $\boldsymbol{\Gamma} = \operatorname{diag}(\mathbf{x}\mathbf{x}^\top,\dots,\mathbf{x}\mathbf{x}^\top)$ and $\boldsymbol{g} = \operatorname{vec}(\mathbf{I}_m)$. Thus, in general, the kernel of $\boldsymbol{\Gamma}$ need not be orthogonal to $\boldsymbol{g}$, and for λ small the loss can be unbounded below as discussed above. Hence, an amplifier/multiplier on the diagonals of $\boldsymbol{\Gamma}$ is needed. We have the following theorem on the estimator using the amplification.

Theorem 15 *Suppose the data matrix $\mathbf{x}$ holds n i.i.d. copies of $\boldsymbol{X} \sim \mathrm{N}(\mathbf{0},\mathbf{K}_0)$. Adopt the amplifying in Section 4 and redefine the loss in (28) as*

$$\frac{1}{2}\operatorname{tr}(\mathbf{K}\mathbf{K}\mathbf{G}) - \operatorname{tr}(\mathbf{K}) + \lambda\|\mathbf{K}\|_1, \quad \mathbf{G}_{jk} = (\mathbf{x}\mathbf{x}^\top)_{jk}\left(\mathbb{1}_{\{j\neq k\}} + (\delta-1)\mathbb{1}_{\{j=k\}}\right), \tag{29}$$

where $1 < \delta < 2 - \left(1 + 80\sqrt{\log m/n}\right)^{-1}$. Let $\hat{\mathbf{K}}$ be the resulting estimator. Let $c^* \equiv 12800\,(\max_j \boldsymbol{\Sigma}_{0,jj})^2$ and $c_1 = 4c_{\boldsymbol{\Gamma}_0}/\alpha$. If for some $\tau > 2$, the regularization parameter and the sample size satisfy

$$\begin{aligned}&\lambda > (2c_{\mathbf{K}_0}(2-\alpha)\sqrt{c^*(\tau\log m + \log 4)/n})/\alpha,\\ &n > \max(c^*c_1^2 d_{\mathbf{K}_0}^2, 2)(\tau\log m + \log 4),\end{aligned}$$

then $\|\hat{\mathbf{K}} - \mathbf{K}_0\|_\infty \le \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda$ with probability $1 - m^{2-\tau}$.

In Corollary 1 of Lin et al. (2016) the same results were shown with $c^* \equiv 3200\,(\max_j \boldsymbol{\Sigma}_{0,jj})^2$ when a unique minimizer exists, but the existence was not guaranteed.

6.3. Generalized Score Matching for Truncated GGMs

Next, we provide theory for the regularized generalized $\boldsymbol{h}$-score matching estimator $\hat{\boldsymbol{\Psi}}$ in the special case of truncated GGMs. Again, assume a common h for all components in $\boldsymbol{h}$.

Theorem 16 *Suppose the data matrix $\mathbf{x}$ holds n i.i.d. copies of $\boldsymbol{X} \sim \mathrm{TN}(\mathbf{0},\mathbf{K}_0)$, where the mean parameter is known to be zero. Assume that $\boldsymbol{h} \in \mathcal{H}_{1,1}$ and that $0 \le h \le M$, $0 \le h' \le M'$ a.s. for constants M, M', and choose $\boldsymbol{\gamma} = (\delta-1)\operatorname{diag}(\boldsymbol{\Gamma})$ with*

$$1 < \delta < C(n,m) \equiv 2 - \left(1 + 4e\max\{6\log m/n, \sqrt{6\log m/n}\}\right)^{-1}.$$

Suppose that the $\boldsymbol{\Gamma}_{0,S_0S_0}$ block of $\boldsymbol{\Gamma}_0$ is invertible and $\boldsymbol{\Gamma}_0$ satisfies the irrepresentability condition (26) with $\alpha \in (0,1]$ and true edge set S_0. Define $c_{\boldsymbol{X}} \equiv 2\max_j\left(2\sqrt{(\mathbf{K}_0^{-1})_{jj}} + \sqrt{e}\,\mathbb{E}_0 X_j\right)$. If for $\tau > 3$ the sample size and the regularization parameter satisfy

$$n > \mathcal{O}\left(\tau\log m\max\left\{\frac{M^2c_{\boldsymbol{\Gamma}_0}^2c_{\boldsymbol{X}}^4d_{\mathbf{K}_0}^2}{\alpha^2}, \frac{Mc_{\boldsymbol{\Gamma}_0}c_{\boldsymbol{X}}^2d_{\mathbf{K}_0}}{\alpha}\right\}\right), \tag{30}$$

$$\lambda > \mathcal{O}\left[(Mc_{\mathbf{K}_0}c_{\boldsymbol{X}}^2 + M'c_{\boldsymbol{X}} + M)\left(\sqrt{\frac{\tau\log m}{n}} + \frac{\tau\log m}{n}\right)\right], \tag{31}$$

then the following statements hold with probability $1 - m^{3-\tau}$:

(a) The regularized generalized $\boldsymbol{h}$-score matching estimator $\hat{\mathbf{K}}$ that minimizes (24) is unique, has its support included in the true support, $\hat{S} \equiv S(\hat{\mathbf{K}}) \subseteq S_0$, and satisfies

$$\|\hat{\mathbf{K}} - \mathbf{K}_0\|_\infty \le \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda,$$

$$|||\hat{\mathbf{K}} - \mathbf{K}_0|||_F \le \frac{c_{\mathbf{\Gamma}_0}}{2-\alpha}\lambda\sqrt{|S_0|},$$
$$|||\hat{\mathbf{K}} - \mathbf{K}_0|||_2 \le \frac{c_{\mathbf{\Gamma}_0}}{2-\alpha}\lambda \min(\sqrt{|S_0|}, d_{\mathbf{K}_0}),$$

where $c_{\mathbf{\Gamma}_0}$ is defined in (25).

(b) Moreover, if

$$\min_{j,k:(j,k)\in S_0} |\kappa_{0,jk}| > \frac{c_{\mathbf{\Gamma}_0}}{2-\alpha}\lambda,$$

then $\hat{S} = S_0$ and $\mathrm{sign}(\hat{\kappa}_{jk}) = \mathrm{sign}(\kappa_{0,jk})$ for all $(j,k) \in S_0$.

The theorem is proved in Appendix A.4, where details on the dependencies on constants are provided. A key ingredient of the proof is a tail bound on $\|\mathbf{\Gamma}_\gamma - \mathbf{\Gamma}_0\|_\infty$, which features products of the $X_j^{(i)}$'s. In Lin et al. (2016), the products are up to fourth order. Using bounded $\boldsymbol{h}$, our products automatically calibrates to a quadratic polynomial when the observed values are large, and resort to higher moments only when they are small. This leads to improved bounds and convergence rates, underscored in the new requirement on the sample size n, which should be compared to $n \ge \mathcal{O}(d^2_{\mathbf{K}_0}(\log m^\tau)^8)$ in Lin et al. (2016).

For the non-centered case, by definition, $c_{\mathbf{\Psi}_0} \equiv |||\mathbf{\Psi_0}^\top|||_{\infty,\infty} \le c_{\mathbf{K}_0} + \|\boldsymbol{\eta}_0\|_\infty$, $d_{\mathbf{\Psi}_0} \le d_{\mathbf{K}_0}+1$. The proof given for Theorem 16 goes through again here, and we have the following consistency results.

Theorem 17 *Suppose the data matrix holds n i.i.d. copies of $\boldsymbol{X} \sim \mathrm{TN}(\boldsymbol{\mu}_0, \mathbf{K}_0)$. Assume that $\boldsymbol{h} \in \mathcal{H}_{1,1}$ and that $0 \le h \le M$, $0 \le h' \le M'$ a.s. for constants M, M'. Let $\boldsymbol{\gamma}$ be a vector of amplifiers that are non-zero only for the diagonal entries of the matrices $\mathbf{\Gamma}_{11,j}$, amplifying those by $(\delta-1)\mathrm{diag}(\mathbf{\Gamma}_{11,j})$ with*

$$1 < \delta < C(n,m) \equiv 2 - \left(1 + 4e\max\{6\log m/n, \sqrt{6\log m/n}\}\right)^{-1}.$$

Suppose further that $\mathbf{\Gamma}_{0,S_0S_0}$ is invertible and satisfies the irrepresentability condition (26) with $\alpha \in (0,1]$. Define $c_{\boldsymbol{X}} \equiv 2\max_j\left(2\sqrt{(\mathbf{K}_0^{-1})_{jj}} + \sqrt{e}\,\mathbb{E}_0 X_j\right)$. Suppose for $\tau > 3$ the sample size and the regularization parameter satisfy

$$n > \mathcal{O}\left(\tau\log m \max\left\{\frac{M^2 c^2_{\mathbf{\Gamma}_0,\mathbf{\Psi}_0} c^4_{\boldsymbol{X}} d^2_{\mathbf{\Psi}_0}}{\alpha^2}, \frac{M c_{\mathbf{\Gamma}_0,\mathbf{\Psi}_0} c^2_{\boldsymbol{X}} d_{\mathbf{\Psi}_0}}{\alpha}\right\}\right), \tag{32}$$

$$\lambda > \mathcal{O}\left[(M c_{\mathbf{\Psi}_0} c^2_{\boldsymbol{X}} + M' c_{\boldsymbol{X}} + M)\left(\sqrt{\frac{\tau\log m}{n}} + \frac{\tau\log m}{n}\right)\right], \tag{33}$$

where $c_{\mathbf{\Gamma}_0,\mathbf{\Psi}_0}$ is $c_{\mathbf{\Gamma}_0}$ as in (25) but with notation $\mathbf{\Psi}_0$ to differentiate it from the centered case. Then the following statements hold with probability $1 - m^{3-\tau}$:

(a) The regularized generalized $\boldsymbol{h}$-score matching estimator $\hat{\mathbf{\Psi}}$ that minimizes (24) is unique, has its support included in the true support, $\hat{S} \equiv S(\hat{\mathbf{\Psi}}) \subseteq S_0$, and satisfies

$$\|\hat{\mathbf{K}} - \mathbf{K}_0\|_\infty \le \frac{c_{\mathbf{\Gamma}_0,\mathbf{\Psi}_0}}{2-\alpha}\lambda, \qquad \|\hat{\boldsymbol{\eta}} - \boldsymbol{\eta}_0\|_\infty \le \frac{c_{\mathbf{\Gamma}_0,\mathbf{\Psi}_0}}{2-\alpha}\lambda,$$

$$|||\hat{\mathbf{K}} - \mathbf{K}_0|||_F \le \frac{c_{\mathbf{\Gamma}_0,\mathbf{\Psi}_0}}{2-\alpha}\lambda\sqrt{|S_0|}, \qquad \|\hat{\boldsymbol{\eta}} - \boldsymbol{\eta}_0\|_F \le \frac{c_{\mathbf{\Gamma}_0,\mathbf{\Psi}_0}}{2-\alpha}\lambda\sqrt{|S_0|},$$
$$|||\hat{\mathbf{K}} - \mathbf{K}_0|||_2 \le \frac{c_{\mathbf{\Gamma}_0,\mathbf{\Psi}_0}}{2-\alpha}\lambda \min\left(\sqrt{|S_0|}, d_{\mathbf{\Psi}_0}\right), \quad \|\hat{\boldsymbol{\eta}} - \boldsymbol{\eta}_0\|_2 \le \frac{c_{\mathbf{\Gamma}_0,\mathbf{\Psi}_0}}{2-\alpha}\lambda \min\left(\sqrt{|S_0|}, d_{\mathbf{\Psi}_0}\right).$$

(b) Moreover, if

$$\min_{j,k:(j,k)\in S_0} |\kappa_{0,jk}| > \frac{c_{\mathbf{\Gamma}_0}}{2-\alpha}\lambda \qquad \textit{and} \qquad \min_{j:(m+1,j)\in S_0} |\eta_{0,j}| > \frac{c_{\mathbf{\Gamma}_0}}{2-\alpha}\lambda,$$

then $\hat{S} = S_0$ and $\mathrm{sign}(\hat{\kappa}_{jk}) = \mathrm{sign}(\kappa_{0,jk})$ for all $(j,k) \in S_0$ and $\mathrm{sign}(\hat{\eta}_j) = \mathrm{sign}(\eta_{0j})$ for $(m+1,j) \in S_0$.

Remark 18 The quantity $c_{\boldsymbol{X}}$ in Theorem 17 depends on $\mathbb{E}_0 X_j$, which in turn depends on the structure of both $\boldsymbol{\mu}_0$ and $\mathbf{K}_0$. If $\mu_{0,j}$ is large compared to $(\mathbf{K}_0)^{-1}_{jj}$, then $c_{\boldsymbol{X}}$ seems to scale as $\boldsymbol{\mu}_0$, which negatively impacts the guarantees stated in Theorem 17. However, as in the one-dimensional case for estimation of μ_0 (Example 3.1), our estimator should automatically adapt to the large mean parameter. This suggests that it might be possible to improve our analysis involving $c_{\boldsymbol{X}}$.

7. Numerical Experiments

In this section, we compare the performance of our estimator with different choices of $\boldsymbol{h}$ to the existing approaches for pairwise interaction power models. In our simulation experiments, we consider $m = 100$ variables and $n = 80$ and $n = 1000$ samples, corresponding to high- and low-dimensional settings. We also tried intermediate sample sizes between these two extremes, but found no interesting result worth reporting. For $n = 80$, amplification is necessary. Except in Section 7.2.2, the amplifier is set based on Theorem 16 to $\delta = C(n,m) = 1.8647$ for truncated GGMs. The same amplifier is also used for settings with other a and b. For $n = 1000$, we consider $\delta = 1$, i.e., no amplification, and $\delta = C(n,m) = 1.6438$ (again, based on Theorem 16). Throughout, we assume a common univariate h for all components in $\boldsymbol{h}$.

7.1. Structure of K

The underlying interaction matrices are selected as follows: Proceeding as in Section 4.2 of Lin et al. (2016), the graph is chosen to have 10 disconnected subgraphs, each containing $m/10$ nodes. Thus, $\mathbf{K}_0$ is block-diagonal. In each block, each lower-triangular element is set to 0 with probability $1-\pi$ for some $\pi \in (0,1)$, and is otherwise drawn from Uniform[0.5, 1]. The upper triangular elements are determined by symmetry. The diagonal elements of $\mathbf{K}_0$ are chosen as a common positive value such that the minimum eigenvalue of $\mathbf{K}_0$ is 0.1.

We generate 5 different true precision matrices $\mathbf{K}_0$, and run 10 trials with each of these precision matrices. For $n = 1000$, we choose $\pi = 0.8$, which is in accordance with Lin et al. (2016). For $n = 80$, we set $\pi = 0.2$. This way $n/(d^2_{\mathbf{K}_0}\log m)$ is roughly constant; recall Theorems 16 and 17 for truncated GGMs.

In Appendix C, we report results on *Erdős-Rényi graphs*, which lead to similar conclusions.

7.2. Truncated GGMs

Given our focus on truncated GGMs and their relevance in graphical modeling applications, we start with experiments for these models.

7.2.1. Choice of $\boldsymbol{h}$

Our estimator requires choosing a function $\boldsymbol{h} : \mathbb{R}_+^m \to \mathbb{R}_+^m$. For simplicity, we will always specify $\boldsymbol{h}(\boldsymbol{x}) = (h(x_1), \ldots, h(x_m))$ for a single non-decreasing univariate function $h : \mathbb{R}_+ \to \mathbb{R}_+$, i.e. all coordinates share the same h function.

As previously explained, $\boldsymbol{h} \in \mathcal{H}_{a,b}$ is a sufficient condition for assumptions (A1)-(A2), as well as (C1)-(C2) in the case of unregularized estimators. Only in the proofs of our theoretical guarantees in Section 6 for truncated GGMs, did we require h to be bounded and to have bounded derivatives. As motivated by the discussion in Section 5.4.2, we consider truncated and untruncated powers, $\min(x, c)$ and x (since $2 - a = 2 - b = 1$); we evaluate this choice by contrasting them with powers $x^{1.5}$ and x^2. We also explore functions like $\log(1 + x)$ that seem natural and are linear near 0. In particular, we make a further comparison to functions linear near 0 with a finite asymptote as $x \nearrow +\infty$ but differentiable everywhere: MCP- (Fan and Li, 2001) and SCAD-like (Zhang, 2010) functions defined below. The results we report are based on selections of best performing choices of h.

$$\mathrm{SCAD}(x; \lambda, \gamma) \equiv \begin{cases} \lambda x & \text{if } 0 \le x \le \lambda, \\ \frac{2\gamma\lambda x - x^2 - \lambda^2}{2(\gamma - 1)} & \text{if } \lambda < x < \gamma\lambda, \\ \frac{\lambda^2(\gamma+1)}{2} & \text{if } x \ge \gamma\lambda; \end{cases} \quad \mathrm{MCP}(x; \lambda, \gamma) \equiv \begin{cases} \lambda x - \frac{x^2}{2\gamma} & \text{if } 0 \le x \le \gamma\lambda, \\ \frac{1}{2}\gamma\lambda^2 & \text{if } x > \gamma\lambda. \end{cases}$$

We do not observe any clear relationship between features such as convexity, differentiability or the slope of h at 0, and performance of the estimator. Nonetheless, for many choices of rather simple functions h, our estimator provides a significant improvement over existing methods. In particular, most h functions that behave linearly for small x, namely $\log(1 + x)$ and x and their truncations, and additionally MCP and SCAD, always perform better than $x^{1.5}$ and x^2. This agrees with our discussion in Section 5.4.2, where $2 - a = 1$ is a reasonable choice of the power for small x; also see Section 7.3. However, we conclude that there is no real gain from making the function smoother by using MCP or SCAD.

Truncated Centered GGMs: For data from a truncated centered Gaussian distribution, we compare our generalized score matching estimator with various choices of h, to *SpaCE JAM* (SJ, Voorman et al., 2014), which estimates graphs using additive models for conditional means, a pseudo-likelihood method *SPACE* (Peng et al., 2009) in the reformulation of Khare et al. (2015), *graphical lasso* (GLASSO, Yuan and Lin, 2007; Friedman et al., 2008), the *neighborhood selection* estimator (NS) of Meinshausen and Bühlmann (2006), and *nonparanormal SKEPTIC* (Liu et al., 2012) with Kendall's τ. Recall that the choice of $h(x) = x^2$ corresponds to the estimator from Lin et al. (2016).

The ROC (*receiver operating characteristic*) curves for different estimators are shown in Figure 3 on Page 26. Each plotted curve corresponds to the average of 50 ROC curves, where the averaging is based on the vertical averaging from Algorithm 3 in Fawcett (2006), and is mean AUC-preserving. The x and y axes of each ROC curve represent the false positive and true positive rates at varying levels of penalty parameter λ, defined as

Centered, $n = 80$, multiplier 1.8647

$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
∞	0.694	0.033	∞	0.702	0.031
2	0.694	0.033	3	0.702	0.031
1	0.692	0.033	2	0.698	0.033
0.5	0.664	0.038	1	0.686	0.030
$\mathrm{MCP}(1, c)$			$\mathrm{SCAD}(1, c)$		
c	Mean	sd	c	Mean	sd
10	0.701	0.032	10	0.702	0.031
5	0.700	0.032	5	0.701	0.032
1	0.672	0.036	2	0.696	0.033
$x^{1.5}$: (0.683, 0.030)			x^2: (0.630, 0.029)		
GLASSO (0.600,0.032)			SPACE: (0.587, 0.031)		
NS: (0.587,0.031)			SJ: (0.540,0.036)		

Centered, $n = 1000$, multiplier 1

$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
2	0.826	0.015	2	0.820	0.014
∞	0.826	0.015	3	0.820	0.015
1	0.824	0.014	∞	0.819	0.015
0.5	0.804	0.015	1	0.817	0.014
$\mathrm{MCP}(1, c)$			$\mathrm{SCAD}(1, c)$		
c	Mean	sd	c	Mean	sd
5	0.824	0.015	2	0.823	0.014
10	0.822	0.015	5	0.822	0.015
1	0.810	0.015	10	0.821	0.015
$x^{1.5}$: (0.782,0.014)			x^2: (0.732,0.015)		
SPACE: (0.780,0.015)			NS: (0.779,0.015)		
GLASSO (0.764,0.014)			SJ: (0.703,0.015)		

Centered, $n = 1000$, multiplier 1.6438

$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
∞	0.857	0.011	3	0.855	0.011
2	0.857	0.011	∞	0.855	0.011
1	0.855	0.011	2	0.854	0.011
0.5	0.833	0.012	1	0.847	0.011
$\mathrm{MCP}(1, c)$			$\mathrm{SCAD}(1, c)$		
c	Mean	sd	c	Mean	sd
5	0.857	0.011	5	0.856	0.011
10	0.856	0.011	10	0.855	0.011
1	0.840	0.012	2	0.855	0.011
$x^{1.5}$: (0.812,0.011)			x^2: (0.736,0.011)		
SPACE: (0.780,0.015)			NS: (0.779,0.015)		
GLASSO (0.764,0.014)			SJ: (0.703,0.015)		

Table 1: Mean and standard deviation of areas under the ROC curves (AUC) using different estimators in the centered setting, with $n = 80$ and multiplier 1.8647, or $n = 1000$ and multiplier 1 and 1.6438. Methods include our estimator with different choices of h, GLASSO, SPACE, neighborhood selection (NS), and Space JAM (SJ).

$$\mathrm{FPR} \equiv \frac{|\hat{S}_{\mathrm{off}} \backslash S_{0,\mathrm{off}}|}{m(m-1) - |S_{0,\mathrm{off}}|} \qquad \text{and} \qquad \mathrm{TPR} \equiv \frac{|\hat{S}_{\mathrm{off}} \cap S_{0,\mathrm{off}}|}{|S_{0,\mathrm{off}}|},$$

where $S_{0,\mathrm{off}} \equiv \{(i,j) : i \neq j \wedge \kappa_{0,ij} \neq 0\}$, and $\hat{S}_{\mathrm{off}} \equiv \{(i,j) : i \neq j \wedge \hat{\kappa}_{ij} \neq 0\}$.

To reduce clutter, we only report the results for the top performing competing methods. In particular, results for nonparanormal SKEPTIC are omitted, as the method always performs the worst in our experiments. The corresponding means and standard deviations of AUCs (*areas under the curves*) over 50 curves are given in Table 1.

Looking at the mean AUCs, with the standard deviations in mind, all choices of h considered here perform better than $h(x) = x^2$ from Hyvärinen (2007) and Lin et al. (2016) and the competing methods. The results for $n = 1000$ in Table 1 also show that the multiplier does help improve the AUCs, a matter to be discussed in Section 7.2.2.

Truncated Non-Centered GGMs: We generate data from a truncated non-centered Gaussian distribution with both parameters $\boldsymbol{\mu}$ and $\mathbf{K}$ unknown. In each trial, we form the true $\mathbf{K}_0$ as in Section 7.1, and generate each component of $\boldsymbol{\mu}_0$ independently from the normal distribution with mean 0 and standard deviation 0.5.

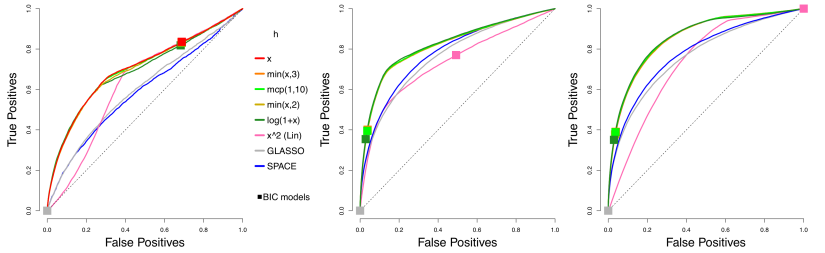
We compare the performance of our *profiled* estimator based on (19), with different h functions, but with no penalty on $\boldsymbol{\eta} \equiv \mathbf{K}\boldsymbol{\mu}$, to SPACE, SpaCE JAM (SJ), GLASSO, and neighborhood selection (NS). As before, we consider 50 trials. Representative ROC curves are plotted in Figure 4, and the corresponding AUCs are summarized in Table 2.

Non-centered profiled, $n = 80$, multiplier 1.8647					
$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
∞	0.632	0.032	∞	0.634	0.032
2	0.632	0.032	3	0.634	0.032
1	0.631	0.032	2	0.632	0.032
0.5	0.619	0.033	1	0.628	0.032
MCP$(1, c)$			SCAD$(1, c)$		
c	Mean	sd	c	Mean	sd
10	0.634	0.032	5	0.634	0.032
5	0.634	0.032	10	0.634	0.032
1	0.622	0.032	2	0.634	0.032
$x^{1.5}$: (0.623,0.031)			x^2: (0.607,0.030)		
GLASSO: (0.614,0.029)			NS: (0.604,0.028)		
SPACE: (0.602,0.029)			SJ: (0.561,0.036)		

Non-centered profiled, $n = 1000$, multiplier 1					
$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
∞	0.783	0.020	2	0.779	0.020
2	0.783	0.020	∞	0.779	0.020
1	0.782	0.020	3	0.779	0.020
0.5	0.767	0.021	0.5	0.758	0.020
MCP$(1, c)$			SCAD$(1, c)$		
c	Mean	sd	c	Mean	sd
5	0.782	0.020	2	0.780	0.020
10	0.780	0.020	5	0.780	0.020
1	0.771	0.021	10	0.779	0.020
$x^{1.5}$: (0.751,0.019)			x^2: (0.713,0.018)		
SPACE: (0.786,0.020)			NS: (0.785,0.02)		
GLASSO (0.770,0.019)			SJ: (0.720,0.019)		

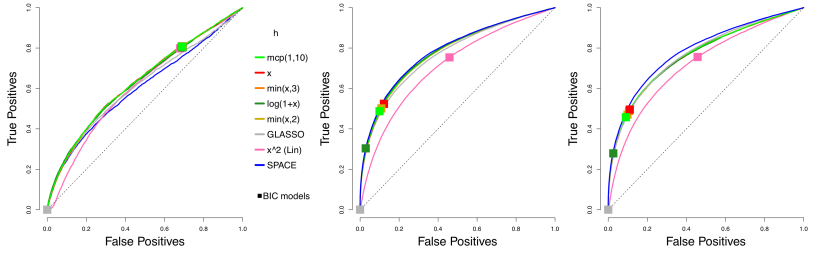
Non-centered profiled, $n = 1000$, multiplier 1.6438					
$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
∞	0.764	0.018	∞	0.766	0.019
2	0.764	0.018	3	0.765	0.019
1	0.762	0.018	2	0.764	0.018
0.5	0.738	0.018	1	0.753	0.018
MCP$(1, c)$			SCAD$(1, c)$		
c	Mean	sd	c	Mean	sd
10	0.766	0.019	10	0.766	0.019
5	0.766	0.019	5	0.766	0.019
1	0.745	0.018	2	0.763	0.018
$x^{1.5}$: (0.748,0.018)			x^2: (0.718,0.017)		
SPACE: (0.786,0.020)			NS: (0.785,0.020)		
GLASSO (0.770,0.019)			SJ: (0.720,0.019)		

Table 2: Mean and standard deviation of AUC using different profiled estimators in the non-centered setting, with $n = 80$ and multiplier 1.8647, or $n = 1000$ and multipliers 1 and 1.6438. Methods include our estimator with different choices of h, GLASSO, SPACE, neighborhood selection (NS), and Space JAM (SJ).



(a) $n = 80$, mult = 1.8647 (b) $n = 1000$, mult = 1 (c) $n = 1000$, mult = 1.6438

Figure 3: Average ROC curves of our centered estimator with various choices of h, compared to SPACE and GLASSO, for the truncated centered GGM case; $m = 100$ variables and $n = 80$ or 1000 samples are considered. Squares indicate average true positive rate (TPR) and false positive rate (FPR) of models picked by eBIC with refitting for the estimator in the same color.



(a) $n = 80$, mult = 1.8647 (b) $n = 1000$, mult = 1 (c) $n = 1000$, mult = 1.6438

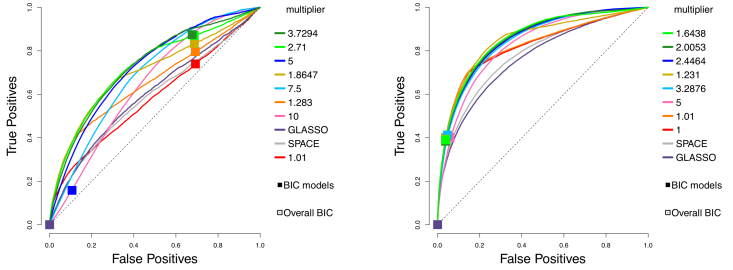
Figure 4: Average ROC curves of our non-centered profiled estimator with various choices of h, compared to SPACE and GLASSO, for the truncated non-centered GGM case; $m = 100$ variables and $n = 80$ or 1000 samples are considered. Squares indicate average true positive rate (TPR) and false positive rate (FPR) of models picked by eBIC with refitting for the estimator in the same color.

Even without tuning the extra penalty parameter on $\boldsymbol{\eta} \equiv \mathbf{K}\boldsymbol{\mu}$, our profiled estimator beats the competing methods by a large margin when $n = 80$. With multipliers 1 and $n = 1000$, our estimators still do better than Space JAM and GLASSO, and have performance comparable to other competing methods. It might appear that the performance of our estimators deteriorate with a multiplier larger than 1; however, as we will see, there can be significant improvement in AUCs if we tune an additional parameter for the multiplier. As in the centered case, the leading h functions in each category perform similarly, and the exact choice is not crucial. Subsequently, we will simply use $h(x) = \min(x, 3)$.

7.2.2. Choice of multiplier

Truncated Centered GGMs: In Figure 5, the ROC curves for GLASSO, SPACE, and our estimator with $h(x) = \min(x, 3)$, but with different levels of amplification, via different choices of multipliers δ, are compared for the centered case of Section 7.2.1.

While Theorem 16 guarantees consistency only for $\delta < C(n, m)$, we observe that there can be a gain from going beyond the *upper-bound multiplier* $C(n, m)$, which is 1.8647 for $n = 80$ and 1.6438 for $n = 1000$ (when $n = 1000$, $C(n, m)$ turns out to be the best-performing multiplier). However, the effect deteriorates fast as the multiplier grows larger. The figure suggests that while some additional gains are possible by tuning over the choice of multiplier, the *upper-bound multiplier* is a good default.



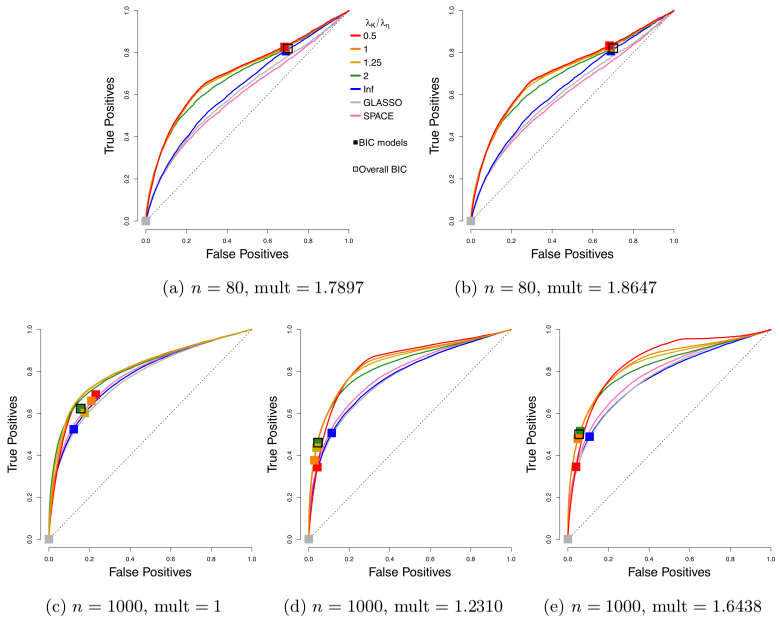
(a) $n = 80$ (b) $n = 1000$

Figure 5: Performance of $\min(x, 3)$ for truncated centered GGMs using different multipliers, compared to GLASSO and SPACE, in the centered setting, $n = 80$ or 1000.

Truncated Non-Centered GGMs : In Figure 6, we consider the non-centered case of Section 7.2.1, and use the non-profiled estimator; that is, the non-centered estimator with ℓ_1 penalty on both $\mathbf{K}$ and $\boldsymbol{\eta} \equiv \mathbf{K}\boldsymbol{\mu}$. The ROC curves are compared to competing methods GLASSO and SPACE. For the choice of amplification in our estimator, we consider the upper-bound multiplier $C(n, m)$ from Theorem 17 as the default. We refer to this as *high* amplification. We also consider lower amplification, with $\delta = 2 - (1 + 24e \log m/n)^{-1}$, referred to as *medium*. For $n = 1000$, we also consider a *low* multiplier 1, which corresponds to no amplification. We compare these possible defaults to a finer grid of multipliers of which we show some representatives in the plots.

We see that among our defaults, the upper-bound choice $C(n, m)$ performs best. Some additional gains are possible by tuning the multiplier over a grid of values containing this choice. Moreover, we see that it can be beneficial to tune over both $\lambda_{\mathbf{K}}$ and $\lambda_{\mathbf{K}}/\lambda_{\boldsymbol{\eta}}$.

We remark that while for each run, the best model picked by BIC falls on the ROC curve, a few squares are off the curve in Figure 6 (c). This is because these squares correspond to the average of the true and false positive rates of the chosen BIC models over 50 runs, potentially due to multimodality of the distribution of the models. Nonetheless, in all cases, the average of the models picked by BIC tuned over both $\lambda_{\mathbf{K}}$ and $\lambda_{\mathbf{K}}/\lambda_{\boldsymbol{\eta}}$ looks reasonable.



(a) $n = 80$, mult $= 1.7897$ (b) $n = 80$, mult $= 1.8647$

(c) $n = 1000$, mult $= 1$ (d) $n = 1000$, mult $= 1.2310$ (e) $n = 1000$, mult $= 1.6438$

Figure 6: Performance of the non-centered estimator with $h(x) = \min(x, 3)$. Each curve corresponds to a different choice of $\lambda_{\mathbf{K}}/\lambda_{\boldsymbol{\eta}}$. Squares indicate models picked by eBIC with refit. The square with black outline has the highest eBIC among all models (combinations of $\lambda_{\mathbf{K}}$, $\lambda_{\boldsymbol{\eta}}$). Multipliers correspond to medium or high for $n = 80$, and low, medium or high for $n = 1000$, respectively.

7.3. Other a/b Models

We now turn to the non-Gaussian ($a \neq 1$ or $b \neq 1$) setting. Based on the observations in Section 5.4.2, we focus on functions of type $\min(x^p, c)$ for some power $p > 0$ and truncation point $c > 0$. For simplicity, for the non-centered models we use the profiled estimator (19) (i.e., $\lambda_{\boldsymbol{\eta}} = 0$) and use the multiplier $C(n, m)$ in Theorem 16 for truncated GGMs as a guidance. We note that tuning over the $\lambda_{\boldsymbol{\eta}}$ parameter and the multiplier can potentially give a significant improvement as seen in Section 7.2.

These simulations suggest that among the class of functions of the form $\min(x^p, c)$, x^{2-a} or $\min(x^{2-a}, c)$ with a moderately large c can be used as the default choice of $h(x)$. This agrees with our findings in Section 7.2.1. We note that bounded h functions were only used in the proof for truncated GGMs, and picking a moderately large truncation point can correspond to having an untruncated power.

7.3.1. Exponential Setting

For the exponential models, $a = b = 1/2$. Since $a = b$, for both centered and non-centered settings, based on the principle in Section 5.4.2, choosing $h(x) = \min(x^{3/2}, c)$ satisfies (A1) and (A2) and also ensures that entries in $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$ are bounded (for small x), while choosing $h(x) = \min(\sqrt{x}, c)$ only guarantees (A1) and (A2).

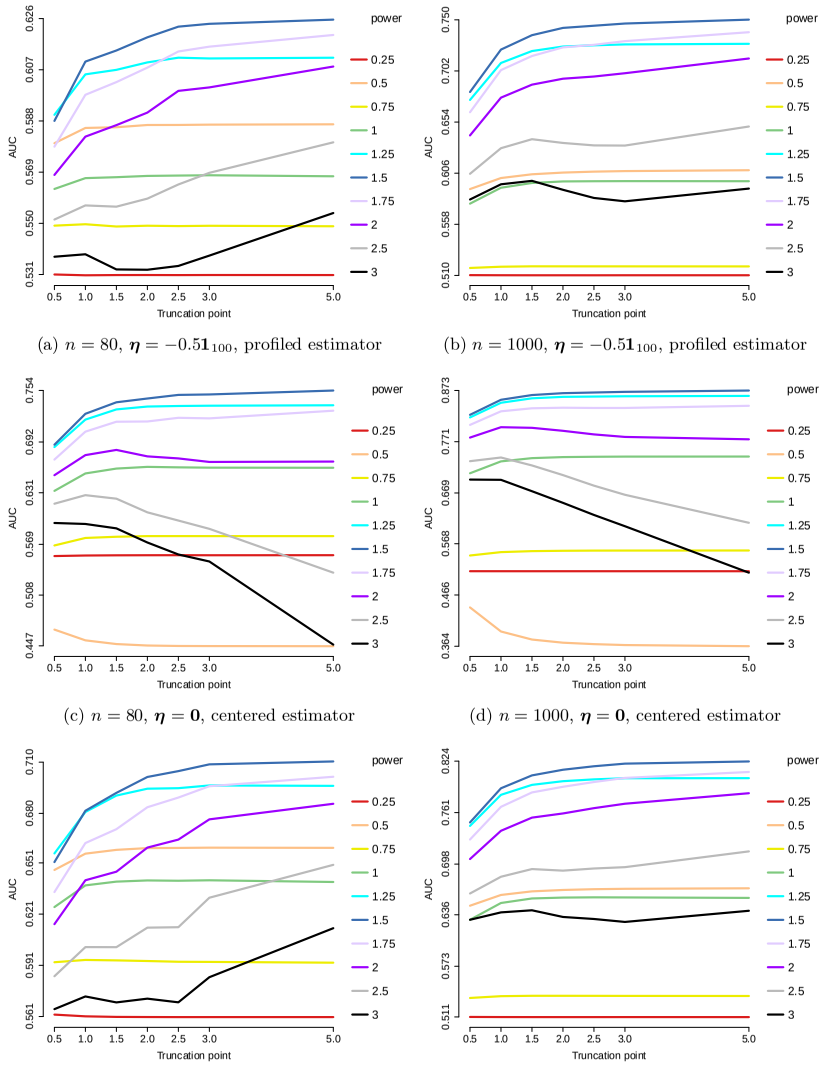
In Figure 7, we present the AUCs for the ROC curves of edge recovery with different choices of $h(x) = \min(x^{\text{pow}}, c)$. As before, we set $n = 80$ or 1000 and $m = 100$, but we use an $\boldsymbol{\eta}_0$ with each component uniformly equal to -0.5, 0 or 0.5; for $\boldsymbol{\eta}_0 \equiv \mathbf{0}$, we assume this information is known and use the centered estimator. The results suggest that $\text{pow} = 3/2 = 2 - a$ is the best choice of power. For this optimal choice, the performance improves with larger c, so x^{2-a} gives the best results. For sub-optimal powers, including truncation gives better results.

7.3.2. Gamma Setting

The centered gamma models reduce to the centered exponential models. Thus, in this section, we only consider the non-centered settings, with $a = 1/2$, $b = 0$. From Section 5.4.2, we have the following choices:

- $\min(x^2, c)$ both satisfies (A1)–(A2) and ensures $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$ are bounded;
- $\min(x^{\max\{3/2, 1-\min_j \eta_{0,j}\}}, c)$ ensures (A1)–(A2) and bounds $\boldsymbol{\Gamma}_{11}$ and $\boldsymbol{g}_1$; by default without prior information on $\boldsymbol{\eta}_0$ this is $\min(x^2, c)$;
- $\min(x^{3/2}, c)$ satisfies both conditions on the interaction part only ($\boldsymbol{x}^a$), but does not guarantee (A1)–(A2);
- $\min(x^{1/2}, c)$ satisfies the sufficient conditions for (A1)–(A2) on the interaction only.

The results are shown in Figure 8, where we consider $n = 80$, 1000, and $\boldsymbol{\eta} = \pm 0.5\mathbf{1}_{100}$. They suggest that $\text{pow} = 2 - a = 1.5$ works consistently well, although slightly outperformed by 1 and 1.25 in one case. As in the exponential case, with the optimal power it is beneficial to choose a large truncation point, or work with an untruncated power $x^{1.5}$. We conclude that the performance is likely only dependent on the $(2 - a)$ power requirement for the ${\boldsymbol{x}^a}^\top \mathbf{K} \boldsymbol{x}^a$ part or $2 - \min_j \eta_{0,j}$; simulations in the next section rule out the possibility of the latter.



(a) $n = 80$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator

(b) $n = 1000$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator

(c) $n = 80$, $\boldsymbol{\eta} = \mathbf{0}$, centered estimator

(d) $n = 1000$, $\boldsymbol{\eta} = \mathbf{0}$, centered estimator

(e) $n = 80$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator

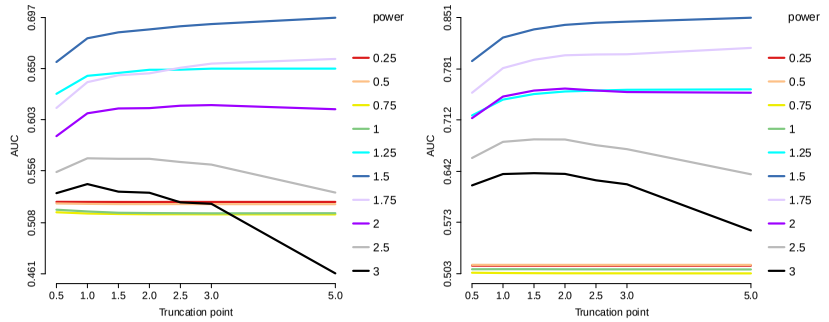
(f) $n = 1000$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator

Figure 7: AUCs for edge recovery using generalized score matching for the exponential models. Each curve represents a different choice of power p in $h(x) = \min(x^p, c)$, and the x axis marks the truncation point c. Colors are sorted by p.

7.3.3. Other Choices of a and b

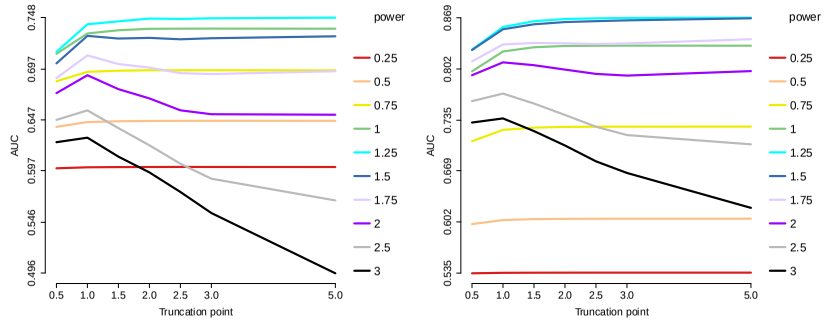
In this section, we consider other choices of a and b. Specifically, $a = 3/2$ and $b = 1/2$ or 0. These combinations are chosen to confirm, in a more extreme setting, that the performance is mainly determined by requirements on the power based on a, which correspond to choosing a power of $1-a$ or $2-a$, but not those on b (or on $\boldsymbol{\eta}$ when $b = 0$) that correspond to $1-b$ and $2-b$. The relationship between these two settings is analogous to that between the exponential and gamma models (same a, b nonzero/zero).

The results are shown in Figures 9 and 10, and indeed confirm that $x^{2-a} = x^{0.5}$ consistently gives the optimal results, even though $\boldsymbol{\eta}^\top \boldsymbol{x}^b$ is in favor of $x^{2-b} = x^{1.5}$ for $b = 0.5$, and $\boldsymbol{\eta}^\top \log(\boldsymbol{x})$ is in favor of x^2 or at least $x^{1-\min_j \eta_{0,j}}$ when $b = 0$. There are two possible explanations for the optimality of $2-a$ over $\max\{2-a, 2-b\}$ or $\max\{2-a, 1-\min_j \eta_{0,j}\}$: (1) The AUC metric is measured only on our interest, edge recovery for the interaction matrix, which only depends on $\boldsymbol{x}^a$; (2) using the profiled estimator weakens the effect of b.



(a) $n = 80$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator

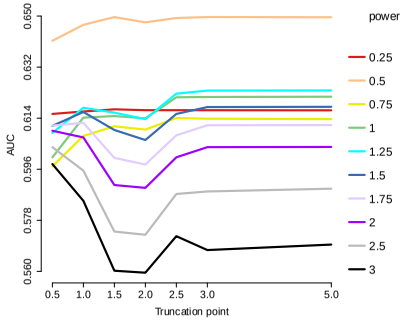
(b) $n = 1000$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator



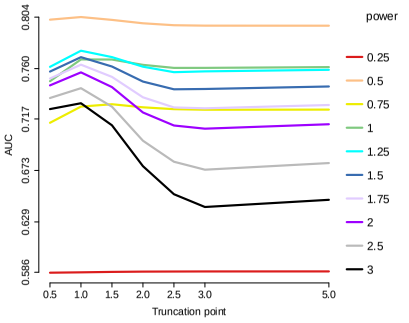
(c) $n = 80$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator

(d) $n = 1000$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator

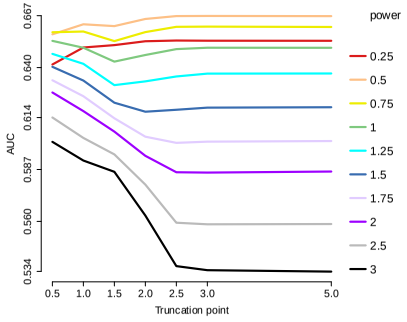
Figure 8: AUCs for edge recovery using generalized score matching for the gamma models. Each curve represents a different choice of power p in $h(x) = \min(x^p, c)$, and the x axis marks the truncation point c. Colors are sorted by p.



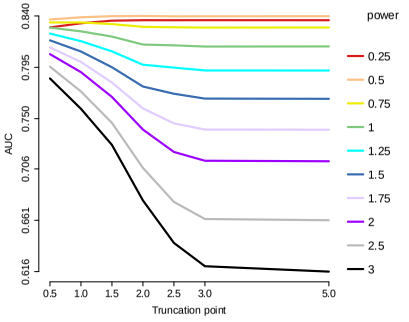
(a) $n = 80$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator



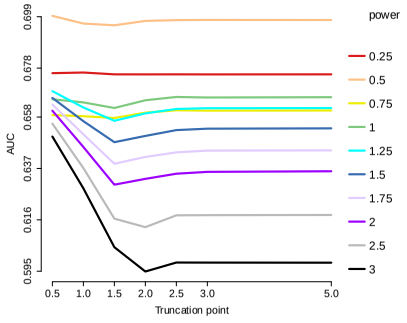
(b) $n = 1000$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator



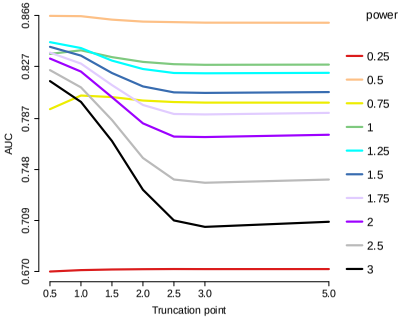
(c) $n = 80$, $\boldsymbol{\eta} = \mathbf{0}$, centered estimator



(d) $n = 1000$, $\boldsymbol{\eta} = \mathbf{0}$, centered estimator

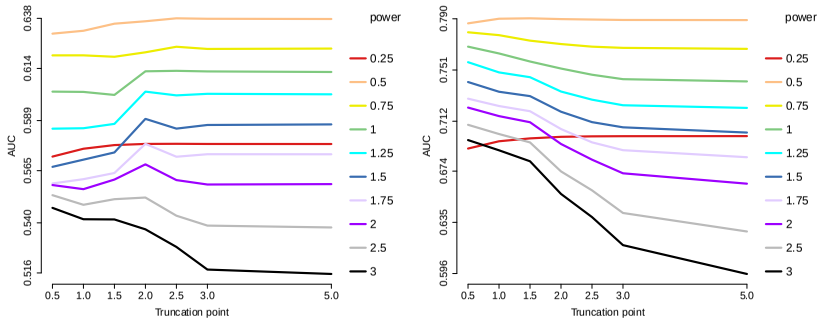


(e) $n = 80$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator



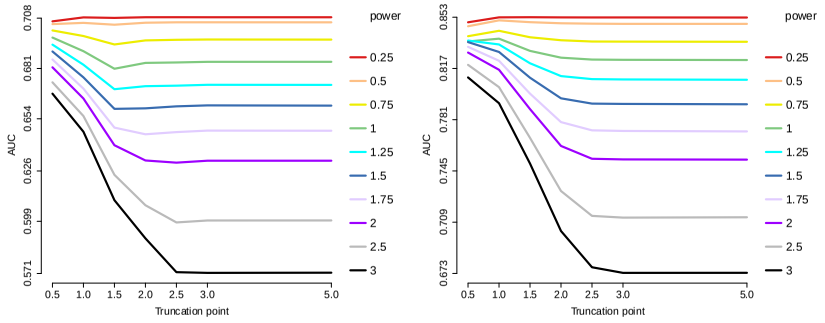
(f) $n = 1000$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator

Figure 9: AUCs for edge recovery using generalized score matching for $a = 3/2$, $b = 1/2$. Each curve represents a different choice of power p in $h(x) = \min(x^p, c)$, and the x axis marks the truncation point c. Colors are sorted by p.



(a) $n = 80$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator

(b) $n = 1000$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator



(c) $n = 80$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator

(d) $n = 1000$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator

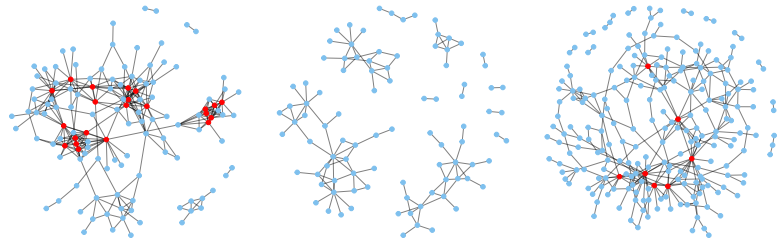
Figure 10: AUCs for edge recovery using generalized score matching for $a = 3/2$, $b = 0$. Each curve represents a different choice of power p in $h(x) = \min(x^p, c)$, and the x axis marks the truncation point c. Colors are sorted by p.

7.4. RNAseq Data

In this section we apply our regularized generalized $\boldsymbol{h}$-score matching estimator for truncated non-centered GGMs to RNAseq data also studied in Lin et al. (2016), since the same model is considered therein. The data consists of $n = 487$ prostate adenocarcinoma samples from The Cancer Genome Atlas (TCGA) data set. Following Lin et al. (2016), we focus on $m = 333$ genes that belong to the known cancer pathways in the Kyoto Encyclopedia of Genes and Genomes (KEGG) and that have no more than 10% missing values. Missing values are set to 0. We choose $h(x) = \min(x, 3)$ and use the upper-bound multiplier (*high*), as discussed in Section 7.2.2. For simplicity, we use the profiled estimator, and choose the regularization parameter λ so that the estimated graph has exactly $m = 333$ edges, all these choices being as in Lin et al. (2016).

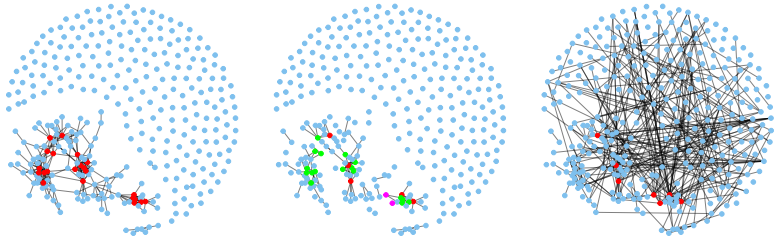
We compare our graph to the one in Lin et al. (2016), which corresponds to $h(x) = x^2$ with no multiplier. Shown in Figure 11 are the estimated graphs, with their intersection in the middle. To improve visualization, isolated nodes are removed and the layouts are optimized for each plot. Red-colored points are the "hub nodes", namely nodes with degree at least 10. In Figure 12, we plot the same graphs in a fixed layout optimized for the graph corresponding to $h = \min(x, 3)$, and include the isolated nodes.

Out of 333 edges, the two estimated graphs share 117 edges in common. Assuming that edges are placed at random between nodes and the two graphs are independent, the distribution of the number R of common edges follow a hypergeometric distribution, so $P(R = r) = \frac{\binom{m}{r}\binom{m(m-1)/2-m}{m-r}}{\binom{m(m-1)/2}{m}}$. For $m = 333$ the probability of at least 117 common edges is essentially zero. The large number of shared edges between the two methods can be explained by the fact that they both minimize the same underlying score-matching loss.



(a) $\min(x, 3)$ (b) Common edges (c) Lin et al.

Figure 11: Graphs estimated by regularized generalized score matching estimator with $h(x) = \min(x, 3)$ with upper-bound multiplier (left) and $h(x) = x^2$ with no multiplier (Lin et al., 2016, right), and their intersection graph (middle). Isolated nodes with no edges are removed, and the layout is optimized for each plot. In (a) and (c), red points indicate nodes with degree at least 10 ("hub nodes").



(a) $\min(x, 3)$ (b) Common edges (c) Lin et al.

Figure 12: Graphs estimated by regularized generalized score matching estimator with $h(x) = \min(x, 3)$ with upper-bound multiplier (left) and $h(x) = x^2$ with no multiplier (Lin et al., 2016, right), and their intersection graph (middle). Isolated nodes are included and the layout is fixed across plots and optimized for graph (a). In (b) the red nodes are hub nodes shared by both graphs, the green ones are hub nodes in graph (a) only, and the magenta ones are hub nodes in graph (c) only.

The graph using $h(x) = \min(x, 3)$ has much more isolated nodes (204) than the other (108), and has a slightly smaller max degree (16 versus 19). Table 3 provides another way of comparing between the two graphs by listing the genes with the highest node degrees.

$\min(x, 3)$ with multiplier 1.63	Lin et al.
LAMB3 (16)	CCNE2 (19)
PIK3CG (16)	**PIK3CG (16)**
MMP2 (15)	BRCA2 (13)
GLI2 (13)	**BIRC5 (12)**
LAMA4 (13)	**LAMB3 (10)**
PDGFRB (13)	**PIK3CD (10)**
PIK3CD (13)	SKP2 (10)
RASSF5 (13)	HRAS (9)
BIRC5 (12)	STAT5B (9)
FLT3 (12)	**GSTP1(8)**
GSTP1 (12)	**PDGFRB (8)**
LAMA2 (12)	
RAC2 (12)	

Table 3: List of genes with the highest node degrees in each estimated graph.

In Table 3 we list the top ten genes in terms of node degree for both estimated graphs. Due to ties, 13 genes are listed for $h(x) = \min(x, 3)$ and 11 for Lin et al. (2016). As noted in Lin et al. (2016), genes with high node-degrees are known to be important in biological

networks (Carter et al., 2004; Jeong et al., 2001; Han et al., 2004). Among these top genes, six are common in both graphs, and are discussed in Lin et al. (2016). We next elaborate on the evidence supporting the first four of the newly discovered genes.

- MMP2 (Matrix metalloproteinase 2): According to Trudel et al. (2003), increased MMP-2 expression is an independent predictor of decreased prostate cancer disease-free survival. Morgia et al. (2005) state that activity of MMP-2 can be useful in diagnosis, therapy, and assessment of malignant progression in prostate cancer.

- GLI2 (GLI family zinc finger 2): GLI2 is a primary mediator of the hedgehog signaling pathway, which has been reported in prostate cancer, and plays a critical role in the malignant phenotype of prostate cancer cells (Thiyagarajan et al., 2007). Its increased level of expression is also related to AI prostate cancer, and may be a therapeutic target in castrate-resistant prostate cancer (Narita et al., 2008).

- LAMA4 (Laminin subunit alpha 4): LAMA4 is consistently upregulated in benign prostatic hyperplasia when compared to normal prostate tissues (Luo et al., 2002).

- RASSF5 (RAS association domain family member 5): The combination of RASSF5 along with four other DNA methylation markers can effectively differentiate between benign prostate biopsy cores from non-cancer patients and cancer cores, and can be used to identify patients at risk without repeat biopsies (Brikun et al., 2014).

We note that the two methods indeed use different estimators (different h functions and multipliers), and it is thus not surprising to see that some of the top genes by one method are not among those for the other. In particular, CCNE2, BRCA2, SKP2 and STAT5B, while previously reported as newly discovered in Lin et al. (2016), are dropped by our new analysis. Testing and inference (potentially using bootstrapping) is an important problem but is beyond the scope of this paper.

8. Discussion

In this paper, we proposed a generalization of the score matching estimator of Hyvärinen (2007), based on scaling the log-gradients to be matched with a suitably chosen function $\boldsymbol{h}$. The generalization retains the advantages of Hyvärinen's method: Estimates can be computed without knowledge of normalizing constants, and for canonical parameters of exponential families, the estimation loss is a quadratic function.

For high-dimensional exponential family graphical models, following Lin et al. (2016), we add an ℓ_1 penalty to regularize the generalized score matching loss. One practical issue that is overlooked in Lin et al. (2016) is the fact that the score matching loss can be unbounded below for a small tuning parameter, when the dimension m exceeds the sample size n. We fix this issue by amplifying the diagonal entries in the quadratic matrix in the definition of the generalized score matching loss by a factor/multiplier, and we give an upper bound on that multiplier that guarantees consistency.

As examples we consider *pairwise interaction power models* on the non-negative orthant $\mathbb{R}^m_+$. Specifically, the considered models are exponential families in which the log density is

the sum of pairwise interactions between entries in of powers $\boldsymbol{x}^a$ plus linearly weighted effects $\boldsymbol{x}^b$, or $\log(\boldsymbol{x})$ when $b = 0$. Our main interest is in the matrix of interaction parameters whose support determines the distributions' conditional independence graph. The considered framework covers truncated normal distributions ($a = b = 1$), exponential square root graphical models ($a = b = 1/2$) from Inouye et al. (2016), as well as a class of multivariate gamma distributions ($a = 1/2$, $b = 0$).

In the case of multivariate truncated normal distributions, where the conditional independence graph is given by the underlying Gaussian inverse covariance matrix, the sample size required for the consistency of our method using bounded $\boldsymbol{h}$ is $\Omega(d^2 \log m)$, where d is the degree of the graph. This matches the rates for Gaussian graphical models in Ravikumar et al. (2011) and Lin et al. (2016). In contrast, the sample complexity for truncated Gaussian models given in Lin et al. (2016) is $\Omega(d^2 \log^8 m)$.

For the considered class of pairwise interaction models, we recommend using the function $\boldsymbol{h}$ with coordinates $h_j(x) = \min\left(x^{2-a}, c\right)$ for some moderately large c, or simply $h_j(x) = x^{2-a}$. While this choice is effective, it would be an interesting problem for future work to develop a method that adaptively chooses an optimized function $\boldsymbol{h}$ from data.

Acknowledgments

This work was partially supported by grant DMS/NIGMS-1561814 from the National Science Foundation (NSF). AS also gratefully acknowledges funding by grant R01-GM114029 from the National Institute of Health (NIH).

Appendix A. Proofs

A.1. Proof of Theorem 3

The following integration by parts lemma is used in the proof of Theorem 3.

Lemma 19 *Let $f, g : \mathbb{R}_+ \to \mathbb{R}$ be functions that are absolutely continuous in every bounded sub-interval of $\mathbb{R}_+$. Then*

$$\lim_{x \nearrow +\infty} f(x)g(x) - \lim_{x \searrow 0^+} f(x)g(x) = \int_0^\infty f(\boldsymbol{x}) \frac{\mathrm{d}g(x)}{\mathrm{d}x}\,\mathrm{d}x + \int_0^\infty g(\boldsymbol{x}) \frac{\mathrm{d}f(x)}{\mathrm{d}x}\,\mathrm{d}x.$$

Proof This is an analog of Lemma 4 from Hyvärinen (2005) that can be proved by integrating the partial derivatives, and follows from the fundamental theorem of calculus for absolutely continuous functions and the product rule. In particular, we work on integrals in bounded $[0, c]$, where the product of two absolutely continuous functions in a bounded interval is again absolutely continuous, and the result is then obtained by letting $c \nearrow +\infty$. ■

Proof [Proof of Theorem 3] Recall the following assumptions from Section 2.2:

(A1) $p_0(\boldsymbol{x}) h_j(x_j) \partial_j \log p(\boldsymbol{x}) \Big|_{x_j \searrow 0^+}^{x_j \nearrow +\infty} = 0\ ,\ \ \forall \boldsymbol{x}_{-j} \in \mathbb{R}_+^{m-1}, \quad \forall p \in \mathcal{P}_+$;

(A2) $\mathbb{E}_{p_0} \|\nabla \log p(\boldsymbol{X}) \circ \boldsymbol{h}^{1/2}(\boldsymbol{X})\|_2^2 < +\infty,\ \ \mathbb{E}_{p_0} \|(\nabla \log p(\boldsymbol{X}) \circ \boldsymbol{h}(\boldsymbol{X}))'\|_1 < +\infty,\ \ \forall p \in \mathcal{P}_+$.

Without explicitly writing the domains $\mathbb{R}_+$ or $\mathbb{R}_+^m$ in all integrals, by (6) we have

$$\begin{aligned} J_{\boldsymbol{h}}(p) &= \frac{1}{2} \int p_0(\boldsymbol{x}) \Big[\|\nabla \log p(\boldsymbol{x}) \circ \boldsymbol{h}^{1/2}(\boldsymbol{x})\|_2^2 \\ &\qquad -2(\nabla \log p(\boldsymbol{x}) \circ \boldsymbol{h}^{1/2}(\boldsymbol{x}))^\top (\nabla \log p_0(\boldsymbol{x}) \circ \boldsymbol{h}^{1/2}(\boldsymbol{x})) + \|\nabla \log p_0(\boldsymbol{x}) \circ \boldsymbol{h}^{1/2}(\boldsymbol{x})\|_2^2 \Big]\,\mathrm{d}\boldsymbol{x} \\ &= \underbrace{\frac{1}{2} \int p_0(\boldsymbol{x}) \sum_{j=1}^m h_j(x_j) \left(\frac{\partial \log p(\boldsymbol{x})}{\partial x_j} \right)^2 \mathrm{d}\boldsymbol{x}}_{\equiv A} + \underbrace{\frac{1}{2} \int p_0(\boldsymbol{x}) \sum_{j=1}^m h_j(x_j) \left(\frac{\partial \log p_0(\boldsymbol{x})}{\partial x_j} \right)^2 \mathrm{d}\boldsymbol{x}}_{\equiv C} \\ &\qquad \underbrace{- \int p_0(\boldsymbol{x}) \sum_{j=1}^m h_j(x_j) \frac{\partial \log p(\boldsymbol{x})}{\partial x_j} \frac{\partial \log p_0(\boldsymbol{x})}{\partial x_j}\,\mathrm{d}\boldsymbol{x}}_{\equiv B}, \end{aligned}$$

where A will simply appear in the final display as is, C is a constant as it only involves the true pdf p_0, and we wish to simplify B by integration by parts. We can split the integral into these three parts since A and C are assumed finite in the first part of (A2), and the integrand in B is integrable since $|2ab| \le a^2 + b^2$. Thus, by linearity and Fubini's theorem, we can write

$$\begin{aligned} B &= -\sum_{j=1}^m \int p_0(\boldsymbol{x}) h_j(x_j) \frac{\partial \log p(\boldsymbol{x})}{\partial x_j} \frac{\partial \log p_0(\boldsymbol{x})}{\partial x_j}\,\mathrm{d}\boldsymbol{x} \\ &= -\sum_{j=1}^m \int \left[\int p_0(\boldsymbol{x}) h_j(x_j) \frac{\partial \log p(\boldsymbol{x})}{\partial x_j} \frac{\partial \log p_0(\boldsymbol{x})}{\partial x_j}\,\mathrm{d}x_j \right] \mathrm{d}\boldsymbol{x}_{-j}. \end{aligned}$$

By the fact that $\frac{\partial \log p_0(\boldsymbol{x})}{\partial x_j} = \frac{1}{p_0(\boldsymbol{x})}\frac{\partial p_0(\boldsymbol{x})}{\partial x_j}$, this can be simplified to

$$B = -\sum_{j=1}^{m}\int\left[\int \frac{\partial p_0(\boldsymbol{x})}{\partial x_j}h_j(x_j)\frac{\partial \log p(\boldsymbol{x})}{\partial x_j}\,\mathrm{d}x_j\right]\mathrm{d}\boldsymbol{x}_{-j}.$$

But, we assume p_0 and p are twice continuously differentiable, for every $j = 1,\dots,m$ and fixed $\boldsymbol{x}_{-j} \in \mathbb{R}_+^{m-1}$. Hence, in every bounded sub-interval of $\mathbb{R}_+$, $p_0(\boldsymbol{x}_{-j}; x_j)$ is an absolutely continuous function of x_j, $\partial_j \log p(\boldsymbol{x}_{-j}, x_j) = \partial_j p(\boldsymbol{x}_{-j}, x_j)/p(\boldsymbol{x}_{-j}, x_j)$ is a continuously differentiable (and hence absolutely continuous) function of x_j by the quotient rule. Thus $h_j(x_j)\partial_j \log p(\boldsymbol{x}_{-j}; x_j)$ is also absolutely continuous by the absolute continuity assumption on h_j. Then, by Lemma 19, where we take $f \equiv p_0(\boldsymbol{x}_{-j}; x_j)$ and $g \equiv h_j(x_j)\partial_j \log p(\boldsymbol{x}_{-j}; x_j)$ as functions of x_j, followed by assumption (A1),

$$\begin{aligned}B =& -\sum_{j=1}^{m}\int\Bigg[\lim_{a\nearrow+\infty, b\searrow 0^+}\left[p_0(\boldsymbol{x}_{-j}; a)h_j(a)\partial_j \log p(\boldsymbol{x}_{-j}, a) - p_0(\boldsymbol{x}_{-j}; b)h_j(b)\partial_j \log p(\boldsymbol{x}_{-j}, b)\right]\\ &- \int p_0(\boldsymbol{x})\frac{\partial\left(h_j(x_j)\partial_j \log p(\boldsymbol{x})\right)}{\partial x_j}\,\mathrm{d}x_j\Bigg]\mathrm{d}\boldsymbol{x}_{-j}\\ =& \sum_{j=1}^{m}\int\left[\int p_0(\boldsymbol{x})\frac{\partial(h_j(x_j)\partial_j \log p(\boldsymbol{x}))}{\partial x_j}\,\mathrm{d}x_j\right]\mathrm{d}\boldsymbol{x}_{-j}.\end{aligned}$$

Justified by the second half of (A2), by Fubini-Tonelli and linearity again

$$\begin{aligned}B &= \sum_{j=1}^{m}\int p_0(\boldsymbol{x})\frac{\partial(h_j(x_j)\partial_j \log p(\boldsymbol{x}))}{\partial x_j}\,\mathrm{d}\boldsymbol{x},\\ &= \sum_{j=1}^{m}\int h_j'(x_j)\frac{\partial \log p(\boldsymbol{x})}{\partial x_j}p_0(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} + \sum_{j=1}^{m}\int h_j(x_j)\frac{\partial^2 \log p(\boldsymbol{x})}{\partial x_j^2}p_0(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}.\end{aligned}$$

Thus,

$$\begin{aligned}&J_{\boldsymbol{h}}(p)\\ =&B + A + C\\ =&\int_{\mathbb{R}_+^m} p_0(\boldsymbol{x})\sum_{j=1}^{m}\left[h_j'(x_j)\frac{\partial \log p(\boldsymbol{x})}{\partial x_j} + h_j(x_j)\frac{\partial^2 \log p(\boldsymbol{x})}{\partial x_j^2} + \frac{1}{2}h_j(x_j)\left(\frac{\partial \log p(\boldsymbol{x})}{\partial x_j}\right)^2\right]\mathrm{d}\boldsymbol{x} + C,\end{aligned}$$

where C is a constant that does not depend on p. ■

A.2. Proof of Theorems and Examples in Section 3

Proof [Proof of Theorem 5] For exponential families and under the assumptions, the empirical loss $\hat{J}_{\boldsymbol{h}}(p_{\boldsymbol{\theta}})$ in (8) becomes (up to an additive constant)

$$\begin{aligned}&\hat{J}_{\boldsymbol{h}}(p_{\boldsymbol{\theta}})\\ =&\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m}\left[h_j'(X_j^{(i)})\frac{\partial \log p_{\boldsymbol{\theta}}(\boldsymbol{X}^{(i)})}{\partial X_j^{(i)}} + h_j(X_j^{(i)})\frac{\partial^2 \log p_{\boldsymbol{\theta}}(\boldsymbol{X}^{(i)})}{\partial (X_j^{(i)})^2}\right.\\ &\left.+\frac{1}{2}h_j(X_j^{(i)})\left(\frac{\partial \log p_{\boldsymbol{\theta}}(\boldsymbol{X}^{(i)})}{\partial X_j^{(i)}}\right)^2\right]\\ =&\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m}\left[h_j'(X_j^{(i)})(\boldsymbol{\theta}^\top \boldsymbol{t}_j'(\boldsymbol{X}^{(i)}) + b_j'(\boldsymbol{X}^{(i)})) + h_j(X_j^{(i)})(\boldsymbol{\theta}^\top \boldsymbol{t}_j''(\boldsymbol{X}^{(i)}) + b_j''(\boldsymbol{X}^{(i)}))\right.\\ &\left.+\frac{1}{2}h_j(X_j^{(i)})(\boldsymbol{\theta}^\top \boldsymbol{t}_j'(\boldsymbol{X}^{(i)}) + b_j'(\boldsymbol{X}^{(i)}))^2\right]\\ =&\frac{1}{n}\left\{\frac{1}{2}\boldsymbol{\theta}^\top\left[\sum_{i=1}^{n}\sum_{j=1}^{m}h_j(X_j^{(i)})\boldsymbol{t}_j'(\boldsymbol{X}^{(i)})\boldsymbol{t}_j'(\boldsymbol{X}^{(i)})^\top\right]\boldsymbol{\theta} + \right.\\ &\left.\left[\sum_{i=1}^{n}\sum_{j=1}^{m}h_j(X_j^{(i)})b_j'(\boldsymbol{X}^{(i)})\boldsymbol{t}_j'(\boldsymbol{X}^{(i)}) + h_j(X_j^{(i)})\boldsymbol{t}_j''(\boldsymbol{X}^{(i)}) + h_j'(X_j^{(i)})\boldsymbol{t}_j'(\boldsymbol{X}^{(i)})\right]^\top\boldsymbol{\theta}\right\} + \text{const},\end{aligned}$$

which is quadratic in $\boldsymbol{\theta}$. Let

$$\boldsymbol{\Gamma}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m}h_j(X_j^{(i)})\boldsymbol{t}_j'(\boldsymbol{X}^{(i)})\boldsymbol{t}_j'(\boldsymbol{X}^{(i)})^\top, \tag{34}$$

$$\boldsymbol{g}(\mathbf{x}) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m}\left[h_j(X_j^{(i)})b_j'(\boldsymbol{X}^{(i)})\boldsymbol{t}_j'(\boldsymbol{X}^{(i)}) + h_j(X_j^{(i)})\boldsymbol{t}_j''(\boldsymbol{X}^{(i)}) + h_j'(X_j^{(i)})\boldsymbol{t}_j'(\boldsymbol{X}_i)\right]. \tag{35}$$

Then we can write $\hat{J}_{\boldsymbol{h}}(p_{\boldsymbol{\theta}}) = \frac{1}{2}\boldsymbol{\theta}^\top\boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta} - \boldsymbol{g}(\mathbf{x})^\top\boldsymbol{\theta} + \text{const}$. ■

Proof [Proof of Theorem 6] By Theorem 5, $\hat{J}_{\boldsymbol{h}}(p_{\boldsymbol{\theta}}) = \frac{1}{2}\boldsymbol{\theta}^\top\boldsymbol{\Gamma}\boldsymbol{\theta} - \boldsymbol{g}^\top\boldsymbol{\theta} + \text{const}$. The minimizer of $\hat{J}_{\boldsymbol{h}}(p_{\boldsymbol{\theta}})$ is thus available in the unique closed form $\hat{\boldsymbol{\theta}} \equiv \boldsymbol{\Gamma}(\mathbf{x})^{-1}\boldsymbol{g}(\mathbf{x})$ as long as $\boldsymbol{\Gamma}$ is invertible (C1). Since $\boldsymbol{\Gamma}$ and $\boldsymbol{g}$ are sample averages, the weak law of large numbers yields that $\boldsymbol{\Gamma} \to_p \mathbb{E}_{p_0}\boldsymbol{\Gamma} \equiv \boldsymbol{\Gamma}_0$ and $\boldsymbol{g} \to_p \mathbb{E}_{p_0}\boldsymbol{g} \equiv \boldsymbol{g}_0$, where existence of $\boldsymbol{\Gamma}_0$ and $\boldsymbol{g}_0$ is assumed in (C2). Since $J_{\boldsymbol{h}}(p_{\boldsymbol{\theta}}) = \mathbb{E}[\hat{J}_{\boldsymbol{h}}(p_{\boldsymbol{\theta}})] = \mathbb{E}[\frac{1}{2}\boldsymbol{\theta}^\top\boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta} - \boldsymbol{g}(\mathbf{x})^\top\boldsymbol{\theta}] = \frac{1}{2}\boldsymbol{\theta}^\top\boldsymbol{\Gamma}_0\boldsymbol{\theta} - \boldsymbol{g}_0\boldsymbol{\theta}$ and we know $\boldsymbol{\theta}_0$ minimizes $J_{\boldsymbol{h}}(p_{\boldsymbol{\theta}})$ by definition, by the first-order condition we must have $\boldsymbol{\Gamma}_0\boldsymbol{\theta}_0 = \boldsymbol{g}_0$. Then by the Lindeberg-Lévy central limit theorem,

$$\sqrt{n}(\boldsymbol{g}(\mathbf{x}) - \boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta}_0) \to_d \mathcal{N}_m(\mathbf{0}, \boldsymbol{\Sigma}_0),$$

where $\boldsymbol{\Sigma}_0 \equiv \mathbb{E}_{p_0}[(\boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta}_0 - \boldsymbol{g}(\mathbf{x}))(\boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta}_0 - \boldsymbol{g}(\mathbf{x}))^\top]$, as long as $\boldsymbol{\Sigma}_0$ exists (C2). Thus, by Slutsky's theorem,

$$\sqrt{n}(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0) \equiv \sqrt{n}(\boldsymbol{\Gamma}(\mathbf{x})^{-1}(\boldsymbol{g}(\mathbf{x}) - \boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\theta}_0)) \to_d \mathcal{N}_r(\mathbf{0}, \boldsymbol{\Gamma}_0^{-1}\boldsymbol{\Sigma}\boldsymbol{\Gamma}_0^{-1}),$$

as long as $\mathbf{\Gamma}_0$ is invertible (C2).

For the second half of the theorem, (C2) $\mathbb{E}_{p_0}\mathbf{\Gamma}(\mathbf{x}) < \infty$ and $\mathbb{E}_{p_0}\boldsymbol{g}(\mathbf{x}) < \infty$ implies $\mathbb{E}_{p_0}|\mathbf{\Gamma}(\mathbf{x})| < \infty$ and $\mathbb{E}_{p_0}|\boldsymbol{g}(\mathbf{x})| < \infty$, so by strong law of large numbers (and a union bound on at most k^2 null sets)

$$\mathbf{\Gamma}(\mathbf{x}) \to_{\text{a.s.}} \mathbf{\Gamma}_0, \quad \boldsymbol{g}(\mathbf{x}) \to_{\text{a.s.}} \boldsymbol{g}_0.$$

Then outside a null set,

$$\hat{\boldsymbol{\theta}} \equiv \mathbf{\Gamma}(\mathbf{x})^{-1}\boldsymbol{g}(\mathbf{x}) \to_{\text{a.s.}} \mathbf{\Gamma}_0^{-1}\boldsymbol{g}_0 = \boldsymbol{\theta}_0.$$

■

Proof [Proof for Example 3.1] We choose to estimate $\theta \equiv \mu/\sigma^2$. Then by (11) and (12),

$$\begin{aligned}
\hat{\mu}_h &= \sigma^2\hat{\theta} \equiv \sigma^2\Gamma(\boldsymbol{x})^{-1}g(\boldsymbol{x}) \\
&= -\sigma^2\left[\sum_{i=1}^n h(X_i)t'(X_i)^2\right]^{-1}\left[\sum_{i=1}^n h(X_i)b'(X_i)t'(X_i) + h(X_i)t''(X_i) + h'(X_i)t'(X_i)\right] \\
&= -\sigma^2\left[\sum_{i=1}^n h(X_i)\right]^{-1}\left[\sum_{i=1}^n -h(X_i)\frac{X_i}{\sigma^2} + h'(X_i)\right].
\end{aligned}$$

By Theorem 6,

$$\begin{aligned}
\sqrt{n}(\hat{\mu}_h - \mu_0) &\to_d \mathcal{N}\left(0, \frac{\sigma^4\mathbb{E}_0\left[-h(X)\frac{X-\mu_0}{\sigma^2} + h'(X)\right]^2}{\mathbb{E}_0^2[h(X)]}\right) \\
&\sim \mathcal{N}\left(0, \frac{\mathbb{E}_0\left[-h(X)(X-\mu_0) + \sigma^2 h'(X)\right]^2}{\mathbb{E}_0^2[h(X)]}\right).
\end{aligned}$$

By integration by parts, (suppressing the dependence of p_{μ_0} on μ_0)

$$\begin{aligned}
&\mathbb{E}_0[h(X)h'(X)(X-\mu_0)] \\
&= \int_0^\infty h'(x)h(x)(x-\mu_0)p(x)\,\mathrm{d}x \\
&= \int_0^\infty h(x)(x-\mu_0)p(x)\,\mathrm{d}h(x) \\
&= h^2(x)(x-\mu_0)p(x)|_0^\infty - \int h(x)\,\mathrm{d}h(x)(x-\mu_0)p(x) \\
&= -\int h^2(x)p(x)\,\mathrm{d}x - \int h(x)h'(x)(x-\mu_0)p(x)\,\mathrm{d}x + \int h^2(x)\frac{(x-\mu_0)^2}{\sigma^2}p(x)\,\mathrm{d}x,
\end{aligned}$$

where the last step follows from the assumptions $\lim_{x\searrow 0^+} h(x) = 0$ and $\lim_{x\nearrow+\infty} h^2(x)(x-\mu_0)p_{\mu_0}(x) = 0$. So

$$\mathbb{E}_0[h(X)h'(X)(X-\mu_0)] = \frac{\mathbb{E}[h^2(X)((X-\mu_0)^2/\sigma^2 - 1)]}{2}. \tag{36}$$

The asymptotic variance is thus

$$\begin{aligned}
&\frac{\mathbb{E}_0\left[-h(X)(X-\mu_0) + \sigma^2 h'(X)\right]^2}{\mathbb{E}_0^2[h(X)]} \\
&= \frac{\mathbb{E}_0\left[h^2(X)(X-\mu_0)^2 - 2\sigma^2 h^2(X)\left((X-\mu_0)^2/\sigma^2 - 1\right)/2 + \sigma^4 h'^2(X)\right]}{\mathbb{E}_0^2[h(X)]} \\
&= \frac{\mathbb{E}_0[\sigma^2 h^2(X) + \sigma^4 h'^2(X)]}{\mathbb{E}_0^2[h(X)]}.
\end{aligned}$$

The Cramér-Rao lower bound follows from taking the second derivative of $\log p_{\mu_0}$ with respect to μ_0. ■

Proof [Proof for Example 3.2] We estimate $\theta \equiv 1/\sigma^2$. By (11) and (12),

$$\begin{aligned}
\hat{\theta} &\equiv \Gamma(\boldsymbol{x})^{-1}g(\boldsymbol{x}) \\
&= -\left[\sum_{i=1}^n h(X_i)t'(X_i)^2\right]^{-1}\left[\sum_{i=1}^n h(X_i)b'(X_i)t'(X_i) + h(X_i)t''(X_i) + h'(X_i)t'(X_i)\right] \\
&= \left[\sum_{i=1}^n h(X_i)(X_i-\mu)^2\right]^{-1}\left[\sum_{i=1}^n h(X_i) + h'(X_i)(X_i-\mu)\right].
\end{aligned}$$

By Theorem 6, $\sqrt{n}(\hat{\theta} - \theta) \to_d \mathcal{N}(0, \varsigma^2)$, where

$$\begin{aligned}
\varsigma^2 &\equiv \frac{\mathbb{E}_0\left[h(X)((X-\mu)^2/\sigma_0^2 - 1) - h'(X)(X-\mu)\right]^2}{\mathbb{E}_0^2[h(X)(X-\mu)^2]} \\
&= \frac{1}{\mathbb{E}_0^2[h(X)(X-\mu)^2]}\Big(\mathbb{E}_0[h^2(X)(X-\mu)^4/\sigma_0^4 - 2h^2(X)(X-\mu)^2/\sigma_0^2 + h^2(X) \\
&\qquad + h'^2(X)(X-\mu)^2 - 2h(X)h'(X)(X-\mu)^3/\sigma_0^2 + 2h(X)h'(X)(X-\mu)\Big).
\end{aligned}$$

By integration by parts, (suppressing the dependence of $p_{\sigma_0^2}$ on σ_0^2)

$$\begin{aligned}
&\mathbb{E}_0[h(X)h'(X)(X-\mu)^3] \\
&= \int_0^\infty h'(x)h(x)(x-\mu)^3p(x)\,\mathrm{d}x \\
&= \int_0^\infty h(x)(x-\mu)^3p(x)\,\mathrm{d}h(x) \\
&= h^2(x)(x-\mu)^3p(x)|_0^\infty - \int h(x)\,\mathrm{d}h(x)(x-\mu)^3p(x) \\
&= -\int h(x)h'(x)(x-\mu)^3p(x)\,\mathrm{d}x - 3\int h^2(x)(x-\mu)^2p(x)\,\mathrm{d}x + \int h^2(x)\frac{(x-\mu)^4}{\sigma_0^2}p(x)\,\mathrm{d}x,
\end{aligned}$$

where the last step follows from the assumptions $\lim_{x\searrow 0^+} h(x) = 0$ and $\lim_{x\nearrow+\infty} h^2(x)(x-\mu)^3 p_{\sigma_0^2}(x) = 0$. Combining this with (36) we get

$$\sqrt{n}(\hat{\theta}-\theta) \to_d \mathcal{N}(0,\varsigma^2) \sim \mathcal{N}\left(0, \frac{2\mathbb{E}_0[h^2(X)(X-\mu)^2/\sigma_0^2] + \mathbb{E}_0[h'^2(X-\mu)^2]}{\mathbb{E}_0^2[h(X)(X-\mu)^2]}\right),$$

and so by the delta method, for $\hat{\sigma}_k^2 \equiv \hat{\theta}^{-1}$,

$$\sqrt{n}(\hat{\sigma}_h^2 - \sigma_0^2) \to_d \mathcal{N}\left(0, \frac{2\sigma_0^6\mathbb{E}_0[h^2(X)(X-\mu)^2] + \sigma_0^8\mathbb{E}_0[h'^2(X-\mu)^2]}{\mathbb{E}_0^2[h(X)(X-\mu)^2]}\right).$$

The Cramér-Rao lower bound follows from taking the second derivative of $\log p_{\sigma_0^2}$ with respect to σ_0^2. ■

A.3. Proof of Theorems in Section 5

Proof [Proof of Theorem 9]
Case $b \neq 0$: We use a strategy similar to that of Inouye et al. (2016). Let $\mathcal{V}_1 = \{\boldsymbol{v} : \|\boldsymbol{v}\|_1 = 1, \boldsymbol{v} \in \mathbb{R}_+^m\}$. Then by Fubini-Tonelli the normalizing constant is,

$$\begin{aligned}
&\int_{\mathbb{R}_+^m} \exp\left(\boldsymbol{\eta}^\top \frac{\boldsymbol{x}^b - \mathbf{1}_m}{b} - \frac{1}{2a}\boldsymbol{x}^{a\top}\mathbf{K}\boldsymbol{x}^a\right) \mathrm{d}\boldsymbol{x} \\
&= \int_{\mathcal{V}_1}\int_0^\infty \exp\left(\boldsymbol{\eta}^\top \frac{z^b\boldsymbol{v}^b - \mathbf{1}_m}{b} - \frac{1}{2a}z^{2a}\boldsymbol{v}^{a\top}\mathbf{K}\boldsymbol{v}^a\right) \mathrm{d}z\,\mathrm{d}\boldsymbol{v} \\
&\propto \int_{\mathcal{V}_1}\int_0^\infty \exp\left(z^b(\boldsymbol{\eta}^\top\boldsymbol{v}^b)/b - z^{2a}(\boldsymbol{v}^{a\top}\mathbf{K}\boldsymbol{v}^a)/(2a)\right) \mathrm{d}z\,\mathrm{d}\boldsymbol{v}.
\end{aligned}$$

Here $\mathcal{V}_1$ is compact and the inner integral, if finite, is continuous in $\boldsymbol{v}$. It thus suffices to show that the inner integral is finite at every single $\boldsymbol{v} \in \mathcal{V}_1$.

Fixing $\boldsymbol{v} \in \mathcal{V}_1$, write $A \equiv A(\boldsymbol{v}) \equiv \boldsymbol{v}^{a\top}\mathbf{K}\boldsymbol{v}^a/(2a)$ and $B \equiv B(\boldsymbol{v}) \equiv (\boldsymbol{\eta}^\top\boldsymbol{v}^b)/b$. We need to show that

$$N(A,B,a,b) \equiv \int_0^\infty \exp\left(-Az^{2a} + Bz^b\right) \mathrm{d}z < +\infty.$$

Recall that (CC1) $\boldsymbol{v}^\top\mathbf{K}\boldsymbol{v} > 0$ for all $\boldsymbol{v} \in \mathbb{R}_+^m\backslash\{\mathbf{0}\}$, so $A > 0$.

(i) Suppose $B \leq 0$. Then $N(A,B,a,b) \leq \int_0^\infty \exp(-Az^{2a})\,\mathrm{d}z = A^{-a/2}\Gamma(1+1/(2a))$, a finite constant since $A > 0$ and $a > 0$.

(ii) Suppose $B > 0$. We first want to bound $\exp(-Az^{2a} + Bz^b) \leq N_0\exp(-Az^{2a}/2)$ by some finite constant $N_0 > 0$, so that $N(A,B,a,b) \leq N_0\int_0^\infty \exp(-Az^{2a}/2)\,\mathrm{d}z$, a finite constant for $a > 0$. Thus, it remains to give conditions so that $\exp(-Az^{2a}/2 + Bz^b)$ is bounded by some finite constant N_0, which by continuity only requires a finite limit as $z \searrow 0$ and as $z \nearrow +\infty$. As $z \nearrow +\infty$, $Bz^b \nearrow +\infty$, while $-Az^{2a}/2 \searrow -\infty$. We thus need $b < 2a$ so that the sum of the two does not go to positive infinity. On the other hand, as $z \searrow 0$, $-Az^{2a}/2 \nearrow 0$, so we need $b > 0$, otherwise $z^b \nearrow +\infty$. In conclusion, we require that $2a > b > 0$.

It thus suffices to require (CC1) and (CC2) $2a > b > 0$ to eliminate restrictions on B, and hence on $\boldsymbol{\eta}$. That is, $\boldsymbol{\eta}$ can take value in the entirety of $\mathbb{R}^m$.

Case $b = 0$: Again in (CC1) we assume $\boldsymbol{v}^\top\mathbf{K}\boldsymbol{v} > 0$ for all $\boldsymbol{v} \in \mathbb{R}_+^m\backslash\{\mathbf{0}\}$. Since $\mathcal{V}_2 \equiv \{\boldsymbol{v} : \|\boldsymbol{v}\|_2 = 1, \boldsymbol{v} \in \mathbb{R}_+^m\}$ is compact and $\boldsymbol{v}^\top\mathbf{K}\boldsymbol{v}$ is continuous in $\boldsymbol{v}$ and strictly positive on $\mathcal{V}_2$, the image of $\mathcal{V}_2$ under $\boldsymbol{v}^\top\mathbf{K}\boldsymbol{v}$ is a compact subset of $(0,\infty)$, i.e. $N_{\mathbf{K}} \equiv \min_{\boldsymbol{v}\in\mathbb{R}_+^m\backslash\{\mathbf{0}\}} \boldsymbol{v}^\top\mathbf{K}\boldsymbol{v}/\boldsymbol{v}^\top\boldsymbol{v} \equiv \min_{\boldsymbol{v}\in\mathcal{V}_2} \boldsymbol{v}^\top\mathbf{K}\boldsymbol{v} > 0$. We thus have

$$\begin{aligned}
&\int_{\mathbb{R}_+^m} \exp\left(\boldsymbol{\eta}^\top\log(\boldsymbol{x}) - \frac{1}{2a}\boldsymbol{x}^{a\top}\mathbf{K}\boldsymbol{x}^a\right)\mathrm{d}\boldsymbol{x} \\
&\leq \int_{\mathbb{R}_+^m} \exp\left(\boldsymbol{\eta}^\top\log(\boldsymbol{x}) - \frac{N_{\mathbf{K}}}{2a}\boldsymbol{x}^{a\top}\boldsymbol{x}^a\right)\mathrm{d}\boldsymbol{x} \\
&= \prod_{j=1}^m \int_0^\infty \exp\left(\eta_j\log(x_j) - \frac{N_{\mathbf{K}}}{2a}x_j^{2a}\right)\mathrm{d}x_j \\
&= \prod_{j=1}^m \left[\Gamma\left(\frac{\eta_j+1}{2a}\right)\frac{(N_{\mathbf{K}}/2a)^{-\frac{\eta_j+1}{2a}}}{2a}\right],
\end{aligned}$$

where the integration follows by change of variable and requires $a > 0$. Assuming $a > 0$, the last quantity is finite if and only if $\boldsymbol{\eta} \succ -\mathbf{1}_m$, by definition of the gamma function.

In conclusion, given conditions (CC1) $\min_{\boldsymbol{v}\in\mathbb{R}_+^m\backslash\{\mathbf{0}\}} \boldsymbol{v}^\top\mathbf{K}\boldsymbol{v} > 0$, (CC2) $2a > b > 0$, and (CC3) $a > 0$, $b = 0$ and $\boldsymbol{\eta} \succ -\mathbf{1}_m$, the unnormalized density (16) has a finite normalizing constant when (CC1) and (CC2) both hold, or (CC1) and (CC3) both hold.

The centered settings, where the term involving $\boldsymbol{x}^b$ is excluded, can be considered as a special case of both (1) and (2) with $\boldsymbol{\eta} \equiv \mathbf{0}$, and thus (CC1) and $a > 0$ are sufficient. ■

Proof [Proof of Theorem 11] Recall assumptions (A1) and (A2):

(A1) $p_0(\boldsymbol{x})h_j(x_j)\partial_j\log p(\boldsymbol{x})\Big|_{x_j\searrow 0^+}^{x_j\nearrow+\infty} = 0\,, \quad \forall \boldsymbol{x}_{-j} \in \mathbb{R}_+^{m-1}, \quad \forall p \in \mathcal{P}_+;$

(A2) $\mathbb{E}_{p_0}\|\nabla\log p(\boldsymbol{X})\circ\boldsymbol{h}^{1/2}(\boldsymbol{X})\|_2^2 < +\infty,\ \ \mathbb{E}_{p_0}\|(\nabla\log p(\boldsymbol{X})\circ\boldsymbol{h}(\boldsymbol{X}))'\|_1 < +\infty,\ \ \forall p \in \mathcal{P}_+.$

Let $\mathbf{K}_0$ and $\boldsymbol{\eta}_0$ be the true parameters so that $p_0 \in \mathcal{P}_+$, with $\mathcal{P}_+$ corresponding to a parameter space in which all parameters satisfy the conditions for a finite normalizing constant. We now give sufficient conditions for h to satisfy (A1) and (A2).

Conditions for (A1): Fix $j = 1,\ldots,m$ and $\boldsymbol{x}_{-j} \in \mathbb{R}_+^{m-1}$. We show that the conditions on h_j imply that the limits go to 0 as $x_j \nearrow +\infty$ and as $x_j \searrow 0^+$, which is stronger than (A1); in fact, from (37) below, the limits cannot go to a nonzero finite constant assuming an h with polynomial tail, since $a > 0$ and $B_1 \equiv \kappa_{0,jj} > 0$ for all j. Now,

$$\begin{aligned}
&p_0(\boldsymbol{x})h_j(x_j)\partial_j\log p(\boldsymbol{x}) \\
&\propto h_j(x_j)\exp\left(-\frac{1}{2a}\boldsymbol{x}^{a\top}\mathbf{K}_0\boldsymbol{x}^a + \boldsymbol{\eta}_0^\top\frac{\boldsymbol{x}^b - \mathbf{1}_m}{b}\right)\partial_j\left(-\frac{1}{2a}\boldsymbol{x}^{a\top}\mathbf{K}\boldsymbol{x}^a + \boldsymbol{\eta}^\top\frac{\boldsymbol{x}^b - \mathbf{1}_m}{b}\right)
\end{aligned}$$

$$\propto h_j(x_j)\exp\left(-\frac{1}{a}(\boldsymbol{k}_{0,j,-j}^\top \boldsymbol{x}_{-j}^a)x_j^a - \frac{\kappa_{0,jj}}{2a}x_j^{2a} + \eta_{0,j}\frac{x_j^b-1}{b}\right)\times \left(-\boldsymbol{k}_{j,-j}^\top \boldsymbol{x}_{-j}^a x_j^{a-1} - \kappa_{jj}x_j^{2a-1} + \eta_j x_j^{b-1}\right)$$

$$\equiv h_j(x_j)\exp\left(\frac{A_1 x_j^a}{a} + \frac{B_1 x_j^{2a}}{2a} + C_1\frac{x_j^b-1}{b}\right)\left(A_2 x_j^{a-1} + B_2 x_j^{2a-1} + C_2 x_j^{b-1}\right), \quad (37)$$

where $A_1 \equiv -\boldsymbol{k}_{0,j,-j}^\top \boldsymbol{x}_{-j}^a$, $A_2 \equiv -\boldsymbol{k}_{j,-j}^\top \boldsymbol{x}_{-j}^a$, $B_1 \equiv -\kappa_{0,jj} < 0$ and $B_2 \equiv -\kappa_{jj} < 0$ by condition (CC1). Finally $C_1 \equiv \eta_{0,j}$, $C_2 \equiv \eta_j$.

(1) Let $x_j \nearrow +\infty$. If $b > 0$, since $2a > b > 0$ and $B_1 < 0$, the exponential term in (37) decreases to 0 exponentially and its reciprocal dominates any polynomial functions. Thus, the entire product goes to 0 if $h_j(x_j)$ grows no faster than polynomially as $x_j \nearrow +\infty$. If $b = 0$, the $C_1 \log x_j$ term is again dominated by $B_1 x_j^{2a}/(2a)$, and the same conclusion holds.

(2) Let $x_j \searrow 0$.

(i) Let $b > 0$. Then the exponential term in (37) goes to constant $\exp(-C_1/b)$, and we only need

$$\lim_{x_j \searrow 0^+} h_j(x_j)(A_2 x_j^{a-1} + B_2 x_j^{2a-1} + C_2 x_j^{b-1}) = 0. \quad (38)$$

- If $a > 1$ and $b > 1$, the second term in (38) is a polynomial with three terms having powers $\geq \min\{a-1, b-1\}$. The product goes to zero if and only if $h_j(x_j) = o(x_j^{\max\{1-a,1-b\}})$ as $x_j \searrow 0$. Note that this is satisfied by any h_j that has a finite right limit at 0.
- If $a = 1$ and $b \geq 1$, or $a \geq 1$ and $b = 1$, then the second term in (38) is a polynomial of non-negative power plus a potentially nonzero constant. A sufficient condition for (38) is thus $\lim_{x_j \searrow 0} h_j(x_j) = 0$.
- If $a < 1$ or $b < 1$, then the second part in (38) is a polynomial having terms with negative degree $\geq \min\{a-1, b-1\}$. To counteract this a sufficient condition is $h_j(x_j) = o(x_j^{\max\{1-a,1-b\}})$.

In conclusion, $\lim_{x_j \searrow 0^+} p_0(\boldsymbol{x}) h_j(x_j) \partial_j \log p(\boldsymbol{x}) = 0$ if and only if

$$\lim_{x_j \searrow 0^+} h_j(x_j)/x_j^{\max\{1-a,1-b\}} = 0.$$

(ii) Now assume $b = 0$. Then, (37) now becomes

$$h_j(x_j)\exp\left(A_1 x_j^a/a + B_1 x_j^{2a}/(2a) + C_1 \log x_j\right)\left(A_2 x_j^{a-1} + B_2 x_j^{2a-1} + C_2/x_j\right).$$

With $C_1 \log x_j$ dominating, the exponential part scales as $x_j^{C_1}$. We thus require

$$\lim_{x_j \searrow 0^+} h_j(x_j)(A_2 x_j^{a-1+C_1} + B_2 x_j^{2a-1+C_1} + C_2 x_j^{C_1-1}) = 0,$$

which by the previous discussion on (38) holds if and only if

$$\lim_{x_j \searrow 0^+} h_j(x_j)/x_j^{1-C_1} = 0$$

since $1 - a - C_1 < 1 - C_1$.

In summary, (A1) is satisfied if $h_j(x_j)$ grows at most polynomially as $x_j \nearrow +\infty$, and $\lim_{x_j \searrow 0^+} h_j(x_j)/x_j^{\max\{1-a,1-b\}} = 0$ if $b > 0$, or $\lim_{x_j \searrow 0^+} h_j(x_j)/x_j^{1-\eta_{0,j}} = 0$ if $b = 0$.

Conditions for (A2): For (A2), we consider powers of x as the h functions for simplicity; conclusions for other functions that have the same tail behavior (big-O scaling) as $x \searrow 0$ and $x \nearrow +\infty$ follow similarly. Sufficiency results for piecewise power functions follow by partitioning, and similarly for other functions h whose function values and derivatives can be bounded by those of some piecewise power function (e.g. truncated powers), since (A2) is on integrability of products involving positive powers of h and h'.

Let $\mathbf{K}_0$ and $\boldsymbol{\eta}_0$ be the true parameters from the parameter space that satisfies the conditions for finite normalizing constant. By part (2) of the proof of Theorem 9, the assumption that $\mathbf{K}_0$ satisfies (CC1) implies that $\min_{\boldsymbol{v} \in \mathbb{R}_+^m \setminus \{\mathbf{0}\}} \boldsymbol{v}^\top \mathbf{K}_0 \boldsymbol{v}/\boldsymbol{v}^\top \boldsymbol{v} \equiv N_{\mathbf{K}_0} > 0$. Then we have the following decomposition

$$p_{\mathbf{K}_0,\boldsymbol{\eta}_0}(\boldsymbol{x}) \equiv \exp\left(-\frac{1}{2a}\boldsymbol{x}^{a\top}\mathbf{K}_0\boldsymbol{x}^a + \boldsymbol{\eta}_0^\top \frac{\boldsymbol{x}^b - \mathbf{1}_m}{b}\right) \leq \prod_{j=1}^m \exp\left(-\frac{N_{\mathbf{K}_0}}{2a}x_j^{2a} + \eta_{0,j}\frac{x_j^b-1}{b}\right).$$

Then for any other $\mathbf{K}$ and $\boldsymbol{\eta}$ in the parameter space, for the first part of (A2) it suffices to show for any $j = 1, \ldots, m$ that $D < \infty$, where

$$D \equiv \int_{\mathbb{R}_+^m} \prod_{j=1}^m \exp\left(-\frac{N_{\mathbf{K}_0}}{2a}x_j^{2a} + \eta_{0,j}\frac{x_j^b-1}{b}\right) h_j(x_j) \times \left(-\kappa_{jj}x_j^{2a-1} - \sum_{i \neq j}\kappa_{ji}x_i^a x_j^{a-1} + \eta_j x_j^{b-1}\right)^2 \mathrm{d}\boldsymbol{x}$$

$$\geq \int_{\mathbb{R}_+^m} p_0(\boldsymbol{x}) h_j(x_j) \left(\partial_j \log p(\boldsymbol{x})\right)^2 \mathrm{d}\boldsymbol{x}.$$

Note that

$$\left(-\kappa_{jj}x_j^{2a-1} - \sum_{i\neq j}\kappa_{ji}x_i^a x_j^{a-1} + \eta_j x_j^{b-1}\right)^2 = \kappa_{jj}^2 x_j^{4a-2} + \sum_{i\neq j,\ell\neq j}\kappa_{ji}\kappa_{j\ell}x_i^a x_\ell^a x_j^{2a-2} + \eta_j^2 x_j^{2b-2} + 2\sum_{i\neq j}\kappa_{jj}\kappa_{ji}x_i^a x_j^{3a-2} - 2\sum_{i\neq j}\kappa_{ji}\eta_j x_i^a x_j^{a+b-2} - 2\kappa_{jj}\eta_j x_j^{2a+b-2}.$$

Thus, plugging this back in the definition of D, we can split D into a sum of six terms D_1 through D_6, each of which is a sum of terms of the form

$$\begin{aligned}&\int_{\mathbb{R}_+^m}\prod_{k=1}^m \exp\left(-\frac{N_{\mathbf{K}_0}}{2a}x_k^{2a}+\eta_{0,k}\frac{x_k^b-1}{b}\right)h_j(x_j)x_i^{\mathrm{pow}_i}x_\ell^{\mathrm{pow}_\ell}x_j^{\mathrm{pow}_j}\,\mathrm{d}\boldsymbol{x}\\ =&\prod_{k\neq j}\int_0^\infty \exp\left(-\frac{N_{\mathbf{K}_0}}{2a}x_k^{2a}+\eta_{0,k}\frac{x_k^b-1}{b}\right)x_k^{\mathrm{pow}_k}\,\mathrm{d}x_k\\ &\qquad\times\int_0^\infty \exp\left(-\frac{N_{\mathbf{K}_0}}{2a}x_j^{2a}+\eta_{0,j}\frac{x_j^b-1}{b}\right)h_j(x_j)x_j^{\mathrm{pow}_j}\,\mathrm{d}x_j\end{aligned}$$

times a constant involving $\mathbf{K}$ and η_j, where $\mathrm{pow}_k \geq 0$ for each $k \neq j$. We have thus decomposed the integral into a product of univariate integrals. Note that

$$\int_0^\infty \exp\left(-\frac{N_{\mathbf{K}_0}}{2a}x_i^{2a}+\eta_{0,i}\frac{x_i^b-1}{b}\right)x_i^{\mathrm{pow}_i}\,\mathrm{d}x_i$$

is finite for all $\mathrm{pow}_i \geq 0$ regardless of whether b is nonzero, since we assumed $\mathbf{K}_0$ and $\boldsymbol{\eta}_0$ to lie in the parameter space with a finite normalizing constant. Indeed, if $b > 0$ then the terms in the exponential is a regular polynomial with positive degree and a negative leading term; if $b = 0$ then integrability follows from $\eta_{0,i} + \mathrm{pow}_i \geq \eta_{0,i} > -1$. Thus, we only need to consider the univariate integral that involve the x_j terms, namely

$$\int_0^\infty \exp\left(-\frac{N_{\mathbf{K}_0}}{2a}x_j^{2a}+\eta_{0,j}\frac{x_j^b-1}{b}\right)h_j(x_j)x_j^{\mathrm{pow}_j}\,\mathrm{d}x_j,$$

where pow_j takes value in $\{4a-2, 2a-2, 2b-2, 3a-2, a+b-2, 2a+b-2\} \subseteq [2\min\{a,b\}-2, 4a-2]$. We split the integral into two parts over $[0,1]$ and $[1,\infty]$, respectively.

- If $b > 0$, on $[0,1]$ the exponential part is bounded above and below by positive constants, and for (A1) we require $h_j(x) = o(x^{1-\min\{a,b\}})$ as $x \searrow 0^+$, so the integrand is $o(x^{\min\{a,b\}-1}) = o(x^{-1})$ and is thus integrable on $[0,1]$. The integrand on $[1,\infty)$ is integrable as in (A1) we assume h to grow at most polynomially.
- If $b = 0$, $\mathrm{pow}_j \in [-2, 4a-2]$ and the integrand becomes

  $$\exp(-N_{\mathbf{K}_0}x_j^{2a}/(2a))h_j(x_j)x_j^{\mathrm{pow}_j+\eta_{0,j}}.$$

  On $[0,1]$, (A1) requires $h_j(x) = o(x^{1-\min_j \eta_{0,j}})$, so $h_j(x_j)x_j^{\mathrm{pow}_j+\eta_{0,j}} = o(x^{-1})$ and the integrand is again integrable. Integrability on $[1,\infty)$ follows similarly to the case with $b > 0$.

Now consider the second part of (A2). By definition $\mathbb{E}_{p_0}\|(\nabla \log p(\boldsymbol{X}) \circ \boldsymbol{h}(\boldsymbol{X}))'\|_1$ equals

$$\int_{\mathbb{R}_+^m} p_0(\boldsymbol{x})\sum_{j=1}^m \left|h_j'(X_j)\partial_j \log p(\boldsymbol{X}) + h_j(X_j)\partial_j^2 \log p(\boldsymbol{X})\right|\mathrm{d}\boldsymbol{x}$$

$$\begin{aligned}\leq\sum_{j=1}^m\int_{\mathbb{R}_+^m}\exp\left(-\frac{N_{\mathbf{K}_0}}{2a}x_j^{2a}+\eta_{0,j}\frac{x_j^b-1}{b}\right)\Bigg|h_j'(x_j)\left(-\kappa_{jj}x_j^{2a-1}-\sum_{i\neq j}\kappa_{ji}x_i^a x_j^{a-1}+\eta_j x_j^{b-1}\right)\\ +h_j(x_j)\left(-\kappa_{jj}(2a-1)x_j^{2a-2}-\sum_{i\neq j}\kappa_{ji}(a-1)x_i^a x_j^{a-2}+(b-1)\eta_j x_j^{b-2}\right)\Bigg|\,\mathrm{d}\boldsymbol{x}.\end{aligned}$$

By the triangle inequality and the fact that $h_j \geq 0$ and $h_j' \geq 0$, similar to the proof for the first part, for each j the integral can be bounded by a sum of six integrals, each of the form

$$\text{const}\times\int_{\mathbb{R}_+^m}\prod_{k=1}^m\exp\left(-\frac{N_{\mathbf{K}_0}}{2a}x_k^{2a}+\eta_{0,k}\frac{x_k^b-1}{b}\right)h_j(x_j)x_i^{\mathrm{pow}_i}x_j^{\mathrm{pow}_j}\,\mathrm{d}\boldsymbol{x},$$

or with h_j replaced by h_j'. Finiteness thus follows from the same type of discussion by noting that $h_j(x) = o\left(x^{1-\min\{a,b\}}\right)$ and $h_j'(x) = o\left(x^{-\min\{a,b\}}\right)$.

We conclude that if the true and the proposed parameters give densities with finite normalizing constants, and if h satisfies assumption (A1), then (A2) is automatically satisfied.

In the centered case where we assume $\boldsymbol{\eta} \equiv \mathbf{0}$, we only need $\lim_{x_j\searrow 0^+} h_j(x_j)/x_j^{1-a} = 0$ as it is a special case with $b = 2a$. ■

A.4. Proof of Theorems in Section 6

Proof [Proof of Corollary 14] By Theorem 13, under assumptions in that theorem, the support of $\hat{\boldsymbol{\Psi}}$ is a subset of the true support of $\boldsymbol{\Psi}_0$, and $\|\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0\|_\infty \leq \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda$. Since $\boldsymbol{\Psi}_0$ has $|S_0|$ nonzero entries,

$$\|\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0\|_F = \left[\sum_{\boldsymbol{\Psi}_{0,jk}\neq 0}(\hat{\boldsymbol{\Psi}}_{jk} - \boldsymbol{\Psi}_{0,jk})^2\right]^{1/2} \leq \sqrt{|S_0|}\|\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0\|_\infty \leq \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda\sqrt{|S_0|}.$$

Similarly, by the definition of matrix ℓ_∞-ℓ_∞ norm,

$$|||\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0|||_2 \leq |||\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0|||_\infty = \max_{j=1,..,m}\sum_{k=1}^m |\hat{\boldsymbol{\Psi}}_{jk} - \boldsymbol{\Psi}_{0,jk}| \leq \frac{c_{\boldsymbol{\Gamma}_0}}{2-\alpha}\lambda d_{\boldsymbol{\Psi}_0}.$$

The result follows by also noting that $|||\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0|||_2 \leq |||\hat{\boldsymbol{\Psi}} - \boldsymbol{\Psi}_0|||_F$. ■

Proof [Proof of Theorem 15] The proof is based on Theorem 13 and a probabilistic bound on $\|\boldsymbol{\Gamma}_{\boldsymbol{\gamma}} - \boldsymbol{\Gamma}_0\|_\infty$, where in the case of centered Gaussian $\boldsymbol{\Gamma} = \mathrm{diag}(\mathbf{x}\mathbf{x}^\top, \ldots, \mathbf{x}\mathbf{x}^\top)$. Denote $\boldsymbol{\Sigma}_0 = \mathbf{K}_0^{-1}$. In particular, given $\tau > 2$ we wish to show that for $\epsilon = 80\sqrt{2}c_0\max_j(\Sigma_{0,jj})$, assuming $c_0 \equiv \sqrt{(\tau\log m + \log 4)/n} < 1/\sqrt{2}$,

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^n X_j^{(i)}X_k^{(i)} + \gamma_{1j}\mathbb{1}_{\{j=k\}} - \mathbb{E}X_jX_k\right| > \epsilon\right) \leq m^{2-\tau},$$

and so the results follow from Theorem 13.

By Lemma 1 of Ravikumar et al. (2011), since $X_j/\sqrt{\Sigma_{0,jj}}$ is Gaussian with mean 0 and standard deviation 1,

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} X_j^{(i)}X_k^{(i)} - \mathbb{E}X_jX_k\right| > t\right) \le 4\exp\left(-\frac{nt^2}{3200\max_j(\Sigma_{0,jj})^2}\right)$$

for $t \in (0, 40\max_j(\Sigma_{0,jj}))$. Denote the event as $\mathcal{E}_{j,k}(t)$. Note that $\mathbb{E}X_j^2 \le \max_j \Sigma_{0,jj} = \epsilon/(80\sqrt{2}c_0)$. Then letting $t = \epsilon/2$ and conditioning on the complement of $\mathcal{E}_{j,j}(\epsilon/2)$, we have

$$n^{-1}\sum_{i=1}^{n} {X_j^{(i)}}^2 \le \mathbb{E}X_j^2 + \epsilon/2 \le \frac{\epsilon}{2}\left(1 + \frac{1}{40\sqrt{2}c_0}\right).$$

Thus, choosing $\gamma_{\ell j} = (\delta-1)\sum_{i=1}^{n} {X_j^{(i)}}^2/n$ for $\ell = 1, \ldots, m$ ($\mathbf{\Gamma}$ has m identical blocks) with $1 < \delta < 1 + \left(1 + 1/(40\sqrt{2}c_0)\right)^{-1}$, by the triangle inequality and a union bound we have

$$\mathbb{P}\left(\max_{j,k}\left|n^{-1}\sum_{i=1}^{n} X_j^{(i)}X_k^{(i)} + \gamma_{1j}\mathbb{1}_{\{j=k\}} - \mathbb{E}X_jX_k\right| > \epsilon\right) \le \mathbb{P}(\mathcal{E}_{j,k}(\epsilon/2)) = m^{2-\tau}.$$

Since $\tau > 2$, it holds that $1 + \left(1 + 1/(40\sqrt{2}c_0)\right)^{-1} = 2 - (1 + 40\sqrt{2}c_0)^{-1}$ is larger than $2 - (1 + 80\sqrt{\log m/n})^{-1} \equiv C(n,m)$, so it is safe to choose any $\delta \in (1, C(n,m))$. Thus by the requirement on ϵ, the theorem statement holds when $n > \max(c^* c_1^2 d_{\mathbf{K}}^2, 2)(\tau\log m + \log 4)$ with $c^* = 12800\max_j(\Sigma_{0,jj})^2$. ■

Proof [Proof of Theorem 16] The proof of Theorem 13 from Lin et al. (2016) does not rely on the fact that the original $\mathbf{\Gamma}$ is an unbiased estimator for the population $\mathbf{\Gamma}_0$, but instead only requires one to bound $\|\mathbf{\Gamma} - \mathbf{\Gamma}_0\|_\infty$. Thus, for $\mathbf{\Gamma}_{\boldsymbol{\gamma}} = \mathbf{\Gamma} + \mathrm{diag}(\boldsymbol{\gamma})$, by Theorem 13 it suffices to prove that for any $\tau > 3$, we can bound $\|\mathbf{\Gamma}(\mathbf{x}) + \mathrm{diag}(\boldsymbol{\gamma}(\mathbf{x})) - \mathbf{\Gamma}_0\|_\infty$ by some ϵ_1 and $\|\boldsymbol{g}(\mathbf{x}) - \boldsymbol{g}_0\|_\infty$ by some ϵ_2, uniformly with probability $1 - m^{3-\tau}$. Recall from (21) that the j^{th} block of $\mathbf{\Gamma}_{\boldsymbol{\gamma}} \in \mathbb{R}^{m^2\times m^2}$ has (k,ℓ)-th entry

$$n^{-1}\sum_{i=1}^{n} X_k^{(i)}X_\ell^{(i)}h_j\left(X_j^{(i)}\right) + \gamma_{kj}\cdot\mathbb{1}_{\{k=\ell\}}.$$

The entry in $\boldsymbol{g} \in \mathbb{R}^{m^2}$ (obtained by linearizing a $m\times m$ matrix) corresponding to (j,k) is

$$n^{-1}\sum_{i=1}^{n} X_k^{(i)}h_j'\left(X_j^{(i)}\right) + n^{-1}\mathbb{1}_{\{j=k\}}\sum_{i=1}^{n} h_j\left(X_j^{(i)}\right).$$

Denote $M \equiv \max_j \sup_{x>0} h_j(x)$, $M' \equiv \max_j \sup_{x>0} h_j'(x)$, and $c_{\boldsymbol{X}} \equiv 2\max_j(2\sqrt{\Sigma_{jj}} + \sqrt{e}\,\mathbb{E}_0 X_j)$. Using results for sub-Gaussian random variables from Lemma 22.2 in Appendix B, we have for any $t_1 > 0$,

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} X_k^{(i)}X_\ell^{(i)}h_j\left(X_j^{(i)}\right) - \mathbb{E}_0X_kX_\ell h_j(X_j)\right| > t_1\right) \le 2\exp\left(-\min\left(\frac{nt_1^2}{2M^2c_{\boldsymbol{X}}^4}, \frac{nt_1}{2Mc_{\boldsymbol{X}}^2}\right)\right).$$

Thus, choosing $\epsilon_1 \equiv 2Mc_{\boldsymbol{X}}^2 c_{n,m}$, where $c_{n,m} \equiv \max\left\{\frac{2(\log m^\tau + \log 6)}{n}, \sqrt{\frac{2(\log m^\tau+\log 6)}{n}}\right\}$, for $\gamma_{kj} \le \epsilon_1/2$, we have

$$\begin{aligned}
&\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} X_k^{(i)}X_\ell^{(i)}h_j\left(X_j^{(i)}\right) + \gamma_{kj}\mathbb{1}_{\{k=\ell\}} - \mathbb{E}_0X_kX_\ell h_j(X_j)\right| > \epsilon_1\right)\\
&\le \mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} X_k^{(i)}X_\ell^{(i)}h_j\left(X_j^{(i)}\right) - \mathbb{E}_0X_kX_\ell h_j(X_j)\right| > \epsilon_1/2\right) \qquad (39)\\
&\le 2\exp\left(-\min\left(\frac{n\epsilon_1^2}{8M^2c_{\boldsymbol{X}}^4}, \frac{n\epsilon_1}{4Mc_{\boldsymbol{X}}^2}\right)\right) \le \frac{1}{3m^\tau}. \qquad (40)
\end{aligned}$$

Denote the event inside the probability in (39) as $\mathcal{E}_{k,\ell,j}(\epsilon_1/2)$.
By definition,

$$c_{\boldsymbol{X}}^2 = 4\max_k\left(4\Sigma_{kk} + 4\sqrt{e}\sqrt{\Sigma_{kk}}\,\mathbb{E}_0X_k + e(\mathbb{E}_0X_k)^2\right) \ge 4e\max_k\left(\Sigma_{kk} + (\mathbb{E}_0X_k)^2\right).$$

By Lemmas 21.2 and 22.1 from Appendix B, $\mathrm{var}(X_k) \le \Sigma_{kk}$, so $c_{\boldsymbol{X}}^2 \ge 4e\max_k \mathbb{E}_0X_k^2 \ge 4e\mathbb{E}_0X_k^2h_j(X_j)/M$. Thus, setting $\epsilon_1 = 2Mc_{\boldsymbol{X}}^2c_{n,m}$, on the complement of $\mathcal{E}_{k,k,j}(\epsilon_1/2)$ we have

$$n^{-1}\sum_{i=1}^{n} {X_k^{(i)}}^2h_j\left(X_j^{(i)}\right) \le \mathbb{E}_0X_k^2h_j(X_j) + \epsilon_1/2 \le \frac{\epsilon_1}{2}\left(1 + \frac{1}{4ec_{n,m}}\right).$$

Then

$$\frac{1}{1 + 1/(4ec_{n,m})}\frac{1}{n}\sum_{i=1}^{n} {X_k^{(i)}}^2h_j\left(X_j^{(i)}\right) \le \epsilon_1/2 \qquad (41)$$

on the complement of $\mathcal{E}_{k,k,j}(\epsilon_1/2)$, again with $c_{n,m} \equiv \max\left\{\frac{2(\log m^\tau+\log 6)}{n}, \sqrt{\frac{2(\log m^\tau+\log 6)}{n}}\right\}$.
Note that the multiplier on the left of (41) is increasing in $c_{n,m}$, and that $2(\log m^\tau + \log 6) > 6\log m$ by the assumption that $\tau > 3$. Thus, if we let

$$\gamma_{kj} \equiv \frac{1}{1 + 1/\left(4e\max\left\{6\log m/n, \sqrt{6\log m/n}\right\}\right)}\frac{1}{n}\sum_{i=1}^{n} {X_k^{(i)}}^2h_j\left(X_j^{(i)}\right),$$

which is just a constant multiple of the (k,k)-th entry of $\mathbf{\Gamma}_j$ itself, with the constant explicitly calculable and a function of p and n only, then for $k = \ell$

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} X_k^{(i)}X_\ell^{(i)}h_j\left(X_j^{(i)}\right) + \gamma_{kj} - \mathbb{E}_0X_kX_\ell h_j(X_j)\right| \ge \epsilon_1\right) \le \mathbb{P}(\mathcal{E}_{k,\ell,j}(\epsilon_1/2)) \le \frac{1}{3m^\tau}.$$

Since this also holds for $k \ne \ell$ without the γ_{kj} term, by a union bound over m^3 events,

$$\mathbb{P}\left(\max_{j,k,\ell}\left|n^{-1}\sum_{i=1}^{n} X_k^{(i)}X_\ell^{(i)}h_j\left(X_j^{(i)}\right) + \gamma_{kj}\mathbb{1}_{\{k=\ell\}} - \mathbb{E}_0X_kX_\ell h_j(X_j)\right| \ge \epsilon_1\right) \le \frac{1}{3m^{\tau-3}}. \qquad (42)$$

Now, on the other hand, Lemma 22.1 and Hoeffding's inequality give for any $t_{2,1}, t_{2,2} > 0$ that

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} X_k^{(i)} h_j'\left(X_j^{(i)}\right) - \mathbb{E}_0 X_k h_j'(X_j)\right| \geq t_{2,1}\right) \leq 2\exp\left(-\frac{nt_{2,1}^2}{2M'^2 c_{\boldsymbol{X}}^2}\right),$$
$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} h_j\left(X_j^{(i)}\right) - \mathbb{E}_0 h_j(X_j)\right| \geq t_{2,2}\right) \leq 2\exp\left(-2nt_{2,2}^2/M^2\right).$$

Choosing $\epsilon_{2,1} \equiv \sqrt{2}M' c_{\boldsymbol{X}}\sqrt{\frac{\log m^{\tau-1}+\log 6}{n}}$, $\epsilon_{2,2} \equiv M\sqrt{\frac{\log m^{\tau-2}+\log 6}{2n}}$ and taking union bounds over m^2, and m events, respectively, we have

$$\mathbb{P}\left(\max_{j,k}\left|n^{-1}\sum_{i=1}^{n} X_k^{(i)} h_j'\left(X_j^{(i)}\right) - \mathbb{E}_0 X_k h_j'(X_j)\right| \geq \epsilon_{2,1}\right) \leq \frac{1}{3m^{\tau-3}}, \tag{43}$$
$$\mathbb{P}\left(\max_{j}\left|n^{-1}\sum_{i=1}^{n} h_j\left(X_j^{(i)}\right) - \mathbb{E}_0 h_j(X_j)\right| \geq \epsilon_{2,2}\right) \leq \frac{1}{3m^{\tau-3}}. \tag{44}$$

Hence, by (42) (43) (44), with probability at least $1 - m^{3-\tau}$, $\|\boldsymbol{\Gamma}_{\boldsymbol{\gamma}}(\mathbf{x}) - \boldsymbol{\Gamma}_0\|_\infty < \epsilon_1$ and $\|\boldsymbol{g}(\mathbf{x}) - \boldsymbol{g}_0\|_\infty < \epsilon_2 \equiv \epsilon_{2,1} + \epsilon_{2,2}$. Consider any $\tau > 3$, and let

$$\begin{aligned}
c_2 &\equiv \frac{6}{\alpha} c_{\boldsymbol{\Gamma}_0}, \\
n &> \max\{2M^2 c_{\boldsymbol{X}}^4 c_2^2 d_{\mathbf{K}_0}^2(\tau\log m + \log 6), 2Mc_{\boldsymbol{X}}^2 c_2 d_{\mathbf{K}_0}(\tau\log m + \log 6)\}, \\
\lambda &> \frac{3(2-\alpha)}{\alpha}\max\{c_{\mathbf{K}_0}\epsilon_1, \epsilon_2\} \\
&\equiv \frac{3(2-\alpha)}{\alpha}\max\left\{4Mc_{\mathbf{K}_0}c_{\boldsymbol{X}}^2\frac{(\log m^\tau + \log 6)}{n},\right. \\
&\quad \left. 2Mc_{\mathbf{K}_0}c_{\boldsymbol{X}}^2\sqrt{\frac{2(\log m^\tau + \log 6)}{n}}, \sqrt{2}M' c_{\boldsymbol{X}}\sqrt{\frac{\log m^{\tau-1}+\log 6}{n}} + M\sqrt{\frac{\log m^{\tau-2}+\log 6}{2n}}\right\}.
\end{aligned}$$

Then $d_{\mathbf{K}_0}\epsilon_1 \leq \alpha/(6c_{\boldsymbol{\Gamma}_0})$ and the results follow from Theorem 13. ■

Proof [Proof of Theorem 17] Similar to the proof of Theorem 16, by Theorem 13 it suffices to prove that for any $\tau > 3$, we can bound $\|\boldsymbol{\Gamma}_{\boldsymbol{\gamma}}(\mathbf{x}) - \boldsymbol{\Gamma}_0\|_\infty$ by some ϵ_1 and $\|\boldsymbol{g}(\mathbf{x}) - \boldsymbol{g}_0\|_\infty$ by some ϵ_2, uniformly with probability $1 - m^{3-\tau}$. Recall that $\boldsymbol{\Gamma} \in \mathbb{R}^{(m^2+m)\times(m^2+m)}$ is a rearrangement of $\boldsymbol{\Gamma}^{(*)}$, which is in turn formed by $\boldsymbol{\Gamma}_{11} \in \mathbb{R}^{m^2\times m^2}$, $\boldsymbol{\Gamma}_{12} \in \mathbb{R}^{m^2\times m}$, $\boldsymbol{\Gamma}_{12}^\top$ and $\boldsymbol{\Gamma}_{22} \in \mathbb{R}^{m\times m}$, all of which are block-diagonal with m blocks.
The j^{th} block of $\boldsymbol{\Gamma}_{11} \in \mathbb{R}^{m^2\times m^2}$ has (k,ℓ)-th entry

$$n^{-1}\sum_{i=1}^{n} X_k^{(i)} X_\ell^{(i)} h_j\left(X_j^{(i)}\right),$$

the k^{th} entry in the j^{th} block of $\boldsymbol{\Gamma}_{12}$ is

$$-n^{-1}\sum_{i=1}^{n} X_k^{(i)} h_j\left(X_j^{(i)}\right),$$

the j^{th} diagonal entry of $\boldsymbol{\Gamma}_{22}$ is

$$n^{-1}\sum_{i=1}^{n} h_j\left(X_j^{(i)}\right).$$

On the other hand, $\boldsymbol{g} \in \mathbb{R}^{(m^2+m)}$ is a rearrangement of $\boldsymbol{g}^{(*)} \equiv [\boldsymbol{g}_1^\top, \boldsymbol{g}_2^\top]^\top$, where the entry in $\boldsymbol{g}_1 \in \mathbb{R}^{m^2}$ (obtained by linearizing a $m \times m$ matrix) corresponding to (j,k), is

$$n^{-1}\sum_{i=1}^{n} X_k^{(i)} h_j'\left(X_j^{(i)}\right) + n^{-1}\mathbb{1}_{\{j=k\}}\sum_{i=1}^{n} h_j\left(X_j^{(i)}\right),$$

while the j-th component of $\boldsymbol{g}_2 \in \mathbb{R}^m$ is

$$-n^{-1}\sum_{i=1}^{n} h_j'\left(X_j^{(i)}\right).$$

Recalling that the bounds in Lemma 22 also hold when $\boldsymbol{\mu} \neq 0$, we may then use bounds similar to those in the proof of Theorem 16, and use union bounds to arrive at analogous consistency results, modulus different constants. The amplifiers $\boldsymbol{\gamma}$ can be incorporated analogously. ■

Appendix B. Auxiliary Lemmas and Definitions

In this appendix, to simplify notation, when it is clear from the context, the operator $\mathbb{E}$ is defined as the expectation under the true distribution, unless otherwise noted.

Definition 20 (Sub-Gaussian and Sub-Exponential Variables)
The sub-Gaussian ($r = 2$) *and* sub-exponential ($r = 1$) norms *of a random variable are*

$$\|X\|_{\psi_r} \equiv \sup_{q\geq 1} q^{-1/r}(\mathbb{E}|X|^{rq})^{1/(rq)} \equiv \sup_{q\geq 1} q^{-1/r}\|X\|_{rq}.$$

If $\|X\|_{\psi_2} < \infty$ *we say* X *is* sub-Gaussian*; if* $\|X\|_{\psi_1} < \infty$ *we call* X sub-exponential. *For a* zero-mean *sub-Gaussian random variable* X *also define the* sub-Gaussian parameter

$$\tau(X) = \inf\{\tau \geq 0 : \mathbb{E}\exp(tX) \leq \exp(\tau^2 t^2/2),\ \forall t \in \mathbb{R}\}.$$

The definition of sub-Gaussian norm here allows for a non-centered variable and differs from the one in Vershynin (2012), which uses $\|X\|_q$. Instead, it coincides with θ_2 in Buldygin and Kozachenko (2000). The sub-Gaussian parameter is defined as in Buldygin and Kozachenko (2000) and the sub-exponential norm as in Vershynin (2012).

Lemma 21 (Properties of Sub-Gaussian and Sub-Exponential Variables)

1) For any X *and* $r = 1, 2$, $\|X - \mathbb{E}X\|_{\psi_r} \leq 2\|X\|_{\psi_r}$ *and* $\|X\|_{\psi_r} \leq \|X - \mathbb{E}X\|_{\psi_r} + |\mathbb{E}X|$, *as long as the expectation and norms are finite.*

2) (Buldygin and Kozachenko, 2000) $\tau(X)$ is a norm on the space of all zero-mean sub-Gaussian variables; so $\tau(X+Y) \le \tau(X) + \tau(Y)$. If X is zero-mean sub-Gaussian, then $\mathrm{var}(X) \le \tau^2(X)$, $\|X\|_{\psi_2} \le 2\tau(X)/\sqrt{e}$, $\tau(X) \le \sqrt{e}\|X\|_{\psi_2}$. If $X_1, \dots, X_n$ are i.i.d. zero-mean sub-Gaussian, $\tau\left(n^{-1}\sum_{i=1}^n X_i\right) \le n^{-1/2}\tau(X_i)$.

3) If X_1 and X_2 are sub-Gaussian (not necessarily independent) with $\|X_1\|_{\psi_2} \le K_1$ and $\|X_2\|_{\psi_2} \le K_2$, then X_1X_2 is sub-exponential with $\|X_1X_2\|_{\psi_1} \le K_1K_2$.

4) (Buldygin and Kozachenko, 2000) If X is zero-mean sub-Gaussian and $q > 0$, then

$$\mathbb{E}|X|^q \le 2(q/e)^{q/2}\tau^q(X).$$

5) (Buldygin and Kozachenko, 2000) If $X_1, \dots, X_n$ are independent zero-mean, sub-Gaussian variables, then for any $\epsilon > 0$,

$$\mathbb{P}(|X_1| \ge \epsilon) \le 2\exp\left(-\frac{\epsilon^2}{2\tau^2(X_1)}\right),$$
$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^n X_i\right| > \epsilon\right) \le 2\exp\left(-\frac{n\epsilon^2}{2\max_i \tau^2(X_i)}\right).$$

6) (Vershynin, 2012) If $X_1, \dots, X_n$ are independent zero-mean sub-exponential random variables with $K \ge \max_i \|X_i\|_{\psi_1}$, then for any $\epsilon > 0$,

$$\mathbb{P}(|X_1| \ge \epsilon) \le 2\exp\left(-\min\left(\frac{\epsilon^2}{8e^2K^2}, \frac{\epsilon}{4eK}\right)\right),$$
$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^n X_i\right| \ge \epsilon\right) \le 2\exp\left(-\min\left(\frac{n\epsilon^2}{8e^2K^2}, \frac{n\epsilon}{4eK}\right)\right).$$

7) (Boucheron et al., 2013) If for X_i i.i.d. there exists some $B > 0$ such that

$$\sup_{q\ge 2}\left(\frac{\mathbb{E}|X|^q}{q!}\right)^{1/q} \le B/2$$

then for all $\epsilon > 0$,

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^n (X_i - \mathbb{E}X_i)\right| \ge \epsilon\right) \le 2\exp\left(-\min\left(\frac{n\epsilon^2}{2B^2}, \frac{n\epsilon}{2B}\right)\right).$$

Proof

1) For $r = 1, 2$, by the triangle inequality, $\|X - \mathbb{E}X\|_{\psi_r} \le \|X\|_{\psi_r} + \|\mathbb{E}X\|_{\psi_r} = \|X\|_{\psi_r} + |\mathbb{E}X| \le \|X\|_{\psi_r} + \mathbb{E}|X| \le 2\|X\|_{\psi_r}$, where in the last step we used the definition of $\|\cdot\|_{\psi_r}$ with $q = 1$ for $r = 1$ and $\mathbb{E}|X| \le (\mathbb{E}|X|^2)^{1/2}$ with $q = 2$ for $r = 2$. On the other hand, $\|X\|_{\psi_r} \le \|X - \mathbb{E}X\|_{\psi_r} + \|\mathbb{E}X\|_{\psi_r} = \|X - \mathbb{E}X\|_{\psi_r} + |\mathbb{E}X|$.

2) These follow from Theorems 1.2 and 1.3 and Lemmas 1.2 and 1.7 from Buldygin and Kozachenko (2000), and $\sqrt[4]{3.1}e^{9/16}/\sqrt{2} \approx 1.6467 \le 1.6487 \approx \sqrt{e}$.

3) By Hölder's inequality (or Cauchy-Schwarz),

$$\begin{aligned}\|X_1X_2\|_{\psi_1} &= \sup_{q\ge1} q^{-1}(\mathbb{E}|X_1X_2|^q)^{1/q} = \sup_{q\ge1} q^{-1}(\mathbb{E}|X_1^qX_2^q|)^{1/q}\\ &\le \sup_{q\ge1} q^{-1}\left[(\mathbb{E}|X_1|^{2q})^{1/2}(\mathbb{E}|X_2|^{2q})^{1/2}\right]^{1/q}\\ &\le \sup_{q\ge1}\left[q^{-1/2}(\mathbb{E}|X_1|^{2q})^{1/2q}\right]\sup_{q\ge1}\left[q^{-1/2}(\mathbb{E}|X_2|^{2q})^{1/2q}\right]\\ &= \|X_1\|_{\psi_2}\|X_2\|_{\psi_2} \le K_1K_2.\end{aligned}$$

4-6) These are Lemma 1.4 and Theorem 1.5 in Buldygin and Kozachenko (2000), and a consequence of Corollary 5.17 in Vershynin (2012).

7) By Theorem 2.10 of Boucheron et al. (2013) wherein we let $v \equiv nB^2/2$ and $c \equiv B/2$, we have

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^n (X_i - \mathbb{E}X_i)\right| \ge \epsilon\right) \le 2\exp\left(-\frac{n\epsilon^2}{B^2 + B\epsilon}\right)$$

for all $\epsilon > 0$. (Theorem 2.10 gives an one-sided bound; bound for the other side is obtained by taking $X_i = -X_i$). The inequality follows by splitting into cases $\epsilon \le B$ and $\epsilon > B$. ■

Lemma 22 *Suppose $\boldsymbol{X}$ follows a truncated normal distribution on $\mathbb{R}^m_+$ with parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma} = \mathbf{K}^{-1} \succ \mathbf{0}$. Let $\boldsymbol{X}^{(1)}, \dots, \boldsymbol{X}^{(n)}$ be i.i.d. copies of $\boldsymbol{X}$, with j-th component of the i-th copy being $X_j^{(i)}$. Then*

1. *For $j = 1, \dots, p$, $\tau(X_j - \mathbb{E}X_j) \le \sqrt{\Sigma_{jj}}$. That is, the sub-Gaussian parameter of any marginal distribution of $\boldsymbol{X}$, after centering, is bounded by the square root of its corresponding diagonal entry in the covariance parameter $\boldsymbol{\Sigma}$. Then, for any $\epsilon > 0$,*

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^n X_j^{(i)} - \mathbb{E}X_j\right| > \epsilon\right) \le 2\exp\left(-\frac{n\epsilon^2}{2\Sigma_{jj}}\right).$$

In particular, if h_0 is a function bounded by M_0, then for any $\epsilon > 0$,

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^n X_j^{(i)}h_0\left(X_k^{(i)}\right) - \mathbb{E}X_jh_0(X_k)\right| \ge \epsilon\right) \le 2\exp\left(-\frac{n\epsilon^2}{8M_0^2(2\sqrt{\Sigma_{jj}} + \sqrt{e}\,\mathbb{E}X_j)^2}\right),$$
$$\tau\left(n^{-1}\sum_{i=1}^n X_j^{(i)}h_0\left(X_k^{(i)}\right) - \mathbb{E}X_jh_0(X_k)\right) \le \frac{2M_0}{\sqrt{n}}\left(2\sqrt{\Sigma_{jj}} + \sqrt{e}\,\mathbb{E}X_j\right),$$
$$\left\|n^{-1}\sum_{i=1}^n X_j^{(i)}h_0\left(X_k^{(i)}\right) - \mathbb{E}X_jh_0(X_k)\right\|_{\psi_2} \le \frac{4M_0}{\sqrt{en}}\left(2\sqrt{\Sigma_{jj}} + \sqrt{e}\,\mathbb{E}X_j\right).$$

2. *For $j, k, \ell \in \{1, \dots, p\}$, if h_0 is a function bounded by M_0, then*

$$\|X_jX_kh_0(X_\ell) - \mathbb{E}X_jX_kh_0(X_\ell)\|_{\psi_1} \le \frac{M_0}{2e}c_{\boldsymbol{X}}^2, \tag{45}$$

where $c_{\boldsymbol{X}} \equiv 2\max_j(2\sqrt{\Sigma_{jj}} + \sqrt{e}\,\mathbb{E}X_j)$. *In particular, for any* $\epsilon > 0$,

$$\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} X_j^{(i)}X_k^{(i)}h_0\left(X_\ell^{(i)}\right) - \mathbb{E}X_jX_kh_0(X_\ell)\right| > \epsilon\right) \leq 2\exp\left(-\min\left(\frac{n\epsilon^2}{2M_0^2c_{\boldsymbol{X}}^4}, \frac{n\epsilon}{2M_0c_{\boldsymbol{X}}^2}\right)\right).$$

Proof [Proof of Lemma 22]
1. Without loss of generality choose $j = 1$. By the definition of sub-Gaussian parameters, we need to show that for all $t \in \mathbb{R}$,

$$\mathbb{E}\exp(tX_1) \leq \exp\left(t^2\Sigma_{11}/2 + t\mathbb{E}X_1\right),$$

which is equivalent to

$$t^2\Sigma_{11}/2 + t\mathbb{E}X_1 - \log\mathbb{E}\exp(tX_1) \geq 0 \quad \forall t \in \mathbb{R}. \tag{46}$$

Since the left-hand side of (46) equals 0 at $t = 0$, it suffices to show that its derivative,

$$t\Sigma_{11} + \mathbb{E}X_1 - \frac{\mathrm{d}\log\mathbb{E}\exp(tX_1)}{\mathrm{d}t} = t\Sigma_{11} + \mathbb{E}X_1 - \frac{\frac{\mathrm{d}\mathbb{E}\exp(tX_1)}{\mathrm{d}t}}{\mathbb{E}\exp(tX_1)}, \tag{47}$$

is non-negative on $(0, \infty)$ and non-positive on $(-\infty, 0)$. By properties of moment-generating functions, $\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}\exp(tX_1)$ evaluated at $t = 0$ equals $\mathbb{E}X_1$, so (47) equals 0 at $t = 0$. It in turn suffices to show the derivative of (47), namely

$$\Sigma_{11} - \frac{\mathrm{d}^2\log\mathbb{E}\exp(tX_1)}{\mathrm{d}t^2} \tag{48}$$

is non-negative in $t \in \mathbb{R}$.

Given any vector $\boldsymbol{v} \in \mathbb{R}^p$, define $\mathbb{R}^p_+ - \boldsymbol{v} \equiv \{\boldsymbol{u} - \boldsymbol{v} : \boldsymbol{u} \in \mathbb{R}^p_+\}$. By Tallis (1961), denoting the first column of $\boldsymbol{\Sigma}$ as $\boldsymbol{\Sigma}_1$, the moment-generating function of the marginal distribution of X_1 is

$$\frac{\int_{\mathbb{R}^p_+ - (\boldsymbol{\mu} + t\boldsymbol{\Sigma}_1)} \exp\left(-\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)\mathrm{d}\boldsymbol{x}}{\int_{\mathbb{R}^p_+ - \boldsymbol{\mu}} \exp\left(-\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)\mathrm{d}\boldsymbol{x}} \exp\left(t\mu_1 + \frac{1}{2}t^2\Sigma_{11}\right).$$

(48) thus becomes

$$-\frac{\mathrm{d}^2}{\mathrm{d}t^2}\log\int_{\mathbb{R}^p_+ - (\boldsymbol{\mu} + t\boldsymbol{\Sigma}_1)} \exp\left(-\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)\mathrm{d}\boldsymbol{x}.$$

Showing this is non-negative in $t \in \mathbb{R}$ is equivalent to showing that the integral itself is log-concave in t. But

$$\int_{\mathbb{R}^p_+ - (\boldsymbol{\mu} + t\boldsymbol{\Sigma}_1)} \exp\left(-\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)\mathrm{d}\boldsymbol{x} = \int_{\mathbb{R}^p} \exp\left(-\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)\mathbb{1}_{\mathbb{R}^p_+ - \boldsymbol{\mu}}(\boldsymbol{x} + t\boldsymbol{\Sigma}_1)\,\mathrm{d}\boldsymbol{x}$$

with $\exp\left(-\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\right)$ log-concave in $\boldsymbol{x}$ and $\mathbb{1}_{\mathbb{R}^p_+ - \boldsymbol{\mu}}(\boldsymbol{x} + t\boldsymbol{\Sigma}_1)$ log-concave in $(\boldsymbol{x}, t)$ since $\mathbb{R}^p_+ - \boldsymbol{\mu}$ is a convex set. Since log-concavity is closed under multiplication and integration

over $\mathbb{R}^p$, the integral is indeed log-concave, and our proof of the bound on the sub-Gaussian parameter of $X_j - \mathbb{E}X_j$ is complete. The tail bound follows from 5) of Lemma 21.

Now by 1) and 2) of Lemma 21,

$$\|X_j\|_{\psi_2} \leq 2\sqrt{\Sigma_{jj}/e} + \mathbb{E}X_j.$$

If h_0 is a function bounded by M_0, then by definition

$$\|X_jh_0(X_k)\|_{\psi_2} \leq M_0\left(2\sqrt{\Sigma_{jj}/e} + \mathbb{E}X_j\right).$$

By 1) and 2) of Lemma 21 again,

$$\begin{aligned}\tau(X_jh_0(X_k) - \mathbb{E}X_jh_0(X_k)) &\leq \sqrt{e}\|X_jh_0(X_k) - \mathbb{E}X_jh_0(X_k)\|_{\psi_2} \\ &\leq 2\sqrt{e}\|X_jh_0(X_k)\|_{\psi_2} \\ &\leq 2M_0(2\sqrt{\Sigma_{jj}} + \sqrt{e}\,\mathbb{E}X_j).\end{aligned}$$

The tail bound thus follows from the first inequality using 5) of Lemma 21. By 2) of the Lemma 21,

$$\tau\left(n^{-1}\sum_{i=1}^{n} X_j^{(i)}h_0\left(X_k^{(i)}\right) - \mathbb{E}X_jh_0(X_k)\right) \leq \frac{2M_0}{\sqrt{n}}\left(2\sqrt{\Sigma_{jj}} + \sqrt{e}\,\mathbb{E}X_j\right),$$
$$\left\|n^{-1}\sum_{i=1}^{n} X_j^{(i)}h_0\left(X_k^{(i)}\right) - \mathbb{E}X_jh_0(X_k)\right\|_{\psi_2} \leq \frac{4M_0}{\sqrt{en}}\left(2\sqrt{\Sigma_{jj}} + \sqrt{e}\,\mathbb{E}X_j\right).$$

2. By the proof of 1) of this lemma, $\|X_j\|_{\psi_2} \leq 2\sqrt{\Sigma_{jj}/e} + \mathbb{E}X_j$, and by 3) of Lemma 21,

$$\|X_jX_k\|_{\psi_1} \leq \left(2\sqrt{\Sigma_{jj}/e} + \mathbb{E}X_j\right)\left(2\sqrt{\Sigma_{kk}/e} + \mathbb{E}X_k\right) \leq \max_j\left(2\sqrt{\Sigma_{jj}/e} + \mathbb{E}X_j\right)^2.$$

Since h_0 is a function bounded by M_0, by definition

$$\|X_jX_kh_0(X_\ell)\|_{\psi_1} \leq M_0\max_j\left(2\sqrt{\Sigma_{jj}/e} + \mathbb{E}X_j\right)^2.$$

Then by 1) of Lemma 21 again,

$$\|X_jX_kh_0(X_\ell) - \mathbb{E}X_jX_kh_0(X_\ell)\|_{\psi_1} \leq 2M_0\max_j\left(2\sqrt{\Sigma_{jj}/e} + \mathbb{E}X_j\right)^2.$$

The tail bound then follows from 6) of Lemma 21. ■

Although not used for our consistency results, in the special case of $h_0 \equiv 1$, we also have the following lemma. The notable difference between bounds (49) below and (45) from Lemma 22.2 is in the constants and dependency on $\mathbb{E}X_j$: The constants in the denominator in the right-hand side of (45) is smaller and thus gives a tighter bound, but (49) is preferred when $\mathbb{E}X_j$ is notably large compared to $\sqrt{\Sigma_{jj}}$, since the constant is only linear in $\mathbb{E}X_j$.

Lemma 23 *Consider the setting in Lemma 22. Then for $j,k \in \{1,\dots,p\}$, for any $\epsilon > 0$,*

$$\mathbb{P}\left(n^{-1}\left|\sum_{i=1}^{n} X_j^{(i)} X_k^{(i)} - \mathbb{E}X_j X_k\right| \geq \epsilon\right) \leq 4\exp\left(-\min\left(\frac{2n\epsilon^2}{C_1^2}, \frac{n\epsilon}{C_1}\right)\right), \tag{49}$$

where $C_1 \equiv 91\max_j \Sigma_{jj} + 72\max_j \mathbb{E}X_j \max_j \sqrt{\Sigma_{jj}}$.

Proof [Proof of Lemma 23] We use a proof similar to Lemma 1 in Ravikumar et al. (2011) (note that $\mathbb{E}X_j$ may be nonzero in our case). Define

$$U_{jk}^{(i)} \equiv X_j^{(i)} + X_k^{(i)}, \quad U_{jk} \equiv X_j + X_k, \quad V_{jk}^{(i)} \equiv X_j^{(i)} - X_k^{(i)}, \quad V_{jk} \equiv X_j - X_k.$$

Since $X_j^{(i)} X_k^{(i)} = \frac{1}{4}\left({U_{jk}^{(i)}}^2 - {V_{jk}^{(i)}}^2\right)$, by union bound we have

$$\begin{aligned}
&\mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} X_j^{(i)} X_k^{(i)} - \mathbb{E}X_j X_k\right| \geq \epsilon\right) \\
&\leq \mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} {U_{jk}^{(i)}}^2 - \mathbb{E}U_{jk}^2\right| \geq 2\epsilon\right) + \mathbb{P}\left(\left|n^{-1}\sum_{i=1}^{n} {V_{jk}^{(i)}}^2 - \mathbb{E}V_{jk}^2\right| \geq 2\epsilon\right).
\end{aligned}$$

We next define

$$\begin{aligned}
&Z_{jk}^{(i)} \equiv {U_{jk}^{(i)}}^2 - \mathbb{E}U_{jk}^2 = {A_{jk}^{(i)}}^2 + B_{jk}^{(i)} + C_{jk}, && \overline{X}_j^{(i)} \equiv X_j^{(i)} - \mathbb{E}X_j, \\
&A_{jk}^{(i)} \equiv \overline{X}_j^{(i)} + \overline{X}_k^{(i)}, \quad B_{jk}^{(i)} \equiv 2(\mathbb{E}X_j + \mathbb{E}X_k)(\overline{X}_j^{(i)} + \overline{X}_k^{(i)}), && C_{jk} \equiv -\mathbb{E}(\overline{X}_j^{(i)} + \overline{X}_k^{(i)})^2.
\end{aligned}$$

Then since τ is a norm by 2) of Lemma 21, A_{jk} is sub-Gaussian with parameter $\leq \sqrt{\Sigma_{jj}} + \sqrt{\Sigma_{kk}}$, and B_{jk} is sub-Gaussian with parameter $\leq 2(\mathbb{E}X_j + \mathbb{E}X_k)\left(\sqrt{\Sigma_{jj}} + \sqrt{\Sigma_{kk}}\right)$. Using 4) of Lemma 21 together with the inequality $(a+b+c)^q \leq (3\max\{a,b,c\})^q \leq 3^q(a^q+b^q+c^q)$ for all $a,b,c \geq 0$ and $q > 0$, we have for any $q \geq 2$

$$\begin{aligned}
(\mathbb{E}|Z_{jk}|^q)^{1/q} &\leq \left(3^q\left(\mathbb{E}|A_{jk}|^{2q} + \mathbb{E}|B_{jk}|^q + |C_{jk}|^q\right)\right)^{1/q} \\
&\leq 3^{1+1/q}\left((\mathbb{E}|A_{jk}|^{2q})^{1/q} + (\mathbb{E}|B_{jk}|^q)^{1/q} + |C_{jk}|\right) \\
&\leq 3^{1+1/q}\left(2^{1/q}(2q/e)\left(\sqrt{\Sigma_{jj}} + \sqrt{\Sigma_{kk}}\right)^2\right. \\
&\qquad \left. + 2^{1/q}\sqrt{q/e}\,2(\mathbb{E}X_j + \mathbb{E}X_k)(\sqrt{\Sigma_{jj}} + \sqrt{\Sigma_{kk}}) + \mathrm{var}(X_j + X_k)\right).
\end{aligned}$$

Using $\mathrm{var}(X+Y) \leq 2(\mathrm{var}(X) + \mathrm{var}(Y))$ and the fact that $\mathrm{var}(X_j) = \mathrm{var}(X_j - \mathbb{E}X_j) \leq \tau^2(X_j - \mathbb{E}X_j) \leq \Sigma_{jj}$ (by 2) of Lemma 21 and 1) of Lemma 22, we then have

$$\begin{aligned}
&\left(\frac{\mathbb{E}|Z_{jk}|^q}{q!}\right)^{1/q} \\
&\quad \leq 3^{1+1/q}\frac{2^{3+1/q}(q/e)\max_j \Sigma_{jj} + 2^{3+1/q}\sqrt{q/e}\max_j \mathbb{E}X_j \cdot \max_j \sqrt{\Sigma_{jj}} + 4\max_j \Sigma_{jj}}{(q!)^{1/q}}.
\end{aligned}$$

Since all three coefficients involving q are decreasing in $q \geq 2$, we have

$$\sup_{q\geq 2}\left(\frac{\mathbb{E}|Z_{jk}|^q}{q!}\right)^{1/q} \leq \left(48\sqrt{3}/e + 6\sqrt{6}\right)\max_j \Sigma_{jj} + 24\sqrt{6/e}\max_j \mathbb{E}X_j \max_j \sqrt{\Sigma_{jj}}.$$

Thus by 7) of Lemma 21, letting $B \equiv \left(91\max_j \Sigma_{jj} + 72\max_j \mathbb{E}X_j \max_j \sqrt{\Sigma_{jj}}\right)$, we have for all $\epsilon > 0$:

$$\mathbb{P}\left(n^{-1}\left|\sum_{i=1}^{n} Z_{jk}^{(i)}\right| \geq 2\epsilon\right) \leq 2\exp\left(-\min\left(\frac{2n\epsilon^2}{B^2}, \frac{n\epsilon}{B}\right)\right).$$

A tail bound for the sample average of V_{jk}^2 can be similarly derived, and the result follows. ■

Appendix C. Simulation Results for Erdös-Rényi Graphs

We revisit the simulations from Section 7 but use Erdös-Rényi (ER) graphs in which each possible edge is independently included with probability π. Independent uniform draws from $[0.5, 1]$ are used to fill the non-zero off-diagonal entries of the symmetric matrix $\mathbf{K}_0$. The diagonal elements are set such that $\mathbf{K}_0$ has minimum eigenvalue 0.1. We choose $\pi = 0.08$ for $n = 1000$, and $\pi = 0.02$ for $n = 80$.

C.1. Truncated GGMs

In this section we present the results for truncated GGMs.

C.1.1. Choice of h

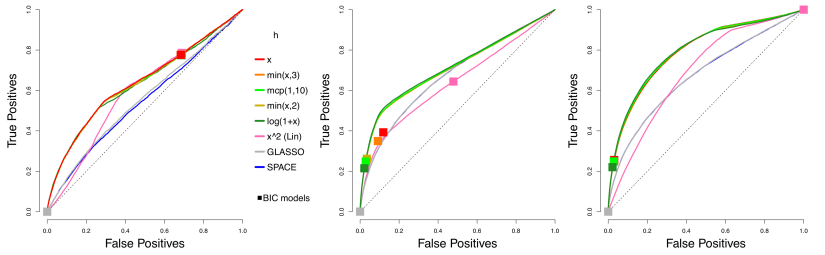
The results for truncated centered GGMs are reported in Table 4 and Figure 13. Those for truncated non-centered GGMs using the profiled estimator are in Table 5 and Figure 14.

Centered, $n = 80$, multiplier 1.8647, ER					
$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
∞	0.632	0.036	∞	0.638	0.035
2	0.632	0.036	3	0.638	0.035
1	0.630	0.035	2	0.635	0.035
0.5	0.613	0.033	1	0.623	0.033
MCP$(1, c)$			SCAD$(1, c)$		
c	Mean	sd	c	Mean	sd
10	0.637	0.035	10	0.638	0.035
5	0.636	0.036	5	0.637	0.035
1	0.617	0.033	2	0.632	0.035
$x^{1.5}$: (0.627, 0.032)			x^2: (0.595,0.028)		
GLASSO (0.553,0.029)			SPACE: (0.544, 0.026)		
NS: (0.543,0.028)			SJ: (0.519,0.028)		

Centered, $n = 1000$, multiplier 1, ER					
$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
∞	0.716	0.016	2	0.710	0.016
2	0.716	0.016	3	0.710	0.016
1	0.715	0.016	1	0.710	0.017
0.5	0.694	0.017	∞	0.709	0.016
MCP$(1, c)$			SCAD$(1, c)$		
c	Mean	sd	c	Mean	sd
5	0.714	0.016	2	0.713	0.016
10	0.711	0.016	5	0.711	0.016
1	0.707	0.017	10	0.710	0.016
$x^{1.5}$: (0.678,0.016)			x^2: (0.64,0.017)		
GLASSO: (0.675,0.016)			SPACE: (0.675,0.016)		
NS: (0.675,0.016)			SJ: (0.624,0.017)		

Centered, $n = 1000$, multiplier 1.6438, ER					
$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
∞	0.796	0.014	∞	0.795	0.014
2	0.796	0.014	3	0.794	0.014
1	0.794	0.014	2	0.792	0.014
0.5	0.772	0.015	1	0.784	0.015
MCP$(1, c)$			SCAD$(1, c)$		
c	Mean	sd	c	Mean	sd
5	0.796	0.014	5	0.795	0.014
10	0.796	0.014	10	0.795	0.014
1	0.778	0.015	2	0.793	0.014
$x^{1.5}$: (0.757,0.015)			x^2: (0.693,0.016)		
GLASSO: (0.675,0.016)			SPACE: (0.675,0.016)		
NS: (0.675,0.016)			SJ: (0.624,0.017)		

Table 4: Mean and standard deviation of areas under the ROC curves (AUC) using different estimators in the centered setting, with $n = 80$ and multiplier 1.8647, or $n = 1000$ and multipliers 1 and 1.6438. Methods include our estimator with different choices of h, GLASSO, SPACE, neighborhood selection (NS), and Space JAM (SJ).



(a) $n = 80$, mult = 1.8647, ER (b) $n = 1000$, mult = 1, ER (c) $n = 1000$, mult = 1.6438, ER

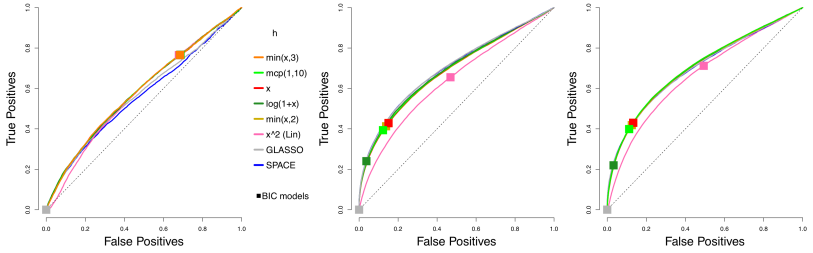
Figure 13: Average ROC curves of our *centered* estimator for $m = 100$ variables and two sample sizes n under various choices of h, compared to SPACE and GLASSO, for the *truncated centered GGM* case. Squares indicate average true positive rate (TPR) and false positive rate (FPR) of models picked by eBIC with refitting for the estimator in the same color.

Non-centered profiled, $n = 80$, multiplier 1.8647, ER					
$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
1	0.588	0.034	3	0.588	0.033
∞	0.588	0.034	∞	0.588	0.033
2	0.588	0.034	2	0.588	0.033
0.5	0.576	0.033	1	0.583	0.033
MCP$(1, c)$			SCAD$(1, c)$		
c	Mean	sd	c	Mean	sd
5	0.588	0.033	5	0.588	0.033
10	0.588	0.033	10	0.588	0.033
1	0.581	0.033	2	0.587	0.033
$x^{1.5}$: (0.582,0.028)			x^2: (0.576,0.028)		
GLASSO: (0.572,0.033)			SPACE: (0.562,0.031)		
NS: (0.560,0.032)			SJ: (0.535,0.027)		

Non-centered profiled, $n = 1000$, multiplier 1, ER					
$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
2	0.692	0.022	1	0.687	0.022
∞	0.692	0.022	∞	0.686	0.022
1	0.691	0.022	3	0.685	0.022
0.5	0.684	0.02	2	0.685	0.022
MCP$(1, c)$			SCAD$(1, c)$		
c	Mean	sd	c	Mean	sd
5	0.689	0.022	2	0.687	0.022
1	0.689	0.020	5	0.687	0.022
10	0.687	0.022	10	0.686	0.022
$x^{1.5}$: (0.663,0.020)			x^2: (0.638,0.019)		
GLASSO (0.700,0.022)			SPACE: (0.699,0.022)		
NS: (0.699,0.022)			SJ: (0.655,0.021)		

Non-centered profiled, $n = 1000$, multiplier 1.6438, ER					
$\min(\log(1+x), c)$			$\min(x, c)$		
c	Mean	sd	c	Mean	sd
2	0.705	0.021	∞	0.705	0.022
∞	0.705	0.021	3	0.705	0.021
1	0.703	0.021	2	0.702	0.022
0.5	0.683	0.019	1	0.695	0.021
MCP$(1, c)$			SCAD$(1, c)$		
c	Mean	sd	c	Mean	sd
5	0.706	0.021	10	0.705	0.022
10	0.706	0.022	5	0.705	0.022
1	0.690	0.019	2	0.703	0.022
$x^{1.5}$: (0.689,0.021)			x^2: (0.664,0.019)		
GLASSO (0.700,0.022)			SPACE: (0.699,0.022)		
NS: (0.699,0.022)			SJ: (0.655,0.021)		

Table 5: Mean and standard deviation of AUC using different profiled estimators in the non-centered setting, with $n = 80$ and multiplier 1.8647, or $n = 1000$ and multipliers 1 and 1.6438. Methods as for Table 4.

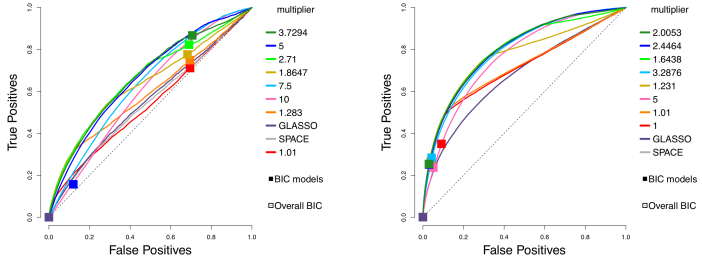


(a) $n = 80$, mult = 1.8647, ER (b) $n = 1000$, mult = 1, ER (c) $n = 1000$, mult = 1.6438, ER

Figure 14: Average for the truncated non-centered GGM case. $n = 80$ or 1000, $m = 100$.

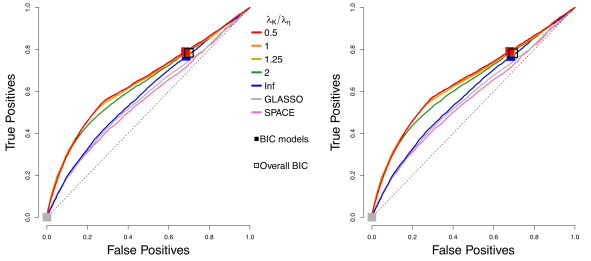
C.1.2. Choice of multiplier

The results for truncated centered GGMs where each curve represents a different multiplier are shown in Figure 15, and those for truncated non-centered GGMs are in Figure 16, where each curve corresponds to a different ratio $\lambda_{\mathbf{K}}/\lambda_{\boldsymbol{\eta}}$.

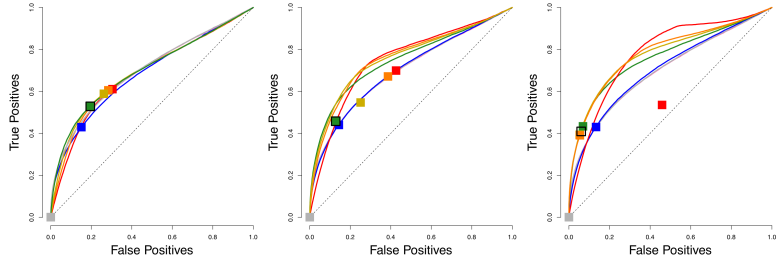


(a) $n = 80$, ER (b) $n = 1000$, ER

Figure 15: Performance of $\min(x, 3)$ for truncated centered GGMs with different multipliers, compared to GLASSO and SPACE, in the centered setting, $n = 80$ or 1000.



(a) $n = 80$, mult = 1.7897, ER (b) $n = 80$, mult = 1.8647, ER

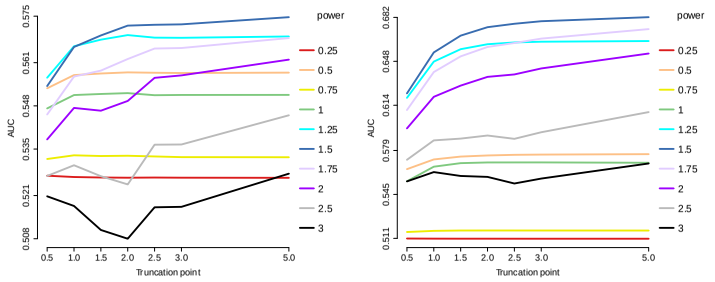


(c) $n = 1000$, mult = 1, ER (d) $n = 1000$, mult = 1.2310, ER (e) $n = 1000$, mult = 1.6438, ER

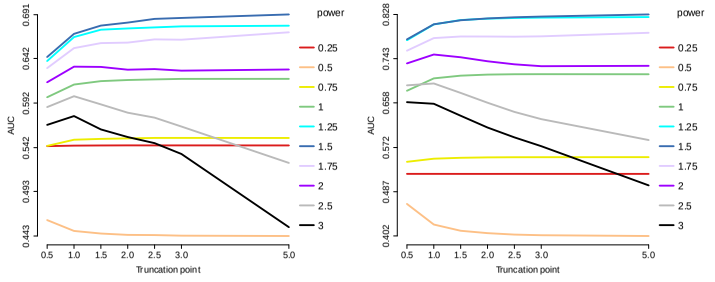
Figure 16: Performance of the non-centered estimator with $h(x) = \min(x, 3)$. Each curve corresponds to a different choice of $\lambda_{\mathbf{K}}/\lambda_{\boldsymbol{\eta}}$. Squares indicate models picked by eBIC with refit. The square with black outline has the highest eBIC among all models (combinations of $\lambda_{\mathbf{K}}$, $\lambda\boldsymbol{\eta}$). The multipliers correspond to medium or high for $n = 80$, and low, medium and high for $n = 1000$, respectively.

C.2. Other a/b Models

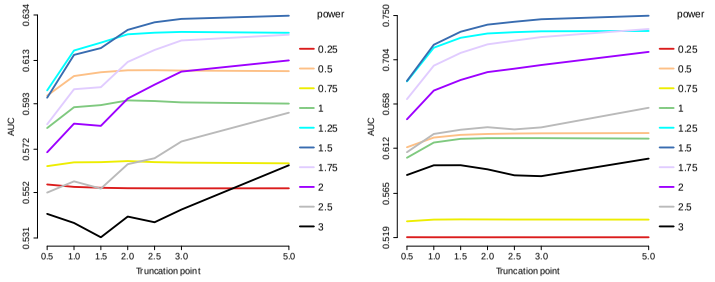
Figure 17 exhibits the results for the exponential models.



(a) $n = 80$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled est, ER (b) $n = 1000$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled est, ER



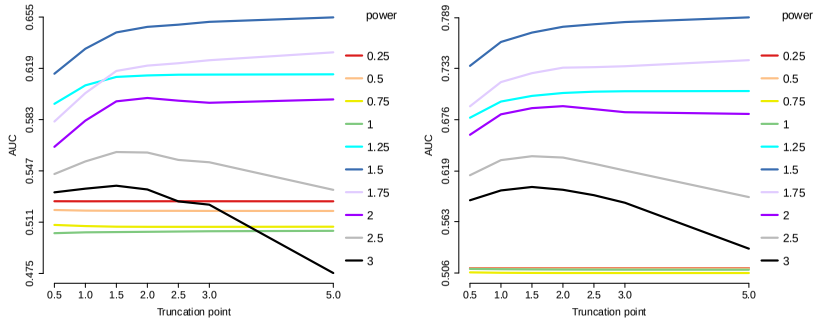
(c) $n = 80$, $\boldsymbol{\eta} = \mathbf{0}$, centered est, ER (d) $n = 1000$, $\boldsymbol{\eta} = \mathbf{0}$, centered est, ER



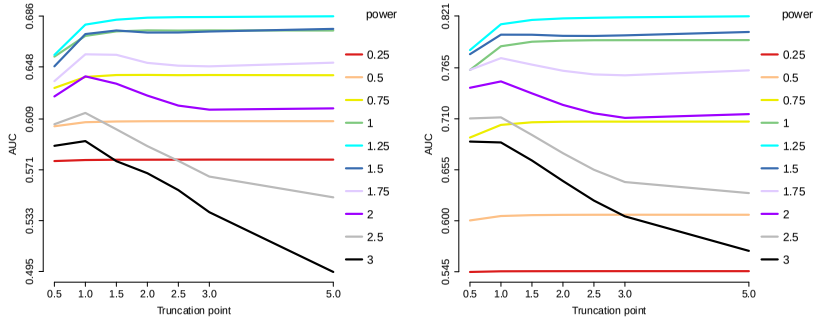
(e) $n = 80$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled est, ER (f) $n = 1000$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled est, ER

Figure 17: AUCs for edge recovery using generalized score matching for the exponential models. Each curve represents a different choice of power p in $h(x) = \min(x^p, c)$, and the x axis marks the truncation point c. Colors are sorted by p.

Figure 18 displays the results for the gamma models.



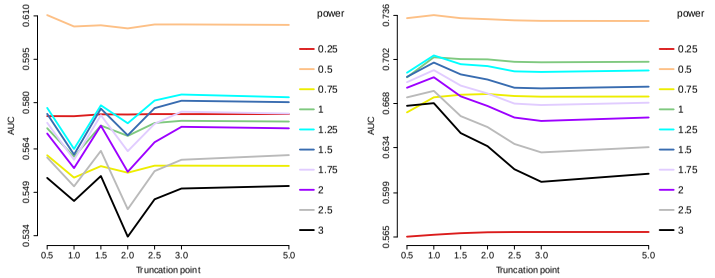
(a) $n = 80$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator, ER (b) $n = 1000$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator, ER



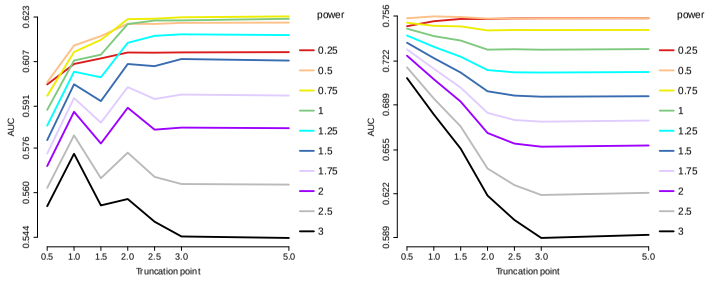
(c) $n = 80$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator, ER (d) $n = 1000$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator, ER

Figure 18: AUCs for edge recovery using generalized score matching for the gamma models. Each curve represents a different choice of power p in $h(x) = \min(x^p, c)$, and the x axis marks the truncation point c. Colors are sorted by p.

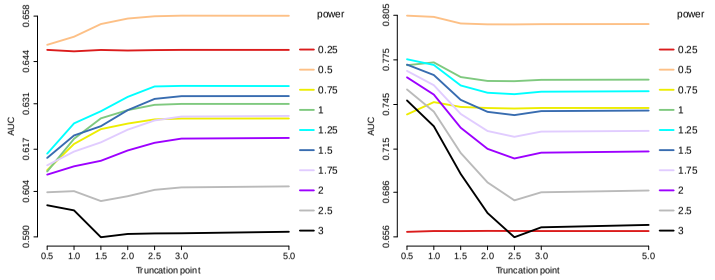
Figures 19 and 20 demonstrate the results for $a = 3/2$ and $b = 1/2$ or $b = 0$, respectively.



(a) $n = 80$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled est, ER (b) $n = 1000$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled est, ER

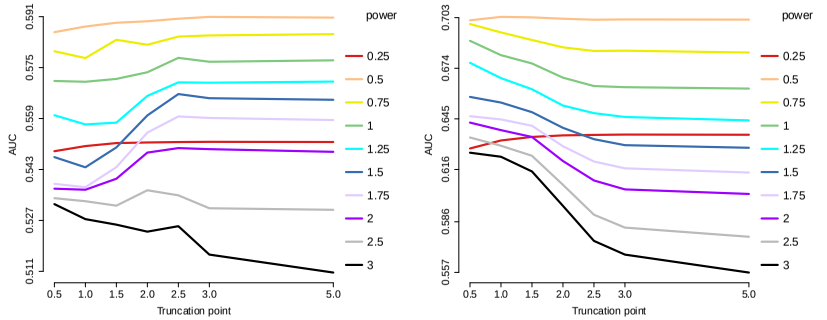


(c) $n = 80$, $\boldsymbol{\eta} = \mathbf{0}$, centered est, ER (d) $n = 1000$, $\boldsymbol{\eta} = \mathbf{0}$, centered est, ER

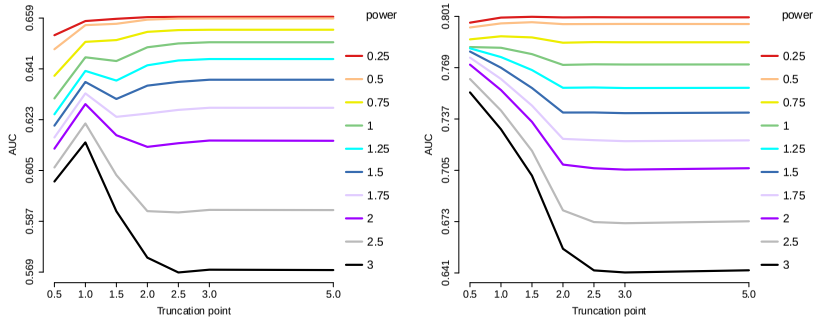


(e) $n = 80$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled est, ER (f) $n = 1000$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled est, ER

Figure 19: AUCs for edge recovery using generalized score matching for $a = 3/2$, $b = 1/2$. Each curve represents a different choice of power p in $h(x) = \min(x^p, c)$, and the x axis marks the truncation point c. Colors are sorted by p.



(a) $n = 80$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator, ER (b) $n = 1000$, $\boldsymbol{\eta} = -0.5\mathbf{1}_{100}$, profiled estimator, ER



(c) $n = 80$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator, ER (d) $n = 1000$, $\boldsymbol{\eta} = 0.5\mathbf{1}_{100}$, profiled estimator, ER

Figure 20: AUCs for edge recovery using generalized score matching for $a = 3/2$, $b = 0$. Each curve represents a different choice of power p in $h(x) = \min(x^p, c)$, and the x axis marks the truncation point c. Colors are sorted by p.

References

Murilo P. Almeida and Basilis Gidas. A variational method for estimating the parameters of MRF from complete or incomplete data. *Ann. Appl. Probab.*, 3(1):103–136, 1993.

Rina Foygel Barber and Mathias Drton. High-dimensional Ising model selection with Bayesian information criteria. *Electron. J. Stat.*, 9(1):567–607, 2015.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities*. Oxford University Press, Oxford, 2013.

Igor Brikun, Deborah Nusskern, Daniel Gillen, Amy Lynn, Daniel Murtagh, John Feczko, William G Nelson, and Diha Freije. A panel of DNA methylation markers reveals extensive methylation in histologically benign prostate biopsy cores from cancer patients. *Biomarker Research*, 2(1):25, 2014.

V. V. Buldygin and Yu. V. Kozachenko. *Metric Characterization of Random Variables and Random Processes*, volume 188 of *Translations of Mathematical Monographs*. American Mathematical Society, Providence, RI, 2000. Translated from the 1998 Russian original by V. Zaiats.

Scott L. Carter, Christian M. Brechbühler, Michael Griffin, and Andrew T. Bond. Gene co-expression network topology provides a framework for molecular characterization of cellular state. *Bioinformatics*, 20(14):2242–2250, 2004.

Jiahua Chen and Zehua Chen. Extended Bayesian information criteria for model selection with large model spaces. *Biometrika*, 95(3):759–771, 2008.

Shizhe Chen, Daniela M. Witten, and Ali Shojaie. Selection and estimation for mixed graphical models. *Biometrika*, 102(1):47–64, 2015.

Adrian Dobra and Alex Lenkoski. Copula Gaussian graphical models and their application to modeling functional disability data. *Ann. Appl. Stat.*, 5(2A):969–993, 2011.

Mathias Drton and Marloes H Maathuis. Structure learning in graphical modeling. *Annual Review of Statistics and Its Application*, 4:365–393, 2017.

Mathias Drton and Michael D. Perlman. Model selection for Gaussian concentration graphs. *Biometrika*, 91(3):591–602, 2004.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *J. Amer. Statist. Assoc.*, 96(456):1348–1360, 2001.

Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

Bernd Fellinghauer, Peter Bühlmann, Martin Ryffel, Michael von Rhein, and Jan D. Reinhardt. Stable graphical model estimation with random forests for discrete, continuous, and mixed variables. *Comput. Statist. Data Anal.*, 64:132–152, 2013.

Rina Foygel and Mathias Drton. Extended Bayesian information criteria for Gaussian graphical models. In *Advances in Neural Information Processing Systems*, pages 604–612, 2010.

Jerome Friedman, Trevor Hastie, Holger Höfling, and Robert Tibshirani. Pathwise coordinate optimization. *Ann. Appl. Stat.*, 1(2):302–332, 2007.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

Jing-Dong J. Han, Nicolas Bertin, Tong Hao, Debra S. Goldberg, Gabriel F. Berriz, Lan V. Zhang, Denis Dupuy, Albertha JM. Walhout, Michael E. Cusick, Frederick P. Roth, et al. Evidence for dynamically organized modularity in the yeast protein–protein interaction network. *Nature*, 430(6995):88, 2004.

Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *J. Mach. Learn. Res.*, 6:695–709, 2005.

Aapo Hyvärinen. Some extensions of score matching. *Comput. Statist. Data Anal.*, 51(5): 2499–2512, 2007.

David Inouye, Pradeep Ravikumar, and Inderjit Dhillon. Square root graphical models: Multivariate generalizations of univariate exponential families that permit positive dependencies. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2445–2453, 2016.

Hawoong Jeong, Sean P. Mason, A-L. Barabási, and Zoltan N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411(6833):41, 2001.

Kshitij Khare, Sang-Yun Oh, and Bala Rajaratnam. A convex pseudolikelihood framework for high dimensional partial correlation estimation with convergence guarantees. *J. R. Stat. Soc. Ser. B. Stat. Methodol.*, 77(4):803–825, 2015.

Steffen L. Lauritzen. *Graphical Models*, volume 17 of *Oxford Statistical Science Series*. The Clarendon Press, Oxford University Press, New York, 1996.

Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *J. Multivariate Anal.*, 88(2):365–411, 2004.

Lina Lin, Mathias Drton, and Ali Shojaie. Estimation of high-dimensional graphical models using regularized score matching. *Electron. J. Stat.*, 10(1):806–854, 2016.

Han Liu, John Lafferty, and Larry Wasserman. The nonparanormal: semiparametric estimation of high dimensional undirected graphs. *J. Mach. Learn. Res.*, 10:2295–2328, 2009.

Han Liu, Fang Han, Ming Yuan, John Lafferty, and Larry Wasserman. High-dimensional semiparametric Gaussian copula graphical models. *Ann. Statist.*, 40(4):2293–2326, 2012.

Weidong Liu and Xi Luo. Fast and adaptive sparse precision matrix estimation in high dimensions. *J. Multivariate Anal.*, 135:153–162, 2015.

Jun Luo, Thomas Dunn, Charles Ewing, Jurga Sauvageot, Yidong Chen, Jeffrey Trent, and William Isaacs. Gene expression signature of benign prostatic hyperplasia revealed by cDNA microarray analysis. *The Prostate*, 51(3):189–200, 2002.

Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. *Ann. Statist.*, 34(3):1436–1462, 2006.

Giuseppe Morgia, Mario Falsaperla, Grazia Malaponte, Massimo Madonia, Manuela Indelicato, Salvatore Travali, and Maria Clorinda Mazzarino. Matrix metalloproteinases as diagnostic (MMP-13) and prognostic (MMP-2, MMP-9) markers of prostate cancer. *Urological Research*, 33(1):44–50, 2005.

Shintaro Narita, Alan So, Susan Ettinger, Norihiro Hayashi, Mototsugu Muramaki, Ladan Fazli, Youngsoo Kim, and Martin E Gleave. GLI2 knockdown using an antisense oligonucleotide induces apoptosis and chemosensitizes cells to paclitaxel in androgen-independent prostate cancer. *Clinical Cancer Research*, 14(18):5769–5777, 2008.

Matthew Parry. Extensive scoring rules. *Electron. J. Stat.*, 10(1):1098–1108, 2016.

Matthew Parry, A. Philip Dawid, and Steffen Lauritzen. Proper local scoring rules. *Ann. Statist.*, 40(1):561–592, 2012.

Jie Peng, Nengfeng Zhou, and Ji Zhu. Partial correlation estimation by joint sparse regression models. *J. Amer. Statist. Assoc.*, 104(486):735–746, 2009.

Pradeep Ravikumar, Martin J. Wainwright, and John D. Lafferty. High-dimensional Ising model selection using ℓ_1-regularized logistic regression. *Ann. Statist.*, 38(3):1287–1319, 2010.

Pradeep Ravikumar, Martin J. Wainwright, Garvesh Raskutti, and Bin Yu. High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. *Electron. J. Stat.*, 5:935–980, 2011.

G. M. Tallis. The moment generating function of the truncated multi-normal distribution. *J. Roy. Statist. Soc. Ser. B*, 23:223–229, 1961.

Saravanan Thiyagarajan, Neehar Bhatia, Shannon Reagan-Shaw, Diana Cozma, Andrei Thomas-Tikhonenko, Nihal Ahmad, and Vladimir S Spiegelman. Role of GLI2 transcription factor in growth and tumorigenicity of prostate cells. *Cancer Research*, 67(22): 10642–10646, 2007.

Robert Tibshirani, Jacob Bien, Jerome Friedman, Trevor Hastie, Noah Simon, Jonathan Taylor, and Ryan J. Tibshirani. Strong rules for discarding predictors in lasso-type problems. *J. R. Stat. Soc. Ser. B. Stat. Methodol.*, 74(2):245–266, 2012.

Dominique Trudel, Yves Fradet, François Meyer, François Harel, and Bernard Têtu. Significance of MMP-2 expression in prostate cancer. *Cancer Research*, 63(23):8511–8515, 2003.

Hannu Väliaho. Criteria for copositive matrices. *Linear Algebra Appl.*, 81:19–34, 1986.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. In *Compressed Sensing*, pages 210–268. Cambridge Univ. Press, Cambridge, 2012.

Arend Voorman, Ali Shojaie, and Daniela Witten. Graph estimation with joint additive models. *Biometrika*, 101(1):85–101, 2014.

Eunho Yang, Pradeep Ravikumar, Genevera I. Allen, and Zhandong Liu. Graphical models via univariate exponential family distributions. *J. Mach. Learn. Res.*, 16:3813–3847, 2015.

Ming Yu, Mladen Kolar, and Varun Gupta. Statistical inference for pairwise graphical models using score matching. In *Advances in Neural Information Processing Systems*, pages 2829–2837, 2016.

Ming Yuan and Yi Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

Małgorzata Żak-Szatkowska and Małgorzata Bogdan. Modified versions of the Bayesian information criterion for sparse generalized linear models. *Comput. Statist. Data Anal.*, 55(11):2908–2924, 2011.

Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *Ann. Statist.*, 38(2):894–942, 2010.

Teng Zhang and Hui Zou. Sparse precision matrix estimation via lasso penalized D-trace loss. *Biometrika*, 101(1):103–120, 2014.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 67(2):301–320, 2005.

Journal of Machine Learning Research 20 (2019) 1-33 Submitted 5/18; Revised 2/19; Published 4/19

Nonuniformity of P-values Can Occur Early in Diverging Dimensions

Yingying Fan fanyingy@marshall.usc.edu
Data Sciences and Operations Department
University of Southern California
Los Angeles, CA 90089, USA

Emre Demirkaya demirkay@usc.edu
Business Analytics & Statistics
The University of Tennessee, Knoxville
Knoxville, TN 37996-4140, USA

Jinchi Lv jinchilv@marshall.usc.edu
Data Sciences and Operations Department
University of Southern California
Los Angeles, CA 90089, USA

Editor: Sara van de Geer

Abstract

Evaluating the joint significance of covariates is of fundamental importance in a wide range of applications. To this end, p-values are frequently employed and produced by algorithms that are powered by classical large-sample asymptotic theory. It is well known that the conventional p-values in Gaussian linear model are valid even when the dimensionality is a non-vanishing fraction of the sample size, but can break down when the design matrix becomes singular in higher dimensions or when the error distribution deviates from Gaussianity. A natural question is when the conventional p-values in generalized linear models become invalid in diverging dimensions. We establish that such a breakdown can occur early in nonlinear models. Our theoretical characterizations are confirmed by simulation studies.

Keywords: Nonuniformity, p-value, breakdown point, generalized linear model, high dimensionality, joint significance testing

1. Introduction

In many applications it is often desirable to evaluate the significance of covariates in a predictive model for some response of interest. Identifying a set of significant covariates can facilitate domain experts to further probe their causal relationships with the response. Ruling out insignificant covariates can also help reduce the fraction of false discoveries and narrow down the scope of follow-up experimental studies by scientists. These tasks certainly require an accurate measure of feature significance in finite samples. The tool of p-values has provided a powerful framework for such investigations.

As p-values are routinely produced by algorithms, practitioners should perhaps be aware that those p-values are usually based on classical large-sample asymptotic theory. For example, marginal p-values have been employed frequently in large-scale applications when the number of covariates p greatly exceeds the number of observations n. Those p-values are based on marginal regression models linking each individual covariate to the response separately. In these marginal regression models, the ratio of sample size to model dimensionality is equal to n, which results in justified p-values as sample size increases. Yet due to the correlations among the covariates, we often would like to investigate the joint significance of a covariate in a regression model conditional on all other covariates, which is the main focus of this paper. A natural question is whether conventional joint p-values continue to be valid in the regime of diverging dimensionality p.

It is well known that fitting the linear regression model with $p > n$ using the ordinary least squares can lead to perfect fit giving rise to zero residual vector, which renders the p-values undefined. When $p \leq n$ and the design matrix is nonsingular, the p-values in the linear regression model are well defined and valid thanks to the exact normality of the least-squares estimator when the random error is Gaussian and the design matrix is deterministic. When the error is non-Gaussian, Huber (1973) showed that the least-squares estimator can still be asymptotically normal under the assumption of $p = o(n)$, but is generally no longer normal when $p = o(n)$ fails to hold, making the conventional p-values inaccurate in higher dimensions. For the asymptotic properties of M-estimators for robust regression, see, for example, Huber (1973); Portnoy (1984, 1985) for the case of diverging dimensionality $p = o(n)$ and Karoui et al. (2013); Bean et al. (2013) for the scenario when the dimensionality p grows proportionally to sample size n.

We have seen that the conventional p-values for the least-squares estimator in linear regression model can start behaving wildly and become invalid when the dimensionality p is of the same order as sample size n and the error distribution deviates from Gaussianity. A natural question is whether similar phenomenon holds for the conventional p-values for the maximum likelihood estimator (MLE) in the setting of diverging-dimensional nonlinear models. More specifically, we aim to answer the question of whether $p \sim n$ is still the breakdown point of the conventional p-values when we move away from the regime of linear regression model, where $\sim$ stands for asymptotic order. To simplify the technical presentation, in this paper we adopt the generalized linear model (GLM) as a specific family of nonlinear models (McCullagh and Nelder, 1989). The GLM with a canonical link assumes that the conditional distribution of $\mathbf{y}$ given $\mathbf{X}$ belongs to the canonical exponential family, having the following density function with respect to some fixed measure

$$f_n(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta}) \equiv \prod_{i=1}^{n} f_0(y_i; \theta_i) = \prod_{i=1}^{n} \left\{ c(y_i) \exp\left[\frac{y_i \theta_i - b(\theta_i)}{\phi} \right] \right\}, \tag{1}$$

where $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_p)$ is an $n \times p$ design matrix with $\mathbf{x}_j = (x_{1j}, \cdots, x_{nj})^T$, $j = 1, \cdots, p$, $\mathbf{y} = (y_1, \cdots, y_n)^T$ is an n-dimensional response vector, $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p)^T$ is a p-dimensional regression coefficient vector, $\{f_0(y; \theta) : \theta \in \mathbb{R}\}$ is a family of distributions in the regular exponential family with dispersion parameter $\phi \in (0, \infty)$, and $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_n)^T = \mathbf{X}\boldsymbol{\beta}$. As is common in GLM, the function $b(\theta)$ in (1) is implicitly assumed to be twice continuously differentiable with $b''(\theta)$ always positive. Popularly used GLMs include the linear regression model, logistic regression model, and Poisson regression model for continuous, binary, and count data of responses, respectively.

©2019 Yingying Fan, Emre Demirkaya and Jinchi Lv.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-314.html.

The key innovation of our paper is the formal justification that the conventional p-values in nonlinear models of GLMs can become invalid in diverging dimensions and such a breakdown can occur *much earlier* than in linear models, which spells out a fundamental difference between linear models and nonlinear models. To begin investigating p-values in diverging-dimensional GLMs, let us gain some insights into this problem by looking at the specific case of logistic regression. Recently, Candès (2016) established an interesting phase transition phenomenon of perfect hyperplane separation for high-dimensional classification with an elegant probabilistic argument. Suppose we are given a random design matrix $\mathbf{X} \sim N(\mathbf{0}, I_n \otimes I_p)$ and arbitrary binary y_i's that are not all the same. The phase transition of perfect hyperplane separation happens at the point $p/n = 1/2$. With such a separating hyperplane, there exist some $\boldsymbol{\beta}^* \in \mathbb{R}^p$ and $t \in \mathbb{R}$ such that $\mathbf{x}_i^T \boldsymbol{\beta}^* > t$ for all cases $y_i = 1$ and $\mathbf{x}_i^T \boldsymbol{\beta}^* < t$ for all controls $y_i = 0$. Let us fit a logistic regression model with an intercept. It is easy to show that multiplying the vector $(-t, (\boldsymbol{\beta}^*)^T)^T$ by a divergence sequence of positive numbers c, we can obtain a sequence of logistic regression fits with the fitted response vector approaching $\mathbf{y} = (y_1, \cdots, y_n)^T$ as $c \to \infty$. As a consequence, the MLE algorithm can return a pretty wild estimate that is close to infinity in topology when the algorithm is set to stop. Clearly, in such a case the p-value of the MLE is no longer justified and meaningful. The results in Candès (2016) have two important implications. First, such results reveal that unlike in linear models, p-values in nonlinear models can break down and behave wildly when p/n is of order 1/2; see Karoui et al. (2013); Bean et al. (2013) and discussions below. Second, these results motivate us to characterize the breakdown point of p-values in nonlinear GLMs with $p \sim n^{\alpha_0}$ in the regime of $\alpha_0 \in [0, 1/2)$. In fact, our results show that the breakdown point can be even much earlier than $n/2$.

It is worth mentioning that our work is different in goals from the limited but growing literature on p-values for high-dimensional nonlinear models, and makes novel contributions to such a problem. The key distinction is that existing work has focused primarily on identifying the scenarios in which conventional p-values or their modifications continue to be valid with some sparsity assumption limiting the growth of intrinsic dimensions. For example, Fan and Peng (2004) established the oracle property including the asymptotic normality for nonconcave penalized likelihood estimators in the scenario of $p = o(n^{1/5})$, while Fan and Lv (2011) extended their results to the GLM setting of non-polynomial (NP) dimensionality. In the latter work, the p-values were proved to be valid under the assumption that the intrinsic dimensionality $s = o(n^{1/3})$. More recent work on high-dimensional inference in nonlinear model settings includes van de Geer et al. (2014); Athey et al. (2016) under sparsity assumptions. In addition, two tests were introduced in Guo and Chen (2016) for high-dimensional GLMs without or with nuisance regression parameters, but the p-values were obtained for testing the global hypothesis for a given set of covariates, which is different from our goal of testing the significance of individual covariates simultaneously. Portnoy (1988) studied the asymptotic behavior of the MLE for exponential families under the classical i.i.d. non-regression setting, but with diverging dimensionality. In contrast, our work under the GLM assumes the regression setting in which the design matrix $\mathbf{X}$ plays an important role in the asymptotic behavior of the MLE $\widehat{\boldsymbol{\beta}}$. The validity of the asymptotic normality of the MLE was established in Portnoy (1988) under the condition of $p = o(n^{1/2})$, but the precise breakdown point in diverging dimensionality was not investigated therein. Another line of work is focused on generating asymptotically valid p-values when p/n converges to a

fixed positive constant. For instance, Karoui et al. (2013) and Bean et al. (2013) considered M-estimators in the linear model and showed that their variance is greater than classically predicted. Based on this result, it is possible to produce p-values by making adjustments for the inflated variance in high dimensions. Recently, Sur and Candès (2018) showed that similar adjustment is possible for the likelihood ratio test (LRT) for logistic regression. Our work differs from this line of work in two important aspects. First, our focus is on the *classical* p-values and their validity. Second, their results concern dimensionality that is comparable to sample size, while we aim to analyze the problem for a lower range of dimensionality and pinpoint the exact breakdown point of p-values.

The rest of the paper is organized as follows. Section 2 provides characterizations of p-values in low dimensions. We establish the nonuniformity of GLM p-values in diverging dimensions in Section 3. Section 4 presents several simulation examples verifying the theoretical phenomenon. We discuss some implications of our results in Section 5. The proofs of all the results are relegated to the Appendix.

2. Characterizations of P-values in Low Dimensions

To pinpoint the breakdown point of GLM p-values in diverging dimensions, we start with characterizing p-values in low dimensions. In contrast to existing work on the asymptotic distribution of the penalized MLE, our results in this section focus on the asymptotic normality of the unpenalized MLE in diverging-dimensional GLMs, which justifies the validity of conventional p-values. Although Theorems 1 and 4 to be presented in Sections 2.2 and A are in the conventional sense of relatively small p, to the best of our knowledge such results are not available in the literature before in terms of the maximum range of dimensionality p without any sparsity assumption.

2.1. Maximum likelihood estimation

For the GLM (1), the log-likelihood $\log f_n(\mathbf{y}; \mathbf{X}, \boldsymbol{\beta})$ of the sample is given, up to an affine transformation, by

$$\ell_n(\boldsymbol{\beta}) = n^{-1} \left[\mathbf{y}^T \mathbf{X}\boldsymbol{\beta} - \mathbf{1}^T \mathbf{b}(\mathbf{X}\boldsymbol{\beta}) \right], \tag{2}$$

where $\mathbf{b}(\boldsymbol{\theta}) = (b(\theta_1), \cdots, b(\theta_n))^T$ for $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_n)^T \in \mathbb{R}^n$. Denote by $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_1, \cdots, \widehat{\beta}_p)^T \in \mathbb{R}^p$ the MLE which is the maximizer of (2), and

$$\boldsymbol{\mu}(\boldsymbol{\theta}) = (b'(\theta_1), \cdots, b'(\theta_n))^T \quad \text{and} \quad \boldsymbol{\Sigma}(\boldsymbol{\theta}) = \operatorname{diag}\{b''(\theta_1), \cdots, b''(\theta_n)\}. \tag{3}$$

A well-known fact is that the n-dimensional response vector $\mathbf{y}$ in GLM (1) has mean vector $\boldsymbol{\mu}(\boldsymbol{\theta})$ and covariance matrix $\phi \boldsymbol{\Sigma}(\boldsymbol{\theta})$. Clearly, the MLE $\widehat{\boldsymbol{\beta}}$ is given by the unique solution to the score equation

$$\mathbf{X}^T [\mathbf{y} - \boldsymbol{\mu}(\mathbf{X}\boldsymbol{\beta})] = \mathbf{0} \tag{4}$$

when the design matrix $\mathbf{X}$ is of full column rank p.

It is worth mentioning that for the linear model, the score equation (4) becomes the well-known normal equation $\mathbf{X}^T\mathbf{y} = \mathbf{X}^T\mathbf{X}\boldsymbol{\beta}$ which admits a closed form solution. On the other hand, equation (4) does not admit a closed form solution in general nonlinear models. This fact due to the nonlinearity of the mean function $\boldsymbol{\mu}(\cdot)$ causes the key difference between

the linear and nonlinear models. In future presentations, we will occasionally use the term *nonlinear GLMs* to exclude the linear model from the family of GLMs when necessary.

We will present in the next two sections some sufficient conditions under which the asymptotic normality of MLE holds. In particular, Section 2.2 concerns the case of fixed design and Section A deals with the case of random design. In addition, Section 2.2 allows for general regression coefficient vector $\boldsymbol{\beta}_0$ and the results extend some existing ones in the literature, while Section A assumes the global null $\boldsymbol{\beta}_0 = \mathbf{0}$ and Gaussian random design which enable us to pinpoint the exact breakdown point of the asymptotic normality for the MLE.

2.2. Conventional p-values in low dimensions under fixed design

Recall that we condition on the design matrix $\mathbf{X}$ in this section. We first introduce a deviation probability bound that facilitates our technical analysis. Consider both cases of bounded responses and unbounded responses. In the latter case, assume that there exist some constants $M, v_0 > 0$ such that

$$\max_{1\le i\le n} E\left\{\exp\left[\frac{|y_i - b'(\theta_{0,i})|}{M}\right] - 1 - \frac{|y_i - b'(\theta_{0,i})|}{M}\right\} M^2 \le \frac{v_0}{2} \tag{5}$$

with $(\theta_{0,1}, \cdots, \theta_{0,n})^T = \boldsymbol{\theta}_0 = \mathbf{X}\boldsymbol{\beta}_0$, where $\boldsymbol{\beta}_0 = (\beta_{0,1}, \cdots, \beta_{0,p})^T$ denotes the true regression coefficient vector in model (1). Then by Fan and Lv (2011, 2013), it holds that for any $\mathbf{a} \in \mathbb{R}^n$,

$$P\left(\left|\mathbf{a}^T\mathbf{Y} - \mathbf{a}^T\boldsymbol{\mu}(\boldsymbol{\theta}_0)\right| > \|\mathbf{a}\|_2\, \varepsilon\right) \le \varphi(\varepsilon), \tag{6}$$

where $\varphi(\varepsilon) = 2e^{-c_1\varepsilon^2}$ with $c_1 > 0$ some constant, and $\varepsilon \in (0, \infty)$ if the responses are bounded and $\varepsilon \in (0, \|\mathbf{a}\|_2/\|\mathbf{a}\|_\infty]$ if the responses are unbounded.

For nonlinear GLMs, the MLE $\widehat{\boldsymbol{\beta}}$ solves the nonlinear score equation (4) whose solution generally does not admit an explicit form. To address such a challenge, we construct a solution to equation (4) in an asymptotically shrinking neighborhood of $\boldsymbol{\beta}_0$ that meets the MLE $\widehat{\boldsymbol{\beta}}$ thanks to the uniqueness of the solution. Specifically, define a neighborhood of $\boldsymbol{\beta}_0$ as

$$\mathcal{N}_0 = \{\boldsymbol{\beta} \in \mathbb{R}^p : \|\boldsymbol{\beta} - \boldsymbol{\beta}_0\|_\infty \le n^{-\gamma}\log n\} \tag{7}$$

for some constant $\gamma \in (0, 1/2]$. Assume that $p = O(n^{\alpha_0})$ for some $\alpha_0 \in (0, \gamma)$ and let $b_n = o\{\min(n^{1/2-\gamma}\sqrt{\log n}, s_n^{-1}n^{2\gamma-\alpha_0-1/2}/(\log n)^2\}$ be a diverging sequence of positive numbers, where s_n is a sequence of positive numbers that will be specified in Theorem 1 below. We need some basic regularity conditions to establish the asymptotic normality of the MLE $\widehat{\boldsymbol{\beta}}$.

Condition 1 *The design matrix* $\mathbf{X}$ *satisfies*

$$\left\|\left[\mathbf{X}^T\boldsymbol{\Sigma}(\boldsymbol{\theta}_0)\mathbf{X}\right]^{-1}\right\|_\infty = O(b_n n^{-1}), \tag{8}$$

$$\max_{\boldsymbol{\beta}\in\mathcal{N}_0} \max_{j=1}^p \lambda_{\max}\left[\mathbf{X}^T\mathrm{diag}\left\{|\mathbf{x}_j| \circ \left|\boldsymbol{\mu}''(\mathbf{X}\boldsymbol{\beta})\right|\right\}\mathbf{X}\right] = O(n) \tag{9}$$

with $\circ$ *denoting the Hadamard product and derivatives understood componentwise. Assume that* $\max_{j=1}^p \|\mathbf{x}_j\|_\infty < c_1^{1/2}\{n/(\log n)\}^{1/2}$ *if the responses are unbounded.*

Condition 2 *The eigenvalues of* $n^{-1}\mathbf{A}_n$ *are bounded away from* 0 *and* ∞, $\sum_{i=1}^n (\mathbf{z}_i^T\mathbf{A}_n^{-1}\mathbf{z}_i)^{3/2} = o(1)$, *and* $\max_{i=1}^n E|y_i - b'(\theta_{0,i})|^3 = O(1)$, *where* $\mathbf{A}_n = \mathbf{X}^T\boldsymbol{\Sigma}(\boldsymbol{\theta}_0)\mathbf{X}$ *and* $(\mathbf{z}_1, \cdots, \mathbf{z}_n)^T = \mathbf{X}$.

Conditions 1 and 2 put some basic restrictions on the design matrix $\mathbf{X}$ and a moment condition on the responses. For the case of linear model, bound (8) becomes $\|(\mathbf{X}^T\mathbf{X})^{-1}\|_\infty = O(b_n/n)$ and bound (9) holds automatically since $b'''(\theta) \equiv 0$. Condition 2 is related to the Lyapunov condition.

Theorem 1 (Asymptotic normality) *Assume that Conditions 1–2 and probability bound* (6) *hold. Then*

a) there exists a unique solution $\widehat{\boldsymbol{\beta}}$ *to score equation (4) in* $\mathcal{N}_0$ *with asymptotic probability one;*

b) the MLE $\widehat{\boldsymbol{\beta}}$ *satisfies that for each vector* $\mathbf{u} \in \mathbb{R}^p$ *with* $\|\mathbf{u}\|_2 = 1$ *and* $\|\mathbf{u}\|_1 = O(s_n)$,

$$(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}(\mathbf{u}^T\widehat{\boldsymbol{\beta}} - \mathbf{u}^T\boldsymbol{\beta}_0) \xrightarrow{\mathscr{D}} N(0, \phi) \tag{10}$$

and specifically for each $1 \le j \le p$,

$$(\mathbf{A}_n^{-1})_{jj}^{-1/2}(\widehat{\beta}_j - \beta_{0,j}) \xrightarrow{\mathscr{D}} N(0, \phi), \tag{11}$$

where $\mathbf{A}_n = \mathbf{X}^T\boldsymbol{\Sigma}(\boldsymbol{\theta}_0)\mathbf{X}$ *and* $(\mathbf{A}_n^{-1})_{jj}$ *denotes the* j*th diagonal entry of matrix* $\mathbf{A}_n^{-1}$.

Theorem 1 establishes the asymptotic normality of the MLE and consequently justifies the validity of the conventional p-values in low dimensions. Note that for simplicity, we present here only the marginal asymptotic normality, and the joint asymptotic normality also holds for the projection of the MLE onto any fixed-dimensional subspace. This result can also be extended to the case of misspecified models; see, for example, Lv and Liu (2014).

As mentioned in the Introduction, the asymptotic normality was shown in Fan and Lv (2011) for nonconcave penalized MLE having intrinsic dimensionality $s = o(n^{1/3})$. In contrast, our result in Theorem 1 allows for the scenario of $p = o(n^{1/2})$ with no sparsity assumption in view of our technical conditions. In particular, we see that the conventional p-values in GLMs generally remain valid in the regime of slowly diverging dimensionality $p = o(n^{1/2})$.

3. Nonuniformity of GLM P-values in Diverging Dimensions

So far we have seen that for nonlinear GLMs, the p-values can be valid when $p = o(n^{1/2})$ as shown in Section 2, and can become meaningless when $p \ge n/2$ as discussed in the Introduction. Apparently, there is a big gap between these two regimes of growth of dimensionality p. To provide some guidance on the practical use of p-values in nonlinear GLMs, it is of crucial importance to characterize their breakdown point. To highlight the main message with simplified technical presentation, hereafter we content ourselves with the specific case of logistic regression model for binary response. Moreover, we investigate the distributional property in (11) for the scenario of true regression coefficient vector $\boldsymbol{\beta}_0 = \mathbf{0}$, that is, under the global null. We argue that this specific model is sufficient for our purpose because if the

conventional p-values derived from MLEs fail (i.e., (11) fails) for at least one $\boldsymbol{\beta}_0$ (in particular $\boldsymbol{\beta}_0 = \mathbf{0}$), then conventional p-values are not justified. Therefore, the breakdown point for logistic regression is at least the breakdown point for general nonlinear GLMs. This argument is fundamentally different from that of proving the overall validity of conventional p-values, where one needs to prove the asymptotic normality of MLEs under general GLMs rather than any specific model.

3.1. The wild side of nonlinear regime

For the logistic regression model (1), we have $b(\theta) = \log(1+e^{\theta})$, $\theta \in \mathbb{R}$ and $\phi = 1$. The mean vector $\boldsymbol{\mu}(\boldsymbol{\theta})$ and covariance matrix $\phi\boldsymbol{\Sigma}(\boldsymbol{\theta})$ of the n-dimensional response vector $\mathbf{y}$ given by (3) now take the familiar form of $\boldsymbol{\mu}(\boldsymbol{\theta}) = \left(\frac{e^{\theta_1}}{1+e^{\theta_1}}, \cdots, \frac{e^{\theta_n}}{1+e^{\theta_n}}\right)^T$ and

$$\boldsymbol{\Sigma}(\boldsymbol{\theta}) = \operatorname{diag}\left\{\frac{e^{\theta_1}}{(1+e^{\theta_1})^2}, \cdots, \frac{e^{\theta_n}}{(1+e^{\theta_n})^2}\right\}$$

with $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_n)^T = \mathbf{X}\boldsymbol{\beta}$. In many real applications, one would like to interpret the significance of each individual covariate produced by algorithms based on the conventional asymptotic normality of the MLE as established in Theorem 1. As argued at the beginning of this section, in order to justify the validity of p-values in GLMs, the underlying theory should at least ensure that the distributional property (11) holds for logistic regression under the global null. As we will see empirically in Section 4, as the dimensionality increases, p-values from logistic regression under the global null have a distribution that is skewed more and more toward zero. Consequently, classical hypothesis testing methods which reject the null hypothesis when p-value is less than the pre-specified level α would result in more false discoveries than the desired level of α. As a result, practitioners may simply lose the theoretical backup and the resulting decisions based on the p-values can become ineffective or even misleading. For this reason, it is important and helpful to identify the breakdown point of p-values in diverging-dimensional logistic regression model under the global null.

Characterizing the breakdown point of p-values in nonlinear GLMs is highly nontrivial and challenging. First, the nonlinearity generally causes the MLE to take no analytical form, which makes it difficult to analyze its behavior in diverging dimensions. Second, conventional probabilistic arguments for establishing the central limit theorem of MLE only enable us to see when the distributional property holds, but not exactly at what point it fails. To address these important challenges, we introduce novel geometric and probabilistic arguments presented later in the proofs of Theorems 2–3 that provide a rather delicate analysis of the MLE. In particular, our arguments unveil that the early breakdown point of p-values in nonlinear GLMs is essentially due to the *nonlinearity* of the mean function $\boldsymbol{\mu}(\cdot)$. This shows that p-values can behave wildly much early on in diverging dimensions when we move away from linear regression model to nonlinear regression models such as the widely applied logistic regression; see the Introduction for detailed discussions on the p-values in diverging-dimensional linear models.

Before presenting the main results, let us look at the specific case of logistic regression model under the global null. In such a scenario, it holds that $\boldsymbol{\theta}_0 = \mathbf{X}\boldsymbol{\beta}_0 = \mathbf{0}$ and thus

$\boldsymbol{\Sigma}(\boldsymbol{\theta}_0) = 4^{-1}I_n$, which results in

$$\mathbf{A}_n = \mathbf{X}^T\boldsymbol{\Sigma}(\boldsymbol{\theta}_0)\mathbf{X} = 4^{-1}\mathbf{X}^T\mathbf{X}.$$

In particular, we see that when $n^{-1}\mathbf{X}^T\mathbf{X}$ is close to the identity matrix I_p, the asymptotic standard deviation of the jth component $\widehat{\beta}_j$ of the MLE $\widehat{\boldsymbol{\beta}}$ is close to $2n^{-1/2}$ when the asymptotic theory in (11) holds. As mentioned in the Introduction, when $p \geq n/2$ the MLE can blow up with excessively large variance, a strong evidence against the distributional property in (11). In fact, one can also observe inflated variance of the MLE relative to what is predicted by the asymptotic theory in (11) even when the dimensionality p grows at a slower rate with sample size n. As a consequence, the conventional p-values given by algorithms according to property (11) can be much biased toward zero and thus produce more significant discoveries than the truth. Such a breakdown of conventional p-values is delineated clearly in the simulation examples presented in Section 4.

3.2. Main results

We now present the formal results on the invalidity of GLM p-values in diverging dimensions.

Theorem 2 (Uniform orthonormal design) [1] *Assume that $n^{-1/2}\mathbf{X}$ is uniformly distributed on the Stiefel manifold $V_p(\mathbb{R}^n)$ consisting of all $n \times p$ orthonormal matrices. Then for the logistic regression model under the global null, the asymptotic normality of the MLE established in (11) fails to hold when $p \sim n^{2/3}$, where $\sim$ stands for asymptotic order.*

Theorem 3 (Correlated Gaussian design) *Assume that $\mathbf{X} \sim N(\mathbf{0}, I_n \otimes \boldsymbol{\Sigma})$ with covariance matrix $\boldsymbol{\Sigma}$ nonsingular. Then for the logistic regression model under the global null, the same conclusion as in Theorem 2 holds.*

Theorem 4 in Appendix A states that under the global null in GLM with Gaussian design, the p-value based on the MLE remains valid as long as the dimensionality p diverges with n at a slower rate than $n^{2/3}$. This together with Theorems 2 and 3 shows that under the global null, the exact breakdown point for the uniformity of p-value is $n^{2/3}$. We acknowledge that these results are mainly for theoretical interests because in practice one cannot check precisely whether the global null assumption holds or not. However, these results clearly suggest that in GLM with diverging dimensionality, one needs to be very cautious when using p-values based on the MLE.

The key ingredients of our new geometric and probabilistic arguments are demonstrated in the proof of Theorem 2 in Section B.3. The assumption that the rescaled random design matrix $n^{-1/2}\mathbf{X}$ has the Haar measure on the Stiefel manifold $V_p(\mathbb{R}^n)$ greatly facilitates our technical analysis. The major theoretical finding is that the nonlinearity of the mean function $\boldsymbol{\mu}(\cdot)$ can be negligible in determining the asymptotic distribution of MLE as given in (11) when the dimensionality p grows at a slower rate than $n^{2/3}$, but such nonlinearity can become dominant and deform the conventional asymptotic normality when p grows at rate $n^{2/3}$ or faster. See the last paragraph of Section B.3 for more detailed in-depth discussions

1. For completeness, we present Theorem 4 in Appendix A which provides a random design version of Theorem 1 under global null and a partial converse of Theorems 2 and 3.

on such an interesting phenomenon. Furthermore, the global null assumption is a crucial component of our geometric and probabilistic argument. The global null assumption along with the distributional assumption on the design matrix ensures the symmetry property of the MLE and the useful fact that the MLE can be asymptotically independent of the random design matrix. In the absence of such an assumption, we may suspect that p-values of the noise variables can be affected by the signal variables due to asymmetry. Indeed, our simulation study in Section 4 reveals that as the number of signal variables increases, the breakdown point of the p-values occurs even earlier.

Theorem 3 further establishes that the invalidity of GLM p-values in high dimensions beyond the scenario of orthonormal design matrices considered in Theorem 2. The breakdown of the conventional p-values occurs regardless of the correlation structure of the covariates.

Our theoretical derivations detailed in the Appendix also suggest that the conventional p-values in nonlinear GLMs can generally fail to be valid when $p \sim n^{\alpha_0}$ with α_0 ranging between 1/2 and 2/3, which differs significantly from the phenomenon for linear models as discussed in the Introduction. The special feature of logistic regression model that the variance function $b''(\theta)$ takes the maximum value 1/4 at natural parameter $\theta = 0$ leads to a higher transition point of $p \sim n^{\alpha_0}$ with $\alpha_0 = 2/3$ for the case of global null $\boldsymbol{\beta}_0 = \mathbf{0}$.

4. Numerical Studies

We now investigate the breakdown point of p-values for nonlinear GLMs in diverging dimensions as predicted by our major theoretical results in Section 3 with several simulation examples. Indeed, these theoretical results are well supported by the numerical studies.

4.1. Simulation examples

Following Theorems 2–3 in Section 3, we consider three examples of the logistic regression model (1). The response vector $\mathbf{y} = (y_1, \cdots, y_n)^T$ has independent components and each y_i has Bernoulli distribution with parameter $e^{\theta_i}/(1+e^{\theta_i})$, where $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_n)^T = \mathbf{X}\boldsymbol{\beta}_0$. In example 1, we generate the $n \times p$ design matrix $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_p)$ such that $n^{-1/2}\mathbf{X}$ is uniformly distributed on the Stiefel manifold $V_p(\mathbb{R}^n)$ as in Theorem 2, while examples 2 and 3 assume that $\mathbf{X} \sim N(\mathbf{0}, I_n \otimes \boldsymbol{\Sigma})$ with covariance matrix $\boldsymbol{\Sigma}$ as in Theorem 3. In particular, we choose $\boldsymbol{\Sigma} = (\rho^{|j-k|})_{1 \le j,k \le p}$ with $\rho = 0, 0.5$, and 0.8 to reflect low, moderate, and high correlation levels among the covariates. Moreover, examples 1 and 2 assume the global null model with $\boldsymbol{\beta}_0 = \mathbf{0}$ following our theoretical results, whereas example 3 allows sparsity $s = \|\boldsymbol{\beta}_0\|_0$ to vary.

To examine the asymptotic results we set the sample size $n = 1000$. In each example, we consider a spectrum of dimensionality p with varying rate of growth with sample size n. As mentioned in the Introduction, the phase transition of perfect hyperplane separation happens at the point $p/n = 1/2$. Recall that Theorems 2–3 establish that the conventional GLM p-values can become invalid when $p \sim n^{2/3}$. We set $p = [n^{\alpha_0}]$ with α_0 in the grid $\{2/3 - 4\delta, \cdots, 2/3 - \delta, 2/3, 2/3 + \delta, \cdots, 2/3 + 4\delta, (\log(n) - \log(2))/\log(n)\}$ for $\delta = 0.05$. For example 3, we pick s signals uniformly at random among all but the first components, where a random half of them are chosen as 3 and the other half are set as -3.

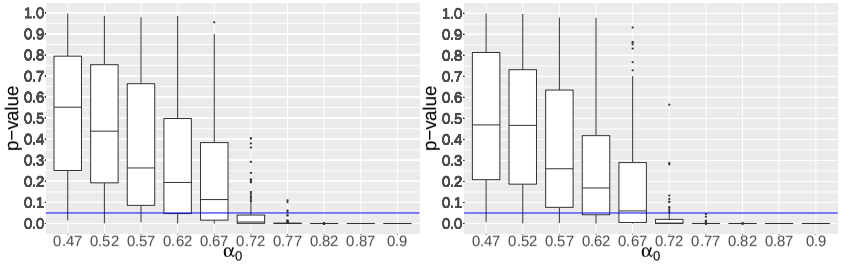
The goal of the simulation examples is to investigate empirically when the conventional GLM p-values could break down in diverging dimensions. When the asymptotic theory for

the MLE in (11) holds, the conventional p-values would be valid and distributed uniformly on the interval $[0, 1]$ under the null hypothesis. Note that the first covariate $\mathbf{x}_1$ is a null variable in each simulation example. Thus in each replication, we calculate the conventional p-value for testing the null hypothesis $H_0 : \beta_{0,1} = 0$. To check the validity of these p-values, we further test their uniformity.

For each simulation example, we first calculate the p-values for a total of $1,000$ replications as described above and then test the uniformity of these $1,000$ p-values using, for example, the Kolmogorov–Smirnov (KS) test (Kolmogorov, 1933; Smirnov, 1948) and the Anderson–Darling (AD) test (Anderson and Darling, 1952, 1954). We repeat this procedure 100 times to obtain a final set of 100 new p-values from each of these two uniformity tests. Specifically, the KS and AD test statistics for testing the uniformity on $[0, 1]$ are defined as

$$\mathrm{KS} = \sup_{x \in [0,1]} |F_m(x) - x| \quad \text{and} \quad \mathrm{AD} = m \int_0^1 \frac{[F_m(x) - x]^2}{x(1-x)}\, dx,$$

respectively, where $F_m(x) = m^{-1} \sum_{i=1}^m I_{(-\infty, x]}(x_i)$ is the empirical distribution function for a given sample $\{x_i\}_{i=1}^m$.

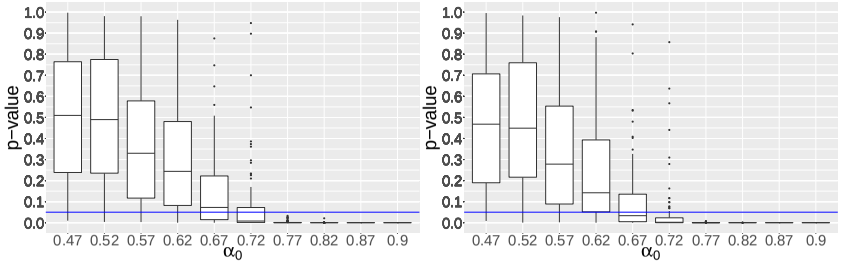


(a) KS test (b) AD test

Figure 1: Results of KS and AD tests for testing the uniformity of GLM p-values in simulation example 1 for diverging-dimensional logistic regression model with uniform orthonormal design under global null. The vertical axis represents the p-value from the KS and AD tests, and the horizontal axis stands for the growth rate α_0 of dimensionality $p = [n^{\alpha_0}]$.

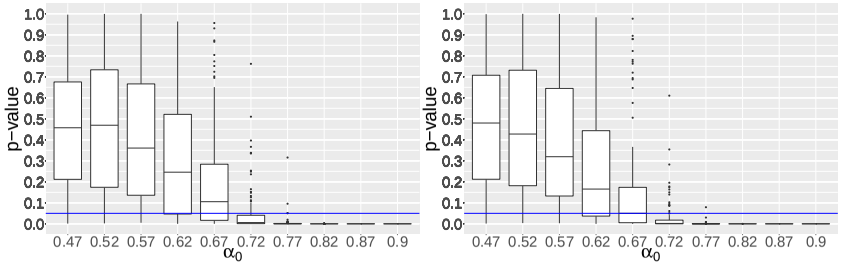
4.2. Testing results

For each simulation example, we apply both KS and AD tests to verify the asymptotic theory for the MLE in (11) by testing the uniformity of conventional p-values at significance level 0.05. As mentioned in Section 4.1, we end up with two sets of 100 new p-values from the KS and AD tests. Figures 1–3 depict the boxplots of the p-values obtained from both KS and AD tests for simulation examples 1–3, respectively. In particular, we observe that the numerical results shown in Figures 1–2 for examples 1–2 are in line with our theoretical results established in Theorems 2–3, respectively, for diverging-dimensional logistic



(a) KS test for $\rho = 0.5$

(b) AD test for $\rho = 0.5$



(c) KS test for $\rho = 0.8$

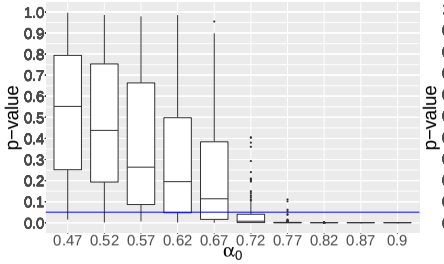
(d) AD test for $\rho = 0.8$

Figure 2: Results of KS and AD tests for testing the uniformity of GLM p-values in simulation example 2 for diverging-dimensional logistic regression model with correlated Gaussian design under global null for varying correlation level ρ. The vertical axis represents the p-value from the KS and AD tests, and the horizontal axis stands for the growth rate α_0 of dimensionality $p = [n^{\alpha_0}]$.

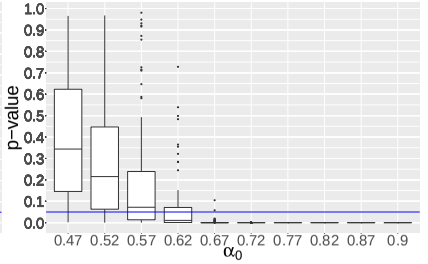
regression model under global null that the conventional p-values break down when $p \sim n^{\alpha_0}$ with $\alpha_0 = 2/3$. Figure 3 for example 3 examines the breakdown point of p-values with varying sparsity s. It is interesting to see that the breakdown point shifts even earlier when s increases as suggested in the discussions in Section 3.2. The results from the AD test are similar so we present only the results from the KS test for simplicity.

To gain further insights into the nonuniformity of the null p-values, we next provide an additional figure in the setting of simulation example 1. Specifically, in Figure 4 we present the histograms of the 1,000 null p-values from the first simulation repetition (out of 100) for each value of α_0. It is seen that as the dimensionality increases (i.e., α_0 increases), the null p-values have a distribution that is skewed more and more toward zero, which is prone to produce more false discoveries if these p-values are used naively in classical hypothesis testing methods.

To further demonstrate the severity of the problem, we estimate the probability of making type I error at significance level a, as the fraction of p-values below a. The means



(a) KS test for $s = 0$



(b) KS test for $s = 2$

Figure 3: Results of KS test for testing the uniformity of GLM p-values in simulation example 3 for diverging-dimensional logistic regression model with uncorrelated Gaussian design under global null for varying sparsity s. The vertical axis represents the p-value from the KS test, and the horizontal axis stands for the growth rate α_0 of dimensionality $p = [n^{\alpha_0}]$.

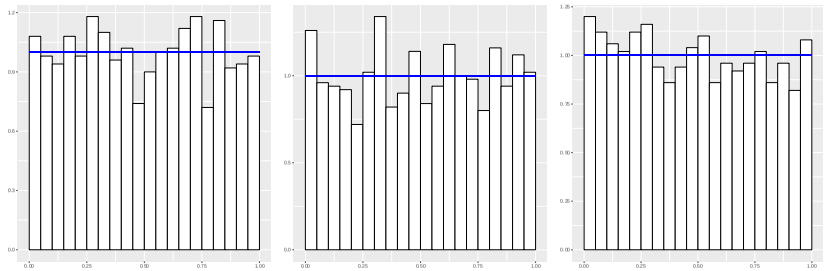
and standard deviations of the estimated probabilities are reported in Table 1 for $a = 0.05$ and 0.1. When the null p-values are distributed uniformly, the probabilities of making type I error should all be close to the target level a. However, Table 1 shows that when the growth rate of dimensionality α_0 approaches or exceeds $2/3$, these probabilities can be much larger than a, which again supports our theoretical findings. Also it is seen that when α_0 is close to but still smaller than $2/3$, the averages of estimated probabilities exceed slightly a, which could be the effect of finite sample size.

	α_0	0.10	0.47	0.57	0.67	0.77	0.87
$a = 0.05$	Mean	0.050	0.052	0.055	0.063	0.082	0.166
	SD	0.006	0.007	0.007	0.007	0.001	0.011
$a = 0.1$	Mean	0.098	0.104	0.107	0.118	0.144	0.247
	SD	0.008	0.010	0.009	0.011	0.012	0.013

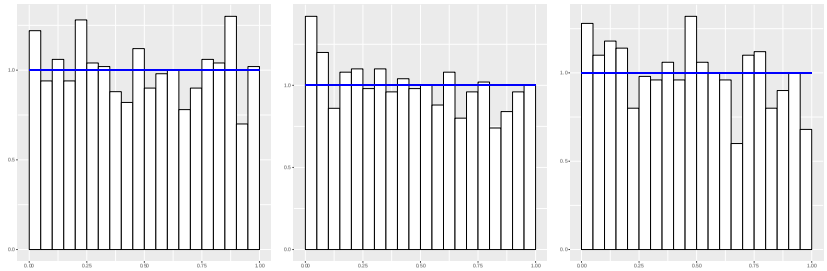
Table 1: Means and standard deviations (SD) for estimated probabilities of making type I error in simulation example 1 with α_0 the growth rate of dimensionality $p = [n^{\alpha_0}]$. Two significance levels $a = 0.05$ and 0.1 are considered.

5. Discussions

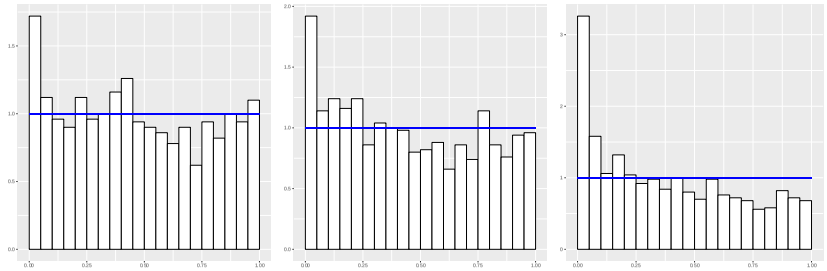
In this paper we have provided characterizations of p-values in nonlinear GLMs with diverging dimensionality. The major findings are that the conventional p-values can remain valid when $p = o(n^{1/2})$, but can become invalid much earlier in nonlinear models than in linear models, where the latter case can allow for $p = o(n)$. In particular, our theoretical results



(a) $\alpha_0 = 0.47$ (b) $\alpha_0 = 0.52$ (c) $\alpha_0 = 0.57$



(d) $\alpha_0 = 0.62$ (e) $\alpha_0 = 0.67$ (f) $\alpha_0 = 0.72$



(g) $\alpha_0 = 0.77$ (h) $\alpha_0 = 0.82$ (i) $\alpha_0 = 0.87$

Figure 4: Histograms of null p-values in simulation example 1 from the first simulation repetition for different growth rates α_0 of dimensionality $p = [n^{\alpha_0}]$.

pinpoint the breakdown point of $p \sim n^{2/3}$ for p-values in diverging-dimensional logistic regression model under global null with uniform orthonormal design and correlated Gaussian design, as evidenced in the numerical results. It would be interesting to investigate such a phenomenon for more general class of random design matrices.

The problem of identifying the breakdown point of p-values becomes even more complicated and challenging when we move away from the setting of global null. Our technical analysis suggests that the breakdown point $p \sim n^{\alpha_0}$ can shift even earlier with α_0 ranging between $1/2$ and $2/3$. But the exact breakdown point can depend upon the number of signals s, the signal magnitude, and the correlation structure among the covariates in a rather complicated fashion. Thus more delicate mathematical analysis is needed to obtain the exact relationship. We leave such a problem for future investigation. Moving beyond the GLM setting will further complicate the theoretical analysis.

As we routinely produce p-values using algorithms, the phenomenon of nonuniformity of p-values occurring early in diverging dimensions unveiled in the paper poses useful cautions to researchers and practitioners when making decisions in real applications using results from p-value based methods. For instance, when testing the joint significance of covariates in diverging-dimensional nonlinear models, the effective sample size requirement should be checked before interpreting the testing results. Indeed, statistical inference in general high-dimensional nonlinear models is particularly challenging since obtaining accurate p-values is generally not easy. One possible route is to bypass the use of p-values in certain tasks including the false discovery rate (FDR) control; see, for example, Barber and Candès (2015); Candès et al. (2018); Fan et al. (2018) for some initial efforts made along this line.

Acknowledgments

This work was supported by NIH Grant 1R01GM131407-01, NSF CAREER Award DMS-1150318, a grant from the Simons Foundation, and Adobe Data Science Research Award. The first and last authors sincerely thank Emmanuel Candès for helpful discussions on this topic. The authors would like to thank the Associate Editor and referees for their valuable comments that helped improve the article substantially.

Appendix A. Conventional P-values in Low Dimensions under Random Design

Under the specific assumption of Gaussian design and global null $\boldsymbol{\beta}_0 = \mathbf{0}$, we can show that the asymptotic normality of MLE continues to hold without previous Conditions 1–2.

Theorem 4 *Assume that $\boldsymbol{\beta}_0 = \mathbf{0}$, the rows of $\mathbf{X}$ are i.i.d. from $N(\mathbf{0}, \boldsymbol{\Sigma})$, $b^{(5)}(\cdot)$ is uniformly bounded in its domain, and $\mathbf{y} - \boldsymbol{\mu}_0$ has uniformly sub-Gaussian components. Then if $p = O(n^{\alpha})$ with some $\alpha \in [0, 2/3)$, we have the componentwise asymptotic normality*

$$(\mathbf{A}_n^{-1})_{jj}^{-1/2} \widehat{\beta}_j \xrightarrow{\mathscr{D}} N(0, \phi),$$

where all the notation is the same as in (11).

Theorem 4 shows that the conclusions of Theorem 1 continue to hold for the case of random design and global null with the major difference that the dimensionality can be

pushed as far as $p \sim n^{2/3}$. The main reasons for presenting Theorem 4 under Gausssian design are twofold. First, Gaussian design is a widely used assumption in the literature. Second, our results on the nonuniformity of GLM p-values in diverging dimensions use geometric and probabilistic arguments which require random design setting; see Section 3 for more details. To contrast more accurately the two regimes and maintain self-contained theory, we have chosen to present Theorem 4 under Gaussian design. On the other hand, we would like to point out that Theorem 4 is not for practitioners who want to justify the usage of classical p-values. The global null assumption of $\boldsymbol{\beta}_0 = \mathbf{0}$ restricts the validity of Theorem 4 in many practical scenarios.

Appendix B. Proofs of Main Results

We provide the detailed proofs of Theorems 1–3 in this Appendix.

B.1. Proof of Theorem 1

To ease the presentation, we split the proof into two parts, where the first part locates the MLE $\widehat{\boldsymbol{\beta}}$ in an asymptotically shrinking neighborhood $\mathcal{N}_0$ of the true regression coefficient vector $\boldsymbol{\beta}_0$ with significant probability and the second part further establishes its asymptotic normality.

Part 1: Existence of a unique solution to score equation (4) in $\mathcal{N}_0$ under Condition 1 and probability bound (6). For simplicity, assume that the design matrix $\mathbf{X}$ is rescaled columnwise such that $\|\mathbf{x}_j\|_2 = \sqrt{n}$ for each $1 \le j \le p$. Consider an event

$$\mathcal{E} = \left\{ \|\boldsymbol{\xi}\|_\infty \le c_1^{-1/2} \sqrt{n \log n} \right\}, \tag{12}$$

where $\boldsymbol{\xi} = (\xi_1, \cdots, \xi_p)^T = \mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}(\boldsymbol{\theta}_0)]$. Note that for unbounded responses, the assumption of $\max_{j=1}^p \|\mathbf{x}_j\|_\infty < c_1^{1/2}\{n/(\log n)\}^{1/2}$ in Condition 1 entails that $c_1^{-1/2}\sqrt{\log n} < \min_{j=1}^p \{\|\mathbf{x}_j\|_2/\|\mathbf{x}_j\|_\infty\}$. Thus by $\|\mathbf{x}_j\|_2 = \sqrt{n}$, probability bound (6), and Bonferroni's inequality, we deduce

$$\begin{aligned} P(\mathcal{E}) &\ge 1 - \sum_{j=1}^p P\left(|\xi_j| > c_1^{-1/2} \sqrt{n \log n} \right) \\ &\ge 1 - 2pn^{-1} = 1 - O\{n^{-(1-\alpha_0)}\}, \end{aligned} \tag{13}$$

since $p = O(n^{\alpha_0})$ for some $\alpha_0 \in (0, \gamma)$ with $\gamma \in (0, 1/2]$ by assumption. Hereafter we condition on the event $\mathcal{E}$ defined in (12) which holds with significant probability.

We will show that for sufficiently large n, the score equation (4) has a solution in the neighborhood $\mathcal{N}_0$ which is a hypercube. Define two vector-valued functions

$$\boldsymbol{\gamma}(\boldsymbol{\beta}) = (\gamma_1(\boldsymbol{\beta}), \cdots, \gamma_p(\boldsymbol{\beta}))^T = \mathbf{X}^T \boldsymbol{\mu}(\mathbf{X}\boldsymbol{\beta})$$

and

$$\boldsymbol{\Psi}(\boldsymbol{\beta}) = \boldsymbol{\gamma}(\boldsymbol{\beta}) - \boldsymbol{\gamma}(\boldsymbol{\beta}_0) - \boldsymbol{\xi}, \quad \boldsymbol{\beta} \in \mathbb{R}^p.$$

Then equation (4) is equivalent to $\boldsymbol{\Psi}(\boldsymbol{\beta}) = \mathbf{0}$. We need to show that the latter has a solution inside the hypercube $\mathcal{N}_0$. To this end, applying a second order Taylor expansion of $\boldsymbol{\gamma}(\boldsymbol{\beta})$ around $\boldsymbol{\beta}_0$ with the Lagrange remainder term componentwise leads to

$$\boldsymbol{\gamma}(\boldsymbol{\beta}) = \boldsymbol{\gamma}(\boldsymbol{\beta}_0) + \mathbf{X}^T \boldsymbol{\Sigma}(\boldsymbol{\theta}_0) \mathbf{X}(\boldsymbol{\beta} - \boldsymbol{\beta}_0) + \mathbf{r}, \tag{14}$$

where $\mathbf{r} = (r_1, \cdots, r_p)^T$ and for each $1 \le j \le p$,

$$r_j = \frac{1}{2} (\boldsymbol{\beta} - \boldsymbol{\beta}_0)^T \nabla^2 \gamma_j(\boldsymbol{\beta}_j) (\boldsymbol{\beta} - \boldsymbol{\beta}_0)$$

with $\boldsymbol{\beta}_j$ some p-dimensional vector lying on the line segment joining $\boldsymbol{\beta}$ and $\boldsymbol{\beta}_0$. It follows from (9) in Condition 1 that

$$\begin{aligned} \|\mathbf{r}\|_\infty &\le \max_{\boldsymbol{\delta} \in \mathcal{N}_0} \max_{j=1}^p \frac{1}{2} \lambda_{\max} \left[\mathbf{X}^T \operatorname{diag}\left\{ |\mathbf{x}_j| \circ \left| \boldsymbol{\mu}''(\mathbf{X}\boldsymbol{\delta}) \right| \right\} \mathbf{X} \right] \|\boldsymbol{\beta} - \boldsymbol{\beta}_0\|_2^2 \\ &= O\left\{ pn^{1-2\gamma} (\log n)^2 \right\}. \end{aligned} \tag{15}$$

Let us define another vector-valued function

$$\overline{\boldsymbol{\Psi}}(\boldsymbol{\beta}) \equiv \left[\mathbf{X}^T \boldsymbol{\Sigma}(\boldsymbol{\theta}_0) \mathbf{X} \right]^{-1} \boldsymbol{\Psi}(\boldsymbol{\beta}) = \boldsymbol{\beta} - \boldsymbol{\beta}_0 + \mathbf{u}, \tag{16}$$

where $\mathbf{u} = -[\mathbf{X}^T \boldsymbol{\Sigma}(\boldsymbol{\theta}_0) \mathbf{X}]^{-1} (\boldsymbol{\xi} - \mathbf{r})$. It follows from (12), (15), and (8) in Condition 1 that for any $\boldsymbol{\beta} \in \mathcal{N}_0$,

$$\begin{aligned} \|\mathbf{u}\|_\infty &\le \left\| \left[\mathbf{X}^T \boldsymbol{\Sigma}(\boldsymbol{\theta}_0) \mathbf{X} \right]^{-1} \right\|_\infty (\|\boldsymbol{\xi}\|_\infty + \|\mathbf{r}\|_\infty) \\ &= O\left[b_n n^{-1/2} \sqrt{\log n} + b_n p n^{-2\gamma} (\log n)^2 \right]. \end{aligned} \tag{17}$$

By the assumptions of $p = O(n^{\alpha_0})$ with constant $\alpha_0 \in (0, \gamma)$ and $b_n = o\{\min(n^{1/2-\gamma}\sqrt{\log n}, n^{2\gamma - \alpha_0 - 1/2}/(\log n)^2\}$, we have

$$\|\mathbf{u}\|_\infty = o(n^{-\gamma} \log n).$$

Thus in light of (16), it holds for large enough n that when $(\boldsymbol{\beta} - \boldsymbol{\beta}_0)_j = n^{-\gamma}\sqrt{\log n}$,

$$\overline{\Psi}_j(\boldsymbol{\beta}) \ge n^{-\gamma} \sqrt{\log n} - \|\mathbf{u}\|_\infty \ge 0, \tag{18}$$

and when $(\boldsymbol{\beta} - \boldsymbol{\beta}_0)_j = -n^{-\gamma}\sqrt{\log n}$,

$$\overline{\Psi}_j(\boldsymbol{\beta}) \le -n^{-\gamma} \sqrt{\log n} + \|\mathbf{u}\|_\infty \le 0, \tag{19}$$

where $\overline{\boldsymbol{\Psi}}(\boldsymbol{\beta}) = (\overline{\Psi}_1(\boldsymbol{\beta}), \cdots, \overline{\Psi}_p(\boldsymbol{\beta}))^T$.

By the continuity of the vector-valued function $\overline{\boldsymbol{\Psi}}(\boldsymbol{\beta})$, (18), and (19), Miranda's existence theorem Vrahatis (1989) ensures that equation $\overline{\boldsymbol{\Psi}}(\boldsymbol{\beta}) = \mathbf{0}$ has a solution $\widehat{\boldsymbol{\beta}}$ in $\mathcal{N}_0$. Clearly, $\widehat{\boldsymbol{\beta}}$ also solves equation $\boldsymbol{\Psi}(\boldsymbol{\beta}) = \mathbf{0}$ in view of (16). Therefore, we have shown that score equation (4) indeed has a solution $\widehat{\boldsymbol{\beta}}$ in $\mathcal{N}_0$. The strict concavity of the log-likelihood function (2) by assumptions for model (1) entails that $\widehat{\boldsymbol{\beta}}$ is the MLE.

Part 2: Conventional asymptotic normality of the MLE $\widehat{\boldsymbol{\beta}}$. Fix any $1 \le j \le p$. In light of (16), we have $\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}_0 = \mathbf{A}_n^{-1}(\boldsymbol{\xi} - \mathbf{r})$, which results in

$$(\mathbf{A}_n^{-1})_{jj}^{-1/2}(\widehat{\beta}_j - \beta_{0,j}) = (\mathbf{A}_n^{-1})_{jj}^{-1/2}\mathbf{e}_j^T\mathbf{A}_n^{-1}\boldsymbol{\xi} - (\mathbf{A}_n^{-1})_{jj}^{-1/2}\mathbf{e}_j^T\mathbf{A}_n^{-1}\mathbf{r} \tag{20}$$

with $\mathbf{e}_j \in \mathbb{R}^p$ having one for the jth component and zero otherwise. Note that since the smallest and largest eigenvalues of $n^{-1}\mathbf{A}_n$ are bounded away from 0 and ∞ by Condition 2, it is easy to show that $(\mathbf{A}_n^{-1})_{jj}^{-1/2}$ is of exact order $n^{1/2}$. In view of (17), it holds on the event $\mathcal{E}$ defined in (12) that

$$\begin{aligned} \left\|\mathbf{A}_n^{-1}\mathbf{r}\right\|_\infty &\le \left\|\left[\mathbf{X}^T\boldsymbol{\Sigma}\left(\boldsymbol{\theta}_0\right)\mathbf{X}\right]^{-1}\right\|_\infty \|\mathbf{r}\|_\infty \\ &= O\left[b_n p n^{-2\gamma}(\log n)^2\right] = o(n^{-1/2}), \end{aligned}$$

since $b_n = o\{n^{2\gamma-\alpha_0-1/2}/(\log n)^2\}$ by assumption. This leads to

$$(\mathbf{A}_n^{-1})_{jj}^{-1/2}\mathbf{e}_j^T\mathbf{A}_n^{-1}\mathbf{r} = O(n^{1/2})\cdot o_P(n^{-1/2}) = o_P(1). \tag{21}$$

It remains to consider the term $(\mathbf{A}_n^{-1})_{jj}^{-1/2}\mathbf{e}_j^T\mathbf{A}_n^{-1}\boldsymbol{\xi} = \sum_{i=1}^n \eta_i$, where $\eta_i = (\mathbf{A}_n^{-1})_{jj}^{-1/2}\mathbf{e}_j^T\mathbf{A}_n^{-1}\mathbf{z}_i[y_i - b'(\theta_{0,i})]$. Clearly, the n random variables η_i's are independent with mean 0 and

$$\sum_{i=1}^n \mathrm{var}(\eta_i) = (\mathbf{A}_n^{-1})_{jj}^{-1}\mathbf{e}_j^T\mathbf{A}_n^{-1}(\phi\mathbf{A}_n)\mathbf{A}_n^{-1}\mathbf{e}_j = \phi.$$

It follows from Condition 2 and the Cauchy–Schwarz inequality that

$$\begin{aligned} \sum_{i=1}^n E\left|\eta_i\right|^3 &= \sum_{i=1}^n \left|(\mathbf{A}_n^{-1})_{jj}^{-1/2}\mathbf{e}_j^T\mathbf{A}_n^{-1}\mathbf{z}_i\right|^3 E\left|y_i - b'\left(\theta_{0,i}\right)\right|^3 \\ &= O(1)\sum_{i=1}^n \left|(\mathbf{A}_n^{-1})_{jj}^{-1/2}\mathbf{e}_j^T\mathbf{A}_n^{-1}\mathbf{z}_i\right|^3 \\ &\le O(1)\sum_{i=1}^n \left\|(\mathbf{A}_n^{-1})_{jj}^{-1/2}\mathbf{e}_j^T\mathbf{A}_n^{-1/2}\right\|_2^3 \left\|\mathbf{A}_n^{-1/2}\mathbf{z}_i\right\|_2^3 \\ &= O(1)\sum_{i=1}^n \left(\mathbf{z}_i^T\mathbf{A}_n^{-1}\mathbf{z}_i\right)^{3/2} = o(1). \end{aligned}$$

Thus an application of Lyapunov's theorem yields

$$(\mathbf{A}_n^{-1})_{jj}^{-1/2}\mathbf{e}_j^T\mathbf{A}_n^{-1}\boldsymbol{\xi} = \sum_{i=1}^n \eta_i \xrightarrow{\mathscr{D}} N(0,\phi). \tag{22}$$

By Slutsky's lemma, we see from (20)–(22) that

$$(\mathbf{A}_n^{-1})_{jj}^{-1/2}(\widehat{\beta}_j - \beta_{0,j}) \xrightarrow{\mathscr{D}} N(0,\phi),$$

showing the asymptotic normality of each component $\widehat{\beta}_j$ of the MLE $\widehat{\boldsymbol{\beta}}$.

We further establish the asymptotic normality for the one-dimensional projections of the MLE $\widehat{\boldsymbol{\beta}}$. Fix an arbitrary vector $\mathbf{u} \in \mathbb{R}^p$ with $\|\mathbf{u}\|_2 = 1$ satisfying the L_1 sparsity bound $\|\mathbf{u}\|_1 = O(s_n)$. In light of (16), we have $\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}_0 = \mathbf{A}_n^{-1}(\boldsymbol{\xi} - \mathbf{r})$, which results in

$$(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}(\mathbf{u}^T\widehat{\boldsymbol{\beta}} - \mathbf{u}^T\boldsymbol{\beta}_0) = (\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}\mathbf{u}^T\mathbf{A}_n^{-1}\boldsymbol{\xi} - (\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{r}. \tag{23}$$

Note that since the smallest and largest eigenvalues of $n^{-1}\mathbf{A}_n$ are bounded away from 0 and ∞ by Condition 2, it is easy to show that $(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}$ is of exact order $n^{1/2}$. In view of (17), it holds on the event $\mathcal{E}$ defined in (12) that

$$\begin{aligned} \left\|\mathbf{A}_n^{-1}\mathbf{r}\right\|_\infty &\le \left\|\left[\mathbf{X}^T\boldsymbol{\Sigma}\left(\boldsymbol{\theta}_0\right)\mathbf{X}\right]^{-1}\right\|_\infty \|\mathbf{r}\|_\infty \\ &= O\left\{b_n p n^{-2\gamma}(\log n)^2\right\} = o(s_n^{-1}n^{-1/2}) \end{aligned}$$

since $b_n = o\{s_n^{-1}n^{2\gamma-\alpha_0-1/2}/(\log n)^2\}$ by assumption. This leads to

$$(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{r} = O(n^{1/2})\cdot\|\mathbf{u}\|_1\cdot\|\mathbf{A}_n^{-1}\mathbf{r}\|_\infty = o_P(1) \tag{24}$$

since $\|\mathbf{u}\|_1 = O(s_n)$ by assumption.

It remains to consider the term $(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}\mathbf{u}^T\mathbf{A}_n^{-1}\boldsymbol{\xi} = \sum_{i=1}^n \eta_i$ with $\eta_i = (\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{z}_i[y_i - b'(\theta_{0,i})]$. Clearly, the n random variables η_i's are independent with mean 0 and

$$\sum_{i=1}^n \mathrm{var}(\eta_i) = (\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1}\mathbf{u}^T\mathbf{A}_n^{-1}(\phi\mathbf{A}_n)\mathbf{A}_n^{-1}\mathbf{u} = \phi.$$

It follows from Condition 2 and the Cauchy–Schwarz inequality that

$$\begin{aligned} \sum_{i=1}^n E\left|\eta_i\right|^3 &= \sum_{i=1}^n \left|(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{z}_i\right|^3 E\left|y_i - b'\left(\theta_{0,i}\right)\right|^3 \\ &= O(1)\sum_{i=1}^n \left|(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{z}_i\right|^3 \\ &\le O(1)\sum_{i=1}^n \left\|(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}\mathbf{u}^T\mathbf{A}_n^{-1/2}\right\|_2^3 \left\|\mathbf{A}_n^{-1/2}\mathbf{z}_i\right\|_2^3 \\ &= O(1)\sum_{i=1}^n \left(\mathbf{z}_i^T\mathbf{A}_n^{-1}\mathbf{z}_i\right)^{3/2} = o(1). \end{aligned}$$

Thus an application of Lyapunov's theorem yields

$$(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}\mathbf{u}^T\mathbf{A}_n^{-1}\boldsymbol{\xi} = \sum_{i=1}^n \eta_i \xrightarrow{\mathscr{D}} N(0,\phi). \tag{25}$$

By Slutsky's lemma, we see from (23)–(25) that

$$(\mathbf{u}^T\mathbf{A}_n^{-1}\mathbf{u})^{-1/2}(\mathbf{u}^T\widehat{\beta} - \mathbf{u}^T\boldsymbol{\beta}_0) \xrightarrow{\mathscr{D}} N(0,\phi),$$

showing the asymptotic normality of any L_1-sparse one-dimensional projection $\mathbf{u}^T\widehat{\boldsymbol{\beta}}$ of the MLE $\widehat{\boldsymbol{\beta}}$. This completes the proof of Theorem 1.

B.2. Proof of Theorem 4

The proof is similar to that for Theorem 1. Without loss of generality, we assume that $\mathbf{\Sigma} = I_p$ because under global null, a rotation of $\mathbf{X}$ yields standard normal rows. First let $\boldsymbol{\xi} = (\xi_1, \cdots, \xi_p)^T = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}_0]$, where $\boldsymbol{\mu}_0 = b'(0)\mathbf{1}$ with $\mathbf{1} = (1, \cdots, 1)^T \in \mathbb{R}^n$ because $\boldsymbol{\beta}_0 = 0$. Then $\mathbf{y} - \boldsymbol{\mu}_0$ has i.i.d. uniform sub-Gaussian components and is independent of $\mathbf{X} = (\mathbf{z}_1, \cdots, \mathbf{z}_p) \in \mathbb{R}^{n\times p}$. Define event

$$\mathcal{E} = \{\|\boldsymbol{\xi}\|_\infty \le c_2\sqrt{n^{-1}\log n}\}.$$

By Lemma 5, it is seen that $P(\mathcal{E}) \ge 1 - o(p^{-a})$. Furthermore, define the neighborhood

$$\mathcal{N}_0 = \{\|\boldsymbol{\beta}\|_\infty \le c_3\sqrt{n^{-1}\log n}\} \tag{26}$$

for some $c_3 > c_2(b''(0))^{-1}$. We next show that the MLE must fall into the region $\mathcal{N}_0$ with probability at least $1 - O(p^{-a})$ following the similar arguments in Theorem 1.

First, we define

$$\boldsymbol{\gamma}(\boldsymbol{\beta}) = (\gamma_1(\boldsymbol{\beta}), \cdots, \gamma_p(\boldsymbol{\beta}))^T \equiv \mathbf{X}^T\boldsymbol{\mu}(\mathbf{X}\boldsymbol{\beta})$$

and

$$\mathbf{\Psi}(\boldsymbol{\beta}) = \boldsymbol{\gamma}(\boldsymbol{\beta}) - \boldsymbol{\gamma}(\boldsymbol{\beta}_0) - \mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}_0], \quad \boldsymbol{\beta} \in \mathbb{R}^p.$$

Applying a forth order Taylor expansion of $\boldsymbol{\gamma}(\boldsymbol{\beta})$ around $\boldsymbol{\beta}_0 = \mathbf{0}$ with the Lagrange remainder term componentwise leads to

$$\boldsymbol{\gamma}(\boldsymbol{\beta}) = \boldsymbol{\gamma}(\boldsymbol{\beta}_0) + b''(0)\mathbf{X}^T\mathbf{X}(\boldsymbol{\beta} - \boldsymbol{\beta}_0) + \mathbf{r} + \mathbf{s} + \mathbf{t},$$

where $\mathbf{r} = (r_1, \cdots, r_p)^T$, $\mathbf{s} = (s_1, \cdots, s_p)^T$, $\mathbf{t} = (t_1, \cdots, t_p)^T$ and for each $1 \le j \le p$,

$$r_j = \frac{b'''(0)}{2}\sum_{i=1}^n x_{ij}(\mathbf{x}_i^T\boldsymbol{\beta})^2 \tag{27}$$

$$s_j = \frac{b^{(4)}(0)}{6}\sum_{i=1}^n x_{ij}(\mathbf{x}_i^T\boldsymbol{\beta})^3 \tag{28}$$

$$t_j = \frac{1}{24}\sum_{i=1}^n b^{(5)}(\mathbf{x}_i^T\widetilde{\boldsymbol{\beta}^j})x_{ij}(\mathbf{x}_i^T\boldsymbol{\beta})^4. \tag{29}$$

with $\widetilde{\boldsymbol{\beta}^j}$ some p-dimensional vector lying on the line segment joining $\boldsymbol{\beta}$ and $\boldsymbol{\beta}_0$.

Let us define another vector-valued function

$$\overline{\mathbf{\Psi}}(\boldsymbol{\beta}) \equiv \left[b''(0)\mathbf{X}^T\mathbf{X}\right]^{-1}\mathbf{\Psi}(\boldsymbol{\beta}) = \boldsymbol{\beta} - \boldsymbol{\beta}_0 + \mathbf{u}, \tag{30}$$

where $\mathbf{u} = -(b''(0))^{-1}\boldsymbol{\xi} + [b''(0)\mathbf{X}^T\mathbf{X}]^{-1}(\mathbf{r} + \mathbf{s} + \mathbf{t})$. It follows from the above derivation that for any $\boldsymbol{\beta} \in \mathcal{N}_0$,

$$\|\mathbf{u}\|_\infty \le \left\|(b''(0))^{-1}\boldsymbol{\xi}\right\|_\infty + \left\|\left[b''(0)\mathbf{X}^T\mathbf{X}\right]^{-1}(\mathbf{r} + \mathbf{s} + \mathbf{t})\right\|_\infty.$$

Now, we bound the terms on the right hand side.

First note that on event $\mathcal{E}$,

$$\left\|(b''(0))^{-1}\boldsymbol{\xi}\right\|_\infty \le (b''(0))^{-1}c_2\sqrt{n\log n}. \tag{31}$$

Then, we consider the next term: $\left\|\left[b''(0)\mathbf{X}^T\mathbf{X}\right]^{-1}(\mathbf{r} + \mathbf{s} + \mathbf{t})\right\|_\infty$. We observe that

$$\begin{aligned}\left\|\left[b''(0)\mathbf{X}^T\mathbf{X}\right]^{-1}(\mathbf{r} + \mathbf{s} + \mathbf{t})\right\|_\infty &\le |b''(0)|^{-1}\left\|\left[n^{-1}\mathbf{X}^T\mathbf{X}\right]^{-1}\right\|_\infty \left\|n^{-1}(\mathbf{r} + \mathbf{s} + \mathbf{t})\right\|_\infty \\ &\le |b''(0)|^{-1}\left\|\left[n^{-1}\mathbf{X}^T\mathbf{X}\right]^{-1}\right\|_\infty \\ &\quad \cdot \left(\left\|n^{-1}\mathbf{r}\right\|_\infty + \left\|n^{-1}\mathbf{s}\right\|_\infty + \left\|n^{-1}\mathbf{t}\right\|_\infty\right).\end{aligned}$$

By Lemma 6, we have that $\left\|\left[n^{-1}\mathbf{X}^T\mathbf{X}\right]^{-1}\right\|_\infty \le 1 + O(pn^{-1/2})$. Lemmas 10, 11, 12 assert that

$$\begin{aligned}&\left(\left\|n^{-1}\mathbf{r}\right\|_\infty + \left\|n^{-1}\mathbf{s}\right\|_\infty + \left\|n^{-1}\mathbf{t}\right\|_\infty\right) \\ &= \{n^{\alpha-5/6}\log n + n^{3/2\alpha-5/4}(\log n)^{3/2} + n^{\alpha-1}(\log n)^{1/2} + n^{2\alpha-3/2}(\log n)^{3/2}\}\sqrt{n^{-1}\log n}.\end{aligned}$$

We combine last two bounds so that we have

$$\left\|\left[b''(0)\mathbf{X}^T\mathbf{X}\right]^{-1}(\mathbf{r} + \mathbf{s} + \mathbf{t})\right\|_\infty = o(\sqrt{n^{-1}\log n}) \tag{32}$$

with probability at least $1 - o(p^{-c})$ when $p = O(n^\alpha)$ with $\alpha < 2/3$.

Combining equations (31) and (32), we obtain that if $p = O(n^\alpha)$ with $\alpha \in [0, 2/3)$, then

$$\|\mathbf{u}\|_\infty \le c_3\sqrt{n^{-1}\log n}.$$

Thus, the MLE must fall into the region $\mathcal{N}_0$ following the similar arguments in Theorem 1.

Next, we show the componentwise asymptotic normality of the MLE $\widehat{\boldsymbol{\beta}}$. By equation (30), we have $\widehat{\boldsymbol{\beta}} = -\mathbf{u} = (b''(0))^{-1}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}_0] - [b''(0)\mathbf{X}^T\mathbf{X}]^{-1}(\mathbf{r} + \mathbf{s} + \mathbf{t})$. So, we can write

$$\hat{\beta}_j = (b''(0))^{-1}n^{-1}\mathbf{e}_j^T\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}_0] + (b''(0))^{-1}T - \mathbf{e}_j^T[b''(0)\mathbf{X}^T\mathbf{X}]^{-1}(\mathbf{r} + \mathbf{s} + \mathbf{t}) \tag{33}$$

where $T = \mathbf{e}_j^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}_0] - n^{-1}\mathbf{e}_j^T\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}_0]$. By Lemma 13 and Equation (32), both $n^{1/2}(b''(0))^{-1}T$ and $n^{1/2}\mathbf{e}_j^T[b''(0)\mathbf{X}^T\mathbf{X}]^{-1}(\mathbf{r}+\mathbf{s}+\mathbf{t})$ converges to zero in probability. So, it is enough to consider the first summand in (33). Now, we show that $n^{-1/2}\mathbf{e}_j^T\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}_0]$ is asymptotically normal. In fact, we can write $\mathbf{e}_j^T\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}_0] = \sum_{i=1}^n x_{ij}y_i$ where each summand $x_{ij}y_i$ is independent over i and has variance $\phi b''(0)$. Moreover, $\sum_{i=1}^n E|x_{ij}y_i|^3 = O(n)$ since $|x_{ij}|^3$ and $|y_i|^3$ are independent and finite mean. So, we apply Lyapunov's theorem to obtain $b''(0)^{-1/2}n^{-1/2}\mathbf{e}_j^T\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}_0] \xrightarrow{\mathscr{D}} N(0, \phi)$. Finally, we know that $b''(0)n(\mathbf{A}_n^{-1})_{jj} \to 1$ in probability from the remark in Theorem 1. Thus, Slutsky's lemma yields

$$(\mathbf{A}_n^{-1})_{jj}^{-1/2}\hat{\beta}_j \xrightarrow{\mathscr{D}} N(0, \phi). \tag{34}$$

This completes the proof of the theorem.

Lemma 5 *Assume that the components of* $\mathbf{y}-\boldsymbol{\mu}_0$ *are uniform sub-Gaussians. That is, there exist a positive constant* C *such that* $P(|(\mathbf{y}-\boldsymbol{\mu}_0)_i|>t)\le C\exp\{-Ct^2\}$ *for all* $1\le i\le n$. *Then, it holds that, for some positive constant* c_2,

$$\|(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty \le c_2\sqrt{n^{-1}\log n}.$$

with asymptotic probability $1-o(p^{-a})$.

Proof We prove the result by conditioning on $\mathbf{X}$. Let $\mathbf{E}=n^{-1}\mathbf{X}^T\mathbf{X}-\mathbf{I}_p$. Then by matrix inversion,

$$\begin{aligned}(n^{-1}\mathbf{X}^T\mathbf{X})^{-1} &= (\mathbf{I}_p+\mathbf{E})^{-1} = \mathbf{I}_p-\sum_{k=1}^{\infty}(-1)^{k+1}(\mathbf{E})^k\\ &= \mathbf{I}_p-\mathbf{E}+\sum_{k=2}^{\infty}(-1)^k(\mathbf{E})^k = 2\mathbf{I}_p-n^{-1}\mathbf{X}^T\mathbf{X}+\sum_{k=2}^{\infty}(-1)^k(\mathbf{E})^k.\end{aligned}$$

Thus, it follows that

$$\begin{aligned}&\|(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty\\ &\le \|2n^{-1}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty+\|n^{-2}\mathbf{X}^T\mathbf{X}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty+\left\|n^{-1}\sum_{k=2}^{\infty}(-1)^k(\mathbf{E})^k\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\right\|_\infty\\ &= \eta_1+\eta_2+\eta_3.\end{aligned}$$

In the rest of the proof, we will bound η_1, η_2 and η_3.

Part 1: Bound of η_1.

First, it is easy to see that

$$\begin{aligned}\eta_1 &= \|2n^{-1}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty\\ &= 2\max_{1\le j\le p}\left|n^{-1}\sum_{i=1}^{n}x_{ij}(\mathbf{y}-\boldsymbol{\mu}_0)_i\right|.\end{aligned}$$

We observe that each summand $x_{ij}(\mathbf{y}-\boldsymbol{\mu}_0)_i$ is the product of two subgaussian random variables, and so satisfies $P(|x_{ij}(\mathbf{y}-\boldsymbol{\mu}_0)_i|>t)\le C\exp(-Ct)$ for some constant $C>0$ by Lemma 1 in Fan et al. (2016). Moreover, $E[x_{ij}(\mathbf{y}-\boldsymbol{\mu}_0)_i]=0$ since x_{ij} and $(\mathbf{y}-\boldsymbol{\mu}_0)_i$ are independent and have zero mean. Thus, we can use Lemma 9 by setting $W_{ij}=x_{ij}(\mathbf{y}-\boldsymbol{\mu}_0)_i$ and $\alpha=1$. So, we get

$$\eta_1 = 2\max_{1\le j\le p}\left|n^{-1}\sum_{i=1}^{n}x_{ij}(\mathbf{y}-\boldsymbol{\mu}_0)_i\right| \le c_2\sqrt{n^{-1}\log p} \tag{35}$$

with probability $1-O(p^{-c})$ for some positive constants c and c_2.

Part 2: Bound of η_2.

Now, we study $\eta_2=\|n^{-2}\mathbf{X}^T\mathbf{X}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty$. Let $\mathbf{z}_k$ be the k-th column of X, that is $\mathbf{z}_k=\mathbf{X}\mathbf{e}_k$. Direct calculations yield

$$\mathbf{e}_k^T\mathbf{X}^T\mathbf{X}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)=\sum_{j=1}^{p}(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0))=\|\mathbf{z}_k\|_2^2\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)+\sum_{j\ne k}^{p}(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0)).$$

Thus, it follows that

$$\left\|(\mathbf{X}^T\mathbf{X}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\right\|_\infty \le \max_k\left|\|\mathbf{z}_k\|_2^2\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right|+\max_k\left|\sum_{j\ne k}^{p}(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0))\right|. \tag{36}$$

First, we consier $\max_k\left|\|\mathbf{z}_k\|_2^2\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right|$. Lemma 14 shows that $\max_k\|\mathbf{z}_k\|_2^2\le O(n)$ with probability $1-O(p^{-c})$. We also have $\max_k\left|\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right|=\frac{n}{2}\eta_1\le O(\sqrt{n\log p})$ by equation (35). It follows that

$$\max_k\left|\|\mathbf{z}_k\|_2^2\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right| \le \max_k\|\mathbf{z}_k\|_2^2\max_k\left|\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right| \le O(n\sqrt{n\log p}). \tag{37}$$

Next, let $a_j=\mathbf{z}_k^T\mathbf{z}_j/\|\mathbf{z}_k\|_2$ and $b_j=\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0)/\|\mathbf{y}-\boldsymbol{\mu}_0\|_2$. Then it is easy to see that conditional on $\mathbf{z}_k$ and $\mathbf{y}$, $a_j\sim N(0,1)$, $b_j\sim N(0,1)$ and $\mathrm{cov}(a_j,b_j|\mathbf{z}_k,\mathbf{y})=\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)/(\|\mathbf{z}_k\|_2\|\mathbf{y}-\boldsymbol{\mu}_0\|_2)$. By (E.6) of Lemma 7 in Fan et al. (2016), it can be shown that

$$\begin{aligned}&P\left(\frac{1}{p-1}\left|\sum_{j\ne k}^{p}(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0))-\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right|\ge c\|\mathbf{z}_k\|_2\|\mathbf{y}-\boldsymbol{\mu}_0\|_2\sqrt{p^{-1}\log p}\Big|\mathbf{z}_k,\mathbf{y}\right)\\ &= P\left(\frac{1}{p-1}\left|\sum_{j\ne k}^{p}a_jb_j-\frac{\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)}{\|\mathbf{z}_k\|_2\|\mathbf{y}-\boldsymbol{\mu}_0\|_2}\right|\ge c\sqrt{p^{-1}\log p}\Big|\mathbf{z}_k,\mathbf{y}\right)\le cp^{-c_1},\end{aligned}$$

where c_1 is some large positive constant independent of $\mathbf{z}_k$ and $\mathbf{y}$. Moreover, we can choose c_1 as large as we want by increasing c. Thus, it follows that

$$P\left(\frac{1}{p-1}\left|\sum_{j\ne k}^{p}(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0))-\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right|\ge c\|\mathbf{z}_k\|_2\|\mathbf{y}-\boldsymbol{\mu}_0\|_2\sqrt{p^{-1}\log p}\right)\le cp^{-c_1}.$$

It follows from probability union bound that

$$P\left(\frac{1}{p-1}\max_k\left|\frac{1}{\|\mathbf{z}_k\|_2\|\mathbf{y}-\boldsymbol{\mu}_0\|_2}\sum_{j\ne k}^{p}(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0))-\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right|\ge c\sqrt{p^{-1}\log p}\right)\le cp^{-c_1+1}.$$

Taking $c_1>1$ yields that with probability at least $1-o(p^{-a})$ for some $a>0$,

$$\max_k\left\{\frac{1}{\|\mathbf{z}_k\|_2\|\mathbf{y}-\boldsymbol{\mu}_0\|_2}\left|\frac{1}{p-1}\sum_{j\ne k}^{p}(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0))-\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right|\right\}\le c\sqrt{p^{-1}\log p}.$$

By Lemma 14, we have $\max_k \|\mathbf{z}_k\|_2 = \sqrt{\max_k \|\mathbf{z}_k\|_2^2} \le O_p(\sqrt{n})$. Therefore, by using the fact that $\|\mathbf{y}-\boldsymbol{\mu}_0\|_2 \le O_p(\sqrt{n})$, we have

$$\begin{aligned}
&\max_k \left|\sum_{j\neq k}^{p}(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0))\right| \\
&\le \max_k \left|\sum_{j\neq k}^{p}[(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0))] - (p-1)\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)\right| + (p-1)\max_k |\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)| \\
&\le p\max_k \|\mathbf{z}_k\|_2 \|\mathbf{y}-\boldsymbol{\mu}_0\|_2 \max_k \left\{ \frac{1}{\|\mathbf{z}_k\|_2\|\mathbf{y}-\boldsymbol{\mu}_0\|_2}\left|\frac{1}{p-1}\sum_{j\neq k}^{p}[(\mathbf{z}_k^T\mathbf{z}_j)(\mathbf{z}_j^T(\mathbf{y}-\boldsymbol{\mu}_0)) - \mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)]\right|\right\} \\
&\quad + p\max_k |\mathbf{z}_k^T(\mathbf{y}-\boldsymbol{\mu}_0)| \\
&\le cpn\sqrt{\log p}\sqrt{p^{-1}\log p} + cp\sqrt{n\log p}. \qquad (38)
\end{aligned}$$

Combining (36)–(38) yields

$$\eta_2 = \|n^{-2}\mathbf{X}^T\mathbf{X}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty \le cp^{1/2}n^{-1}\log p = o(\sqrt{n^{-1}\log n}). \qquad (39)$$

Part 3: Bound of η_3.

Finally, we study η_3. We observe that $\eta_3 \le \left\|\sum_{k=2}^{\infty}(-1)^{k+1}(\mathbf{E})^k\right\|_\infty \|n^{-1}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty$. Lemma 7 proves that $\left\|\sum_{k=2}^{\infty}(-1)^{k+1}(\mathbf{E})^k\right\|_\infty \le O(p^{3/2}n^{-1})$ while equation (35) shows that $\|n^{-1}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty = O(\sqrt{n^{-1}\log p})$ with probability $1-O(p^{-c})$. Putting these facts together, we obtain

$$\eta_3 \le O(p^{3/2}n^{-1}\sqrt{n^{-1}\log p}) = o(\sqrt{n^{-1}\log n}) \qquad (40)$$

where we use $p = O(n^{\alpha_0})$ with $\alpha_0 \in [0, 2/3)$.

Combining equations (35), (39), and (40), we obtain that with probability at least $1-o(p^{-a})$,

$$\|(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)\|_\infty \le c\sqrt{n^{-1}\log n}.$$

■

Lemma 6 *Under the assumptions of Theorem 4, $\|(n^{-1}\mathbf{X}^T\mathbf{X})^{-1}\|_\infty \le 1 + O(pn^{-1/2})$ with probability $1-O(p^{-c})$.*

Proof Let $\mathbf{E} = n^{-1}\mathbf{X}^T\mathbf{X} - \mathbf{I}_p$. Then, $\|\mathbf{E}\|_2 \le C(p/n)^{1/2}$ for some constant C with probability $1-O(p^{-c})$ by Theorem 4.6.1 in Vershynin (2016). Furthermore, by matrix inversion, we get

$$(n^{-1}\mathbf{X}^T\mathbf{X})^{-1} = (\mathbf{I}_p+\mathbf{E})^{-1} = \mathbf{I}_p - \sum_{k=1}^{\infty}(-1)^{k+1}(\mathbf{E})^k.$$

Now, we take the norm and use triangle inequalities to get

$$\begin{aligned}
\|(n^{-1}\mathbf{X}^T\mathbf{X})^{-1}\|_\infty &\le \|\mathbf{I}_p\|_\infty + \sum_{k=1}^{\infty}\|\mathbf{E}^k\|_\infty \le 1 + p^{1/2}\sum_{k=1}^{\infty}\|\mathbf{E}^k\|_2 \\
&\le 1 + p^{1/2}\sum_{k=1}^{\infty}\|\mathbf{E}\|_2^k \le 1 + Cp^{1/2}\sum_{k=1}^{\infty}((p/n)^{1/2})^k \\
&\le 1 + Cp^{1/2}(p/n)^{1/2}
\end{aligned}$$

where we use the fact that p/n is bounded by a constant less than 1. ■

Lemma 7 *In the same setting as Lemma 6, if $\mathbf{E} = n^{-1}\mathbf{X}^T\mathbf{X} - \mathbf{I}_p$, then $\left\|\sum_{k=2}^{\infty}(-1)^{k+1}(\mathbf{E})^k\right\|_\infty \le Cp^{3/2}n^{-1}$, with probability $1-O(p^{-c})$.*

Proof Again, we use that $\|\mathbf{E}\|_2 \le C(p/n)^{1/2}$ for some constant C with probability $1-O(p^{-c})$. By similar calculations as in Lemma 6, we deduce

$$\begin{aligned}
\left\|\sum_{k=2}^{\infty}(-1)^{k+1}(\mathbf{E})^k\right\|_\infty &\le \sum_{k=2}^{\infty}\left\|(-1)^{k+1}(\mathbf{E})^k\right\|_\infty \le \sum_{k=2}^{\infty}p^{1/2}\left\|(\mathbf{E})^k\right\|_2 \\
&= \sum_{k=2}^{\infty}p^{1/2}\|\mathbf{E}\|_2^k \le \sum_{k=2}^{\infty}p^{1/2}(p/n)^{k/2} \le Cp^{3/2}n^{-1}.
\end{aligned}$$

■

Lemma 8 *Let $\mathbf{W}_j$ be nonnegative random variables for $1 \le j \le p$ that are not necessarily independent. If $P(W_j > t) \le C_1\exp(-C_2a_nt^2)$ for some constants C_1 and C_2 and for some sequence a_n, then for any $c > 0$, $\max_{1\le j\le p} W_j \le ((c+1)/C_2)^{1/2}a_n^{-1/2}(\log p)^{1/2}$ with probability at least $1-O(p^{-c})$.*

Proof Using union bound, we get

$$P(\max_{1\le j\le p} W_j > t) \le \sum_{1\le j\le p} P(W_j > t) \le pC_1\exp(-C_2a_nt^2).$$

Taking $t = a_n^{-1/2}(\log p)^{1/2}((c+1)/C_2)^{1/2}$ concludes the proof since then

$$P(\max_{1\le j\le p} W_j > a_n^{-1/2}(\log p)^{1/2}((c+1)/C_2)^{1/2}) \le C_1p^{-c}.$$

■

Lemma 9 *Let W_{ij} be random variables which are independent over the index i. Assume that there are constants C_1 and C_2 such that $P(|W_{ij}| > t) \le C_1 \exp(-C_2 t^\alpha)$ with $0 < \alpha \le 1$. Then, with probability $1 - O(p^{-c})$,*

$$\max_{0\le j\le p} \left| n^{-1} \sum_{i=1}^{n} (W_{ij} - EW_{ij}) \right| \le C n^{-(1/2)\alpha} (\log p)^{1/2},$$

for some positive constants c and C.

Proof We have $P(|n^{-1}\sum_{i=1}^n (W_{ij} - EW_{ij})| > t) \le C_3 \exp(-C_4 n^\alpha t^2)$ by Lemma 6 of Fan et al. (2016) where C_3 and C_4 are some positive constants which only depend on C_1 and C_2. This probability bound shows that the assumption of Lemma 8 holds with $a_n = n^\alpha$. Thus, Lemma 8 finishes the proof. ■

Lemma 10 *With probability $1 - O(p^{-c})$, the vector $\mathbf{r}$ defined in (27) satisfies the bound $\|n^{-1}\mathbf{r}\|_\infty = O(n^{\alpha - 5/6} \log n \sqrt{n^{-1}\log n})$.*

Proof We begin by observing that both x_{ij} and $(\mathbf{x}_i^T \boldsymbol{\beta}/\|\boldsymbol{\beta}\|_2)$ are standard normal variables. So, using Lemma 1 of Fan et al. (2016), we have $P(x_{ij}(\mathbf{x}_i^T\boldsymbol{\beta}/\|\boldsymbol{\beta}\|_2)^2 > t) \le C\exp(-Ct^{2/3})$ for some constant C which does not depend $\boldsymbol{\beta}$. It is easy to see that $x_{ij}(\mathbf{x}_i^T\boldsymbol{\beta})^2$ are independent random variables across i's with mean 0. By Lemma 9, $\max_{1\le j\le p} |n^{-1}\sum_{i=1}^n x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^2|$ is of order $O(n^{-1/3}(\log p)^{1/2})$. Moreover, $\|\boldsymbol{\beta}\|_2 \le p^{1/2}\|\boldsymbol{\beta}\|_\infty \le O(p^{1/2}\sqrt{n^{-1}\log n})$ when $\boldsymbol{\beta} \in \mathcal{N}_0$. Therefore,

$$\begin{aligned}
\|n^{-1}\mathbf{r}\|_\infty &= \max_{1\le j\le p} \left| \frac{b^{(3)}(0)}{2} \|\boldsymbol{\beta}\|_2^2 n^{-1} \sum_{i=1}^n x_{ij} (\mathbf{x}_i^T \boldsymbol{\beta}/\|\boldsymbol{\beta}\|_2)^2 \right| \\
&\le Cpn^{-1}(\log n) n^{-1/3} (\log p)^{1/2} = Cn^{\alpha - 4/3} (\log n)^{3/2} \\
&= O(n^{\alpha - 5/6} (\log n) \sqrt{n^{-1}\log n}),
\end{aligned}$$

since $p = O(n^\alpha)$. ■

Lemma 11 *With probability $1 - O(p^{-c})$, the vector $\mathbf{s}$ defined in (28) satisfies the bound $\|n^{-1}\mathbf{s}\|_\infty = O((n^{3/2\alpha - 5/4}(\log n)^{3/2} + (n^{\alpha-1}(\log n)^{1/2})\sqrt{n^{-1}\log n})$.*

Proof First, observe that for some constant C, $|n^{-1}s_j| \le C\|\boldsymbol{\beta}\|_2^3 n^{-1}\sum_{i=1}^n x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^3$. Moreover, the summands $x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^3$ are independent over i and they satisfy the probability bound $P(|x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^3| > t) \le C\exp(-Ct^{1/2})$ by Lemma 1 of Fan et al. (2016). Thus, by Lemma 9, we obtain

$$\max_{1\le j\le p} \left| n^{-1} \sum_{i=1}^n \left(x_{ij} \left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^3 - E\left[\left(x_{ij} \frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2} \right)^3 \right] \right) \right| = O(n^{-1/4}(\log p)^{1/2}).$$

Now, we calculate the expected value of the summand $x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^3$. We decompose $\mathbf{x}_i^T\boldsymbol{\beta}$ as $x_{ij}\beta_j + \mathbf{x}_{i,-j}^T\boldsymbol{\beta}_{-j}$ where $\mathbf{x}_{i,-j}$ and $\boldsymbol{\beta}_{-j}$ are the vectors $\mathbf{x}_i$ and $\boldsymbol{\beta}$ whose jth entry is removed. We use the independence of $\mathbf{x}_{i,-j}$ and x_{ij} and get

$$\begin{aligned}
E\left[x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^3 \right] &= \frac{1}{\|\boldsymbol{\beta}\|_2^3} E\left[x_{ij}\left(x_{ij}\beta_j + \mathbf{x}_{i,-j}^T\boldsymbol{\beta}_{-j}\right)^3 \right] \\
&= \frac{1}{\|\boldsymbol{\beta}\|_2^3} E\left[x_{ij}^4\beta_j^3 + 3x_{ij}^3\beta_j^2\left(\mathbf{x}_{i,-j}^T\boldsymbol{\beta}_{-j}\right) + 3x_{ij}^2\beta_j\left(\mathbf{x}_{i,-j}^T\boldsymbol{\beta}_{-j}\right)^2 + x_{ij}\left(\mathbf{x}_{i,-j}^T\boldsymbol{\beta}_{-j}\right)^3 \right] \\
&= \frac{1}{\|\boldsymbol{\beta}\|_2^3}\left[3\beta_j^3 + 3\beta_j\|\boldsymbol{\beta}_{-j}\|_2^2 \right] \\
&= \frac{3\beta_j}{\|\boldsymbol{\beta}\|_2}.
\end{aligned}$$

Finally, we can combine the result of Lemma 9 and the expected value of $x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^3$. We bound $\|n^{-1}\mathbf{s}\|_\infty$ as follows

$$\begin{aligned}
\|n^{-1}\mathbf{s}\|_\infty &\le C\|\boldsymbol{\beta}\|_2^3 \max_{1\le j\le p} \left| n^{-1}\sum_{i=1}^n x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^3 \right| \\
&\le O\left(\|\boldsymbol{\beta}\|_2^3 (n^{-1/4}(\log p)^{1/2} + \frac{\|\boldsymbol{\beta}\|_\infty}{\|\boldsymbol{\beta}\|_2}) \right) \\
&\le O\left(\|\boldsymbol{\beta}\|_2^3 n^{-1/4}(\log p)^{1/2} + \|\boldsymbol{\beta}\|_\infty \|\boldsymbol{\beta}\|_2^2 \right).
\end{aligned}$$

Since $\boldsymbol{\beta} \in \mathcal{N}_0$, we have $\|\boldsymbol{\beta}\|_2 = O(p^{1/2}n^{-1/2}(\log p)^{1/2})$ and $\|\boldsymbol{\beta}\|_\infty = O(n^{-1/2}(\log p)^{1/2})$. Thus, $\|n^{-1}\mathbf{s}\|_\infty = O((n^{3/2\alpha-5/4}(\log n)^{3/2} + (n^{\alpha-1}(\log n)^{1/2})\sqrt{n^{-1}\log n})$ when $p = O(n^\alpha)$. ■

Lemma 12 *With probability $1 - O(p^{-c})$, the vector $\mathbf{t}$ defined in (29) satisfies the bound $\|n^{-1}\mathbf{t}\|_\infty = O(n^{2\alpha - 3/2}(\log n)^{3/2}\sqrt{n^{-1}\log n})$.*

Proof The proof is similar to the proof of Lemma 11. Since $b^{(5)}(\cdot)$ is uniformly bounded, $|n^{-1}t_j| \le C\|\boldsymbol{\beta}\|_2^4 n^{-1}\sum_{i=1}^n \left| x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^4 \right|$ for some constant C. We focus on the summands $\left| x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^4 \right|$ which are independent across i. Moreover, repeated application of Lemma 1 of Fan et al. (2016) yields $P(x_{ij}(\mathbf{x}_i^T\boldsymbol{\beta}/\|\boldsymbol{\beta}\|_2)^2 > t) \le C\exp(-Ct^{2/5})$ for some constant C independent of $\boldsymbol{\beta}$. We can bound the expected value of the summand by Cauchy-Schwartz: $E\left[\left| x_{ij}\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^4 \right| \right] \le \left(Ex_{ij}^2 E\left[\left(\frac{\mathbf{x}_i^T\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|_2}\right)^8 \right] \right)^{1/2} = \sqrt{105}$. So, by Lemma 9, we get

$$\begin{aligned}
\|n^{-1}\mathbf{t}\|_\infty &\le C\|\boldsymbol{\beta}\|_2^4 (\sqrt{105} + n^{-1/5}(\log p)^{1/2}) \\
&= O(\|\boldsymbol{\beta}\|_2^4) = O(p^2 n^{-2}(\log n)^2).
\end{aligned}$$

Finally, we can deduce that $\|n^{-1}\mathbf{t}\|_\infty = O(n^{2\alpha-3/2}(\log n)^{3/2}\sqrt{n^{-1}\log n})$ when $p = O(n^\alpha)$. ■

Lemma 13 *Let* $T = \mathbf{e}_j^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T[\mathbf{y}-\boldsymbol{\mu}_0] - n^{-1}\mathbf{e}_j^T\mathbf{X}^T[\mathbf{y}-\boldsymbol{\mu}_0]$. *Under the assumptions of Theorem 4, we have*

$$\mathbf{e}_j^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T[\mathbf{y}-\boldsymbol{\mu}_0] - n^{-1}\mathbf{e}_j^T\mathbf{X}^T[\mathbf{y}-\boldsymbol{\mu}_0] = o_p(n^{-1/2}). \tag{41}$$

Proof Since $\mathbf{X}$ and $\mathbf{y}$ are independent, expectation of T is clearly zero. Then, we consider the variance of T. To this end, we condition on X. We can calculate the conditional variance of T as follows

$$\begin{aligned}\mathrm{var}[T|\mathbf{X}] &= \mathrm{var}[\mathbf{e}_j^T((\mathbf{X}^T\mathbf{X})^{-1} - n^{-1}I_p)\mathbf{X}^T(\mathbf{y}-\boldsymbol{\mu}_0)|\mathbf{X}] \\ &= \phi b''(0)\mathbf{e}_j^T((\mathbf{X}^T\mathbf{X})^{-1} - n^{-1}I_p)\mathbf{X}^T\mathbf{X}((\mathbf{X}^T\mathbf{X})^{-1} - n^{-1}I_p)\mathbf{e}_j\end{aligned}$$

where we use $\mathrm{var}[y] = \phi b''(0)I_n$. When we define $\mathbf{E} = n^{-1}\mathbf{X}^T\mathbf{X} - I_p$, simple calculations show that

$$\begin{aligned}\mathrm{Var}[T|\mathbf{X}] &= \phi b''(0)n^{-1}\mathbf{e}_j^T((n^{-1}\mathbf{X}^T\mathbf{X})^{-1} - I_p) + (n^{-1}\mathbf{X}^T\mathbf{X} - I_p))\mathbf{e}_j \\ &= \phi b''(0)n^{-1}\mathbf{e}_j^T\left(\sum_{k=2}^{\infty}(-1)^k\mathbf{E}^k\right)\mathbf{e}_j.\end{aligned}$$

Now, we can obtain the unconditional variance using the law of total variance.

$$\begin{aligned}\mathrm{var}[T] &= E[\mathrm{var}[T|\mathbf{X}]] + \mathrm{var}[E[T|\mathbf{X}]] \\ &= \phi b''(0)n^{-1}\mathbf{e}_j^T E\left(\sum_{k=2}^{\infty}(-1)^k\mathbf{E}^k\right)\mathbf{e}_j.\end{aligned}$$

Thus, using Lemma 7, we can show that $\mathrm{var}[T] = o(n^{-1})$. Finally, we use Chebyshev's inequality $P(|T| > n^{-1/2}) \le n\mathrm{var}[T] = o(1)$. So, we conclude that $T = o_p(n^{-1/2})$ ■

Lemma 14 *Let* x_{ij} *be standard normal random variables for* $1 \le i \le n$ *and* $1 \le j \le p$. *Then,* $\max_{1\le j\le p}\sum_{i=1}^n x_{ij}^2 \le n + O(n^{1/2}(\log p)^{1/2})$ *with probability* $1 - O(p^{-c})$ *for some positive constant* c. *Consequently, when* $\log p = O(n^\alpha)$ *for some* $0 < \alpha \le 1$, *we have* $\max_{1\le j\le p}\sum_{i=1}^n x_{ij}^2 = O(n)$, *for large enough* n *with probability* $1 - O(p^{-c})$.

Proof Since x_{ij} is a standard normal variable, x_{ij}^2 is subexponential random variable whose mean is 1. So, Lemma 9 entails that

$$\max_{1\le j\le p}\left|n^{-1}\sum_{i=1}^n(x_{ij}^2 - 1)\right| = O(n^{-1/2}(\log p)^{1/2})$$

with probability $1 - O(p^{-c})$. Thus, simple calculations yields

$$\max_{1\le j\le p}\sum_{i=1}^n x_{ij}^2 = \max_{1\le j\le p}\left|n + \sum_{i=1}^n(x_{ij}^2 - 1)\right| \le n + O(n^{1/2}(\log p)^{1/2})$$

with probability $1 - O(p^{-c})$. ■

B.3. Proof of Theorem 2

To prove the conclusion in Theorem 2, we use the proof by contradiction. Let us make an assumption (A) that the asymptotic normality (11) in Theorem 1 which has been proved to hold when $p = o(n^{1/2})$ continues to hold when $p \sim n^{\alpha_0}$ for some constant $1/2 < \alpha_0 \le 1$, where $\sim$ stands for asymptotic order. As shown in Section 3.1, in the case of logistic regression under global null (that is, $\boldsymbol{\beta}_0 = \mathbf{0}$) with deterministic rescaled orthonormal design matrix $\mathbf{X}$ (in the sense of $n^{-1}\mathbf{X}^T\mathbf{X} = I_p$) the limiting distribution in (11) by assumption (A) becomes

$$2^{-1}n^{1/2}\widehat{\beta}_j \xrightarrow{\mathscr{D}} N(0,1), \tag{42}$$

where $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_1, \cdots, \widehat{\beta}_p)^T$ is the MLE.

Let us now assume that the rescaled random design matrix $n^{-1/2}\mathbf{X}$ is uniformly distributed on the Stiefel manifold $V_p(\mathbb{R}^n)$ which can be thought of as the space of all $n \times p$ orthonormal matrices. Then it follows from (42) that

$$2^{-1}n^{1/2}\widehat{\beta}_j \xrightarrow{\mathscr{D}} N(0,1) \ \text{ conditional on } \mathbf{X}. \tag{43}$$

Based on the limiting distribution in (43), we can make two observations. First, it holds that

$$2^{-1}n^{1/2}\widehat{\beta}_j \xrightarrow{\mathscr{D}} N(0,1) \tag{44}$$

unconditional on the design matrix $\mathbf{X}$. Second, $\widehat{\beta}_j$ is asymptotically independent of the design matrix $\mathbf{X}$, and so is the MLE $\widehat{\boldsymbol{\beta}}$.

Since the distribution of $n^{-1/2}\mathbf{X}$ is assumed to be the Haar measure on the Stiefel manifold $V_p(\mathbb{R}^n)$, we have

$$n^{-1/2}\mathbf{X}\mathbf{Q} \overset{d}{=} n^{-1/2}\mathbf{X}, \tag{45}$$

where $\mathbf{Q}$ is any fixed $p \times p$ orthogonal matrix and $\overset{d}{=}$ stands for equal in distribution. Recall that the MLE $\widehat{\boldsymbol{\beta}}$ solves the score equation (4), which is in turn equivalent to equation

$$\mathbf{Q}^T\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}(\mathbf{X}\boldsymbol{\beta})] = \mathbf{0} \tag{46}$$

since $\mathbf{Q}$ is orthogonal. We now use the fact that the model is under global null which entails that the response vector $\mathbf{y}$ is independent of the design matrix $\mathbf{X}$. Combining this fact with (45)–(46) yields

$$\mathbf{Q}^T\widehat{\boldsymbol{\beta}} \overset{d}{=} \widehat{\boldsymbol{\beta}} \tag{47}$$

by noting that $\mathbf{X}\boldsymbol{\beta} = (\mathbf{X}\mathbf{Q})(\mathbf{Q}^T\boldsymbol{\beta})$. Since the distributional identity (47) holds for any fixed $p \times p$ orthogonal matrix $\mathbf{Q}$, we conclude that the MLE $\widehat{\boldsymbol{\beta}}$ has a spherical distribution on $\mathbb{R}^p$. It is a well-known fact that all the marginal characteristic functions of a spherical distribution have the same generator. Such a fact along with (44) entails that

$$2^{-1}n^{1/2}\widehat{\boldsymbol{\beta}} \text{ is asymptotically close to } N(\mathbf{0}, I_p). \tag{48}$$

To simplify the exposition, let us now make the asymptotic limit exact and assume that

$$\widehat{\boldsymbol{\beta}} \sim N(\mathbf{0}, 4n^{-1}I_p) \text{ and is independent of } \mathbf{X}. \tag{49}$$

The remaining analysis focuses on the score equation (4) which is solved exactly by the MLE $\widehat{\boldsymbol{\beta}}$, that is,

$$\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}(\mathbf{X}\widehat{\boldsymbol{\beta}})] = \mathbf{0}, \tag{50}$$

which leads to

$$\boldsymbol{\xi} \equiv n^{-1/2}\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}(\mathbf{0})] = n^{-1/2}\mathbf{X}^T[\boldsymbol{\mu}(\mathbf{X}\widehat{\boldsymbol{\beta}}) - \boldsymbol{\mu}(\mathbf{0})] \equiv \boldsymbol{\eta}. \tag{51}$$

Let us first consider the random variable $\boldsymbol{\xi}$ defined in (51). Note that $2[\mathbf{y} - \boldsymbol{\mu}(\mathbf{0})]$ has independent and identically distributed (i.i.d.) components each taking value 1 or -1 with equal probability $1/2$, and is independent of $\mathbf{X}$. Thus since $n^{-1/2}\mathbf{X}$ is uniformly distributed on the Stiefel manifold $V_p(\mathbb{R}^n)$, it is easy to see that

$$\boldsymbol{\xi} = n^{-1/2}\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}(\mathbf{0})] \stackrel{d}{=} 2^{-1}n^{-1/2}\mathbf{X}^T\mathbf{1}, \tag{52}$$

where $\mathbf{1} \in \mathbb{R}^n$ is a vector with all components being one. Using similar arguments as before, we can show that $\boldsymbol{\xi}$ has a spherical distribution on $\mathbb{R}^p$. Thus the joint distribution of $\boldsymbol{\xi}$ is determined completely by the marginal distribution of $\boldsymbol{\xi}$. For each $1 \le j \le p$, denote by ξ_j the jth component of $\boldsymbol{\xi} = 2^{-1}n^{-1/2}\mathbf{X}^T\mathbf{1}$ using the distributional representation in (52). Let $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_p)$ with each $\mathbf{x}_j \in \mathbb{R}^n$. Then we have

$$\xi_j = 2^{-1}n^{-1/2}\mathbf{x}_j^T\mathbf{1} \stackrel{d}{=} 2^{-1}(n^{1/2}/\|\widetilde{\mathbf{x}}_j\|_2)n^{-1/2}\widetilde{\mathbf{x}}_j^T\mathbf{1}, \tag{53}$$

where $\widetilde{\mathbf{x}}_j \sim N(\mathbf{0}, 4^{-1}I_n)$. It follows from (53) and the concentration phenomenon of Gaussian measures that each ξ_j is asymptotically close to $N(0, 4^{-1})$ and thus consequently $\boldsymbol{\xi}$ is asymptotically close to $N(\mathbf{0}, 4^{-1}I_p)$. *A key fact (i) for the finite-sample distribution of $\boldsymbol{\xi}$ is that the standard deviation of each component ξ_j converges to $1/2$ at rate $O_P(n^{-1/2})$ that does not depend upon the dimensionality p at all.*

We now turn our attention to the second term $\boldsymbol{\eta}$ defined in (51). In view of (49) and the fact that $n^{-1/2}\mathbf{X}$ is uniformly distributed on the Stiefel manifold $V_p(\mathbb{R}^n)$, we can show that with significant probability,

$$\|\mathbf{X}\widehat{\boldsymbol{\beta}}\|_\infty \le o(1) \tag{54}$$

for $p \sim n^{\alpha_0}$ with $\alpha_0 < 1$. The uniform bound in (54) enables us to apply the mean value theorem for the vector-valued function $\boldsymbol{\eta}$ around $\boldsymbol{\beta}_0 = \mathbf{0}$, which results in

$$\begin{aligned}\boldsymbol{\eta} &= n^{-1/2}\mathbf{X}^T[\boldsymbol{\mu}(\mathbf{X}\widehat{\boldsymbol{\beta}}) - \boldsymbol{\mu}(\mathbf{0})] = 4^{-1}n^{-1/2}\mathbf{X}^T\mathbf{X}\widehat{\boldsymbol{\beta}} + \mathbf{r} \\ &= 4^{-1}n^{1/2}\widehat{\boldsymbol{\beta}} + \mathbf{r}\end{aligned} \tag{55}$$

since $n^{-1/2}\mathbf{X}$ is assumed to be orthonormal, where

$$\mathbf{r} = n^{-1/2}\mathbf{X}^T\left\{\int_0^1 \left[\boldsymbol{\Sigma}(t\mathbf{X}\widehat{\boldsymbol{\beta}}) - 4^{-1}I_n\right] dt\right\}\mathbf{X}\widehat{\boldsymbol{\beta}}. \tag{56}$$

Here, the remainder term $\mathbf{r} = (r_1, \cdots, r_p)^T \in \mathbb{R}^p$ is stochastic and each component r_j is generally of order $O_P\{p^{1/2}n^{-1/2}\}$ in light of (49) when the true model may deviate from the global null case of $\boldsymbol{\beta}_0 = \mathbf{0}$.

Since our focus in this theorem is the logistic regression model under the global null, we can in fact claim that each component r_j is generally of order $O_P\{pn^{-1}\}$, which is a better rate of convergence than the one mentioned above thanks to the assumption of $\boldsymbol{\beta}_0 = \mathbf{0}$. To prove this claim, note that the variance function $b''(\theta)$ is symmetric in $\theta \in \mathbb{R}$ and takes the maximum value $1/4$ at $\theta = 0$. Thus in view of (54), we can show that with significant probability,

$$4^{-1}I_n - \boldsymbol{\Sigma}(t\mathbf{X}\widehat{\boldsymbol{\beta}}) \ge c\mathrm{diag}\{(t\mathbf{X}\widehat{\boldsymbol{\beta}}) \circ (t\mathbf{X}\widehat{\boldsymbol{\beta}})\} = ct^2\mathrm{diag}\{(\mathbf{X}\widehat{\boldsymbol{\beta}}) \circ (\mathbf{X}\widehat{\boldsymbol{\beta}})\} \tag{57}$$

for all $t \in [0, 1]$, where $c > 0$ is some constant and $\ge$ stands for the inequality for positive semidefinite matrices. Moreover, it follows from (49) and the fact that $n^{-1/2}\mathbf{X}$ is uniformly distributed on the Stiefel manifold $V_p(\mathbb{R}^n)$ that with significant probability, all the n components of $\mathbf{X}\widehat{\boldsymbol{\beta}}$ are concentrated in the order of $p^{1/2}n^{-1/2}$. This result along with (57) and the fact that $n^{-1}\mathbf{X}^T\mathbf{X} = I_p$ entails that with significant probability,

$$\begin{aligned}&n^{-1/2}\mathbf{X}^T\left\{\int_0^1 \left[4^{-1}I_n - \boldsymbol{\Sigma}(t\mathbf{X}\widehat{\boldsymbol{\beta}})\right] dt\right\}\mathbf{X} \\ &\ge n^{-1/2}\mathbf{X}^T\left\{\int_0^1 c_* t^2 pn^{-1} dt\right\}\mathbf{X} \\ &= 3^{-1}c_* pn^{-3/2}\mathbf{X}^T\mathbf{X} = 3^{-1}c_* pn^{-1/2}I_p,\end{aligned} \tag{58}$$

where $c_* > 0$ is some constant. Thus combining (56), (58), and (49) proves the above claim.

We make two important observations about the remainder term $\mathbf{r}$ in (55). First, $\mathbf{r}$ has a spherical distribution on $\mathbb{R}^p$. This is because by (55) and (51) it holds that

$$\mathbf{r} = \boldsymbol{\eta} - 4^{-1}n^{1/2}\widehat{\boldsymbol{\beta}} = \boldsymbol{\xi} - 4^{-1}n^{1/2}\widehat{\boldsymbol{\beta}},$$

which has a spherical distribution on $\mathbb{R}^p$. Thus the joint distribution of $\mathbf{r}$ is determined completely by the marginal distribution of $\mathbf{r}$. Second, for the nonlinear setting of logistic regression model, the appearance of the remainder term $\mathbf{r}$ in (55) is due solely to *the nonlinearity of the mean function* $\boldsymbol{\mu}(\cdot)$, and we have shown that each component r_j can indeed achieve the worst-case order pn^{-1} in probability. For each $1 \le j \le p$, denote by η_j the jth component of $\boldsymbol{\eta}$. Then in view of (49) and (55), *a key fact (ii) for the finite-sample distribution of $\boldsymbol{\eta}$ is that the standard deviation of each component η_j converges to $1/2$ at rate $O_P\{pn^{-1}\}$ that generally does depend upon the dimensionality p.*

Finally, we are ready to compare the two random variables $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ on the two sides of equation (51). Since equation (51) is a distributional identity in $\mathbb{R}^p$, naturally the square root of the sum of varξ_j's and the square root of the sum of varη_j's are expected to converge to the common value $2^{-1}p^{1/2}$ at rates that are asymptotically negligible. However, the former has rate $p^{1/2}O_P(n^{-1/2}) = O_P\{p^{1/2}n^{-1/2}\}$, whereas the latter has rate $p^{1/2}O_P\{pn^{-1}\} = O_P\{p^{3/2}n^{-1}\}$. A key consequence is that when $p \sim n^{\alpha_0}$ for some constant $2/3 \le \alpha_0 < 1$, there is a profound difference between the two asymptotic rates in that the former rate is $O_P\{n^{-(1-\alpha_0)/2}\} = o_P(1)$, while the latter rate becomes $O_P\{n^{3\alpha_0/2-1}\}$ which is now asymptotically diverging or nonvanishing. Such an intrinsic asymptotic difference is, however, prohibited by the distributional identity (51) in $\mathbb{R}^p$, which results in a contradiction. Therefore, we have now argued that assumption (A) we started with for

$2/3 \le \alpha_0 < 1$ must be false, that is, the asymptotic normality (11) which has been proved to hold when $p = o(n^{1/2})$ generally would not continue to hold when $p \sim n^{\alpha_0}$ with constant $2/3 \le \alpha_0 \le 1$. In other words, we have proved the invalidity of the conventional GLM p-values in this regime of diverging dimensionality, which concludes the proof of Theorem 2.

B.4. Proof of Theorem 3

By assumption, $\mathbf{X} \sim N(\mathbf{0}, I_n \otimes \mathbf{\Sigma})$ with covariance matrix $\mathbf{\Sigma}$ nonsingular. Let us first make a useful observation. For the general case of nonsingular covariance matrix $\mathbf{\Sigma}$, we can introduce a change of variable by letting $\widetilde{\boldsymbol{\beta}} = \mathbf{\Sigma}^{1/2}\boldsymbol{\beta}$ and correspondingly $\widetilde{\mathbf{X}} = \mathbf{X}\mathbf{\Sigma}^{-1/2}$. Clearly, $\widetilde{\mathbf{X}} \sim N(\mathbf{0}, I_n \otimes I_p)$ and the MLE for the transformed parameter vector $\widetilde{\boldsymbol{\beta}}$ is exactly $\mathbf{\Sigma}^{1/2}\widehat{\boldsymbol{\beta}}$, where $\widehat{\boldsymbol{\beta}}$ denotes the MLE under the original design matrix $\mathbf{X}$. Thus to show the breakdown point of the conventional asymptotic normality of the MLE, it suffices to focus on the specific case of $\mathbf{X} \sim N(\mathbf{0}, I_n \otimes I_p)$.

Hereafter we assume that $\mathbf{X} \sim N(\mathbf{0}, I_n \otimes I_p)$ with $p = o(n)$. The rest of the arguments are similar to those in the proof of Theorem 2 in Section B.3 except for some modifications needed for the case of Gaussian design. Specifically, for the case of logistic regression model under global null (that is, $\boldsymbol{\beta}_0 = \mathbf{0}$), the limiting distribution in (11) becomes

$$2^{-1}n^{1/2}\widehat{\beta}_j \xrightarrow{\mathscr{D}} N(0,1), \tag{59}$$

since $n^{-1}\mathbf{X}^T\mathbf{X} \to I_p$ almost surely in spectrum and thus $4^{-1}n(\mathbf{A}_n^{-1})_{jj} \to 1$ in probability as $n \to \infty$. Here, we have used a claim that both the largest and smallest eigenvalues of $n^{-1}\mathbf{X}^T\mathbf{X}$ converge to 1 almost surely as $n \to \infty$ for the case of $p = o(n)$, which can be shown by using the classical results from random matrix theory (RMT) Geman (1980); Silverstein (1985); Bai (1999).

Note that since $\mathbf{X} \sim N(0, I_n \otimes I_p)$, it holds that

$$n^{-1/2}\mathbf{X}\mathbf{Q} \overset{d}{=} n^{-1/2}\mathbf{X}, \tag{60}$$

where $\mathbf{Q}$ is any fixed $p \times p$ orthogonal matrix and $\overset{d}{=}$ stands for equal in distribution. By $\mathbf{X} \sim N(0, I_n \otimes I_p)$, it is also easy to see that

$$\boldsymbol{\xi} = n^{-1/2}\mathbf{X}^T[\mathbf{y} - \boldsymbol{\mu}(\mathbf{0})] \overset{d}{=} 2^{-1}n^{-1/2}\mathbf{X}^T\mathbf{1}, \tag{61}$$

where $\mathbf{1} \in \mathbb{R}^n$ is a vector with all components being one. In view of (49) and the assumption of $\mathbf{X} \sim N(0, I_n \otimes I_p)$, we can show that with significant probability,

$$\|\mathbf{X}\widehat{\boldsymbol{\beta}}\|_\infty \le o(1) \tag{62}$$

for $p \sim n^{\alpha_0}$ with constant $\alpha_0 < 1$. It holds further that with significant probability, all the n components of $\mathbf{X}\widehat{\boldsymbol{\beta}}$ are concentrated in the order of $p^{1/2}n^{-1/2}$. This result along with (57) and the fact that $n^{-1}\mathbf{X}^T\mathbf{X} \to I_p$ almost surely in spectrum entails that with asymptotic

probability one,

$$\begin{aligned} n^{-1/2}\mathbf{X}^T\left\{\int_0^1 \left[4^{-1}I_n - \mathbf{\Sigma}(t\mathbf{X}\widehat{\boldsymbol{\beta}})\right] dt\right\}\mathbf{X} \\ \ge n^{-1/2}\mathbf{X}^T\left\{\int_0^1 c_* t^2 p n^{-1} dt\right\}\mathbf{X} \\ = 3^{-1}c_* p n^{-3/2}\mathbf{X}^T\mathbf{X} \to 3^{-1}c_* p n^{-1/2} I_p, \end{aligned} \tag{63}$$

where $c_* > 0$ is some constant. This completes the proof of Theorem 3.

References

T. W. Anderson and D. A. Darling. Asymptotic theory of certain "goodness-of-fit" criteria based on stochastic processes. *Annals of Mathematical Statistics*, 23:193–212, 1952.

T. W. Anderson and D. A. Darling. A test of goodness-of-fit. *Journal of the American Statistical Association*, 49:765–769, 1954.

Susan Athey, Guido W. Imbens, and Stefan Wager. Efficient inference of average treatment effects in high dimensions via approximate residual balancing. *arXiv preprint arXiv:1604.07125*, 2016.

Z. D. Bai. Methodologies in spectral analysis of large dimensional random matrices, a review. *Statist. Sin.*, 9:611–677, 1999.

Rina Foygel Barber and Emmanuel J. Candès. Controlling the false discovery rate via knockoffs. *Ann. Statist.*, 43:2055–2085, 2015.

Derek Bean, Peter J. Bickel, Noureddine E. Karoui, and Bin Yu. Optimal M-estimation in high-dimensional regression. *Proceedings of the National Academy of Sciences of the United States of America*, 110:14563–14568, 2013.

E. J. Candès. Private communication. 2016.

Emmanuel Candès, Yingying Fan, Lucas Janson, and Jinchi Lv. Panning for gold: 'model-X' knockoffs for high dimensional controlled variable selection. *Journal of the Royal Statistical Society Series B*, 80:551–577, 2018.

J. Fan and H. Peng. Nonconcave penalized likelihood with diverging number of parameters. *Ann. Statist.*, 32:928–961, 2004.

Jianqing Fan and Jinchi Lv. Nonconcave penalized likelihood with NP-dimensionality. *IEEE Transactions on Information Theory*, 57:5467–5484, 2011.

Y. Fan and J. Lv. Asymptotic equivalence of regularization methods in thresholded parameter space. *Journal of the American Statistical Association*, 108:1044–1061, 2013.

Yingying Fan, Yinfei Kong, Daoji Li, and Jinchi Lv. Interaction pursuit with feature screening and selection. *arXiv preprint arXiv:1605.08933*, 2016.

Yingying Fan, Emre Demirkaya, Gaorong Li, and Jinchi Lv. RANK: large-scale inference with graphical nonlinear knockoffs. *Journal of the American Statistical Association, to appear*, 2018.

S. Geman. A limit theorem for the norm of random matrices. *Ann. Probab.*, 8:252–261, 1980.

Bin Guo and Song Xi Chen. Tests for high dimensional generalized linear models. *J. R. Statist. Soc. B*, 78:1079–1102, 2016.

P. J. Huber. Robust regression: Asymptotics, conjectures and Monte Carlo. *The Annals of Statistics*, 1:799–821, 1973.

Noureddine E. Karoui, Derek Bean, Peter J. Bickel, Chinghway Lim, and Bin Yu. On robust regression with high-dimensional predictors. *Proceedings of the National Academy of Sciences of the United States of America*, 110:14557–14562, 2013.

A. Kolmogorov. Sulla determinazione empirica di una legge di distribuzione. *G. Ist. Ital. Attuari.*, 4:83–91, 1933.

J. Lv and J. S. Liu. Model selection principles in misspecified models. *Journal of the Royal Statistical Society Series B*, 76:141–167, 2014.

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

S. Portnoy. Asymptotic behavior of M-estimators of p regression parameters when p^2/n is large. i. consistency. *The Annals of Statistics*, 12:1298–1309, 1984.

S. Portnoy. Asymptotic behavior of M-estimators of p regression parameters when p^2/n is large; ii. normal approximation. *The Annals of Statistics*, 13:1403–1417, 1985.

S. Portnoy. Asymptotic behavior of likelihood methods for exponential families when the number of parameters tends to infinity. *The Annals of Statistics*, 16:356–366, 1988.

J. W. Silverstein. The smallest eigenvalue of a large dimensional wishart matrix. *Ann. Probab.*, 13:1364–1368, 1985.

N. Smirnov. Table for estimating the goodness of fit of empirical distributions. *Annals of Mathematical Statistics*, 19:279–281, 1948.

Pragya Sur and Emmanuel J Candès. A modern maximum-likelihood theory for high-dimensional logistic regression. *arXiv preprint arXiv:1803.06964*, 2018.

Sara van de Geer, Peter Bühlmann, Ya'acov Ritov, and Ruben Dezeure. On asymptotically optimal confidence regions and tests for high-dimensional models. *Ann. Statist.*, 42:1166–1202, 2014.

Roman Vershynin. High-dimensional probability. *An Introduction with Applications*, 2016.

Michael N Vrahatis. A short proof and a generalization of Miranda's existence theorem. *Proceedings of the American Mathematical Society*, 107:701–703, 1989.

Journal of Machine Learning Research 20 (2019) 1–39 Submitted 5/18; Revised 11/18; Published 3/19

Prediction Risk for the Horseshoe Regression

Anindya Bhadra BHADRA@PURDUE.EDU
Department of Statistics
Purdue University
West Lafayette, IN 47907, USA

Jyotishka Datta JD033@UARK.EDU
Department of Mathematical Sciences
University of Arkansas
Fayetteville, AR 72701, USA

Yunfan Li LI896@PURDUE.EDU
Department of Statistics
Purdue University
West Lafayette, IN 47907, USA

Nicholas G. Polson NGP@CHICAGOBOOTH.EDU
Booth School of Business
University of Chicago
Chicago, IL 60637, USA

Brandon Willard BRANDONWILLARD@GMAIL.COM
Booth School of Business
University of Chicago
Chicago, IL 60637, USA

Editor: Robert McCulloch

Abstract

We show that prediction performance for global-local shrinkage regression can overcome two major difficulties of global shrinkage regression: (i) the amount of relative shrinkage is monotone in the singular values of the design matrix and (ii) the shrinkage is determined by a single tuning parameter. Specifically, we show that the horseshoe regression, with heavy-tailed component-specific local shrinkage parameters, in conjunction with a global parameter providing shrinkage towards zero, alleviates both these difficulties and consequently, results in an improved risk for prediction. Numerical demonstrations of improved prediction over competing approaches in simulations and in a pharmacogenomics data set confirm our theoretical findings.

Keywords: Global-local Priors, Principal Components, Shrinkage Regression, Stein's Unbiased Risk Rstimate

©2019 Anindya Bhadra, Jyotishka Datta, Yunfan Li, Nicholas G. Polson, and Brandon Willard.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-321.html.

1. Introduction

We develop theoretical results on prediction risk in the high-dimensional linear regression model

$$y = X\beta + \epsilon, \tag{1}$$

where $y \in \mathbb{R}^n, X \in \mathbb{R}^{n\times p}, \beta \in \mathbb{R}^p$, $\epsilon \sim \mathcal{N}(0, \sigma^2\mathbf{I}_n)$ with $p > n$, and the design matrix X is assumed fixed. Let $\hat{\beta}$ denote the estimate of β based on the observed data y and design X. Let y^* denote a future observation generated from the same model, independent of y. Define the quadratic predictive risk

$$R = \mathbb{E}_{y^*,y|X,\beta}(y^* - X\hat{\beta})^2, \tag{2}$$

where the subscript denotes that the expectation is with respect to the data generating distribution, with X and β held fixed. We focus on comparing estimators $\hat{\beta}$ according to the criterion of Equation (2) in a *non-asymptotic* fixed n, fixed $p > n$ setting. Our approach follows the paradigm of Stein (1956), with risk results that are valid for all p and n, rather than asymptotic oracle properties. Our specific contribution, established by Theorem 5.1 and Corollary 5.1, is to identify the shortcomings of some commonly used global shrinkage estimators in prediction, with shrinkage driven by a single tuning parameter, and to demonstrate that under certain conditions, suitably-chosen component-specific local shrinkage parameters can result in theoretically lower predictive risk.

1.1. Connections with existing global shrinkage regression approaches

We define shrinkage estimators with a single tuning parameter as "global." Examples include ridge regression (Hoerl and Kennard, 1970) and principal components regression or PCR (Jolliffe, 1982), and they remain popular in prediction under the high-dimensional model of Equation (1). Shrinkage methods enjoy a number of advantages over simultaneous shrinkage and selection-based methods such as the lasso (Tibshirani, 1996) and comfortably outperform them in predictive performance in certain situations. Prominent among these is when the predictors are correlated and the resulting lasso estimate is unstable, whereas ridge or PCR estimates are not (see, e.g, the discussion in Chapter 3 of Hastie et al., 2009). On the theoretical side, Polson and Scott (2012a) used a representation devised by Frank and Friedman (1993) to show that many commonly used high-dimensional shrinkage regression estimates, such as the estimates of ridge regression, regression with g-prior (Zellner, 1986) and PCR, can be viewed as posterior means under a unified framework of "global" shrinkage prior on the regression coefficients that are suitably orthogonalized. Polson and Scott (2012a) also demonstrated that purely global shrinkage regression methods suffer from two major difficulties: (i) the amount of relative shrinkage is monotone in the singular values of the design matrix and (ii) the shrinkage is determined by a single tuning parameter. Both of these factors can translate to poor out of sample prediction performance, which they demonstrated numerically.

Polson and Scott (2012a) further provided numerical evidence that the difficulties mentioned above can be resolved by allowing "local," component-specific shrinkage terms, in conjunction with a global shrinkage parameter as used in ridge or PCR, giving rise to the

so-called "global-local" shrinkage regression models. Specifically, Polson and Scott (2012a) demonstrated by simulations that using the horseshoe prior of Carvalho et al. (2010) on the regression coefficients performed well over a variety of competitors in terms of predictive performance, including the lasso, ridge, PCR and sparse partial least squares (Chun and Keles, 2010). While these empirical results are encouraging, a theoretical investigation of the conditions required for the horseshoe regression to outperform a global shrinkage regression method such as ridge or PCR in terms of prediction has been lacking. Our work bridges this theoretical gap by developing formal tools for comparing the finite sample predictive risk for shrinkage methods.

1.2. Regression with non-convex penalties

The ℓ_1 or ℓ_2 penalties that correspond to lasso or ridge regression are convex. While this simplifies the computation, it also results in a number of drawbacks such as bias in estimating large signals (Fan and Li, 2001). This problem can be remedied using non-convex ℓ_q penalties for $0 < q < 1$, but this introduces other problems such as non-uniqueness of solutions and greater computational burden. Prominent examples of non-convex penalties include the smoothly clipped absolute deviation or SCAD (Fan and Li, 2001) and the minimax concave penalty of MCP (Zhang, 2010). In particular, the MCP estimate enjoys a number of asymptotic optimality properties and conditions required for an iterative computational algorithm to reach the global optimum are available (Mazumder et al., 2011). The optimality results, however, are valid only in an asymptotic regime with various assumptions on the design X and do not characterize finite sample risk properties. Nevertheless, the univariate MCP estimator is identical to the firm shrinkage estimator of Gao and Bruce (1997), who provide explicit finite sample expressions for predictive risk, a fact utilized later in Section 6.

1.3. Finite sample estimates of predictive risk

The quadratic risk in Equation (2) involves the future observation y^* and must be estimated. Developing a formal estimate based on the training data (X, y) to compare predictive performance of competing regression methods is important in both frequentist and Bayesian settings. This is because the frequentist tuning parameter or the Bayesian hyper-parameters can then be chosen to minimize the estimated predictive risk, if prediction of future observations is the main modeling goal. A *finite sample unbiased estimate* of R in Equation (2) is given by Stein's unbiased risk estimate or SURE (Stein, 1981).

We will focus on SURE as our estimate of R for the remainder of this article, which is an example of a model-based covariance penalty. Other examples of covariance penalties include Mallows' C_p (Mallows, 1973), Akaike's information criterion (Akaike, 1974) and risk inflation criterion (Foster and George, 1994). Nonparametric penalties include the generalized cross validation of Craven and Wahba (1978), which has the advantage of being model free but usually produces a prediction error estimate with high variance (Efron, 1983). The relationship between the covariance penalties and nonparametric approaches were further explored by Efron (2004), who showed the covariance penalties to be a Rao-Blackwellized version of the nonparametric penalties. Thus, Efron (2004) concluded that model-based penalties such as SURE or Mallows' C_p (the two coincide for models where the fit is linear

in the response variable) offer substantially lower variance in estimating the prediction risk, assuming of course the model is true. From a computational perspective, calculating SURE, when it is explicitly available, is substantially less burdensome than performing cross validation, which usually requires several Monte Carlo replications. Furthermore, SURE, an estimate of quadratic risk in prediction, also has connections with the Kullback–Leiber risk for the predictive density (George et al., 2006).

1.4. Outline of main contributions

Our main contribution is to analyze the finite sample predictive risk of global shrinkage regression methods, examine where these methods fall short, and demonstrate a remedy using local shrinkage parameters. The main results are summarized as follows.

1.4.1. Theoretical findings

The key technique to our innovation is an orthogonalized representation first employed by Frank and Friedman (1993) that allows shrinkage regression estimates to be viewed as posterior means under some suitable priors. This is formulated in Section 2. Using this representation in Sections 3 and 4, we devise general, explicit and numerically stable techniques for computing SURE for regression models that can be employed to compare the performances of global as well as horseshoe regressions. We characterize the finite sample risk properties of all competing methods by computing expectations of SURE. Consequently, all results provided in our article are valid under minimal assumptions on the design matrix, similar to the risk results by Stein (1956), where the only requirement is $n > 2$. This is at a contrast with most existing results in linear regression focusing on asymptotic minimax risk that require various assumptions on the singular values of X (e.g., Raskutti et al., 2011; Castillo et al., 2015; Dobriban and Wager, 2017).

Using the developed tools for SURE, we provide explicit finite sample risk comparisons between the global ridge and global-local horseshoe regressions in Section 5, where for all methods the tuning parameter is chosen to optimize SURE. Specifically, we demonstrate that the horseshoe regression can outperform the optimal ridge regression in prediction when most true signals are zero, but a few are large. We also compare risk of the horseshoe regression with non-convex penalized likelihood approaches such as MCP in Section 6 and show that when most of the true signals are away from zero, the risk of MCP can be quite large, unlike that of the horseshoe regression.

1.4.2. Empirical findings

Extensive numerical results are provided in Section 7 and Supplementary Section S.1. Our simulation results treat three distinct regimes: (i) sparse-robust: where most true signals are zero and a few are large, (ii) null: where all signals are zero and (iii) dense: where all signals are large. Our major finding is that the horseshoe regression outperforms the other methods in (i). Moreover, it is not much worse than ridge in (iii) and adaptive lasso in (ii), which are usually the best performers in these settings. Being a shrinkage estimate, the results for the horseshoe are numerically stable, unlike that of the selection-based estimators in the dense case. We conclude with a demonstration on real data in Section 8 and by outlining some possible extensions of the current work in Section 9.

2. Shrinkage regression estimates as posterior means

Let $X = UDW^T$ be the singular value decomposition of the design matrix. Let $D = \text{diag}(d_1, \ldots, d_n)$ with $d_1 \geq \ldots \geq d_n > 0$ and $\text{Rank}(D) = \min(n, p) = n$. Define $Z = UD$ and $\alpha = W^T\beta$. Then the regression model of Equation (1) can be reformulated as:

$$y = Z\alpha + \epsilon. \tag{3}$$

The ordinary least squared (OLS) estimate of α is $\hat{\alpha} = (Z^TZ)^{-1}Z^Ty = D^{-1}U^Ty$. Following the original results by Frank and Friedman (1993), several authors have used the well-known orthogonalization technique (Polson and Scott, 2012a; Clyde et al., 1996; Denison and George, 2012) to demonstrate that the estimates of many shrinkage regression methods can be expressed in terms of the posterior mean of the "orthogonalized" regression coefficients α under the following hierarchical model:

$$(\hat{\alpha}_i \mid \alpha_i, \sigma^2) \overset{ind}{\sim} \mathcal{N}(\alpha_i, \sigma^2 d_i^{-2}), \tag{4}$$

$$(\alpha_i \mid \sigma^2, \tau^2, \lambda_i^2) \overset{ind}{\sim} \mathcal{N}(0, \sigma^2\tau^2\lambda_i^2), \tag{5}$$

with $\sigma^2, \tau^2 > 0$. The global term τ controls the amount of shrinkage and the fixed λ_i^2 terms depend on the method at hand. Given λ_i and τ, the estimate for β under the global shrinkage prior, denoted by $\tilde{\beta}$, can be expressed in terms of the posterior mean estimate for α as follows:

$$\tilde{\alpha}_i = \frac{\tau^2\lambda_i^2 d_i^2}{1 + \tau^2\lambda_i^2 d_i^2}\hat{\alpha}_i, \quad \text{and} \quad \tilde{\beta} = \sum_{i=1}^{n} \tilde{\alpha}_i w_i, \tag{6}$$

where $\tilde{\alpha}_i = E(\alpha_i \mid \tau, \lambda_i^2, X, y)$; w_i is a $p \times 1$ vector and is the ith column of the $p \times n$ matrix W and the term $\tau^2\lambda_i^2 d_i^2/(1+\tau^2\lambda_i^2 d_i^2) \in (0, 1)$ is the shrinkage factor. The expression from Equation (6) makes it clear that it is the orthogonalized OLS estimates $\hat{\alpha}_i$s that are shrunk. We shall show that this orthogonalized representation is also particularly suitable for calculating the prediction risk estimate. The reason is tied to the independence assumption that is now feasible in Equations (4) and (5). To give a few concrete examples, we note below that several popular shrinkage regression models fall under the framework of Equations (4–5):

1. For ridge regression, $\lambda_i^2 = 1, \forall i$, and we have $\tilde{\alpha}_i = \tau^2 d_i^2 \hat{\alpha}_i/(1 + \tau^2 d_i^2)$.
2. For K component PCR, λ_i^2 is infinite for the first K components and then 0. Thus, $\tilde{\alpha}_i = \hat{\alpha}_i$ for $i = 1, \ldots, K$ and $\tilde{\alpha}_i = 0$ for $i = K+1, \ldots, n$.
3. For regression with g-prior, $\lambda_i^2 = d_i^{-2}$ and we have $\tilde{\alpha}_i = \tau^2\hat{\alpha}_i/(1+\tau^2)$ for $i = 1, \ldots, n$.

This shows the amount of relative shrinkage $\tilde{\alpha}_i/\hat{\alpha}_i$ is constant in d_i for PCR and g-prior and is monotone in d_i for ridge regression. In none of these cases it depends on the OLS estimate $\hat{\alpha}_i$ (consequently, on y). In the next section we quantify the effect of this behavior on the prediction risk estimate.

3. Stein's unbiased risk estimate for global shrinkage regression

Define the fit $\tilde{y} = X\tilde{\beta} = Z\tilde{\alpha}$, where $\tilde{\alpha}$ is the posterior mean of α. As noted by Stein (1981), the fitted risk is an underestimation of the prediction risk, and SURE for prediction is defined as:

$$SURE = ||y - \tilde{y}||^2 + 2\sigma^2 \sum_{i=1}^{n} \frac{\partial \tilde{y}_i}{\partial y_i},$$

where the $\sum_{i=1}^{n}(\partial\tilde{y}_i/\partial y_i)$ term is also known as the "degrees of freedom" (Efron, 2004). By Tweedie's formula (Masreliez, 1975; Pericchi and Smith, 1992) that relates the posterior mean with the marginals; we have for a Gaussian model of Equations (4–5) that: $\tilde{\alpha} = \hat{\alpha} + \sigma^2 D^{-2}\nabla_{\hat{\alpha}} \log m(\hat{\alpha})$, where $m(\hat{\alpha})$ is the marginal for $\hat{\alpha}$. Noting $y = Z\hat{\alpha}$ yields $\tilde{y} = y + \sigma^2 UD^{-1}\nabla_{\hat{\alpha}} \log m(\hat{\alpha})$. Using the independence of α_is, the formula for SURE becomes

$$SURE = \sigma^4 \sum_{i=1}^{n} d_i^{-2} \left\{ \frac{\partial}{\partial \hat{\alpha}_i} \log m(\hat{\alpha}_i) \right\}^2 + 2\sigma^2 \sum_{i=1}^{n} \left\{ 1 + \sigma^2 d_i^{-2} \frac{\partial^2}{\partial \hat{\alpha}_i^2} \log m(\hat{\alpha}_i) \right\}. \tag{7}$$

Thus, the prediction risk estimate for shrinkage regression can be quantified in terms of the first two derivatives of the log marginal for $\hat{\alpha}$. Integrating out α_i from Equations (4–5) yields in all these cases,

$$(\hat{\alpha}_i \mid \sigma^2, \tau^2, \lambda_i^2) \overset{ind}{\sim} \mathcal{N}(0, \sigma^2(d_i^{-2} + \tau^2\lambda_i^2)).$$

The marginal of $\hat{\alpha}$ is given by

$$m(\hat{\alpha}) \propto \prod_{i=1}^{n} \exp\left\{ -\frac{\hat{\alpha}_i^2/2}{\sigma^2(d_i^{-2} + \tau^2\lambda_i^2)} \right\},$$

which yields

$$\frac{\partial \log m(\hat{\alpha}_i)}{\partial \hat{\alpha}_i} = \frac{-\hat{\alpha}_i}{\sigma^2(d_i^{-2} + \tau^2\lambda_i^2)}; \qquad \frac{\partial^2 \log m(\hat{\alpha}_i)}{\partial \hat{\alpha}_i^2} = \frac{-1}{\sigma^2(d_i^{-2} + \tau^2\lambda_i^2)}. \tag{8}$$

Therefore, Equation (7) reduces to the following expression for SURE for global shrinkage regressions: $SURE = \sum_{i=1}^{n} SURE_i$, where,

$$SURE_i = \frac{\hat{\alpha}_i^2 d_i^2}{(1 + \tau^2\lambda_i^2 d_i^2)^2} + 2\sigma^2 \frac{\tau^2\lambda_i^2 d_i^2}{(1 + \tau^2\lambda_i^2 d_i^2)}. \tag{9}$$

From a computational perspective, the expression in Equation (9) is attractive, as it avoids costly matrix inversions. For a given σ one can choose τ to minimize the prediction risk, which amounts to a one-dimensional optimization. Note that in our notation, $d_1 \geq d_2 \ldots \geq d_n > 0$. Clearly, this is the SURE when λ_is are fixed and finite (e.g., ridge regression). For K component PCR, only the first K terms appear in the sum. The d_i terms are features of the design matrix X and one may try to control the prediction risk by varying τ. When $\tau \to \infty$, $SURE \to 2n\sigma^2$, the risk of prediction with ordinary least squares (unbiased). When $\tau \to 0$, we get the mean-only model (zero variance): $SURE \to \sum_{i=1}^{n} \hat{\alpha}_i^2 d_i^2$. Regression models with $\tau \in (0, \infty)$ represent a bias-variance tradeoff. Following are the two major difficulties of global shrinkage regression:

1. The first term of Equation (9) shows that $SURE$ is increased by those components for which $\hat{\alpha}_i^2 d_i^2$ is large. Choosing a large τ alleviates this problem, but at the expense of an $SURE_i$ of $2\sigma^2$ even for components for which $\hat{\alpha}_i^2 d_i^2$ is small (due to the second term in Equation (9)). Thus, it might be beneficial to *differentially minimize* the effect of the components for which $\hat{\alpha}_i^2 d_i^2$ is large, while ensuring those for which $\hat{\alpha}_i^2 d_i^2$ is small make a contribution less than $2\sigma^2$ to $SURE$. Yet, regression models with λ_i fixed, such as ridge, PCR, regression with g-priors, provide no mechanism for achieving this, since the relative shrinkage, defined as the ratio $\tilde{\alpha}_i/\hat{\alpha}_i$, equals $\tau^2\lambda_i^2 d_i^2/(1+\tau^2\lambda_i^2 d_i^2)$, and is solely driven by a single quantity τ.

2. Equation (6) shows that the relative shrinkage for $\hat{\alpha}_i$ is monotone in d_i; that is, those $\hat{\alpha}_i$ corresponding to a smaller d_i are necessarily shrunk more (in a relative amount). This is only sensible in the case where one has reasons to believe the low variance eigen-directions (i.e., principal components) of the design matrix are not important predictors of the response variables, an assumption that can be violated in real data (Polson and Scott, 2012a).

In the light of these two problems, we proceed to demonstrate that putting a heavy-tailed prior on λ_is, in combination with a suitably small value of τ to enable global-local shrinkage can resolve both these issues. The intuition behind this is that a small value of a *global* parameter τ enables shrinkage towards zero for all the components while the heavy tails of the *local* or component-specific λ_i terms ensure the components with large values of $\hat{\alpha}_i d_i$ are not shrunk too much, and allow the λ_i terms to be learned from the data. Simultaneously ensuring both of these factors helps in controlling the prediction risk for both the noise as well as the signal terms.

4. Stein's unbiased risk estimate for the horseshoe regression

The global-local horseshoe shrinkage regression of Polson and Scott (2012a) extends the global shrinkage regression models of the previous section by putting a local (component-specific), heavy-tailed half-Cauchy prior on the λ_i terms that allow these terms to be learned from the data, in addition to a global τ. The model equations become:

$$(\hat{\alpha}_i \mid \alpha_i, \sigma^2) \overset{ind}{\sim} \mathcal{N}(\alpha_i, \sigma^2 d_i^{-2}), \tag{10}$$

$$(\alpha_i \mid \sigma^2, \tau^2, \lambda_i^2) \overset{ind}{\sim} \mathcal{N}(0, \sigma^2\tau^2\lambda_i^2), \tag{11}$$

$$\lambda_i \overset{ind}{\sim} C^+(0,1), \tag{12}$$

with $\sigma^2, \tau^2 > 0$ and $C^+(0,1)$ denotes a standard half-Cauchy random variable with density $p(\lambda_i) = (2/\pi)(1+\lambda_i^2)^{-1}$. The posterior mean $\tilde{\alpha}$ and the regression estimate $\tilde{\beta}$ are then obtained analogously to Equation (6), with the only difference being one uses the posterior mean $\mathbb{E}(\lambda_i \mid \hat{\alpha}_i, \tau)$ instead of a fixed λ_i. The marginal prior on α_is that is obtained as a normal scale mixture by integrating out λ_is from Equations (11) and (12) is called the horseshoe prior (Carvalho et al., 2010). Improved mean square error over competing approaches in regression has been empirically observed by Polson and Scott (2012a) with horseshoe prior on α_is. The intuitive explanation for this improved performance is that a heavy tailed prior of λ_i leaves the large α_i terms of Equation (11) un-shrunk in the posterior, whereas

the global τ term provides shrinkage towards zero for all components (see, for example, the discussion by Polson and Scott, 2012b; Bhadra et al., 2017; Carvalho et al., 2010, and the references therein). However, no explicit formulation of the prediction risk under horseshoe shrinkage is available so far and we demonstrate below the heavy-tailed priors on λ_i terms, in addition to a global τ, can be beneficial in controlling the overall prediction risk.

Under the model of Equations (10–12), after integrating out α_i from the first two equations, we have,

$$(\hat{\alpha}_i \mid \sigma^2, \tau^2, \lambda_i^2) \overset{ind}{\sim} \mathcal{N}(0, \sigma^2(d_i^{-2} + \tau^2\lambda_i^2)).$$

We have, $p(\lambda_i) \propto 1/(1+\lambda_i^2)$. Thus, the marginal of $\hat{\alpha}$, denoted by $m(\hat{\alpha})$, is given up to a constant of proportionality by

$$\begin{aligned} m(\hat{\alpha}) =& \prod_{i=1}^n \int_0^\infty \mathcal{N}(\hat{\alpha}_i \mid 0, \sigma^2(d_i^{-2} + \tau^2\lambda_i^2)) p(\lambda_i) d\lambda_i \\ \propto& (2\pi\sigma^2)^{-n/2} \prod_{i=1}^n \int_0^\infty \exp\left\{ -\frac{\hat{\alpha}_i^2 d_i^2/2}{\sigma^2(1+\tau^2 d_i^2\lambda_i^2)} \right\} \frac{d_i}{(1+\tau^2 d_i^2\lambda_i^2)^{1/2}} \frac{1}{1+\lambda_i^2} d\lambda_i. \end{aligned} \tag{13}$$

This integral involves the normalizing constant of a compound confluent hypergeometric distribution that can be computed using a result of Gordy (1998).

Proposition 4.1 *(Gordy, 1998). The compound confluent hypergeometric (CCH) density is given by*

$$\mathrm{CCH}(x; p, q, r, s, \nu, \theta) = \frac{x^{p-1}(1-\nu x)^{q-1}\{\theta + (1-\theta)\nu x\}^{-r} \exp(-sx)}{B(p,q) H(p,q,r,s,\nu,\theta)},$$

for $0 < x < 1/\nu$, where the parameters satisfy $p > 0$, $q > 0$, $r \in \mathbb{R}$, $s \in \mathbb{R}$, $0 \le \nu \le 1$ and $\theta > 0$. Here $B(p,q)$ is the beta function and the function $H(\cdot)$ is given by

$$H(p,q,r,s,\nu,\theta) = \nu^{-p} \exp(-s/\nu) \Phi_1(q, r, p+q, s/\nu, 1-\theta),$$

where Φ_1 is the confluent hypergeometric function of two variables, given by

$$\Phi_1(\alpha, \beta, \gamma, x_1, x_2) = \sum_{m=0}^\infty \sum_{n=0}^\infty \frac{(\alpha)_m (\beta)_n}{(\gamma)_{m+n} m! n!} x_1^m x_2^n, \tag{14}$$

where $(a)_k$ denotes the rising factorial with $(a)_0 = 1$, $(a)_1 = a$ and $(a)_k = (a+k-1)(a)_{k-1}$.

We present our first result in the following theorem and show that the marginal $m(\hat{\alpha})$ and all its derivatives lend themselves to a series representation in terms of the first and second moments of a random variable that follows a CCH distribution. Consequently, we quantify SURE for the horseshoe regression.

Theorem 4.1 *Denote $m'(\hat{\alpha}_i) = (\partial/\partial\hat{\alpha}_i) m(\hat{\alpha}_i)$ and $m''(\hat{\alpha}_i) = (\partial^2/\partial\hat{\alpha}_i^2) m(\hat{\alpha}_i)$. Then, the following holds.*

A. *SURE for the horseshoe shrinkage regression model defined by Equations (10–12) is given by* $SURE = \sum_{i=1}^{n} SURE_i$, *where the component-wise contribution* $SURE_i$ *is given by*

$$SURE_i \;=\; 2\sigma^2 - \sigma^4 d_i^{-2}\left\{\frac{m'(\hat{\alpha}_i)}{m(\hat{\alpha}_i)}\right\}^2 + 2\sigma^4 d_i^{-2}\frac{m''(\hat{\alpha}_i)}{m(\hat{\alpha}_i)}. \tag{15}$$

B. *Under independent standard half-Cauchy prior on* λ_i*s, for the second and third terms in Equation (15) we have:*

$$\frac{m'(\hat{\alpha}_i)}{m(\hat{\alpha}_i)} = -\frac{\hat{\alpha}_i d_i^2}{\sigma^2}\mathbb{E}(Z_i), \quad \textit{and} \quad \frac{m''(\hat{\alpha}_i)}{m(\hat{\alpha}_i)} = -\frac{d_i^2}{\sigma^2}\mathbb{E}(Z_i) + \frac{\hat{\alpha}_i^2 d_i^4}{\sigma^4}\mathbb{E}(Z_i^2),$$

where $(Z_i \mid \hat{\alpha}_i, \sigma, \tau)$ *follows a* $\mathrm{CCH}(p=1, q=1/2, r=1, s=\hat{\alpha}_i^2 d_i^2/2\sigma^2, v=1, \theta = 1/\tau^2 d_i^2)$ *distribution.*

A proof is given in Appendix A.1. Theorem 4.1 provides a computationally tractable mechanism for calculating SURE for the horseshoe shrinkage regression in terms of the moments of CCH random variables. Gordy (1998) provides a simple formula for all integer moments of CCH random variables. Specifically, he shows if $X \sim \mathrm{CCH}(x; p, q, r, s, \nu, \theta)$ then

$$\mathbb{E}(X^k) = \frac{(p)_k}{(p+q)_k}\frac{H(p+k, q, r, s, \nu, \theta)}{H(p, q, r, s, \nu, \theta)}, \tag{16}$$

for integers $k \geq 1$. Moreover, as demonstrated by Gordy (1998), these moments can be numerically evaluated quite easily over a range of parameter values and calculations remain very stable. A consequence of this explicit formula for SURE is that the global shrinkage parameter τ can now be chosen to minimize SURE by performing a one-dimensional numerical optimization. Another consequence is that an application of Theorem 3 of Carvalho et al. (2010) shows

$$\lim_{|\hat{\alpha}_i|\to\infty}\frac{m'(\hat{\alpha}_i)}{m(\hat{\alpha}_i)} = \lim_{|\hat{\alpha}_i|\to\infty}\frac{\partial \log m(\hat{\alpha}_i)}{\partial \hat{\alpha}_i} = 0,$$

with high probability, where $m(\hat{\alpha}_i)$ is the marginal under the horseshoe prior. Recall that the posterior mean $\tilde{\alpha}_i$ and the OLS estimate $\hat{\alpha}_i$ are related by Tweedie's formula as $\tilde{\alpha}_i = \hat{\alpha}_i + \sigma^2 d_i^{-2}\partial \log m(\hat{\alpha}_i)/\partial\hat{\alpha}_i$. Thus, $\tilde{\alpha}_i \approx \hat{\alpha}_i$, with high probability, as $|\hat{\alpha}_i| \to \infty$, for any fixed d_i and σ for the horseshoe regression. Since $\hat{\alpha}_i$ is unbiased for α_i, the resultant horseshoe posterior mean is also seen to be unbiased when $|\hat{\alpha}_i|$ is large. Compare with the resultant $\tilde{\alpha}_i$ for global shrinkage regression of Equation (6), which is monotone decreasing in d_i, and therefore can be highly biased if a true large $|\alpha_i|$ corresponds to a small d_i. Perhaps more importantly, we can use the expression from Theorem 4.1 to estimate the prediction risk of the horseshoe regression for the signal and the noise terms. First we treat the case when $|\hat{\alpha}_i|$ is large. We have the following result.

Theorem 4.2 *Define* $s_i = \hat{\alpha}_i^2 d_i^2/2\sigma^2$, $\theta_i = (\tau^2 d_i^2)^{-1}$. *For any* $s_i \geq 1, \theta_i \geq 1$, *we have for the horseshoe regression of Equations (10–12) that*

$$\left\{1 - \theta_i(\tilde{C}_1 + \tilde{C}_2)\frac{(1+s_i)}{s_i^2} - \theta_i^2(\tilde{C}_1+\tilde{C}_2)^2\frac{(1+s_i)^2}{s_i^3}\right\} \leq \frac{SURE_i}{2\sigma^2} \leq \left\{1 + 2\theta_i(1+s_i)\left(\frac{C_1}{s_i^2} + \frac{C_2}{s_i^{3/2}}\right)\right\},$$

where $C_1 = \{1 - 5/(2e)\}^{-1/2} \approx 3.53, C_2 = 16/15, \tilde{C}_1 = (1-2/e)^{-1/2} \approx 1.95, \tilde{C}_2 = 4/3$, *are constants.*

A proof is given in Appendix A.2. Our result is non-asymptotic, i.e., it is valid for any $s_i \geq 1$. However, an easy consequence is that $SURE_i \to 2\sigma^2$, almost surely, as $s_i \to \infty$, provided $\tau^2 \leq d_i^{-2}$. An intuitive explanation of this result is that component-specific shrinkage is feasible in the horseshoe regression model due to the heavy-tailed λ_i terms, which prevents the signal terms from getting shrunk too much and consequently, making a large contribution to SURE due to a large bias. With just a global parameter τ, this component-specific shrinkage is not possible. A comparison of $SURE_i$ resulting from Theorem 4.2 with that from Equation (9) demonstrates using global-local horseshoe shrinkage, we can rectify a major shortcoming of global shrinkage regression, in that the terms with large s_i do not make a large contribution to the prediction risk. Moreover, the main consequence of Theorem 4.2, that is $SURE_i \to 2\sigma^2$, almost surely, as $s_i \to \infty$, holds for a larger class of "global-local" priors, of which the horseshoe is a special case.

Theorem 4.3 *Consider the hierarchy of Equations (10–11) and suppose the prior on* λ_i *in Equation (12) satisfies* $p(\lambda_i^2) \sim (\lambda_i^2)^{a-1}L(\lambda_i^2)$ *as* $\lambda_i^2 \to \infty$, *where* $f(x) \sim g(x)$ *means* $\lim_{x\to\infty} f(x)/g(x) = 1$. *Assume* $a \leq 0$ *and* $L(\cdot)$ *is a slowly-varying function, defined as* $\lim_{|x|\to\infty} L(tx)/L(x) = 1$ *for all* $t \in (0, \infty)$. *Then we have* $SURE_i \to 2\sigma^2$, *almost surely, as* $s_i \to \infty$.

A proof is given in Appendix A.3. Densities that satisfy $p(\lambda_i^2) \sim (\lambda_i^2)^{a-1}L(\lambda_i^2)$ as $\lambda_i^2 \to \infty$ are sometimes called regularly varying or heavy-tailed. Clearly, the horseshoe prior is a special case, since for the standard half-Cauchy we have $p(\lambda_i) \propto 1/(1+\lambda_i^2)$, which yields by a change of variables $p(\lambda_i^2) = (\lambda_i^2)^{-3/2}\{\lambda_i^2/(1+\lambda_i^2)\}$, which is of the form $(\lambda_i^2)^{a-1}L(\lambda_i^2)$ with $a = -1/2$ since $L(\lambda_i^2) = \lambda_i^2/(1+\lambda_i^2)$ is seen to be slowly-varying. Other priors that fall in this framework are the horseshoe+ prior of Bhadra et al. (2017), for which $p(\lambda_i) \propto \log(\lambda_i)/(\lambda_i^2 - 1) = \lambda_i^{-2}L(\lambda_i^2)$ with $L(\lambda_i^2) = \log(\lambda_i)\lambda_i^2/(\lambda_i^2-1)$. Ghosh et al. (2016) show that the generalized double Pareto prior (Armagan et al., 2013) and the three parameter beta prior (Armagan et al., 2011) also fall in this framework. Thus, Theorem 4.3 generalizes the main consequence of Theorem 4.2 to a broader class of priors in the asymptotic sense as $s_i \to \infty$.

Next, for the case when $|\hat{\alpha}_i|$ is small, we have the following result for estimating the prediction risk of the horseshoe regression.

Theorem 4.4 *Define* $s_i = \hat{\alpha}_i^2 d_i^2/2\sigma^2$ *and* $\theta_i = (\tau^2 d_i^2)^{-1}$. *Then the following statements are true for the horseshoe regression.*

A. $SURE_i$ *is an increasing function of* s_i *in the interval* $s_i \in [0, 1]$ *for any fixed* τ.

B. *When* $s_i = 0$, *we have that* $SURE_i$ *is a monotone increasing function of* τ, *and is bounded in the interval* $(0, 2\sigma^2/3]$ *when* $\tau^2 d_i^2 \in (0, 1]$.

C. *When* $s_i = 1$, *we have that* $SURE_i$ *is bounded in the interval* $(0, 1.93\sigma^2]$ *when* $\tau^2 d_i^2 \in (0, 1]$.

A proof is given in Appendix A.4. This theorem establishes that: (i) the terms with smaller s_i in the interval $[0, 1]$ contribute less to SURE, with the minimum achieved at $s_i = 0$ (these terms can be thought of as the noise terms) and (ii) if τ is chosen to be sufficiently small, the terms for which $s_i = 0$, has an upper bound on SURE at $2\sigma^2/3$. Note that the OLS estimator has risk $2\sigma^2$ for these terms. At $s_i = 0$, the PCR risk is either 0 or $2\sigma^2$, depending on whether the term is or is not included. A commonly used technique for shrinkage regressions is to choose the global τ to minimize a data-dependent estimate of the risk, such as C_L or SURE (Mallows, 1973). The ridge regression SURE at $s_i = 0$ is an increasing function of τ and thus, it might make sense to choose a small τ if all s_i terms were small. However, in the presence of some s_i terms that are large, ridge regression cannot choose a very small τ, since the large s_i terms will then be heavily shrunk and contribute too much to SURE. This is not the case with global-local shrinkage regression methods such as the horseshoe, which can still choose a small τ to mitigate the contribution from the noise terms and rely on the heavy-tailed λ_i terms to ensure large signals are not shrunk too much. Consequently, the ridge regression risk estimate is usually larger than the global-local regression risk estimate even for very small s_i terms, when some terms with large s_i are present along with mostly noise terms. At this point, the results concern the risk estimate (i.e., SURE) rather than risk itself, the discussion of which is deferred until Section 5.

To summarize the theoretical findings, Theorem 4.2 together with Theorem 4.4 establishes that the horseshoe regression is effective in handling both very large and very small values of $\hat{\alpha}_i^2 d_i^2$. Specifically, Theorem 4.4 asserts that a small enough τ shrinks the noise terms towards zero, minimizing their contribution to SURE. Whereas, according to Theorem 4.2, the heavy tails of the Cauchy priors for the λ_i terms ensure the large signals are not shrunk too much and ensures a SURE of $2\sigma^2$ for these terms, which is an improvement over purely global methods of shrinkage.

5. Prediction risk for the global and horseshoe regressions

In this section we compare the theoretical prediction risks of global and global-local horseshoe shrinkage regressions. While SURE is a data-dependent estimate of the theoretical risk, these two quantities are equal in expectation for all n. We use a concentration argument to derive conditions under which the horseshoe regression will outperform global shrinkage regression, e.g., ridge regression, in terms of predictive risk. While the analysis seems difficult for an arbitrary design matrix X, we are able to treat the case of ridge regression for orthogonal design, i.e., $X^T X = I$. Clearly, if the SVD of X is written as $X = UDV^T$, then we have $D = I$ and for ridge regression $\lambda_i = 1$ for all i. Thus, for orthogonal design, Equations (4) and (5) become

$$\begin{aligned}(\hat{\alpha}_i \mid \alpha_i, \sigma^2) &\overset{ind}{\sim} \mathcal{N}(\alpha_i, \sigma^2),\\ (\alpha_i \mid \sigma^2, \tau^2, \lambda_i^2) &\overset{ind}{\sim} \mathcal{N}(0, \sigma^2\tau^2),\end{aligned}$$

where τ is the global shrinkage parameter. Since the fit in this model is linear in $\hat{\alpha}_i$, SURE is equivalent to Mallows' C_L. Equation (14) of Mallows (1973) shows that if τ is chosen to

minimize C_L, then the optimal ridge estimate is given in closed form by

$$\alpha_i^\star = \left(1 - \frac{n\sigma^2}{\sum_{i=1}^n \hat{\alpha}_i^2}\right)\hat{\alpha}_i.$$

Alternatively, the solution can be directly obtained from Equation (9) by taking $d_i = \lambda_i = 1$ for all i and by setting $\tau^\star = \text{argmin}_\tau \sum_{i=1}^n SURE_i$. It is perhaps interesting to note that this "optimal" ridge estimate, where the tuning parameter is allowed to depend on the data, is no longer linear in $\hat{\alpha}$. In fact, the optimal solution $\alpha^\star$ can be seen to be closely related to the James–Stein estimate of α and its risk can therefore be quantified using the risk bounds on the James–Stein estimate. As expected due to the global nature of ridge regression, the relative shrinkage $\alpha_i^\star/\hat{\alpha}_i$ of the optimal solution only depends on $|\hat{\alpha}|^2 = \sum_{i=1}^n \hat{\alpha}_i^2$ but not on the individual components of $\hat{\alpha}$. Theorem 1 of Casella and Hwang (1982) shows that

$$1 - \frac{n-2}{n+|\alpha|^2} \le \frac{R(\alpha, \alpha^\star)}{R(\alpha, \hat{\alpha})} \le 1 - \frac{(n-2)^2}{n}\left(\frac{1}{n-2+|\alpha|^2}\right).$$

Consequently, if $|\alpha|^2/n \to c$ as $n \to \infty$ then the James–Stein estimate satisfies

$$\lim_{n\to\infty} \frac{R(\alpha, \alpha^\star)}{R(\alpha, \hat{\alpha})} = \frac{c}{c+1}.$$

Thus, $\alpha^\star$ offers large benefits over the least squares estimate $\hat{\alpha}$ for small c but it is practically equivalent to the least squares estimate for large c. The prediction risk of the least squares estimate for $p > n$ is simply $2n\sigma^2$, or an average component-specific risk of $2\sigma^2$. We first show that when true $\alpha_i = 0$, the component-specific risk bound of the horseshoe shrinkage regression with a fixed $\tau = 1$ (i.e., the case of purely local shrinkage) is less than $2\sigma^2$. We have the following result.

Theorem 5.1 *(Prediction risk for the purely local horseshoe regression). Let $D = I$ and let the global shrinkage parameter in the horseshoe regression be $\tau^2 = 1$. When true $\alpha_i = 0$, an upper bound of the component-wise risk of the purely local horseshoe regression is $1.75\sigma^2 < 2\sigma^2$.*

A proof can be found in Appendix A.5. The proof uses the fact that the actual risk can be obtained by computing the expectation of SURE. We split the domains of integration into three distinct regions and use the bounds on SURE from Theorems 4.2 and 4.4, as appropriate.

When true α_i is large enough, a consequence of Theorem 4.2 is that the component-specific risk for global-local shrinkage regression is $2\sigma^2$. This is because SURE in this case is almost surely equal to $2\sigma^2$ and $\hat{\alpha}_i$ is concentrated around true α_i. Therefore, it is established that if only a few components of true α are large and the rest are zero in such a way that $|\alpha|^2/n$ is large, then the horseshoe regression with fixed $\tau = 1$ outperforms ridge regression in terms of predictive risk. The benefit arises from a lower risk for the $\alpha_i = 0$ terms. On the other hand, if all components of true α are zero or all are large, the horseshoe regression need not outperform ridge regression.

Although Theorem 5.1 shows the horseshoe regression with a fixed $\tau = 1$ outperforms the optimal ridge regression in predictive risk when $\alpha = 0$, a useful corollary is that the optimal horseshoe regression still outperforms the optimal ridge regression, where the optimal global tuning parameters for both methods are chosen by minimizing their respective SURE.

Corollary 5.1 *(Prediction risk for the optimal horseshoe regression). Let* $SURE_{HS}(\tau = 1)$ *and* $SURE_{HS}(\tau = \tau^*_{HS})$ *denote the SURE for the horseshoe regression with fixed* $\tau = 1$ *and* $\tau = \tau^*_{HS} = \text{argmin}_\tau\,(SURE_{HS}(\tau))$*. Then, for any* α*,* $R(\alpha, \hat{\alpha}^{HS}(\tau = \tau^*_{HS})) \leq R(\alpha, \hat{\alpha}^{HS}(\tau = 1))$*.*

Proof

$$R(\alpha, \hat{\alpha}^{HS}(\tau = \tau^*_{HS})) = \mathbb{E}_{\hat{\alpha}|\alpha}(SURE_{HS}(\tau = \tau^*_{HS})) \leq \mathbb{E}_{\hat{\alpha}|\alpha}(SURE_{HS}(\tau = 1)) = R(\alpha, \hat{\alpha}^{HS}(\tau = 1)).$$

■

Clearly, τ^*_{HS} is a function of the data and this complicates exact prediction risk calculations for the optimal horseshoe regression as an expectation of SURE as in Theorem 5.1. It is not clear if an explicit minimizer of SURE analogous to Equation (14) of Mallows (1973) for ridge regression can be obtained for the horseshoe regression. Nevertheless, Corollary 5.1 shows the risk for the horseshoe regression can only decrease further if one sets $\tau = \tau^*_{HS}$, similar to the risk result of Stein (1956). This holds because the expectations of SURE are computed with respect to the distribution of $\hat{\alpha}$, which is independent of τ given true α.

6. Risk comparisons with other non-convex regressions

In this section we compare the risk of the proposed horseshoe regression with other approaches that are not shrinkage methods. Specifically, we consider the minimax concave penalty (MCP) of Zhang (2010). Again, for simplicity assume that the design matrix X is orthogonal. As pointed out by Zhang (2010), in this case the solution to the MCP estimator is available in closed form and reduces to the firm shrinkage estimator of Gao and Bruce (1997), which is given by

$$\delta_{\lambda,\gamma}(\hat{\alpha}_i) = \begin{cases} 0, & \text{if } |\hat{\alpha}_i| \leq \lambda, \\ \text{sign}(\hat{\alpha}_i)\frac{\gamma(|\hat{\alpha}_i| - \lambda)}{\gamma - 1}, & \text{if } \lambda \leq |\hat{\alpha}_i| \leq \gamma\lambda, \\ \hat{\alpha}_i, & \text{if } |\hat{\alpha}_i| > \gamma\lambda, \end{cases}$$

for $\lambda > 0$ and $\gamma > 1$. For a fixed λ, soft and hard thresholding estimators are obtained as $\gamma \to \infty$ and $\gamma \to 1+$ respectively. An explicit expression for the risk of this estimator is given in Theorem 1 of Gao and Bruce (1997), from which it can be seen easily that $R(\delta_{\lambda,\gamma}) > \lambda^2\{1/2 - \Phi(-2\lambda)\}$ when $\alpha_i = \lambda$ for any fixed $\lambda > 0$ and $\gamma > 1$, where $\Phi(\cdot)$ is the standard normal distribution function. Thus, for MCP to work well, a small value for λ is essential. However, λ is the threshold below which the estimates are shrunk to zero and a large λ is favored in a "dense" situation, where there are many true parameters several standard deviations away from zero. While this behavior is not necessarily a problem for the MCP, since it is designed with a sparse situation in mind, it is perhaps desirable to avoid a large risk at a given λ. The horseshoe regression achieves exactly that, since its component-specific risk in a dense case is $2\sigma^2$ by Theorem 4.2. In Supplementary Section S.1 we verify that the MCP performs worse than both global and global-local shrinkage methods in a dense situation under a variety of designs X.

Table 1: The true orthogonalized regression coefficients α_{0i}, their ordinary least squared estimates $\hat{\alpha}_i$, and singular values d_i of the design matrix, for $n = 100$ and $p = 500$.

i	α_{0i}	$\hat{\alpha}_i$	d_i	$\hat{\alpha}_i d_i$
1	0.10	0.10	635.10	62.13
2	-0.44	-0.32	3.16	-1.00
...	...	...	...	...
5	-0.13	0.30	3.05	0.91
6	10.07	10.22	3.02	30.88
...	...	...	...	...
29	0.46	0.60	2.53	1.53
30	10.47	11.07	2.51	27.76
...	...	...	...	...
56	0.35	0.57	2.07	1.18
57	10.23	10.66	2.07	22.05
...	...	...	...	...
66	-0.00	-0.35	1.90	-0.66
67	11.14	11.52	1.88	21.70
...	...	...	...	...
95	-0.82	-0.56	1.42	-0.79
96	9.60	10.21	1.40	14.26
...	...	...	...	...
100	0.61	0.91	1.27	1.15

7. Numerical examples

We simulate data where $n = 100$, and consider the cases $p = 100, 200, 300, 400, 500$. Let B be a $p \times k$ factor loading matrix, with all entries equal to 1. Let F_i be $k \times 1$ matrix of factor values, with all entries drawn independently from $\mathcal{N}(0, 1)$. The ith row of the $n \times p$ design matrix X is generated by a factor model, with number of factors $k = 8$, as follows:

$$X_i = BF_i + \xi_i, \quad \xi_i \sim \mathcal{N}(0, 0.1), \quad \text{for} \quad i = 1, \ldots, n.$$

Thus, the columns of X are correlated. Let $X = UDW^T$ denote the singular value decomposition of X. The observations y are generated from Equation (3) with $\sigma^2 = 1$, where for the true orthogonalized regression coefficients α_0, the 6, 30, 57, 67, and 96th components are randomly selected as signals, and the remaining 95 components are noise terms. Coefficients of the signals are generated by a $\mathcal{N}(10, 0.5)$ distribution, and coefficients of the noise terms are generated by a $\mathcal{N}(0, 0.5)$ distribution. For the case $n = 100$ and $p = 500$, some of the true orthogonalized regression coefficients α_0, their ordinary least squared estimates $\hat{\alpha}$, and the corresponding singular values d of the design matrix, are shown in Table 1.

Table 2 lists the SURE for prediction and actual out of sample sum of squared prediction error (SSE) for the ridge, lasso, PCR and horseshoe regressions. Out of sample prediction error of the adaptive lasso is also included in the comparisons, although we are unaware of a formula for computing the SURE for the adaptive lasso. SURE for ridge and PCR can be

Table 2: SURE and average out of sample prediction SSE (standard deviation of SSE) on one training set and 200 testing sets for the competing methods for $n = 100$, for ridge regression (RR), the lasso regression (LASSO), the adaptive lasso (A_LASSO), principal components regression (PCR) and the horseshoe regression (HS). The lowest SURE in each row is in italics and the lowest average prediction SSE is in bold. A formula for SURE is unavailable for the adaptive lasso.

	RR		LASSO		A_LASSO	PCR		HS	
p	SURE	SSE	SURE	SSE	SSE	SURE	SSE	SURE	SSE
100	159.02	168.24	125.37	128.98	127.22	162.23	179.81	*120.59*	**126.33**
		(23.87)		(18.80)	(18.10)		(25.51)		(18.77)
200	187.38	174.92	140.99	132.46	151.89	213.90	191.33	*139.32*	**126.99**
		(21.13)		(18.38)	(20.47)		(22.62)		(17.29)
300	192.78	191.91	*147.83*	145.04	153.64	260.65	253.00	151.24	**136.67**
		(22.95)		(19.89)	(21.19)		(26.58)		(18.73)
400	195.02	182.55	148.56	165.63	178.98	346.19	292.02	*147.69*	**143.91**
		(22.70)		(21.55)	(20.12)		(28.98)		(18.41)
500	196.11	188.78	159.95	**159.56**	186.23	386.50	366.88	*144.97*	160.11
		(22.33)		(19.94)	(23.50)		(39.38)		(20.29)

computed by an application of Equation (9) and SURE for the horseshoe regression is given by Theorem 4.1. SURE for the lasso is calculated using the result given by Tibshirani and Taylor (2012). In each case, the model is trained on 100 samples. We report the SSE on 100 testing samples, averaged over 200 testing data sets, and their standard deviations. For ridge, lasso, PCR and horseshoe regression, the global shrinkage parameters were chosen to minimize SURE for prediction. In adaptive lasso, the shrinkage parameters were chosen by cross validation due to SURE being unavailable. It can be seen that SURE in most cases are within one standard deviation of the actual out of sample prediction SSE, suggesting SURE is an accurate method for evaluating actual out of sample prediction performance. When $p = 100, 200, 300, 400$, horseshoe regression has the lowest prediction SSE. When $p = 500$, SSE of the lasso and horseshoe regression are close, and the lasso performs marginally better. The horseshoe regression also has the lowest SURE in all but one cases. Generally, SURE increases with p for all methods. The SURE for ridge regression approaches the OLS risk, which is $2n\sigma^2 = 200$ in these situations. SURE for PCR is larger than the OLS risk and PCR happens to be the poorest performer in most settings. Performance of the adaptive lasso also degrades compared to the lasso and the horseshoe, which remain the two best performers. Finally, the horseshoe regression outperforms the lasso in four out of the five settings we considered.

Figure 1 shows contribution to SURE by each component for $n = 100$ and $p = 500$, for ridge, PCR, lasso and horseshoe regressions. The components are ordered left to right on the x-axis by decreasing magnitude of d_i, and SURE for prediction on each component are shown on the y-axis. Note from Table 1 that the $6, 30, 57, 67$ and 96th components are the signals, meaning these terms correspond to a large α_0. The PCR risk on the 96th component is 203.22, which is out of range for the y-axis in the plot. For this data set, PCR selects

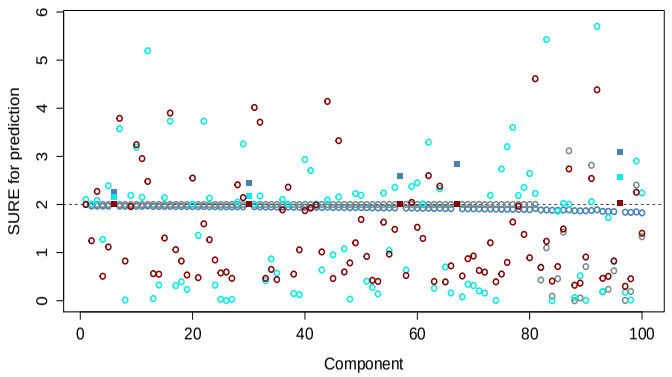


Figure 1: Component-wise SURE for ridge (blue), PCR (gray), lasso (cyan), and horseshoe regression (red), for $n = 100$ and $p = 500$. Signal components are shown in solid squares and noise components shown in blank circles. Dashed horizontal line is at $2\sigma^2 = 2$.

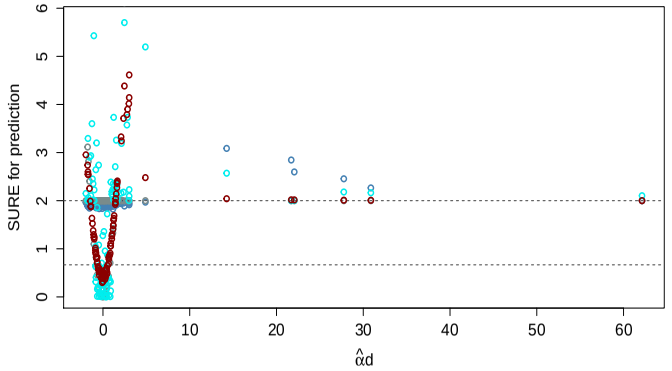


Figure 2: SURE for ridge (blue), PCR (gray), lasso (cyan) and horseshoe regression (red), versus $\hat{\alpha}d$, where $\hat{\alpha}$ is the OLS estimate of the orthogonalized regression coefficient, and d is the singular value, for $n = 100$ and $p = 500$. Dashed horizontal lines are at $2\sigma^2 = 2$ and $2\sigma^2/3 = 0.67$.

81 components, and therefore SURE for the first 81 components equal to $2\sigma^2 = 2$ and the SURE is equal to $\hat{\alpha}_i^2 d_i^2$ for $i = 82, \ldots, 100$. Component-wise SURE for ridge regression are large on the signal components, and is decreasing as the singular values d decrease on the other components. But due to the large global shrinkage parameter τ ridge must select in presence of both large signals and noise terms, the magnitude of improvement over the OLS risk $2\sigma^2$ is small for the noise terms. On the other hand, the horseshoe estimator does not shrink the components with large $\hat{\alpha}_i d_i$ heavily and therefore the horseshoe SURE on the signal components are almost equal to $2\sigma^2$ (according to Theorem 4.2). SURE for the horseshoe is also much smaller than $2\sigma^2$ on many of the noise components. Lasso also appears to be quite effective for the noise terms, but its performance for the signal components is generally not as effective as the horseshoe.

Figure 2 takes a fresh look at the same results and shows component-wise SURE plotted against $\hat{\alpha}_i d_i$. The signal components as well as the first component in Table 1 have $\hat{\alpha}_i d_i > 10$. Horseshoe SURE converges to $2\sigma^2$ for large $\hat{\alpha}_i d_i$, as expected from Theorem 4.2. For these components, the SURE for both ridge and lasso are larger than $2\sigma^2$, due to the bias introduced in estimating large signals by these methods (see also Theorem 1 of Carvalho et al., 2010). When ${\hat{\alpha}_i}^2 d_i^2 \approx 0$, risks for lasso and horseshoe are comparable, with lasso being slightly better. This is because an estimate can be exactly zero for the lasso, but not for the horseshoe, which is a shrinkage method (as opposed to a selection method). Nevertheless, the upper bound on SURE for the horseshoe regression at $2\sigma^2/3$ when ${\hat{\alpha}_i}^2 d_i^2 \approx 0$ and provided τ is chosen to be small enough so that $\tau^2 \leq d_i^{-2}$, as established by Theorem 4.4, can be verified from Figure 2.

Additional simulation results are presented in Supplementary Section S.1, where we (i) treat a higher dimensional case ($p = 1000$), (i) perform comparisons with non-convex MCP (Zhang, 2010) and SCAD (Fan and Li, 2001) regressions, (iii) explore different choices of X and (iv) explore the effect of the choice of α. The main finding is that the horseshoe regression is often the best performer when α has a sparse-robust structure as in Table 1, that is most elements are very small while a few are large so that $|\alpha|^2$ is large. This is consistent with the theoretical results of Sections 5 and 6.

8. Assessing out of sample prediction in a pharmacogenomics data set

We compare the out of sample prediction error of the horseshoe regression with ridge regression, PCR, the lasso, the adaptive lasso, MCP and SCAD on a pharmacogenomics data set. The data were originally described by Szakács et al. (2004), in which the authors studied 60 cancer cell lines in the publicly available NCI-60 database (https://dtp.cancer.gov/discovery_development/nci-2011), the normal-exponential-gamma (Griffin and Brown, 2010), the generalized double 60/). The goal here is to predict the expression of the human ABC transporter genes (responses) using some compounds or drugs (predictors) at which 50% inhibition of cellular growth for the cell lines are induced. The NCI-60 database includes the concentration level of 1429 such compounds, out of which we use 853, which did not have any missing values, as predictors. We investigate the expression levels of transporter genes A1 to A12 (except for A11, which we omit due to missing values), and B1. Thus, in our study X is a $n \times p$ matrix of predictors with $n = 60, p = 853$ and Y is a n-dimensional response vector for each of the 12 candidate transporter genes under consideration.

To test the performance of the methods, we split each data set into training and testing sets, with 75% (45 out of 60) of the observations in the training sets. We standardize each response by subtracting the mean and dividing by the standard deviation. We fit the model on the training data, and then calculate mean squared prediction error (prediction MSE) on the testing data. This is repeated for 20 random splits of the data into training and testing sets. The tuning parameters in ridge regression, the lasso, the adaptive lasso, SCAD and MCP are chosen by five-fold cross validation on the training data. Similarly, the number of components in PCR and the global shrinkage parameter τ for horseshoe regression are chosen by cross validation as well. It is possible to use SURE to select the tuning parameters or the number of components, but one needs an estimate of the standard deviation of the errors in high-dimensional regressions. This is a problem of recent interest, as the OLS estimate of σ^2 is not well-defined in the $p > n$ case. Unfortunately, some of the existing methods we tried, such as the method of moments estimator of Dicker (2014), often resulted in unreasonable estimates for σ^2, such as negative numbers. Thus, we stick to cross validation here, as it is not necessary to estimate the residual standard deviation in that case.

The average prediction MSE over 20 random training-testing splits for the competing methods is reported in Table 3. Average prediction MSE for responses A1, A8 and A10 are around or larger than 1 for all of the methods. Since the responses are standardized before analysis, we might conclude that none of the methods performed well for these cases. Among the remaining nine cases, the horseshoe regression substantially outperforms the other methods for A3, A4, A9, A12 and B1. It is comparable to PCR for A5 and A7, and is comparable to the adaptive lasso for A6, which are the best performers in the respective cases. Overall, the horseshoe regression performed the best in 5 among the total 12 cases we considered.

9. Concluding remarks

We outlined some situations where the horseshoe regression is expected to perform better compared to some other commonly used "global" shrinkage or selection alternatives for high-dimensional regression. Specifically, we demonstrated that the global term helps in mitigating the prediction risk arising from the noise terms, and an appropriate choice for the tails of the local terms is crucial for controlling the risk due to the signal terms. For this article we have used the horseshoe prior as our choice for the global-local prior. However, in recent years, several other priors have been developed that fall in this class. This includes the horseshoe+ (Bhadra et al., 2017, 2016), the three-parameter beta (Armagan et al., Pareto (Armagan et al., 2013), the generalized shrinkage prior (Denison and George, 2012) and the Dirichlet–Laplace prior (Bhattacharya et al., 2015). Empirical Bayes approaches have also appeared (Martin and Walker, 2014) and the spike and slab priors have made a resurgence due to recently developed efficient computational approaches (Ročková and George, 2014; Ročková and George, 2016). Especially in the light of Theorem 4.3, we expect the results developed in this article for the horseshoe to foreshadow similar results when many of these alternatives are deployed. A particular advantage of using the horseshoe prior seems to be the tractable expression for SURE, as developed in Theorem 4.1.

Table 3: Average out of sample mean squared prediction error computed on 20 random training-testing splits (number of splits out of 20 with lowest prediction MSE), for each of the 12 human ABC transporter genes (A1–A10, A12, B1) in the pharmacogenomics example. Methods under consideration are ridge regression (RR), principal components regression (PCR) , the lasso, the adaptive lasso (A_LASSO), the minimax concave penalty (MCP), the smoothly clipped absolute deviation (SCAD) penalty, and the horseshoe regression (HS). Lowest prediction MSE and largest number of splits with the lowest prediction MSE for each response in bold.

Response	RR	PCR	LASSO	A_LASSO	MCP	SCAD	HS
A1	1.12	1.10	**1.00**	**1.00**	1.01	1.06	1.30
	(2)	(5)	**(7)**	(2)	(1)	(1)	(2)
A2	1.00	1.04	0.95	0.93	**0.92**	0.99	1.15
	(3)	(1)	**(7)**	(5)	(1)	(0)	(3)
A3	0.77	0.91	1.11	0.90	0.92	1.06	**0.65**
	(1)	(0)	(0)	(0)	(1)	(0)	**(18)**
A4	0.92	0.95	0.97	0.96	0.93	0.99	**0.79**
	(2)	(0)	(2)	(2)	(2)	(0)	**(12)**
A5	0.82	**0.77**	1.06	0.81	0.83	0.94	0.79
	(1)	**(6)**	(4)	(1)	(2)	(0)	**(6)**
A6	0.93	0.92	0.98	**0.86**	0.87	0.90	0.95
	(4)	(0)	(3)	(5)	(0)	(2)	**(6)**
A7	0.92	**0.83**	0.92	0.93	0.99	0.93	0.85
	(0)	**(8)**	(1)	(4)	(0)	(0)	(7)
A8	1.08	1.05	1.14	**1.01**	**1.01**	1.15	1.34
	(6)	(4)	**(6)**	(4)	(0)	(0)	(0)
A9	0.57	0.64	0.81	0.67	0.77	0.68	**0.55**
	(4)	(0)	(0)	(6)	(0)	(1)	**(9)**
A10	1.18	1.04	**1.00**	1.01	**1.00**	1.06	1.33
	(0)	**(7)**	(4)	(3)	(2)	(0)	(4)
A12	1.01	1.12	1.09	1.01	1.02	1.05	**0.80**
	(0)	(0)	(2)	(2)	(1)	(0)	**(15)**
B1	0.53	0.59	0.70	0.63	0.91	0.70	**0.46**
	(1)	(0)	(3)	(2)	(1)	(3)	**(10)**

Whether this advantage translates to some of the other global-local priors mentioned above is an open question. Following the approach of Stein (1981), our risk results are developed in a non-asymptotic setting (finite n, finite $p > n$). In the normal means model, finite sample risk properties in estimation under heavy-tailed priors have been considered by Polson and Scott (2012b). However, their work does not consider (a) predictive risk or (b) a linear regression model. Global-local priors such as the horseshoe and horseshoe+ are known to be minimax in estimation in the Gaussian sequence model (van der Pas et al., 2014, 2016). For linear regression, frequentist minimax risk results are discussed by Raskutti et al. (2011); and Castillo et al. (2015) have shown that spike and slab priors achieve minimax prediction

risk in regression. Whether the prediction risk for the horseshoe regression is optimal in an asymptotic sense is an important question to investigate and recent asymptotic prediction risk results for ridge regression (Dobriban and Wager, 2017) should prove helpful for comparing with global shrinkage methods. Another possible direction for future investigation might be to explore the implications of our findings on the predictive density in terms of an appropriate metric, say the Kullback-Leibler loss, following the results of George et al. (2006).

Acknowledgements

The authors are grateful for constructive suggestions by the reviewers and the action editor. Bhadra and Polson are supported by Grant No. DMS-1613063 by the US National Science Foundation.

Appendix A. Proofs

A.1. Proof of Theorem 4.1

Part A follows from Equation (7) with standard algebraic manipulations. To prove part B, define $Z_i = 1/(1+\tau^2\lambda_i^2 d_i^2)$. Then, from Equation (13)

$$\begin{aligned} m(\hat{\alpha}) &= (2\pi\sigma^2)^{-n/2}\prod_{i=1}^{n}\int_0^1 \exp(-z_i\hat{\alpha}_i^2 d_i^2/2\sigma^2) d_i z_i^{1/2}\left(\frac{z_i\tau^2 d_i^2}{1-z_i+z_i\tau^2 d_i^2}\right)\frac{1}{\tau d_i}(1-z_i)^{-1/2}z_i^{-3/2}dz_i \\ &= (2\pi\sigma^2)^{-n/2}\prod_{i=1}^{n}\int_0^1 \exp(-z_i\hat{\alpha}_i^2 d_i^2/2\sigma^2)(1-z_i)^{-1/2}\left\{\frac{1}{\tau^2 d_i^2}+\left(1-\frac{1}{\tau^2 d_i^2}\right)z_i\right\}^{-1}dz_i. \end{aligned}$$

From the definition of the compound confluent hypergeometric (CCH) density in Gordy (1998), the result of the integral is proportional to the normalizing constant of the CCH density and we have from Proposition 4.1 that,

$$m(\hat{\alpha}) \propto (2\pi\sigma^2)^{-n/2}\prod_{i=1}^{n} H\left(1,\frac{1}{2},1,\frac{\hat{\alpha}_i^2 d_i^2}{2\sigma^2},1,\frac{1}{\tau^2 d_i^2}\right).$$

In addition, the random variable $(Z_i \mid \hat{\alpha}_i,\sigma,\tau)$ follows a $\text{CCH}(1,1/2,1,\hat{\alpha}_i^2 d_i^2/2\sigma^2,1,1/\tau^2 d_i^2)$ distribution. Lemma 3 of Gordy (1998) gives,

$$\frac{d}{ds}H(p,q,r,s,\nu,\theta) = -\frac{p}{p+q}H(p+1,q,r,s,\nu,\theta).$$

This yields after some algebra that,

$$\begin{aligned} \frac{m'(\hat{\alpha}_i)}{m(\hat{\alpha}_i)} &= -\frac{2}{3}\frac{H\left(2,\frac{1}{2},1,\frac{\hat{\alpha}_i^2 d_i^2}{2\sigma^2},1,\frac{1}{\tau^2 d_i^2}\right)}{H\left(1,\frac{1}{2},1,\frac{\hat{\alpha}_i^2 d_i^2}{2\sigma^2},1,\frac{1}{\tau^2 d_i^2}\right)}\frac{\hat{\alpha}_i d_i^2}{\sigma^2}, \\ \frac{m''(\hat{\alpha}_i)}{m(\hat{\alpha}_i)} &= \frac{-\frac{2}{3}H\left(2,\frac{1}{2},1,\frac{\hat{\alpha}_i^2 d_i^2}{2\sigma^2},1,\frac{1}{\tau^2 d_i^2}\right)\frac{d_i^2}{\sigma^2}+\frac{8}{15}H\left(3,\frac{1}{2},1,\frac{\hat{\alpha}_i^2 d_i^2}{2\sigma^2},1,\frac{1}{\tau^2 d_i^2}\right)\frac{\hat{\alpha}_i^2 d_i^4}{\sigma^4}}{H\left(1,\frac{1}{2},1,\frac{\hat{\alpha}_i^2 d_i^2}{2\sigma^2},1,\frac{1}{\tau^2 d_i^2}\right)}. \end{aligned}$$

The correctness of the assertion

$$\frac{m'(\hat{\alpha}_i)}{m(\hat{\alpha}_i)} = -\frac{\hat{\alpha}_i d_i^2}{\sigma^2}\mathbb{E}(Z_i), \quad \text{and} \quad \frac{m''(\hat{\alpha}_i)}{m(\hat{\alpha}_i)} = -\frac{d_i^2}{\sigma^2}\mathbb{E}(Z_i) + \frac{\hat{\alpha}_i^2 d_i^4}{\sigma^4}\mathbb{E}(Z_i^2),$$

can then be verified using Equation (16), completing the proof.

A.2. Proof of Theorem 4.2

Define $s_i = \hat{\alpha}_i^2 d_i^2/2\sigma^2$ and $\theta_i = (\tau^2 d_i^2)^{-1}$, with$\theta_i \geq 1, s_i \geq 1$. From Theorem 4.1, the component-wise SURE is

$$\begin{aligned} SURE_i =& 2\sigma^2 - 2\sigma^2\mathbb{E}(Z_i) - \hat{\alpha}_i^2 d_i^2\{\mathbb{E}(Z_i)\}^2 + 2\hat{\alpha_i}^2 d_i^2\mathbb{E}(Z_i^2) \\ =& 2\sigma^2[1 - \mathbb{E}(Z_i) + 2s_i\mathbb{E}(Z_i^2) - s_i\{\mathbb{E}(Z_i)\}^2], \end{aligned} \tag{A.1}$$

Thus,

$$2\sigma^2[1 - \mathbb{E}(Z_i) - s_i\{\mathbb{E}(Z_i)\}^2] \leq SURE_i \leq 2\sigma^2[1 + 2s_i\mathbb{E}(Z_i^2)].$$

To find bounds on SURE, we need upper bounds on $\mathbb{E}(Z_i^2)$ and $\mathbb{E}(Z_i)$. Clearly, $\theta_i^{-1} \leq \{\theta_i + (1-\theta_i)z_i\}^{-1} \leq 1$, when $\theta_i \geq 1$. Let $a_i = \log(s_i^{5/2})/s_i$. Then $a_i \in [0, 5/(2e))$ when $s_i \geq 1$. Now,

$$\mathbb{E}(Z_i^2) = \frac{\int_0^1 z_i^2(1-z_i)^{-\frac{1}{2}}\{\theta_i + (1-\theta_i)z_i\}^{-1}\exp(-s_i z_i)dz_i}{\int_0^1 (1-z_i)^{-\frac{1}{2}}\{\theta_i + (1-\theta_i)z_i\}^{-1}\exp(-s_i z_i)dz_i},$$

An upper bound to the numerator of $\mathbb{E}(Z_i^2)$ can be found as follows.

$$\begin{aligned} &\int_0^1 z_i^2(1-z_i)^{-\frac{1}{2}}\{\theta_i + (1-\theta_i)z_i\}^{-1}\exp(-s_i z_i)dz_i \\ &\leq \int_0^1 z_i^2(1-z_i)^{-\frac{1}{2}}\exp(-s_i z_i)dz_i \\ &= \int_0^{a_i} z_i^2(1-z_i)^{-\frac{1}{2}}\exp(-s_i z_i)dz_i + \int_{a_i}^1 z_i^2(1-z_i)^{-\frac{1}{2}}\exp(-s_i z_i)dz_i \\ &\leq (1-a_i)^{-\frac{1}{2}}\int_0^{a_i} z_i^2\exp(-s_i z_i)dz_i + \exp(-a_i s_i)\int_{a_i}^1 z_i^2(1-z_i)^{-\frac{1}{2}}dz_i \\ &= (1-a_i)^{-\frac{1}{2}}\frac{2}{s_i^3}\left\{1 - \left(1 + a_i s_i + \frac{a_i^2 s_i^2}{2}\right)\exp(-a_i s_i)\right\} + \exp(-a_i s_i)\int_{a_i}^1 z_i^2(1-z_i)^{-\frac{1}{2}}dz_i \\ &\leq \{1 - 5/(2e)\}^{-\frac{1}{2}}\frac{2}{s_i^3} + \frac{1}{s_i^{5/2}}\int_0^1 z_i^2(1-z_i)^{-\frac{1}{2}}dz_i \\ &= \frac{C_1}{s_i^3} + \frac{C_2}{s_i^{5/2}}, \end{aligned}$$

where $C_1 = \{1 - 5/(2e)\}^{-\frac{1}{2}} \approx 3.53$ and $C_2 = \int_0^1 z_i^2(1-z_i)^{-\frac{1}{2}}dz_i = \Gamma(1/2)\Gamma(3)/\Gamma(3.5) = 16/15$. Similarly, a lower bound on the denominator of $\mathbb{E}(Z_i^2)$ is

$$\begin{aligned} &\int_0^1 (1-z_i)^{-\frac{1}{2}}\{\theta_i + (1-\theta_i)z_i\}^{-1}\exp(-s_i z_i)dz_i \\ &\geq \theta_i^{-1}\int_0^1 \exp(-s_i z_i)dz_i \\ &= \theta_i^{-1}\left\{\frac{1-\exp(-s_i)}{s_i}\right\} \geq \frac{1}{\theta_i(1+s_i)}, \end{aligned}$$

Thus, combining the upper bound on the numerator and the lower bound on the denominator

$$\mathbb{E}(Z_i^2) \leq \theta_i(1+s_i)\left(\frac{C_1}{s_i^3} + \frac{C_2}{s_i^{5/2}}\right).$$

Thus,

$$\begin{aligned} SURE_i &\leq 2\sigma^2[1 + 2s_i\mathbb{E}(Z_i^2)] \\ &\leq 2\sigma^2\left\{1 + 2\theta_i(1+s_i)\left(\frac{C_1}{s_i^2} + \frac{C_2}{s_i^{3/2}}\right)\right\}. \end{aligned} \tag{A.2}$$

An upper bound to the numerator of $\mathbb{E}(Z_i)$ can be found as follows. Let $\tilde{a}_i = \log(s_i^2)/s_i$. Then, $\tilde{a}_i \in [0, 2/e)$ for $s_i \geq 1$.

$$\begin{aligned} &\int_0^1 z_i(1-z_i)^{-\frac{1}{2}}\{\theta_i + (1-\theta_i)z_i\}^{-1}\exp(-s_i z_i)dz_i \\ &\leq \int_0^1 z_i(1-z_i)^{-\frac{1}{2}}\exp(-s_i z_i)dz_i \\ &= \int_0^{\tilde{a}_i} z_i(1-z_i)^{-\frac{1}{2}}\exp(-s_i z_i)dz_i + \int_{\tilde{a}_i}^1 z_i(1-z_i)^{-\frac{1}{2}}\exp(-s_i z_i)dz_i \\ &\leq (1-\tilde{a}_i)^{-\frac{1}{2}}\int_0^{\tilde{a}_i} z_i\exp(-s_i z_i)dz_i + \exp(-\tilde{a}_i s_i)\int_{a_i}^1 z_i(1-z_i)^{-\frac{1}{2}}dz_i \\ &= (1-\tilde{a}_i)^{-\frac{1}{2}}\frac{1}{s_i^2}\left\{1 - (1 + \tilde{a}_i s_i)\exp(-\tilde{a}_i s_i)\right\} + \exp(-\tilde{a}_i s_i)\int_{\tilde{a}_i}^1 z_i(1-z_i)^{-\frac{1}{2}}dz_i \\ &\leq (1-2/e)^{-\frac{1}{2}}\frac{1}{s_i^2} + \frac{1}{s_i^2}\int_0^1 z_i(1-z_i)^{-\frac{1}{2}}dz_i \\ &= \frac{\tilde{C}_1}{s_i^2} + \frac{\tilde{C}_2}{s_i^2}, \end{aligned}$$

where $\tilde{C}_1 = (1-2/e)^{-1/2} \approx 1.95$ and $\tilde{C}_2 = \int_0^1 z_i(1-z_i)^{-\frac{1}{2}}dz_i = \Gamma(1/2)\Gamma(2)/\Gamma(2.5) = 4/3$. The lower bound on the denominator is the same as before. Thus,

$$\mathbb{E}(Z_i) \leq \frac{\theta_i(1+s_i)}{s_i^2}\left(\tilde{C}_1 + \tilde{C}_2\right).$$

Thus,

$$\begin{aligned} SURE_i &\geq 2\sigma^2[1-\mathbb{E}(Z_i)-s_i\{\mathbb{E}(Z_i)\}^2] \\ &\geq 2\sigma^2\left\{1-\theta_i(\tilde{C}_1+\tilde{C}_2)\frac{(1+s_i)}{s_i^2}-\theta_i^2(\tilde{C}_1+\tilde{C}_2)^2\frac{(1+s_i)^2}{s_i^3}\right\}. \end{aligned} \tag{A.3}$$

Thus, combining Equations (A.2) and (A.3) we get

$$\left\{1-\theta_i(\tilde{C}_1+\tilde{C}_2)\frac{(1+s_i)}{s_i^2}-\theta_i^2(\tilde{C}_1+\tilde{C}_2)^2\frac{(1+s_i)^2}{s_i^3}\right\} \leq \frac{SURE_i}{2\sigma^2} \leq \left\{1+2\theta_i(1+s_i)\left(\frac{C_1}{s_i^2}+\frac{C_2}{s_i^{3/2}}\right)\right\},$$

for $s_i \geq 1, \theta_i \geq 1$.

A.3. Proof of Theorem 4.3

Our proof is similar to the proof of Theorem 1 of Polson and Scott (2011). Note from Equations (10–11) that integrating out α_i we have

$$\hat{\alpha}_i \mid \lambda_i^2, \sigma^2, \tau^2 \overset{ind}{\sim} \mathcal{N}(0, \sigma^2(d_i^{-2}+\tau^2\lambda_i^2)).$$

Let $p(\lambda_i^2) \sim (\lambda_i^2)^{a-1}L(\lambda_i^2)$, as $\lambda_i^2 \to \infty$ where $a \leq 0$. Define $u_i = \sigma^2(d_i^{-2}+\tau^2\lambda_i^2)$. Then, as in Theorem 1 of Polson and Scott (2011), we have

$$p(u_i) \sim u_i^{a-1}L(u_i), \text{ as } u_i \to \infty.$$

The marginal of $\hat{\alpha}_i$ is then given by

$$m(\hat{\alpha}_i) = \int \frac{1}{\sqrt{2\pi u_i}} \exp\{-\hat{\alpha}_i^2/(2u_i)\}p(u_i)du_i.$$

An application of Theorem 6.1 of Barndorff-Nielsen et al. (1982) shows that

$$m(\hat{\alpha}_i) \sim |\hat{\alpha}_i|^{2a-1}L(|\hat{\alpha}_i|) \text{ as } |\hat{\alpha}_i| \to \infty.$$

Thus, for large $|\hat{\alpha}_i|$

$$\frac{\partial \log m(\hat{\alpha}_i)}{\partial \hat{\alpha}_i} = \frac{(2a-1)}{|\hat{\alpha}_i|} + \frac{\partial \log L(|\hat{\alpha}_i|)}{\partial \hat{\alpha}_i}. \tag{A.4}$$

Clearly, the first term in Equation (A.4) goes to zero as $|\hat{\alpha}_i| \to \infty$. For the second term, we need to invoke the celebrated representation theorem by Karamata. A proof can be found in Bingham et al. (1989).

Result A.1 *(Karamata's representation theorem). A function L is slowly varying if and only if there exists $B > 0$ such that for all $x \geq B$ the function can be written in the form*

$$L(x) = \exp\left(\eta(x) + \int_B^x \frac{\varepsilon(t)}{t}\,dt\right),$$

where $\eta(x)$ is a bounded measurable function of a real variable converging to a finite number as x goes to infinity $\varepsilon(x)$ is a bounded measurable function of a real variable converging to zero as x goes to infinity.

Thus, using the properties of $\eta(x)$ and $\varepsilon(x)$ from the result above

$$\frac{d\log(L(x))}{dx} = \eta'(x) + \frac{\varepsilon(x)}{x} \to 0 \quad \text{as} \quad x \to \infty.$$

Using this in Equation (A.4) shows $\partial \log m(\hat{\alpha}_i)/\partial\hat{\alpha}_i \to 0$ as $|\hat{\alpha}_i| \to \infty$. By similar calculations, $\partial^2 \log m(\hat{\alpha}_i)/\partial^2\hat{\alpha}_i \to 0$ as $|\hat{\alpha}_i| \to \infty$. From Equation (7)

$$SURE_i = \sigma^4 d_i^{-2}\left\{\frac{\partial}{\partial\hat{\alpha}_i}\log m(\hat{\alpha}_i)\right\}^2 + 2\sigma^2\left\{1+\sigma^2 d_i^{-2}\frac{\partial^2}{\partial\hat{\alpha}_i^2}\log m(\hat{\alpha}_i)\right\}.$$

Thus, $SURE_i \to 2\sigma^2$, almost surely, as $|\hat{\alpha}_i| \to \infty$.

A.4. Proof of Theorem 4.4

The proof of Theorem 4.4 makes use of technical lemmas in Appendix A.6.

Recall from Appendix A.1 that if we define $Z_i = 1/(1+\tau^2\lambda_i^2 d_i^2)$ then the density of Z_i is given by

$$(Z_i \mid \hat{\alpha}_i, d_i, \tau, \sigma^2) \sim \text{CCH}\left(Z_i \mid 1, \frac{1}{2}, 1, \frac{\hat{\alpha}_i^2 d_i^2}{2\sigma^2}, 1, \frac{1}{\tau^2 d_i^2}\right). \tag{A.5}$$

Then SURE is given by $SURE = \sum_{i=1}^n SURE_i$ with

$$\begin{aligned} SURE_i &= 2\sigma^2[1-\mathbb{E}(Z_i)+2s_i\mathbb{E}(Z_i^2)-s_i\{\mathbb{E}(Z_i)\}^2] \\ &= 2\sigma^2[1-\mathbb{E}(Z_i)+s_i\mathbb{E}(Z_i^2)+s_i\text{Var}(Z_i)], \end{aligned} \tag{A.6}$$

where $s_i = \hat{\alpha}_i^2 d_i^2/2\sigma^2$. Thus,

$$\begin{aligned} \frac{\partial\{SURE_i\}}{\partial s_i} &= -2\sigma^2\frac{\partial\mathbb{E}(Z_i)}{\partial s_i} + 2\sigma^2\frac{\partial}{\partial s_i}\{s_i\mathbb{E}(Z_i^2)\} + 2\sigma^2\frac{\partial}{\partial s_i}\{s_i\text{Var}(Z_i)\} \\ &:= \text{I} + \text{II} + \text{III}. \end{aligned} \tag{A.7}$$

Now, as a corollary to Lemma A.1, $(\partial/\partial s_i)\mathbb{E}(Z_i) = \{\mathbb{E}(Z_i)\}^2 - \mathbb{E}(Z_i^2) = -\text{Var}(Z_i) < 0$, giving $\text{I} > 0$. The strict inequality follows from the fact that Z_i is not almost surely a constant for any $s_i \in \mathbb{R}$ and $(\partial/\partial s_i)\mathbb{E}(Z_i)$ is continuous at $s_i = 0$. Next, consider II. Define $\theta_i = (\tau^2 d_i^2)^{-1}$ and let $0 \leq s_i \leq 1$. Then,

$$\begin{aligned} \frac{\partial}{\partial s_i}\{s_i\mathbb{E}(Z_i^2)\} &= \mathbb{E}(Z_i^2) + s_i\frac{\partial}{\partial s_i}\mathbb{E}(Z_i^2) \\ &= \mathbb{E}(Z_i^2) + s_i\{\mathbb{E}(Z_i)\mathbb{E}(Z_i^2) - \mathbb{E}(Z_i^3)\} \qquad \text{(by Lemma A.1)} \\ &= s_i\mathbb{E}(Z_i)\mathbb{E}(Z_i^2) + \{\mathbb{E}(Z_i^2) - s_i\mathbb{E}(Z_i^3)\}. \end{aligned}$$

Now, clearly, the first term, $s_i\mathbb{E}(Z_i)\mathbb{E}(Z_i^2) \geq 0$. We also have $Z_i^2 - s_i Z_i^3 = Z_i^2(1-s_i Z_i) \geq 0$ a.s. when $0 \leq Z_i \leq 1$ a.s. and $0 \leq s_i \leq 1$. Thus, the second term $\mathbb{E}(Z_i^2) - s_i\mathbb{E}(Z_i^3) \geq 0$. Putting the terms together gives $\text{II} \geq 0$. Finally, consider III. Denote $\mathbb{E}(Z_i) = \mu_i$. Then,

$$\begin{aligned} \frac{\partial}{\partial s_i}\{s_i\text{Var}(Z_i)\} &= \text{Var}(Z_i) + s_i\frac{\partial}{\partial s_i}\{\text{Var}(Z_i)\} \\ &= \text{Var}(Z_i) - s_i\frac{\partial^2\mathbb{E}(Z_i)}{\partial s_i^2} \\ &= \mathbb{E}\{(Z_i-\mu_i)^2\} - s_i\mathbb{E}\{(Z_i-\mu_i)^3\} \qquad \text{(by Lemma A.2)} \\ &= \mathbb{E}[(Z_i-\mu_i)^2\{1-s_i(Z_i-\mu_i)\}]. \end{aligned}$$

Now, $(Z_i-\mu_i)^2\{1-s_i(Z_i-\mu_i)\}\geq 0$ a.s. when $0\leq Z_i\leq 1$ a.s. and $0\leq s_i\leq 1$ and thus, III ≥ 0. Using I, II and III in Equation (A.7) yields $SURE_i$ is an increasing function of s_i when $0\leq s_i\leq 1$, completing the proof of Part A.

To prove Part B, we need to derive an upper bound on SURE when $s_i=0$. First, consider $s_i=0$ and $0<\theta_i\leq 1$. we have from Equation (A.6) that $SURE_i=2\sigma^2(1-\mathbb{E}Z_i)$. By Lemma A.3, $(\partial/\partial\theta_i)\mathbb{E}(Z_i)>0$ and $SURE_i$ is a monotone decreasing function of θ_i, where $\theta_i=(\tau^2d_i^2)^{-1}$. Next consider the case where $s_i=0$ and $\theta_i\in(1,\infty)$. Define $\tilde{Z}_i=1-Z_i\in(0,1)$ when $Z_i\in(0,1)$. Then, by Equation (A.11) and a formula on Page 9 of Gordy (1998), we have that $\tilde{Z}_i$ also follows a CCH distribution. Specifically,

$$(\tilde{Z}_i\mid\hat{\alpha}_i,d_i,\tau,\sigma^2)\sim\text{CCH}\left(\tilde{Z}_i\mid\frac{1}{2},1,1,-\frac{\hat{\alpha}_i^2d_i^2}{2\sigma^2},1,\tau^2d_i^2\right),$$

and we have $SURE_i=2\sigma^2\mathbb{E}(\tilde{Z}_i)$. Define $\tilde{\theta}_i=\theta_i^{-1}=\tau^2d_i^2$. Then by Lemma A.3, $(\partial/\partial\tilde{\theta}_i)\mathbb{E}(\tilde{Z}_i)=-\text{Cov}(\tilde{Z}_i,\tilde{W}_i)>0$ on $0<\tilde{\theta}_i<1$. Therefore, $SURE_i$ is a monotone increasing function of $\tilde{\theta}_i$ on $0<\tilde{\theta}_i<1$, or equivalently a monotone decreasing function of θ_i on $\theta_i\in(1,\infty)$.

Thus, combining the two cases above, we get that SURE at $s_i=0$ is a monotone decreasing function of θ_i for any $\theta_i\in(0,\infty)$, or equivalently, an increasing function of $\tau^2d_i^2$. Since $0\leq\tilde{Z}_i\leq 1$ almost surely, a natural upper bound on $SURE_i$ is $2\sigma^2$. However, it is possible to do better provided τ is chosen sufficiently small. Assume that $\tau^2\leq d_i^{-2}$. Then, since $SURE_i$ is monotone increasing in θ_i, the upper bound on SURE is achieved when $\theta_i=(\tau^2d_i^2)^{-1}=1$. In this case, $\mathbb{E}(Z_i)$ has a particularly simple expression, given by

$$\begin{aligned}\mathbb{E}(Z_i)=&\frac{\int_0^1 z_i(1-z_i)^{-\frac{1}{2}}\{\theta_i+(1-\theta_i)z_i\}^{-1}dz_i}{\int_0^1(1-z_i)^{-\frac{1}{2}}\{\theta_i+(1-\theta_i)z_i\}^{-1}dz_i}\\ =&\frac{\int_0^1 z_i(1-z_i)^{-\frac{1}{2}}dz_i}{\int_0^1(1-z_i)^{-\frac{1}{2}}dz_i}=\frac{2}{3}.\end{aligned}\tag{A.8}$$

Thus, $\sup SURE_i=2\sigma^2(1-\mathbb{E}Z_i)=2\sigma^2/3$, completing the proof of Part B.

To prove Part C, we first note that when $s_i=1$ we have

$$SURE_i=2\sigma^2[1-\mathbb{E}(Z_i)|_{s_i=1}+2\mathbb{E}(Z_i^2)|_{s_i=1}-\{\mathbb{E}(Z_i)|_{s_i=1}\}^2]$$

where $E(Z_i)$ and $E(Z_i^2)$ are evaluated at $s_i=1$. Recall that when $\theta_i\geq 1$ and $z_i\in(0,1)$ we have $\theta_i^{-1}\leq\{\theta_i+(1-\theta_i)z_i\}^{-1}\leq 1$. Thus,

$$\begin{aligned}\mathbb{E}(Z_i^2)|_{s_i=1}=&\frac{\int_0^1 z_i^2(1-z_i)^{-\frac{1}{2}}\{\theta_i+(1-\theta_i)z_i\}^{-1}\exp(-z_i)dz_i}{\int_0^1(1-z_i)^{-\frac{1}{2}}\{\theta_i+(1-\theta_i)z_i\}^{-1}\exp(-z_i)dz_i}\\ \leq&\frac{\int_0^1 z_i^2(1-z_i)^{-\frac{1}{2}}\exp(-z_i)dz_i}{\theta_i^{-1}\int_0^1(1-z_i)^{-\frac{1}{2}}\exp(-z_i)dz_i}\approx\theta_i\frac{0.459}{1.076}=0.43\theta_i,\end{aligned}\tag{A.9}$$

and

$$\begin{aligned}\mathbb{E}(Z_i)|_{s_i=1}=&\frac{\int_0^1 z_i(1-z_i)^{-\frac{1}{2}}\{\theta_i+(1-\theta_i)z_i\}^{-1}\exp(-z_i)dz_i}{\int_0^1(1-z_i)^{-\frac{1}{2}}\{\theta_i+(1-\theta_i)z_i\}^{-1}\exp(-z_i)dz_i},\\ \geq&\frac{\theta_i^{-1}\int_0^1 z_i(1-z_i)^{-\frac{1}{2}}\exp(-z_i)dz_i}{\int_0^1(1-z_i)^{-\frac{1}{2}}\exp(-z_i)dz_i}\approx\theta_i^{-1}\frac{0.614}{1.076}=0.57\theta_i^{-1}.\end{aligned}\tag{A.10}$$

Thus,

$$SURE_i\leq 2\sigma^2\left[1-\frac{0.57}{\theta_i}+0.86\theta_i-\left(\frac{0.57}{\theta_i}\right)^2\right].$$

When $\theta_i=1$, it can be seen that $SURE_i\leq 1.93\sigma^2$.

A.5. Proof of Theorem 5.1

The proof of Theorem 5.1 makes use of technical lemmas in Appendix A.6.

Recall from Appendix A.1 that if we define $Z_i=1/(1+\tau^2\lambda_i^2d_i^2)$ then the density of Z_i is given by

$$(Z_i\mid\hat{\alpha}_i,d_i,\tau,\sigma^2)\sim\text{CCH}\left(Z_i\mid 1,1/2,1,s_i,1,\theta_i\right).\tag{A.11}$$

where $s_i=\hat{\alpha}_i^2d_i^2/2\sigma^2$ and $\theta_i=(\tau^2d_i^2)^{-1}$. Consider the case where $d_i=1$ for all i and $\tau^2=1$, i.e., $\theta_i=1$ for all i. From Equation (A.6), the risk estimate is $SURE=\sum_{i=1}^n SURE_i$ with

$$\begin{aligned}SURE_i&=2\sigma^2[1-\mathbb{E}(Z_i)+s_i\mathbb{E}(Z_i^2)+s_i\text{Var}(Z_i)],\\ &\leq 2\sigma^2[1-\mathbb{E}(Z_i)+s_i+s_i\text{Var}(Z_i)]=\check{R}_i.\end{aligned}$$

We begin by showing that the upper bound $\check{R}_i=2\sigma^2[1-\mathbb{E}(Z_i)+s_i+s_i\text{Var}(Z_i)]$ is convex in s_i when $s_i\in(0,1)$. It suffices to show $-\mathbb{E}(Z_i)$ and $s_i\text{Var}(Z_i)$ are separately convex. First, $(\partial^2/\partial^2 s_i)\mathbb{E}(Z_i)=\mathbb{E}\{(Z_i-\mu_i)^3\}\leq 0$, by Lemmas A.2 and A.4, proving $-\mathbb{E}(Z_i)$ is convex. Next,

$$\begin{aligned}\frac{\partial^2}{\partial s_i^2}\{s_i\text{Var}(Z_i)\}&=\frac{\partial}{\partial s_i}\left[\text{Var}(Z_i)+s_i\frac{\partial}{\partial s_i}\{\text{Var}(Z_i)\}\right]\\ &=2\frac{\partial}{\partial s_i}\{\text{Var}(Z_i)\}+s_i\frac{\partial^2}{\partial s_i^2}\{\text{Var}(Z_i)\}\\ &=-2\mathbb{E}(Z_i-\mu_i)^3-s_i\frac{\partial}{\partial s_i}\mathbb{E}(Z_i-\mu_i)^3\quad\text{(by Lemma A.2)}\\ &=-2\mathbb{E}(Z_i-\mu_i)^3+s_i\mathbb{E}(Z_i-\mu_i)^4,\quad\text{(by Lemma A.5)}\\ &\geq 0,\end{aligned}$$

where the last inequality follows by Lemma A.4. Thus, since $\check{R}_i$ is convex, it lies entirely below the straight line joining the two end points for $s_i\in(0,1)$. But $\check{R}_i|_{s_i=0}\leq 2\sigma^2/3=0.67\sigma^2$ (by Equation (A.8)) and

$$\check{R}_i|_{s_i=1}\leq 2\sigma^2\left[1-0.57+1+0.43-(0.57)^2\right]=3.07\sigma^2,$$

by Equations (A.9) and (A.10).Thus, by convexity

$$SURE_i \leq \check{R}_i \leq 0.67\sigma^2 + s_i(3.07 - 0.67)\sigma^2 = (0.67 + 2.4s_i)\sigma^2 \text{ for } s_i \in (0,1) \tag{A.12}$$

We remark here that our simulations suggest $SURE_i$ itself is convex, not just the upper bound $\check{R}_i$, although a proof seems elusive. Nevertheless, as we shall see below, the convexity of $\check{R}_i$ is sufficient for our purposes.

Next, consider the interval $s_i \in (1,3)$. Noting that both $\mathbb{E}(Z_i)$ and $\mathbb{E}(Z_i^2)$ are monotone decreasing functions of s_i we have

$$SURE_i \leq 2\sigma^2[1 - \mathbb{E}(Z_i)|_{s_i=3} + 2s_i\{\mathbb{E}(Z_i^2)|_{s_i=1}\} - s_i\{\mathbb{E}(Z_i)|_{s_i=3}\}^2]$$

But,

$$\mathbb{E}(Z_i)|_{s_i=3,\theta_i=1} = \frac{\int_0^1 z_i(1-z_i)^{-\frac{1}{2}}\exp(-3z_i)dz_i}{\int_0^1 (1-z_i)^{-\frac{1}{2}}\exp(-3z_i)dz_i} = 0.35.$$

$\mathbb{E}(Z_i^2)|_{s_i=1} < 0.43$ from Equation (A.9). Thus,

$$SURE_i \leq 2\sigma^2[1 - 0.35 + 0.86s_i - s_i(0.35)^2\}^2] = 2\sigma^2(0.65 + 0.74s_i) \text{ for } s_i \in (1,3). \tag{A.13}$$

Using the upper bound from Theorem 4.2,

$$SURE_i \leq 11.55\sigma^2 \text{ for } s_i \geq 3. \tag{A.14}$$

When $\alpha_i = 0$, we have that $\hat{\alpha}_i \sim \mathcal{N}(0, \sigma^2 d_i^{-2})$. Thus, $\hat{\alpha}_i^2 d_i^2/\sigma^2 \sim \chi^2(1)$. Since $s_i = \hat{\alpha}_i^2 d_i^2/2\sigma^2$ we have that $p(s_i) = (\pi)^{-1/2} s_i^{-1/2}\exp(-s_i)$ for $s_i \in (0,\infty)$. Combining Equations (A.12), (A.13) and (A.14) we have

$$\begin{aligned}
\text{Risk}_i = \mathbb{E}(SURE_i) \leq & \int_0^1 \sigma^2(0.67 + 2.4s_i)\pi^{-1/2}s_i^{-1/2}\exp(-s_i)ds_i \\
& + \int_1^3 2\sigma^2(0.65 + 0.74s_i)\pi^{-1/2}s_i^{-1/2}\exp(-s_i)ds_i \\
& + \int_3^\infty 11.55\sigma^2\pi^{-1/2}s_i^{-1/2}\exp(-s_i)ds_i \\
& = 1.75\sigma^2.
\end{aligned}$$

A.6. Technical lemmas

Lemma A.1 *If $Z \sim \text{CCH}(p,q,r,s,\nu,\theta)$, then $(\partial/\partial s)\mathbb{E}(Z^k) = \mathbb{E}(Z)\mathbb{E}(Z^k) - \mathbb{E}(Z^{k+1})$.*

Lemma A.2 *If $Z \sim \text{CCH}(p,q,r,s,\nu,\theta)$, then $(\partial^2/\partial^2 s)\mathbb{E}(Z) = -(\partial/\partial s)\text{Var}(Z) = \mathbb{E}\{(Z-\mu)^3\}$, where $\mu = \mathbb{E}(Z)$.*

Lemma A.3 *If $Z \sim \text{CCH}(p,q,r,s,\nu,\theta)$, then $(\partial/\partial\theta)\mathbb{E}(Z) = -\text{Cov}(Z,W)$, for $W = (1-\nu Z)\{\theta + (1-\theta)\nu Z\}^{-1}$. If $0 < \theta \leq 1$ then $(\partial/\partial\theta)\mathbb{E}(Z) > 0$.*

Lemma A.4 *If $Z \sim \text{CCH}(p,q,r,s,1,1)$ with $q > p$, then $\mathbb{E}(Z-\mu)^3 \leq 0$, where $\mu = \mathbb{E}(Z)$.*

Lemma A.5 *If $Z \sim \text{CCH}(p,q,r,s,\nu,\theta)$, then $(\partial/\partial s)\mathbb{E}(Z-\mu)^3 = -\mathbb{E}\{(Z-\mu)^4\}$, where $\mu = \mathbb{E}(Z)$.*

A.6.1. Proof of Lemma A.1

Let, $Z \sim \text{CCH}(p,q,r,s,\nu,\theta)$. Then for any integer k

$$\mathbb{E}(Z^k) = \frac{\int_0^{1/\nu} z^{k+p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}.$$

Thus,

$$\begin{aligned}
\frac{\partial}{\partial s}\mathbb{E}(Z^k) = & \frac{\int_0^{1/\nu} -z^{k+p}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz} \\
& - \left[\frac{\int_0^{1/\nu} z^{k+p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}\right. \\
& \left. \times \frac{\int_0^{1/\nu} -z^{p}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}\right] \\
= & -\mathbb{E}(Z^{k+1}) + \mathbb{E}(Z)\mathbb{E}(Z^k).
\end{aligned}$$

For an alternative proof directly using the $H(\cdot)$ functions, see Appendix D of Gordy (1998).

A.6.2. Proof of Lemma A.2

Let, $Z \sim \text{CCH}(p,q,r,s,\nu,\theta)$. From Lemma A.1, $(\partial/\partial s)\mathbb{E}(Z) = -\mathbb{E}(Z^2) + \{\mathbb{E}(Z)\}^2 = -\text{Var}(Z)$. Let $\mu = \mathbb{E}(Z)$. Then,

$$\begin{aligned}
\frac{\partial^2}{\partial s^2}\mathbb{E}(Z) = & -\frac{\partial}{\partial s}\text{Var}(Z) \\
= & -\frac{\partial}{\partial s}\left[\frac{\int_0^{1/\nu}(z-\mu)^2 z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}\right] \\
= & \frac{\int_0^{1/\nu}(z-\mu)^2 z^{p}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz} \\
& - \left[\frac{\int_0^{1/\nu}(z-\mu)^2 z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}\right. \\
& \left. \times \frac{\int_0^{1/\nu} z^{p}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta + (1-\theta)\nu z\}^{-r}\exp(-sz)dz}\right] \\
= & \text{Cov}(Z,(Z-\mu)^2) \\
= & \mathbb{E}[(Z-\mu)\{(Z-\mu)^2 - \mathbb{E}(Z-\mu)^2\}] \\
= & \mathbb{E}\{(Z-\mu)^3\} - \text{Var}(Z)\mathbb{E}(Z-\mu) = \mathbb{E}\{(Z-\mu)^3\}.
\end{aligned}$$

A.6.3. Proof of Lemma A.3

Let $Z \sim \text{CCH}(p, q, r, s, \nu, \theta)$ and $W = (1 - \nu Z)\{\theta + (1 - \theta)\nu Z\}^{-1}$. Then,

$$\begin{aligned}\frac{\partial}{\partial\theta}\mathbb{E}(Z) =& -\frac{\int_0^{1/\nu} z^p(1-\nu z)^q\{\theta+(1-\theta)\nu z\}^{-(r+1)}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}\\ &+\left[\frac{\int_0^{1/\nu} z^p(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}\right.\\ &\left.\times\frac{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q}\{\theta+(1-\theta)\nu z\}^{-(r+1)}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}\right]\\ =& -\mathbb{E}(ZW)+\mathbb{E}(Z)\mathbb{E}(W) = -\text{Cov}(Z, W).\end{aligned}$$

When $0 < \theta \leq 1$, it is obvious that Z and W are negatively correlated, and thus $-\text{Cov}(Z, W) > 0$.

A.6.4. Proof of Lemma A.4

Let $Z \sim \text{CCH}(p, q, r, s, 1, 1)$. Then,

$$\mathbb{E}(Z-\mu)^3 = \frac{\int_0^1 (z-\mu)^3 z^{p-1}(1-z)^{q-1}\exp(-sz)dz}{\int_0^1 z^{p-1}(1-z)^{q-1}\exp(-sz)dz},$$

which can be seen to have the same sign as the third central moment, or skewness of a $\text{Beta}(p, q)$ random variable, which is negative when $q > p$.

A.6.5. Proof of Lemma A.5

Let, $Z \sim \text{CCH}(p, q, r, s, \nu, \theta)$. Let $\mu = \mathbb{E}(Z)$. Then,

$$\begin{aligned}\frac{\partial}{\partial s}\mathbb{E}(Z-\mu)^3 =& -\frac{\int_0^{1/\nu}(z-\mu)^3 z^p(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}\\ &+\left[\frac{\int_0^{1/\nu}(z-\mu)^3 z^{p-1}(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}\right.\\ &\left.\times\frac{\int_0^{1/\nu} z^{p}(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}{\int_0^{1/\nu} z^{p-1}(1-\nu z)^{q-1}\{\theta+(1-\theta)\nu z\}^{-r}\exp(-sz)dz}\right]\\ =& -\text{Cov}(Z, (Z-\mu)^3)\\ =& -\mathbb{E}[(Z-\mu)\{(Z-\mu)^3 - \mathbb{E}(Z-\mu)^3\}]\\ =& -\mathbb{E}\{(Z-\mu)^4\} + \mathbb{E}(Z-\mu)^3\mathbb{E}(Z-\mu) = -\mathbb{E}\{(Z-\mu)^4\}.\end{aligned}$$

References

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, December 1974. ISSN 0018-9286. doi: 10.1109/TAC.1974.1100705. URL `http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1100705`.

Artin Armagan, Merlise Clyde, and David B Dunson. Generalized beta mixtures of Gaussians. In J. Shawe-Taylor, R. S. Zemel, P. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 523–531, 2011.

Artin Armagan, David B Dunson, and Jaeyong Lee. Generalized double Pareto shrinkage. *Statistica Sinica*, 23(1):119–143, 2013.

O Barndorff-Nielsen, John Kent, and Michael Sørensen. Normal variance-mean mixtures and z distributions. *International Statistical Review*, 50:145–159, 1982.

Anindya Bhadra, Jyotishka Datta, Nicholas G Polson, and Brandon Willard. Default bayesian analysis with global-local shrinkage priors. *Biometrika*, 103(4):955–969, 2016.

Anindya Bhadra, Jyotishka Datta, Nicholas G Polson, and Brandon Willard. The horseshoe+ estimator of ultra-sparse signals. *Bayesian Analysis*, 12:1105–1131, 2017.

Anirban Bhattacharya, Debdeep Pati, Natesh S. Pillai, and David B. Dunson. Dirichlet-Laplace priors for optimal shrinkage. *Journal of the American Statistical Association*, 110:1479–1490, 2015. doi: 10.1080/01621459.2014.960967. URL `http://dx.doi.org/10.1080/01621459.2014.960967`.

Nicholas H Bingham, Charles M Goldie, and Jef L Teugels. *Regular variation*, volume 27 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, Cambridge, 1989.

Carlos M Carvalho, Nicholas G Polson, and James G Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97:465–480, 2010.

George Casella and Jiunn Tzon Hwang. Limit expressions for the risk of James–Stein estimators. *Canadian Journal of Statistics*, 10(4):305–309, 1982.

Ismaël Castillo, Johannes Schmidt-Hieber, and Aad Van der Vaart. Bayesian linear regression with sparse priors. *The Annals of Statistics*, 43(5):1986–2018, 2015.

Hyonho Chun and Sündüz Keles. Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(1):3–25, 2010.

Merlise Clyde, Heather Desimone, and Giovanni Parmigiani. Prediction via orthogonalized model mixing. *Journal of the American Statistical Association*, 91(435):1197–1208, 1996.

Peter Craven and Grace Wahba. Smoothing noisy data with spline functions. *Numerische Mathematik*, 31(4):377–403, 1978.

David G. T. Denison and Edward I. George. *Bayesian prediction with adaptive ridge estimators*, volume 8 of *IMS Collections*, pages 215–234. Institute of Mathematical Statistics, Beachwood, Ohio, USA, 2012. doi: 10.1214/11-IMSCOLL815. URL http://dx.doi.org/10.1214/11-IMSCOLL815.

Lee H Dicker. Variance estimation in high-dimensional linear models. *Biometrika*, 101(2): 269–284, 2014.

Edgar Dobriban and Stefan Wager. High-dimensional asymptotics of prediction: Ridge regression and classification. *Annals of Statistics*, to appear, 2017.

Bradley Efron. Estimating the error rate of a prediction rule: improvement on cross-validation. *Journal of the American Statistical Association*, 78(382):316–331, 1983.

Bradley Efron. The estimation of prediction error: covariance penalties and cross-validation (with discussion). *Journal of the American Statistical Association*, 99(467):619–642, 2004.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

Dean P Foster and Edward I George. The risk inflation criterion for multiple regression. *The Annals of Statistics*, 22:1947–1975, 1994.

Ildiko E. Frank and Jerome H. Friedman. A statistical view of some chemometrics regression tools. *Technometrics*, 35(2):109–135, 1993. ISSN 00401706. URL http://www.jstor.org/stable/1269656.

Hong-Ye Gao and Andrew G Bruce. Waveshrink with firm shrinkage. *Statistica Sinica*, pages 855–874, 1997.

Edward I. George, Feng Liang, and Xinyi Xu. Improved minimax predictive densities under Kullback-Leibler loss. *The Annals of Statistics*, 34(1):78–91, 02 2006. doi: 10.1214/009053606000000155. URL http://dx.doi.org/10.1214/009053606000000155.

Prasenjit Ghosh, Xueying Tang, Malay Ghosh, and Arijit Chakrabarti. Asymptotic properties of bayes risk of a general class of shrinkage priors in multiple hypothesis testing under sparsity. *Bayesian Analysis*, 11(3):753–796, 09 2016. doi: 10.1214/15-BA973. URL http://dx.doi.org/10.1214/15-BA973.

Michael B Gordy. A generalization of generalized beta distributions. In *Finance and Economics Discussion Series*. Division of Research and Statistics, Division of Monetary Affairs, Federal Reserve Board, 1998.

Jim E Griffin and Philip J Brown. Inference with normal-gamma prior distributions in regression problems. *Bayesian Analysis*, 5(1):171–188, 2010.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. Springer, New York, 2nd edition, 2009. URL http://www-stat.stanford.edu/~tibs/ElemStatLearn/.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. doi: 10.1080/00401706.1970.10488634. URL http://www.tandfonline.com/doi/abs/10.1080/00401706.1970.10488634.

Ian T. Jolliffe. A note on the use of principal components in regression. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 31(3):300–303, 1982.

C. L. Mallows. Some comments on C_p. *Technometrics*, 15(4):661–675, 1973. doi: 10.1080/00401706.1973.10489103. URL http://dx.doi.org/10.1080/00401706.1973.10489103.

Ryan Martin and Stephen G. Walker. Asymptotically minimax empirical Bayes estimation of a sparse normal mean vector. *Electronic Journal of Statistics*, 8(2):2188–2206, 2014. doi: 10.1214/14-EJS949. URL http://dx.doi.org/10.1214/14-EJS949.

C. Masreliez. Approximate non-Gaussian filtering with linear state and observation relations. *IEEE Transactions on Automatic Control*, 20(1):107–110, Feb 1975. ISSN 0018-9286. doi: 10.1109/TAC.1975.1100882.

Rahul Mazumder, Jerome H Friedman, and Trevor Hastie. Sparsenet: Coordinate descent with nonconvex penalties. *Journal of the American Statistical Association*, 106(495): 1125–1138, 2011.

LR Pericchi and AFM Smith. Exact and approximate posterior moments for a normal location parameter. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 54:793–804, 1992.

Nicholas G Polson and James G Scott. Shrink globally, act locally: sparse Bayesian regularization and prediction. In J. M. Bernardo, M. J. Bayarri, James O. Berger, A. P. Dawid, David Heckerman, Adrian F. M. Smith, and Mike West, editors, *Bayesian Statistics 9*, Oxford, 2011. Oxford University Press.

Nicholas G Polson and James G Scott. Local shrinkage rules, Lévy processes, and regularized regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(2):287–311, 2012a.

Nicholas G. Polson and James G. Scott. On the half-Cauchy prior for a global scale parameter. *Bayesian Analysis*, 7(4):887–902, 12 2012b. doi: 10.1214/12-BA730. URL http://dx.doi.org/10.1214/12-BA730.

Garvesh Raskutti, Martin J Wainwright, and Bin Yu. Minimax rates of estimation for high-dimensional linear regression over ℓ_q-balls. *IEEE Transactions on Information Theory*, 57(10):6976–6994, 2011.

Veronika Ročková and Edward I George. The spike-and-slab lasso. *Journal of the American Statistical Association*, to appear, 2016.

Veronika Ročková and Edward I. George. EMVS: The EM approach to Bayesian variable selection. *Journal of the American Statistical Association*, 109(506):828–846, 2014. doi: 10.1080/01621459.2013.869223. URL http://dx.doi.org/10.1080/01621459.2013.869223.

Charles M Stein. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, pages 197–206, Berkeley, Calif., 1956. University of California Press. URL https://projecteuclid.org/euclid.bsmsp/1200501656.

Charles M. Stein. Estimation of the mean of a multivariate normal distribution. *The Annals of Statistics*, 9(6):1135–1151, 11 1981. doi: 10.1214/aos/1176345632. URL http://dx.doi.org/10.1214/aos/1176345632.

Gergely Szakács, Jean-Philippe Annereau, Samir Lababidi, Uma Shankavaram, Angela Arciello, Kimberly J. Bussey, William Reinhold, Yanping Guo, Gary D. Kruh, Mark Reimers, John N. Weinstein, and Michael M. Gottesman. Predicting drug sensitivity and resistance: Profiling ABC transporter genes in cancer cells. *Cancer Cell*, 6(2):129 – 137, 2004. ISSN 1535-6108. doi: http://dx.doi.org/10.1016/j.ccr.2004.06.026. URL http://www.sciencedirect.com/science/article/pii/S1535610804002065.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 58:267–288, 1996.

Ryan J. Tibshirani and Jonathan Taylor. Degrees of freedom in lasso problems. *The Annals of Statistics*, 40(2):1198–1232, 04 2012. doi: 10.1214/12-AOS1003. URL http://dx.doi.org/10.1214/12-AOS1003.

SL van der Pas, BJK Kleijn, and AW van der Vaart. The horseshoe estimator: Posterior concentration around nearly black vectors. *Electronic Journal of Statistics*, 8:2585–2618, 2014.

S.L. van der Pas, J-B. Salomond, and J. Schmidt-Hieber. Conditions for posterior contraction in the sparse normal means problem. *Electronic Journal of Statistics*, 10(1): 976–1000, 2016.

A. Zellner. On assessing prior distributions and Bayesian regression analysis with g-prior distributions. In P. K. Goel and A. Zellner, editors, *Bayesian Inference and Decision Techniques: Essays in Honor of Bruno de Finetti*, pages 233–243. Elsevier, North-Holland, 1986.

Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942, 2010.

Supplementary Material to
Prediction risk for the horseshoe regression

Anindya Bhadra
Department of Statistics, Purdue University, 250 N. University Street, West Lafayette, IN 47907, USA.
bhadra@purdue.edu

Jyotishka Datta
Department of Mathematical Sciences, University of Arkansas, Fayetteville, AR 72701, USA.
jd033@uark.edu

Yunfan Li
Department of Statistics, Purdue University, 250 N. University Street, West Lafayette, IN 47907, USA.
li896@purdue.edu

Nicholas G. Polson and Brandon Willard
The University of Chicago Booth School of Business, 5807 S. Woodlawn Ave., Chicago, IL 60637, USA.
ngp@chicagobooth.edu, brandonwillard@gmail.com

S.1. Additional simulations

We provide additional simulation results, complementing the results in Table 2. For each simulation setting, we report SURE when a formula is available. We also report the average out of sample prediction SSE (standard deviation of SSE) computed based on one training set and 200 testing sets. For each setting, $n = 100$. The methods under consideration are ridge regression (RR), principal components regression (PCR), the lasso, the adaptive lasso (A_LASSO), the minimax concave penalty (MCP), the smoothly clipped absolute deviation (SCAD) penalty and the proposed horseshoe regression (HS). The method with the **lowest SSE** is in bold and that with *lowest SURE* is in italics for each setting. The features of these additional simulations include the following.

1. We explore a higher dimensional case ($p = 1000$) for each setting.
2. We incorporate two non-convex regression methods for comparisons. These are SCAD (Fan and Li, 2001) and MCP (Zhang, 2010).
3. We explore different choices of the design matrix X. These include three cases: (i) X is generated from a factor model, where it is relatively ill-conditioned (as in Table 2), (ii) X is generated from a standard normal, where it is well-conditioned and (iii) X is exactly orthogonal, with all singular values equal to 1. These are reported in corresponding table captions.
4. We explore different choices of true α. These include three cases: (i) Sparse-robust α, where most elements of α are close to zero and a few are large, (ii) null α, where all elements of α are zero and (iii) dense α, where all elements are non-zero. Exact settings and the value of $||\alpha||^2$ are reported in the table captions.

The major finding is that the horseshoe regression outperforms the other global shrinkage methods (ridge and PCR) when α is sparse-robust, which is consistent with the theoretical observation in Section 5. It also outperforms the other selection-based methods in this case. On the other hand, the dense α case is most often favorable to ridge regression, while the null α case is favorable to selection-based methods such as the lasso, adaptive lasso, MCP or SCAD, due to the ability of these methods to produce exact zero estimates. However, the selection-based methods perform considerably worse compared to both global and global-local shrinkage methods in the dense α case.

Table S.1: Sparse-robust α (five large coefficients equal to 10 and other coefficients equal to 0.5 or -0.5 randomly, giving $\sum_{i=1}^{n} \alpha_i^2 = 523.75$); X generated by a factor model with 4 factors, each factor follows a standard normal distribution; d_1/d_n is the ratio of largest and smallest singular values of X.

		RR		PCR		LASSO		A_LASSO	MCP	SCAD	HS	
p	d_1/d_n	SURE	SSE	SURE	SSE	SURE	SSE	SSE	SSE	SSE	SURE	SSE
100	2360.43	165.45	159.83	163.80	161.62	122.78	145.07	132.25	127.07	127.85	*116.01*	**123.07**
			(22.02)		(21.28)		(19.39)	(17.57)	(16.71)	(17.19)		(16.43)
200	28.47	188.13	206.39	217.40	244.71	174.48	162.94	**148.41**	154.01	157.73	*160.89*	152.37
			(28.61)		(29.80)		(24.44)	(22.48)	(23.17)	(23.23)		(22.75)
300	22.76	192.35	212.05	266.84	280.25	*155.26*	190.09	175.46	172.18	176.29	157.50	**164.17**
			(28.50)		(32.62)		(26.20)	(22.17)	(21.55)	(22.19)		(22.85)
400	21.81	194.73	199.36	337.32	328.48	179.45	182.89	197.25	199.08	198.40	*172.63*	**165.15**
			(28.75)		(34.79)		(27.41)	(25.02)	(25.52)	(25.31)		(24.67)
500	18.18	196.03	180.12	410.82	379.03	158.07	173.82	223.21	224.91	226.76	*166.10*	**161.77**
			(27.16)		(39.41)		(26.50)	(27.98)	(29.65)	(29.26)		(24.22)
1000	15.20	197.91	184.86	669.69	736.69	196.83	205.28	345.26	344.04	344.04	*191.64*	**182.18**
			(26.42)		(56.58)		(29.56)	(36.60)	(37.34)	(37.34)		(25.43)

Table S.2: Null α ($\sum_{i=1}^{n} \alpha_i^2 = 0$); X is the same as in Table S.1.

	RR		PCR		LASSO		A_LASSO	MCP	SCAD	HS	
p	SURE	SSE	SURE	SSE	SURE	SSE	SSE	SSE	SSE	SURE	SSE
100	*88.23*	100.86	92.85	113.28	87.36	100.81	**100.70**	100.81	100.81	92.42	102.31
		(13.20)		(14.91)		(13.29)	(13.21)	(13.29)	(13.29)		(13.72)
200	121.30	107.68	128.83	115.65	*117.90*	105.77	**100.32**	104.39	101.78	122.29	111.39
		(15.70)		(16.28)		(15.06)	(14.80)	(14.93)	(14.89)		(16.12)
300	125.78	101.36	139.96	124.37	*108.85*	111.85	**101.30**	104.89	102.91	119.67	112.00
		(13.99)		(17.35)		(15.37)	(14.02)	(14.27)	(14.00)		(15.76)
400	113.00	99.50	113.50	99.41	*102.81*	111.92	114.62	**99.40**	110.30	113.42	107.20
		(13.12)		(13.09)		(15.51)	(15.80)	(13.20)	(15.20)		(14.90)
500	90.74	101.04	*88.26*	107.31	90.26	99.49	**99.06**	99.49	99.49	101.55	102.93
		(14.17)		(15.08)		(14.16)	(14.04)	(14.16)	(14.16)		(14.68)
1000	88.86	100.34	85.67	103.47	*82.51*	100.43	**99.52**	100.43	100.41	99.73	104.84
		(14.00)		(14.29)		(13.90)	(13.70)	(13.90)	(14.00)		(14.96)

Table S.3: Dense α (all coefficients equal to 2, giving $\sum_{i=1}^{n} \alpha_i^2 = 400$); X is the same as in Table S.1.

	RR		PCR		LASSO		A_LASSO	MCP	SCAD	HS	
p	SURE	SSE	SURE	SSE	SURE	SSE	SSE	SSE	SSE	SURE	SSE
100	*162.49*	**159.94**	177.47	175.19	194.86	203.89	504.55	491.67	491.67	185.46	173.89
		(21.60)		(22.36)		(28.11)	(46.13)	(45.31)	(45.31)		(23.63)
200	*183.75*	**200.92**	196.06	233.12	211.99	232.36	960.77	895.83	911.94	204.10	228.18
		(27.97)		(31.18)		(31.06)	(59.48)	(60.85)	(60.19)		(31.21)
300	*189.38*	**209.92**	200.39	225.92	216.01	524.84	1344.27	1298.80	1298.80	206.99	227.55
		(27.88)		(30.15)		(69.45)	(71.29)	(77.97)	(77.97)		(29.98)
400	*193.02*	**195.01**	197.74	217.68	218.16	306.15	1768.05	1675.73	1675.73	207.81	213.91
		(28.92)		(31.02)		(42.65)	(78.92)	(75.86)	(75.86)		(31.14)
500	*194.85*	**175.46**	208.87	201.18	220.34	743.40	2154.61	2082.54	2081.42	207.93	188.93
		(26.52)		(29.07)		(100.54)	(92.70)	(92.93)	(93.37)		(28.10)
1000	*197.37*	**181.65**	247.40	197.59	224.75	210.78	4280.80	4075.00	4075.00	203.47	186.48
		(26.50)		(27.76)		(29.70)	(145.28)	(138.72)	(138.72)		(26.95)

Table S.4: Sparse-robust α (five large coefficients equal to 10 and other coefficients equal to 0.5 or -0.5 randomly, giving $\sum_{i=1}^{n} \alpha_i^2 = 523.75$); X follows a standard normal distribution; d_1/d_n is the ratio of largest and smallest singular values of X.

		RR		PCR		LASSO		A_LASSO	MCP	SCAD	HS	
p	d_1/d_n	SURE	SSE	SURE	SSE	SURE	SSE	SSE	SSE	SSE	SURE	SSE
100	351.2	196.72	188.67	228.78	231.34	207.63	425.67	2537.23	2573.27	2573.27	*195.22*	**188.52**
			(29.04)		(34.36)		(59.68)	(112.58)	(128.46)	(128.46)		(28.90)
200	5.73	*199.84*	**193.41**	221.35	206.25	218.26	1618.40	4849.94	4915.72	4964.26	201.91	194.14
			(28.36)		(28.28)		(211.54)	(146.61)	(186.74)	(186.55)		(28.45)
300	3.63	*199.91*	**217.43**	8538.46	8082.93	222.62	1926.13	13132.01	7316.38	7316.39	200.92	219.97
			(27.91)		(320.98)		(248.97)	(281.12)	(218.82)	(218.83)		(28.13)
400	2.89	*199.94*	**197.47**	228.38	223.43	224.53	2384.04	9593.41	9695.47	9695.47	200.31	197.60
			(27.43)		(31.13)		(299.58)	(210.42)	(323.72)	(323.72)		(27.46)
500	2.51	*199.95*	**193.86**	256.09	273.63	224.96	471.73	11980.15	11991.11	11991.11	200.15	194.17
			(27.26)		(33.97)		(60.52)	(235.77)	(272.80)	(272.80)		(27.24)
1000	1.88	199.98	185.85	605.64	560.45	222.18	5781.13	23866.06	24566.13	24566.13	*199.96*	**185.79**
			(27.17)		(45.94)		(759.08)	(326.06)	(941.16)	(941.16)		(27.17)

Table S.5: Null α ($\sum_{i=1}^{n} \alpha_i^2 = 0$); X is the same as in Table S.4.

	RR		PCR		LASSO		A_LASSO	MCP	SCAD	HS	
p	SURE	SSE	SURE	SSE	SURE	SSE	SSE	SSE	SSE	SURE	SSE
100	118.45	119.12	96.35	106.88	*92.06*	101.21	**100.52**	101.21	101.21	119.11	114.69
		(18.19)		(15.18)		(14.33)	(14.20)	(14.33)	(14.33)		(17.47)
200	136.93	135.02	96.49	100.14	*94.54*	100.15	**100.13**	100.39	102.06	126.19	126.34
		(21.74)		(14.77)		(14.69)	(14.70)	(14.88)	(15.26)		(20.13)
300	152.52	160.29	119.00	131.94	*118.15*	100.71	**100.49**	100.71	100.71	140.91	140.08
		(21.61)		(18.01)		(14.48)	(14.37)	(14.48)	(14.48)		(18.93)
400	158.64	159.13	100.88	104.06	*96.30*	103.07	**100.46**	103.03	103.03	138.62	132.14
		(23.15)		(15.59)		(15.11)	(14.82)	(15.04)	(15.04)		(19.62)
500	166.06	158.83	98.64	98.10	*94.30*	100.36	**97.99**	100.36	100.36	140.14	131.53
		(23.35)		(14.50)		(14.79)	(14.50)	(14.79)	(14.79)		(19.59)
1000	181.23	169.22	89.95	100.66	*87.79*	100.07	**99.80**	100.66	100.51	141.11	138.94
		(25.25)		(14.10)		(14.03)	(14.00)	(14.08)	(14.07)		(21.12)

Table S.6: Dense α (all coefficients equal to 2, giving $\sum_{i=1}^{n} \alpha_i^2 = 400$); X is the same as in Table S.4.

	RR		PCR		LASSO		A_LASSO	MCP	SCAD	HS	
p	SURE	SSE	SURE	SSE	SURE	SSE	SSE	SSE	SSE	SURE	SSE
100	*193.13*	**188.60**	206.31	200.53	222.52	210.23	40019.73	40063.42	40063.42	199.25	191.74
		(28.91)		(29.51)		(31.36)	(690.71)	(717.27)	(717.27)		(29.37)
200	*199.76*	**193.93**	392.38	349.52	224.73	316.05	80016.11	80187.49	80187.49	200.14	194.48
		(28.41)		(37.90)		(42.77)	(983.86)	(1102.52)	(1102.52)		(28.50)
300	*199.88*	**217.60**	400.63	445.11	222.50	16845.87	120071.46	123161.75	123161.75	200.02	217.75
		(27.93)		(43.70)		(2167.53)	(1191.24)	(3757.26)	(3757.26)		(27.92)
400	*199.92*	**196.61**	627.97	618.38	222.51	43325.17	159926.82	161662.45	161662.45	200.00	196.70
		(27.35)		(59.10)		(5447.99)	(1418.60)	(3304.65)	(3304.65)		(27.35)
500	*199.94*	**193.02**	794.03	823.27	225.01	6497.32	199982.59	200043.69	200043.69	200.00	193.33
		(27.16)		(62.69)		(824.72)	(1550.64)	(1647.47)	(1647.47)		(27.18)
1000	*199.97*	**185.98**	2116.68	2108.78	224.77	3145.06	399770.90	400168.82	400168.82	200.03	186.02
		(27.17)		(101.50)		(411.60)	(2359.10)	(2934.25)	(2934.25)		(27.17)

Table S.7: Sparse-robust α (five large coefficients equal to 10 and other coefficients equal to 0.5 or -0.5 randomly, giving $\sum_{i=1}^{n} \alpha_i^2 = 523.75$); X with all singular values equal to 1.

	RR		PCR		LASSO		A_LASSO	MCP	SCAD	HS	
p	SURE	SSE	SURE	SSE	SURE	SSE	SSE	SSE	SSE	SURE	SSE
100	183.50	179.99	291.45	275.49	139.29	139.39	129.30	126.70	126.19	*131.81*	**122.60**
		(25.31)		(32.44)		(20.36)	(19.20)	(18.79)	(18.83)		(18.72)
200	184.47	196.14	261.65	277.35	135.93	150.17	**128.76**	129.25	129.58	*129.15*	131.16
		(28.79)		(33.30)		(21.76)	(17.75)	(17.61)	(17.90)		(18.63)
300	182.50	192.24	267.96	269.04	126.35	146.72	132.26	132.05	132.17	*119.09*	**128.72**
		(25.37)		(30.40)		(18.96)	(17.85)	(17.87)	(17.73)		(17.34)
400	184.03	178.58	311.01	287.95	145.28	139.68	128.94	127.57	127.41	*130.13*	**123.83**
		(25.20)		(32.98)		(19.20)	(17.40)	(17.42)	(17.43)		(17.02)
500	183.81	173.44	278.35	268.85	147.54	139.74	126.65	127.19	126.30	*132.70*	**120.78**
		(24.08)		(30.78)		(19.98)	(18.36)	(18.31)	(18.17)		(17.29)
1000	185.36	166.39	280.59	262.54	124.52	130.61	128.83	129.78	129.47	*119.24*	**125.49**
		(23.16)		(30.01)		(18.02)	(17.50)	(17.70)	(17.62)		(17.37)

Table S.8: Null α ($\sum_{i=1}^{n} \alpha_i^2 = 0$); X with all singular values equal to 1.

	RR		PCR		LASSO		A_LASSO	MCP	SCAD	HS	
p	SURE	SSE	SURE	SSE	SURE	SSE	SSE	SSE	SSE	SURE	SSE
100	94.70	**100.13**	97.63	102.62	*94.54*	100.15	**100.13**	100.15	100.15	99.92	101.44
		(14.71)		(15.37)		(14.69)	(14.70)	(14.69)	(14.69)		(15.06)
200	115.52	103.43	111.09	118.11	*109.16*	122.81	100.49	112.22	**100.80**	116.62	106.72
		(14.80)		(16.91)		(17.79)	(14.37)	(16.20)	(14.53)		(15.47)
300	98.74	100.49	99.45	113.35	*96.40*	103.03	**100.46**	103.03	103.03	103.10	102.40
		(14.80)		(16.49)		(15.04)	(14.82)	(15.04)	(15.04)		(14.87)
400	96.78	**97.99**	103.88	102.24	*94.02*	103.08	**97.99**	101.71	103.17	99.64	101.01
		(14.49)		(15.12)		(14.96)	(14.50)	(14.81)	(14.97)		(14.78)
500	88.55	**99.97**	89.06	100.91	*87.74*	100.65	99.98	100.65	100.65	93.63	101.53
		(14.81)		(14.72)		(14.87)	(14.83)	(14.87)	(14.87)		(14.98)
1000	88.87	**100.94**	91.96	107.30	*88.45*	101.14	100.95	101.62	101.14	94.34	102.26
		(14.17)		(15.40)		(14.14)	(14.17)	(14.30)	(14.14)		(14.48)

Table S.9: Dense α (all coefficients equal to 2, giving $\sum_{i=1}^{n} \alpha_i^2 = 400$); X with all singular values equal to 1.

	RR		PCR		LASSO		A_LASSO	MCP	SCAD	HS	
p	SURE	SSE	SURE	SSE	SURE	SSE	SSE	SSE	SSE	SURE	SSE
100	*177.89*	**183.69**	220.87	200.41	203.16	307.44	502.55	505.43	505.43	200.80	204.16
		(25.16)		(26.35)		(43.19)	(41.53)	(43.20)	(43.20)		(27.46)
200	*181.88*	**188.39**	207.49	239.95	214.11	255.88	499.88	498.84	498.84	205.16	217.46
		(27.10)		(33.45)		(35.61)	(41.69)	(42.90)	(42.90)		(30.81)
300	*176.64*	**193.53**	215.03	205.00	196.19	250.76	496.84	497.60	495.93	199.33	212.19
		(25.76)		(26.70)		(32.66)	(45.74)	(47.41)	(45.90)		(27.68)
400	*174.62*	**195.83**	248.90	249.51	209.80	221.41	495.21	494.17	494.17	198.84	206.49
		(26.36)		(32.46)		(29.85)	(40.50)	(40.89)	(40.89)		(28.41)
500	*179.13*	**173.13**	202.78	192.97	214.36	201.27	501.67	503.19	503.19	205.16	193.40
		(23.18)		(25.08)		(25.63)	(38.90)	(38.93)	(38.93)		(25.60)
1000	*179.32*	**173.71**	225.50	195.46	209.35	248.20	503.44	507.29	507.29	204.51	194.73
		(24.14)		(25.78)		(30.87)	(41.48)	(41.38)	(41.38)		(26.67)

Journal of Machine Learning Research 20 (2019) 1-33 Submitted 7/18; Revised 2/19; Published 4/19

Quantification Under Prior Probability Shift: the Ratio Estimator and its Extensions

Afonso Fernandes Vaz AFONSOFVAZ@GMAIL.COM
Rafael Izbicki RAFAELIZBICKI@GMAIL.COM
Rafael Bassi Stern RBSTERN@GMAIL.COM
Department of Statistics
Federal University of São Carlos
São Carlos, SP 13565-905, Brazil

Editor: Charles Elkan

Abstract

The quantification problem consists of determining the prevalence of a given label in a target population. However, one often has access to the labels in a sample from the training population but not in the target population. A common assumption in this situation is that of prior probability shift, that is, once the labels are known, the distribution of the features is the same in the training and target populations. In this paper, we derive a new lower bound for the risk of the quantification problem under the prior shift assumption. Complementing this lower bound, we present a new approximately minimax class of estimators, ratio estimators, which generalize several previous proposals in the literature. Using a weaker version of the prior shift assumption, which can be tested, we show that ratio estimators can be used to build confidence intervals for the quantification problem. We also extend the ratio estimator so that it can: (i) incorporate labels from the target population, when they are available and (ii) estimate how the prevalence of positive labels varies according to a function of certain covariates.

Keywords: quantification, prior probability shift, data set shift, domain shift, semi-supervised learning

1. Introduction

In several applications of binary classifiers, predicting the labels of individual observations *per se* is less important than evaluating the proportion of each label on an unlabeled target data set. The latter task is called quantification (Forman, 2008). For example, a company may be interested in evaluating the proportion of users who like each of their products, without access to labeled reviews of these products.

A common approach to such a problem is to (i) train a classifier for the user's evaluation based on labeled reviews of other products, and (ii) apply this classifier to the unlabeled target set and use the proportion of users who are classified as liking the product as an estimator. However, it is known that this two-step approach, known as "classify and count", fails because of domain shift (Forman, 2006; Tasche, 2016). In order to deal with this problem, several improvements have been proposed under an assumption named prior shift (Saerens et al., 2002; Forman, 2008; Bella et al., 2010; Barranquero et al., 2015). A particular estimator that successfully performs quantification is the adjusted count (AC) estimator (Gart and Buck, 1966; Saerens et al., 2002; Forman, 2008). Part of the success of the AC estimator is explained in Tasche (2017) by showing that it is Fisher consistent. However, there are more properties one might desire of an estimator.

In order to investigate these properties, Vaz et al. (2017) introduces the ratio estimator, which is a generalization of the AC estimator. Vaz et al. (2017) derives the asymptotic mean squared error of the ratio estimator. Here, we show that the ratio estimator is approximately minimax and consistent under the prior probability shift assumption. In order to derive this result, we prove a new lower bound for the risk of the quantification problem under the prior shift assumption. This lower bound is general and applies to every method under the prior probability shift assumption. We also derive a central limit theorem for the ratio estimator which helps to explain its good performance and leads to a method for building confidence intervals for the quantification problem. This result also allows us to propose a new type of ratio estimator based on Reproducing Kernel Hilbert spaces. Since the AC estimator and the method in Bella et al. (2010) are special cases of the ratio estimator, they benefit from all of the results above.

It is important to evaluate whether the prior probability shift assumption indeed holds, otherwise the AC method can perform poorly (Tasche, 2017). We show that the ratio estimator works under an assumption that is less stringent than the prior shift assumption. Moreover, we show how this assumption can be tested. We are not aware of other methods to test the prior shift and related assumptions.

We also generalize the ratio estimator to two extensions of the quantification problem. In the first scenario, some labels are available in the target population. The combined estimator extends the ratio estimator in order to incorporate these labels and obtain a larger effective sample size. The second scenario considers that the prevalence of each label varies according to additional covariates. This generalization allows one to use unlabeled data to identify e.g. how the approval of a product varies with age. In this scenario, we introduce the regression ratio estimator, which offers improvements over the standard methods that are used in sentiment analyses (Wang et al., 2012).

Section 2 discusses the standard quantification problem under the prior probability shift assumption. Subsection 2.1 provides new lower bounds for the risk in this scenario. Subsection 2.2 introduces the ratio estimator, uses the result from the previous subsection to show that it is approximately minimax and also derives its convergence rate and a central limit theorem. Subsection 2.3 uses the asymptotic behavior of the ratio estimator to propose a new type of ratio estimator based on Reproducing Kernel Hilbert spaces. Finally, the ratio estimator requires a weaker version of prior probability shift to obtain consistency. Subsection 2.4 discusses a new algorithm for testing this assumption.

Section 3 proposes extensions of the ratio estimator to scenarios which are more general than the standard quantification problem. Subsection 3.1 proposes the combined estimator, for cases in which some labels are available in the population of interest. Subsection 3.2 proposes the ratio regression estimator, for the situation in which the prevalence of a given label varies according to a covariate. All proofs are presented in the appendix; code and data used for the experiments is available at `https://github.com/afonsofvaz/ratio_estimator`.

©2019 Afonso Fernandes Vaz, Rafael Izbicki and Rafael Bassi Stern.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v20/18-456.html`.

2. Quantification under prior probability shift

In order to formally approach the quantification problem, we use the same notation as in Wasserman (2006). If $\mathbf{Z}_1 \in \mathbb{R}^{d_1}$ and $\mathbf{Z}_2 \in \mathbb{R}^{d_2}$ are random vectors and $R \subset \mathbb{R}^{d_1}$, then $\mathbb{P}(\mathbf{Z}_1 \in R|\mathbf{Z}_2)$ is the conditional probability that $\mathbf{Z}_1$ is in R given $\mathbf{Z}_2$. Using $\mathbb{P}$, one can obtain $F_{\mathbf{Z}_1|\mathbf{Z}_2}$, $f_{\mathbf{Z}_1|\mathbf{Z}_2}$, $\mathbb{E}[\mathbf{Z}_1|\mathbf{Z}_2]$, and $\mathbb{V}[\mathbf{Z}_1|\mathbf{Z}_2]$ which are, respectively, the conditional distribution, density, expected value and variance of $\mathbf{Z}_1$ given $\mathbf{Z}_2$. Marginal properties of $\mathbf{Z}_1$ are indicated by omitting the conditioning random variable. Also, if $(\mathbf{Z}_n)_{n\in\mathbb{N}}$ is a sequence of random vectors, then $\mathbf{Z}_n \xrightarrow{a.s.} \mathbf{Z}$, $\mathbf{Z}_n \xrightarrow{\mathbb{P}} \mathbf{Z}$, and $\mathbf{Z}_n \rightsquigarrow \mathbf{Z}$ indicate respectively, that $\mathbf{Z}_n$ converges almost surely, in probability, and in distribution to $\mathbf{Z}$. In order to express the rate at which convergence occurs, it is useful to use $\mathcal{O}$ and Ω notation. If $(a_n)_{n\in\mathbb{N}}$ is a sequence in $\mathbb{R}$, then $a_n = \mathcal{O}(g(n))$ if there exists c such that, for every n, $a_n \leq c \cdot g(n)$ and $a_n = \Omega(g(n))$ if there exists c such that, for every n, $a_n \geq c \cdot g(n)$. Finally, $\mathbb{I}$ is the indicator function. An expression such as $\mathbb{I}(g(\mathbf{X}) \in A)$ is equal to 1 when $g(\mathbf{X}) \in A$ and to 0 when $g(\mathbf{X}) \notin A$.

In the quantification problem, for each sample instance $i \in \{1, \ldots, n\}$, $(\mathbf{X}_i, Y_i, S_i)$ is a vector of random variables such that $\mathbf{X}_i \in \mathbb{R}^d$ are features, $Y_i \in \{0, 1\}$ is a label of interest and $S_i \in \{0, 1\}$ is the indicator that this instance has been labeled. That is, whenever $S_i = 0$, then Y_i is not observed. Note that S_i can be random.

In the above framework, some subsets of the instances are frequently used. The sets $A_k := \{i \in \{1, \ldots, n\} : S_i = k\}$ represent the labeled ($k = 1$) and unlabeled ($k = 0$) instances. Similarly, $A_{k,j} := \{i \in \{1, \ldots, n\} : S_i = k \text{ and } Y_i = j\}$ represent the instances that are labeled ($k = 1$) or unlabeled ($k = 0$) and have a positive ($j = 1$) or a zero ($j = 0$) label. Also the number of instances that are unlabeled, labeled or that have label j are denoted, respectively, by $n_U := |A_0|$, $n_L := |A_1|$ and $n_j := |A_{1,j}|$.

In a quantification problem, one wishes to estimate $\theta := \mathbb{P}(Y = 1|S = 0)$, that is, the prevalence of positive labels among unlabeled samples. This prevalence is not assumed to be the same as the one over labeled sets, $\mathbb{P}(Y = 1|S = 1)$. The estimator for θ can depend only on the available data, that is, the features of all instances and the labels that were obtained. Formally, letting Z_i^j denote $(Z_i, \ldots, Z_j)$, a valid estimator is a function of $\mathbf{X}_1^n$, S_1^n and $(Y_i)_{i\in A_1}$. The set of all such valid estimators is denoted by $\mathcal{S}$.

In the standard formulation of the prior probability shift problem, $\{(\mathbf{X}_i, Y_i)\}_{i\in A_0}$ is called the *target population* (since the labels are unavailable), and $\{(\mathbf{X}_i, Y_i)\}_{i\in A_1}$ is called the training population (Tasche, 2017). It is common for both populations to be i.i.d.,

Assumption 1

- $(S_1, \mathbf{X}_1, Y_1), \ldots, (S_n, \mathbf{X}_n, Y_n)$ *are independent.*
- *For every* $s \in \{0, 1\}$, $(\mathbf{X}_1, Y_1)|S_1 = s, \ldots, (\mathbf{X}_n, Y_n)|S_n = s$ *are identically distributed.*

Unless additional assumptions are made, it is not possible to learn about θ using solely the observed data. One assumption that allows learning about θ is the prior probability shift, which states that "the class-conditional feature distributions of the training and test sets are the same" (Fawcett and Flach, 2005). Prior shift is formalized in Assumption 2.

Assumption 2 ***[Prior probability shift]*** *For every* $(y_1, \ldots, y_n) \in \{0, 1\}^n$, $(\mathbf{X}_1, \ldots, \mathbf{X}_n)$ *is stochastically independent of* $(S_1, \ldots, S_n)$ *conditionally on* $(Y_1, \ldots, Y_n) = (y_1, \ldots, y_n)$.

Although Assumption 2 is written in a different way than in papers such as Moreno-Torres et al. (2012), the content is similar. While Moreno-Torres et al. (2012) uses a subscript on the probability function to determine which is the reference population, we perform this task using the random variable, S. For instance, the probability that an instance from the target population has the label "1" is referred in previous notation and in this paper, respectively, as $\mathbb{P}_{tg}(Y_i = 1)$ and $\mathbb{P}(Y_i = 1|S_i = 0)$. Using this translation, Assumption 2 is the same as the prior probability shift in Moreno-Torres et al. (2012). Assumption 2 holds if and only if $f_{\mathbf{X}|Y,S=0} \equiv f_{\mathbf{X}|Y,S=1}$, that is, $\mathbb{P}_{tg}(\mathbf{x}|y) \equiv \mathbb{P}_{tr}(\mathbf{x}|y)$.

2.1. Lower bound on the risk for quantification under prior probability shift

Under Assumptions 1 and 2 it is possible to learn about θ from the features and labels that are available in the quantification problem. For example, one can use the features and labels in the training population to learn about $f_{\mathbf{X}|Y=0}$ and $f_{\mathbf{X}|Y=1}$. Also, if these densities are sufficiently different, then one can combine the information about them to the features in the target population to learn about the unknown labels in this population and, therefore, about θ. Definition 1 formally presents two classes in which the possible values of $f_{\mathbf{X}|Y=0}$ and $f_{\mathbf{X}|Y=1}$ are separable.

Definition 1 *Let* $f_i(\mathbf{x}) = f_{\mathbf{X}|Y=i}(\mathbf{x})$, $\epsilon, K > 0$ *and* $g : \mathbb{R}^d \to \mathbb{R}$ *be a non-constant function.*

$$\begin{cases} \mathcal{F}_{\mathcal{L}_1,\epsilon} := \{(f_0, f_1) : \|f_0 - f_1\|_1 \geq \epsilon\} \\ \mathcal{F}_{g,K,\epsilon} := \left\{(f_0, f_1) : \mathbb{E}_{f_i}[g(\mathbf{X})^2|Y = i] \leq K, \text{ and } |\mathbb{E}_{f_1}[g(\mathbf{X})|Y = 1] - \mathbb{E}_{f_0}[g(\mathbf{X})|Y = 0]| \geq \epsilon\right\} \end{cases}$$

Under the classes in Definition 1 it is possible to learn about θ and the learning rate depends on both the number of labeled and unlabeled instances. A lower bound for how these sample sizes affect the rate at which one learns about θ is presented in Theorem 3.

Definition 2 *Let* $\mathcal{F}$ *be a collection of* (f_0, f_1). *The minimax rate,* $M(\mathcal{F})$, *for estimating* θ *under the squared loss,* $\mathcal{F}$, *and Assumptions 1 and 2 is*

$$M(\mathcal{F}) = \inf_{\widehat{\theta}\in\mathcal{S}} \sup_{(f_0,f_1)\in\mathcal{F};\theta\in[0,1]} \mathbb{E}_{f_0,f_1,\theta}\left[(\widehat{\theta} - \theta)^2 \middle| S_1^n\right]$$

Theorem 3 $M(\mathcal{F}_{\mathcal{L}_1,\epsilon}) \geq \Omega(\max(n_L^{-1}, n_U^{-1}))$ *and* $M(\mathcal{F}_{g,K,\epsilon}) \geq \Omega(\max(n_L^{-1}, n_U^{-1}))$.

Theorem 3 shows that it is not possible to obtain an estimator for θ which has convergence rate faster than $\Omega(\max(n_L^{-1}, n_U^{-1}))$. In particular, it is not possible to learn θ by observing solely a limited amount of labels. The following subsection introduces the ratio estimator for θ, which achieves the lower bound in Theorem 3 under $\mathcal{F}_{g,K,\epsilon}$.

2.2. The ratio estimator and its theoretical properties

Definition 4 (Ratio estimator) *Let* $g : \mathbb{R}^d \longrightarrow \mathbb{R}$. *The untrimmed ratio estimator for* θ *based on* g, $\widehat{\theta}_{UR}$, *is*

$$\widehat{\theta}_{UR} := \frac{\frac{\sum_{i\in A_0} g(\mathbf{X}_i)}{n_U} - \frac{\sum_{i\in A_{1,0}} g(\mathbf{X}_i)}{n_0}}{\frac{\sum_{i\in A_{1,1}} g(\mathbf{X}_i)}{n_1} - \frac{\sum_{i\in A_{1,0}} g(\mathbf{X}_i)}{n_0}}$$

Since $\theta \in [0,1]$, the ratio estimator, $\widehat{\theta}_R$, is

$$\widehat{\theta}_R = \max(0, \min(1, \widehat{\theta}_R))$$

The ratio estimator generalizes estimators which were previously proposed in the literature. This fact follows from observing that the terms in the untrimmed ratio estimator are sample averages of $g(\mathbf{X})$ among three groups of instances: unlabeled instances, instances labeled as 0, and instances labeled as 1. For instance, the adjusted count (AC) estimator (Gart and Buck, 1966; Saerens et al., 2002; Forman, 2008; Tasche, 2017) is the a ratio estimator when $g(\mathbf{x}) \in \{0,1\}$, that is, $g(\mathbf{x})$ is the output of a classifier for Y. Also, the estimator in Bella et al. (2010) is a ratio estimator when $g(\mathbf{x}) = \widehat{\mathbb{P}}(Y=1|\mathbf{x})$, that is, $g(\mathbf{x})$ is a soft classifier for Y.

Remark 5 *The ratio estimator can be generalized to the case in which $Y_i \in \{0,1,\ldots,k\}$. In this case, let $g : \mathbb{R}^d \to \mathbb{R}^k$ be a fixed function. By defining G as a $k \times (k+1)$ matrix such that $G_{i,j} = \mathbb{E}[g_i(\mathbf{X})|Y = j-1, S = 1]$, $p \in \mathbb{R}^{k+1}$ such that $p_i = \mathbb{P}(Y = j-1|S=0)$, and $g \in \mathbb{R}^k$ such that $g_i = \mathbb{E}[g_i(\mathbf{X})|S=0]$, $\widehat{\theta}_{UR}$ is obtained by solving the linear system*

$$\begin{cases} \widehat{g} &= \widehat{G} \cdot \widehat{\theta}_{UR} \\ 1 &= \mathbf{1}^t \cdot \widehat{\theta}_{UR} \end{cases}, \text{ where } \widehat{g}_i = \frac{\sum_{k \in A_0} g_i(\mathbf{X}_k)}{n_U} \text{ and } \widehat{G}_{i,j} = \frac{\sum_{k \in A_{1,j}} g_i(\mathbf{X}_k)}{n_j}$$

Since $\widehat{\theta}_{UR}$ might have negative components, it is generally inadmissible according to the squared error (de Finetti, 2017)[p.90-91] that is, there exist estimators which have a squared error strictly smaller than $\widehat{\theta}_{UR}$. The ratio estimator, $\widehat{\theta}_R$ satisfies this property and is the projection of $\widehat{\theta}_{UR}$ onto the simplex (Michelot, 1986): $\widehat{\theta}_R = \arg\min_{\hat{p}:\hat{p}\geq 0, \sum_i \hat{p}_i = 1} \|\widehat{\theta}_{UR} - \hat{p}\|_2^2$.

Similarly to the AC estimator (Tasche, 2017), the ratio estimator is Fisher consistent under weak assumptions.[1]. They are described in Assumptions 3 and 4.

Assumption 3 (Weak prior shift) *The function, g, is such that $g(\mathbf{X})_1^n$ is stochastically independent of $\mathbf{S}_1^n$ conditionally on $\mathbf{Y}_1^n = y_1^n$.*

Assumption 4 (Separability) *The function, g, is such that*

1. *$\mathbb{E}[g(\mathbf{X}_i)|Y_i = j, S_i = 1]$ are defined, for $j \in \{0,1\}$.*
2. *$\mathbb{E}[g(\mathbf{X}_i)|Y_i = 1, S_i = 1] - \mathbb{E}[g(\mathbf{X}_i)|Y_i = 0, S_i = 1] \neq 0$*

The condition in Assumption 3 is a relaxed type of prior probability shift that is strictly weaker than Assumption 2. Assumption 4 requires two more conditions of $g(\mathbf{x})$. According to condition 1, the population versions of the expectations in Definition 4 are defined. Condition 2 states that the ratio estimator calculated on these population parameters is defined, that is, there is no division by 0.

Theorem 6 *Under Assumptions 1, 3 and 4, $\widehat{\theta}_{UR}$ and $\widehat{\theta}_R$ are Fisher consistent for θ.*

1. Although Fisher consistency is typically not equivalent to consistency in probability (Gerow, 1989; Kass and Vos, 2011), in the sequence we show that the ratio estimator is consistent in both senses.

It is also possible to guarantee a finite population bound on the mean squared error of $\widehat{\theta}_R$. This result is obtained in Theorem 7, which substitutes Assumption 4 by the stronger condition that $(f_0, f_1) \in \mathcal{F}_{g,K,\epsilon}$.

Theorem 7 *Under Assumptions 1 and 3,*

$$\sup_{(f_0,f_1)\in\mathcal{F}_{g,K,\epsilon}} \mathbb{E}_{f_0,f_1}\left[\left(\widehat{\theta}_R - \theta\right)^2 \middle| S_1^n\right] \leq \mathcal{O}(\max(n_L^{-1}, n_U^{-1}))$$

Under the assumptions of Theorem 7, if $n_U \gg n_L$, then the convergence of the mean squared error of the ratio estimator is the same as the one that would have been obtained if one observed solely n_L labels from the target population and used the sample's label proportions to estimate θ. The same type of result cannot generally be obtained for the untrimmed ratio estimator, since the trimming is necessary to guarantee that the ratio of random variables does not have infinite variance. While these conclusions are similar to the ones obtained from Theorem 3 in Lipton et al. (2018), there exist two main differences. First, while the former assumes that there are 2 labels only, the latter applies to an arbitrary number of labels. Second, Theorem 7 upper bounds the squared error by $\mathcal{O}(\max(n_L^{-1}, n_U^{-1}))$, which is slightly tighter than the bound of $\mathcal{O}\left(\max\left(\frac{\log n_L}{n_L}, \frac{\log n_U}{n_U}\right)\right)$ in Lipton et al. (2018).

It follows from Theorem 3 and Theorem 7 that the ratio estimator satisfies several desirable properties. These properties are presented in Definition 8 and Corollary 9.

Definition 8 *Let $\mathcal{S}$ and $\mathcal{F}$ be, respectively, the classes of estimators and distributions over the data under consideration. An estimator $\widehat{\theta}^* \in \mathcal{S}$ is approximately minimax for estimating θ under the squared error loss if*

$$\mathcal{O}\left(\sup_{(f_0,f_1)\in\mathcal{F}_{g,K,\epsilon}} \mathbb{E}_{f_0,f_1}\left[\left(\widehat{\theta}^* - \theta\right)^2 \middle| S_1^n\right]\right) = \Omega\left(\inf_{\widehat{\theta}\in\mathcal{S}} \sup_{(f_0,f_1)\in\mathcal{F};\theta\in[0,1]} \mathbb{E}_{f_0,f_1,\theta}\left[(\widehat{\theta} - \theta)^2 \middle| S_1^n\right]\right)$$

That is, the squared error of $\widehat{\theta}^*$ attains the optimal rate of convergence.

Corollary 9 *Under Assumptions 1 and 3, if there exists $\epsilon, K > 0$ such that $(f_0, f_1) \in \mathcal{F}_{g,K,\epsilon}$, then $\widehat{\theta}_R$ is consistent for θ in probability and in $\mathcal{L}_2$ as $n_U \xrightarrow{\mathbb{P}} \infty$ and $n_L \xrightarrow{\mathbb{P}} \infty$. Also, under Assumptions 1, 3, and $\mathcal{F}_{g,K,\epsilon}$, $\widehat{\theta}_R$ is approximately minimax.*

Corollary 9 shows that the ratio estimator converges to θ under a weaker version of the prior probability shift assumption and that the rate of this convergence is minimax (i.e., it is the same rate as that of the minimax estimator). Since the estimators from Gart and Buck (1966); Saerens et al. (2002); Forman (2008); Bella et al. (2010) are particular cases of the untrimmed ratio estimator, their trimmed versions also converge to θ under the weak prior shift.

The ratio estimator also satisfies a central limit theorem. In order to obtain this result, besides requiring Assumptions 1, 3 and 4, it is also necessary to require that conditionally on Y, $g(\mathbf{X})$ has bounded variance and that the number of labeled samples goes to infinity. These conditions are described in Assumption 5. The central limit theorem is presented in Theorem 10.

Assumption 5

1. $\mathbb{V}[g(\mathbf{X}_i)|Y_i = j] < \infty$, *for every* $j \in \{0, 1\}$.

2. *There exists* $h(n) \geq 0$ *such that* $\lim_{n\to\infty} \frac{h(n)}{n} < 1$, $\lim_{n\to\infty} h(n) = \infty$, *and* $\frac{n_L}{h(n)} \xrightarrow{\mathbb{P}} 1$.

Theorem 10 *Define* $\mu_j := \mathbb{E}[g(\mathbf{X}_1)|Y_1 = j]$, $\sigma_j^2 := \mathbb{V}[g(\mathbf{X}_1)|Y_1 = j]$, $p_L := \lim_{n\to\infty} \frac{h(n)}{n}$, *and* $p_{j|L} := \mathbb{P}(Y = j|S = 1)$. *Under Assumptions 1, 3, 4 and 5,*

1. *If* $p_L \neq 0$, *then*

$$\sqrt{n}(\widehat{\theta}_R - \theta) \rightsquigarrow N\left(0, \frac{\frac{(1-\theta)\sigma_0^2 + \theta\sigma_1^2 + (\mu_1-\mu_0)^2\theta(1-\theta)}{1-p_L} + \frac{(1-\theta)^2\sigma_0^2}{p_L p_{0|L}} + \frac{\theta^2\sigma_1^2}{p_L p_{1|L}}}{(\mu_1 - \mu_0)^2}\right)$$

2. *If* $p_L = 0$, *then*

$$\sqrt{h(n)}(\widehat{\theta}_R - \theta) \rightsquigarrow N\left(0, \frac{\frac{(1-\theta)^2\sigma_0^2}{p_{0|L}} + \frac{\theta^2\sigma_1^2}{p_{1|L}}}{(\mu_1 - \mu_0)^2}\right)$$

It is possible to use Theorem 10 to obtain an approximate confidence interval for θ. This interval is obtained by inverting the convergence results in Theorem 10, and substituting θ for $\widehat{\theta}_R$ and the population parameters, μ_0, μ_1, σ_0^2, σ_1^2, p_L, $p_{0|L}$ and $p_{1|L}$, by their respective empirical averages. This confidence interval may also be used to test hypothesis such as $H_0 : \theta \in \Theta_0$.

Theorem 10 also provides an approximation for the mean squared error of $\widehat{\theta}_R$. This approximation for the common case in which $n_U \gg n_L$ is presented in the following corollary.

Corollary 11 *Under Assumptions 1, 3, 4 and 5, if* $p_L = 0$ ($n_U \gg n_L$), *then*

$$MSE(\widehat{\theta}_R) \approx \frac{1}{n_L(\mu_1 - \mu_0)^2}\left(\frac{\sigma_0^2(1-\theta)^2}{p_{0|L}} + \frac{\sigma_1^2\theta^2}{p_{1|L}}\right) \tag{1}$$

Corollary 11 brings some insights on how g should be chosen in order for $\widehat{\theta}_R$ to be an accurate estimator of θ. For instance, it shows that one should choose g such that $|\mu_1 - \mu_0|$ is large and both σ_0^2 and σ_1^2 are small. This implies that the distributions of $g(\mathbf{X})|Y = 1$ and $g(\mathbf{X})|Y = 0$ should place most of their masses in regions that are far apart. This conclusion explains the success of the methods in Forman (2008), in which $g(\mathbf{x})$ is a classifier, and Bella et al. (2010), in which $g(\mathbf{x})$ is an estimate of $\mathbb{P}(Y = 1|\mathbf{x})$.

One of the main deficiencies of the standard AC estimator is that its denominator can be very close to zero, which makes it very unstable (due to a large variance). In order to handle this, we can explicitly use the approximation of the MSE (Corollary 11) to choose better functions g. This procedure is discussed in the following subsection.

2.3. Choosing g via approximate MSE minimization

One possible criterion for the choice of g is the minimization of $MSE(\widehat{\theta}_R)$, defined in Corollary 11. However, the latter depends on unobservable quantities. An alternative is to minimize an estimate of $MSE(\widehat{\theta}_R)$. This estimate is presented in Definition 12.

Definition 12 *Let* $\hat{\theta}$ *be an estimator of* θ *and, for each* $i \in \{0, 1\}$, *let*

$$\widehat{\mu}_i = n_i^{-1}\sum_{A_{1,i}} g(\mathbf{X}_i) \qquad \widehat{\sigma}_i^2 = n_i^{-1}\sum_{A_{1,i}} (g(\mathbf{X}_i) - \widehat{\mu}_i)^2 \qquad \widehat{p}_{i|L} = \frac{n_i}{n_0 + n_1}$$

The empirical MSE of the ratio estimator induced by g, $\widehat{MSE}(g)$, *is*

$$\widehat{MSE}(g) \approx \frac{1}{n_L(\widehat{\mu}_1 - \widehat{\mu}_0)^2}\left(\frac{\widehat{\sigma}_0^2(1-\widehat{\theta})^2}{\widehat{p}_{0|L}} + \frac{\widehat{\sigma}_1^2\widehat{\theta}^2}{\widehat{p}_{1|L}}\right)$$

In order to avoid overfitting, we perform the minimization of $\widehat{MSE}(g)$ on a Reproducing Kernel Hilbert Space (RKHS; Wahba (1990)). More precisely, if K is a Mercer kernel and $\mathcal{H}_K$ is the RKHS associated to K, then we choose g^* as

$$g^* := \arg\min_{g\in\mathcal{H}_K} \widehat{\text{MSE}}(g) \tag{2}$$

In the following, Theorem 13 presents a characterization of g^* in eq. 2.

Theorem 13 *Let* K *be a Mercer kernel and* $\mathcal{H}_K$ *the corresponding RKHS. Also,*

- $\mathbb{K}$*: the Gram matrix defined for* $(i, j) \in A_1^2$ *and such that* $(\mathbb{K})_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.
- m_i*: A vector of size* $|A_1|$ *and such that, for each* $k \in A_1$, $m_{i,k} = \frac{\sum_{j\in A_{1,i}} K(\mathbf{x}_j, \mathbf{x}_k)}{n_i}$.
- $M = (m_1 - m_0)(m_1 - m_0)^t$.
- $\widehat{\Sigma}_i$*: a* $|A_1|\times|A_1|$ *matrix such that* $(\widehat{\Sigma}_i)_{k,l}$ *is the sample covariance between* $(K(\mathbf{x}_j, \mathbf{x}_k))_{j\in A_{1,i}}$ *and* $(K(\mathbf{x}_j, \mathbf{x}_l))_{j\in A_{1,i}}$.
- N*: a* $|A_1| \times |A_1|$ *matrix such that* $N = \frac{\widehat{\theta}^2}{\widehat{p}_{1|L}}\widehat{\Sigma}_1 + \frac{(1-\widehat{\theta})^2}{\widehat{p}_{0|L}}\widehat{\Sigma}_0$.
- $\mathbf{w}^* = \arg\min_{w\in\mathbb{R}^{n_L}} \frac{\mathbf{w}^t N\mathbf{w}}{\mathbf{w}^t M\mathbf{w}}$

The function g^* *in eq. 2 satisfies* $g^*(\mathbf{x}) = \sum_{i\in A_1} w_i^* K(\mathbf{x}, \mathbf{x}_i)$.

The vector, $\mathbf{w}^*$ in Theorem 13 is the eigenvector associated to the largest eigenvalue in absolute value, λ^*, of the generalized eigenvalue problem, $M\mathbf{w}^* = \lambda^* N\mathbf{w}^*$. If N is invertible, $\mathbf{w}^*$ is the eigenvector associated to the largest eigenvalue in absolute value of $N^{-1}M$. Alternatively, if N is not invertible one can substitute N in Theorem 12 by $(N + \gamma\mathbb{1})^{-1}$, where $\mathbb{1}$ is the identity matrix and γ is a small number that makes $N+\gamma\mathbb{1}$ invertible. Adding γ to the diagonal also also adds regularization and can therefore lead to an improved solution. In practice we choose γ via data-splitting.

The results in this and in the previous section rely on the weak prior shift assumption. As shown in the next subsection, one of the advantages of this assumption to the regular prior shift is that it is easier to test.

2.4. Testing the weak prior shift assumption

The following proposition is useful for testing the weak prior shift assumption:

Proposition 14 *Under Assumption 3, there exists* $0 \leq p \leq 1$ *such that*

$$pF_{g(\mathbf{X})|S=1,Y=1} + (1-p)F_{g(\mathbf{X})|S=1,Y=0} = F_{g(\mathbf{X})|S=0}.$$

It follows from Proposition 14 that Assumption 3 entails the hypothesis:

$$H_0 : \exists 0 \leq p \leq 1 \text{ such that } pF_{g(\mathbf{X})|S=1,Y=1} + (1-p)F_{g(\mathbf{X})|S=1,Y=0} = F_{g(\mathbf{X})|S=0}$$

In the following, we show how to construct an hypothesis test for H_0 when $g(\mathbf{x})$ is continuous. Since the weak prior shift entails H_0, if this test is used to test the weak prior shift, it will have the correct type I error. However, H_0 may hold when Assumption 3 is false. A specific example in which H_0 is satisfied but Assumption 3 is not satisfied is given in the following example.

Example 1 *If* $F_{g(\mathbf{X})|S=0,Y=1} = F_{g(\mathbf{X})|S=0,Y=0}$ *and* $F_{g(\mathbf{X})|S=0,Y=1} = F_{g(\mathbf{X})|S=1,Y=1}$*, then there exists* p *such that* $pF_{g(\mathbf{X})|S=1,Y=1} + (1-p)F_{g(\mathbf{X})|S=1,Y=0} = F_{g(\mathbf{X})|S=0}$ *(namely,* $p = 1$*) even if* $F_{g(\mathbf{X})|S=1,Y=0} \neq F_{g(\mathbf{X})|S=0,Y=0}$*. In this case, Assumption 3 does not hold and* H_0 *is satisfied.*

The following statistic, T, measures disagreement with H_0:

$$T = \inf_{0 \leq p \leq 1} d\left(p\widehat{F}_{g(\mathbf{X})|S=1,Y=1} + (1-p)\widehat{F}_{g(\mathbf{X})|S=1,Y=0}, \widehat{F}_{g(\mathbf{X})|S=0}\right),$$

where d is a distance between cumulative distributions, such as the Kolmogorov distance, and $\widehat{F}$ are the empirical cumulative distributions (Wasserman, 2013):

$$\widehat{F}_{g(\mathbf{X})|S=1,Y=i}(w) = \frac{1}{|A_{1,i}|}\sum_{i \in A_{1,i}} \mathbb{I}(g(\mathbf{X}_i) \leq w), \quad \widehat{F}_{g(\mathbf{X})|S=0}(w) = \frac{1}{|A_0|}\sum_{i \in A_0} \mathbb{I}(g(\mathbf{X}_i) \leq w).$$

Algorithm 1, which is presented below, obtains a p-value for H_0 based on T. The algorithm uses kernel smoothers (Wasserman, 2013) to estimate the conditional densities of $g(\mathbf{X})$ given Y, $f(g(\mathbf{x})|Y=0)$ and $f(g(\mathbf{x})|Y=1)$, by $\widehat{f}(g(\mathbf{x})|Y=0)$ and $\widehat{f}(g(\mathbf{x})|Y=1)$.

Note that our test is different from those proposed by Saerens et al. (2002) and Lipton et al. (2018). While our test evaluates whether the prior shift assumption is reasonable for the observed data, the above tests *assume prior shift* and evaluate whether the prevalence of positive labels in the unlabeled sample is the same as that in the labeled sample, that is, $\mathbb{P}(Y=1|S=0) = \mathbb{P}(Y=1|S=1)$.

The following subsection performs several experiments to test the performance of Algorithm 1 and of the ratio estimators which were discussed in previous subsections.

Algorithm 1 p-value for testing the weak prior shift assumption

Input: Labeled and unlabeled sample, number of Monte Carlo simulations, B **Output:** p-value

1: Compute T_{obs}, the test statistic for the observed data
2: Compute $\widehat{\theta}$, an estimate of θ
3: Compute $\widehat{f}(g(\mathbf{x})|Y=1)$ and $\widehat{f}(g(\mathbf{x})|Y=0)$.
4: **for all** $i \in \{1, \ldots, B\}$ **do**
5: Sample $W_1^{(0)}, \ldots, W_{n_0}^{(0)} \sim \widehat{f}(g(\mathbf{x})|Y=0)$ and $W_1^{(1)}, \ldots, W_{n_1}^{(1)} \sim \widehat{f}(g(\mathbf{x})|Y=1)$
6: Sample $W_1^{(U)}, \ldots, W_{n_U}^{(U)} \sim \widehat{\theta}\widehat{f}(g(\mathbf{x})|Y=1) + (1-\widehat{\theta})\widehat{f}(g(\mathbf{x})|Y=0)$
7: Compute T_i, the test statistic based on the labeled and unlabeled samples:

$$\{(W_i^{(0)}, Y_i = 0)\}_{i=1}^{n_0} \cup \{(W_i^{(1)}, Y_i = 1)\}_{i=1}^{n_1}; \qquad \{W_1^{(U)}, \ldots, W_{n_u}^{(U)}\}$$

8: **end for**
9: **return** $\frac{1}{B}\sum_{i=1}^{B} \mathbb{I}\left(T_{\text{obs}} \geq T_i\right)$

Estimator class	Specific method	Criteria for choosing $g(x)$
Classify and count		Logistic regression (LR), k-NN, random forest (RF).
Ratio	Forman (2006)	Logistic regression (LR), k-NN, random forest (RF).
	Bella et al. (2010)	Logistic regression (LR), k-NN, random forest (RF).
	RKHS	Linear kernel (Linear), Gaussian kernel (Gauss).
	EM (Saerens et al., 2002)	Logistic regression (LR), k-NN, random forest (RF).

Table 1: Methods compared in the experiments.

2.5. Experiments

Next, we compare the errors of the ratio estimator and of the classify and count estimator based on the estimator g when using various methods for obtaining g. We also include comparisons with the EM methods by Saerens et al. (2002). Table 1 summarizes all the variants that were tested.

We compare the above methods in five data sets: Candles (Freeman et al., 2013; Izbicki and Stern, 2013), Bank Marketing (Moro et al., 2011), SPAM e-mail (Blake, 1998), Wisconsin Breast Cancer (Mangasarian, 1990) and Blocks Classification (Malerba et al., 1996). Each database was transformed into a prior shift problem by choosing at random n_1 (n_0) instances among the ones labeled as 1 (0) to be marked as labeled instances and by choosing n_U instances to be marked as unlabeled. Each unlabeled unit is taken with probability θ randomly among the instances labeled as 1 in the original data set and with probability $1-\theta$ among those labeled as 0. The quantification sample sizes used in each of these data sets are described in Table 2. For all data sets we let θ vary in $\{0.1; 0.2; 0.3; 0.4; 0.5\}$ and repeated the generation and testing 100 times for each of the 11 methods in Table 1.

Figure 1 represents the average of the mean squared error (MSE; red point) and a confidence interval for the MSE (vertical blue bar) for each setting and method[2]. Figure 2 shows the number of experiments in which each method had the best average MSE,

2. For the sake of visualization, we omit all estimators based on K-NN on this plot. Figure 9 in Appendix B contains all methods.

	data set	n_U	n_L	n_1	n_0
1	cancer	100	300	150	150
2	candles	300	300	150	150
3	block	800	300	150	150
4	spam	2000	300	150	150
5	bank	10000	300	150	150

Table 2: Sample sizes for each data set.

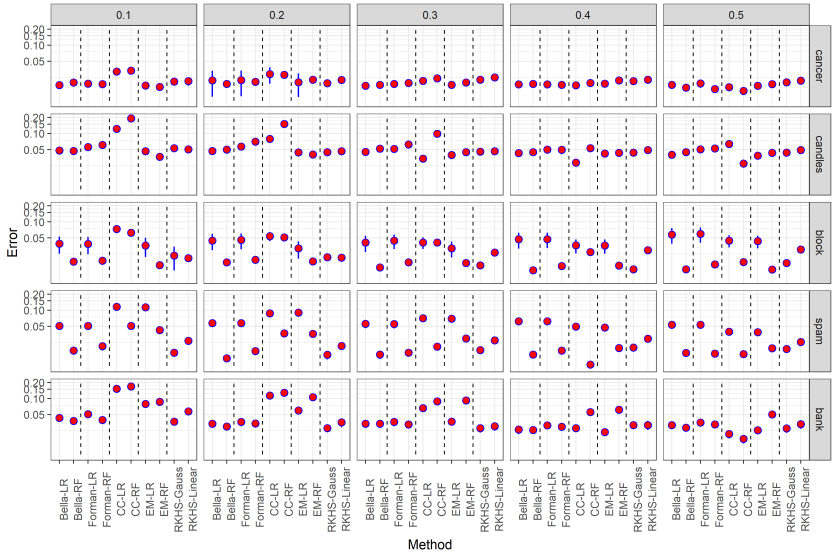


Figure 1: Root mean square deviation of each method by setting in logarithmic scale.

considering all data sets and θ values. These plots indicate that the ratio estimator generally performs better than the classify and count estimator for all choices of g. The main exception to this rule occurs when $\theta \approx 0.5$ and hence there is no prior shift. Also, the method in Bella et al. (2010) performs better than the one in Forman (2006) in essentially all scenarios. This suggests that soft classifiers might lead to better ratio estimators than hard classifiers. Moreover, the best performance is usually achieve when g is based on Random Forest, which corroborates that choosing a good classifier is key to having a good estimate of θ. The EM method was found to be very competitive in general, although the ratio estimators had better performance in some cases (e.g., for the bank data set). Finally, the RKHS approach is a competitive method, especially when using the Gaussian kernel. For additional figures related to this experiment see Appendix B.

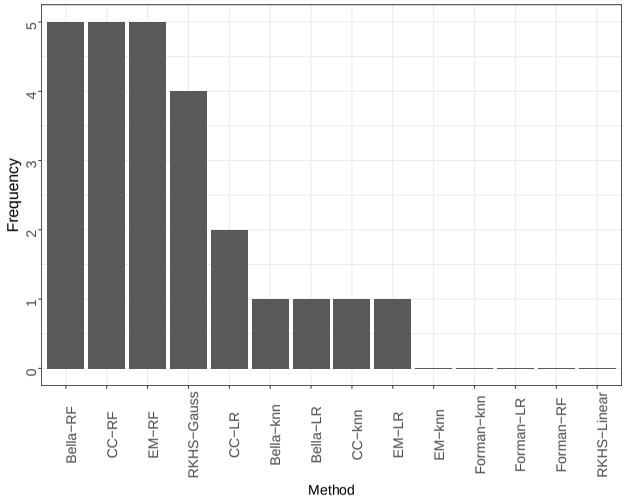


Figure 2: Number of times in which each method presented best MSE.

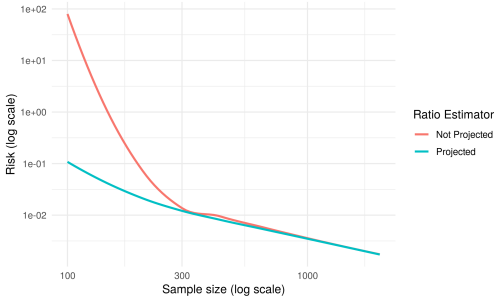


Figure 3: Mean squared error of the ratio estimator for the multiclass problem.

For all of ratio estimators, data sets and values for θ above, we construct confidence intervals for θ based on Theorem 10. We find that in all but one scenario the empirical coverage was at least as high as the specified value of 95%. The empirical coverage in the exception was 94%. The intervals constructed using Theorem 10 seem to be conservative.

Next, we simulate data using the following multiclass setting: $\mathbf{X}|Y = y \sim N(\mu_y, \Sigma)$ with $\Sigma = I_{10}$, $\mu_1 = (0, \ldots, 0)$, $\mu_2 = (0.75, \ldots, 0.75)$, $\mu_3 = (1.25, \ldots, 1.25)$, $\mathbb{P}(Y = 1|S = 1) = 0.2$,

Gaussian	**Exponential**
$g(\mathbf{x})\|S=1,Y=0 \sim \mathrm{N}(0,1)$,	$g(\mathbf{x})\|S=1,Y=0 \sim \mathrm{Exp}(1)$,
$g(\mathbf{x})\|S=1,Y=1 \sim \mathrm{N}(2,1)$,	$g(\mathbf{x})\|S=1,Y=1 \sim \mathrm{Exp}(5)$,
$g(\mathbf{x})\|S=0,Y=0 \sim \mathrm{N}(\gamma,1)$	$g(\mathbf{x})\|S=0,Y=0 \sim \mathrm{Exp}(\gamma)$
$g(\mathbf{x})\|S=0,Y=1 \sim \mathrm{N}(2,1)$,	$g(\mathbf{x})\|S=0,Y=1 \sim \mathrm{Exp}(5)$,
$\mathbb{P}(Y=1\|S=0)=0.6$,	$\mathbb{P}(Y=1\|S=0)=0.6$,
$\mathbb{P}(Y=1\|S=1)=0.2$	$\mathbb{P}(Y=1\|S=1)=0.2$
Gaussian-Exponential	**Beta**
$g(\mathbf{x})\|S=1,Y=0 \sim \mathrm{N}(1,1)$,	$g(\mathbf{x})\|S=1,Y=0 \sim \mathrm{Beta}(1,1)$,
$g(\mathbf{x})\|S=1,Y=1 \sim \mathrm{Exp}(1)$,	$g(\mathbf{x})\|S=1,Y=1 \sim \mathrm{Beta}(1,10)$,
$g(\mathbf{x})\|S=0,Y=0 \sim \mathrm{N}(\gamma,1)$	$g(\mathbf{x})\|S=0,Y=0 \sim \mathrm{Beta}(\gamma,1)$
$g(\mathbf{x})\|S=0,Y=1 \sim \mathrm{Exp}(1)$,	$g(\mathbf{x})\|S=0,Y=1 \sim \mathrm{Beta}(1,10)$,
$\mathbb{P}(Y=1\|S=0)=0.6$,	$\mathbb{P}(Y=1\|S=0)=0.6$,
$\mathbb{P}(Y=1\|S=1)=0.2$	$\mathbb{P}(Y=1\|S=1)=0.2$

Table 3: Scenarios used for testing the weak prior shift assumption.

$\mathbb{P}(Y=2|S=1)=0.3$ $\mathbb{P}(Y=1|S=0)=0.25$, and $\mathbb{P}(Y=2|S=0)=0.10$. We use a multivariate logistic regression to compute $g_1(\mathbf{x})=\widehat{P}(Y=1|\mathbf{x},S=1)$ and $g_2(\mathbf{x})=\widehat{P}(Y=2|\mathbf{x},S=1)$. Figure 3 indicates that the mean squared error of the multiclass ratio estimator goes to zero as the sample size increases. Moreover, it shows that projecting the raw estimator to the simplex improves the convergence, especially for small sample sizes.

We also evaluate the power of the weak prior shift test in Section 2.4. In order to test the weak prior shift, we generate data according to 4 scenarios, which are presented in table 2.5. In all of these scenarios, the weak prior shift assumption holds for a single value of γ. Figure 4 presents the power of the weak prior shift test in each scenario using a level of significance of $\alpha=5\%$. Besides the test achieving the level of 5% when weak prior shift holds, it also has a high power whenever the marginal distribution of $g(\mathbf{X})$ differs over the labeled and over the unlabeled data. The reason why such test presents minima when the weak priori shift assumption does not hold is described in Example 1.

The following section discusses extensions of the ratio estimator to scenarios that are more general than the standard quantification problem.

3. Extensions of the quantification problem

3.1. Combined estimator

Sometimes, a few labels are available in the target population ($S=0$). Let $A_0^*\subset A_0$ denote the indices of these labeled sample instances. In this scenario, it is possible to obtain an estimate of θ that combines the ratio estimator with the additional labels which are available. The labeled estimator of θ is defined as:

$$\widehat{\theta}_L := \frac{1}{|A_0^*|}\sum_{i\in A_0^*} Y_i$$

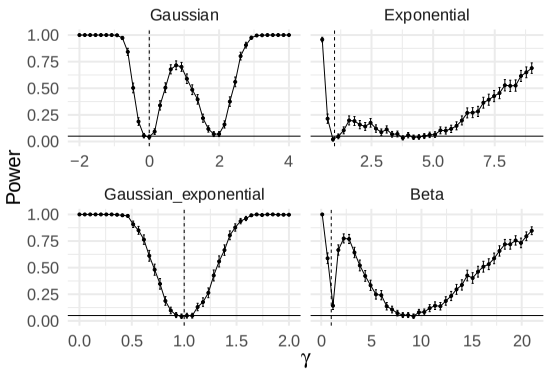


Figure 4: Power of the weak prior shift test at level $\alpha=5\%$. The dashed vertical lines indicate the value of γ for which the weak prior shift holds.

It is possible to better estimate θ by combining the labeled estimator, $\widehat{\theta}_L$, with the ratio estimator, $\widehat{\theta}_R$. We propose to combine these estimators by means of a convex combination,

$$\widehat{\theta}_C = w\widehat{\theta}_R + (1-w)\widehat{\theta}_L, \tag{3}$$

which we name the *combined* estimator. The following theorem provides an insight on how to choose w.

Theorem 15 *Under Assumptions 1, 3, 4 and 5, the value of w that minimizes the mean squared error of the combined estimator (Equation 3) is*

$$w^* = MSE[\widehat{\theta}_L] \times (MSE[\widehat{\theta}_L] + MSE[\widehat{\theta}_R])^{-1}.$$

In practice, $MSE[\widehat{\theta}_L]$ and $MSE[\widehat{\theta}_R]$ need to be estimated. Note that $MSE[\widehat{\theta}_L] = \theta(1-\theta)\times|A_0^*|^{-1}$ and $MSE[\widehat{\theta}_R]$ is given by Theorem 10. We therefore use $\widehat{w}=\widehat{MSE}[\widehat{\theta}_L]\times(\widehat{MSE}[\widehat{\theta}_L]+\widehat{MSE}[\widehat{\theta}_R])^{-1}$, where $\widehat{MSE}[\widehat{\theta}_L]$ and $\widehat{MSE}[\widehat{\theta}_R]$ are obtained by substituting the parameters in $MSE[\widehat{\theta}_L]$ and $MSE[\widehat{\theta}_R]$ by their corresponding empirical averages.

We evaluate the combined estimator under the same scenarios used for the ratio estimator in Section 2.5 and using $\theta=0.3$. For each scenario, we consider 10, 20, 30, 40 or 50 available labels from the target population. Figure 5 presents the errors for each setting scenario and number of available labels in the target population.[3] When one of $\widehat{\theta}_L$ and $\widehat{\theta}_R$ has an error which is much lower than the other, than this lowest error is comparable to that of the combined estimator. Also, when $\widehat{\theta}_L$ and $\widehat{\theta}_R$ have similar errors, then the error of the combined estimator is approximately $\sqrt{2}^{-1}$ times this common error. These results indicate that a few labels from the target population can improve the estimation of θ.

3. Similar plots (with similar conclusions) for $\theta\in\{0.1,0.2,0.4,0.5\}$ can be found in Figures 12—15 in Appendix B.

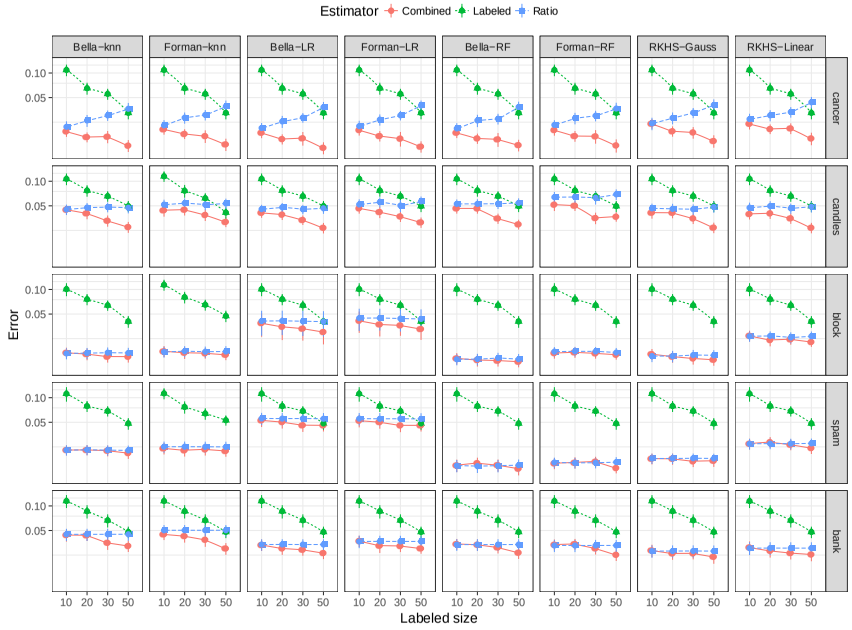


Figure 5: Root mean square deviation in logarithmic scale for each data set, method and estimator by size of the labeled sample and using $\theta = 0.3$

3.2. Regression quantification

As a generalization of the quantification problem, one might be interested on how the prevalence of Y in the target population varies according to a new set of covariates, $\mathbf{Z}$. For example, suppose that a company implements a program of continuous improvement for one of its products. In order to measure the effects of the program, it is necessary to evaluate how the proportion of positive reviews for the product varies over time. This problem fits into the generalization of the quantification problem when taking $\mathbf{Z}$ to be the date at which each review was posted. This problem is often called sentiment analysis (Wang et al., 2012) and is usually solved using a classify and count approach, which suffers from the same downsides as the ones discussed in the standard quantification problem. Other approaches can be found in Hofer and Krempl (2013).

The ratio estimator can be extended to this regression setting. Let the new sample instances be $(\mathbf{X}_1, \mathbf{Z}_1, Y_1, S_1), \ldots, (\mathbf{X}_n, \mathbf{Z}_n, Y_n, S_n)$, where $\mathbf{X}$, Y and S have the same interpretation as in the quantification problem and $\mathbf{Z}$ is the new covariate of interest. The goal

in the new setting is to estimate

$$\theta(\mathbf{z}) := \mathbb{P}(Y = 1 | S = 0, \mathbf{z}),$$

the proportion of positive labels in the target population when $\mathbf{Z} = \mathbf{z}$. In order to estimate $\theta(\mathbf{z})$, it is necessary to make additional assumptions on how $\mathbf{Z}$ relates to the other variables in the quantification problem. One such assumption is presented below.

Assumption 6 *$g(\mathbf{X})$ is stochastically independent of $\mathbf{Z}$ conditionally on Y and S.*

In the scenario in which $\mathbf{X}$ are written reviews of products and Z is time, Assumption 6 states that, if the label of a product is known, then the time at which the label was given does not affect the written review. This assumption motivates the definition of the regression ratio estimator.

Definition 16 *The untrimmed regression ratio estimator, $\widehat{\theta}_{URR}(\mathbf{z})$, is*

$$\widehat{\theta}_{URR}(\mathbf{z}) = \frac{\widehat{\mathbb{E}}[g(\mathbf{X})|S = 0, \mathbf{z}] - \widehat{\mathbb{E}}[g(\mathbf{X})|Y = 0, S = 1]}{\widehat{\mathbb{E}}[g(\mathbf{X})|Y = 1, S = 1] - \widehat{\mathbb{E}}[g(\mathbf{X})|Y = 0, S = 1]},$$

where $\widehat{\mathbb{E}}[g(\mathbf{X})|Y = 0, S = 1]$ and $\widehat{\mathbb{E}}[g(\mathbf{X})|Y = 1, S = 1]$ are the same empirical averages as in Definition 4 and $\widehat{\mathbb{E}}[g(\mathbf{X})|S = 0, \mathbf{z}]$ is an estimate of the regression function $\mathbb{E}[g(\mathbf{X})|S = 0, \mathbf{z}]$. For instance $\widehat{\mathbb{E}}[g(\mathbf{X})|S = 0, \mathbf{z}]$ could be the Nadaraya-Watson regression estimator (Nadaraya, 1964) based on the target population for $g(\mathbf{X})$ given $\mathbf{Z}$. The regression ratio estimator, $\widehat{\theta}_{RR}(\mathbf{z})$, is $\max(0, \min(1, \widehat{\theta}_{URR}(\mathbf{z})))$.

Next, we derive an upper bound on the rate of convergence of the regression ratio estimator.

Theorem 17 *Under Assumptions 1, 3 and 6,*

$$\mathbb{E}\left[\left(\widehat{\theta}_{RR}(\mathbf{Z}) - \theta(\mathbf{Z})\right)^2 \middle| S_1^n\right] \leq \mathcal{O}\left(\max\left(\mathbb{E}\left[\left(\widehat{\mathbb{E}}[g(\mathbf{X})|S = 0, \mathbf{Z}] - \mathbb{E}[g(\mathbf{X})|S = 0, \mathbf{Z}]\right)^2 | S_1^n\right], n_L^{-1}\right)\right)$$

Theorem 17 shows that the integrated mean squared error of $\widehat{\theta}_{RR}$ depends both on n_L and on the integrated mean squared error of $\widehat{\mathbb{E}}[g(\mathbf{X})|S = 0, \mathbf{Z}]$ with respect to the regression function, $\mathbb{E}[g(\mathbf{X})|S = 0, \mathbf{Z}]$. If one uses standard nonparametric methods to estimate $\mathbb{E}[g(\mathbf{X})|S = 0, \mathbf{Z}]$, then it is possible to prove the consistency of $\widehat{\theta}_{RR}$ under weak additional assumptions. For example, if in the target population the pairs $(g(\mathbf{X}_1), \mathbf{Z}_1), \ldots, (g(\mathbf{X}_{n_U}), \mathbf{Z}_{n_U})$ are i.i.d., the regression function $\mathbb{E}[g(\mathbf{X})|S = 0, \mathbf{z}]$ is sufficiently smooth over $\mathbf{z}$ and $\widehat{\mathbb{E}}[g(\mathbf{X})|S = 0, \mathbf{z}]$ is the Nadaraya-Watson kernel estimator with bandwidth $h_{n_U} = O(n_U^{-1/5})$, then it follows (Wasserman, 2006)[p.73] that

$$\mathbb{E}\left[\left(\widehat{\theta}_{RR}(\mathbf{Z}) - \theta(\mathbf{Z})\right)^2 \middle| S_1^n\right] \leq \mathcal{O}\left(\max(n_U^{-4/5}, n_L^{-1})\right)$$

In the equation above, the rate of convergence over n_U is slower than the one obtained in Theorem 7. This is expected, since the rates of convergence of nonparametric estimators are typically slower than those of sample means.

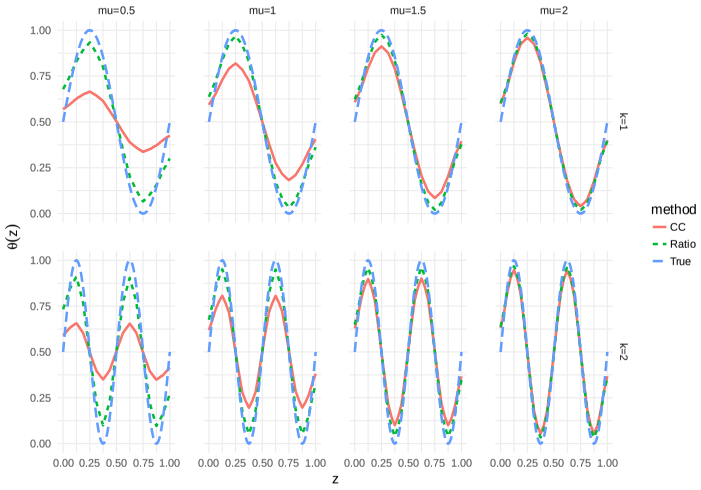


Figure 6: Average of the fitted regression in each setting.

Besides showing the consistency of the regression ratio estimator, we also show that it outperforms the classify and count method in artificial data sets. Specifically, we run the regression ratio estimator using the Nadaraya-Watson kernel estimator and generate data sets under the following specifications:

- $Z \sim U(0,1)$
- $\mathbb{P}(Y=1|S=1,Z=z)=0.5$
- $\mathbb{P}(Y=1|S=0,Z=z)=0.5(\sin(2zk\pi)+1)$, for $k \in \{1,2\}$
- $X|Y=0 \sim N(\mu,1)$ and $X|Y=1 \sim N(-\mu,1)$, for $\mu \in \{0.5,1,1.5,2\}$
- $n_L = n_U = 10^3$
- $g(X)$ is the Bayes classifier, i.e. $g(X)=\mathbb{I}(X>0)$

For each combination of k and μ, 400 independent data sets were generated. Figure 6 presents the average curve fitted for $\theta(\mathbf{z})$ using the regression ratio and classify and count estimators. One can observe that, while for small values of μ the ratio regression outperforms the classify and count estimator, for large values of μ both estimators are similar. This occurs because when μ is large, the classification problem of determining the value of Y is easier and both methods perform well. One can also observe from Figure 6 that the classify and count method performs worse than the regression ratio estimator

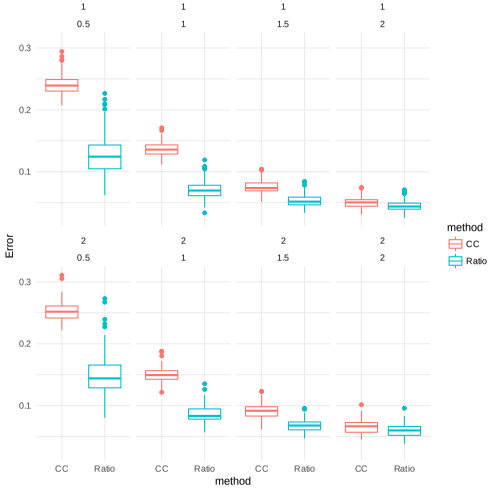


Figure 7: Boxplots of the root mean square deviation in each setting.

because its fit is generally smoother than the true regression curve. Figure 7 summarizes the mean squared error of each method in each setting. The regression ratio estimator leads to substantial improvements over the classify and count method.

4. Final remarks

We present the ratio estimator for the problem of quantification, show that it is approximately minimax under the prior probability shift assumption, and provide an hypothesis test for this assumption. Since the methods in Forman (2006) and Bella et al. (2010) are particular instances of the ratio estimator, it follows that they are also approximately minimax. The lower bound on the risk that we derive is of independent interest and can be used to investigate the optimality of other quantification methods. We also derive the limiting distribution of the ratio estimator, which allows the derivation of a ratio estimator based on Reproducing Kernel Hilbert Spaces and of confidence intervals for the quantification problem. A simulation study shows that the ratio estimator based on Reproducing Kernel Hilbert spaces is a competitive new alternative.

Besides the above results, we also generalize the ratio estimator to two other scenarios. In the first one, we consider the case in which some labels are available in the target population. The combined estimator uses these labels and the ratio estimator to obtain a larger effective sample size than the ratio estimator. In the second scenario, we consider the prevalence of positive labels varies according to a new variable, $\mathbf{Z}$. We show that, under Assumption 6, the regression ratio estimator can be made consistent for $\theta(\mathbf{Z})$. A still

unresolved issue is how much it is possible to relax Assumption 6 while still being able to learn $\theta(\mathbf{Z})$.

Acknowledgments

This work was partially supported by FAPESP grant 2017/03363-8 and CNPq grant 306943/2017-4. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Appendix A. Proofs

Lemma 18 *Let $\mathbb{X}_U = (\mathbf{X}_u)_{u\in A_0}$, $\mathbf{Y}_U = (Y_u)_{u\in A_0}$, $\mathbb{X}_L = (\mathbf{X}_l)_{l\in A_1}$, and $\mathbf{Y}_L = (Y_l)_{l\in A_1}$. Under Assumptions 1 and 2, if $\theta \sim$ Uniform$(0,1)$, then $\mathbb{E}[\mathbb{V}[\theta|\mathbb{X}_U, \mathbb{X}_L, \mathbf{Y}_L]] \geq \Omega(n_U^{-1})$. Under the same assumptions, if $\mathbb{P}(\theta = 0.5 - n_L^{-1}) = \mathbb{P}(\theta = 0.5 + n_L^{-1}) = 0.5$, $f_0 =$ Bernoulli(0), $\alpha \sim$ Uniform$(\epsilon^*, 1)$ $f_1|\alpha =$ Bernoulli(α), then $\mathbb{E}[\mathbb{V}[\theta|\mathbb{X}_U, \mathbb{X}_L, \mathbf{Y}_L]] \geq \Omega(n_L^{-1})$.*

Proof It follows from Assumptions 1 and 2 that the dependency relations between data and parameters can be represented by figure 8.

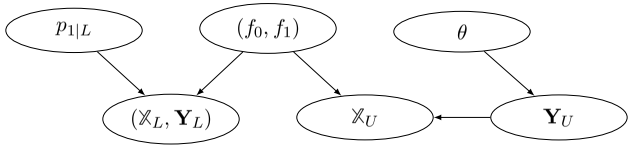


Figure 8: Dependency relations between data and parameters in the prior shift model.

If $\theta \sim U(0,1)$, then

$$\begin{aligned}\mathbb{E}[\mathbb{V}[\theta|\mathbb{X}_U, \mathbb{X}_L, \mathbf{Y}_L]|S_1^n] &\geq \mathbb{E}[\mathbb{V}[\theta|\mathbb{X}_U, \mathbb{X}_L, \mathbf{Y}_L, \mathbf{Y}_U]|S_1^n] \\ &= \mathbb{E}[\mathbb{V}[\theta|\mathbf{Y}_U]|S_1^n] && \text{fig. 8} \\ &= \Omega(n_U^{-1}) && \mathbf{Y}_U|\theta \text{ i.i.d. Bernoulli}(\theta) \end{aligned} \quad (4)$$

Next, let $\mathbb{P}(\theta = 0.5 - n_L^{-0.5}) = \mathbb{P}(\theta = 0.5 + n_L^{-0.5}) = 0.5$, $f_0 =$ Bernoulli(0) $\alpha \sim U(\epsilon, 1)$ and $f_1|\alpha =$ Bernoulli(α). Define $\mathbb{X}_{L,1} = (\mathbf{X}_l)_{l\in A_{1,1}}$ and note that

$$f(\alpha|\mathbb{X}_L, \mathbf{Y}_L) \propto \alpha^{n_{L,1}\bar{\mathbb{X}}_{L,1}}(1-\alpha)^{n_{L,1}(1-\bar{\mathbb{X}}_{L,1})}\mathbb{I}(\alpha \in (\epsilon, 1)) \quad (5)$$

Let $\lambda := \theta\alpha$ and observe that, for every $\lambda \in \left(\epsilon(0.5 + n_L^{-0.5}), 0.5 - n_L^{-0.5}\right]$,

$$\begin{aligned}\mathbb{P}(\theta = 0.5 + n_L^{-0.5}|\lambda, \mathbb{X}_L, \mathbf{Y}_L) &= \frac{f_\alpha(\lambda(0.5 + n_L^{-0.5})^{-1}|\mathbb{X}_L, \mathbf{Y}_L)}{f_\alpha(\lambda(0.5 + n_L^{-0.5})^{-1}|\mathbb{X}_L, \mathbf{Y}_L) + f_\alpha(\lambda(0.5 - n_L^{-0.5})^{-1}|\mathbb{X}_L, \mathbf{Y}_L)} \\ &= \frac{1}{1 + \frac{\left(\frac{\lambda}{0.5-n_L^{-0.5}}\right)^{n_{L,1}\bar{\mathbb{X}}_{L,1}}\left(1-\frac{\lambda}{0.5-n_L^{-0.5}}\right)^{n_{L,1}(1-\bar{\mathbb{X}}_{L,1})}}{\left(\frac{\lambda}{0.5+n_L^{-0.5}}\right)^{n_{L,1}\bar{\mathbb{X}}_{L,1}}\left(1-\frac{\lambda}{0.5+n_L^{-0.5}}\right)^{n_{L,1}(1-\bar{\mathbb{X}}_{L,1})}}} && eq.5 \\ &= \frac{1}{1 + \left(1 + \frac{4}{n_L^{0.5}-2}\right)^{n_{L,1}}\left(1 - \frac{4}{(1-2\theta\alpha)n_L^{0.5}+2}\right)^{n_{L,1}(1-\bar{\mathbb{X}}_{L,1})}} && \lambda = \theta\alpha \end{aligned} \quad (6)$$

Let $\gamma_{n_L} := \mathbb{P}(\theta = 0.5 + n_L^{-1}|\lambda, \mathbb{X}_L, \mathbf{Y}_L)$. Note that $\frac{n_{L,1}}{n_L} \xrightarrow{a.s.} p_{1|L}$ and $\bar{\mathbb{X}}_{L,1} \xrightarrow{a.s.} \alpha$. Therefore, since $|\theta - 0.5| \leq n_L^{-0.5}$, it follows from eq. 6 that γ_{n_L} converges a.s. to a quantity between $\frac{1}{1+\exp\left(\frac{-8\alpha p_{1|L}}{1-\alpha}\right)}$ and $\frac{1}{1+\exp\left(\frac{8\alpha p_{1|L}}{1-\alpha}\right)}$. That is,

$$\mathbb{E}[\gamma_{n_L}(1 - \gamma_{n_L})|S_1^n] \geq \Omega(1). \quad (7)$$

Note that $\bar{\mathbb{X}}_U$ is sufficient for (θ, α) and $\bar{\mathbb{X}}_U$ converges a.s. to λ. Therefore,

$$\begin{aligned}\mathbb{E}[\mathbb{V}[\theta|\mathbb{X}_U, \mathbb{X}_L, \mathbf{Y}_L]|S_1^n] &= \mathbb{E}[\mathbb{V}[\theta|\bar{\mathbb{X}}_U, \mathbb{X}_L, \mathbf{Y}_L]|S_1^n] \\ &\geq \mathbb{E}[\mathbb{V}[\theta|\lambda, \mathbb{X}_L, \mathbf{Y}_L]|S_1^n] \\ &\geq 4n_L^{-1}\mathbb{E}\left[\gamma_{n_L}(1 - \gamma_{n,L})\mathbb{I}\left(\lambda \in \left(\epsilon(0.5 + n_L^{-1}), 0.5 - n_L^{-1}\right]\right)|S_1^n\right] \end{aligned} \quad (8)$$

Since $\mathbb{P}(\lambda \in \epsilon(0.5 + n_L^{-1}), 0.5 - n_L^{-1}) \geq 0.5$, it follows from eqs. 7 and 8 that

$$\mathbb{E}[\mathbb{V}[\theta|\mathbb{X}_U, \mathbb{X}_L, \mathbf{Y}_L]|S_1^n] \geq \Omega(n_L^{-1}) \quad (9)$$

The proof follows from combining eqs. 4 and 9. ■

Proof [Theorem 3] We wish to find a lower bound for the minimax risk given a constraint, $\mathcal{F}$. In order to do so, we use the result that the minimax risk is lower bounded by the Bayes risk of any Bayes estimator associated to a prior with support in $\mathcal{F}$ (Wasserman, 2006; Esteves et al., 2017). Since we consider the squared error loss, the Bayes risk of the Bayes estimator is $\mathbb{E}[\mathbb{V}[\theta|\mathbb{X}_U, \mathbb{X}_L, \mathbf{Y}_L]]$. Hence, if there exists two priors with support in $\mathcal{F}$ such that the first one satisfies $\mathbb{E}[\mathbb{V}[\theta|\mathbb{X}_U, \mathbb{X}_L, \mathbf{Y}_L]] \geq \Omega(n_L^{-1})$ and the second one satisfies $\mathbb{E}[\mathbb{V}[\theta|\mathbb{X}_U, \mathbb{X}_L, \mathbf{Y}_L]] \geq \Omega(n_U^{-1})$, then we can conclude that the minimax risk is lower bounded by $\Omega(\max(n_L^{-1}, n_U^{-1}))$. Lemma 18 can be used to determine these priors for the classes $\mathcal{F}_{\mathcal{L}_1,\epsilon}$ and $\mathcal{F}_{g,k,\epsilon}$. The proof for $\mathcal{F}_{\mathcal{L}_1,\epsilon}$ follows from taking $\epsilon^* = \epsilon$ in Lemma 18. Next, if $\mathcal{F}_{g,k,\epsilon} \neq \emptyset$, then there exist a and b such that $\frac{\epsilon}{|g(a)-g(b)|} < 1$. Without loss of generality, let $a = 0$ and $b = 1$. The proof follows from taking $\epsilon^* = \frac{\epsilon}{|g(1)-g(0)|}$ in Lemma 18. ■

Lemma 19 *For every function, g, under Assumption 3,*

$$\theta := \mathbb{P}(Y=1|S=0) = \frac{\mathbb{E}[g(\mathbf{X})|S=0] - \mathbb{E}[g(\mathbf{X})|Y=0,S=1]}{\mathbb{E}[g(\mathbf{X})|Y=1,S=1] - \mathbb{E}[g(\mathbf{X})|Y=0,S=1]}$$

Proof Let $f(\mathbf{x})$ denote the density of $\mathbf{X}$. Note that

$$\begin{aligned}
g(\mathbf{x})f(\mathbf{x}|S=0) &= \sum_{j=0}^{1} g(\mathbf{x})f(\mathbf{x}|Y=j,S=0)\mathbb{P}(Y=j|S=0) && \text{Law of total prob.}\\
\mathbb{E}[g(\mathbf{X})|S=0] &= \sum_{j=0}^{1} \mathbb{E}[g(\mathbf{X})|Y=j,S=0]\mathbb{P}(Y=j|S=0) && \text{Integration over } \mathbf{x}\\
&= \sum_{j=0}^{1} \mathbb{E}[g(\mathbf{X})|Y=j,S=1]\mathbb{P}(Y=j|S=0) && \text{Assumption 3} \quad (10)
\end{aligned}$$

Isolating $\mathbb{P}(Y=1|S=0)$ in equation 10 yields

$$\theta := \mathbb{P}(Y=1|S=0) = \frac{\mathbb{E}[g(\mathbf{X})|S=0] - \mathbb{E}[g(\mathbf{X})|Y=0,S=1]}{\mathbb{E}[g(\mathbf{X})|Y=1,S=1] - \mathbb{E}[g(\mathbf{X})|Y=0,S=1]}$$

■

Proof [Theorem 6] Follows directly from the definition of $\widehat{\theta}_{UR}$ and $\widehat{\theta}_{R}$, and Lemma 19. ■

Lemma 20 *Let Z_1 and Z_2 be random variables such that $\mathbb{E}[Z_2] \neq 0$ and $\frac{\mathbb{E}[Z_1]}{\mathbb{E}[Z_2]} \in [0,1]$. Define $T = \max\left(0, \min\left(1, \frac{Z_1}{Z_2}\right)\right)$. For every random variable, S, and $\epsilon_1, \epsilon_2 \in (0,1)$.*

$$\mathbb{E}\left[\left(T - \frac{\mathbb{E}[Z_1|S]}{\mathbb{E}[Z_2|S]}\right)^2 \middle| S\right] \leq \frac{4(|\mathbb{E}[Z_1|S]| + \epsilon_1)\max\left(\mathbb{V}[Z_1|S], \mathbb{V}[Z_2|S]\right)}{\min(1, (1-\epsilon_2)^4\mathbb{E}[Z_2|S]^4)} + \epsilon_1^{-2}\mathbb{V}[Z_1|S] + (\epsilon_2\mathbb{E}[Z_2|S])^{-2}\mathbb{V}[Z_2|S]$$

Proof It follows from Taylor's expansion of $\frac{Z_1}{Z_2}$ that there exists $Z_{1,*}$ bounded between $\mathbb{E}[Z_1|S]$ and Z_1, and $Z_{2,*}$ between $\mathbb{E}[Z_2|S]$ and Z_2 such that

$$\frac{Z_1}{Z_2} = \frac{\mathbb{E}[Z_1|S]}{\mathbb{E}[Z_2|S]} + \frac{1}{Z_{2,*}}(Z_1 - \mathbb{E}[Z_1|S]) - \frac{Z_{1,*}}{Z_{2,*}^2}(Z_2 - \mathbb{E}[Z_2|S])$$

Therefore, by letting $A = \{|Z_1 - \mathbb{E}[Z_1|S]| \leq \epsilon_1, |Z_2 - \mathbb{E}[Z_2|S]| \leq \epsilon_2\mathbb{E}[Z_2|S]\}$, obtain

$$\begin{aligned}
&\mathbb{E}\left[\left(\frac{Z_1}{Z_2} - \frac{\mathbb{E}[Z_1|S]}{\mathbb{E}[Z_2|S]}\right)^2 \middle| A, S\right]\mathbb{P}(A|S)\\
=&\mathbb{E}\left[\left(\frac{1}{Z_{2,*}}(Z_1 - \mathbb{E}[Z_1|S]) - \frac{Z_{1,*}}{Z_{2,*}^2}(Z_2 - \mathbb{E}[Z_2|S])\right)^2 \middle| A, S\right]\mathbb{P}(A|S)\\
\leq& 4\max\left(\mathbb{E}\left[\frac{1}{Z_{2,*}^2}(Z_1 - \mathbb{E}[Z_1|S])^2 \middle| A, S\right], \mathbb{E}\left[\frac{Z_{1,*}^2}{Z_{2,*}^4}(Z_2 - \mathbb{E}[Z_2|S])^2 \middle| A, S\right]\right)\mathbb{P}(A|S)\\
\leq& \frac{4(|\mathbb{E}[Z_1|S]| + \epsilon_1)\max\left(E\left[(Z_1 - \mathbb{E}[Z_1|S])^2 \middle| A, S\right], E\left[(Z_2 - \mathbb{E}[Z_2|S])^2 \middle| A, S\right]\right)\mathbb{P}(A|S)}{\min(1, (1-\epsilon_2)^4\mathbb{E}[Z_2|S]^4)}\\
\leq& \frac{4(|\mathbb{E}[Z_1|S]| + \epsilon_1)\max\left(\mathbb{V}[Z_1|S], \mathbb{V}[Z_2|S]\right)}{\min(1, (1-\epsilon_2)^4\mathbb{E}[Z_2|S]^4)} \qquad (11)
\end{aligned}$$

Finally, obtain that

$$\begin{aligned}
E\left[\left(T - \frac{\mathbb{E}[Z_1|S]}{\mathbb{E}[Z_2|S]}\right)^2 \middle| S\right] &= \mathbb{E}\left[\mathbb{E}\left[\left(T - \frac{\mathbb{E}[Z_1|S]}{\mathbb{E}[Z_2|S]}\right)^2 \middle| \mathbb{I}_A, S\right] \middle| S\right]\\
&\leq \mathbb{E}\left[\left(\frac{Z_1}{Z_2} - \frac{\mathbb{E}[Z_1|S]}{\mathbb{E}[Z_2|S]}\right)^2 \middle| A, S\right]\mathbb{P}(A|S) + \mathbb{P}(A^c|S) && T, \frac{\mathbb{E}[Z_1]}{\mathbb{E}[Z_2]} \in [0,1]\\
&\leq \frac{4(|\mathbb{E}[Z_1|S]| + \epsilon_1)\max\left(\mathbb{V}[Z_1|S], \mathbb{V}[Z_2|S]\right)}{\min(1, (1-\epsilon_2)^4\mathbb{E}[Z_2|S]^4)} + \mathbb{P}(A^c|S) && \text{eq. 11}
\end{aligned}$$

The result follows from applying the union bound and Chebyshev's inequality to obtain

$$\begin{aligned}
\mathbb{P}(A^c|S) &\leq \mathbb{P}(|Z_1 - \mathbb{E}[Z_1|S]| > \epsilon_1|S) + \mathbb{P}(|Z_2 - \mathbb{E}[Z_2|S]| > \epsilon_2\mathbb{E}[Z_2|S]|S)\\
&\leq \epsilon_1^{-2}\mathbb{V}[Z_1|S] + (\epsilon_2\mathbb{E}[Z_2|S])^{-2}\mathbb{V}[Z_2|S]
\end{aligned}$$

■

Proof [Theorem 7] Define $Z_1 = \frac{\sum_{i\in A_0} g(\mathbf{X}_i)}{|A_0|} - \frac{\sum_{i\in A_{1,0}} g(\mathbf{X}_i)}{|A_{1,0}|}$ and also $Z_2 = \frac{\sum_{i\in A_{1,1}} g(\mathbf{X}_i)}{|A_{1,1}|} - \frac{\sum_{i\in A_{1,0}} g(\mathbf{X}_i)}{|A_{1,0}|}$. Note that

$$\begin{aligned}
\mathbb{E}\left[\frac{\sum_{i\in A_0} g(\mathbf{X}_i)}{|A_0|}\middle|\mathbf{S}_1^n\right] &= \mathbb{E}[g(\mathbf{X})|S=0]\\
\mathbb{E}\left[\frac{\sum_{i\in A_{1,j}} g(\mathbf{X}_i)}{|A_{1,j}|}\middle|\mathbf{S}_1^n\right] &= \mathbb{E}[g(\mathbf{X})|Y=j,S=1]
\end{aligned}$$

It follows from Lemma 19 that $\theta = \frac{\mathbb{E}[Z_1|\mathbf{S}_1^n]}{\mathbb{E}[Z_2|\mathbf{S}_1^n]}$. With $T := \widehat{\theta}_R = \max\left(0, \min\left(1, \frac{Z_1}{Z_2}\right)\right)$, obtain

$$\begin{aligned}
\mathbb{E}\left[\left(\widehat{\theta}_R - \theta\right)^2 \middle| \mathbf{S}_1^n\right] &= \mathbb{E}\left[\left(T - \frac{\mathbb{E}[Z_1|\mathbf{S}_1^n]}{\mathbb{E}[Z_2|\mathbf{S}_1^n]}\right)^2 \middle| \mathbf{S}_1^n\right] \\
&\leq \frac{4(|\mathbb{E}[Z_1|\mathbf{S}_1^n]| + \epsilon_1)\max\left(\mathbb{V}[Z_1|\mathbf{S}_1^n], \mathbb{V}[Z_2|\mathbf{S}_1^n]\right)}{\min(1, (1-\epsilon_2)^4\mathbb{E}[Z_2|\mathbf{S}_1^n]^4)} \\
&+ \epsilon_1^{-2}\mathbb{V}[Z_1|\mathbf{S}_1^n] + (\epsilon_2\mathbb{E}[Z_2|\mathbf{S}_1^n])^{-2}\mathbb{V}[Z_2|\mathbf{S}_1^n] \qquad \text{Lemma 20}
\end{aligned}$$

The result follows from observing that, under $\mathcal{F}_{g,K,\epsilon}$, $\mathbb{E}[Z_1|\mathbf{S}_1^n]$ and $\mathbb{E}[Z_2|\mathbf{S}_1^n]$ are bounded by constants, $\mathbb{V}[Z_1|\mathbf{S}_1^n] = \mathcal{O}(\max(n_L^{-1}, n_U^{-1}))$, and $\mathbb{V}[Z_2|\mathbf{S}_1^n] = \mathcal{O}(n_L^{-1})$. ■

Proof [Theorem 10] The proof strategy is divided into two parts. The first part consists of proving a joint central limit theorem for the three sample averages that appear in the untrimmed ratio estimator. The second part uses this central limit theorem and the delta method to complete the proof for each case that is considered in the theorem.

The main challenge appears when proving the central limit theorem for the sample averages that appear in the ratio estimator. This occurs since these averages are not marginally independent. However, they are independent conditional on the values of $\mathbf{Y}$ and $\mathbf{S}$. This conditional independence can be used to calculate the limiting behavior of the characteristic function of the standardized averages, which completes this part of the proof.

We tidy the proof by using the following notation: $\mu_U := \mathbb{E}[g(\mathbf{X}_i)|S_i = 0]$, $\sigma_U^2 := \mathbb{V}[g(\mathbf{X}_i)|S_i = 0]$, $Z_{U,n} := \frac{\sqrt{n_U}}{\sigma_U}\left(\frac{\sum_{i=1}^n g(\mathbf{X}_i)\mathbb{I}(S_i=0)}{n_U} - \mu_U\right)$, $Z_{j,n} := \frac{\sqrt{n_j}}{\sigma_j}\left(\frac{\sum_{i=1}^n g(\mathbf{X}_i)\mathbb{I}(S_i=0,Y_i=j)}{n_j} - \mu_j\right)$, $F_i = \mathbb{I}(S_i = 1)(Y_i + 1)$, $A_U = \{F_1 = 0\}$, $A_0 = \{F_1 = 1\}$, and $A_1 = \{F_1 = 2\}$. Note that

$$\begin{aligned}
\lim_{n\to\infty} \phi_{Z_{U,n},Z_{0,n},Z_{1,n}}(t_U, t_0, t_1) &= \lim_{n\to\infty} \mathbb{E}\left[\mathbb{E}\left[\exp\left(\sum_{j\in\{U,0,1\}} it_j Z_{j,n}\right)\middle| F_1, \ldots, F_n\right]\right] \\
&= \lim_{n\to\infty} \mathbb{E}\left[\prod_{j\in\{U,0,1\}} \mathbb{E}\left[\exp\left(it_j Z_{j,n}\right)\middle| F_1, \ldots, F_n\right]\right] \\
&= \lim_{n\to\infty} \mathbb{E}\left[\prod_{j\in\{U,0,1\}} \left(\phi_{\frac{g(\mathbf{X}_1)-\mu_j}{\sigma_j}|A_j}(t_j n_j^{-0.5})\right)^{n_j}\right] \qquad (12)
\end{aligned}$$

It follows from the Central Limit Theorem for i.i.d. random variables that, for every $j \in \{U, 0, 1\}$, $\phi^{n_j}_{\frac{g(\mathbf{X}_1)-\mu_j}{\sigma_j}|A_j}(t_j n_j^{-0.5}) \to \exp(-0.5t_j^2)$ as $n_j \to \infty$. Since $n_j \xrightarrow{a.s.} \infty$, conclude from eq. 12 and the dominated convergence theorem that

$$\lim_{n\to\infty} \phi_{Z_{U,n},Z_{0,n},Z_{1,n}}(t_U, t_0, t_1) = \prod_{j\in\{U,0,1\}} \exp(-0.5t_j^2)$$

and, using $\mathbb{1}$ as the identity matrix, obtain

$$(Z_{U,n}, Z_{0,n}, Z_{1,n}) \rightsquigarrow N(0, \mathbb{1}) \qquad (13)$$

Assume that $p_L \neq 0$. In this case, since $\frac{n_L}{n} \xrightarrow{\mathbb{P}} p_L$, it follows from eq. 13 that

$$\sqrt{n}\left(\frac{\sum_{i=1}^n g(\mathbf{X}_i)\mathbb{I}(S_i=0)}{n_U} - \mu_U, \frac{\sum_{i=1}^n g(\mathbf{X}_i)\mathbb{I}(S_i=0,Y_i=0)}{n_0} - \mu_0, \frac{\sum_{i=1}^n g(\mathbf{X}_i)\mathbb{I}(S_i=0,Y_i=1)}{n_1} - \mu_1\right)$$

converges in distribution to $N\left(0, diag\left(\frac{\sigma_U^2}{1-p_L}, \frac{\sigma_0^2}{p_L p_{0|L}}, \frac{\sigma_1^2}{p_L p_{1|L}}\right)\right)$. Since $\theta = \frac{\mu_U - \mu_0}{\mu_1 - \mu_0}$ (Lemma 19) and $\widehat{\theta}_{UR} = \frac{\frac{\sum_{i=1}^n g(\mathbf{X}_i)\mathbb{I}(S_i=0)}{n_U} - \frac{\sum_{i=1}^n g(\mathbf{X}_i)\mathbb{I}(S_i=0,Y_i=0)}{n_0}}{\frac{\sum_{i=1}^n g(\mathbf{X}_i)\mathbb{I}(S_i=0,Y_i=1)}{n_1} - \frac{\sum_{i=1}^n g(\mathbf{X}_i)\mathbb{I}(S_i=0,Y_i=0)}{n_0}}$, it follows from the delta method (Casella and Berger, 2002) that

$$\sqrt{n}(\widehat{\theta}_{UR} - \theta) \rightsquigarrow N\left(0, \frac{\sigma_U^2(1-p_L)^{-1}}{(\mu_1-\mu_0)^2} + \frac{(\mu_U-\mu_1)^2\sigma_0^2(p_L p_{0|L})^{-1}}{(\mu_1-\mu_0)^4} + \frac{(\mu_U-\mu_0)^2\sigma_1^2(p_L p_{1|L})^{-1}}{(\mu_1-\mu_0)^4}\right)$$

Since $\mu_U = (1-\theta)\mu_0 + \theta\mu_1$ and $\sigma_U^2 = (1-\theta)\sigma_0^2 + \theta\sigma_1^2 + (\mu_1-\mu_0)^2\theta(1-\theta)$ obtain that

$$\sqrt{n}(\widehat{\theta}_{UR} - \theta) \rightsquigarrow N\left(0, \frac{\frac{(1-\theta)\sigma_0^2 + \theta\sigma_1^2 + (\mu_1-\mu_0)^2\theta(1-\theta)}{1-p_L} + \frac{(1-\theta)^2\sigma_0^2}{p_L p_{0|L}} + \frac{\theta^2\sigma_1^2}{p_L p_{1|L}}}{(\mu_1-\mu_0)^2}\right)$$

Next, assume that $p_L = 0$. Obtain that $\sqrt{h(n)}(Z_{U,n} - \mu_U) \xrightarrow{\mathbb{P}} 0$ and

$$\sqrt{h(n)}\left(\frac{\sqrt{p_{0|L}}}{\sigma_0}(Z_{0,n} - \mu_0), \frac{\sqrt{p_{1|L}}}{\sigma_1}(Z_{1,n} - \mu_1)\right) \rightsquigarrow N(0, \mathbb{1})$$

It follows from the delta method and Slutsky's theorem that

$$\sqrt{h(n)}(\widehat{\theta}_{UR} - \theta) \rightsquigarrow N\left(0, \frac{\frac{(1-\theta)^2\sigma_0^2}{p_{0|L}} + \frac{\theta^2\sigma_1^2}{p_{1|L}}}{(\mu_1-\mu_0)^2}\right)$$

The same convergence results hold for $\widehat{\theta}_R$ since the derivative of the trimming function is 1 around θ. ■

Proof [Theorem 13] It follows from the Representer Theorem (Wahba, 1990) that, for every $g \in \mathcal{H}_K$, $g(\mathbf{x}) = \sum_{k\in A_1} w_k K(\mathbf{x}, \mathbf{x}_k)$. Using this fact, for every $i \in \{0, 1\}$,

$$\begin{aligned}
\widehat{\mu}_i &= \frac{\sum_{j\in A_{1,i}} g(\mathbf{x}_j)}{n_i} = \frac{\sum_{k\in A_1} w_k \sum_{j\in A_{1,i}} K(\mathbf{x}_j, \mathbf{x}_k)}{n_i} = \mathbf{w}^t m_i \\
\widehat{\sigma}_i^2 &= \frac{\sum_{j\in A_{1,i}} (g(\mathbf{x}_j) - \widehat{\mu}_i)^2}{n_i} = \mathbf{w}^t \widehat{\Sigma}_i \mathbf{w}
\end{aligned}$$

Therefore, for every $g \in \mathcal{H}_K$,

$$\widehat{\mathrm{MSE}}(g) = \frac{\mathbf{w}^t N \mathbf{w}}{\mathbf{w}^t M \mathbf{w}}.$$

Proof [Proposition 14]

$$\begin{aligned} F_{g(\mathbf{X})|S=0} &= \theta F_{g(\mathbf{X})|S=0,Y=1} + (1-\theta)F_{g(\mathbf{X})|S=0,Y=0} \\ &= \theta F_{g(\mathbf{X})|S=1,Y=1} + (1-\theta)F_{g(\mathbf{X})|S=1,Y=0} \qquad 3 \end{aligned}$$

Thus, there exists $0 \leq p \leq 1$ such that $F_{g(\mathbf{X})|S=0} = pF_{g(\mathbf{X})|S=1,Y=1} + (1-p)F_{g(\mathbf{X})|S=1,Y=0}$. ■

Proof [Theorem 15]

$$\begin{aligned} MSE[\widehat{\theta}_C] &= \mathbb{E}\left[\left((w\widehat{\theta}_R + (1-w)\widehat{\theta}_L) - \theta\right)^2\right] = \mathbb{E}\left[\left((w(\widehat{\theta}_R - \theta) + (1-w)(\widehat{\theta}_L - \theta)\right)^2\right] \\ &= w^2 MSE[\widehat{\theta}_R] + (1-w)^2 MSE[\widehat{\theta}_L] + 2w(1-w)\mathbb{E}[(\widehat{\theta}_R - \theta)(\widehat{\theta}_L - \theta)] \qquad \widehat{\theta}_R \text{ indep. } \widehat{\theta}_L, \mathbb{E}[\widehat{\theta}_L] = \theta \\ &= w^2 MSE[\widehat{\theta}_R] + (1-w)^2 MSE[\widehat{\theta}_L] \end{aligned}$$

It follows that $MSE[\widehat{\theta}_C]$ is minimized by taking $w = MSE[\widehat{\theta}_L] \times (MSE[\widehat{\theta}_L] + MSE[\widehat{\theta}_R])^{-1}$. ■

Lemma 21 *For every function, g, under Assumptions 3 and 6*

$$\theta(\mathbf{z}) = \frac{\mathbb{E}[g(\mathbf{X})|S=0,\mathbf{z}] - \mathbb{E}[g(\mathbf{X})|Y=0,S=1]}{\mathbb{E}[g(\mathbf{X})|Y=1,S=1] - \mathbb{E}[g(\mathbf{X})|Y=0,S=1]}$$

Proof For every $\mathbf{z} \in \mathbb{R}^{d_z}$, Let $f(\mathbf{x})$ denote the density of $\mathbf{X}$. Note that

$$\begin{aligned} g(\mathbf{x})f(\mathbf{x}|S=0,\mathbf{z}) &= \sum_{j=0}^{1} g(\mathbf{x})f(\mathbf{x}|Y=j,S=0,\mathbf{z})\mathbb{P}(Y=j|S=0,\mathbf{z}) \quad \text{Law of total prob.} \\ \mathbb{E}[g(\mathbf{X})|S=0,\mathbf{z}] &= \sum_{j=0}^{1} \mathbb{E}[g(\mathbf{X})|Y=j,S=0,\mathbf{z}]\mathbb{P}(Y=j|S=0,\mathbf{z}) \quad \text{Integration over } \mathbf{x} \\ &= \sum_{j=0}^{1} \mathbb{E}[g(\mathbf{X})|Y=j,S=0]\mathbb{P}(Y=j|S=0,\mathbf{z}) \quad \text{Assumption 6} \\ &= \sum_{j=0}^{1} \mathbb{E}[g(\mathbf{X})|Y=j,S=1]\mathbb{P}(Y=j|S=0,\mathbf{z}) \quad \text{Assumption 3} \end{aligned} \tag{14}$$

Isolating $\mathbb{P}(Y=1|S=0,\mathbf{z})$ in equation 14 yields

$$\theta(\mathbf{z}) := \mathbb{P}(Y=1|S=0,\mathbf{z}) = \frac{\mathbb{E}[g(\mathbf{X})|S=0,\mathbf{z}] - \mathbb{E}[g(\mathbf{X})|Y=0,S=1]}{\mathbb{E}[g(\mathbf{X})|Y=1,S=1] - \mathbb{E}[g(\mathbf{X})|Y=0,S=1]}$$

Proof [Theorem 17] The main difference between this proof and the one of Theorem 7 is that $\widehat{\mathbb{E}}[g(\mathbf{X})|S=0,\mathbf{z}]$ is usually biased for $\mathbb{E}[g(\mathbf{X})|S=0,\mathbf{z}]$. The proof strategy consists of isolating this bias term from the squared error and then replicating steps which are similar to the ones in Theorem 7.

In order to present the proof in a compact form, some special notation is used. Specifically, $h_0(\mathbf{z}) = \mathbb{E}[g(\mathbf{X})|S=0,\mathbf{z}]$, $\hat{h}_0(\mathbf{z}) = \widehat{\mathbb{E}}[g(\mathbf{X})|S=0,\mathbf{z}]$, $h_{1,i} = \mathbb{E}[g(\mathbf{X})|S=1,Y=i]$, and $\hat{h}_{1,i} = \widehat{\mathbb{E}}[g(\mathbf{X})|S=1,Y=i]$. Using this notation and Definition 16, note that $\hat{\theta}_{RR}(\mathbf{Z}) = \max\left(0, \min\left(1, \frac{\hat{h}_0(\mathbf{z}) - \hat{h}_{1,0}}{\hat{h}_{1,1} - \hat{h}_{1,0}}\right)\right)$. Therefore,

$$\begin{aligned} &\mathbb{E}\left[\left(\widehat{\theta}_{RR}(\mathbf{Z}) - \theta(\mathbf{Z})\right)^2 \middle| S_1^n\right] \\ =&\mathbb{E}\left[\left(\widehat{\theta}_{RR}(\mathbf{Z}) - \frac{h_0(\mathbf{Z}) - h_{1,0}}{h_{1,1} - h_{1,0}}\right)^2 \middle| S_1^n\right] \quad \text{Lemma 21} \\ \leq&\mathcal{O}\left(\mathbb{E}\left[\left(\widehat{\theta}_{RR}(\mathbf{Z}) - \frac{\mathbb{E}[\hat{h}_0(\mathbf{Z})|S_1^n] - h_{1,0}}{h_{1,1} - h_{1,0}}\right)^2 + \left(\frac{\mathbb{E}[\hat{h}_0(\mathbf{Z})|S_1^n] - h_{1,0}}{h_{1,1} - h_{1,0}} - \frac{h_0(\mathbf{Z}) - h_{1,0}}{h_{1,1} - h_{1,0}}\right)^2 \middle| S_1^n\right]\right) \\ \leq&\mathcal{O}\left(\mathbb{V}[\hat{h}_{1,0}|S_1^n] + \mathbb{V}[\hat{h}_{1,1}|S_1^n] + \mathbb{V}[\hat{h}_0(\mathbf{Z})|S_1^n] + \left(\mathbb{E}[\hat{h}_0(\mathbf{Z})|S_1^n] - h_0(\mathbf{Z})\right)^2\right) \quad \text{Lemma 20} \\ =&\mathcal{O}\left(\max\left(\mathbb{V}[\hat{h}_{1,0}|S_1^n], \mathbb{V}[\hat{h}_{1,1}|S_1^n], \mathbb{E}\left[(\hat{h}_0(\mathbf{Z}) - h_0(\mathbf{z}))^2|S_1^n\right]\right)\right) \\ =&\mathcal{O}\left(\max\left(n_L^{-1}, \mathbb{E}\left[(\hat{h}_0(\mathbf{Z}) - h_0(\mathbf{z}))^2|S_1^n\right]\right)\right) \end{aligned}$$

The last equality follows from observing that $\hat{h}_{1,0}$ and $\hat{h}_{1,1}$ are sample averages. ■

Appendix B. Additional Figures

These are additional figures to the experiments of Sections 2.5 and 3.1.

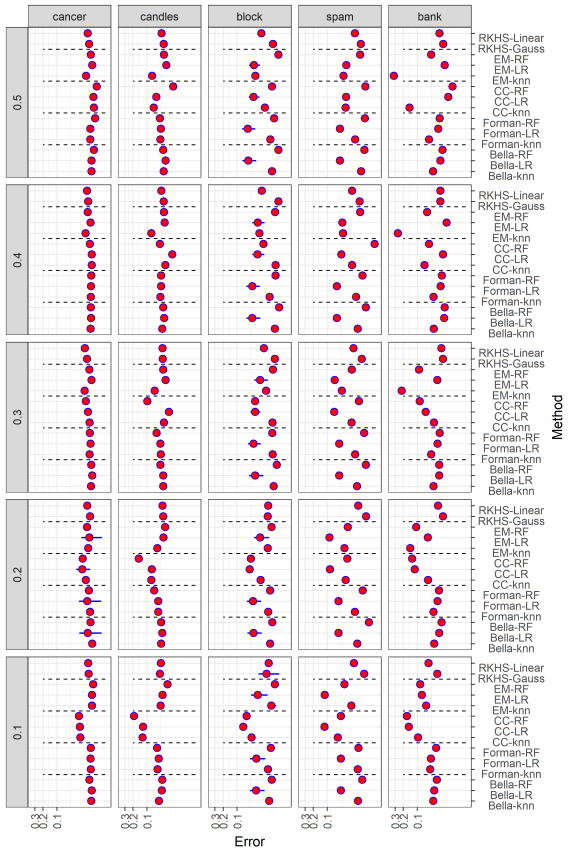


Figure 9: Root mean square deviation of each method by setting in logarithmic scale (including K-NN estimator).

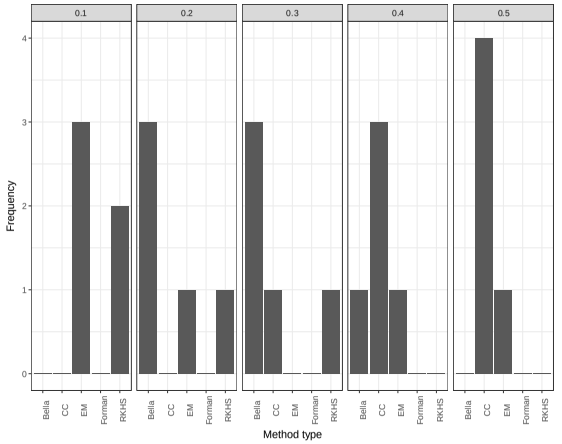


Figure 10: Number of times in which each specific method presents smaller MSE by θ values.

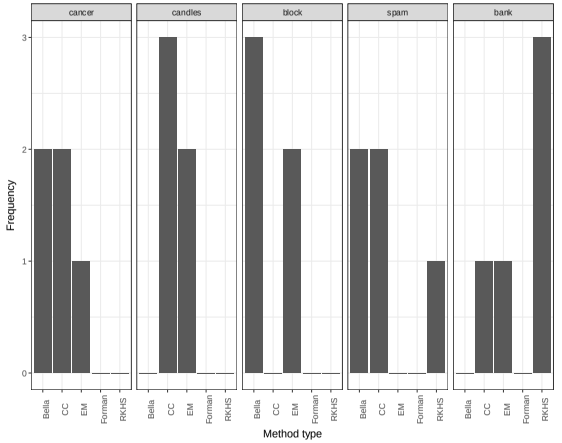


Figure 11: Number of times in which each specific method presents smaller MSE by data set.

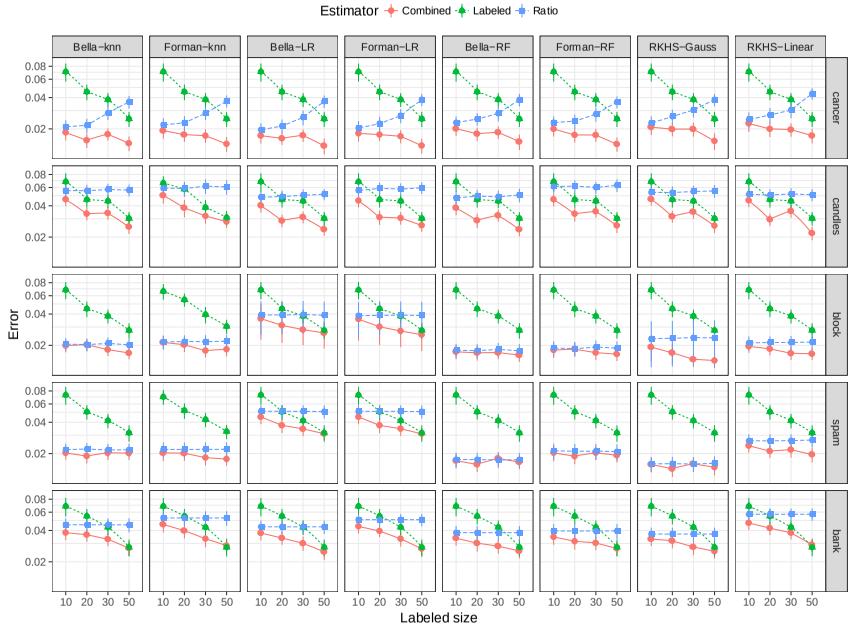


Figure 12: Root mean square deviation in logarithmic scale for each data set, method and estimator by size of the labeled sample and using $\theta = 0.1$.

References

J. Barranquero, J. Díez, and J. J. del Coz. Quantification-oriented learning based on reliable classifiers. *Pattern Recognition*, 48(2):591–604, 2015.

A. Bella, C. Ferri, J. Hernández-Orallo, and M. J. Ramirez-Quintana. Quantification via probability estimators. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 737–742. IEEE, 2010.

C. Blake. Uci repository of machine learning databases. *http://www. ics. uci. edu/˜ mlearn/MLRepository. html*, 1998.

G. Casella and R. L. Berger. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA, 2002.

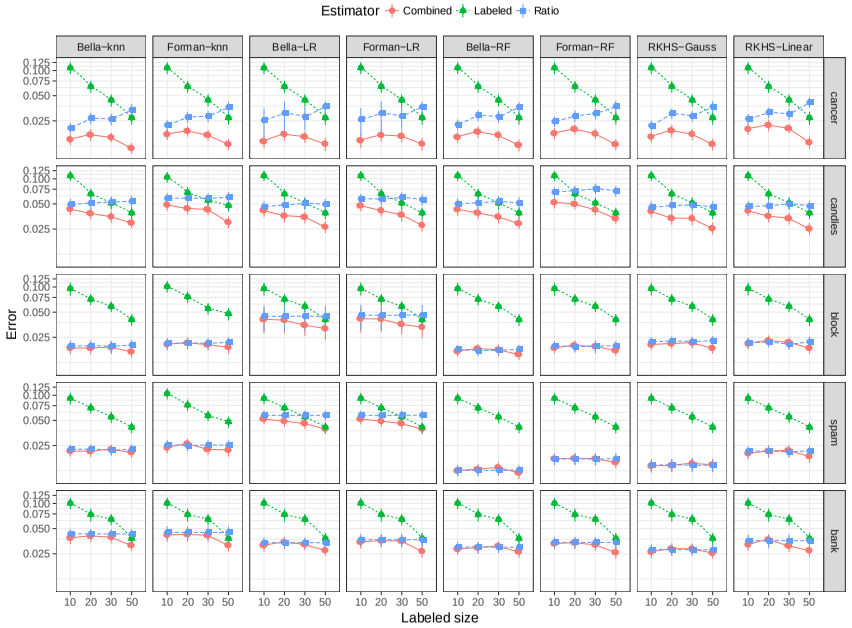


Figure 13: Root mean square deviation in logarithmic scale for each data set, method and estimator by size of the labeled sample and using $\theta = 0.2$.

B. de Finetti. *Theory of probability: a critical introductory treatment*, volume 6. John Wiley & Sons, 2017.

L. G. Esteves, R. Izbicki, and R. B. Stern. Teaching decision theory proof strategies using a crowdsourcing problem. *The American Statistician*, 71(4):336–343, 2017.

T. Fawcett and P. A. Flach. A response to Webb and Ting's *On the application of ROC analysis to predict classification performance under varying class distributions. Machine Learning*, 58(1):33–38, 2005.

G. Forman. Quantifying trends accurately despite classifier error and class imbalance. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 157–166. ACM, 2006.

G. Forman. Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery*, 17(2):164–206, 2008.

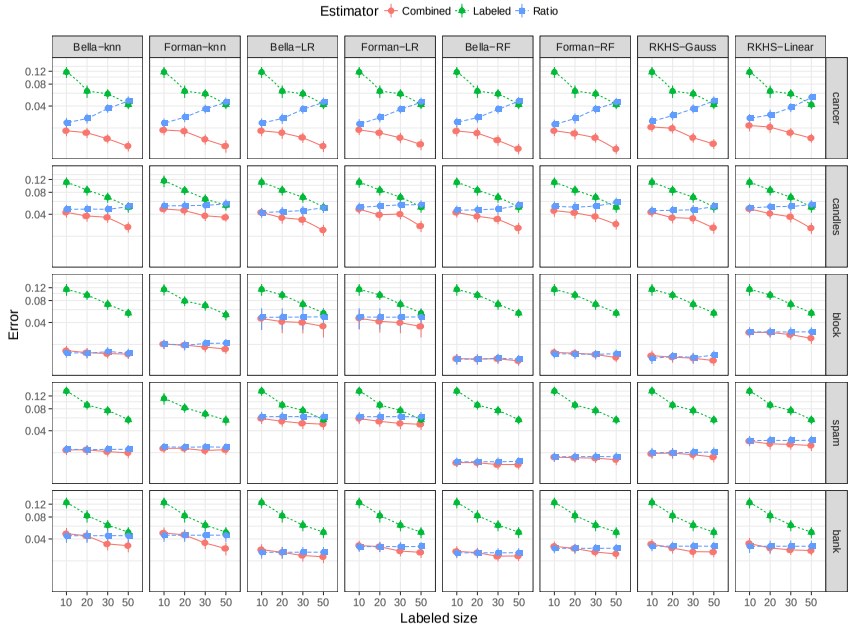


Figure 14: Root mean square deviation in logarithmic scale for each data set, method and estimator by size of the labeled sample and using $\theta = 0.4$.

PE Freeman, R Izbicki, AB Lee, JA Newman, CJ Conselice, AM Koekemoer, JM Lotz, and M Mozena. New image statistics for detecting disturbed galaxy morphologies at high redshift. *Monthly Notices of the Royal Astronomical Society*, 434(1):282–295, 2013.

J. J. Gart and A. A. Buck. Comparison of a screening test and a reference test in epidemiologic studies. ii. a probabilistic model for the comparison of diagnostic tests. *American Journal of Epidemiology*, 83(3):593–602, 1966.

K. Gerow. Fisher consistency-the evolution of a concept: It's hard to get it right the first time, 1989.

V. Hofer and G. Krempl. Drift mining in data: A framework for addressing drift in classification. *Computational Statistics & Data Analysis*, 57(1):377–391, 2013.

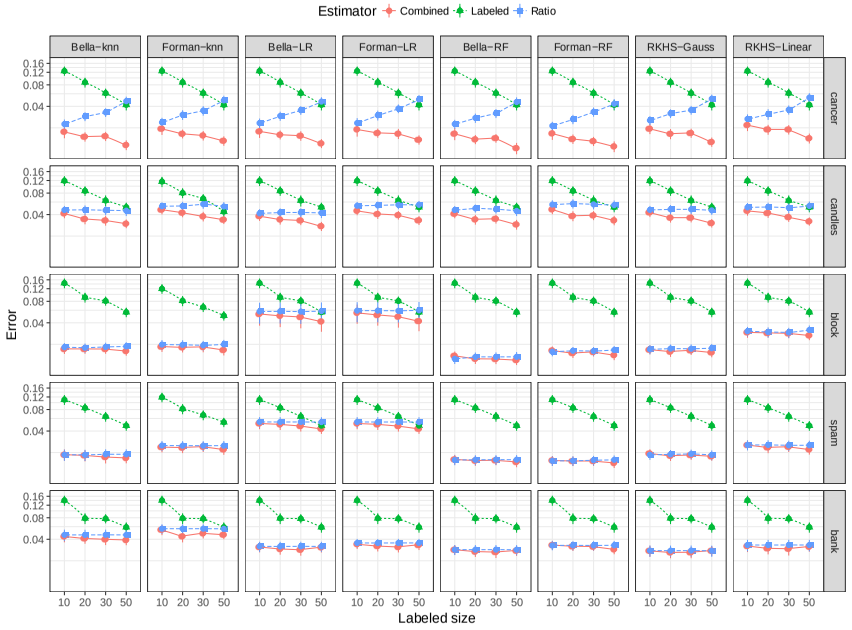


Figure 15: Root mean square deviation in logarithmic scale for each data set, method and estimator by size of the labeled sample and using $\theta = 0.5$.

R. Izbicki and R. B. Stern. Learning with many experts: model selection and sparsity. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 6(6):565–577, 2013.

R. E. Kass and P. W. Vos. *Geometrical foundations of asymptotic inference*, volume 908. John Wiley & Sons, 2011.

Z. C. Lipton, Y. X. Wang, and A. Smola. Detecting and correcting for label shift with black box predictors. *International Conference on Machine Learning (ICML)*, 2018.

D. Malerba, F. Esposito, and G. Semeraro. A further comparison of simplification methods for decision-tree induction. In *Learning from data*, pages 365–374. Springer, 1996.

O. L. Mangasarian. Cancer diagnosis via linear programming. *SIAM news*, 23(5):18, 1990.

C. Michelot. A finite algorithm for finding the projection of a point onto the canonical simplex of n. *Journal of Optimization Theory and Applications*, 50(1):195–200, 1986.

J. G. Moreno-Torres, T. Raeder, R. Alaiz-RodríGuez, N. V. Chawla, and F. Herrera. A unifying view on dataset shift in classification. *Pattern Recognition*, 45(1):521–530, 2012.

S. Moro, R. Laureano, and P. Cortez. Using data mining for bank direct marketing: An application of the crisp-dm methodology. In *Proceedings of European Simulation and Modelling Conference-ESM'2011*, pages 117–121. Eurosis, 2011.

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

M. Saerens, P. Latinne, and C. Decaestecker. Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. *Neural computation*, 14(1):21–41, 2002.

D. Tasche. Does quantification without adjustments work? *arXiv preprint arXiv:1602.08780*, 2016.

D. Tasche. Fisher consistency for prior probability shift. *Journal of Machine Learning Research*, 18(95):1–32, 2017. URL `http://jmlr.org/papers/v18/17-048.html`.

A. Vaz, R. Izbicki, and R. B. Stern. Prior shift using the ratio estimator. In *International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering*, pages 25–35. Springer, 2017.

G. Wahba. *Spline models for observational data*, volume 59. Siam, 1990.

H. Wang, D. Can, A. Kazemzadeh, F. Bar, and S. Narayanan. A system for real-time twitter sentiment analysis of 2012 u.s. presidential election cycle. In *Proceedings of the ACL 2012 System Demonstrations*, ACL '12, pages 115–120, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

L. Wasserman. *All of nonparametric statistics*. Springer, 2006.

L. Wasserman. *All of statistics: a concise course in statistical inference*. Springer Science & Business Media, 2013.

Journal of Machine Learning Research 20 (2019) 1-37 Submitted 10/18; Revised 3/19; Published 4/19

Learning to Match via Inverse Optimal Transport

Ruilin Li RUILIN.LI@GATECH.EDU
School of Mathematics
School of Computational Science and Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA

Xiaojing Ye XYE@GSU.EDU
Department of Mathematics and Statistics
Georgia State University
Atlanta, GA 30302, USA

Haomin Zhou HMZHOU@MATH.GATECH.EDU
School of Mathematics
Georgia Institute of Technology
Atlanta, GA 30332, USA

Hongyuan Zha ZHA@CC.GATECH.EDU
School of Computational Science and Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA

Editor: Xiaotong Shen

Abstract

We propose a unified data-driven framework based on inverse optimal transport that can learn adaptive, nonlinear interaction cost function from noisy and incomplete empirical matching matrix and predict new matching in various matching contexts. We emphasize that the discrete optimal transport plays the role of a variational principle which gives rise to an optimization based framework for modeling the observed empirical matching data. Our formulation leads to a non-convex optimization problem which can be solved efficiently by an alternating optimization method. A key novel aspect of our formulation is the incorporation of marginal relaxation via regularized Wasserstein distance, significantly improving the robustness of the method in the face of noisy or missing empirical matching data. Our model falls into the category of prescriptive models, which not only predict potential future matching, but is also able to explain what leads to empirical matching and quantifies the impact of changes in matching factors. The proposed approach has wide applicability including predicting matching in online dating, labor market, college application and crowdsourcing. We back up our claims with numerical experiments on both synthetic data and real world data sets.

Keywords: Matching, Inverse Problem, Optimal Transport, Robustification, Variational Inference

©2019 Ruilin Li, Xiaojing Ye, Haomin Zhou and Hongyuan Zha.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-700.html.

1. Introduction

Matching is a key problem at the heart of many real-world applications, including online dating (Hitsch et al., 2010), labor market (David, 2001), crowdsourcing (Yuen et al., 2011), marriage (Becker, 1973), paper-to-reviewer assignment (Charlin et al., 2011), kidney transplant donor matching (Dickerson and Sandholm, 2015) and ad allocation (Mehta et al., 2013). Owing to the wide applicability and great importance of matching, 2012 Nobel prize in economics were awarded to two economists Lloyd Shapley and Alvin Roth for their fundamental theoretic work (Gale and Shapley, 1962) and substantive empirical investigations, experiments and practical design (Roth and Sotomayor, 1989, 1992) on matching. A good matching of individuals from two sides (e.g., men vs. women, students vs. school, papers vs. reviewers) is essential to the overall health of the specific market/community. However, matching is a challenging problem due to two major complications: individuals from both sides exhibit various observable and latent features, which makes "suitability of a match" far more complex to assess; and the matching is implicitly, but significantly, influenced by the supply limitations of individuals from each side, so that the supply of an item can only satisfy a small number of users even though it is preferred by many. These two issues must be properly tackled in an optimal matching system.

In many matching problems, feature and preference data can be collected from individuals of either or both sides of the matching. Then a central planner may use such data sets to infer suitable matching or assignment. The feature and preference data collected in this way, however, can be incomplete, noisy, and biased for two reasons:

- an individual may not be aware of the competitors from her own side and/or limited quantity of her preferred match from the opposite side
- collection of a full spectrum of features is inherently difficult or even infeasible (e.g., a student's merit outside of her school curriculum in college admission, or religious belief of a person in a marriage, may not be included in the collected data)

The former factor prevents individuals from listing their orders of preferences and positioning themselves strategically in the market, and the latter results in feature data set that is incomplete and biased.

One possible approach is to use observed (and perhaps latent) features of individuals to generate rating matrix for user-item combinations as in many recomender systems (RS). However, this approach is not suitable given the bias and noise in collected feature or preference data and limited supply constraints in our matching problems. For example, in a standard movie RS problem, a movie can receive numerous high ratings and be watched by many people. In contrary, in a matching-based college admission problem, a student can enter only one college. Therefore, an optimal matching cannot be obtained solely based on personal ratings and preferences—the population of both sides also need to be taken into consideration in a matching problem. This significant difference between standard recommendation and matching demands for new theoretical and algorithmic developments.

Our approach to tackle the aforementioned challenges in matching inference is to consider a generalized framework based on inverse optimal transport, where the diversified population of each side of the matching is naturally modeled as a probability distribution,

and the bilateral preference of two individuals in a potential match is captured by a matching reward (or equivalently, negative matching cost). More specifically, we obtain kernel representation of the cost by learning the feature interaction matrix from the matching data, under which the total social surplus is supposed to be maximal in a healthy matching market as suggested by economists (Carlier and Ekeland, 2010). Moreover, we employ a robust and flexible Wasserstein metric to learn feature-enriched marginal distributions, which proves to be very effective and robust in dealing with incomplete/noisy data in the matching problem.

From a broader perspective, our approach is in the framework of optimization based on variational principles—the observed data are results of some optimization with an unknown objective function (or a known objective function with unknown parameters) that models the problem, and the goal is to learn the objective function (or its parameters) from the data. This approach is a type of prescriptive analytics: it exploits the motivation and mechanism of the subject, and produces results that are interpretable and meaningful to human. The solution process is more instructive and can make use of the observed data more effectively. In this broader sense, our proposed approach based on inverse optimal transport is in a similar spirit as inverse reinforcement learning (Ng et al., 2000). Furthermore, the learned objective can be used to understand the effect of various factors in a matching and infer optimal matching strategy given new data. For instance, in online dating, riders allocation, and many other settings, the central planners (Tinder, OkCupid, Uber, Lyft, etc.) can use such prescriptive models to improve customer experience and align social good with their own profit goal.

Our work is the first to establish a systematic framework for optimal matching learning using incomplete, noisy data under *limited-supply constraints*. In particular, we advocate a nonlinear representation of cost/reward in a matching and view the matching strategy as a solution of (regularized) optimal transport. The equilibrium of certain matching markets, such as marriage, with simplifying assumptions, coincide with optimal transport plans (Becker, 1973). Even for matching markets with complex structure and factors, whose matching mechanism is not yet completely unveiled, the proposed model serves as a powerful modeling tool to study those matchings. In terms of algorithmic development, we derive a highly efficient learning method to estimate the parameters in the cost function representation in the presence of computationally complex Wasserstein metrics. Numerical results show that our method contrasts favorably to other matching approaches in terms of robustness and efficiency, and can be used to infer optimal matching for new data sets accurately.

The rest of this paper is organized as follows: we briefly summarize related work in Section 2 and review discrete optimal transport and its regularized version as well as their close connections in Section 3. Section 4 describes the setup of proposed model and introduces our robust formulation via regularized Wasserstein distance, which tries to capture matching mechanism by leveraging regularized optimal transport. The derivation of optimization algorithm is detailed in Section 5. We evaluate our model in section 6 on both synthetic data and real-world data sets. The last section concludes the paper and points to several directions for potential future research.

2. Related Work

In this section, we briefly summarize some related work, including matching, ecological inference, recommender systems, distance metric learning and reciprocal recommendation.

2.1. Matching

Matching has been widely studied in economics community since the seminal work of Koopmans and Beckmann (1957). Gale and Shapley (1962) studied optimal matching in college admission, marriage market and proposed the famous Gale-Shapley algorithm. Becker (1973) gave a theoretic analysis in marriage market matching. Roth and Sotomayor (1992) did a thorough study and analysis in two-sided matching. Chiappori et al. (2010); Carlier and Ekeland (2010) used optimal transport theory to study the equilibrium of certain matching markets. Galichon and Salanié (2010) theoretically justified the usage of entropy-regularized optimal transport plan to model empirical matching in the presence of unobserved characteristics. Another interesting work (Charlin et al., 2011) proposed to predict optimal matching from learning suitability score in paper-to-review context where they used well-known linear regression, collaborative filtering algorithms to learn suitability scores. There are also some work studying dynamic matching theory and applications such as kidney exchange (Dickerson et al., 2012; Dickerson and Sandholm, 2015) and barter exchange (Anderson et al., 2017; Ashlagi et al., 2017).

A recent work closely related to ours is (Dupuy et al., 2016), where they worked with regularized optimal transport plan and modeled the cost by a bilinear form using an affinity matrix learned from data. By contrast, our work models the cost using a nonlinear kernel representation and incorporate regularized Wasserstein distance to tackle the challenging issues due the incomplete and noisy data in real-world matching problems.

2.2. Ecological Inference

Ecological inference infers the nature of individual level behavior using aggregate (historically called "ecological") data, and is of particular interest to political scientists, sociologists, historians and epidemiologists. Due to privacy or cost issue, individual level data are eluding from researchers, hence the inference made through aggregate data are often subject to ecological fallacy [1]. Previously, people proposed neighborhood model (Freedman et al., 1991), ecological regression (Goodman, 1953) and King's method (King, 2013). A recent progress (Flaxman et al., 2015) is made by using additional information and leverage kernel embedding of distributions, distribution regression to approach this problem.

Our work differs from classical ecological inference problem and methods in four ways. First, we assume access to empirical matching at individual-level granularity which is not available in standard ecological inference setting. Second, in our framework, we focus on learning the preference of two sides in the matching and propose a novel and efficient method to learn it, after which inference/prediction problem becomes trivial as preference is known. Third, different from previous statistical methods, we adopt a model-based approach, leverages optimal transport to model matching and draw a connection between these two fields. Lastly, thanks to the model-based approach, we are able to shed light on

1. https://en.wikipedia.org/wiki/Ecological_fallacy

what factors lead to empirical matching and quantitatively estimate the influence caused by changes of those factors, which are beyond the reach of traditional statistical approaches.

2.3. Recommender Systems

Collaborative filtering (CF) type recommender systems share many similarities with optimal matching problem as both need to learn user preference from rating/matching data and predict rating/matching in a collaborative manner. Matrix-factorization based models (Mnih and Salakhutdinov, 2008; Salakhutdinov and Mnih, 2008) enjoyed great success in Netflix Prize Competition. Rendle (2010, 2012) proposed factorization machine model with strong sparse predictive power and ability to mimic several state-of-the-art, specific factorization methods. Recently there is trend of combining collaborative filtering with deep learning (He and Chua, 2017; He et al., 2017). Most items recommended by conventional recommender systems, however, are non-exclusive and can be consumed by many customers such as movies and music. They do not take supply limit of either or both sides into consideration hence may perform poorly in matching context.

2.4. Distance Metric Learning

Our model essentially aims to learn an adaptive, nonlinear representation of the matching cost. This is closely related to, but more general than, ground metric learning. Prior research on learning different distance metrics in various contexts are fruitful, such as learning cosine similarity for face verification (Nguyen and Bai, 2010), learning Mahalanobis distance for clustering (Xing et al., 2003) and face identification (Guillaumin et al., 2009). However, distance learning for optimal transport distance is largely unexplored. Cuturi and Avis (2014) proposed to learn the ground metric by minimizing the difference of two convex polyhedral functions. Wang and Guibas (2012) formulated a SVM-like minimization problem to learn Earth Mover's distance. Both approaches work with Wasserstein distance which involves solving linear programming as subroutine hence may be computationally too expensive. This paper works with regularized optimal transport distance, involving solving a matrix scaling problem as subroutine which is much lighter than linear programming.

2.5. Reciprocal Recommendation

Another line of related research is reciprocal recommendation (Brozovsky and Petricek, 2007; Pizzato et al., 2013), which also tries to model two-side preference by computing reciprocal score via a hand-craft score function. By a sharp contrast, our model learns how two sides interact with each other from observed noisy/incomplete matching in a data-driven fashion.

3. Background and Preliminaries

In this section, we present Kantorovich's formulation of optimal transportation problem (in discretized setting) and its regularized version.

3.1. Optimal Transport

Given two probability vectors $\boldsymbol{\mu} \in \Sigma_m$ and $\boldsymbol{\nu} \in \Sigma_n$, where $\Sigma_d := \{\boldsymbol{x} \in \mathbb{R}_+^d | \mathbf{1}^T \boldsymbol{x} = 1\}$ is the standard $(d-1)$-dimensional probability simplex, denote the transport polytope of $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ by

$$U(\boldsymbol{\mu}, \boldsymbol{\nu}) := \{\pi \in \mathbb{R}_+^{m \times n} | \pi \mathbf{1} = \boldsymbol{\mu}, \pi^T \mathbf{1} = \boldsymbol{\nu}\}$$

namely the set of all $m \times n$ non-negative matrices satisfying marginal constraints specified by $\boldsymbol{\mu}, \boldsymbol{\nu}$. Note that $U(\boldsymbol{\mu}, \boldsymbol{\nu})$ is a convex, closed and bounded set containing joint probability distributions with $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ as marginals. Furthermore, if given a cost matrix $C = [C_{ij}] \in \mathbb{R}^{m \times n}$ where C_{ij} measures the cost of moving a unit mass from μ_i to ν_j, define

$$d(C, \boldsymbol{\mu}, \boldsymbol{\nu}) := \min_{\pi \in U(\boldsymbol{\mu}, \boldsymbol{\nu})} \langle \pi, C \rangle$$

where $\langle A, B \rangle = \mathrm{Tr}(A^T B)$ is the Frobenius inner product for matrices. This quantity describes how to optimally redistribute $\boldsymbol{\mu}$ to $\boldsymbol{\nu}$ so that the total cost is minimized, hence providing a means to measure the similarity between the two distributions. In particular, when $C \in \mathcal{M}^d$, that is, C is in the cone of distance matrices (Brickell et al., 2008), defined as

$$\mathcal{M}^d := \{C \in \mathbb{R}_+^{d \times d} | C_{ii} = 0, C_{ij} = C_{ji}, C_{ij} \leq C_{ik} + C_{kj}, \forall i, j, k\},$$

then it is shown that $d(C)$ is a distance (or metric) on Σ_d (Villani, 2008), named the optimal transport distance (also known as the 1-Wasserstein distance or the earth mover distance). The minimizer π is called the optimal transport plan.

In discrete case, computing OT distance amounts to solving a linear programming problem, for which there exists dedicated algorithm with time complexity $\mathcal{O}(n^3 \log n)$ (Pele and Werman, 2009). Nevertheless, this is still too computationally expensive in large scale settings. In addition, OT plan π typically admits a sparse form which is not robust in data-driven applications. We refer readers to Villani (2008); Peyré et al. (2017) for a thorough theoretical and computational treatment of optimal transport.

3.2. Regularized Optimal Transport

To address the aforementioned computational difficulty, Cuturi (2013) proposed to use a computationally-friendly approximation of OT distance by introducing entropic regularization. This also mitigates the sparsity and improve the smoothness of OT plan. Concretely, consider

$$d_\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu}) := \min_{\pi \in U(\boldsymbol{\mu}, \boldsymbol{\nu})} \{\langle \pi, C \rangle - H(\pi)/\lambda\}$$

where $H(\pi)$ is the discrete entropy defined by

$$H(\pi) = -\sum_{i,j=1}^{m,n} \pi_{ij} (\log \pi_{ij} - 1),$$

and $\lambda > 0$ is the regularization parameter controlling the trade-off between sparsity and uniformity of π. We refer the above quantity as regularized optimal transport (ROT) distance (regularized Wasserstein distance) though it is *not* an actual distance measure.

Algorithm 1 Sinkhorn-Knopp Algorithm

Input: marginal distributions $\boldsymbol{\mu}, \boldsymbol{\nu}$, cost matrix C, regularization parameter λ
$K = \exp(-\lambda C)$
$\boldsymbol{a} = \mathbf{1}$
while not converge **do**
  $\boldsymbol{b} \leftarrow \frac{\boldsymbol{\nu}}{K^T \boldsymbol{a}}$
  $\boldsymbol{a} \leftarrow \frac{\boldsymbol{\mu}}{K\boldsymbol{b}}$
end while
$\pi = \mathbf{diag}(\boldsymbol{a}) K \mathbf{diag}(\boldsymbol{b})$
return $\pi, \boldsymbol{a}, \boldsymbol{b}$

Due to the strict convexity introduced by entropy, $d_\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu})$ admits a unique minimizer with full support $\pi^\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu})$, which we call regularized optimal transport plan in the sequel. The ROT plan π^λ has a semi-closed form solution

$$\pi^\lambda = \mathbf{diag}(\boldsymbol{a}) K \mathbf{diag}(\boldsymbol{b}) \tag{1}$$

where $\boldsymbol{a} \in \mathbb{R}^m, \boldsymbol{b} \in \mathbb{R}^n$ are positive vectors and are uniquely determined up to a multiplicative constant and $K := \exp(-\lambda C)$ is the component-wise exponential of $-\lambda C$. We can efficiently compute $\boldsymbol{a}$ and $\boldsymbol{b}$ by Sinkhorn-Knopp matrix scaling algorithm (Sinkhorn and Knopp, 1967), also known as iterative proportional fitting procedure (IPFP). The algorithm alternately scales rows and columns of K to fit the specified marginals. See Algorithm 1 for detailed description of the Sinkhorn-Knopp algorithm.

Not surprisingly, we have

$$\lim_{\lambda \to \infty} d_\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu}) = d(C, \boldsymbol{\mu}, \boldsymbol{\nu})$$

ROT distance converges to OT distance as λ tends to infinity, i.e., entropic regularization diminishes. Moreover, let

$$\Pi(C, \boldsymbol{\mu}, \boldsymbol{\nu}) = \{\pi | \langle \pi, C \rangle = \min_{\pi \in U(\boldsymbol{\mu}, \boldsymbol{\nu})} \langle \pi, C \rangle\}$$

be the set of all OT plan and

$$\pi^\star = \operatorname*{arg\,max}_{\pi \in \Pi(C, \boldsymbol{\mu}, \boldsymbol{\nu})} H(\pi)$$

be the joint distribution with highest entropy within $\Pi(C, \boldsymbol{\mu}, \boldsymbol{\nu})$, then

$$\lim_{\lambda \to \infty} \pi^\lambda = \pi^\star$$

in another word, ROT plan converges to the most uniform OT plan and the rate of convergence is exponential, as shown by Cominetti and San Martín (1994). The generalization of entropic regularization, Tsallis entropy regularized optimal transport also receives more and more attention and is studied by Muzellec et al. (2017).

ROT has more favorable computational properties than OT does, as it only involves component-wise operation and matrix-vector multiplication, all of which are of quadratic

complexity, and can be parallelized (Cuturi, 2013). This fact makes ROT popular for measuring dissimilarity between potentially unnormalized distributions in many research fields: machine learning (Geneway et al., 2017; Rolet et al., 2016; Laclau et al., 2017), computer vision (Cuturi and Doucet, 2014) and image processing (Papadakis, 2015).

Besides computation efficiency, we argue in next section why it is more appropriate to use ROT in our setting from a modeling perspective.

4. Learning to Match

For the ease of exposition, we refer two sides of the matching market as users and items. The methodology is suitable in various applications where optimal matching is considered under supply limitations, such as marriage market, cab hailing, college admission, organ allocation, paper matching ans so on. Suppose we have m user profiles $\{\boldsymbol{u}_i\}_{i \in [m]} \subset \mathbb{R}^p$, n item profiles $\{\boldsymbol{v}_j\}_{j \in [n]} \subset \mathbb{R}^q$ and N_{ij}, the count of times $(\boldsymbol{u}_i, \boldsymbol{v}_j)$ appears in matching. Let $N = \sum_{i,j=1}^{m,n} N_{ij}$ be the number of all matchings, $[\hat{\pi}_{ij}] = [N_{ij}/N]$ be the observed matching matrix and $\hat{\boldsymbol{\mu}} = \hat{\pi}\mathbf{1}, \hat{\boldsymbol{\nu}} = \hat{\pi}^T\mathbf{1}$ be the sample marginals. Suppose we are also given two cost matrices C_u and C_v, measuring user-user dissimilarity and item-item dissimilarity respectively, we can then select two appropriate constants λ_u and λ_v and use $d_{\lambda_u}(C_u, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2)$ and $d_{\lambda_v}(C_v, \boldsymbol{\nu}_1, \boldsymbol{\nu}_2)$ to measure the dissimilarity of probability distributions $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ over user profile space and that of $\boldsymbol{\nu}_1, \boldsymbol{\nu}_2$ over item profile space.

4.1. Modeling Observed Matching Matrix

Becker (1973) pointed out that equilibrium of some matching markets coincide with optimal transport plans which are often highly sparse. The implication of this theory is far from being realistic, though, as we observe heterogeneous matchings in real world. Galichon and Salanié (2015) argued that there are latent features having significant impact on matching but unfortunately unobservable to researchers. Hence they proposed to leverage a combination of pure optimal transport plan and mutual information of two sides of matching to model empirical matching data which is exactly entropy-regularized optimal transport.

Furthermore, the observed matching matrix $\hat{\pi}$ (hence the empirical marginals) often contains noisy, corrupted, and/or missing entries, consequently it is more robust to employ a regularized optimal transport plan rather than enforce an exact matching to empirical data in cost function learning.

To that end, we propose to use regularized optimal transport plan $\pi^\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu})$ in our learning task. This also has several important benefits that take the following aspects into modeling consideration in addition to unobserved latent features:

- **Enforced Diversity.** Diversity is enforced in certain matchings as is the case when admission committee making decisions on applicants, diversity is often an important criterion and underrepresented minorities may be preferred. Entropy term captures the uncertainty introduced by diversity. The idea of connecting entropy with matching to capture/promote diversity is also adopted, for example, by Agrawal et al. (2018) and Ahmed et al. (2017).

- **Aggregated Data.** Sometimes due to privacy issues or insufficient number of matched pairs, only grouped or aggregated data, rather than individual data are available. Accordingly, the aggregated matching is usually denser than individual level matching and is less likely to exhibit sparsity.

4.2. Cost Function via Kernel Representation

The cost function $C_{ij} = c(\boldsymbol{u}_i, \boldsymbol{v}_j)$ is of critical importance as it determines utility loss of user $\boldsymbol{u}_i$ and item $\boldsymbol{v}_j$. The lower the cost is, the more likely user $\boldsymbol{u}_i$ will match item $\boldsymbol{v}_j$, subject to supply limit of items. A main contribution of this work is to learn an adaptive, nonlinear representation of the cost function from empirical matching data. To that end, we present several properties of cost function in optimal matching that support the feasibility.

First of all, we show in the following proposition that the cost function C is not unique in general but can be uniquely determined in a special and important case.

Proposition 1 *Given two marginal probability vectors $\boldsymbol{\mu} \in \Sigma_m$, $\boldsymbol{\nu} \in \Sigma_n$, define $F : \mathbb{R}^{m\times n} \to U(\boldsymbol{\mu}, \boldsymbol{\nu})$, $F(C) = \pi^\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu})$ is the ROT plan of C. Then F is in general not injective, however, when $m = n$ and F is restricted on $\mathcal{M}^n$, $F_{|\mathcal{M}^n}(C)$ is injective.*

Proof One can easily verify that F is well-defined from the strict convexity of ROT. The optimality condition of ROT reads as

$$\pi^\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu}) = \exp(\lambda(-C + \boldsymbol{a}\mathbf{1}^T + \mathbf{1}\boldsymbol{b}^T))$$

where $\boldsymbol{a} \in \mathbb{R}^m$ and $\boldsymbol{b} \in \mathbb{R}^n$ are Lagrangian multipliers dependent on C and λ such that $\pi^\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu}) \in U(\boldsymbol{\mu}, \boldsymbol{\nu})$. Therefore, $\pi^\lambda(C + \epsilon\mathbf{1}\mathbf{1}^T, \boldsymbol{\mu}, \boldsymbol{\nu}) = \exp(\lambda(-C - \epsilon\mathbf{1}\mathbf{1}^T + (\boldsymbol{a} + \epsilon\mathbf{1})\mathbf{1}^T + \mathbf{1}\boldsymbol{b}^T)) = \pi^\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu})$ for any $\epsilon > 0$. Therefore F is in general not injective.

If $m = n$ and $C_1, C_2 \in \mathcal{M}^n$, by the semi-closed form (1) of ROT plan, there exist positive vectors $\boldsymbol{a}_1, \boldsymbol{b}_1$ and $\boldsymbol{a}_2, \boldsymbol{b}_2$ such that

$$\pi^\lambda(C_1, \boldsymbol{\mu}, \boldsymbol{\nu}) = \mathbf{diag}(\boldsymbol{a}_1) \exp(-\lambda C_1) \mathbf{diag}(\boldsymbol{b}_1)$$
$$\pi^\lambda(C_2, \boldsymbol{\mu}, \boldsymbol{\nu}) = \mathbf{diag}(\boldsymbol{a}_2) \exp(-\lambda C_2) \mathbf{diag}(\boldsymbol{b}_2)$$

If $\pi^\lambda(C_1, \boldsymbol{\mu}, \boldsymbol{\nu}) = \pi^\lambda(C_2, \boldsymbol{\mu}, \boldsymbol{\nu})$, we have

$$\exp(-\lambda C_1) = \mathbf{diag}(\boldsymbol{a}) \exp(-\lambda C_2) \mathbf{diag}(\boldsymbol{b})$$

where $\exp(\cdot)$ is component-wise exponential, $\boldsymbol{a} = \log \frac{\boldsymbol{a}_2}{\boldsymbol{a}_1}$, $\boldsymbol{b} = \log \frac{\boldsymbol{b}_2}{\boldsymbol{b}_1}$.

Since C_1, C_2 are symmetric matrices, it follows that $\boldsymbol{a} = s\boldsymbol{b}$. By appropriately rescaling $\boldsymbol{a}$ and $\boldsymbol{b}$ to make them equal, we have

$$\exp(-\lambda C_1) = \mathbf{diag}(\boldsymbol{w}) \exp(-\lambda C_2) \mathbf{diag}(\boldsymbol{w})$$

where $\boldsymbol{w} = \boldsymbol{a}/\sqrt{s}$. Inspecting (i, i) entry of both sides, we immediately conclude that $\boldsymbol{w} = \mathbf{1}$ and $C_1 = C_2$. ■

Actually, the general non-uniqueness or non-identifiability of cost C is quite natural. For instance, in an online auction setting, if all bidders raise their bids by the same amount, the result of the auction will not change because the rank of bidders remain the same and the original winner still wins the auction. Therefore, by observing empirical matching alone, we can not determine cost matrix definitively without further assumption. Proposition 1 guarantees the uniqueness of learned cost if we model it as a distance matrix, e.g. Mahalanobis distance ($C_{ij} = \sqrt{(\boldsymbol{u}_i - \boldsymbol{v}_j)^T M(\boldsymbol{u}_i - \boldsymbol{v}_j)}$, where M is a positive definite matrix). However, in many cases, cost may grow nonlinearly in the difference of features. An even more serious issue is that if the number of features of two sides of matching are inconsistent or two sides do not lie in the same feature space at all, it would be infeasible to use a distance metric to capture the cost between them due to such dimension incompatibility.

Therefore, as generalized distance functions (Schölkopf, 2001), kernel representation which is able to measure matching cost even when features of two sides do not lie in the same feature space can be leveraged to model the cost function, i.e.,

$$c(\boldsymbol{u}_i, \boldsymbol{v}_j) = k(G\boldsymbol{u}_i, D\boldsymbol{v}_j)$$

where $k(\boldsymbol{x}, \boldsymbol{y})$ is a specific (possibly nonlinear) kernel, $G \in \mathbb{R}^{r\times p}$ and $D \in \mathbb{R}^{r\times q}$ are two unknown linear transformations to be learned. $G\boldsymbol{u}, D\boldsymbol{v}$ can be interpreted as the latent profile associated with users and items and are studied by Agarwal and Chen (2009).

For a wide class of commonly used kernels including linear kernel, polynomial kernel and sigmoid kernel, they depend only on the inner product of two arguments through an activation function f, i.e. $k(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x}^T\boldsymbol{y})$. For such kernels, we have

$$c(\boldsymbol{u}_i, \boldsymbol{v}_j) = f(\boldsymbol{u}_i^T G^T D \boldsymbol{v}_j)$$

and it suffices to learn $A = G^T D$. In this case, cost matrix

$$C(A) = f(U^T A V)$$

is parametrized by A and we refer A as interaction matrix. Here we apply f component-wise on $U^T AV$. For ease of presentation, we will work with kernels of this form in the sequel. With kernel function representation, it is still likely that a matching matrix corresponds to multiple cost matrices, and we will be contented with finding one of them that explains the observed empirical matching.

4.3. Kernel Inference with Wasserstein Marginal Regularization

A straight forward way to learn $C(A)$ in kernel representation is estimating parameter A through minimizing negative log likelihood

$$\min_A -\sum_{i=1}^m \sum_{j=1}^n \hat{\pi}_{ij} \log \pi_{ij} \tag{2}$$

where $\pi = \pi^\lambda(C(A), \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\nu}})$, i.e., one enforces the optimal plan π to satisfy $\pi\mathbf{1} = \hat{\boldsymbol{\mu}}$ and $\pi^T\mathbf{1} = \hat{\boldsymbol{\nu}}$. Note that (2) is equivalent to minimizing the reverse Kullback-Leibler divergence (Bishop, 2006) of ROT plan π with respect to empirical matching $\hat{\pi}$, i.e.,

$$\min_A \mathrm{KL}(\hat{\pi} \| \pi)$$

This is the formulation proposed in Dupuy et al. (2016) which we refer as inverse optimal transport formulation (IOT) in the sequel.

In this variation principle based framework, the ROT plan π has the same marginals as the empirical matching $\hat{\pi}$ does, which is reasonable if the marginal information of empirical matching is sufficiently accurate. In practice, however, the size of samples available is hardly enough, hence the empirical marginals inferred from samples can be incomplete and noisy. To see why this is the case, suppose the ground space $\mathcal{D} = \{1, -1\}^d$, $\boldsymbol{\mu}, \boldsymbol{\nu} \in \mathbb{R}^{2^d}$ are two discrete probability distributions over $\mathcal{D}$ and $\pi^0 \in \mathbb{R}^{2^d \times 2^d}$ is the ground truth matching matrix between $\boldsymbol{\mu}, \boldsymbol{\nu}$. Let $\{X_1, X_2, \cdots, X_N\} \subset \mathbb{R}^{2^d \times 2^d}$ be i.i.d random matrices, where $X_1 = E_{ij}$ with probability π^0_{ij}. Then empirical matching matrix is given by $\hat{\pi} = \bar{X} = \frac{1}{N}\sum_{n=1}^{N} X_n$ and two empirical marginals are $\hat{\boldsymbol{\mu}} = \bar{X}\mathbf{1}, \hat{\boldsymbol{\nu}} = \bar{X}^T\mathbf{1}$. By Bretagnolle-Huber-Carol inequality (Bretagnolle and Huber, 1979; Wellner et al., 2013), for any $\epsilon > 0$, we have

$$\mathbb{P}(\|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_1 > \epsilon) < 2^{2^d} \exp(\frac{-N}{2}\epsilon^2) = \exp(2^d \ln 2 - \frac{\epsilon^2}{2}N)$$
$$\mathbb{P}(\|\hat{\boldsymbol{\nu}} - \boldsymbol{\nu}\|_1 > \epsilon) < 2^{2^d} \exp(\frac{-N}{2}\epsilon^2) = \exp(2^d \ln 2 - \frac{\epsilon^2}{2}N)$$

Hence

$$\begin{aligned}\mathbb{P}(\max\{\|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_1, \|\hat{\boldsymbol{\nu}} - \boldsymbol{\nu}\|_1\} > \epsilon) &= \mathbb{P}(\{\|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_1 > \epsilon\} \cup \{\|\hat{\boldsymbol{\nu}} - \boldsymbol{\nu}\|_1 > \epsilon\}) \\ &\leq \mathbb{P}(\|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_1 > \epsilon) + \mathbb{P}(\|\hat{\boldsymbol{\nu}} - \boldsymbol{\nu}\|_1 > \epsilon) \\ &< 2\exp(2^d \ln 2 - \frac{\epsilon^2}{2}N)\end{aligned}$$

To ensure with at least $1 - \delta$ probability, the maximum error in marginal distributions $\max\{\|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_1, \|\hat{\boldsymbol{\nu}} - \boldsymbol{\nu}\|_1\} \leq \epsilon$, we might need $N(\epsilon, \delta) = \frac{2}{\epsilon^2}(2^d \ln 2 + \ln\frac{2}{\delta})$ samples. Note that $N(\epsilon, \delta)$ is quadratic in $\frac{1}{\epsilon}$ and exponential in d, indicating that practically there are hardly enough samples for us to obtain accurate estimate of marginal distributions $\boldsymbol{\mu}, \boldsymbol{\nu}$.

If using IOT formulation with noisy estimate of marginal distributions, it causes a *systematic error* no smaller than $\max\{\|\Delta\boldsymbol{\mu}\|_1, \|\Delta\boldsymbol{\nu}\|_1\}$ as shown in proposition 2.

Proposition 2 *If empirical $\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\nu}}$ are off from true $\boldsymbol{\mu}, \boldsymbol{\nu}$ by $\Delta\boldsymbol{\mu}, \Delta\boldsymbol{\nu}$, then the matching matrix π_{IOT} recovered by solving equation (2) has error lower bounded by*

$$\|\pi_0 - \pi_{IOT}\|_1 \geq \max\{\|\Delta\boldsymbol{\mu}\|_1, \|\Delta\boldsymbol{\nu}\|_1\}$$

where $\|\pi\|_1 = \sum_{i,j=1}^{m,n} |\pi_{ij}|$, $\hat{\boldsymbol{\mu}}, \boldsymbol{\mu} \in \mathbb{R}^m$ and $\hat{\boldsymbol{\nu}}, \boldsymbol{\nu} \in \mathbb{R}^n$, π_0 is the ground truth matching matrix, $\pi_{IOT} = \pi^{\lambda}(C(A^\star), \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\nu}})$ and $A^\star$ is the solution of equation (2).

Proof We know $\pi_0 \in U(\boldsymbol{\mu}, \boldsymbol{\nu})$ and $\pi_{\text{IOT}} \in U(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\nu}})$. By triangle inequality, we have

$$\sum_{i=1}^{m}\sum_{j=1}^{n} |(\pi_0)_{ij} - (\pi_{IOT})_{ij}| \geq \sum_{i=1}^{m} |\sum_{j=1}^{n}(\pi_0)_{ij} - (\pi_{IOT})_{ij}| = \sum_{i=1}^{m} |\boldsymbol{\mu}_i - \hat{\boldsymbol{\mu}}_i| = \|\Delta\boldsymbol{\mu}\|_1$$

Interchanging the summation and applying triangle inequality again, we obtain

$$\sum_{j=1}^{n}\sum_{i=1}^{m} |(\pi_0)_{ij} - (\pi_{IOT})_{ij}| \geq \|\Delta\boldsymbol{\nu}\|_1$$

Therefore, we conclude

$$\|\pi_0 - \pi_{\text{IOT}}\|_1 \geq \max\{\|\Delta\boldsymbol{\mu}\|_1, \|\Delta\boldsymbol{\nu}\|_1\}$$

■

We have seen that inaccurate marginal information can serious harm the recovery performance of ground truth matching matrix. Not unexpectedly, it could mislead us to learn an inaccurate cost matrix as well, as stated in proposition 4.

Lemma 3 *Suppose $M \in \mathbb{R}^{m\times n}$ and $f(\boldsymbol{a}, \boldsymbol{b}) = \|\boldsymbol{a}\mathbf{1}^T + \mathbf{1}\boldsymbol{b}^T - M\|_F^2$. Then we have*

$$f(\boldsymbol{a}, \boldsymbol{b}) \geq \|M\|_F^2 - \boldsymbol{f}^T A^+ \boldsymbol{f}$$

where $\boldsymbol{f} = [(M\mathbf{1})^T, \mathbf{1}^T M]^T$, $A = \begin{bmatrix} nI_{m\times m} & \mathbf{1}_m\mathbf{1}_n^T \\ \mathbf{1}_n\mathbf{1}_m^T & mI_{n\times n}\end{bmatrix}$, A^+ is the Moore-Penrose inverse of matrix A and $\|M\|_F = \sqrt{\sum_{i=1}^m\sum_{j=1}^n M_{ij}^2}$ is Frobenius norm. In particular, if M can not be written as $M = \boldsymbol{a}\mathbf{1}^T + \mathbf{1}\boldsymbol{b}^T$, the lower bound is strictly positive, i.e.,

$$f(\boldsymbol{a}, \boldsymbol{b}) \geq \|M\|_F^2 - \boldsymbol{f}^T A^+ \boldsymbol{f} > 0$$

Proof See Appendix A. ■

Proposition 4 *Suppose $\pi_0 \in \mathbb{R}^{m\times n}$ is the ground truth matching matrix, $\hat{\pi} \in \mathbb{R}^{m\times n}$ is an empirical matching matrix. Let C_0 be the ground truth cost matrix giving rise to π_0 and $C_{IOT} = \arg\min_{C\in\mathbb{R}^{m\times n}} KL(\hat{\pi}||C, \hat{\pi}\mathbf{1}, \hat{\pi}^T\mathbf{1})$ be the learned cost matrix via IOT formulation that gives rise to $\hat{\pi}$, i.e. $\pi_0 = \pi^{\lambda}(C_0, \pi_0\mathbf{1}, \pi_0^T\mathbf{1})$ and $\hat{\pi} = \pi^{\lambda}(C_{IOT}, \hat{\pi}\mathbf{1}, \hat{\pi}^T\mathbf{1})$. Denote $\Delta C = C_0 - C_{IOT}$ and $\Delta\log\pi = \log\pi_0 - \log\hat{\pi}$ and further assume $(\Delta\log\pi)_{ij}$ are independent (absolutely) continuous random variables (w.r.t. Lebesgue measure), we have*

$$\|\Delta C\|_F^2 \geq \frac{1}{\lambda^2}(\|\Delta\log\pi\|_F^2 - \boldsymbol{f}^T A^+ \boldsymbol{f}) > 0 \qquad a.e. \tag{3}$$

where $\boldsymbol{f} = [(\Delta\log\pi\mathbf{1})^T, \mathbf{1}^T\Delta\log\pi]^T$, $A = \begin{bmatrix} nI_{m\times m} & \mathbf{1}_m\mathbf{1}_n^T \\ \mathbf{1}_n\mathbf{1}_m^T & mI_{n\times n}\end{bmatrix}$, A^+ is the Moore-Penrose inverse of matrix A and $\|M\|_F = \sqrt{\sum_{i=1}^m\sum_{j=1}^n M_{ij}^2}$ is Frobenius norm.

Proof First we show that $\min_{C\in\mathbb{R}^{m\times n}} KL(\hat{\pi}||C, \hat{\pi}\mathbf{1}, \hat{\pi}^T\mathbf{1}) = 0$, i.e., $\hat{\pi} = \pi^{\lambda}(C_{IOT}, \hat{\pi}\mathbf{1}^T, \hat{\pi}^T\mathbf{1})$, any empirical matching can be realized as regularized optimal transport plan for some cost matrix.

Given arbitrary $\boldsymbol{\alpha} \in \mathbb{R}^m, \boldsymbol{\beta} \in \mathbb{R}^n$, let $\tilde{C} = \boldsymbol{\alpha}\mathbf{1}^\top + \mathbf{1}\boldsymbol{\beta}^\top - \frac{1}{\lambda}\log\hat{\pi}$, we see easily verify that $\hat{\pi}, \boldsymbol{\alpha}, \boldsymbol{\beta}$ solve the KKT condition of ROT

$$\min_{\pi \in U(\hat{\pi}\mathbf{1}, \hat{\pi}^T\mathbf{1})} \langle \tilde{C}, \pi \rangle - \frac{H(\pi)}{\lambda}$$

Hence $\hat{\pi} = \pi^\lambda(\tilde{C}, \hat{\pi}\mathbf{1}^T, \hat{\pi}^T\mathbf{1})$ and $\min_{C \in \mathbb{R}^{m\times n}} KL(\hat{\pi}||C, \hat{\pi}\mathbf{1}, \hat{\pi}^T\mathbf{1}) = 0$. Therefore, any minimizer C_{IOT} (need not be $\tilde{C}$) of the problem $\min_{C \in \mathbb{R}^{m\times n}} KL(\hat{\pi}||C, \hat{\pi}\mathbf{1}, \hat{\pi}^T\mathbf{1})$ must satisfy

$$\hat{\pi} = \pi^\lambda(C_{IOT}, \hat{\pi}\mathbf{1}^T, \hat{\pi}^T\mathbf{1})$$

By the optimality condition of ROT, we know that there exist $\boldsymbol{a}, \boldsymbol{b}$ such that

$$\pi^\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu}) = \exp(\lambda(-C + \boldsymbol{a}^T\mathbf{1} + \mathbf{1}\boldsymbol{b}^T))$$

hence there exist $\boldsymbol{a}_0, \boldsymbol{b}_0, \hat{\boldsymbol{a}}, \hat{\boldsymbol{b}}$ such that

$$C_0 = \boldsymbol{a}_0\mathbf{1}^T + \mathbf{1}\boldsymbol{b}_0^T - \frac{1}{\lambda}\log\pi_0$$
$$C_{IOT} = \hat{\boldsymbol{a}}\mathbf{1}^T + \mathbf{1}\hat{\boldsymbol{b}}^T - \frac{1}{\lambda}\log\hat{\pi}$$

Take difference and denote $\boldsymbol{a}_0 - \hat{\boldsymbol{a}}, \boldsymbol{b}_0 - \hat{\boldsymbol{b}}$ by $\Delta\boldsymbol{a}, \Delta\boldsymbol{b}$ respectively, we have

$$\Delta C = \Delta\boldsymbol{a}\mathbf{1}^T + \mathbf{1}\Delta\boldsymbol{b}^T - \frac{1}{\lambda}\Delta\log\pi$$

Since singular matrices have zero Lebesgue measure and $(\Delta\log\pi)_{ij}$ are independent continuous random variables, we have

$$\mathbb{P}(\Delta\log\pi = \boldsymbol{a}\mathbf{1}^T + \mathbf{1}\boldsymbol{b}^T) \le \mathbb{P}(\det(\Delta\log\pi) = 0) = 0$$

By lemma 3, we obtain

$$\|\Delta C\|_F^2 \ge \frac{1}{\lambda^2}(\|\Delta\log\pi\|_F^2 - \boldsymbol{f}^T A^+ \boldsymbol{f}) > 0 \qquad a.e.$$

■

If we use the inaccurate cost matrix learned via IOT approach, it could negatively affect the quality of future matching prediction, as justified in proposition 5.

Proposition 5 *Let C_0 be any ground truth cost matrix, C_{IOT} be any learned cost matrix via IOT formulation and assume $C_{IOT} \notin \{C|C = C_0 + \boldsymbol{a}\mathbf{1}^T + \mathbf{1}\boldsymbol{b}^T$ for some $\boldsymbol{a}, \boldsymbol{b}\}$. Suppose the ground truth matching matrix is $\pi_0 = \pi^\lambda(C_0, \boldsymbol{\mu}, \boldsymbol{\nu})$ and the predicted matching matrix is $\pi_{predict} = \pi^\lambda(C_{IOT}, \boldsymbol{\mu}, \boldsymbol{\nu})$. Denote $\Delta C = C_0 - C_{IOT}$ and $\Delta\log\pi = \log\pi_0 - \log\pi_{predict}$, we have*

$$\|\Delta\log\pi\|_F^2 \ge \lambda^2(\|\Delta C\|_F^2 - \boldsymbol{f}^T A^+ \boldsymbol{f}) > 0 \tag{4}$$

where $\boldsymbol{f} = [(\Delta C\mathbf{1})^T, \mathbf{1}^T\Delta C]^T$, $A = \begin{bmatrix} nI_{m\times m} & \mathbf{1}_m\mathbf{1}_n^T \\ \mathbf{1}_n\mathbf{1}_m^T & mI_{n\times n} \end{bmatrix}$, A^+ is the Moore-Penrose inverse of matrix A and $\|M\|_F = \sqrt{\sum_{i=1}^m\sum_{j=1}^n M_{ij}^2}$ is Frobenius norm.

Proof The proof is almost identical to that of proposition 4 except for interchanging the role of ΔC and $\Delta\log\pi$. We hence omit the details here. ■

To address aforementioned issues, we hence propose a more robust formulation with Wasserstein marginal relaxation, dropping the hard marginal constraint. Concretely, we consider the following optimization problem.

$$\min_{A, \boldsymbol{\mu}\in\Sigma_m, \boldsymbol{\nu}\in\Sigma_n} -\sum_{i=1}^m\sum_{j=1}^n \hat{\pi}_{ij}\log\pi_{ij} + \delta\big(d_{\lambda_u}(C_u, \boldsymbol{\mu}, \hat{\boldsymbol{\mu}}) + d_{\lambda_v}(C_v, \boldsymbol{\nu}, \hat{\boldsymbol{\nu}})\big) \tag{5}$$

where $\pi = \pi^\lambda(C(A), \boldsymbol{\mu}, \boldsymbol{\nu})$ is the regularized optimal transport plan, δ is the relaxation parameter controlling the fitness of marginals, $\lambda, \lambda_u, \lambda_v$ are hyper-parameters controlling the regularity of regularized Wasserstein distance. We refer this formulation as robust inverse optimal transport (RIOT) in the sequel. Interestingly, we note that Chizat et al. (2016) proposed a similar but different formulation in solving unbalanced optimal transport problem.

The intuition of this RIOT formulation is that instead of enforcing noisy empirical marginals as hard constraints, we incorporate them as soft constraints in objective function. We use regularized Wasserstein distance as regularization because of the following reasons:

- as approximated Wasserstein distance, it drives $\boldsymbol{\mu}, \boldsymbol{\nu}$ to $\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\nu}}$, but at the same time it also allows some uncertainty hence is able to robustify the result;
- in presence of missing entries in marginals, Wassertein distance is still well defined while other measures such as KL are not;
- Wasserstein distance can be applied to continuous, discrete, or even mixed distributions;
- computation of regularized Wasserstein distance (Cuturi, 2013) is efficient and hence potentially more scalable for large scale problem (5) in practice.

To be precise, the robustness in RIOT specifically means that RIOT outperforms IOT, in terms of the quality of learned matching matrix and cost matrix, when the observed joint distributions $\hat{\pi}$ and marginal distributions $\hat{\mu}$ and $\hat{\nu}$ are inaccurate due to insufficient and noisy samples, which is generally the case in real-world applications. In this situation, enforcing hard constraints to the inaccurate distributions is shown to cause severe bias in the estimation of the ground cost C, which will further induce errors in the matching when applied to testing data. The robustness of RIOT over IOT is verified by empirical experiment results in Section 6.1.

We assume access to C_u and C_v in our model because learning user-user/item-item similarity is relatively easier than our task, there are many existing work dedicated to that end (Cheung and Tian, 2004; Agarwal and Bharadwaj, 2013) and we want to single out and highlight our main contribution—learning the cost matrix that gives rise to observed matching and leverage it to infer matching for new data sets. In fact, our framework can also be extended to learn C_u and C_v jointly if needed, the optimization algorithm of which tends to be much more complex, though. See Appendix B for the extension. We postpone the detailed algorithmic derivation of the solution to (5) to next section.

4.4. Predict New Matching

After obtaining interaction matrix A from solving RIOT, we may then leverage it to predict new matching. Concretely, for a group of new users $\{\tilde{\boldsymbol{u}}_i\}_{i\in[m]}$ and items $\{\tilde{\boldsymbol{v}}_j\}_{j\in[n]}$, two marginal distributions, i.e., users profile distribution $\tilde{\boldsymbol{\mu}}$ and item profile distribution $\tilde{\boldsymbol{\nu}}$ can be easily obtained. First compute the cost matrix $\tilde{C}_{ij} = f(\tilde{\boldsymbol{u}}_i^T A \tilde{\boldsymbol{v}}_j)$ using kernel representation and apply Sinkhorn-Knopp algorithm to computing $\tilde{\pi}^\lambda(\tilde{C}, \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\nu}})$, which gives us the predicted matching of the given groups of users and items.

See Figure 1 for illustration of the complete pipeline or proposed learning-to-match framework.

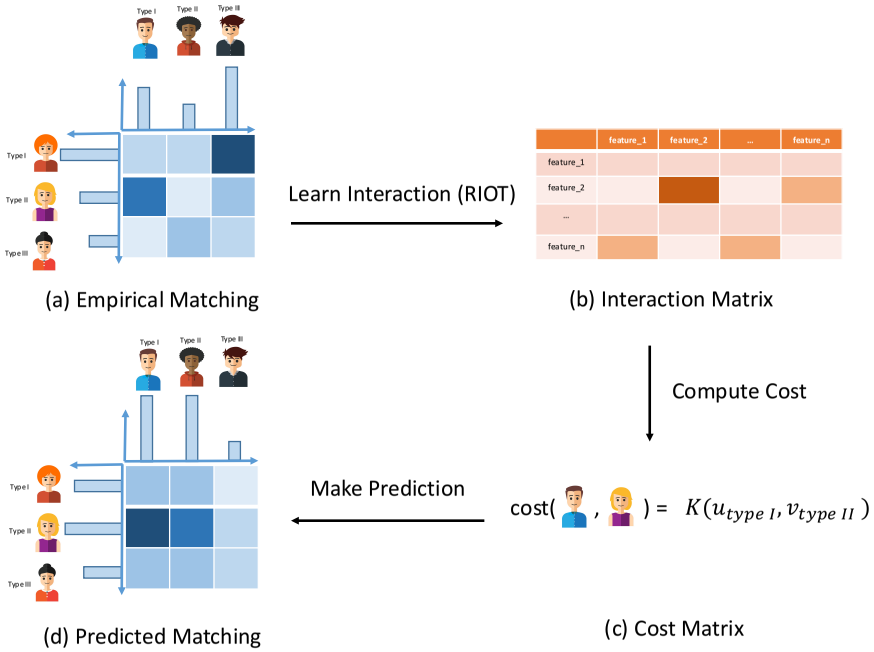


Figure 1: From noisy (a) empirical matching matrix, we learn (b) the interaction matrix via our proposed RIOT formulation. We then use kernel representation to compute (c) cost matrix and predict (d) matching matrix for new data. $\boldsymbol{u}_{type\ I}, \boldsymbol{v}_{type\ II}$ are feature vectors of type I men and type II women.

5. Derivation of Optimization Algorithm

Since the constraint set of ROT problem satisfies Slater's condition (Boyd and Vandenberghe, 2004), we have by strong duality that

$$d_\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu}) = \max_{\boldsymbol{z}} \langle \boldsymbol{z}, \boldsymbol{\mu} \rangle + \langle \boldsymbol{z}^C, \boldsymbol{\nu} \rangle - \frac{1}{\lambda}$$

where $z_j^C = \frac{1}{\lambda}\log c_j - \frac{1}{\lambda}\log(\sum_{i=1}^m e^{\lambda(z_i - C_{ij})})$. $\boldsymbol{z}, \boldsymbol{z}^C$ are essentially the Lagrangian multipliers corresponding to constraints $\pi\mathbf{1} = \boldsymbol{\mu}$ and $\pi^T\mathbf{1} = \boldsymbol{\nu}$. See also Genevay et al. (2016). Hence we have

$$d_{\lambda_u}(C_u, \boldsymbol{\mu}, \hat{\boldsymbol{\mu}}) = \max_{\boldsymbol{z}} \langle \boldsymbol{z}, \boldsymbol{\mu} \rangle + \langle \boldsymbol{z}^{C_u}, \hat{\boldsymbol{\mu}} \rangle - \frac{1}{\lambda_u}$$

$$d_{\lambda_v}(C_v, \boldsymbol{\nu}, \hat{\boldsymbol{\nu}}) = \max_{\boldsymbol{w}} \langle \boldsymbol{w}, \boldsymbol{\nu} \rangle + \langle \boldsymbol{w}^{C_v}, \hat{\boldsymbol{\nu}} \rangle - \frac{1}{\lambda_v}$$

where $z_j^{C_u} = \frac{1}{\lambda_u}\log \hat{r}_j - \frac{1}{\lambda_u}\log(\sum_{i=1}^m e^{\lambda_u(z_i - C_{uij})})$ and $w_j^{C_v} = \frac{1}{\lambda_v}\log \hat{c}_j - \frac{1}{\lambda_v}\log(\sum_{i=1}^n e^{\lambda_v(w_i - C_{vij})})$. Given sample marginals, once $\boldsymbol{z}, \boldsymbol{w}$ are fixed, $\boldsymbol{z}^{C_u}, \boldsymbol{w}^{C_v}$ are also fixed. We can then convert (5) into a min-max problem

$$\min_{A,\boldsymbol{\mu},\boldsymbol{\nu}} \max_{\boldsymbol{z},\boldsymbol{w}} -\sum_{i=1}^m \sum_{j=1}^n \hat{\pi}_{ij} \log \pi_{ij} + \delta\big(\langle \boldsymbol{z}, \boldsymbol{\mu} \rangle + \langle \boldsymbol{z}^{C_u}, \hat{\boldsymbol{\mu}} \rangle + \langle \boldsymbol{w}, \boldsymbol{\nu} \rangle + \langle \boldsymbol{w}^{C_v}, \hat{\boldsymbol{\nu}} \rangle\big) \tag{6}$$

where constants are omitted. The optimal solution is a saddle-point of the objective in (6). To solve this min-max problem, we alternately update the primal variable $(A, \boldsymbol{\mu}, \boldsymbol{\nu})$ and dual variable $(\boldsymbol{z}, \boldsymbol{w})$, each time with the other ones fixed.

5.1. Update $(A, \boldsymbol{\mu}, \boldsymbol{\nu})$ for fixed $(\boldsymbol{z}, \boldsymbol{w})$

Now $\boldsymbol{z}, \boldsymbol{w}, \boldsymbol{z}^{C_u}, \boldsymbol{w}^{C_v}$ are all fixed. Note that

$$\pi_{ij} = e^{\lambda(a_i + b_j - C_{ij})}$$

for some positive vectors $\boldsymbol{a}, \boldsymbol{b}$, such that $\pi\mathbf{1} = \boldsymbol{\mu}$, $\pi^T\mathbf{1} = \boldsymbol{\nu}$, and $\mathbf{1}^T\pi\mathbf{1} = 1$. Thus we may rewrite the minimization in this stage as

$$\begin{aligned} \min_{A,\boldsymbol{a},\boldsymbol{b}} \quad & \sum_{j=1}^n \hat{\pi}_{ij} \log \pi_{ij} + \delta(\langle z, \pi\mathbf{1} \rangle + \langle w, \pi^T\mathbf{1} \rangle) \\ \text{s.t.} \quad & \sum_{i=1}^m \sum_{j=1}^n e^{\lambda(a_i + b_j - C_{ij})} = 1 \end{aligned} \tag{7}$$

For fixed A, denote the optimum of objective function in equation (7) subject to the constraint by $E(C(A))$. Recall that the ultimate goal in this step is to find the interaction matrix A that cost C depends on, such that the minimum above can be attained. For any A, we have kernel representation $C(A)$ parameterized by interaction matrix A. Therefore the minimization above is equivalent to

$$\min_A E(C(A))$$

To minimize $E(C(A))$, the critical step is to evaluate gradient $\nabla_A E(C(A))$ and by envelope theorem (Milgrom and Segal, 2002) we have

Proposition 6 *The gradient $\nabla_A E(C(A))$ is*

$$\nabla_A E = \sum_{i=1}^{m}\sum_{j=1}^{n} \lambda[\hat{\pi}_{ij} + (\theta - \delta(z_i + w_j))\pi_{ij}]C'_{ij}(A)$$

where θ is the Lagrangian multiplier of the constrained minimization problem in equation (7).

Proof By chain rule, we have that

$$\nabla_A E = \sum_{i=1}^{m}\sum_{j=1}^{n} \frac{\partial E}{\partial C_{ij}}\frac{\partial C_{ij}}{A}$$

With the kernel representation, $C'_{ij}(A)$ is easily available. For fixed $C = C(A)$, by envelop theorem (Milgrom and Segal, 2002), we have

$$\begin{aligned}\nabla_{C_{ij}} E(C) &= \frac{\partial}{\partial C} - \sum_{i=1}^{m}\sum_{j=1}^{n}\hat{\pi}_{ij}\log\pi_{ij} + \delta\langle \boldsymbol{z}, \pi\mathbf{1}\rangle + \delta\langle \boldsymbol{w}, \pi^T\mathbf{1}\rangle - \theta(\sum_{i,j=1}^{m,n} e^{\lambda(a_i+b_j-C_{ij})}) \\ &= (-\frac{\hat{\pi}_{ij}}{\pi_{ij}} + \delta(z_i + w_j) - \theta)\frac{\partial \pi_{ij}}{\partial C_{ij}} \\ &= \lambda[\hat{\pi}_{ij} + (\theta - \delta(z_i + w_j))\pi_{ij}]\end{aligned}$$

■

Hence in each evaluation of $\nabla_C E$, we need to solve $E(C(A))$ once. If we denote $\xi_i = e^{\lambda a_i}$, $\eta_j = e^{\lambda b_j}$, $Z_{ij} = e^{-\lambda C_{ij}}$ and $M_{ij} = \delta(z_i + w_j)Z_{ij}$, then computing $E(C(A))$ is equivalent to solving

$$\begin{aligned}\min_{\boldsymbol{\xi},\boldsymbol{\eta}} \quad & -\langle\hat{\boldsymbol{\mu}}, \log\boldsymbol{\xi}\rangle - \langle\hat{\boldsymbol{\nu}}, \log\boldsymbol{\eta}\rangle + \boldsymbol{\xi}^T M\boldsymbol{\eta} \\ \text{s.t.} \quad & \boldsymbol{\xi}^T Z\boldsymbol{\eta} = 1\end{aligned} \tag{8}$$

Note that this is a non-convex optimization problem, both the objective function and constraints are non-convex which is difficult to solve in general. However, once we fix $\boldsymbol{\eta}$, the problem with respect to $\boldsymbol{\xi}$ alone is a convex problem and vice versa. We can solve this problem efficiently by alternately updating $\boldsymbol{\xi}, \boldsymbol{\eta}$.

Proposition 7 *Denote the objective in equation (8) by*

$$h(\boldsymbol{\xi}, \boldsymbol{\eta}) = -\langle\hat{\boldsymbol{\mu}}, \log\boldsymbol{\xi}\rangle - \langle\hat{\boldsymbol{\nu}}, \log\boldsymbol{\eta}\rangle + \boldsymbol{\xi}^T M\boldsymbol{\eta}$$

Initialize $\boldsymbol{\xi}^{(0)}, \boldsymbol{\eta}^{(0)}$ and alternately update $\boldsymbol{\xi}^{(k)}, \boldsymbol{\eta}^{(k)}$ in the following fashion

$$\boldsymbol{\xi}^{(k)} = \underset{\boldsymbol{\xi}^T Z\boldsymbol{\eta}^{(k-1)}=1}{\arg\min}\ h(\boldsymbol{\xi}, \boldsymbol{\eta}^{(k-1)}) \tag{9}$$

$$\boldsymbol{\eta}^{(k)} = \underset{\boldsymbol{\xi}^{(k)^T} Z\boldsymbol{\eta}=1}{\arg\min}\ h(\boldsymbol{\xi}^{(k)}, \boldsymbol{\eta}) \tag{10}$$

If $\lim_{k\to\infty}(\boldsymbol{\xi}^{(k)}, \boldsymbol{\eta}^{(k)}, \theta_1^{(k)}, \theta_2^{(k)}) = (\boldsymbol{\xi}^\star, \boldsymbol{\eta}^\star, \theta_1^\star, \theta_2^\star)$, then $\theta_1^\star = \theta_2^\star$ and $(\boldsymbol{\xi}^\star, \boldsymbol{\eta}^\star)$ is a local minimizer of h, where $\theta_1^{(k)}$ and $\theta_2^{(k)}$ are Lagrangian multipliers corresponding to problem (9) and (10) respectively.

Proof From the definition of $E(C(A))$ in equation (7) it is easily seen that $h(\boldsymbol{\xi}, \boldsymbol{\eta})$ is lower bounded. Moreover, since

$$h(\boldsymbol{\xi}^{(k)}, \boldsymbol{\eta}^{(k)}) \le h(\boldsymbol{\xi}^{(k)}, \boldsymbol{\eta}^{(k-1)}) \le h(\boldsymbol{\xi}^{(k-1)}, \boldsymbol{\eta}^{(k-1)})$$

there exists a convergent subsequence of $\{h(\boldsymbol{\xi}^{(k)}, \boldsymbol{\eta}^{(k)})\}$ and we denote the limit by $h^\star$.

The KKT condition of equation (9) and (10) are

$$-\frac{\hat{\boldsymbol{\mu}}}{\boldsymbol{\xi}^{(k)}} + M\boldsymbol{\eta}^{(k-1)} - \theta_1^{(k)} Z\boldsymbol{\eta}^{(k-1)} = 0 \tag{11}$$

$$\boldsymbol{\xi}^{(k)^T} Z\boldsymbol{\eta}^{(k-1)} = 1$$

$$-\frac{\hat{\boldsymbol{\nu}}}{\boldsymbol{\eta}^{(k)}} + M^T\boldsymbol{\xi}^{(k)} - \theta_2^{(k)} Z^T\boldsymbol{\xi}^{(k)} = 0 \tag{12}$$

$$\boldsymbol{\xi}^{(k)^T} Z\boldsymbol{\eta}^{(k)} = 1$$

Let k tend to infinity and take inner product with $\boldsymbol{\xi}^\star$ for equation (11) and take inner product with $\boldsymbol{\eta}^\star$ for equation (12), compare two equations and use the fact that both $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\nu}}$ are probability vectors, we find that $\theta_1^\star = \theta_2^\star$ and $(\boldsymbol{\xi}^\star, \boldsymbol{\eta}^\star)$ solves the KKT condition of constrained problem (9) and (10). Therefore, $(\boldsymbol{\xi}^\star, \boldsymbol{\eta}^\star)$ is a local minimizer and $h^\star = h(\boldsymbol{\xi}^\star, \boldsymbol{\eta}^\star)$ is a local minimum. ■

Once we obtain $(\boldsymbol{\xi}^\star, \boldsymbol{\eta}^\star, \theta^\star)$, we can then get $(\boldsymbol{a}, \boldsymbol{b}, \theta)$ by setting $\boldsymbol{a} = \frac{1}{\lambda}\log\boldsymbol{\xi}^\star$ and $\boldsymbol{b} = \frac{1}{\lambda}\log\boldsymbol{\eta}^\star$ and $\theta = \theta^\star$. Then plug in $(\boldsymbol{a}, \boldsymbol{b}, \theta)$ to evaluate $\nabla_C E$ for the current $C = C(A)$.

A careful analysis of the KKT condition of equations (9) and (10) shows that $\theta_1^{(k)}$ and $\theta_2^{(k)}$ are roots of

$$p(\theta) = \left\langle \frac{\hat{\boldsymbol{\mu}} \odot (Z\boldsymbol{\eta}^{(k-1)})}{(M - \theta Z)\boldsymbol{\eta}^{(k-1)}}, \mathbf{1}\right\rangle,\ \ q(\theta) = \left\langle \frac{\hat{\boldsymbol{\nu}} \odot (Z^T\boldsymbol{\xi}^{(k)})}{(M - \theta Z)^T\boldsymbol{\xi}^{(k)}}, \mathbf{1}\right\rangle$$

respectively. The univariate root finding problem can be solved efficiently by off-the-shelf package. After obtaining $\theta_1^{(k)}, \theta_2^{(k)}$, we can update

$$\boldsymbol{\xi}^{(k)} = \frac{\hat{\boldsymbol{\mu}}}{(M - \theta_1^{(k)} Z)\boldsymbol{\eta}^{(k-1)}},\ \ \boldsymbol{\eta}^{(k)} = \frac{\hat{\boldsymbol{\nu}}}{(M - \theta_2^{(k)} Z)^T\boldsymbol{\xi}^{(k)}}$$

directly. Computationally, this approach to solving problem (9) and (10) is much cheaper than gradient-type iterative methods when m and/or n are large.

5.2. Update $(\boldsymbol{z}, \boldsymbol{w})$ for fixed $(A, \boldsymbol{\mu}, \boldsymbol{\nu})$

When $(A, \boldsymbol{\mu}, \boldsymbol{\nu})$ are fixed, π is also fixed, we then only need to solve

$$\max_{\boldsymbol{z},\boldsymbol{w}} \langle \boldsymbol{z}, \pi \mathbf{1} \rangle + \langle \boldsymbol{z}^{C_u}, \hat{\boldsymbol{\mu}} \rangle + \langle \boldsymbol{w}, \pi^T \mathbf{1} \rangle + \langle \boldsymbol{w}^{C_v}, \hat{\boldsymbol{\nu}} \rangle$$

and one immediately recognizes this is equivalent to applying Sinkhorn-Knopp algorithm to compute $d_{\lambda_u}(C_u, \boldsymbol{\mu}, \hat{\boldsymbol{\mu}})$ and $d_{\lambda_v}(C_v, \boldsymbol{\nu}, \hat{\boldsymbol{\nu}})$.

To summarize, in each iteration, we perform a gradient-type update for A, followed by two calls of Sinkhorn-Knopp algorithm to compute $d_{\lambda_u}(C_u, \boldsymbol{\mu}, \hat{\boldsymbol{\mu}})$ and $d_{\lambda_v}(C_v, \boldsymbol{\nu}, \hat{\boldsymbol{\nu}})$. Algorithm 2 details the algorithm. Note that this algorithm only finds a local minimum of equation 5.

Algorithm 2 Solve RIOT

Input: observed matching matrix $\hat{\pi}$, cost matrices C_u, C_v, regularization parameter $\lambda, \lambda_u, \lambda_v$

for $l = 1, 2, \cdots, L$ **do**

$Z \leftarrow \exp(-\lambda C)$

$M \leftarrow \delta(\boldsymbol{z}\mathbf{1}^T + \mathbf{1}\boldsymbol{w}^T) \odot Z$

Initialize $\boldsymbol{\xi}^{(0)}, \boldsymbol{\eta}^{(0)}$

for $k = 1, 2, \cdots, K$ **do**

$\theta_1^{(k)} \leftarrow$ root of $p(\theta)$

$\theta_2^{(k)} \leftarrow$ root of $q(\theta)$

$\boldsymbol{\xi}^{(k)} \leftarrow \frac{\hat{\boldsymbol{\mu}}}{(M - \theta_1^{(k)} Z)\boldsymbol{\eta}^{(k-1)}}$

$\boldsymbol{\eta}^{(k)} \leftarrow \frac{\hat{\boldsymbol{\nu}}}{(M - \theta_2^{(k)} Z)^T \boldsymbol{\xi}^{(k)}}$

end for

$\boldsymbol{a} \leftarrow \frac{1}{\lambda} \log \boldsymbol{\xi}^{(k)}, \quad \boldsymbol{b} \leftarrow \frac{1}{\lambda} \boldsymbol{\eta}^{(k)}, \quad \theta = \theta_1^{(k)}$

$\pi \leftarrow \exp(\lambda(\boldsymbol{a}\mathbf{1}^T + \mathbf{1}\boldsymbol{b}^T - C))$

$\nabla_A \leftarrow \sum_{i,j=1}^{m,n} \lambda[\hat{\pi}_{ij} + (\theta - \delta(z_i + w_j))\pi_{ij}] C'_{ij}(A)$

$A \leftarrow A - s\nabla_A$

$\boldsymbol{a}_1 \leftarrow$ Sinkhorn-Knopp$(C_u, \pi\mathbf{1}, \hat{\boldsymbol{\mu}}, \lambda_u)[1]$

$\boldsymbol{a}_2 \leftarrow$ Sinkhorn-Knopp$(C_v, \pi^T\mathbf{1}, \hat{\boldsymbol{\nu}}, \lambda_v)[1]$

$\boldsymbol{z} \leftarrow \frac{1}{\lambda_u} \log \boldsymbol{a}_1, \quad \boldsymbol{w} \leftarrow \frac{1}{\lambda_v} \log \boldsymbol{a}_2$

end for

6. Experiments

In this section, we evaluate our proposed RIOT model on both synthetic data and real world data sets. For synthetic data set, we illustrate its robustness against IOT and show our model can achieve better performance in learning cost matrix C than IOT could. For election data set, we show our method can effectively learn meaningful preference of voters based on their demographics. For taxi trip data set, we demonstrate that the proposed model is able to predict matching of taxi drivers and passengers fairly accurate. For marriage

data set, we demonstrate the applicability of RIOT in predicting new matching and make recommendation accordingly by comparing it with baseline and state-of-art recommender systems.

6.1. Synthetic Data

We set $\lambda = \lambda_u = \lambda_v = 1$ and simulate $m = 10$ user profiles $\{\boldsymbol{u}_i\} \subset \mathbb{R}^{10}$, $n = 10$ item profiles $\{\boldsymbol{v}_j\} \subset \mathbb{R}^8$, two probability vectors $\boldsymbol{\mu}_0, \boldsymbol{\nu}_0 \in \mathbb{R}^{10}$, an interaction matrix A_0 of size 10×8 and pick polynomial kernel $k(\boldsymbol{x}, \boldsymbol{y}) = (\gamma \boldsymbol{x}^T \boldsymbol{y} + c_0)^d$ where $\gamma = 0.05, c_0 = 1, d = 2$, hence $C_{0ij} = (0.05\boldsymbol{u}_i^T A \boldsymbol{v}_j + 1)^2$. For C_u, C_v, we randomly generate m and n points from $\mathcal{N}(\mathbf{0}, 5I_2)$ on plane and use their Euclidean distance matrix as C_u and C_v. The ground truth entropy-regularized optimal transport plan is given by $\pi_0 = \pi^\lambda(C_0, \boldsymbol{\mu}_0, \boldsymbol{\nu}_0)$. We independently sample N samples from π_0 and then compute empirical matching matrix $\hat{\pi}$ accordingly. In algorithm 2, we set the number of iterations of inner loop $K = 20$.

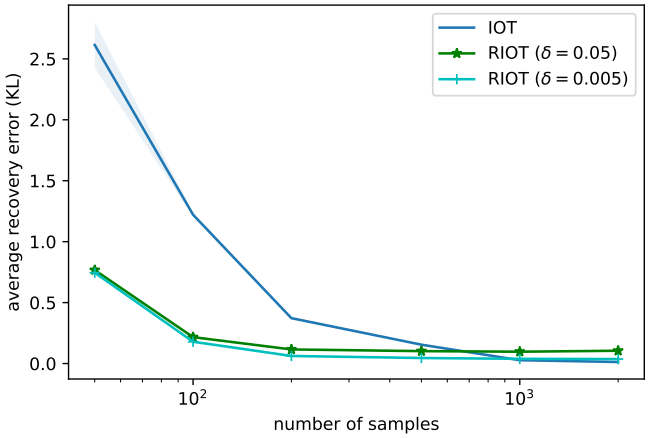
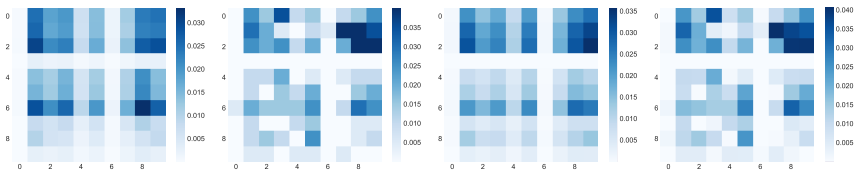


Figure 2: Comparison of recovery performance of fixed marginal approach (IOT) and marginal relaxation approach (RIOT). For sample size N, we sample from π_0, compute empiricla matching $\hat{\pi}$ and run algorithm 2 for 50 times. The shaded region is one standard deviation.

6.1.1. Improved Robustness

To produce Figure 2, we set the number of iterations in outer loop $L = 50$, learning rate $s = 10$. For each $N \in \{50, 100, 200, 500, 1000, 2000\}$ we run algorithm 2 and record Kullback-Leibler divergence between learned matching matrix π_{IOT}, π_{RIOT} and ground truth matching matrix π_0. Figure 2 shows that RIOT with different relaxation parameters



(a) π_0 (b) $\hat{\pi}$ (c) π_{RIOT} (d) π_{IOT}

Figure 3: Comparison of (a) actual matching matrix π_0 (b) noised matching matrix $\hat{\pi}$ ($\text{KL}(\pi_0\|\hat{\pi}) = 1.269$) (c) matching matrix π_{RIOT} learned by marginal relaxation approach ($\text{KL}(\pi_0\|\pi_{\text{RIOT}}) = 0.219$) (d) matching matrix π_{IOT} learned with fixed marginal approach ($\text{KL}(\pi_0\|\pi_{\text{IOT}}) = 0.464$) (sample size $N = 200$)

demonstrate improved robustness than IOT does when noise size is large. If δ is set too large (e.g. $\delta = 0.05$), however, the entropy term tends to dominate and negatively affect recovery performance when noise size is modestly small. If δ is tuned carefully (e.g. $\delta = 0.005$), RIOT can achieve comparable performance even when noise size is quite small. Moreover, we observe that curves corresponding to different δ intersect with the curve of fixed marginal at different noise size. Therefore, when prior knowledge or statistical estimate of noise size is available, we may tune δ accordingly to achieve best practical performance. In addition, we observe notable variation of KL divergence between π_0 and π_{IOT} when sample size is small (e.g., $N = 50$), in contrast, one standard deviation of $KL(\pi_0||\pi_{RIOT})$ is negligible, supporting our argument of RIOT being more robust than IOT.

To produce Figure 3, we set sample size $N = 200$ and relaxation parameter $\delta = 0.001$ with other parameters same as those for producing Figure 2. Figure 3 visually illustrates π_0, $\hat{\pi}$, π_{RIOT} and π_{IOT} and we see that when sample size is small, sampling noise can significantly corrupts= the ground truth matching matrix. π_{RIOT} exhibits less distortion compared to π_{IOT}, which demonstrates improved robustness again. Numerical results also back up our observation.

$$\text{KL}(\pi_0\|\hat{\pi}) = 1.269,\ \text{KL}(\pi_0\|\pi_{\text{RIOT}}) = 0.219,\ \text{KL}(\pi_0\|\pi_{\text{IOT}}) = 0.464$$

Compared to IOT, marginal relaxation via regularized Wasserstein distance does help improve the robustness of solution.

6.1.2. Superior Learning Performance

To produce Figure 4, we set sample size $N = 200$, relaxation parameter $\delta = 0.001$, the number of iterations in outer loop $L = 100$ and learning rate $s = 1$, we then run algorithm 2 to compare the performance of learning cost matrix C_0. To avoid non-uniqueness/non-identifiability issue, we use

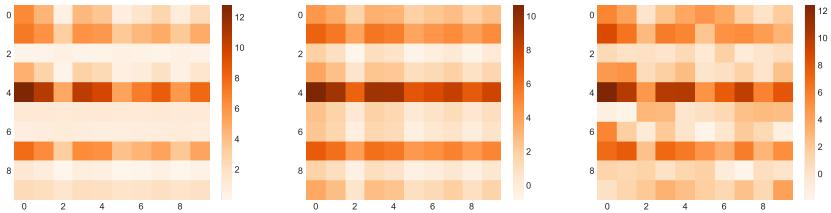
$$d(C_1, C_2) = \min_{D=\boldsymbol{a}\mathbf{1}^T+\mathbf{1}\boldsymbol{b}^T+C_1} \|D - C_2)\|_F$$

to measure the closeness of cost matrices C_1 and C_2 and denote the minimizer of $d(C, C_0)$ by $\tilde{C}$. The results are shown below,

$$d(C_{RIOT}, C_0) = 7.831, \quad d(C_{IOT}, C_0) = 12.439$$

where C_{RIOT}, C_{IOT} are cost matrices learned by RIOT formulation and IOT formulation respectively. Compared to C_{IOT}, C_{RIOT} learned via our proposed method almost halves the distance to ground truth cost matrix. Figure 4 also illustrates that our model can learn the structure of cost matrix better than IOT does. Our approach improves the learning performance and is able to reveal the structure of ground truth cost matrix.

To sum up, we show that with appropriately tuned relaxation parameter, RIOT is superior to IOT in terms of both robustness and learning performance.



(a) C_0 (b) $\tilde{C}_{\text{RIOT}}$ (c) $\tilde{C}_{\text{IOT}}$

Figure 4: Comparison of (a) ground truth cost matrix C_0 (b) $\tilde{C}_{\text{RIOT}}$, the minimizer of $d(C_{RIOT}, C_0)$ and (c) $\tilde{C}_{\text{IOT}}$, the minimizer of $d(C_{IOT}, C_0)$

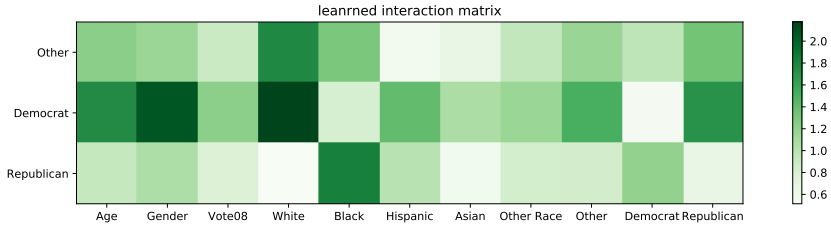


Figure 5: Interaction matrix learned by RIOT for election data set

6.2. Election data set

We show in this subsection that RIOT can effectively learn the user-item interaction by applying it to 2012 presidential election data of Florida. The experiment setup is similar to that of Muzellec et al. (2017)[2].

The data set contains more than 9.23×10^6 voters profile, each voter has features like gender, age, race, party and whether the voter voted in 2008 election. 0-1 encoding are used for gender (M:1, F:0) and voting in 2008 (Yes:1, No:0), age is linearly mapped onto $[0, 1]$ and we use one-hot encoding for both race and party. We obtain the empiricl matching data from exit poll provided by GALLUP[3]. The empirical matching matrix $\hat{\pi}$ is 3-by-5 matrix, candidates are either Democratic or Republican, or from a third party and voters are categorized into five races (White, Black, Hispanic, Asian and Other). We use mean profile as features for each race and one-hot encoding as features for candidates. We set $\lambda = \lambda_u = \lambda_v = 1, \delta = 5 \times 10^{-3}, K = 20$, for C_u, C_v, we randomly generate m and n points from $\mathcal{N}(\mathbf{0}, 5I_2)$ on plane and use their Euclidean distance matrix as C_u and C_v. Polynomial kernel $C_{ij} = (0.2\boldsymbol{u}_i^T A \boldsymbol{v}_j + 1)^2$ is used for this experiment. We run RIOT for this data set and the learned interaction matrix is shown in Figure 5.

In Figure 5, the lighter the color of a cell is, the lower the cost caused by that feature combination is. Take 'Age' column as an example, ('Democratic', 'Age') is the darkest and ('Republican', 'Age') is the lightest among the column, it means that elder voters are more likely to favor Republican candidate as it has lower cost compared to supporting Democratic candidate. Other cells can be interpreted in a similar manner.

From Figure 5, we see that most white voters tend to support Republican candidate Romney while black voters tend to support Democratic candidate Obama, Democratic and Republican voters tend to support candidate from their own party, elder voters tend to support Romney while female voters tend to support Obama. All above observations are consistent with CNN's[4] exit polls. This demonstrates that RIOT can learn meaningful interaction preference from empirical matching effectively.

6.3. New York Taxi data set

We demonstrate in this subsection that the proposed RIOT framework is able to predict fairly accurate matching on New York Taxi data set [5]. This data set contains 1458644 taxi trip records from January to June in 2016 in New York city. Each trip record is associated with one of the two data vendors (Creative Mobile Technologies, LLC and VeriFone Inc.) and contains detailed trip information such as pickup/drop-off time, longitude, latitude and so on. As no unique identifiers of taxis are provided, we can not predict new matching on individual level. Instead, we predict matching between data vendors and passengers (a passenger is matched with one of the data vendors if a taxi associated with that data vendor rides with the passenger).

2. part of the experiment in this subsection is based on the code kindly shared by Boris Muzellec(https://github.com/BorisMuzellec/TROT)
3. http://news.gallup.com/poll/160373/democrats-racially-diverse-republicans-mostly-white.aspx
4. http://www.cnn.com/election/2012/results/state/FL/president/
5. https://www.kaggle.com/c/nyc-taxi-trip-duration/data

To reflect the proximity of passengers and taxis, we cluster all trip records into 50 regions and plot them in Figure 6. If a passenger and a taxi are in the same region, it indicates they are close to each other and it is desirable to match them up. Further, since we do not have real-time location of taxis, we use the last known drop-off location as taxis' current location. This assumption is usually not true in large time scale as taxis are likely to leave the region and search for next passenger. To alleviate this issue, we only use trip records within a short time period, 6:00-6:30pm on Friday, June 3rd to predict matching of 6:00-6:30pm on Friday, June 10. Moreover, this is typically the rush hour in New York city and location of taxis are not likely to change dramatically during the period. Vendors' features are the distribution of associated taxis across 50 regions, i.e., $U \in \mathbb{R}^{50\times 2}$, passengers' features are simply the one-hot encoding of their current location, i.e, $V \in \mathbb{R}^{50\times 50}$. So the interaction matrix $A \in \mathbb{R}^{50\times 50}$. We set $\lambda = \lambda_u = \lambda_v = 1, \delta = 1\times 10^{-3}, K = 20$, for C_u, C_v, we randomly generate m and n points from $\mathcal{N}(\mathbf{0}, 5I_2)$ on plane and use their Euclidean distance matrix as C_u and C_v. Linear kernel $C_{ij} = 0.2\boldsymbol{u}_i^T A \boldsymbol{v}_j + 1$ is used for this experiment.

The comparison of the actual matching π_{new} and the predicted matching $\pi_{predicted}$ is shown in Figure 7. Visually speaking, we see that the predicted matching is able to capture the pattern of actual empirical matching and the prediction is fairly accurate. Quantitative result is also reported, measured in Kullback-Leibler divergence

$$\mathrm{KL}(\pi_{new} || \pi_{predicted}) = 0.1659.$$

6.4. Marriage data set

In this subsection, we illustrate the applicability of our model in suggesting new matching and it can make more accurate and realistic recommendations than conventional recommender systems do. Once the interaction between two sides of matching market is learned, one may use that to predict matching for new groups and make recommendations accordingly. We compare our RIOT with baseline random predictor model (Random), classical SVD model (Koren et al., 2009) and item-based collaborative filtering model (itemKNN) (Cremonesi et al., 2010), probabilistic matrix factorization model (PMF) (Mnih and Salakhutdinov, 2008) and the state-of-art factorization machine model (FM) (Rendle, 2012). To fit conventional recommender systems in our setting, one possible approach is simply treating each cell of matching matrix as rating and ignoring the underlying matching mechanism. In RIOT, we set $\lambda = \lambda_u = \lambda_v = 1$, relaxation parameter $\delta = 0.001$, inner iteration $K = 20$ and use polynomial kernel $k(\boldsymbol{x}, \boldsymbol{y}) = (0.2\boldsymbol{x}^T\boldsymbol{y} + 0.8)^2$. For C_u, C_v, we randomly generate m and n points from $\mathcal{N}(\mathbf{0}, 5I_2)$ on plane and use their Euclidean distance matrix as C_u and C_v.

We evaluate all models on Dutch Household Survey (DHS) data set [6] from 2005 to 2014 excluding 2008 (due to data field inconsistency). After data cleaning, the data set consists of 2475 pairs of couple. For each person we extract 11 features including education level, height, weight, health and 6 characteristic traits, namely irresponsible, accurate, ever-ready, disciplined, ordered, clumsy and detail-oriented. Education levels are first categorized into elementary, middle and high and then mapped linearly onto $[0, 1]$. Height and weight are normalized by dividing the largest height/weight. Health and characteristic

6. https://www.dhsdata.nl/site/users/login



Figure 6: Clusters of Taxi Trip Records

features are measured on 0-5 scale and rescaled onto $[0, 1]$. We use k-means algorithm to cluster men/women into $n_{\text{cluster}} = 50$ groups, respectively. We select each cluster center as representative. Performing clustering can better illustrate the applicability of our model in distributional setting and also helps reduce problem size. We train all models on training data set and measure error between predicted and test matching matrix by root mean

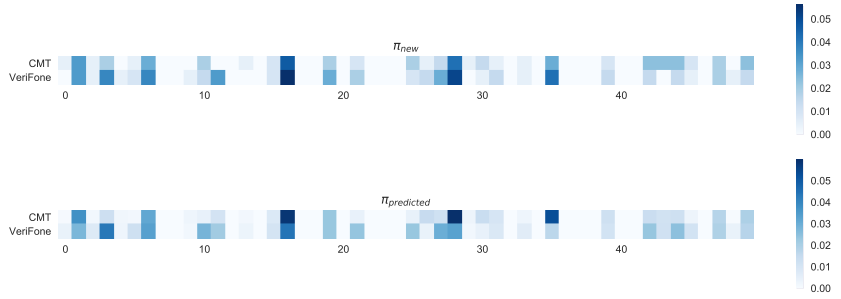


Figure 7: Comparison between actual matching π_{new} (top) and predicted matching $\pi_{predicted}$ (bottom) between 6:00-6:30pm on Friday, June 10th in New York city. Ticks of x-axis are labels of regions.

square error (RMSE) and mean absolute error (MAE) using 5-fold cross-validation. The result is shown in Table 1.

	Random	PMF	SVD	itemKNN	RIOT	FM
RMSE	54.5	8.4	29.9	2.4	**2.3**	3.6
MAE	36.6	2.0	16.8	1.6	**1.5**	2.8

Table 1: Average error of 5-fold cross-validation measured in RMSE and MAE ($\times 10^{-4}$)

In both measures, RIOT beats other conventional RS competitors. The comparison clearly shows that being able to take supply limitation into consideration and capture matching mechanism is of critical importance in suggest matching in such context and our proposed RIOT model can do a better job than conventional recommender systems do.

7. Conclusion

In this paper, we develop a novel, unified, data-driven inverse-optimal-transport-based matching framework RIOT which can learn adaptive, nonlinear interaction preference from noisy/incomplete empirical matching matrix in various matching contexts. The proposed RIOT is shown to be more robust than the state of the art IOT formulation and exhibits better performance in learning cost. Moreover, our framework can be extended to make recommendations based on predicted matching and outperforms conventional recommender systems in matching context.

In the future, our work can be continued in multiple ways. First, our model does batch prediction for a group of users and items and we would like to develop online algorithm to deal with streaming data and make matching suggestion for a previous unseen user/item in an online fashion. A recent method proposed by Perrot et al. (2016) that allows to

update the plan using out-of-sample data without recomputing might be useful. From business standpoint, we may study optimal pricing within our framework, i.e., how to set a reasonable price and adjust item distribution in a most profitable way (Azaria et al., 2013). In addition, we hope to combine impressive expressiveness of deep neural networks to further boost the performance of our proposed model.

Acknowledgments

We would like to acknowledge support for this project from the National Science Foundation grants CMMI-1745382, DMS-1620342, DMS-1818886, DMS-1830225 and DMS-1620345, the Office of Naval Research grant N00014-18-1-2852. The authors are grateful to anonymous reviewers and the action editor for their insightful comments and suggestions which helped to improve the manuscript significantly.

Appendix A. Proof of Lemma 3

Proof

$$f(\boldsymbol{a},\boldsymbol{b}) = \boldsymbol{x}^T A\boldsymbol{x} - 2\boldsymbol{f}^T\boldsymbol{x} + \|M\|_F^2$$

where $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{a}\\ \boldsymbol{b}\end{bmatrix}$, $\boldsymbol{f} = [(M\mathbf{1})^T, \mathbf{1}^T M]^T$ and $A = \begin{bmatrix} nI_{m\times m} & \mathbf{1}_m\mathbf{1}_n^T \\ \mathbf{1}_n\mathbf{1}_m^T & mI_{n\times n}\end{bmatrix}$. Note that A is a positive semi-definite matrix, the algebraic multiplicity of its 0 eigenvalue is 1 and $\text{null}(A) = \text{span}\{[\mathbf{1}_m^T, -\mathbf{1}_n^T]^T\}$. Moreover, $\boldsymbol{f} \perp \text{null}(A)$, hence the quadratic form $f(\boldsymbol{a},\boldsymbol{b})$ admits a minimum in $\text{null}(A)^\perp$ and it is straightforward to obtain

$$\begin{aligned}\min_{\boldsymbol{a},\boldsymbol{b}} f(\boldsymbol{a},\boldsymbol{b}) &= \min_{\boldsymbol{x}} \boldsymbol{x}^T A\boldsymbol{x} - 2\boldsymbol{f}^T\boldsymbol{x} + \|M\|_F^2 \\ &= \|M\|_F^2 - \boldsymbol{f}^T A^+ \boldsymbol{f}\end{aligned}$$

where A^+ is the Moore-Penrose inverse of matrix A. If M can not be written as $M = \boldsymbol{a}\mathbf{1}^T + \mathbf{1}\boldsymbol{b}^T$, then the minimum of $f(\boldsymbol{a},\boldsymbol{b})$ must be positive, hence

$$f(\boldsymbol{a},\boldsymbol{b}) \geq \min_{\boldsymbol{a},\boldsymbol{b}} f(\boldsymbol{a},\boldsymbol{b}) = \|M\|_F^2 - \boldsymbol{f}^T A^+ \boldsymbol{f} > 0$$

. ■

Appendix B. Extension of RIOT model to learn C_u and C_v jointly

In this section, we extend proposed RIOT model to settings where C_u and C_v are unknown and need to be learned jointly with the main cost matrix $C(A)$. Following the same derivation, we end up with an optimization problem almost identical to the one in equation (5), i.e.,

$$\min_{A,\boldsymbol{\mu}\in\Sigma_m,\boldsymbol{\nu}\in\Sigma_n,C_u,C_v} -\sum_{i=1}^{m}\sum_{j=1}^{n}\hat{\pi}_{ij}\log\pi_{ij}+\delta\big(d_{\lambda_u}(C_u,\boldsymbol{\mu},\hat{\boldsymbol{\mu}})+d_{\lambda_v}(C_v,\boldsymbol{\nu},\hat{\boldsymbol{\nu}})\big) \tag{13}$$

except for the fact that now we need to optimize two additional variables C_u and C_v. Genericly, inverse problems are usually not well-posed, in our cases, if no constraints imposed on C_u, C_v, one could trivially let, say $C_u = 0_{m\times m}, C_v = 0_{n\times n}$. To avoid such ill-posedness, we assume that $C_u \in \mathcal{M}^m \cap \Sigma^{m\times m}, C_v \in \mathcal{M}^n \cap \Sigma^{n\times n}$, where $\mathcal{M}^d$ is the cone of $d \times d$ distance matrix and Σ^d is $d-1$ simplex (see Section 3 for definition). Other regularization can also be explored.

By strong duality, we may convert equation (13) to its dual problem in a similar fashion as equation (6),

$$\min_{A,\boldsymbol{\mu},\boldsymbol{\nu},C_u,C_v}\max_{\boldsymbol{z},\boldsymbol{w}} -\sum_{i=1}^{m}\sum_{j=1}^{n}\hat{\pi}_{ij}\log\pi_{ij}+\delta\big(\langle\boldsymbol{z},\boldsymbol{\mu}\rangle+\langle\boldsymbol{z}^{C_u},\hat{\boldsymbol{\mu}}\rangle+\langle\boldsymbol{w},\boldsymbol{\nu}\rangle+\langle\boldsymbol{w}^{C_v},\hat{\boldsymbol{\nu}}\rangle\big)$$

where $z_j^{C_u}=\frac{1}{\lambda_u}\log\hat{r}_j-\frac{1}{\lambda_u}\log(\sum_{i=1}^m e^{\lambda_u(z_i-C_{uij})})$ and $w_j^{C_v}=\frac{1}{\lambda_v}\log\hat{c}_j-\frac{1}{\lambda_v}\log(\sum_{i=1}^n e^{\lambda_v(w_i-C_{vij})})$.

One way to solve equation (14), without too many changes of proposed algorithm in Section 5, is to rewrite it as

$$\min_{A,\boldsymbol{\mu},\boldsymbol{\nu}}\min_{C_u,C_v}\max_{\boldsymbol{z},\boldsymbol{w}} -\sum_{i=1}^{m}\sum_{j=1}^{n}\hat{\pi}_{ij}\log\pi_{ij}+\delta\big(\langle\boldsymbol{z},\boldsymbol{\mu}\rangle+\langle\boldsymbol{z}^{C_u},\hat{\boldsymbol{\mu}}\rangle+\langle\boldsymbol{w},\boldsymbol{\nu}\rangle+\langle\boldsymbol{w}^{C_v},\hat{\boldsymbol{\nu}}\rangle\big) \tag{14}$$

and alternatively update three groups of variables $(A,\boldsymbol{\mu},\boldsymbol{\nu})$, (C_u,C_v) and $(\boldsymbol{z},\boldsymbol{w})$.

B.1. Update $(A,\boldsymbol{\mu},\boldsymbol{\nu})$, with (C_u,C_v) and $(\boldsymbol{z},\boldsymbol{w})$ fixed

Once (C_u,C_v) and $(\boldsymbol{z},\boldsymbol{w})$ fixed, $\boldsymbol{z}^{C_u},\boldsymbol{w}^{C_v}$ are fixed as well, hence the optimization problem at this stage becomes

$$\min_{A,\boldsymbol{\mu},\boldsymbol{\nu}} -\sum_{i=1}^{m}\sum_{j=1}^{n}\hat{\pi}_{ij}\log\pi_{ij}+\delta\big(\langle\boldsymbol{z},\boldsymbol{\mu}\rangle+\langle\boldsymbol{w},\boldsymbol{\nu}\rangle\big)$$

where constants are omitted. This minimization problem is identical to that in subsection 5.1. Please see detailed update scheme for $(A,\boldsymbol{\mu},\boldsymbol{\nu})$ there.

B.2. Update (C_u,C_v), with $(A,\boldsymbol{\mu},\boldsymbol{\nu})$ and $(\boldsymbol{z},\boldsymbol{w})$ fixed

If $(A,\boldsymbol{\mu},\boldsymbol{\nu})$ and $(\boldsymbol{z},\boldsymbol{w})$ are fixed, π is also fixed as it is the regularized OT plan determined by parameters $(A,\boldsymbol{\mu},\boldsymbol{\nu})$, hence the optimization problem at this stage becomes

$$\min_{C_u\in\mathcal{M}^m\cap\Sigma^{m\times m},C_v\in\mathcal{M}^n\cap\Sigma^{n\times n}}\delta\big(\langle\boldsymbol{z}^{C_u},\hat{\boldsymbol{\mu}}\rangle+\langle\boldsymbol{w}^{C_v},\hat{\boldsymbol{\nu}}\rangle\big)$$

and can be further splitted into two independent optimization problems

$$\min_{C_u\in\mathcal{M}^m\cap\Sigma^{m\times m}}\delta\langle\boldsymbol{z}^{C_u},\hat{\boldsymbol{\mu}}\rangle,\qquad \min_{C_v\in\mathcal{M}^n\cap\Sigma^{n\times n}}\delta\langle\boldsymbol{w}^{C_v},\hat{\boldsymbol{\nu}}\rangle \tag{15}$$

which can be solved simultaneously.

Both $\mathcal{M}^d$ and $\Sigma^{d\times d}$ are convex sets, so is their intersection. Therefore we can perform projected gradient method to solve two separate minimization problems in equation (15).

B.3. Update $(\boldsymbol{z},\boldsymbol{w})$, with $(A,\boldsymbol{\mu},\boldsymbol{\nu})$ and (C_u,C_v) fixed

When $(A,\boldsymbol{\mu},\boldsymbol{\nu})$ and (C_u,C_v) are fixed, π is also fixed, we then only need to solve

$$\max_{\boldsymbol{z},\boldsymbol{w}}\langle\boldsymbol{z},\pi\mathbf{1}\rangle+\langle\boldsymbol{z}^{C_u},\hat{\boldsymbol{\mu}}\rangle+\langle\boldsymbol{w},\pi^T\mathbf{1}\rangle+\langle\boldsymbol{w}^{C_v},\hat{\boldsymbol{\nu}}\rangle$$

and one immediately recognizes this is equivalent to applying Sinkhorn-Knopp algorithm to compute $d_{\lambda_u}(C_u,\boldsymbol{\mu},\hat{\boldsymbol{\mu}})$ and $d_{\lambda_v}(C_v,\boldsymbol{\nu},\hat{\boldsymbol{\nu}})$.

To summarize, to jointly learn $C(A)$, C_u and C_v, we formulate an optimization problem similar to that in equation (5) and propose an alternating algorithm to solve it by alternately update $(A,\boldsymbol{\mu},\boldsymbol{\nu})$, (C_u,C_v) and $(\boldsymbol{z},\boldsymbol{w})$. Practically, the update of (C_u,C_v) requires expensive projection onto $\mathcal{M}^d\cap\Sigma^{d\times d}$, therefore we suggest learning C_u, C_v first and then using RIOT formulation to learn the main cost matrix $C(A)$, rather than learning three cost matrices simultaneously.

Appendix C. Hyper-parameter Tuning

There are many hyper-parameters in RIOT formulation, including regularization parameter $\lambda, \lambda_u, \lambda_v$, relaxation parameter δ, kernel function $k(\cdot,\cdot)$, cost matrix for two sides C_u, C_v and the number of iteration in inner loop K. With so many hyper-parameters, it would be very expensive to perform grid-search type tuning. Two parameters, relaxation parameter δ and kernel $k(\cdot,\cdot)$, turn out to be critical to the performance of RIOT. Therefore, we fixed other hyper-parameters (based on coarse-tuning) and (fine) tuned these two hyper-parameters.

- $\lambda, \lambda_u, \lambda_v$: as mentioned in main paper, one main motivation of introducing entropy regularization to optimal transport is computation. Too large λ usually causes numerical instability in Sinkhorn-Knopp algorithm, hence we set $\lambda = \lambda_u = \lambda_v = 1$ across all experiments.
- C_u, C_v: Randomly generate m and n points from $\mathcal{N}(\mathbf{0}, 5I_2)$ on plane and use their Euclidean distance matrix as C_u and C_v. This setting is consistent across all experiments.
- K: the number of iterations in inner loop should be large enough to ensure the iterative algorithms described in proposition 7 to converge. In all experiments, we observed the inner loop usually converges within 20 steps, hence we set $K = 20$.
- δ: Both small and large δ can regularize the problem quite well compared to IOT when sample size is small. When sample size is large, empirical matching matrix is usually accurate enough. In this case, large δ can introduce large bias whereas small δ has comparable performance of IOT. In practice, we suggest users to select small relaxation parameter. In our experiments, we found 10^{-3} is an appropriate order to work with.
- $k(\cdot,\cdot)$: we primarily work with polynomial kernels (including linear kernels) in the paper. Kernel function was chosen from $\{k(\boldsymbol{x},\boldsymbol{y}) = (\gamma \boldsymbol{x}^T\boldsymbol{y} + c_0)^d | \gamma \in \{0.05, 0.1, 0.2\}, c_0 \in \{0.5, 0.8, 1, 2\}, d \in \{1, 2, 3\}\}$

Appendix D. Table of Notations

$\boldsymbol{\mu}, \boldsymbol{\nu}$	marginal probability distributions
Σ_d	$d-1$-dimensional probability simplex
$U(\boldsymbol{\mu}, \boldsymbol{\nu})$	transport polytope
C	cost matrix
A	interaction matrix
C_{IOT}	cost matrix learned by IOT formulation
C_{RIOT}	cost matrix learned by RIOT formulation
λ	regularization parameter of entropy regularization
$d^\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu})$	regularized optimal transport distance
$\pi^\lambda(C, \boldsymbol{\mu}, \boldsymbol{\nu})$	regularized optimal transport plan
$\hat{\pi}$	empirical matching matrix
$\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\nu}}$	empirical marginal distributions
δ	relaxation parameter in RIOT formulation

References

Deepak Agarwal and Bee-Chung Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 19–28. ACM, 2009.

Vinti Agarwal and KK Bharadwaj. A collaborative filtering framework for friends recommendation in social networks based on interaction intensity and adaptive user similarity. *Social Network Analysis and Mining*, 3(3):359–379, 2013.

Shipra Agrawal, Morteza Zadimoghaddam, and Vahab Mirrokni. Proportional allocation: Simple, distributed, and diverse matching with high entropy. In *International Conference on Machine Learning*, pages 99–108, 2018.

Faez Ahmed, John P Dickerson, and Mark Fuge. Diverse weighted bipartite b-matching. *arXiv preprint arXiv:1702.07134*, 2017.

Ross Anderson, Itai Ashlagi, David Gamarnik, and Yash Kanoria. Efficient dynamic barter exchange. *Operations Research*, 65(6):1446–1459, 2017.

Itai Ashlagi, Maximilien Burq, Patrick Jaillet, and Vahideh Manshadi. On matching and thickness in heterogeneous dynamic markets. 2017.

Amos Azaria, Avinatan Hassidim, Sarit Kraus, Adi Eshkol, Ofer Weintraub, and Irit Netanely. Movie recommender system for profit maximization. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 121–128. ACM, 2013.

Gary S Becker. A theory of marriage: Part i. *Journal of Political economy*, 81(4):813–846, 1973.

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

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

J Bretagnolle and C Huber. Estimation des densités: risque minimax. *Probability Theory and Related Fields*, 47(2):119–137, 1979.

Justin Brickell, Inderjit S Dhillon, Suvrit Sra, and Joel A Tropp. The metric nearness problem. *SIAM Journal on Matrix Analysis and Applications*, 30(1):375–396, 2008.

Lukas Brozovsky and Vaclav Petricek. Recommender system for online dating service. *arXiv preprint cs/0703042*, 2007.

Guillaume Carlier and Ivar Ekeland. Matching for teams. *Economic theory*, 42(2):397–418, 2010.

Laurent Charlin, Richard Zemel, and Craig Boutilier. A framework for optimizing paper matching. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 86–95. AUAI Press, 2011.

Kwok-Wai Cheung and Lily F Tian. Learning user similarity and rating style for collaborative recommendation. *Information Retrieval*, 7(3-4):395–410, 2004.

Pierre-André Chiappori, Robert J McCann, and Lars P Nesheim. Hedonic price equilibria, stable matching, and optimal transport: equivalence, topology, and uniqueness. *Economic Theory*, 42(2):317–354, 2010.

Lenaic Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. Scaling algorithms for unbalanced transport problems. *arXiv preprint arXiv:1607.05816*, 2016.

Roberto Cominetti and Jaime San Martín. Asymptotic analysis of the exponential penalty trajectory in linear programming. *Mathematical Programming*, 67(1-3):169–187, 1994.

Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 39–46. ACM, 2010.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pages 2292–2300, 2013.

Marco Cuturi and David Avis. Ground metric learning. *Journal of Machine Learning Research*, 15(1):533–564, 2014.

Marco Cuturi and Arnaud Doucet. Fast computation of wasserstein barycenters. In *International Conference on Machine Learning*, pages 685–693, 2014.

H David. Wiring the labor market. *Journal of Economic Perspectives*, 15(1):25–40, 2001.

John P Dickerson and Tuomas Sandholm. Futurematch: Combining human value judgments and machine learning to match in dynamic environments. In *AAAI*, pages 622–628, 2015.

John P Dickerson, Ariel D Procaccia, and Tuomas Sandholm. Dynamic matching via weighted myopia with application to kidney exchange. In *AAAI*, 2012.

Arnaud Dupuy, Alfred Galichon, and Yifei Sun. Estimating matching affinity matrix under low-rank constraints. 2016.

Seth R Flaxman, Yu-Xiang Wang, and Alexander J Smola. Who supported obama in 2012?: Ecological inference through distribution regression. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 289–298. ACM, 2015.

David A Freedman, Stephen P Klein, Jerome Sacks, Charles A Smyth, and Charles G Everett. Ecological regression and voting rights. *Evaluation Review*, 15(6):673–711, 1991.

David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

Alfred Galichon and Bernard Salanié. Matching with trade-offs: Revealed preferences over competing characteristics. 2010.

Alfred Galichon and Bernard Salanié. Cupids invisible hand: Social surplus and identification in matching models. 2015.

Aude Genevay, Marco Cuturi, Gabriel Peyré, and Francis Bach. Stochastic optimization for large-scale optimal transport. In *Advances in Neural Information Processing Systems*, pages 3440–3448, 2016.

Aude Geneway, Gabriel Peyré, Marco Cuturi, et al. Learning generative models with sinkhorn divergences. Technical report, 2017.

Leo A Goodman. Ecological regressions and behavior of individuals. *American sociological review*, 1953.

Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Is that you? metric learning approaches for face identification. In *Computer Vision, 2009 IEEE 12th international conference on*, pages 498–505. IEEE, 2009.

Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 355–364. ACM, 2017.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 173–182. International World Wide Web Conferences Steering Committee, 2017.

Gunter J Hitsch, Ali Hortaçsu, and Dan Ariely. Matching and sorting in online dating. *American Economic Review*, 100(1):130–63, 2010.

Gary King. *A solution to the ecological inference problem: Reconstructing individual behavior from aggregate data*. Princeton University Press, 2013.

Tjalling C Koopmans and Martin Beckmann. Assignment problems and the location of economic activities. *Econometrica: journal of the Econometric Society*, pages 53–76, 1957.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8), 2009.

Charlotte Laclau, Ievgen Redko, Basarab Matei, Younes Bennani, and Vincent Brault. Co-clustering through optimal transport. *arXiv preprint arXiv:1705.06189*, 2017.

Aranyak Mehta et al. Online matching and ad allocation. *Foundations and Trends® in Theoretical Computer Science*, 8(4):265–368, 2013.

Paul Milgrom and Ilya Segal. Envelope theorems for arbitrary choice sets. *Econometrica*, 70(2):583–601, 2002.

Andriy Mnih and Ruslan R Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2008.

Boris Muzellec, Richard Nock, Giorgio Patrini, and Frank Nielsen. Tsallis regularized optimal transport and ecological inference. In *AAAI*, pages 2387–2393, 2017.

Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, pages 663–670, 2000.

Hieu V Nguyen and Li Bai. Cosine similarity metric learning for face verification. In *Asian conference on computer vision*, pages 709–720. Springer, 2010.

Nicolas Papadakis. *Optimal Transport for Image Processing*. PhD thesis, Université de Bordeaux, 2015.

Ofir Pele and Michael Werman. Fast and robust earth mover's distances. In *Computer vision, 2009 IEEE 12th international conference on*, pages 460–467. IEEE, 2009.

Michaël Perrot, Nicolas Courty, Rémi Flamary, and Amaury Habrard. Mapping estimation for discrete optimal transport. In *Advances in Neural Information Processing Systems*, pages 4197–4205, 2016.

Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport. Technical report, 2017.

Luiz Pizzato, Tomasz Rej, Joshua Akehurst, Irena Koprinska, Kalina Yacef, and Judy Kay. Recommending people to people: the nature of reciprocal recommenders with a case study in online dating. *User Modeling and User-Adapted Interaction*, 23(5):447–488, 2013.

Steffen Rendle. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 995–1000. IEEE, 2010.

Steffen Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.

Antoine Rolet, Marco Cuturi, and Gabriel Peyré. Fast dictionary learning with a smoothed wasserstein loss. In *Artificial Intelligence and Statistics*, pages 630–638, 2016.

Alvin E Roth and Marilda Sotomayor. The college admissions problem revisited. *Econometrica: Journal of the Econometric Society*, pages 559–570, 1989.

Alvin E Roth and Marilda A Oliveira Sotomayor. *Two-sided matching: A study in game-theoretic modeling and analysis*. Number 18. Cambridge university press, 1992.

Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th international conference on Machine learning*, pages 880–887. ACM, 2008.

Bernhard Schölkopf. The kernel trick for distances. In *Advances in neural information processing systems*, pages 301–307, 2001.

Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Fan Wang and Leonidas J Guibas. Supervised earth movers distance learning and its computer vision applications. In *European Conference on Computer Vision*, pages 442–455. Springer, 2012.

Jon Wellner et al. *Weak convergence and empirical processes: with applications to statistics*. Springer Science & Business Media, 2013.

Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 521–528, 2003.

Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. Task matching in crowdsourcing. In *Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing*, pages 409–412. IEEE, 2011.

Journal of Machine Learning Research 20 (2019) 1-8 Submitted 10/18; Published 3/19

Tight Lower Bounds on the VC-dimension of Geometric Set Systems

Mónika Csikós monika.csikos@esiee.fr
Nabil H. Mustafa mustafan@esiee.fr
Université Paris-Est, LIGM, Equipe A3SI, ESIEE Paris,
Cité Descartes - 2 boulevard Blaise Pascal,
93162 Noisy-le-Grand Cedex, France.

Andrey Kupavskii kupavskii@yandex.ru
Laboratory of Advanced Combinatorics and Networks Applications,
Moscow Institute of Physics and Technology,
9, Institutskiy per., Dolgoprudniy, 141701, Russia and
Mathematical Institute, University of Oxford,
Woodstock Rd, Oxford OX2 6GG, UK

Editor: Alexander Rakhlin

Abstract

The VC-dimension of a set system is a way to capture its complexity and has been a key parameter studied extensively in machine learning and geometry communities. In this paper, we resolve two longstanding open problems on bounding the VC-dimension of two fundamental set systems: k-fold unions/intersections of half-spaces and the simplices set system. Among other implications, it settles an open question in machine learning that was first studied in the foundational paper of Blumer et al. (1989) as well as by Eisenstat and Angluin (2007) and Johnson (2008).

Keywords: VC-dimension, union of concepts, intersection of concepts, combinatorial problems, PAC learning

1. Introduction

Let $(X, \mathcal{R})$ be a *set system*, where X is a set of elements and $\mathcal{R}$ is a set of subsets of X. In the theory of learning, the elements of $\mathcal{R}$ are also called *concepts*, and $\mathcal{R}$ is called a *concept class* on X. For any integer $k \geq 2$, define the *k-fold union* of $\mathcal{R}$ as the set system induced on X by the ranges

$$\mathcal{R}^{k\cup} = \{R_1 \cup \cdots \cup R_k \colon R_1, \ldots, R_k \in \mathcal{R}\}.$$

Similarly, one can define the *k-fold intersection* of $\mathcal{R}$, denoted by $\mathcal{R}^{k\cap}$, as the set system consisting of all subsets derived from the common intersection of at most k sets of $\mathcal{R}$. Note that as the subsets $R_1, \ldots, R_k$ need not necessarily be distinct, we have $\mathcal{R} \subseteq \mathcal{R}^{k\cup}$ and $\mathcal{R} \subseteq \mathcal{R}^{k\cap}$. Analogously, the *$k$-fold symmetric difference* of $\mathcal{R}$ is defined as $\mathcal{R}^{k\oplus} = \{R_1 \oplus \cdots \oplus R_k \colon R_1, \ldots, R_k \in \mathcal{R}\}$, where $R_1 \oplus \cdots \oplus R_k$ is the set of those elements that are contained in an odd number of $R_1, \ldots, R_k$.

One of the fundamental measures of complexity of a set system is its *Vapnik-Chervonenkis dimension*, or in short, *VC-dimension*. Given a set system $(X, \mathcal{R})$, for any set $Y \subseteq X$, define the *projection* of $\mathcal{R}$ onto Y as

$$\mathcal{R}|_Y = \{Y \cap R \colon R \in \mathcal{R}\}.$$

We say that $\mathcal{R}$ *shatters* Y if $|\mathcal{R}|_Y| = 2^{|Y|}$; in other words, any subset of Y can be derived as the intersection of Y with an element of $\mathcal{R}$. The *VC-dimension* of $\mathcal{R}$, denoted by VC-dim$(\mathcal{R})$, is the size of the largest subset of X that can be shattered by $\mathcal{R}$. Originally introduced in statistical learning by Vapnik and Chervonenkis (1971), it has turned out to be a key parameter in several areas, including learning theory, combinatorics, and computational geometry.

1.1. Learning Theory.

In learning theory, the VC-dimension of a concept class measures the difficulty of learning a concept of the class. The foundational paper of Blumer et al. (1989) states that "the essential condition for distribution-free learnability is finiteness of the Vapnik-Chervonenkis dimension". Among their results, they prove the following theorem.

Theorem A (Blumer et al. 1989) *Let $(X, \mathcal{R})$ be a set system and k be any positive integer. Then*

$$\text{VC-dim}\left(\mathcal{R}^{k\cup}\right) = O\Big(\text{VC-dim}\left(\mathcal{R}\right) \cdot k \log k\Big),$$

$$\text{VC-dim}\left(\mathcal{R}^{k\cap}\right) = O\Big(\text{VC-dim}\left(\mathcal{R}\right) \cdot k \log k\Big).$$

Moreover, there are set systems such that $\text{VC-dim}\left(\mathcal{R}^{k\cup}\right) = \Omega\left(\text{VC-dim}\left(\mathcal{R}\right) \cdot k\right)$ *and* $\text{VC-dim}\left(\mathcal{R}^{k\cap}\right) = \Omega\left(\text{VC-dim}\left(\mathcal{R}\right) \cdot k\right)$.

Remark. The upper bound holds in a more general setting: for any fixed set-theoretic expression $F(R_1, R_2, \ldots, R_k)$ (consisting of operations of set union, intersection, and difference) and range set $\mathcal{R}^{k*} := \{F(R_1, \ldots, R_k) : R_1, \ldots, R_k \in \mathcal{R}\}$, we have VC-dim$(\mathcal{R}^{k*}) = O($VC-dim $(\mathcal{R}) \cdot k \log k)$ (see eg. Matoušek, 2002, chap. 10). In particular, $\text{VC-dim}\left(\mathcal{R}^{k\oplus}\right) = O\left(\text{VC-dim}\left(\mathcal{R}\right) \cdot k \log k\right)$.

They also considered the question of whether the upper bounds of Theorem A are tight in the most basic geometric case when $X \subseteq \mathbb{R}^d$ is a set of points and $\mathcal{R}$ is the projection of the family of all half-spaces of $\mathbb{R}^d$ onto X. They proved that the VC-dimension of the k-fold union of half-spaces in two dimensions is exactly $2k+1$. For general dimensions $d \geq 3$, they upper-bound the VC-dimension of the k-fold union of half-spaces by $O(d \cdot k \log k)$. This follows from Theorem A together with the fact that the VC-dimension of the set system induced by half-spaces in $\mathbb{R}^d$ is $d+1$. The same upper bound holds for the VC-dimension of the k-fold intersection of half-spaces in $\mathbb{R}^d$. Later Dobkin and Gunopulos (1995) showed that the VC-dimension of the k-fold union of half-spaces in $\mathbb{R}^3$ is upper-bounded by $4k$.

Eisenstat and Angluin (2007) proved, by giving a probabilistic construction of an abstract set system, that the upper bound of Theorem A is asymptotically tight if $\text{VC-dim}\left(\mathcal{R}\right) \geq 5$ and that for $\text{VC-dim}\left(\mathcal{R}\right) = 1$, an upper bound of k holds which is tight. A few years later, Eisenstat (2009) filled the gap by showing that there exists a set system $(X, \mathcal{R})$ of VC-dimension 2 such that $\text{VC-dim}\left(\mathcal{R}^{k\cup}\right) = \Omega\left(\text{VC-dim}\left(\mathcal{R}\right) \cdot k \log k\right)$.

For $d \geq 4$, the current best upper-bound for the k-fold union and the k-fold intersection of half-spaces in $\mathbb{R}^d$ is still the one given by Theorem A almost 30 years ago, while the lower-bound has remained $\Omega\left(\text{VC-dim}(\mathcal{R}) \cdot k\right)$. We refer the reader to the PhD thesis of Johnson (2008) for a summary of the bounds on VC-dimensions of these basic combinatorial and geometric set systems.

1.2. Computational Geometry.

The following set system is fundamental in computational geometry. Given a set $\mathcal{H}$ of hyperplanes in $\mathbb{R}^d$, define

$$\Delta\left(\mathcal{H}\right) = \Big\{\mathcal{H}' \subseteq \mathcal{H} \colon \exists \text{ an open } d\text{-dimensional simplex } S \text{ in } \mathbb{R}^d \text{ such that } H \in \mathcal{H} \text{ intersects } S \text{ if and only if } H \in \mathcal{H}'\Big\}.$$

©2019 Mónika Csikós, Andrey Kupavskii, and Nabil H. Mustafa.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-719.html.

Its importance derives from the fact that it is the set system underlying the construction of *cuttings* via random sampling (see Chazelle and Friedman 1990 as well as the recent survey of Mustafa and Varadarajan 2017, to appear). Cuttings are the key tool for fast point-location algorithms and were studied in detail recently by Ezra et al. (2017). They provided the best bounds so far for the VC-dimension of $\Delta(\mathcal{H})$.

Lemma B (Ezra et al. 2017) *For $d \geq 9$, we have*

$$d\,(d+1) \leq \text{VC-dim}\left(\Delta\left(\mathcal{H}\right)\right) \leq 5 \cdot d^2 \log d.$$

2. Our Results

For some time now, it has generally been expected that $\text{VC-dim}\left(\mathcal{R}^{k\cup}\right) = \text{VC-dim}\left(\mathcal{R}^{k\cap}\right) = O\left(dk\right)$ for the k-fold unions and intersections of half-spaces. This upper-bound indeed holds for a related notion: the *primal shattering dimension* of the k-fold unions and intersections of half-spaces is $O(dk)$. In fact, as it was pointed out by Bachem (2018), several papers in learning theory assume the same for VC-dimension. Likewise for computational geometry literature: for example, the coreset size bounds in the constructions of Feldman and Langberg (2011), Balcan et al. (2013), and Lucic et al. (2016) would require an additional $\log k$ factor in the coreset size—if the upper-bound of Theorem A was tight for the k-fold intersection of half-spaces.

In this paper, we resolve the question of VC-dimension for the above two set systems. Our proofs are short and we make an effort to keep them self-contained.

We show an optimal lower-bound on the VC-dimension of the k-fold union, the k-fold intersection, and the k-fold symmetric difference of half-spaces in $\mathbb{R}^d$, matching the $O(d \cdot k \log k)$ upper bound of Theorem A, and thus settling affirmatively one of the main open questions studied by Eisenstat and Angluin (2007), Johnson (2008), and Eisenstat (2009).

Theorem 1 *Let k be a given positive integer and $d \geq 4$ be an integer. Then there exists a set $\mathcal{P}$ of points in $\mathbb{R}^d$ such that the set system $\mathcal{R}$ induced on $\mathcal{P}$ by half-spaces satisfies*

$$\text{a)} \quad \text{VC-dim}\left(\mathcal{R}^{k\cup}\right) = \Omega\Big(\text{VC-dim}(\mathcal{R}) \cdot k \log k\Big) = \Omega\Big(d \cdot k \log k\Big),$$

$$\text{b)} \quad \text{VC-dim}\left(\mathcal{R}^{k\cap}\right) = \Omega\Big(\text{VC-dim}(\mathcal{R}) \cdot k \log k\Big) = \Omega\Big(d \cdot k \log k\Big),$$

$$\text{c)} \quad \text{VC-dim}\left(\mathcal{R}^{k\oplus}\right) = \Omega\Big(\text{VC-dim}(\mathcal{R}) \cdot k \log k\Big) = \Omega\Big(d \cdot k \log k\Big).$$

Remark 1. This statement also provides a non-probabilistic proof of the lower-bound of Eisenstat and Angluin (2007).

Remark 2. Note that a set of points is shattered by (the k-fold union of) closed half-spaces if and only if it is shattered by (the k-fold union of) open half-spaces. Thus Theorem 1 also holds for open half-spaces.

Remark 3. Observe that if $\overline{\mathcal{R}} := \{\mathbb{R}^d \setminus R\colon\ R \in \mathcal{R}\}$, then $\text{VC-dim}(\overline{\mathcal{R}}) = \text{VC-dim}(\mathcal{R})$ and

$$\text{VC-dim}\left(\mathcal{R}^{k\cap}\right) = \text{VC-dim}\left(\overline{\mathcal{R}^{k\cap}}\right) = \text{VC-dim}\left(\overline{\mathcal{R}}^{k\cup}\right).$$

holds by the De Morgan laws. Since for half-spaces $\overline{\mathcal{R}} = \mathcal{R}$, part a) of Theorem 1 implies part b).

Using Theorem 1, we show an asymptotically optimal bound on the VC-dimension of $\Delta(\mathcal{H})$, improving the bound of Ezra et al. (2017) and resolving a question that was studied in the computational geometry community starting in the 1980s.

Theorem 2 *Let $d \geq 4$ be a given integer. Then there exists a set $\mathcal{H}$ of $\Omega\left(d^2 \log d\right)$ hyperplanes in $\mathbb{R}^d$ such that $\Delta\left(\mathcal{H}\right) = 2^{\mathcal{H}}$, that is, we have*

$$\text{VC-dim}\left(\Delta\left(\mathcal{H}\right)\right) = \Omega\left(d^2 \log d\right).$$

Remark. In fact, we prove a more general result bounding the VC-dimension of the set system induced by intersection of hyperplanes with k-dimensional simplices in $\mathbb{R}^d$. See Section 4 for details.

Organization. Section 3 contains the proof of Theorem 1 and Section 4 contains the proof of Theorem 2.

3. Proof of Theorem 1.

We prove the theorem for d even, starting from $d = 4$. The asymptotic lower-bound for odd values of d follows from the one in $\mathbb{R}^{d-1}$. The starting point of the proof is the following lemma.

Lemma 3 (Kupavskii et al., 2016, Lemma 2) *Let $n, d' \geq 2$ be integers. Then there exists a set $\mathcal{B}_{n,d'}$ of axis-parallel boxes in $\mathbb{R}^{d'}$, with $|\mathcal{B}_{n,d'}| = (d'-1)\,(n+1)\,2^{n-2}$, such that for any subset $\mathcal{S} \subseteq \mathcal{B}_{n,d'}$, there exists a set $\mathcal{Q}$ of points in $\mathbb{R}^{d'}$ such that $|\mathcal{Q}| = 2^{n-1}$ and*

(i) $|\mathcal{Q} \cap B| = 1$ for any $B \in \mathcal{B}_{n,d'} \setminus \mathcal{S}$, and

(ii) $\mathcal{Q} \cap B = \emptyset$ for any $B \in \mathcal{S}$.

Remark. In Kupavskii et al. (2016) this lemma is stated in a weaker form, however the above stronger statement is implicit in their proof.

Let $d' = \left\lfloor \frac{d}{2} \right\rfloor$ and apply Lemma 3 with $n = \lfloor \log k \rfloor + 1$ in $\mathbb{R}^{d'}$ to get a set $\mathcal{B} = \mathcal{B}_{n,d'}$ of boxes in $\mathbb{R}^{d'}$. By translation, we can assume that all coordinates of points lying in each box in $\mathcal{B}$ are positive. Now we will construct the following mappings:

$$\begin{array}{rcl} \text{set of boxes } \mathcal{B} \text{ in } \mathbb{R}^{d/2} & \xrightarrow{\ \pi\ } & \text{points in } \mathbb{R}^d, \\ \text{points in } \mathbb{R}^{d/2} & \xrightarrow{\ \beta\ } & \text{boxes in } \mathbb{R}^d \xrightarrow{\ \gamma\ } \text{half-spaces in } \mathbb{R}^d. \end{array}$$

We will then prove the key property of these mappings, that for any $q \in \mathbb{R}^{d/2}$ and $B \in \mathcal{B}$

$$q \in B \quad \overset{(1)}{\Longleftrightarrow} \quad \pi(B) \in \beta(q) \quad \overset{(2)}{\Longleftrightarrow} \quad \pi(B) \in \gamma(\beta(q)).$$

We first define mappings π and β, with $\pi : \mathcal{B} \to \mathbb{R}^d$ and with β mapping points in $\mathbb{R}^{d'}$ to axis-parallel boxes in $\mathbb{R}^d$, such that for any $B \in \mathcal{B}$ and $q \in \mathbb{R}^{d'}$, we have

$$q \in B \iff \pi(B) \in \beta(q). \tag{1}$$

- Let $B \in \mathcal{B}$ be defined as the product of d' intervals:

  $$B = [x_1, x_1'] \times [x_2, x_2'] \times \cdots \times [x_{d'}, x_{d'}'], \quad \text{with} \quad x_i, x_i' > 0 \text{ for each } i \in [d'].$$

  Then π maps B in $\mathbb{R}^{d'}$ to the following point in $\mathbb{R}^d$ (see Pach and Tardos, 2013)

  $$\pi(B) = \left(x_1, \frac{1}{x_1'}, x_2, \frac{1}{x_2'}, \ldots, x_{d'}, \frac{1}{x_{d'}'}\right) \in \mathbb{R}^d,$$

  and we let $\pi(\mathcal{B}) := \{\pi(B)\colon B \in \mathcal{B}\}$.

- For any point $q = (q_1, q_2, \ldots, q_{d'}) \in \mathbb{R}^{d'}$, define $\beta(q)$ to be the box

$$\beta(q) = [0, q_1] \times \left[0, \frac{1}{q_1}\right] \times \cdots \times [0, q_{d'}] \times \left[0, \frac{1}{q_{d'}}\right] \subset \mathbb{R}^d.$$

Proposition 4 *The mappings π and β satisfy* (1).

Proof Let $q = (q_1, q_2, \ldots, q_{d'})$ be a point in $\mathbb{R}^{d'}$ and $B = [x_1, x_1'] \times [x_2, x_2'] \times \cdots \times [x_{d'}, x_{d'}']$ be an axis-parallel box in $\mathbb{R}^{d'}$. Then $q \in B$ if and only if $x_i \le q_i \le x_i'$ for all $i \in [d']$.

On the other hand, $\pi(B)$ lies in $\beta(q)$ if and only if $0 \le x_i \le q_i$ and $0 \le 1/x_i' \le 1/q_i$ for each $i \in [d']$—or equivalently, $0 \le x_i \le q_i$ and $q_i \le x_i'$ for each $i \in [d']$.

Note that these two conditions are exactly the same, implying (1). ∎

Recall that $\mathcal{B} = \mathcal{B}_{n,d'}$ is the set of boxes provided by Lemma 3 with parameters $d' = \lfloor \frac{d}{2} \rfloor$ and $n = \lfloor \log k \rfloor + 1$. Thus we have

$$|\mathcal{B}| = \left(\left\lfloor \frac{d}{2} \right\rfloor - 1 \right) \lfloor \log k \rfloor 2^{\lfloor \log k \rfloor - 1}.$$

Proposition 5 ("Lifted dual version" of Lemma 3) *$\pi(\mathcal{B})$ is a set of $\left(\lfloor \frac{d}{2} \rfloor - 1 \right) \lfloor \log k \rfloor 2^{\lfloor \log k \rfloor - 1}$ points in $\mathbb{R}^d$ such that for any subset $\mathcal{T} \subseteq \pi(\mathcal{B})$, there is a set $\mathcal{Q}^*$ of at most k axis-parallel boxes in $\mathbb{R}^d$ such that*

(i) each point of $\mathcal{T}$ is contained in exactly one box in $\mathcal{Q}^*$, and

(ii) no point of $\pi(\mathcal{B}) \setminus \mathcal{T}$ is contained in any box of $\mathcal{Q}^*$.

In particular, $\pi(\mathcal{B})$ is shattered by the set system induced by the k-fold union of axis-parallel boxes in $\mathbb{R}^d$ and also shattered by the set system induced by the k-fold symmetric difference of axis-parallel boxes in $\mathbb{R}^d$.

Proof Let $\mathcal{S} = \left\{ \pi^{-1}(p) \ : \ p \in \pi(\mathcal{B}) \setminus \mathcal{T} \right\}$. By Lemma 3, there is a set $\mathcal{Q}$ of at most k points in $\mathbb{R}^{d'}$ such that (i), (ii) of Lemma 3 hold for $\mathcal{S}$ and $\mathcal{Q}$. Letting $\mathcal{Q}^* = \{\beta(q) : q \in \mathcal{Q}\}$, the claim follows from (1). ∎

Next we define the function $\gamma(\cdot)$ mapping boxes in $\mathbb{R}^d$ to half-spaces in $\mathbb{R}^d$ such that for any point $p \in \pi(\mathcal{B})$ and box $B = \beta(q)$, we have

$$p \in B \iff p \in \gamma(B). \tag{2}$$

For every $i \in [d]$, let $0 < \alpha_{i,1} < \alpha_{i,2} < \ldots$ denote the sequence of distinct values of the x_i-coordinates of the elements of $\pi(\mathcal{B})$. Every such sequence has length at most $|\pi(\mathcal{B})|$. By re-scaling the coordinates, we can assume that

$$\text{for each } i \in [d] \text{ and } j \le |\pi(\mathcal{B})|, \qquad \frac{\alpha_{i,j+1}}{\alpha_{i,j}} > d. \tag{3}$$

Denote the resulting point set by $\mathcal{P}$. Note that scaling along each coordinate does not change incidences with respect to axis-parallel boxes, thus Proposition 5 still holds if we replace $\pi(\mathcal{B})$ by $\mathcal{P}$ and that $|\mathcal{P}| = |\pi(\mathcal{B})| = |\mathcal{B}|$.

We claim that $\mathcal{P}$ is shattered by the set system induced by the k-fold union of half-spaces in $\mathbb{R}^d$ and also shattered by the set system induced by the k-fold symmetric difference of half-spaces in $\mathbb{R}^d$. To see that, let $\mathcal{P}'$ be any subset of $\mathcal{P}$. Let $\mathcal{Q}^*$ be the set of axis-parallel boxes corresponding to $\mathcal{P}'$ provided by Proposition 5.

For each box $B \in \mathcal{Q}^*$, we can re-scale B if necessary, without changing its intersection with $\mathcal{P}$ so that B is of the form

$$B = [0, b_1] \times [0, b_2] \times \cdots \times [0, b_d],$$

where each b_i is equal to α_{i,j_i}, for a suitable j_i. Now for each box $B \in \mathcal{Q}^*$, we define a half-space $\gamma(B)$ as the set of points $(x_1, \ldots, x_d) \in \mathbb{R}^d$ satisfying

$$\frac{x_1}{b_1} + \frac{x_2}{b_2} + \cdots + \frac{x_d}{b_d} \le d. \tag{4}$$

Clearly for any point $(x_1, \ldots, x_d)$ contained in the box B, we have $x_i \in [0, b_i]$ and thus each term on the left-hand side of the Equation (4) is at most 1. This implies that $B \subset \gamma(B)$ and so any point in the box B lies in the half-space $\gamma(B)$.

On the other hand, for any point $(x_1, \ldots, x_d) \in \mathcal{P} \setminus B$, there exists an index $i \in [d]$ such that $x_i > b_i$. By (3), we have that $x_i/b_i > d$, and thus $(x_1, \ldots, x_d)$ cannot lie in the half-space $\gamma(B)$.

Consider the set of at most k half-spaces defined as $\mathcal{H} = \{\gamma(B) : B \in \mathcal{Q}^*\}$. Now by Proposition 5 and (2), we have

(i) each point of $\mathcal{P}'$ is contained in exactly one half-space in $\mathcal{H}$, and

(ii) no point of $\mathcal{P} \setminus \mathcal{P}'$ is contained in any half-space of $\mathcal{H}$.

In other words, the union as well as the symmetric difference of the half-spaces in $\mathcal{H}$ contains precisely the set $\mathcal{P}'$. As this is true for any $\mathcal{P}' \subseteq \mathcal{P}$, the k-fold union of half-spaces in $\mathbb{R}^d$ shatters $\mathcal{P}$ and the same holds for the k-fold symmetric difference of half-spaces in $\mathbb{R}^d$. Finally, we have

$$|\mathcal{P}| = |\mathcal{B}| = \left(\left\lfloor \frac{d}{2} \right\rfloor - 1 \right) \lfloor \log k \rfloor 2^{\lfloor \log k \rfloor - 1} = \Omega\left(d \cdot k \log k\right),$$

as desired. ∎

4. Proof of Theorem 2.

Given a set $\mathcal{H}$ of hyperplanes in $\mathbb{R}^d$, define the set system

$$\Delta_k(\mathcal{H}) = \Big\{ \mathcal{H}' \subseteq \mathcal{H} \colon \exists \text{ an open } k\text{-dimensional simplex } S \text{ in } \mathbb{R}^d \text{ such that } H \in \mathcal{H}' \text{ if and only if } H \text{ intersects } S \Big\}.$$

We prove the following more general theorem from which Theorem 2 follows immediately by setting $k = d$.

Theorem 6 *For any integer $d \ge 4$ and $k \le d$, there exists a set $\mathcal{H}$ of $\Omega(dk \log k)$ hyperplanes in $\mathbb{R}^d$ such that $|\Delta_k(\mathcal{H})| = 2^{|\mathcal{H}|}$, that is, we have*

$$\text{VC-dim}\left(\Delta_k(\mathcal{H})\right) = \Omega\left(d \cdot k \log k\right).$$

Proof Apply Theorem 1 to get a set $\mathcal{P}$ of $\Omega(dk \log k)$ points in $\mathbb{R}^d$ that is shattered by the k-fold union of open half-spaces. Using point-hyperplane duality (see eg. Matoušek, 2002), map each point $p \in \mathcal{P}$ to a hyperplane $\alpha(p)$ by

$$p = (p_1, \ldots, p_d) \quad \longmapsto \quad \alpha(p) := \left\{ (x_1, \ldots, x_d) : p_1 x_1 + p_2 x_2 + \cdots + p_{d-1} x_{d-1} - x_d = -p_d \right\}.$$

Our desired set $\mathcal{H}$ of hyperplanes will simply be

$$\mathcal{H} = \left\{ \alpha(p) \colon p \in \mathcal{P} \right\}.$$

It is easy to check that the mapping α is injective and thus $|\mathcal{H}| = |\mathcal{P}| = \Omega(dk \log k)$. We claim that $\mathcal{H}$ is shattered by the set system induced by open k-dimensional simplices; in other words, for any $\mathcal{H}' \subseteq \mathcal{H}$, there exists a k-dimensional simplex S such that the interior of S intersects each hyperplane of $\mathcal{H}'$, and no hyperplane of $\mathcal{H} \setminus \mathcal{H}'$.

Fix any $\mathcal{H}' \subseteq \mathcal{H}$ and let $\mathcal{P}' = \alpha^{-1}(\mathcal{H}')$ be the corresponding points of $\mathcal{P}$. Then there exists a set $\mathfrak{H}(\mathcal{P}')$ of k open half-spaces whose union contains all points in $\mathcal{P}'$ and no point in $\mathcal{P} \setminus \mathcal{P}'$. From Equation (4), it follows that each half-space in $\mathfrak{H}(P')$ is of the form

$$\frac{x_1}{b_1} + \frac{x_2}{b_2} + \cdots + \frac{x_d}{b_d} < d,$$

where $b_1, \ldots, b_d$ are positive reals. Map each half-space $H \in \mathfrak{H}(\mathcal{P}')$ to the point $\delta(H)$, given by

$$H = \left\{(x_1, \ldots, x_d) : \frac{x_1}{b_1} + \frac{x_2}{b_2} + \cdots + \frac{x_d}{b_d} < d\right\} \quad \longmapsto \quad \delta(H) := \left(\frac{b_d}{b_1}, \ldots, \frac{b_d}{b_{d-1}}, d \cdot b_d\right).$$

It is easy to verify that for a point $p \in \mathbb{R}^d$ and the half-space H, we have

$$\begin{aligned} p \in H &\iff \frac{p_1}{b_1} + \frac{p_2}{b_2} + \cdots + \frac{p_d}{b_d} < d \\ &\iff p_1 \cdot \frac{b_d}{b_1} + p_2 \cdot \frac{b_d}{b_2} + \cdots + p_{d-1} \cdot \frac{b_d}{b_{d-1}} + p_d < d \cdot b_d \\ &\iff \text{point } \left(\frac{b_d}{b_1}, \frac{b_d}{b_2}, \ldots, \frac{b_d}{b_{d-1}}, d \cdot b_d\right) \text{ lies strictly above the hyperplane} \\ &\qquad\qquad p_1 x_1 + p_2 x_2 + \cdots + p_{d-1} x_{d-1} - x_d = -p_d \\ &\iff \text{the point } \delta(H) \text{ lies strictly above the hyperplane } \alpha(p). \end{aligned} \tag{5}$$

Here we crucially needed the fact that all half-spaces in $\mathfrak{H}(\mathcal{P}')$ are downward facing, that is, each half-space in $\mathfrak{H}(\mathcal{P}')$ contains the origin, which lies below (with respect to the x_d-coordinate) its bounding hyperplane.

Now consider the k open half-spaces in $\mathfrak{H}(\mathcal{P}')$ and let

$$\Delta' = \{\delta(H) \colon H \in \mathfrak{H}(\mathcal{P}')\}$$

be k points in $\mathbb{R}^d$. From the relation (5), it follows that

- As each point $p \in \mathcal{P}'$ lies in *some* half-space $H \in \mathfrak{H}(\mathcal{P}')$, the point $\delta(H)$ lies strictly above the hyperplane $\alpha(p)$ in $\mathcal{H}$—or equivalently, the hyperplane $\alpha(p)$ has at least one of the k points in the set Δ' lying strictly above it.
- For each point $p \in \mathcal{P} \setminus \mathcal{P}'$, all the k points in Δ' lie on or below the hyperplane $\alpha(p) \in \mathcal{H}$.

Then, by the above discussion, $H \in \mathcal{H}'$ if and only if one of these is true:

1. H intersects the interior of $\texttt{conv}\,(\Delta')$ and so at least one vertex of Δ' lies strictly above H, or
2. H does not intersect $\texttt{conv}\,(\Delta')$, but then *all* vertices of Δ' lie strictly above H.

Finally consider the k-dimensional simplex

$$S = \texttt{conv}\left(\Delta' \bigcup (0, \ldots, 0, -\infty)\right).$$

Clearly, a hyperplane $H \in \mathcal{H}$ intersects the interior of S if and only if $H \in \mathcal{H}'$. Note that the point $(0, \ldots, 0, -\infty)$ can be any point $(0, \ldots, 0, t)$ for a small-enough value of $t \in \mathbb{R}$. This concludes the proof. ∎

Acknowledgements. The research of Mónika Csikós and Nabil H. Mustafa was supported by the grant ANR SAGA (JCJC-14-CE25-0016-01). The research of Andrey Kupavskii was partially supported by the EPSRC grant no. EP/N019504/1 and by the Council for the Support of Leading Scientific Schools of the President of the Russian Federation (grant no. N.Sh.-6760.2018.1).
We thank Yaqiao Li for his valuable comments, which improved the statement of Theorem 1.

References

O. Bachem. *Sampling for Large-Scale Clustering*. PhD thesis, ETH Zürich, 2018.

M.-F. Balcan, S. Ehrlich, and Y. Liang. Distributed k-means and k-median clustering on general topologies. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2013.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, 36(4):929–965, October 1989.

B. Chazelle and J. Friedman. A deterministic view of random sampling and its use in geometry. *Combinatorica*, 10(3):229–249, 1990.

D. P. Dobkin and D. Gunopulos. Concept learning with geometric hypotheses. In *Proceedings of Computational Learning Theory (COLT)*, pages 329–336, 1995.

D. Eisenstat. k-fold unions of low-dimensional concept classes. *Information Processing Letters*, 109 (23-24):1232–1234, 2009.

D. Eisenstat and D. Angluin. The VC dimension of k-fold union. *Information Processing Letters*, 101(5):181 – 184, 2007.

E. Ezra, S. Har-Peled, H. Kaplan, and M. Sharir. Decomposing arrangements of hyperplanes: VC-dimension, combinatorial dimension, and point location. *ArXiv e-prints*, December 2017.

D. Feldman and M. Langberg. A unified framework for approximating and clustering data. In *Proceedings of Symposium on the Theory of Computing (STOC)*, pages 569–578, 2011.

H. Johnson. *Definable Families of Finite Vapnik Chernonenkis Dimension*. PhD thesis, University of Maryland, 2008.

A. Kupavskii, N. H. Mustafa, and J. Pach. New lower bounds for epsilon-nets. In *Proceedings of Symposium on Computational Geometry (SoCG)*, pages 54:1–54:16, 2016.

M. Lucic, O. Bachem, and A. Krause. Strong Coresets for Hard and Soft Bregman Clustering with Applications to Exponential Family Mixtures. In *Proceedings of Artificial Intelligence and Statistics (AISTATS)*, 2016.

J. Matoušek. *Lectures in Discrete Geometry*. Springer-Verlag, New York, NY, 2002.

N. H. Mustafa and K. Varadarajan. Epsilon-approximations and epsilon-nets. In J. E. Goodman, J. O'Rourke, and C. D. Tóth, editors, *Handbook of Discrete and Computational Geometry*. CRC Press LLC, 2017, to appear.

J. Pach and G. Tardos. Tight lower bounds for the size of epsilon-nets. *J. Amer. Math. Soc.*, 26(3): 645–658, 2013.

V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

Journal of Machine Learning Research 20 (2019) 1-5 Submitted 12/18; Revised 3/19; Published 4/19

SMART: An Open Source Data Labeling Platform for Supervised Learning

Rob Chew rchew@rti.org
Michael Wenger mwenger@rti.org
Caroline Kery ckery1@umbc.edu
Jason Nance jnance@rti.org
Keith Richards dkrichards@rti.org
Emily Hadley ehadley@rti.org
Peter Baumgartner pbaumgartner@rti.org
Center for Data Science
RTI International
Research Triangle Park, NC 27709, USA

Editor: Alexandre Gramfort

Abstract

SMART is an open source web application designed to help data scientists and research teams efficiently build labeled training data sets for supervised machine learning tasks. SMART provides users with an intuitive interface for creating labeled data sets, supports active learning to help reduce the required amount of labeled data, and incorporates inter-rater reliability statistics to provide insight into label quality. SMART is designed to be platform agnostic and easily deployable to meet the needs of as many different research teams as possible. The project website[1] contains links to the code repository and extensive user documentation.

Keywords: software, supervised learning, data labeling, active learning, open source

1. Introduction

Over the past decade, supervised machine learning has made tremendous strides, achieving or surpassing human-level performance on tasks diverse as image recognition (Real et al., 2017), speech recognition (Saon et al., 2017) and language modeling (Zilly et al., 2016). While many of these achievements are at least partially due to algorithmic advances and specialized computing resources, they also all rely on large labeled data sets. However, in the research community and in industry, it is often acknowledged that the main bottleneck in machine learning adoption is no longer algorithms or hardware, but creating sufficiently large labeled data sets to address meaningful tasks.

To address this issue, we created SMART, an open source application designed to lessen manual labeling effort while maintaining data quality. SMART uses active learning (Settles, 2012) to help reduce the number of labels necessary to develop a reliable model, inter-rater reliability statistics (Cohen, 1960; Fleiss, 1971) to ensure label quality, and data visualization to help project administrators manage coding projects. SMART allows for multiple coders per project and supports on-premise installation, essential for projects where crowdsourcing is prohibited (e.g., when the data set is proprietary or sensitive in nature). While SMART was primarily designed for applied researchers and analysts needing to generate labelled training data, machine learning and human-computer interaction researchers can also use SMART to study how humans label data. This could range from running experiments on how coders interact to assessing active learning algorithms on their total labelling burden (time, costs, etc.).

SMART is part of a broad collection of commercial and open source data labeling platforms. While providing some overlap in functionality, these software products and services range widely in their intended use cases and pricing. Commercial Platform as a Service (PaaS) applications that specialize in data labelling (e.g., Figure Eight) tend to be the most full-featured and put a premium on the usability and scalability of their products. For a fee, these services provide analysts with a platform to organize, annotate, and export their labeled data. In contrast, commercial crowdsourcing platforms (e.g., Amazon Mechanical Turk) provide both annotation templates and access to an on-demand labelling workforce. While crowdsourcing may be an inexpensive and reliable way to perform many micro-tasks, it may be inappropriate for more complex tasks, those that require expert judgement, or for data that is confidential or classified. For open source projects, specialized research software applications focusing on methods like active learning (Settles, 2011) and adaptive sampling (Scott Sievert et al., 2017) are more common than platforms taking a holistic approach to the data labelling workflow. In essence, SMART is the open source alternative to commercial data labelling platforms, providing a powerful set of core features that can be modified and extended by the community to address new use cases. At the time of writing, SMART primarily supports text classification, though it can be extended to model other tasks and data types.

2. Using SMART

SMART is built using an open-source technology stack with the intention of being platform agnostic and easy to deploy. Specifically, SMART is a web application using React, Bootstrap, d3, and webpack for front-end interactions and Django, Redis, and PostgreSQL to support back-end operations. Docker and Docker-Compose[2] are used for OS-level virtualization to aid configuration management and ease deployment in new environments. Docker and Docker-Compose are also SMART's only dependencies; instructions to install SMART using these systems can be found online in the user documentation.[3]

Project Creation. Users create their own projects using a multi-step workflow. To start a project, users provide basic project information (project name and description), declare the set of classes they will annotate in the project, and upload their unlabeled data. Optionally, they can also set user permissions for multi-user projects, modify advanced project settings, and upload an instructional codebook to help define labeling tasks for users. When uploading project data, users have the option of providing a unique identifier to help link exported records and they can also provide pre-labeled data to initialize the active learning models. The advanced settings allow the user to customize the active learning

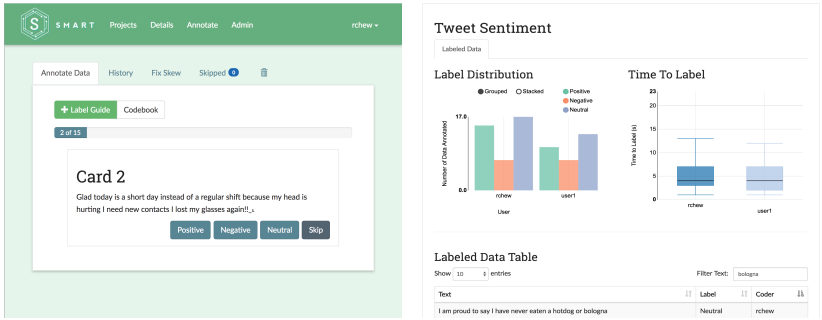
1. https://rtiinternational.github.io/SMART/

2. https://www.docker.com/

3. https://smart-app.readthedocs.io/en/latest/tutorial-installation.html

©2019 Rob Chew, Michael Wenger, Caroline Kery, Jason Nance, Keith Richards, Emily Hadley, and Peter Baumgartner.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-859.html



(a) Annotation Page.

(b) Administrative Dashbboard Page.

Figure 1: In this example, the SMART interface (a) prompts users to rate the sentiment of a tweet. The progress bar indicates the number of tweets labeled against the number assigned to the user in the current batch. Users with administrative privileges can access visualizations of a project's coding progress (b). The Label Distribution chart conveys the number of annotated tweets per class whereas the Time to Label chart summarizes the distribution of time spent per coder labeling tweets. Figures showcasing additional features of SMART are available in the on-line user documentation.

model, set the batch size for the number of observations to label prior to re-running computations, and to enable/disable inter-rater reliability. While sensible default settings are provided, all settings can be customized to meet the specific needs of the labeling project.

Annotating Data. SMART's annotation page allows users to perform several different tasks relating to labeling data. All project members have access to the main labeling interface (Figure 1a) as well as their personal annotation history where they can view and modify previously labeled records. For each record, users can either provide a label or skip the observation. Skipped records are forwarded to a project administrator for adjudication. To help with the labeling task, the annotation page also contains the instructional codebook and the project's labels and definitions (if included during project creation), providing easy access to important annotation resources. Privileged users can also label specific records if the label distribution is highly skewed (Attenberg and Provost, 2010), resolve skipped data elements, and remove records from the project that are completely out-of-scope or otherwise deemed unusable. If inter-rater reliability is enabled, privileged users can also adjudicate double-coded records in which there is label disagreement.

Administrative Dashboard. To help project owners manage their labeling tasks, privileged users have access to an administrator dashboard page that includes data visualizations of the labeling process, performance metrics for the active learning model, and inter-rater reliability metrics across labelers.

The main labeled data view includes two interactive visualizations and an informative data table (Figure 1b). The label distribution chart shows the count of each label per user to compare labeler progress and identify drastic differences in label distributions that might indicate a user's misunderstanding of the labeling task. The time to label chart is a box and whisker plot detailing how long each user is on the annotation page before selecting a label. This chart can help indicate if certain labelers are taking much less time than expected (indicating possible falsification) or more time than expected (indicating potential barriers to labeling).

The active learning model view provides information on the performance of a classifier trained on the labeled project data. The purpose of this view is to help provide insight on when the data labeling has hit a natural saturation point (additional labels are not improving model performance) and to add transparency to the active learning mechanism. SMART currently supports uncertainty sampling (Lewis and Gale, 1994) as the active learning algorithm using least confident, margin, and entropy as measures of uncertainty (Settles, 2012). Currently, several classifier types are supported through the Scikit-learn python library (Pedregosa et al., 2011), and text data elements are vectorized using a term frequency-inverse document frequency (tf-idf) representation (Salton and Buckley, 1988). After each batch of data is labeled, the model is retrained, and its performance is updated and displayed in an interactive data visualization.

Finally, the inter-rater reliability (IRR) view lets privileged users understand how consistently labelers agree on labels that are double-coded. This view is particularly useful early in the labeling process to discover if there is ambiguity in the class definitions or if the task is underspecified. It also provides an indication of how well humans would be expected to perform on the task, a useful benchmark to contextualize model performance. SMART uses Cohens kappa coefficient (Cohen, 1960) to measure IRR between two independent labelers and Fleiss kappa (Fleiss, 1971) for more than two labelers. Users can also view the percent overall agreement and the pairwise percent agreement for unadjusted metrics comparing consistency across labelers. As a visual summary, SMART provides a confusion matrix of labeler agreement in the form of a dynamic heat map. This visualization is valuable in diagnosing if there are particular classes in which labelers disagree, providing targeted insight into potentially ambiguous class definitions.

Exporting Data. SMART allows users to download both the labeled data and the underlying classification model used for active learning (if enabled for the project). For all projects, SMART returns a zipped file of labeled data, sorted by label. If active learning is enabled, SMART will also provide several relevant serialized Python objects using the Pickle protocol, including the preprocessed version of the input data as a tf-idf matrix, the trained classifier, and the fit vectorizer to convert text data into tf-idf vectors. Lastly, a Dockerfile and Jupyter Notebook are provided to assist users in recreating the application environment and running the sample code.

Acknowledgments

We would like to acknowledge the National Consortium for Data Science and RTI International for their generous support in funding the development of SMART. We would also like to thank Lucy Liu for creation of the SMART logo as well as valuable feedback, guidance, and user testing from Annice Kim, Maggie Cress, Jennifer Unangst, and Joey Morris.

References

Josh Attenberg and Foster Provost. Why label when you can search?: Alternatives to active learning for applying human resources to build classification models under extreme class imbalance. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 423–432, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0055-1. doi: 10.1145/1835804.1835859. URL `http://doi.acm.org/10.1145/1835804.1835859`.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960. doi: 10.1177/001316446002000104. URL `https://doi.org/10.1177/001316446002000104`.

Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. Springer-Verlag New York, Inc., 1994.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le, and Alex Kurakin. Large-scale evolution of image classifiers. *arXiv preprint arXiv:1703.01041*, 2017.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

George Saon, Gakuto Kurata, Tom Sercu, Kartik Audhkhasi, Samuel Thomas, Dimitrios Dimitriadis, Xiaodong Cui, Bhuvana Ramabhadran, Michael Picheny, Lynn-Li Lim, et al. English conversational telephone speech recognition by humans and machines. *arXiv preprint arXiv:1703.02136*, 2017.

Scott Sievert, Daniel Ross, Lalit Jain, Kevin Jamieson, Rob Nowak, and Robert Mankoff. NEXT: A system to easily connect crowdsourcing and adaptive data collection. In Katy Huff, David Lippa, Dillon Niederhut, and M Pacer, editors, *Proceedings of the 16th Python in Science Conference*, pages 113 – 119, 2017. doi: 10.25080/shinma-7f4c6e7-010.

Burr Settles. Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1467–1478. Association for Computational Linguistics, 2011.

Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

Julian Georg Zilly, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. Recurrent highway networks. *arXiv preprint arXiv:1607.03474*, 2016.

Journal of Machine Learning Research 20 (2019) 1-28 Submitted 12/18; Revised 4/19; Published 5/19

On the optimality of the Hedge algorithm in the stochastic regime

Jaouad Mourtada JAOUAD.MOURTADA@POLYTECHNIQUE.EDU
Centre de Mathématiques Appliquées
École Polytechnique
Palaiseau, France

Stéphane Gaïffas STEPHANE.GAIFFAS@LPSM.PARIS
Laboratoire de Probabilités, Statistique et Modélisation
Université Paris Diderot
Paris, France

Editor: Laurent Orseau

Abstract

In this paper, we study the behavior of the Hedge algorithm in the online stochastic setting. We prove that anytime Hedge with decreasing learning rate, which is one of the simplest algorithm for the problem of prediction with expert advice, is remarkably both worst-case optimal and adaptive to the easier stochastic and adversarial with a gap problems. This shows that, in spite of its small, non-adaptive learning rate, Hedge possesses the same optimal regret guarantee in the stochastic case as recently introduced adaptive algorithms. Moreover, our analysis exhibits qualitative differences with other versions of the Hedge algorithm, such as the fixed-horizon variant (with constant learning rate) and the one based on the so-called "doubling trick", both of which fail to adapt to the easier stochastic setting. Finally, we determine the intrinsic limitations of anytime Hedge in the stochastic case, and discuss the improvements provided by more adaptive algorithms.

Keywords. Online learning; prediction with expert advice; Hedge; adaptive algorithms.

1. Introduction

The standard setting of *prediction with expert advice* (Littlestone and Warmuth, 1994; Freund and Schapire, 1997; Vovk, 1998; Cesa-Bianchi and Lugosi, 2006) aims to provide sound strategies for sequential prediction that combine the forecasts from different sources. More precisely, in the so-called *Hedge problem* (Freund and Schapire, 1997), at each round the learner has to output a probability distribution on a finite set of *experts* $\{1, \dots, M\}$; the losses of the experts are then revealed, and the learner incurs the expected loss from its chosen probability distribution. The goal is then to control the *regret*, defined as the difference between the cumulative loss of the learner and that of the best expert (with smallest loss). This online prediction problem is typically considered in the *individual sequences* framework, where the losses may be arbitrary and in fact set by an adversary that seeks to maximize the regret. This leads to regret bounds that hold under virtually no assumption (Cesa-Bianchi and Lugosi, 2006).

©2019 Jaouad Mourtada and Stéphane Gaïffas.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-869.html.

In this setting, arguably the simplest and most standard strategy is the *Hedge algorithm* (Freund and Schapire, 1997), also called the *exponentially weighted averaged forecaster* (Cesa-Bianchi and Lugosi, 2006). This algorithm depends on a time-varying parameter η_t called the *learning rate*, which quantifies by how much the algorithm departs from its initial probability distribution to put more weight on the currently leading experts. Given a known finite time horizon T, the standard tuning of the learning rate is fixed and given by $\eta_t = \eta \propto \sqrt{\log(M)/T}$, which guarantees an optimal worst-case regret of order $O(\sqrt{T \log M})$. Alternatively, when T is unknown, one can set $\eta_t \propto \sqrt{\log(M)/t}$ at round t, which leads to an *anytime* $O(\sqrt{T \log M})$ regret bound valid for all $T \geqslant 1$.

While worst-case regret bounds are robust and always valid, they turn out to be overly pessimistic in some situations. A recent line of research (Cesa-Bianchi et al., 2007; de Rooij et al., 2014; Gaillard et al., 2014; Koolen et al., 2014; Sani et al., 2014; Koolen and van Erven, 2015; Luo and Schapire, 2015) designs algorithms that combine $O(\sqrt{T \log M})$ worst-case regret guarantees with an improved regret on easier instances of the problem. An interesting example of such an easier instance is the stochastic problem, where it is assumed that the losses are stochastic and that at each round the expected loss of a "best" expert is smaller than those of the other experts by some gap Δ. Such algorithms rely either on a more careful, data-dependent tuning of the learning rate η_t (Cesa-Bianchi et al., 2007; de Rooij et al., 2014; Koolen et al., 2014; Gaillard et al., 2014), or on more sophisticated strategies (Koolen and van Erven, 2015; Luo and Schapire, 2015). As shown by Gaillard et al. (2014) (see also Koolen et al. 2016), one particular type of adaptive regret bounds (so-called *second-order bounds*) implies at the same time a $O(\sqrt{T \log M})$ worst-case bound and a better *constant* $O(\log(M)/\Delta)$ bound in the stochastic problem with gap Δ. Arguably starting with the early work on second-order bounds (Cesa-Bianchi et al., 2007), the design of online learning algorithms that combine robust worst-case guarantees with improved performance on easier instances has been an active research goal in recent years (de Rooij et al., 2014; Gaillard et al., 2014; Koolen et al., 2014; Sani et al., 2014). However, to the best of our knowledge, existing work on the Hedge problem has focused on developing new adaptive algorithms rather than on analyzing the behavior of "conservative" algorithms in favorable scenarios. Owing to the fact that the standard Hedge algorithm is designed for — and analyzed in — the adversarial setting (Littlestone and Warmuth, 1994; Freund and Schapire, 1997; Cesa-Bianchi and Lugosi, 2006), and that its parameters are not tuned adaptively to obtain better bounds in easier instances, it may be considered as overly conservative and not adapted to stochastic environments.

Our contribution. This paper fills a gap in the existing literature by providing an analysis of the standard Hedge algorithm in the stochastic setting. We show that the anytime Hedge algorithm with default learning rate $\eta_t \propto \sqrt{\log(M)/t}$ actually *adapts* to the stochastic setting, in which it achieves an optimal *constant* $O(\log(M)/\Delta)$ regret bound *without any dedicated tuning* for the easier instance, which might be surprising at first sight. This contrasts with previous works, which require the construction of new adaptive (and more involved) algorithms. Remarkably, this property is *not* shared by the variant of Hedge for a known fixed-horizon T with constant learning rate $\eta \propto \sqrt{\log(M)/T}$, since it suffers a $\Theta(\sqrt{T \log M})$ regret even in easier instances. This exhibits a strong difference between the performances of the anytime and the fixed-horizon variants of the Hedge algorithm.

Given the aforementioned adaptivity of Decreasing Hedge, one may wonder whether there is in fact any benefit in using more sophisticated algorithms in the stochastic regime. We answer this question affirmatively, by considering a more refined measure of complexity of a stochastic instance than the gap Δ. Specifically, we show that Decreasing Hedge does not admit improved regret under Bernstein conditions, which are standard low-noise conditions from statistical learning (Mammen and Tsybakov, 1999; Tsybakov, 2004; Bartlett and Mendelson, 2006). By contrast, it was shown by Koolen et al. (2016) that algorithms which satisfy some adaptive adversarial regret bound achieve improved regret under Bernstein conditions. Finally, we characterize the behavior of Decreasing Hedge in the stochastic regime, by showing that its eventual regret on *any* stochastic instance is governed by the gap Δ.

Related work. In the bandit setting, where the feedback only consists of the loss of the selected action, there has also been some interest in "best-of-both-worlds" algorithms that combine optimal $O(\sqrt{MT})$ worst-case regret in the adversarial regime with improved $O(M \log T)$ regret (up to logarithmic factors) in the stochastic case (Bubeck and Slivkins, 2012; Seldin and Slivkins, 2014; Auer and Chiang, 2016). In particular, Seldin and Slivkins (2014); Seldin and Lugosi (2017) showed that by augmenting the standard EXP3 algorithm for the adversarial regime (an analogue of Hedge with $\Theta(1/\sqrt{t})$ learning rate) with a special-purpose gap detection mechanism, one can achieve poly-logarithmic regret in the stochastic case. This result is strengthened in some recent follow-up work (Zimmert and Seldin, 2019; Zimmert et al., 2019), that appeared since the completion of the first version of the present paper, which obtains optimal regret in the stochastic and adversarial regimes through a variant of the Follow-The-Regularized-Leader (FTRL) algorithm with $\Theta(1/\sqrt{t})$ learning rate and a proper regularizer choice. This result can be seen as an analogue in the bandit case of our upper bound for Decreasing Hedge. Note that, in the bandit setting, the hardness of an instance is essentially characterized by the gap Δ (Bubeck and Cesa-Bianchi, 2012); in particular, the Bernstein condition, which depends on the correlations between the losses of the experts, cannot be exploited under bandit feedback, where one only observes one arm at each round. Hence, it appears that the negative part of our results (on the limitations of Hedge) does not have an analogue in the bandit case.

A similar adaptivity result for FTRL with decreasing $\Theta(1/\sqrt{t})$ learning rate has been observed in a different context by Huang et al. (2017). Specifically, it is shown that, in the case of online linear optimization on a Euclidean ball, FTRL with squared norm regularizer and learning rate $\Theta(1/\sqrt{t})$ achieves $O(\log T)$ regret when the loss vectors are i.i.d. This result is an analogue of our upper bound for Hedge, since this algorithm corresponds to FTRL on the simplex with entropic regularizer (Cesa-Bianchi and Lugosi, 2006; Hazan, 2016). On the other hand, the simplex lacks the curvature of the Euclidean ball, which is important to achieve small regret; here, the improved regret is ensured by a condition on the distribution, namely the existence of a gap Δ. Our lower bound for Hedge shows that this condition is necessary, thereby characterizing the long-term regret of FTRL on the simplex with entropic regularizer. In the case of the Euclidean ball with squared norm regularizer, the norm of the expected loss vector appears to play a similar role, as shown by the upper bound from Huang et al. (2017).

Outline. We define the setting of prediction with expert advice and the Hedge algorithm in Section 2, and we recall herein its standard worst-case regret bound. In Section 3, we consider the behavior of the Hedge algorithm on easier instances, namely the stochastic setting with a gap Δ on the best expert. Under an i.i.d assumption on the sequence of losses, we provide in Theorem 2 an upper bound on the regret of order $(\log M)/\Delta$ for Decreasing Hedge. In Proposition 4, we prove that the rate $(\log M)/\Delta$ cannot be improved in this setting. In Theorem 5 and Corollary 6, we extend the regret guarantees to the adversarial with a gap setting, where a leading expert linearly outperforms the others. These results stand for any Hedge algorithm which is worst-case optimal and with any learning rate which is larger than the one of Decreasing Hedge, namely $O(\sqrt{\log M/t})$. In Proposition 7, we prove the sub-optimality of the fixed-horizon Hedge algorithm, and of another version of Hedge based on the so-called "doubling trick". In Section 4, we discuss the advantages of adaptive Hedge algorithms, and explain what the limitations of Decreasing Hedge are compared to such versions. We include numerical illustrations of our theoretical findings in Section 5, conclude in Section 6 and provide the proofs in Section 7.

2. The expert problem and the Hedge algorithm

In the Hedge setting, also called *decision-theoretic online learning* (Freund and Schapire, 1997), the learner and its adversary (the Environment) sequentially compete on the following game: at each round $t \geqslant 1$,

1. the Learner chooses a probability vector $\boldsymbol{v}_t = (v_{i,t})_{1 \leqslant i \leqslant M}$ on the M experts $1, \dots, M$;
2. the Environment picks a bounded loss vector $\boldsymbol{\ell}_t = (\ell_{i,t})_{1 \leqslant i \leqslant M} \in [0,1]^M$, where $\ell_{i,t}$ is the loss of expert i at round t, while the Learner suffers loss $\widehat{\ell}_t = \boldsymbol{v}_t^\top \boldsymbol{\ell}_t$.

The goal of the Learner is to control its *regret*

$$R_T = \sum_{t=1}^{T} \widehat{\ell}_t - \min_{1 \leqslant i \leqslant M} \sum_{t=1}^{T} \ell_{i,t} \tag{1}$$

for every $T \geqslant 1$, irrespective of the sequence of loss vectors $\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots$ chosen by the Environment. One of the most standard algorithms for this setting is the *Hedge* algorithm. The Hedge algorithm, also called the exponentially weighted averaged forecaster, uses the vector of probabilities $\boldsymbol{v}_t = (v_{i,t})_{1 \leqslant i \leqslant M}$ given by

$$v_{i,t} = \frac{e^{-\eta_t L_{i,t-1}}}{\sum_{j=1}^{M} e^{-\eta_t L_{j,t-1}}} \tag{2}$$

at each $t \geqslant 1$, where $L_{i,T} = \sum_{t=1}^{T} \ell_{i,t}$ denotes the cumulative loss of expert i for every $T \geqslant 1$. Let us also denote $\widehat{L}_T := \sum_{t=1}^{T} \widehat{\ell}_t$ and $R_{i,T} = \widehat{L}_T - L_{i,T}$ the regret with respect to expert i. We consider in this paper the following variants of Hedge, where $c_0 > 0$ is a constant.

Decreasing Hedge (Auer et al., 2002). This is Hedge with the sequence of learning rates $\eta_t = c_0 \sqrt{\log(M)/t}$.

Constant Hedge (Littlestone and Warmuth, 1994). Given a finite time horizon $T \geqslant 1$, this is Hedge with constant learning rate $\eta_t = c_0\sqrt{\log(M)/T}$.

Hedge with doubling trick (Cesa-Bianchi et al., 1997; Cesa-Bianchi and Lugosi, 2006). This variant of Hedge uses a constant learning rate on geometrically increasing intervals, restarting the algorithm at the beginning of each interval. Namely, it uses

$$v_{i,t} = \frac{\exp(-\eta_t \sum_{s=T_k}^{t-1} \ell_{i,s})}{\sum_{j=1}^{M} \exp(-\eta_t \sum_{s=T_k}^{t-1} \ell_{j,s})}, \tag{3}$$

with $T_l = 2^l$ for $l \geqslant 0$, $k \in \mathbf{N}$ such that $T_k \leqslant t < T_{k+1}$ and $\eta_t = c_0\sqrt{\log(M)/T_k}$.

Let us recall the following standard regret bound for the Hedge algorithm from Chernov and Zhdanov (2010).

Proposition 1 *Let $\eta_1, \eta_2, \dots$ be a decreasing sequence of learning rates. The Hedge algorithm* (2) *satisfies the following regret bound:*

$$R_T \leqslant \frac{1}{\eta_T}\log M + \frac{1}{8}\sum_{t=1}^{T}\eta_t\,. \tag{4}$$

In particular, the choice $\eta_t = 2\sqrt{\log(M)/t}$ yields a regret bound of $\sqrt{T\log M}$ for every $T \geqslant 1$.

Note that the regret bound stated in Equation (4) holds for every sequence of losses $\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots$, which makes it valid under no assumption (aside from the boundedness of the losses). The worst-case regret bound in $O(\sqrt{T\log M})$ is achieved by Decreasing Hedge, Hedge with doubling trick and Constant Hedge (whenever T is known in advance). The $O(\sqrt{T\log M})$ rate cannot be improved either by Hedge or any other algorithm: it is known to be the minimax optimal regret (Cesa-Bianchi and Lugosi, 2006). Contrary to Constant Hedge, Decreasing Hedge is anytime, in the sense that it achieves the $O(\sqrt{T\log M})$ regret bound simultaneously for each $T \geqslant 1$. We note that this worst-case regret analysis fails to exhibit any difference between these three algorithms.

In many cases, this $\sqrt{T}$ regret bound is pessimistic, and more "aggressive" strategies (such as the follow-the-leader algorithm, which plays at each round the uniform distribution on the experts with smallest loss, Cesa-Bianchi and Lugosi, 2006) may achieve constant regret in easier instances, even though they lack regret guarantees in the adversarial regime. We show in Section 3 below that Decreasing Hedge is actually better than both Constant Hedge and Hedge with doubling trick in some easier instance of the problem (including in the stochastic setting). This entails that Decreasing Hedge is actually able to adapt, without any modification, to the easiness of the problem considered.

3. Regret of Hedge variants on easy instances

In this section, we depart from the worst-case regret analysis and study the regret of the considered variants of the Hedge algorithm on easier instances of the prediction with expert advice problem.

3.1. Optimal regret for Decreasing Hedge in the stochastic regime

We examine the behavior of Decreasing Hedge in the stochastic regime, where the losses are the realization of some (unknown) stochastic process. More precisely, we consider the standard i.i.d. case, where the loss vectors $\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots$ are i.i.d. (independence holds over rounds, but not necessarily across experts). In this setting, the regret can be much smaller than the worst-case $\sqrt{T\log M}$ regret, since the best expert (with smallest expected loss) will dominate the rest after some time. Following Gaillard et al. (2014); Luo and Schapire (2015), the easiness parameter we consider in this case, which governs the time needed for the best expert to have the smallest cumulative loss and hence the incurred regret, is the sub-optimality gap $\Delta = \min_{i \neq i^*} \mathbb{E}[\ell_{i,t} - \ell_{i^*,t}]$, where $i^* = \operatorname{argmin}_i \mathbb{E}[\ell_{i,t}]$.

We show below that, despite the fact that Decreasing Hedge is designed for the worst-case setting described in Section 2, it is able to adapt to the easier problem considered here, Indeed, Theorem 2 shows that Decreasing Hedge achieves a *constant*, and in fact *optimal* (by Proposition 4 below) regret bound in this setting, in spite of its "conservative" learning rate.

With the exception of the high-probability bound of Corollary 6, the upper and lower bounds in the stochastic case are stated for the *pseudo-regret* $\mathcal{R}_T = \mathbb{E}[R_{i^*,T}]$ (similar bounds hold for the the expected regret $\mathbb{E}[R_T]$, since $\mathcal{R}_T \leqslant \mathbb{E}[R_T]$ and by Remark 14 in Section 7.1).

Theorem 2 *Let $M \geqslant 3$. Assume that the loss vectors $\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots$ are i.i.d. random variables, where $\boldsymbol{\ell}_t = (\ell_{i,t})_{1 \leqslant i \leqslant M}$. Also, assume that there exists $i^* \in \{1, \dots, M\}$ and $\Delta > 0$ such that*

$$\mathbb{E}[\ell_{i,t} - \ell_{i^*,t}] \geqslant \Delta \tag{5}$$

for every $i \neq i^*$. Then, the Decreasing Hedge algorithm with learning rate $\eta_t = 2\sqrt{(\log M)/t}$ achieves the following pseudo-regret bound: for every $T \geqslant 1$,

$$\mathcal{R}_T \leqslant \frac{4\log M + 25}{\Delta}\,. \tag{6}$$

The proof of Theorem 2 is given in Section 7.1. Theorem 2 proves that, in the stochastic setting with a gap Δ, the Decreasing Hedge algorithm achieves a regret $O(\log(M)/\Delta)$, without any prior knowledge of Δ. This matches the guarantees of adaptive Hedge algorithms which are explicitly designed to adapt to easier instances (Gaillard et al., 2014; Luo and Schapire, 2015). This result may seem surprising at first: indeed, adaptive exponential weights algorithms that combine optimal regret in the adversarial setting and constant regret in easier scenarios, such as Hedge with a second-order tuning (Cesa-Bianchi et al., 2007) or AdaHedge (de Rooij et al., 2014), typically use a data-dependent learning rate η_t that adapts to the properties of the losses. While the learning rate η_t chosen by these algorithms may be as low as the worst-case tuning $\eta_t \propto \sqrt{\log(M)/t}$, in the stochastic case those algorithms will use larger, lower-bounded learning rates to ensure constant regret. As Theorem 2 above shows, it turns out that the data-independent, "safe" learning rates $\eta_t \propto \sqrt{\log(M)/t}$ used by "vanilla" Decreasing Hedge are still large enough to adapt to the stochastic case.

Idea of the proof. The idea of the proof of Theorem 2 is to divide time in two phases: a short initial phase $[\![1, t_1]\!]$, where $t_1 = O(\frac{\log M}{\Delta^2})$, and a second phase $[\![t_1, T]\!]$. The initial phase is dominated by noise, and regret during this period is bounded through the worst-case regret bound of Proposition 1, which gives a regret of $O(\sqrt{t_1 \log M}) = O(\frac{\log M}{\Delta})$. In the second phase, the best expert dominates the rest, and the weights concentrate on this best expert fast enough that the total regret incurred is small. The control of the regret in the second phase relies on the critical fact that, if η_t is at least as large as $\sqrt{(\log M)/t}$, then the following two things occur simultaneously at $t_1 \asymp \frac{\log M}{\Delta^2}$, namely at the beginning of the late phase:

1. with high probability, the best expert i^* dominates all the others linearly: for every $i \neq i^*$ and $t \geqslant t_1$, $L_{i,t} - L_{i^*,t} \geqslant \frac{\Delta t}{2}$;
2. the total weight of all suboptimal experts is controlled: $\sum_{i \neq i^*} v_{i,t_1} \leqslant \frac{1}{2}$. If $\eta_t \geqslant \sqrt{(\log M)/t}$ and the first condition holds, this amounts to $M \exp(-\frac{\Delta}{2}\sqrt{t \log M}) \leqslant \frac{1}{2}$, namely $t_1 \gtrsim \frac{\log M}{\Delta^2}$.

In other words, the learning rate $\eta_t \asymp \sqrt{(\log M)/t}$ ensures that the total weight of suboptimal experts starts vanishing at about the same time as when the best expert starts to dominate the others with a large probability (and remarkably, this property holds for every value of the sub-optimality gap Δ). Finally, the upper bound on the regret in the second phase rests on the two conditions above, together with the bound $\sum_{t \geqslant 1} e^{-c\sqrt{t}} = O(\frac{1}{c^2})$ for $c > 0$.

Remark 3 *The fact that $\sum_{t \geqslant 1} e^{-c\sqrt{t}} = O(1/c^2)$ is also used in the analysis of the EXP3++ bandit algorithm (Seldin and Slivkins, 2014, Lemma 10). In the expert setting considered here, summing the contribution of all experts (which suffices in the bandit setting to obtain the correct order of regret) would yield a significantly suboptimal $O(M/\Delta)$ regret bound, with a linear dependence on the number of experts M. In our case, the decomposition of the regret in two phases, which is explained above, removes the linear dependence on M and allows to obtain the optimal rate $(\log M)/\Delta$.*

We complement Theorem 2 by showing that the $O((\log M)/\Delta)$ regret under the gap condition cannot be improved, in the sense that its dependence on both M and Δ is optimal.

Proposition 4 *Let $\Delta \in (0, \frac{1}{4})$, $M \geqslant 4$ and $T \geqslant (\log M)/(16\Delta^2)$. Then, for any algorithm for the Hedge setting, there exists an i.i.d. distribution over the sequence of losses $(\boldsymbol{\ell}_t)_{t \geqslant 1}$ such that:*

- *there exists $i^* \in \{1, \dots, M\}$ such that, for any $i \neq i^*$, $\mathbb{E}[\ell_{i,t} - \ell_{i^*,t}] \geqslant \Delta$;*
- *the pseudo-regret of the algorithm satisfies:*

$$\mathcal{R}_T \geqslant \frac{\log M}{256\Delta}\,. \tag{7}$$

The proof of Proposition 4 is given in Section 7.2. Proposition 4 generalizes the well-known minimax lower bound of $\Theta(\sqrt{T \log M})$, which is recovered by taking $\Delta \asymp \sqrt{(\log M)/T}$.

3.2. Small regret for Decreasing Hedge in the adversarial with a gap problem

In this section, we extend the regret guarantee of Decreasing Hedge in the stochastic setting (Theorem 2), by showing that it holds for more general algorithms and under more general assumptions. Specifically, we consider an "adversarial with a gap" regime, similar to the one introduced by Seldin and Slivkins (2014) in the bandit case, where the leading expert linearly outperforms the others after some time. As Theorem 5 shows, essentially the same regret guarantee can be obtained in this case, up to an additional $\log(\Delta^{-1})/\Delta$ term. Theorem 5 also applies to any Hedge algorithm whose (possibly data-dependent) learning rate η_t is at least as large as that of Decreasing Hedge, and which satisfies a $O(\sqrt{T \log M})$ worst-case regret bound; this includes algorithms with *anytime* first and second-order tuning of the learning rate (Auer et al., 2002; Cesa-Bianchi et al., 2007; de Rooij et al., 2014). In what follows, we will assume $M \geqslant 3$ for convenience, similar results holds for $M = 2$.

Theorem 5 *Let $M \geqslant 3$. Assume that there exists $\tau_0 \geqslant 1$, $\Delta \in (0, 1)$ and $i^* \in \{1, \dots, M\}$ such that, for every $t \geqslant \tau_0$ and $i \neq i^*$, one has*

$$L_{i,t} - L_{i^*,t} \geqslant \Delta t. \tag{8}$$

Consider any Hedge algorithm with (possibly data-dependent) learning rate η_t such that

- *$\eta_t \geqslant c_0\sqrt{(\log M)/t}$ for some constant $c_0 > 0$;*
- *it admits the following worst-case regret bound: $R_T \leqslant c_1\sqrt{T \log M}$ for every $T \geqslant 1$, for some $c_1 > 0$.*

Then, for every $T \geqslant 1$, the regret of this algorithm is upper bounded as

$$R_T \leqslant c_1\sqrt{\tau_0 \log M} + \frac{c_2 \log M + c_3 \log \Delta^{-1} + c_4}{\Delta} \tag{9}$$

where $c_2 = c_1 + \frac{\sqrt{8}}{c_0}$, $c_3 = \frac{\sqrt{8}}{c_0}$ and $c_4 = \frac{16}{c_0^2}$.

The idea of the proof of Theorem 5 is the same as that of Theorem 2, the only difference being the slightly longer initial phase to account for the adversarial nature of the losses. As a consequence of the general bound of Theorem 5, we can recover the guarantee of Theorem 2 (up to an additional $\log(\Delta^{-1})/\Delta$ term), both in expectation and with high probability, under more general stochastic assumptions than i.i.d. over time. The proofs of Theorem 5 and Corollary 6 are provided in Section 7.3.

Corollary 6 *Assume that the losses $(\ell_{i,t})_{1 \leqslant i \leqslant M, t \geqslant 1}$ are random variables. Also, denoting $\mathcal{F}_t = \sigma\big((\ell_{i,s})_{1 \leqslant i \leqslant M, 1 \leqslant s \leqslant t}\big)$, assume that there exists i^* and $\Delta > 0$ such that*

$$\mathbb{E}\left[\ell_{i,t} - \ell_{i^*,t} \,|\, \mathcal{F}_{t-1}\right] \geqslant \Delta \tag{10}$$

for every $i \neq i^*$ and every $t \geqslant 1$. Then, for any Hedge algorithm satisfying the conditions of Theorem 5, and every $T \geqslant 1$:

$$\mathcal{R}_T \leqslant (5c_1 + 2c_2)\frac{\log M}{\Delta} + 2c_3\frac{\log \Delta^{-1}}{\Delta} + \frac{2c_4}{\Delta}, \tag{11}$$

with c_1, c_2, c_3, c_4 as in Theorem 5. In addition, for every $\varepsilon \in (0,1)$, we have

$$R_T \leqslant \left(c_1\sqrt{8} + 2c_2\right)\frac{\log M}{\Delta} + c_1\frac{\sqrt{8\log M \log \varepsilon^{-1}}}{\Delta} + 2c_3\frac{\log \Delta^{-1}}{\Delta} + \frac{2c_4}{\Delta} \tag{12}$$

with probability at least $1-\varepsilon$.

3.3. Constant Hedge and Hedge with the doubling trick do not adapt to the stochastic case

Now, we show that the adaptivity of Decreasing Hedge to gaps in the losses, established in Sections 3.1 and 3.2, is not shared by the two closely related Constant Hedge and Hedge with the doubling trick, despite the fact that they both achieve the minimax optimal worst-case $O(\sqrt{T\log M})$ regret. Proposition 7 below shows that both algorithms fail to achieve a constant regret, and in fact to improve over their worst-case $\Theta(\sqrt{T\log M})$ regret guarantee, even in the extreme case of experts with constant losses 0 (for the leader), and 1 for the rest (*i.e.*, $\Delta = 1$).

Proposition 7 *Let $T \geqslant 1$, $M \geqslant 2$, and consider the experts $i = 1, \dots, M$ with losses $\ell_{1,t} = 0$, $\ell_{i,t} = 1$ $(1 \leqslant t \leqslant T, 2 \leqslant i \leqslant M)$. Then, the pseudo-regret of Constant Hedge with learning rate $\eta_t = c_0\sqrt{\log(M)/T}$ (where $c_0 > 0$ is a numerical constant) is lower bounded as follows*:

$$\mathcal{R}_T \geqslant \min\left(\frac{\sqrt{T\log M}}{3c_0}, \frac{T}{3}\right). \tag{13}$$

In addition, Hedge with doubling trick (3) *also suffers a pseudo-regret satisfying*

$$\mathcal{R}_T \geqslant \min\left(\frac{\sqrt{T\log M}}{6c_0}, \frac{T}{12}\right). \tag{14}$$

The proof of Proposition 7 is given in Section 7.4. Although Hedge with a doubling trick is typically considered as overly conservative and only suitable for worst-case scenarios Cesa-Bianchi and Lugosi, 2006 (especially due to its periodic restarts, after which it discards past observations), to the best of our knowledge Proposition 7 (together with Theorem 2) is the first to formally demonstrate the advantage of Decreasing Hedge over the doubling trick version. This implies that Decreasing Hedge should not be seen as merely a substitute for Constant Hedge to achieve anytime regret bounds. Indeed, even when the horizon T is fixed, Decreasing Hedge outperforms Constant Hedge in the stochastic setting.

4. Limitations of Decreasing Hedge in the stochastic case

In this section, we explore the limitations of the simple Decreasing Hedge algorithm in the stochastic regime, and exhibit situations where it performs worse than more sophisticated algorithms. The starting observation is that the sub-optimality gap Δ is a rather brittle measure of "hardness" of a stochastic instance, which does not fully reflect the achievable rates. We therefore consider the following fast-rate condition from statistical learning, which refines the sub-optimality gap as a measure of complexity of a stochastic instance.

Definition 8 (Bernstein condition) *Assume that the losses $\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots$ are the realization of a stochastic process. Denote $\mathcal{F}_t = \sigma(\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_t)$ the σ-algebra generated by $\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_t$. For $\beta \in [0,1]$ and $B > 0$, the losses are said to satisfy the* (β, B)-Bernstein condition *if there exists i^* such that, for every $t \geqslant 1$ and $i \neq i^*$,*

$$\mathbb{E}[(\ell_{i,t} - \ell_{i^*,t})^2 \,|\, \mathcal{F}_{t-1}] \leqslant B\mathbb{E}[\ell_{i,t} - \ell_{i^*,t} \,|\, \mathcal{F}_{t-1}]^\beta. \tag{15}$$

The Bernstein condition (Bartlett and Mendelson, 2006), a generalization of the Tsybakov margin condition (Tsybakov, 2004; Mammen and Tsybakov, 1999), is a geometric property on the losses which enables to obtain fast rates (e.g., faster than $O(1/\sqrt{n})$ for parametric classes) in statistical learning; we refer to van Erven et al. (2015) for a discussion of fast rates conditions. The Bernstein condition (15) quantifies the "easiness" of a stochastic instance, and generalizes the gap condition considered in the previous section (see Example 1 below). Roughly speaking, it states that good experts (with near-optimal expected loss) are highly correlated with the best expert. In the examples below, we assume that the loss vectors $\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots$ are i.i.d.

Example 1 (Gap implies Bernstein) *If $\Delta_i = \mathbb{E}[\ell_{i,t} - \ell_{i^*,t}] \geqslant \Delta$ for $i \neq i^*$, then the $(1, \frac{1}{\Delta})$-Bernstein condition holds (Koolen et al., 2016, Lemma 4). Furthermore, letting $\alpha = \mathbb{E}[\ell_{i^*,t}]$ denote the expected loss of the best expert, the $(1, 1+\frac{2\alpha}{\Delta})$-Bernstein condition holds. Indeed, for any $i \neq i^*$, denoting $\mu_i := \mathbb{E}[\ell_{i,t}] = \alpha + \Delta_i$, we have (since $(u-v)^2 \leqslant \max(u^2, v^2) \leqslant u^2 + v^2 \leqslant u + v$ for $u, v \in [0,1]$):*

$$\mathbb{E}\left[(\ell_{i,t} - \ell_{i^*,t})^2\right] \leqslant \mathbb{E}\left[\ell_{i,t} + \ell_{i^*,t}\right] = \frac{\mu_i + \alpha}{\mu_i - \alpha}\mathbb{E}\left[\ell_{i,t} - \ell_{i^*,t}\right] = \left(1 + \frac{2\alpha}{\Delta_i}\right)\mathbb{E}\left[\ell_{i,t} - \ell_{i^*,t}\right],$$

which establishes the claim since $\Delta_i \geqslant \Delta$. This provides an improvement when α is small.

Example 2 (Bernstein without a gap) *Let P be a distribution on $\mathcal{X} \times \{0,1\}$, where $\mathcal{X}$ is some measurable space. Assume that $(X_1, Y_1), (X_2, Y_2) \dots$ are i.i.d. samples from P, and that the experts $i \in \{1, \dots, M\}$ correspond to classifiers $f_i : \mathcal{X} \to \{0,1\}$: $\ell_{i,t} = \mathbf{1}(f_i(X_t) \neq Y_t)$, and that expert i^* is the Bayes classifier: $f_{i^*}(X) = \mathbf{1}(\eta(X) \geqslant 1/2)$, where $\eta(X) = \mathbb{P}(Y = 1 \,|\, X)$. Tsybakov's low noise condition (Tsybakov, 2004), namely $\mathbb{P}(|2\eta(X) - 1| \leqslant t) \leqslant Ct^\kappa$ for some $C > 0$, $\kappa \geqslant 0$ and every $t > 0$, implies the $(\frac{\kappa}{\kappa+1}, B)$-Bernstein condition for some B (see, e.g., Boucheron et al., 2005). In addition, under the Massart condition (Massart and Nédélec, 2006) that $|\eta(X) - 1/2| \geqslant c > 0$, the $(1, 1/(2c))$-Bernstein condition holds. Note that these conditions may hold even with an arbitrarily small sub-optimality gap Δ, since the f_i, $i \neq i^*$, may be arbitrary.*

Theorem 9 below shows that Decreasing Hedge fails to achieve improved rates under Bernstein conditions.

Theorem 9 *For every $T \geqslant 1$, there exists a $(1,1)$-Bernstein stochastic instance on which the pseudo-regret of the Decreasing Hedge algorithm with $\eta_t = c_0\sqrt{(\log M)/t}$ satisfies $\mathcal{R}_T \geqslant \frac{1}{3}\min(\frac{1}{c_0}\sqrt{T\log M}, T)$.*

The proof of Theorem 9 is given in Section 7.6. By contrast, it was shown by Koolen et al. (2016) (and implicitly used by Gaillard et al., 2014) that some adaptive algorithms with data-dependent regret bounds enjoy improved regret under the Bernstein condition. For the sake of completeness, we state this fact in Proposition 10 below, which corresponds to Koolen et al. (2016, Theorem 2), but where the dependence on B is made explicit. We also only provide a bound in expectation, which considerably simplifies the proof. The proof of Proposition 10, which uses the same ideas as Gaillard et al. (2014, Theorem 11), is provided in Section 7.5.

Proposition 10 *Consider an algorithm for the Hedge problem which satisfies the following regret bound: for every $i \in \{1, \dots, M\}$,*

$$R_{i,T} \leqslant C_1 \sqrt{(\log M) \sum_{t=1}^{T} (\widehat{\ell}_t - \ell_{i,t})^2} + C_2 \log M \tag{16}$$

where $C_1, C_2 > 0$ are constants. Assume that the losses satisfy the (β, B)-Bernstein condition. Then, the pseudo-regret of the algorithm satisfies:

$$\mathcal{R}_T \leqslant C_3 (B \log M)^{\frac{1}{2-\beta}} T^{\frac{1-\beta}{2-\beta}} + C_4 \log M \tag{17}$$

where $C_3 = \max(1, 4C_1^2)$ and $C_4 = 2C_2$.

The data-dependent regret bound (16), a "second-order" bound, is satisfied by adaptive algorithms such as Adapt-ML-Prod (Gaillard et al., 2014) and Squint (Koolen and van Erven, 2015). A slightly different variant of second-order regret bounds, which depends on some cumulative variance of the losses across experts, has been considered by Cesa-Bianchi et al. (2007); de Rooij et al. (2014), and is achieved by Hedge algorithms with a data-dependent tuning of the learning rate. Second-order bounds refine so-called *first-order* bounds (Cesa-Bianchi et al., 1997; Auer et al., 2002; Cesa-Bianchi and Lugosi, 2006), which are adversarial regret bounds that scale as $O(\sqrt{L_T^* \log M} + \log M)$, where L_T^* denotes the cumulative loss of the best expert. While first-order bounds may still scale as the worst-case $O(\sqrt{T \log M})$ rate in a typical stochastic instance (where the best expert has a positive expected loss), second-order algorithms are known to achieve constant $O((\log M)/\Delta)$ regret in the stochastic case with gap Δ (Gaillard et al., 2014; Koolen and van Erven, 2015).

Theorem 9, in light of Proposition 10, clarifies where the advantage of second-order algorithms compared to Decreasing Hedge lies: unlike the latter, they can exploit Bernstein conditions on the losses. The contrast is most apparent for Bernstein instances with $\beta = 1$. By Example 1, the existence of a gap Δ implies that the $(1, B)$-Bernstein condition holds with $B \leqslant \frac{1}{\Delta}$. However, as shown by Example 2, B can in fact be much smaller than Δ, in which case the regret bound (17) satisfied by second-order algorithms, namely $O(B \log M)$, significantly improves over the upper bound of $O((\log M)/\Delta)$ of Decreasing Hedge from Theorem 2. Theorem 9 provides an instance where the difference does occur, in the most pronounced case where $B = 1$, so that second-order algorithms enjoy small $O(\log M)$ regret, while Decreasing Hedge suffers $\Theta(\sqrt{T \log M})$ regret.

Remark 11 *The advantage of larger learning rates on some stochastic instances may be understood intuitively as follows. Consider an instance with B small but small gap Δ. The learning rate of Decreasing Hedge is large enough so that it can rule out bad experts (with large enough gap Δ_i) at the optimal rate (i.e., at time $(\log M)/\Delta_i^2$). However, once these bad experts are ruled out, near-optimal experts (with small gap Δ_i) are ruled out late (after $(\log M)/\Delta_i^2$ rounds). On the other hand, the Bernstein assumption entails that those experts are highly correlated with the best expert, the amount of noise on the relative losses of these near-optimal experts is small, so that a larger learning rate could be safely used and would enable to dismiss near-optimal experts sooner.*

Setting the Bernstein condition aside, we conclude by investigating the intrinsic limitations of Decreasing Hedge in the stochastic setting. Indeed, it is natural to ask whether Decreasing Hedge can exploit some other regularity of a stochastic instance, apart from the gap Δ. Theorem 12 shows that this is in fact not the case.

Theorem 12 *For every i.i.d. (over time) stochastic instance with a unique best expert*

$$i^* = \operatorname*{argmin}_{1 \leqslant i \leqslant M} \mathbb{E}[\ell_{i,t}],$$

the pseudo-regret of Decreasing Hedge (with $c_0 \geqslant 1$) satisfies

$$\mathcal{R}_T \geqslant \frac{1}{450 c_0^4 (\log M)^2 \Delta}$$

for $T \geqslant \frac{1}{4\Delta^2}$, where $\Delta := \inf_{i \neq i^*} \mathbb{E}[\ell_{i,t} - \ell_{i^*,t}]$.

Theorem 12 shows (together with the upper bound of Theorem 2) that the eventual regret of Decreasing Hedge on *any* stochastic instance is determined by the sub-optimality gap Δ, and scales (up to a $\log^3 M$ factor, depending on the number of near-optimal experts) as $\Theta(\frac{1}{\Delta})$. This characterizes the behavior of Decreasing Hedge on any stochastic instance.

5. Experiments

In this section, we illustrate our theoretical results by numerical experiments that compare the behavior of various Hedge algorithms in the stochastic regime.

Algorithms. We consider the following algorithms: `hedge` is Decreasing Hedge with the default learning rates $\eta_t = 2\sqrt{\log(M)/t}$, `hedge_constant` is Constant Hedge with constant learning rate $\eta_t = \sqrt{8 \log(M)/T}$, `hedge_doubling` is Hedge with doubling trick with $c_0 = \sqrt{8}$, `adahedge` is the AdaHedge algorithm from de Rooij et al. (2014), which is a variant of the Hedge algorithm with a data-dependent tuning of the learning rate η_t (based on $\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_{t-1}$). As shown in the note Koolen (2018), AdaHedge also benefits from Bernstein conditions. A related algorithm, namely Hedge with second-order tuning of the learning rate (Cesa-Bianchi et al., 2007), performed similarly to AdaHedge on the examples considered below, and was therefore not included. `FTL` is Follow-the-Leader (Cesa-Bianchi and Lugosi, 2006) which puts all mass on the expert with the smallest loss (breaking ties randomly). While FTL serves as a benchmark in the stochastic setting, unlike the other algorithms it lacks any guarantee in the adversarial regime, where its worst-case regret is *linear* in T.

Results. We report in Figure 1 the cumulative regrets of the considered algorithms in four examples. The results for the stochastic instances (a), (b) and (c) described below are averaged over 50 trials.

(*a*) *Stochastic instance with a gap.* This is the standard instance considered in this paper. The losses are drawn independently from Bernoulli distributions (one of parameter 0.3, 2 of parameter 0.4 and 7 of parameter 0.5, so that $M = 10$ and $\Delta = 0.1$). The results of Figure 1a confirm our theoretical results: Decreasing Hedge achieves a small, constant regret which is close to that of AdaHedge and FTL, while Constant Hedge and Hedge with doubling trick suffer a larger regret of order $\sqrt{T}$ (note that, although the expected regret of Constant Hedge converges in this case, the value of this limit depends on its learning rate and hence on T).

(*b*) *"Hard" stochastic instance.* This example has a zero gap $\Delta = 0$ between the two leading experts and $M = 10$, which makes it "hard" from the standpoint of Theorem 2 (which no longer applies in this limit case). The losses are drawn from independent Bernoulli distributions, of parameters 0.5 for the 2 leading experts, and 0.7 for the 8 remaining ones. Although all algorithms suffer an unavoidable $\Theta(\sqrt{T})$ regret due to pure noise, Decreasing Hedge, AdaHedge and FTL achieve better regret than the two conservative Hedge variants (Figure 1b). This is due to the fact that for the former algorithms, the weights of suboptimal experts decrease quickly and only induce a constant regret.

(*c*) *Small loss for the best expert.* In this experiment, we illustrate one advantage of adaptive Hedge algorithms such as AdaHedge over Decreasing Hedge, namely the fact that they admit improved regret bounds when the leading expert has small loss. We considered in this experiment $M = 10$, $\Delta = 0.04$ and the leading expert is $\mathsf{Beta}(0.04, 0.96)$, then 4 $\mathsf{Beta}(0.08, 0.92)$, then 5 $\mathsf{Beta}(0.5, 0.5)$.

(*d*) *Adversarial with a gap instance.* This simple instance is not random, and satisfies the assumptions of Theorem 5. It is defined by $M = 3$, $\Delta = 0.04$, $\ell_{3,t} = \frac{3}{4}$ for $t \geqslant 1$, $(\ell_{1,t}, \ell_{2,t}) = (\frac{1}{2}, 0)$ if $t = 1$, $(0, 1)$ if $t \geqslant 80$ or if t is even, and $(1, 0)$ otherwise. FTL suffers linear regret in the first phase, while Constant Hedge and Hedge with doubling trick suffer $\Theta(\sqrt{T})$ during the second phase.

6. Conclusion

In this article, we carried the regret analysis of the standard exponential weights (Hedge) algorithm in the stochastic expert setting, closing a gap in the existing literature. Our analysis reveals that, despite being tuned for the worst-case adversarial setting and lacking any adaptive tuning of the learning rate, Decreasing Hedge achieves optimal regret in the stochastic setting. This property also enables one to distinguish it qualitatively from other variants including the one with fixed (horizon-dependent) learning rate or the one with doubling trick, which both fail to adapt to gaps in the losses. To the best of our knowledge, this is the first result that shows the superiority of the decreasing learning rate over the doubling trick. In addition, it suggests that, even for a fixed time horizon T, the decreasing learning rate tuning should be favored over the constant one.

Finally, we showed that the regret of Decreasing Hedge on any stochastic instance is essentially characterized by the sub-optimality gap Δ. This shows that adaptive algorithms,

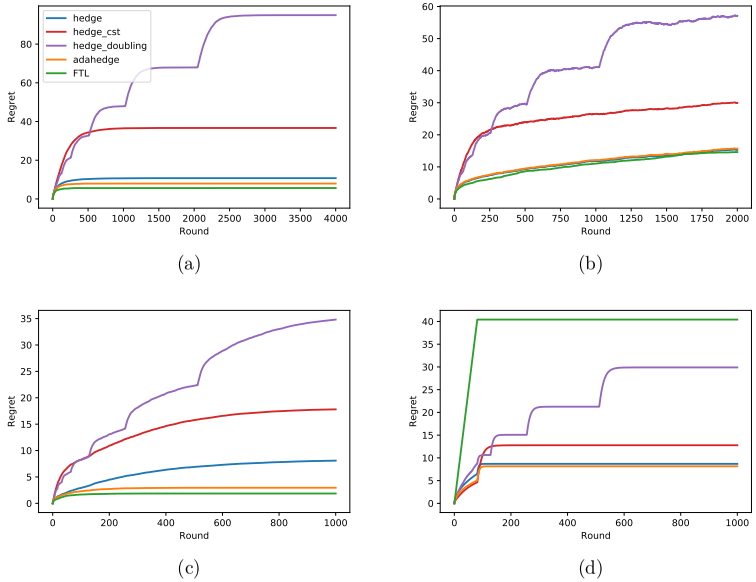


Figure 1: Cumulative regret of Hedge algorithms on four examples, see text for a precise description and discussion about the results. (a) Stochastic instance with a gap; (b) "Hard" stochastic instance; (c) Small loss for the best expert; (d) Adversarial with a gap instance.

including algorithms achieving second-order regret bounds, can actually outperform Decreasing Hedge on some stochastic instances that exhibit a more refined form of "easiness".

A link with stochastic optimization. Our results have a similar flavor to a well-known result (Moulines and Bach, 2011) about stochastic optimization: stochastic gradient descent (SGD) with learning rate $\eta_t \propto 1/\sqrt{t}$ (which is tuned for the convex case but not for the non-strongly convex case) and Polyak-Ruppert averaging achieves a fast $O(1/(\mu t))$ excess risk rate for μ-strongly convex problems, without the knowledge of μ. However, this link stops here since the two results are of a significantly different nature: the $O(1/(\mu t))$ rate is satisfied only by SGD with Polyak-Ruppert averaging, and it does not come from a regret bound; even in the μ-strongly convex case, it can be seen that SGD with step-size $\eta_t \propto 1/\sqrt{t}$ suffers a $\Theta(\sqrt{t})$ regret. In fact, the opposite phenomenon occurs: in stochastic optimization, SGD uses a *larger* $\Theta(1/\sqrt{t})$ step-size than the $\Theta(1/(\mu t))$ step size which exploits the knowledge of strong convexity, but the effect of this larger step-size is balanced by the averaging. By contrast, in the expert setting, Hedge uses a *smaller* $\Theta(\sqrt{(\log M)/t})$ learning rate than the constant, large enough learning rate which exploits the knowledge of the stochastic nature of the problem.

Acknowledgments. The authors wish to warmly thank all four Anonymous Reviewers for their helpful feedback and insights on this work. In particular, the proof of Proposition 4 was proposed by an Anonymous Referee, which allowed to shorten our initial proof.

7. Proofs

We now provide the proofs of the results from the previous Sections, by order of appearance in the text.

7.1. Proof of Theorem 2

Let $t_0 = \left\lceil \frac{8\log M}{\Delta^2} \right\rceil$, so that $\sqrt{t_0} \leqslant \sqrt{1 + \frac{8\log M}{\Delta^2}} \leqslant 1 + \frac{\sqrt{8\log M}}{\Delta}$ (since $\sqrt{a+b} \leqslant \sqrt{a} + \sqrt{b}$ for $a, b \geqslant 0$). The worst-case regret bound of Hedge (Proposition 1) shows that for $1 \leqslant T \leqslant t_0$:

$$R_{i^*,T} \leqslant \sqrt{T\log M} \leqslant \sqrt{t_0 \log M} \leqslant \sqrt{\log M} + \frac{2\sqrt{2}\log M}{\Delta} \leqslant \frac{4\log M}{\Delta} \tag{18}$$

(since $\log M \geqslant 1$ as $M \geqslant 3$, $\Delta \leqslant 1$ and $2\sqrt{2} \leqslant 3$), which establishes (6) for $T \leqslant t_0$. In order to prove (6) for $T \geqslant t_0 + 1$, we start by decomposing the regret with respect to i^* as

$$R_{i^*,T} = \widehat{L}_T - L_{i^*,T} = \widehat{L}_{t_0} - L_{i^*,t_0} + \sum_{t=t_0+1}^{T} (\widehat{\ell}_t - \ell_{i^*,t}) \,. \tag{19}$$

Since $\widehat{L}_{t_0} - L_{i^*,t_0} \leqslant R_{t_0}$ is controlled by (18), it remains to upper bound the second term in (19). First, for every $t \geqslant t_0 + 1$,

$$\widehat{\ell}_t - \ell_{i^*,t} = \sum_{i \neq i^*} v_{i,t}(\ell_{i,t} - \ell_{i^*,t}) \,. \tag{20}$$

Since $\boldsymbol{\ell}_t$ is independent of $\boldsymbol{v}_t$ (which is $\sigma(\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_{t-1})$-measurable), taking the expectation in (20) yields, denoting $\Delta_i = \mathbb{E}[\ell_{i,t} - \ell_{i^*,t}]$,

$$\mathbb{E}[\widehat{\ell}_t - \ell_{i^*,t}] = \sum_{i \neq i^*} \Delta_i \mathbb{E}[v_{i,t}] \,. \tag{21}$$

First, for every $i \neq i^*$, applying Hoeffding's inequality to the i.i.d. centered variables $Z_{i,t} := -\ell_{i,t} + \ell_{i^*,t} + \Delta_i$, which belong to $[-1 + \Delta_i, 1 + \Delta_i]$, yields

$$\begin{aligned} \mathbb{P}\left(L_{i,t-1} - L_{i^*,t-1} < \frac{\Delta_i (t-1)}{2} \right) &= \mathbb{P}\left(\sum_{s=1}^{t-1} Z_{i,s} > \frac{\Delta_i (t-1)}{2} \right) \\ &\leqslant e^{-\frac{t-1}{2}(\Delta_i/2)^2} \\ &= e^{-(t-1)\Delta_i^2/8} \,. \end{aligned} \tag{22}$$

On the other hand, if $L_{i,t-1} - L_{i^*,t-1} \geqslant \Delta_i (t-1)/2$, then

$$\begin{aligned} v_{i,t} &= \frac{e^{-\eta_t (L_{i,t-1} - L_{i^*,t-1})}}{1 + \sum_{j \neq i^*} e^{-\eta_t (L_{j,t-1} - L_{i^*,t-1})}} \\ &\leqslant e^{-2\sqrt{(\log M)/t} \times \Delta_i (t-1)/2} \\ &\leqslant e^{-\Delta_i \sqrt{(t-1)(\log M)/2}} \end{aligned} \tag{23}$$

since $t \leqslant 2(t-1)$. It follows from (23) and (22) that, for $t \geqslant t_0 + 1 \geqslant 2$,

$$\begin{aligned} \mathbb{E}[v_{i,t}] &\leqslant \mathbb{P}\left(L_{i,t-1} - L_{i^*,t-1} > \frac{\Delta_i (t-1)}{2} \right) + e^{-\Delta_i \sqrt{(t-1)(\log M)/2}} \\ &\leqslant e^{-(t-1)\Delta_i^2/8} + e^{-\Delta_i \sqrt{(t-1)(\log M)/2}} \,. \end{aligned} \tag{24}$$

Now, a simple analysis of functions shows that the functions $f_1(u) = ue^{-u}$ and $f_2(u) = ue^{-u^2/2}$ are decreasing on $[1, +\infty)$. Since $\Delta_i \geqslant \Delta$, this entails that

$$\Delta_i e^{-(t-1)\Delta_i^2/8} = \frac{2}{\sqrt{t-1}} f_2\left(\frac{\sqrt{t-1}\Delta_i}{2} \right) \leqslant \frac{2}{\sqrt{t-1}} f_2\left(\frac{\sqrt{t-1}\Delta}{2} \right) = \Delta e^{-(t-1)\Delta^2/8} \tag{25}$$

provided that $\frac{\sqrt{t-1}\Delta}{2} \geqslant 1$, *i.e.* $t \geqslant 1 + \frac{4}{\Delta^2}$, which is the case since $t \geqslant t_0 + 1 \geqslant 1 + \frac{8\log M}{\Delta^2}$. Likewise,

$$\Delta_i e^{-\Delta_i \sqrt{(t-1)(\log M)/2}} \leqslant \Delta e^{-\Delta\sqrt{(t-1)(\log M)/2}} \tag{26}$$

if $\Delta\sqrt{(t-1)(\log M)/2} \geqslant 1$, *i.e.* $t \geqslant 1 + \frac{2}{(\log M)\Delta^2}$, which is ensured by $t \geqslant t_0 + 1$. It follows from (21), (24), (25) and (26) that for every $t \geqslant t_0 + 1$:

$$\begin{aligned} \mathbb{E}[\widehat{\ell}_t - \ell_{i^*,t}] &\leqslant M\Delta e^{-(t-1)\Delta^2/8} + M\Delta e^{-\Delta\sqrt{(t-1)(\log M)/2}} \\ &= \left(Me^{-t_0\Delta^2/8}\right)\left(\Delta e^{-(t-t_0-1)\Delta^2/8}\right) + \left(Me^{-\Delta\sqrt{(t-1)(\log M)/8}}\right)\left(\Delta e^{-\Delta\sqrt{(t-1)(\log M)/8}}\right) \\ &\leqslant \Delta e^{-(t-t_0-1)\Delta^2/8} + \Delta e^{-\Delta\sqrt{(t-1)/8}} \end{aligned} \tag{27}$$

where inequality (27) comes from the bound $Me^{-t_0\Delta^2/8} \leqslant 1$ (since $t_0 \geqslant \frac{8\log M}{\Delta^2}$) and from the fact that $Me^{-\Delta\sqrt{(t-1)(\log M)/8}} \leqslant 1$ amounts to $t \geqslant 1 + \frac{8\log M}{\Delta^2}$, that is, to $t \geqslant t_0 + 1$. Summing inequality (27) yields, for every $T \geqslant t_0 + 1$,

$$\begin{aligned}
\mathbb{E}[\sum_{t=t_0+1}^{T}(\ell_t - \ell_{i^*,t})] &\leqslant \sum_{t=t_0+1}^{T}\left\{\Delta e^{-(t-t_0-1)\Delta^2/8} + \Delta e^{-\Delta\sqrt{(t-1)/8}}\right\} \\
&\leqslant \Delta\sum_{t\geqslant 0} e^{-t\Delta^2/8} + \Delta\sum_{t\geqslant 1} e^{-(\Delta/\sqrt{8})\sqrt{t}} \\
&\leqslant \Delta\left(1 + \frac{8}{\Delta^2}\right) + \Delta\times\frac{2}{(\Delta/\sqrt{8})^2} \qquad (28)\\
&\leqslant \frac{25}{\Delta} \qquad (29)
\end{aligned}$$

where inequality (28) comes from Lemma 13 below. Finally, combining inequalities (18) and (28) yields the pseudo-regret bound $\mathcal{R}_T \leqslant \frac{4\log M + 25}{\Delta}$.

Lemma 13 *For every $\alpha > 0$,*

$$\sum_{t\geqslant 1} e^{-\alpha t} \leqslant \frac{1}{\alpha} \qquad (30)$$

$$\sum_{t\geqslant 1} e^{-\alpha\sqrt{t}} \leqslant \frac{2}{\alpha^2}\,. \qquad (31)$$

Proof Since the functions $t \mapsto e^{-\alpha t}$ and $t \mapsto e^{-\alpha\sqrt{t}}$ are decreasing on $\mathbf{R}^+$, we have

$$\begin{aligned}
\sum_{t\geqslant 1} e^{-\alpha t} &\leqslant \int_0^\infty e^{-\alpha t}\mathrm{d}t = \frac{1}{\alpha} \\
\sum_{t\geqslant 1} e^{-\alpha\sqrt{t}} &\leqslant \int_0^{+\infty} e^{-\alpha\sqrt{t}}\mathrm{d}t \underset{u=\alpha\sqrt{t}}{=} \frac{2}{\alpha^2}\int_0^{+\infty} ue^{-u}\mathrm{d}u = \frac{2}{\alpha^2}.
\end{aligned}$$

■

Remark 14 *While the upper bound of Theorem 2 is stated for the pseudo-regret $\mathcal{R}_T$, a similar upper bound holds for the expected regret $\mathbb{E}[R_T]$. Indeed, under the assumptions of Theorem 2, for every $T \geqslant \frac{4\log M}{\Delta^2}$, we have $\mathbb{E}[R_T] \leqslant \mathcal{R}_T + \frac{1.1}{\Delta}$.*

Proof Note that $\mathbb{E}[R_T] - \mathcal{R}_T = \mathbb{E}[L_{i^*,T} - \min_{1\leqslant i\leqslant T} L_{i,T}]$. For every $a \geqslant 0$, Hoeffding's inequality (applied to the i.i.d. centered variables $\ell_{i^*,t} - \ell_{i,t} + \Delta_i \in [-1+\Delta_i, 1+\Delta_i]$,

$1 \leqslant t \leqslant T$) entails

$$\begin{aligned}
\mathbb{P}\left(L_{i^*,T} - \min_{1\leqslant i\leqslant T} L_{i,T} \geqslant a\right) &\leqslant \sum_{i\neq i^*}\mathbb{P}\left(L_{i^*,T} - L_{i,T} + \Delta_i T \geqslant \Delta_i T + a\right) \\
&\leqslant \sum_{i\neq i^*} e^{-(\Delta_i T + a)^2/(2T)} \qquad (32)\\
&\leqslant Me^{-T\Delta^2/2}e^{-a^2/(2T)} \\
&\leqslant e^{-T\Delta^2/4}e^{-a^2/(2T)}\,, \qquad (33)
\end{aligned}$$

where inequality (33) comes from the fact that $Me^{-T\Delta^2/4} \leqslant 1$ since $T \geqslant \frac{4\log M}{\Delta^2}$. Since the random variable $L_{i^*,T} - \min_{1\leqslant i\leqslant T} L_{i,T}$ is nonnegative, this implies that

$$\begin{aligned}
\mathbb{E}\left[L_{i^*,T} - \min_{1\leqslant i\leqslant T} L_{i,T}\right] &= \int_0^\infty \mathbb{P}\left(L_{i^*,T} - \min_{1\leqslant i\leqslant T} L_{i,T} \geqslant a\right)\mathrm{d}a \\
&\leqslant e^{-T\Delta^2/4}\int_0^\infty e^{-a^2/(2T)}\mathrm{d}a \\
&= \sqrt{\frac{\pi}{2}}\cdot\sqrt{T}e^{-T\Delta^2/4} \\
&= \frac{\sqrt{\pi}}{\Delta}\left[\Delta\sqrt{T/2}\cdot e^{-(\Delta\sqrt{T/2})^2/2}\right] \\
&\leqslant \frac{\sqrt{\pi/e}}{\Delta} \qquad (34)
\end{aligned}$$

where inequality (34) comes from the fact that the function $u \mapsto ue^{-u^2/2}$ attains its maximum on $\mathbf{R}^+$ at $u = 1$. This concludes the proof, since $\sqrt{\pi/e} \leqslant 1.1$. ■

7.2. Proof of Proposition 4

Fix M, Δ and T as in Proposition 4. For $i^* \in \{1,\dots,M\}$, denote $\mathbb{P}_{i^*}$ the following distribution on $[0,1]^{M\times T}$: if $(\ell_{i,t})_{1\leqslant i\leqslant M, 1\leqslant t\leqslant T} \sim \mathbb{P}_{i^*}$, then the variables $\ell_{i,t}$ are independent Bernoulli variables, of parameter $\frac{1}{2} - \Delta$ if $i = i^*$ and $\frac{1}{2}$ otherwise; also, denote by $\mathbb{E}_{i^*}$ the expectation with respect to $\mathbb{P}_{i^*}$. Let $\mathcal{A} = (A_t)_{1\leqslant t\leqslant T}$ be any Hedging algorithm, where $A_t : [0,1]^{M\times(t-1)} \to \mathcal{P}_M$ maps past losses $(\boldsymbol{\ell}_1,\dots,\boldsymbol{\ell}_{t-1})$ to an element of the probability simplex $\mathcal{P}_M \subset \mathbf{R}^M$ on $\{1,\dots,M\}$. For any $i^* \in \{1,\dots,M\}$, let $\mathcal{R}_T(i^*,\mathcal{A})$ denote the pseudo-regret of algorithm $\mathcal{A}$ under the distribution $\mathbb{P}_{i^*}$. Since $\boldsymbol{\ell}_t$ is independent of $\boldsymbol{v}_t$ under $\mathbb{P}_{i^*}$, we have

$$\mathcal{R}_T(i^*,\mathcal{A}) = \sum_{t=1}^{T}\sum_{i\neq i^*}\mathbb{E}_{i^*}\left[v_{i,t}(\ell_{i,t} - \ell_{i^*,t})\right] = \Delta\sum_{t=1}^{T}\sum_{i\neq i^*}\mathbb{E}_{i^*}[v_{i,t}] = \Delta\sum_{t=1}^{T}\mathbb{E}_{i^*}[1 - v_{i^*,t}] \qquad (35)$$

with $\boldsymbol{v}_t := A_t(\boldsymbol{\ell}_1,\dots,\boldsymbol{\ell}_{t-1})$. It follows from Equation (35) that, for every $\mathcal{A}$ and i^*, $\mathcal{R}_T(i^*,\mathcal{A})$ increases with T. Hence, without loss of generality we may assume that $T =$

$\lfloor (\log M)/(16\Delta^2) \rfloor$. The maximum pseudo-regret of $\mathcal{A}$ on the instances $\mathbb{P}_{i^*}$ is lower-bounded as follows:

$$\sup_{1 \leqslant i^* \leqslant M} \mathcal{R}_T(i^*, \mathcal{A}) \geqslant \frac{1}{M} \sum_{1 \leqslant i^* \leqslant M} \mathcal{R}_T(i^*, \mathcal{A}) = \frac{1}{M} \sum_{1 \leqslant i^* \leqslant M} \Delta \sum_{t=1}^{T} \mathbb{E}_{i^*}[1 - v_{i^*,t}] \,. \tag{36}$$

We now "randomize" the algorithm $\mathcal{A}$, by replacing it with a randomized algorithm which picks expert i at time t with probability $v_{i,t}$. Formally, let $\widetilde{P} = \mathcal{U}([0,1])^{\otimes T}$ be the distribution of T independent uniform random variables on $[0,1]$, and denote $\widetilde{\mathbb{P}}_{i^*} = \mathbb{P}_{i^*} \otimes \widetilde{P}$ for $i^* \in \{1, \dots, M\}$. Furthermore, for every $\boldsymbol{v} \in \mathcal{P}_M$, let $I_{\boldsymbol{v}} : [0,1] \to \{1, \dots, M\}$ be a measurable map such that $\mathbb{P}(I_{\boldsymbol{v}}(U) = i) = v_i$ for every $i \in \{1, \dots, M\}$, where $U \sim \mathcal{U}([0,1])$. For every sequence of losses $\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_T$ and random variables $U_1, \dots, U_T$ and every $1 \leqslant t \leqslant T$, let $I_t = I_{\boldsymbol{v}_t}(U_t)$, where $\boldsymbol{v}_t = A_t(\boldsymbol{\ell}_1, \dots, \boldsymbol{\ell}_t)$.

Denote by $\widetilde{\mathbb{E}}_{i^*}$ the expectation with respect to $\widetilde{\mathbb{P}}_{i^*}$. By definition of $I_{\boldsymbol{v}}$, we have $\mathbb{E}_{i^*}[v_{i^*,t}] = \widetilde{\mathbb{E}}_{i^*}[\mathbf{1}(I_t = i^*)]$ so that, denoting $N_i = \sum_{i=1}^{T} \mathbf{1}(I_t = i)$ the number of times expert i is picked,

$$\sum_{t=1}^{T} \mathbb{E}_{i^*}[1 - v_{i^*,t}] = \widetilde{\mathbb{E}}_{i^*}[T - N_{i^*}] \geqslant \mathbb{P}_{i^*}(N_{i^*} \leqslant T/2) \cdot \frac{T}{2} \,.$$

Hence, letting $A_i \subseteq [0,1]^{M \times T} \times [0,1]^T$ be the event $\{N_i > T/2\}$, Equation (36) implies that

$$\sup_{1 \leqslant i^* \leqslant M} \mathcal{R}_T(i^*, \mathcal{A}) \geqslant \frac{\Delta T}{2} \times \frac{1}{M} \sum_{1 \leqslant i^* \leqslant M} \left(1 - \widetilde{\mathbb{P}}_{i^*}(A_{i^*})\right) . \tag{37}$$

It now remains to upper bound $\frac{1}{M} \sum_{i^*} \widetilde{\mathbb{P}}_{i^*}(A_{i^*})$. To do this, first note that the events A_{i^*}, $1 \leqslant i^* \leqslant M$, are pairwise disjoint. Hence, Fano's inequality (see Gerchinovitz et al., 2017, p.2) implies that, for every distribution $\widetilde{\mathbb{Q}}$ on $[0,1]^{M \times T} \times [0,1]^T$,

$$\frac{1}{M} \sum_{1 \leqslant i^* \leqslant M} \widetilde{\mathbb{P}}_{i^*}(A_{i^*}) \leqslant \frac{1}{\log M} \left\{ \frac{1}{M} \sum_{1 \leqslant i^* \leqslant M} \mathrm{KL}(\widetilde{\mathbb{P}}_{i^*}, \widetilde{\mathbb{Q}}) + \log 2 \right\} \tag{38}$$

where $\mathrm{KL}(\mathbb{P}, \mathbb{Q})$ denotes the Kullback-Leibler divergence between $\mathbb{P}$ and $\mathbb{Q}$. Here, we take $\widetilde{\mathbb{Q}} = \mathbb{Q} \otimes \widetilde{P}$, where $\mathbb{Q}$ is the product of Bernoulli distributions $\mathcal{B}(1/2)^{\otimes T}$. This choice leads to

$$\mathrm{KL}(\widetilde{\mathbb{P}}_{i^*}, \widetilde{\mathbb{Q}}) = \mathrm{KL}(\mathbb{P}_{i^*}, \mathbb{Q}) = T \cdot \mathrm{KL}(\mathcal{B}(1/2 - \Delta), \mathcal{B}(1/2)) \leqslant 4T\Delta^2 \leqslant \frac{\log M}{4} \,,$$

where the first bound is obtained by comparing KL and χ^2 divergences (Tsybakov, 2009, Lemma 2.7). Hence, inequality (38) becomes (recalling that $M \geqslant 4$)

$$\frac{1}{M} \sum_{1 \leqslant i^* \leqslant M} \widetilde{\mathbb{P}}_{i^*}(A_{i^*}) \leqslant \frac{(\log M)/4}{\log M} + \frac{\log 2}{\log M} \leqslant \frac{3}{4} \,;$$

plugging this into (37) yields, noting that $T = \lfloor (\log M)/(16\Delta^2) \rfloor \geqslant (\log M)/(32\Delta^2)$ since $(\log M)/(16\Delta^2) \geqslant 1$ (as $M \geqslant 4$ and $\Delta \leqslant \frac{1}{4}$),

$$\sup_{1 \leqslant i^* \leqslant M} \mathcal{R}_T(i^*, \mathcal{A}) \geqslant \frac{\Delta T}{2} \times \frac{1}{4} \geqslant \frac{\log M}{256\Delta} \,.$$

This concludes the proof.

7.3. Proof of Theorem 5 and Corollary 6

Let t_0 be the smallest integer $t \geqslant 1$ such that $Me^{-c_0\Delta\sqrt{t \log(M)/8}} \leqslant \Delta$, namely $t_0 = \left\lceil \frac{8}{c_0^2\Delta^2} \frac{\log^2(M/\Delta)}{\log M} \right\rceil$. Note that $\sqrt{t_0} \leqslant \sqrt{1 + \frac{8}{c_0^2\Delta^2} \frac{\log^2(M/\Delta)}{\log M}} \leqslant 1 + \frac{\sqrt{8}}{c_0\Delta} \frac{\log(M/\Delta)}{\sqrt{\log M}}$. Let $t_1 := t_0 \vee \tau_0$. For every $T \leqslant t_1$, the regret bound in the assumption of Theorem 5 implies

$$\begin{aligned} R_T &\leqslant c_1 \sqrt{T \log M} \\ &\leqslant c_1 \sqrt{\tau_0 \log M} + c_1 \sqrt{t_0 \log M} \\ &\leqslant c_1 \sqrt{\tau_0 \log M} + c_1 \sqrt{\log M} + \frac{\sqrt{8} \log(M/\Delta)}{c_0 \Delta} \end{aligned} \tag{39}$$

which implies (9) with $c_2 = c_1 + \frac{\sqrt{8}}{c_0}$ and $c_3 = \frac{\sqrt{8}}{c_0}$ (since $1 \leqslant \sqrt{\log M} \leqslant \frac{\log M}{\Delta}$). From now on, assume that $T \geqslant t_1 + 1$. Since $T \geqslant \tau_0$, we have $R_T = \widehat{L}_T - L_{i^*,T}$, so that

$$R_T = \widehat{L}_{t_1} - L_{i^*,t_1} + \sum_{t=t_1+1}^{T} \left(\widehat{\ell}_t - \ell_{i^*,t}\right) . \tag{40}$$

In addition, we have for $t \geqslant t_1 + 1$

$$\begin{aligned} \widehat{\ell}_t - \ell_{i^*,t} &= \sum_{i \neq i^*} v_{i,t}(\ell_{i,t} - \ell_{i^*,t}) \\ &\leqslant \sum_{i \neq i^*} v_{i,t} \\ &= \sum_{i \neq i^*} \frac{e^{-\eta_t(L_{i,t-1} - L_{i^*,t-1})}}{1 + \sum_{j \neq i^*} e^{-\eta_t(L_{j,t-1} - L_{i^*,t-1})}} \\ &\leqslant \sum_{i \neq i^*} e^{-c_0\sqrt{(\log M)/t} \times \Delta(t-1)} \qquad (41) \\ &\leqslant Me^{-c_0\Delta\sqrt{(t-1)(\log M)/2}} \\ &\leqslant \left(Me^{-c_0\Delta\sqrt{t_0(\log M)/8}}\right) e^{-c_0\Delta\sqrt{(t-1)/8}} \qquad (42) \\ &\leqslant \Delta e^{-c_0\Delta\sqrt{(t-1)/8}} \qquad (43) \end{aligned}$$

where (41) comes from the fact that $\eta_t \geqslant c_0\sqrt{(\log M)/t}$ and $L_{i,t-1} - L_{i^*,t-1} \geqslant \Delta(t-1)$ (since $t - 1 \geqslant t_1 \geqslant \tau_0$), (42) from the fact that $t - 1 \geqslant t_0$ and $\log M \geqslant 1$, and (43) from the

fact that $Me^{-c_0\Delta\sqrt{t_0(\log M)/8}} \leqslant \Delta$. Summing inequality (43), we obtain

$$\begin{aligned}\sum_{t=t_1+1}^{T}(\widehat{\ell}_t - \ell_{i^*,t}) &\leqslant \sum_{t=t_1+1}^{T}\Delta e^{-c_0\Delta\sqrt{(t-1)/8}}\\ &\leqslant \Delta\sum_{t\geqslant 1}e^{-c_0\Delta\sqrt{t/8}}\\ &\leqslant \Delta\times\frac{2}{(c_0\Delta/\sqrt{8})^2} \qquad (44)\\ &= \frac{16}{c_0^2\Delta} \qquad (45)\end{aligned}$$

where (44) follows from Lemma 13. Combining (40), (39) and (45) proves Theorem 5 with $c_2 = c_1 + \frac{\sqrt{8}}{c_0}$, $c_3 = \frac{\sqrt{8}}{c_0}$ and $c_4 = \frac{16}{c_0^2}$.

Proof [Proof of Corollary 6] Define $\tau = \sup\{t \geqslant 0, \exists i \neq i^*, L_{i,t} - L_{i^*,t} \leqslant \frac{\Delta t}{2}\}$. By Lemma 15 below, for every $\varepsilon > 0$ we have, with probability at least $1-\varepsilon$, $\tau \leqslant 8(\log M + \log\varepsilon^{-1})/\Delta^2$. By Theorem 5, this implies that, with probability at least $1-\varepsilon$,

$$\begin{aligned}R_T &\leqslant c_1\sqrt{\tau\log M} + \frac{c_2\log M + c_3\log\Delta^{-1} + c_4}{\Delta/2}\\ &\leqslant \Big(c_1\sqrt{8} + 2c_2\Big)\frac{\log M}{\Delta} + c_1\frac{\sqrt{8\log M\log\varepsilon^{-1}}}{\Delta} + 2c_3\frac{\log\Delta^{-1}}{\Delta} + \frac{2c_4}{\Delta}\end{aligned}$$

where c_2, c_3, c_4 are the constants of Theorem 5. The bound (11) on the pseudo-regret is obtained similarly from Theorem 5, by using the fact that $\mathcal{R}_T \leqslant \mathbb{E}[R_T]$ and

$$\mathbb{E}[\sqrt{\tau\log M}] \leqslant \sqrt{\mathbb{E}[\tau]\log M} \leqslant \sqrt{\log M}\sqrt{1 + \frac{8(\log M + 1)}{\Delta^2}} \leqslant \sqrt{\log M}\Big(1 + \frac{\sqrt{8\log M} + 1}{\Delta}\Big)$$

which is smaller than $(2+\sqrt{8})(\log M)/\Delta \leqslant 5(\log M)/\Delta$ since $M \geqslant 3$ and $\Delta \leqslant 1$. ■

Lemma 15 *Let $(\ell_{i,t})_{1\leqslant i\leqslant M, t\geqslant 1}$ be as in Theorem 2. Denote $\tau = \sup\{t \geqslant 0, \exists i \neq i^*, L_{i,t} - L_{i^*,t} \leqslant \frac{\Delta t}{2}\}$. We have*

$$\mathbb{E}[\tau] \leqslant 1 + \frac{8(\log M + 1)}{\Delta^2}\,, \qquad (46)$$

and for every $\varepsilon \in (0,1)$,

$$\mathbb{P}\Big(\tau \geqslant \frac{8(\log M + \log\varepsilon^{-1})}{\Delta^2}\Big) \leqslant \varepsilon\,. \qquad (47)$$

Proof [Proof of Lemma 15] For every $i \neq i^*$ and $t \geqslant 1$, let $\Delta_{i,t} := \mathbb{E}[\ell_{i,t} - \ell_{i^*,t} \,|\, \mathcal{F}_{t-1}]$. Using the Hoeffding-Azuma's maximal inequality to the $(\mathcal{F}_t)_{t\geqslant 1}$-martingale difference sequence $Z_{i,t} = -(L_{i,t} - L_{i^*,t}) + \Delta_{i,t}$ (such that $\Delta_{i,t} - 1 \leqslant Z_{i,t} \leqslant \Delta_{i,t} + 1$), together with the fact that $\Delta_{i,t} \geqslant \Delta$, implies that

$$\mathbb{P}\left(\exists t \geqslant t_0, L_{i,t} - L_{i^*,t} \leqslant \frac{\Delta t}{2}\right) \leqslant \mathbb{P}\left(\sup_{t\geqslant t_0}\frac{1}{t}\left(\sum_{s=1}^{t}Z_{i,s}\right) \geqslant \frac{\Delta}{2}\right) \leqslant e^{-t_0\Delta^2/8}\,. \qquad (48)$$

By a union bound, equation (48) implies that

$$\mathbb{P}\left(\tau \geqslant t_0\right) \leqslant Me^{-t_0\Delta^2/8}\,. \qquad (49)$$

Solving for the probability level in (49) yields the high probability bound (47) on τ. The bound on τ in expectation (46) ensues by integrating the high-probability bound over ε. ■

We recall Hoeffding-Azuma's maximal inequality for bounded martingale difference sequences (Hoeffding, 1963; Azuma, 1967). While it follows from a standard argument, we provide a short proof for completeness, since the inequality given in Proposition 16 below differs slightly from the one given in Hoeffding (1963).

Proposition 16 (Hoeffding-Azuma's maximal inequality) *Let $(Z_t)_{t\geqslant 1}$ be a sequence of random variables adapted to a filtration $(\mathcal{F}_t)_{t\geqslant 1}$. Assume that Z_t is a martingale difference sequence: $\mathbb{E}[Z_t \,|\, \mathcal{F}_{t-1}] = 0$ for any $t \geqslant 1$, and that $A_t - 1 \leqslant Z_t \leqslant A_t + 1$ almost surely, where A_t is $\mathcal{F}_{t-1}$-measurable. Then, denoting $S_n := \sum_{t=1}^{n} Z_t$, we have for every $n \geqslant 1$ and $a \geqslant 0$:*

$$\mathbb{P}\left(\sup_{m\geqslant n}\frac{S_m}{m} \geqslant a\right) \leqslant e^{-na^2/2}\,. \qquad (50)$$

Proof Fix $\lambda > 0$. By Hoeffding's inequality, $\mathbb{E}[e^{\lambda Z_t} \,|\, \mathcal{F}_{t-1}] \leqslant e^{\lambda^2/2}$, so that the sequence $M_t^\lambda := \exp\big(\lambda S_t - \lambda^2 t/2\big)$ is a positive supermartingale. Hence, Doob's supermartingale inequality implies that for $\varepsilon \in (0,1]$:

$$\mathbb{P}\Big(\sup_{t\geqslant 1} M_t^\lambda \geqslant \frac{1}{\varepsilon}\Big) \leqslant \frac{\mathbb{E}[M_0^\lambda]}{1/\varepsilon} = \varepsilon\,. \qquad (51)$$

Rearranging (51) and letting $\lambda = \sqrt{2\log(1/\varepsilon)/n}$ yields: with probability $1-\varepsilon$, for every $t \geqslant n$,

$$\frac{S_t}{t} \leqslant \frac{\log(1/\varepsilon)}{\lambda t} + \frac{\lambda}{2} = \sqrt{\frac{\log(1/\varepsilon)}{2}}\left(\frac{\sqrt{n}}{t} + \frac{1}{\sqrt{t}}\right) \leqslant \sqrt{\frac{2\log(1/\varepsilon)}{n}}\,. \qquad (52)$$

Setting $\varepsilon = e^{-na^2/2}$ in (52) gives the desired bound. ■

7.4. Proof of Proposition 7

Note that, since the loss vectors $\boldsymbol{\ell}_t$ are in fact deterministic, $\mathcal{R}_T = R_T$. Denoting $(v_{i,t})_{1\leqslant i\leqslant M}$ the weights selected by the Constant Hedge algorithm at time t, and letting $c = c_0\sqrt{\log M}$, we have

$$\begin{aligned}R_T &= \sum_{t=1}^{T}\sum_{i=2}^{M} v_{i,t}(\ell_{i,t} - \ell_{1,t})\\ &= \sum_{t=1}^{T}\sum_{i=2}^{M}\frac{\exp\big(-\frac{c}{\sqrt{T}}(L_{i,t-1} - L_{1,t-1})\big)}{1 + \sum_{2\leqslant i'\leqslant M}\exp\big(-\frac{c}{\sqrt{T}}(L_{i',t-1} - L_{1,t-1})\big)}\\ &= \sum_{t=1}^{T}\frac{(M-1)\exp\big(-\frac{c}{\sqrt{T}}(t-1)\big)}{1 + (M-1)\exp\big(-\frac{c}{\sqrt{T}}(t-1)\big)}\,. \qquad (53)\end{aligned}$$

Now, let $t_0 \geqslant 0$ be the largest integer such that $(M-1)\exp(-\frac{c}{\sqrt{T}}t) \geqslant 1/2$, namely

$$t_0 = \Big\lfloor \frac{\sqrt{T}}{c} \log(2(M-1)) \Big\rfloor .$$

It follows from Equation (53) that

$$R_T \geqslant \sum_{t=1}^{T\wedge(t_0+1)} \frac{(M-1)\exp\big(-\frac{c}{\sqrt{T}}(t-1)\big)}{1+(M-1)\exp\big(-\frac{c}{\sqrt{T}}(t-1)\big)} \geqslant \frac{1}{3}\min(T, t_0+1) \tag{54}$$

where the second inequality comes from the fact that $\frac{x}{1+x} \geqslant \frac{1}{3}$ for $x \geqslant \frac{1}{2}$, which we apply to $x = (M-1)\exp(-\frac{c}{\sqrt{T}}(t-1)) \geqslant \frac{1}{2}$ for $t \leqslant T \wedge (t_0+1) \leqslant t_0+1$. In order to establish inequality (13), it remains to note that

$$t_0 + 1 \geqslant \frac{\sqrt{T}}{c}\log\big(2(M-1)\big) \geqslant \frac{\sqrt{T\log M}}{c_0},$$

since $2(M-1) \geqslant M$ and $c = \sqrt{c_0 \log M}$.

Now, consider the Hedge algorithm with doubling trick. Assume that $T \geqslant 2$, and let $k \geqslant 1$ such that $T_k \leqslant T < T_{k+1}$. Since $R_T = \sum_{t=1}^{T}\sum_{2\leqslant i\leqslant M} v_{i,t}(\ell_{i,t} - \ell_{1,t})$ and each of the terms in the sum is nonnegative, R_T is lower bounded by the cumulative regret on the period $[\![T_{k-1}, T_k - 1]\!]$. During this period of length T_{k-1}, the algorithm reduces to the Hedge algorithm with constant learning rate $c_0\sqrt{\log(M)/T_{k-1}}$, so that the above bound (13) applies; further bounding $T_{k-1} \geqslant \frac{T}{4}$ establishes (14).

7.5. Proof of Proposition 10

By convexity of $x \mapsto x^2$ and concavity of $x \mapsto x^\beta$, we have:

$$\begin{aligned}
\mathbb{E}[(\widehat{\ell}_t - \ell_{i^*,t})^2] &\leqslant \mathbb{E}\Big[\sum_{i=1}^{M} v_{i,t}(\ell_{i,t} - \ell_{i^*,t})^2\Big] && (55)\\
&= \mathbb{E}\Big[\sum_{i=1}^{M} v_{i,t}\mathbb{E}\left[(\ell_{i,t} - \ell_{i^*,t})^2 \,|\, \mathcal{F}_{t-1}\right]\Big] \\
&\leqslant B\mathbb{E}\Big[\sum_{i=1}^{M} v_{i,t}\mathbb{E}\left[\ell_{i,t} - \ell_{i^*,t} \,|\, \mathcal{F}_{t-1}\right]^\beta\Big] && (56)\\
&\leqslant B\mathbb{E}\Big[\sum_{i=1}^{M} v_{i,t}\mathbb{E}\left[\ell_{i,t} - \ell_{i^*,t} \,|\, \mathcal{F}_{t-1}\right]\Big]^\beta && (57)\\
&= B\mathbb{E}[\widehat{\ell}_t - \ell_{i^*,t}]^\beta && (58)
\end{aligned}$$

where inequalities (55) and (57) come from Jensen's inequality, and (56) from the Bernstein condition (15). Taking the expectation of the regret bound (16), we obtain

$$\begin{aligned}
\mathbb{E}[R_{i^*,T}] &\leqslant \mathbb{E}\left[C_1\sqrt{(\log M)\sum_{t=1}^{T}(\widehat{\ell}_t - \ell_{i^*,t})^2} + C_2\log M\right] \\
&\leqslant C_1\sqrt{(\log M)\sum_{t=1}^{T}\mathbb{E}\big[(\widehat{\ell}_t - \ell_{i^*,t})^2\big]} + C_2\log M && (59)\\
&\leqslant C_1\sqrt{(\log M)B\sum_{t=1}^{T}\mathbb{E}\big[\widehat{\ell}_t - \ell_{i^*,t}\big]^\beta} + C_2\log M \\
&= C_1\sqrt{BT\log M}\left(\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\big[\widehat{\ell}_t - \ell_{i^*,t}\big]^\beta\right)^{1/2} + C_2\log M \\
&\leqslant C_1\sqrt{BT\log M}\left(\frac{\mathbb{E}[R_{i^*,T}]}{T}\right)^{\beta/2} + C_2\log M && (60)
\end{aligned}$$

where inequalities (59) and (60) come from Jensen's inequality. Letting $r = \mathbb{E}[R_{i^*,T}]/T$ and $u = (\log M)/T$, inequality (60) writes $r \leqslant C_1\sqrt{Bu}r^{\beta/2} + C_2 u$. This implies that (depending on which of these two terms is larger) either $r \leqslant 2C_2 u$, or $r \leqslant 2C_1\sqrt{Bu}r^{\beta/2}$, and the latter condition amounts to $r \leqslant (2C_1)^{2/(2-\beta)}(Bu)^{1/(2-\beta)}$. This entails that

$$r \leqslant (2C_1)^{\frac{2}{2-\beta}}(Bu)^{\frac{1}{2-\beta}} + 2C_2 u\,,$$

which amounts to

$$\mathbb{E}[R_{i^*,T}] \leqslant C_3(B\log M)^{\frac{1}{2-\beta}}T^{\frac{1-\beta}{2-\beta}} + C_4\log M \tag{61}$$

where $C_3 = (2C_1)^{2/(2-\beta)} \leqslant \max(1, 4C_1^2)$ and $C_4 = 2C_2$.

7.6. Proof of Theorem 9

Consider the constant losses $\ell_{1,t} = 0$, $\ell_{i,t} = \Delta$ where $\Delta = 1 \wedge c_0^{-1}\sqrt{(\log M)/T}$. These losses satisfy the $(1,1)$-Bernstein condition since, for every $i > 1$, $\mathbb{E}[(\ell_{i,t} - \ell_{1,t})^2] = \Delta^2 \leqslant \Delta = \mathbb{E}[\ell_{i,t} - \ell_{1,t}]$. On the other hand, the regret of the Hedge algorithm with learning rate

$\eta_t = c_0\sqrt{(\log M)/t}$ writes

$$\begin{aligned}
\mathcal{R}_T &= \sum_{t=1}^{T}\sum_{i\neq 1}\mathbb{E}[v_{i,t}(\ell_{i,t}-\ell_{1,t})] \\
&= \Delta\sum_{t=1}^{T}\frac{(M-1)e^{-\eta_t\Delta(t-1)}}{1+(M-1)e^{-\eta_t\Delta(t-1)}} \\
&\geqslant \frac{\Delta}{3}\sum_{t=1}^{T}\mathbf{1}\Big((M-1)e^{-\eta_t\Delta(t-1)}\geqslant\frac{1}{2}\Big) \\
&\geqslant \frac{\Delta}{3}\sum_{t=1}^{T}\mathbf{1}\left(Me^{-c_0\Delta\sqrt{(t-1)\log M}}\geqslant 1\right) \qquad (62)\\
&\geqslant \frac{\Delta}{3}\times\min\left(\frac{\log M}{c_0^2\Delta^2},T\right) \\
&= \frac{1}{3}\min\left(\frac{1}{c_0}\sqrt{T\log M},T\right), \qquad (63)
\end{aligned}$$

where (62) relies on the inequalities $2(M-1)\geqslant M$ and $(t-1)/\sqrt{t}\leqslant\sqrt{t-1}$ for $M\geqslant 2, t\geqslant 1$, while (63) is obtained by noting that $(\log M)/(c_0^2\Delta^2)\geqslant T$ since $\Delta\leqslant c_0^{-1}\sqrt{(\log M)/T}$ and substituting for Δ.

7.7. Proof of Theorem 12

Assume that the loss vectors $\boldsymbol{\ell}_1,\boldsymbol{\ell}_2,\ldots$ are i.i.d., and denote $i^*=\operatorname{argmin}_{1\leqslant i\leqslant M}\mathbb{E}[\ell_{i,t}]$ (which is assumed to be unique), $\Delta=\min_{i\neq i^*}\Delta_i>0$ where $\Delta_i=\mathbb{E}[\ell_{i,t}-\ell_{i^*,t}]$ and $j\in\{1,\ldots,M\}$ such that $\Delta_j=\Delta$. The Decreasing Hedge algorithm with learning rate $\eta_t=c_0\sqrt{(\log M)/t}$ satisfies

$$\begin{aligned}
\mathcal{R}_T &= \sum_{t=1}^{T}\sum_{i\neq i^*}\mathbb{E}[v_{i,t}]\Delta_i \\
&\geqslant \Delta\sum_{t=1}^{T}\mathbb{E}\left[\frac{\sum_{i\neq i^*}e^{-\eta_t(L_{i,t-1}-L_{i^*,t-1})}}{1+\sum_{i\neq i^*}e^{-\eta_t(L_{i,t-1}-L_{i^*,t-1})}}\right] \\
&\geqslant \Delta\sum_{t=1}^{T}\mathbb{E}\left[\frac{e^{-\eta_t(L_{j,t-1}-L_{i^*,t-1})}}{1+e^{-\eta_t(L_{j,t-1}-L_{i^*,t-1})}}\right] \qquad (64)\\
&\geqslant \frac{\Delta}{3}\sum_{t=1}^{T}\mathbb{E}\left[\mathbf{1}\left(e^{-\eta_t(L_{j,t-1}-L_{i^*,t-1})}\geqslant\frac{1}{2}\right)\right] \\
&= \frac{\Delta}{3}\sum_{t=1}^{T}\mathbb{P}\left(\eta_t(L_{j,t-1}-L_{i^*,t-1})\leqslant\log 2\right) \qquad (65)
\end{aligned}$$

where (64) relies on the fact that the function $x\mapsto\frac{x}{1+x}$ is increasing on $\mathbf{R}^+$. Denoting $a=(\log 2)/(c_0\sqrt{\log M})$, we have for every $1\leqslant t\leqslant 1+\frac{a^2}{4\Delta^2}$:

$$\begin{aligned}
\mathbb{P}\left(\eta_t(L_{j,t-1}-L_{i^*,t-1})>\log 2\right) &= \mathbb{P}\left(L_{j,t-1}-L_{i^*,t-1}-\Delta(t-1)>a\sqrt{t}-\Delta(t-1)\right) \\
&\leqslant \mathbb{P}\left(L_{j,t-1}-L_{i^*,t-1}-\Delta(t-1)>\frac{a\sqrt{t-1}}{2}\right) \qquad (66)\\
&\leqslant e^{-a^2/8} \qquad (67)
\end{aligned}$$

where inequality (66) stems from the fact that $\Delta(t-1)\leqslant\frac{a\sqrt{t-1}}{2}$ (since $t\leqslant 1+\frac{a^2}{4\Delta^2}$), while (67) is a consequence of Hoeffding's bound applied to the i.i.d. $[-1-\Delta,1-\Delta]$-valued random variables $\ell_{j,s}-\ell_{i^*,s}-\Delta$, $1\leqslant s\leqslant t-1$. Assuming that $c_0\geqslant 1$, we have $a\leqslant\sqrt{\log 2}\leqslant 1$, so that by concavity of the function $x\mapsto 1-e^{-x/8}$, $1-e^{-a^2/8}\geqslant(1-e^{-1/8})a^2$. Combining this with inequalities (65) and (67) and using the fact that $\left\lfloor 1+\frac{a^2}{4\Delta^2}\right\rfloor\geqslant\frac{a^2}{4\Delta^2}$, we obtain for $T\geqslant\frac{1}{4\Delta^2}\geqslant\frac{a^2}{4\Delta^2}$:

$$\mathbb{E}\left[R_T\right]\geqslant\frac{\Delta}{3}\min\left(\frac{a^2}{4\Delta^2},T\right)(1-e^{-1/8})a^2=\frac{(1-e^{-1/8})a^4}{12\Delta}\geqslant\frac{1}{450c_0^4(\log M)^2\Delta}, \qquad (68)$$

where the last inequality comes from the fact that $(\log 2)^4(1-e^{-1/8})/12\geqslant\frac{1}{450}$.

References

Peter Auer and Chao-Kai Chiang. An algorithm with nearly optimal pseudo-regret for both stochastic and adversarial bandits. In *Proceedings of the 29th Annual Conference on Learning Theory (COLT)*, pages 116–120, 2016.

Peter Auer, Nicolò Cesa-Bianchi, and Claudio Gentile. Adaptive and self-confident on-line learning algorithms. *Journal of Computer and System Sciences*, 64(1):48–75, 2002.

Kazuoki Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal, Second Series*, 19(3):357–367, 1967.

Peter L. Bartlett and Shahar Mendelson. Empirical minimization. *Probability Theory and Related Fields*, 135(3):311–334, 2006.

Stéphane Boucheron, Olivier Bousquet, and Gábor Lugosi. Theory of classification: A survey of some recent advances. *ESAIM: probability and statistics*, 9:323–375, 2005.

Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1): 1–122, 2012.

Sébastien Bubeck and Aleksandrs Slivkins. The best of both worlds: Stochastic and adversarial bandits. In *Proceedings of the 25th Annual Conference on Learning Theory (COLT)*, pages 42.1–42.23, 2012.

Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, Cambridge, New York, USA, 2006.

Nicolò Cesa-Bianchi, Yoav Freund, David Haussler, David P. Helmbold, Robert E. Schapire, and Manfred K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

Nicolò Cesa-Bianchi, Yishay Mansour, and Gilles Stoltz. Improved second-order bounds for prediction with expert advice. *Machine Learning*, 66:321–352, 2007.

Alexey Chernov and Fedor Zhdanov. Prediction with expert advice under discounted loss. In *International Conference on Algorithmic Learning Theory (ALT)*, pages 255–269, 2010.

Steven de Rooij, Tim van Erven, Peter Grünwald, and Wouter M. Koolen. Follow the leader if you can, hedge if you must. *Journal of Machine Learning Research*, 15:1281–1316, 2014.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

Pierre Gaillard, Gilles Stoltz, and Tim van Erven. A second-order bound with excess losses. In *Proceedings of the 27th Annual Conference on Learning Theory (COLT)*, pages 176–196, 2014.

Sebastien Gerchinovitz, Pierre Ménard, and Gilles Stoltz. Fano's inequality for random variables. *arXiv preprint arXiv:1702.05985*, 2017.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

Ruitong Huang, Tor Lattimore, András György, and Csaba Szepesvári. Following the leader and fast rates in online linear prediction: Curved constraint sets and other regularities. *Journal of Machine Learning Research*, 18(145):1–31, 2017.

Wouter M. Koolen. Bounded excess risk for adahedge. `http://blog.wouterkoolen.info/ConstantRegret4AdaHedge/post.html`, 2018. [Online; accessed 3-04-2019].

Wouter M. Koolen and Tim van Erven. Second-order quantile methods for experts and combinatorial games. In *Proceedings of the 28th Annual Conference on Learning Theory (COLT)*, pages 1155–75, 2015.

Wouter M. Koolen, Tim van Erven, and Peter D. Grünwald. Learning the learning rate for prediction with expert advice. In *Advances in Neural Information Processing Systems 27*, pages 2294–2302, 2014.

Wouter M. Koolen, Peter Grünwald, and Tim van Erven. Combining adversarial guarantees and stochastic fast rates in online learning. In *Advances in Neural Information Processing Systems 29*, pages 4457–4465, 2016.

Nick Littlestone and Manfred K Warmuth. The weighted majority algorithm. *Information and computation*, 108(2):212–261, 1994.

Haipeng Luo and Robert E. Schapire. Achieving all with no parameters: AdaNormalHedge. In *Proceedings of the 28th Annual Conference on Learning Theory (COLT)*, pages 1286–1304, 2015.

Enno Mammen and Alexandre B. Tsybakov. Smooth discrimination analysis. *The Annals of Statistics*, 27(6):1808–1829, 1999.

Pascal Massart and Élodie Nédélec. Risk bounds for statistical learning. *The Annals of Statistics*, 34(5):2326–2366, 2006.

Eric Moulines and Francis Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems 24*, pages 451–459, 2011.

Amir Sani, Gergely Neu, and Alessandro Lazaric. Exploiting easy data in online optimization. In *Advances in Neural Information Processing Systems 27*, pages 810–818, 2014.

Yevgeny Seldin and Gábor Lugosi. An improved parametrization and analysis of the EXP3++ algorithm for stochastic and adversarial bandits. In *Proceedings of the 2017 Conference on Learning Theory (COLT)*, pages 1743–1759, 2017.

Yevgeny Seldin and Aleksandrs Slivkins. One practical algorithm for both stochastic and adversarial bandits. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1287–1295, 2014.

Alexandre B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics*, 32(1):135–166, 2004.

Alexandre B. Tsybakov. *Introduction to nonparametric estimation*. Springer, 2009.

Tim van Erven, Peter D. Grünwald, Nishant A. Mehta, Mark D. Reid, and Robert C. Williamson. Fast rates in statistical and online learning. *Journal of Machine Learning Research*, 16(1):1793–1861, 2015.

Vladimir Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56(2):153–173, 1998.

Julian Zimmert and Yevgeny Seldin. An optimal algorithm for stochastic and adversarial bandits. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

Julian Zimmert, Haipeng Luo, and Chen-Yu Wei. Beating stochastic and adversarial semi-bandits optimally and simultaneously. *arXiv preprint arXiv:1901.08779*, 2019.

Journal of Machine Learning Research 20 (2019) 1-40 Submitted 1/19; Published 2/19

Differentiable Game Mechanics

Alistair Letcher* AHP.LETCHER@GMAIL.COM
University of Oxford

David Balduzzi* DBALDUZZI@GOOGLE.COM
DeepMind

Sébastien Racanière SRACANIERE@GOOGLE.COM
DeepMind

James Martens JAMESMARTENS@GOOGLE.COM
DeepMind

Jakob Foerster JAKOBFOERSTER@GMAIL.COM
University of Oxford

Karl Tuyls KARLTUYLS@GOOGLE.COM
DeepMind

Thore Graepel THORE@GOOGLE.COM
DeepMind

Editor: Kilian Weinberger

Abstract

Deep learning is built on the foundational guarantee that gradient descent on an objective function converges to local minima. Unfortunately, this guarantee fails in settings, such as generative adversarial nets, that exhibit multiple interacting losses. The behavior of gradient-based methods in games is not well understood – and is becoming increasingly important as adversarial and multi-objective architectures proliferate. In this paper, we develop new tools to understand and control the dynamics in n-player differentiable games.

The key result is to decompose the game Jacobian into two components. The first, symmetric component, is related to potential games, which reduce to gradient descent on an implicit function. The second, antisymmetric component, relates to *Hamiltonian games*, a new class of games that obey a conservation law akin to conservation laws in classical mechanical systems. The decomposition motivates *Symplectic Gradient Adjustment* (SGA), a new algorithm for finding stable fixed points in differentiable games. Basic experiments show SGA is competitive with recently proposed algorithms for finding stable fixed points in GANs – while at the same time being applicable to, and having guarantees in, much more general cases.

Keywords: game theory, generative adversarial networks, deep learning, classical mechanics, hamiltonian mechanics, gradient descent, dynamical systems

1. Introduction

A significant fraction of recent progress in machine learning has been based on applying gradient descent to optimize the parameters of neural networks with respect to an objective function. The objective functions are carefully designed to encode particular tasks such as supervised learning. A basic result is that gradient descent converges to a local minimum of the objective function under a broad range of conditions (Lee et al., 2017). However, there is a growing set of algorithms that do not optimize a single objective function, including: generative adversarial networks (Goodfellow et al., 2014; Zhu et al., 2017), proximal gradient TD learning (Liu et al., 2016), multi-level optimization (Pfau and Vinyals, 2016), synthetic gradients (Jaderberg et al., 2017), hierarchical reinforcement learning (Wayne and Abbott, 2014; Vezhnevets et al., 2017), intrinsic curiosity (Pathak et al., 2017; Burda et al., 2019), and imaginative agents (Racanière et al., 2017). In effect, the models are trained via games played by cooperating and competing modules.

The time-average of iterates of gradient descent, and other more general no-regret algorithms, are guaranteed to converge to coarse correlated equilibria in games (Stoltz and Lugosi, 2007). However, the dynamics do not converge to Nash equilibria – and do not even stabilize in general (Mertikopoulos et al., 2018; Papadimitriou and Piliouras, 2018). Concretely, cyclic behaviors emerge even in simple cases, see example 1.

This paper presents an analysis of the second-order structure of game dynamics that allows to identify two classes of games, potential and Hamiltonian, that are easy to solve separately. We then derive *symplectic gradient adjustment*[1] (SGA), a method for finding stable fixed points in games. SGA's performance is evaluated in basic experiments.

1.1. Background and Problem Description

Tractable algorithms that converge to Nash equilibria have been found for restricted classes of games: potential games, two-player zero-sum games, and a few others (Hart and Mas-Colell, 2013). Finding Nash equilibria can be reformulated as a nonlinear complementarity problem, but these are 'hopelessly impractical to solve' in general (Shoham and Leyton-Brown, 2008) because the problem is PPAD hard (Daskalakis et al., 2009).

Players are primarily neural nets in our setting. For computational reasons we restrict to gradient-based methods, even though game-theorists have considered a much broader range of techniques. Losses are not necessarily convex in *any* of their parameters, so Nash equilibria are not guaranteed to exist. Even leaving existence aside, finding Nash equilibria in nonconvex games is analogous to, but much harder than, finding global minima in neural nets – which is not realistic with gradient-based methods.

There are at least three problems with gradient-based methods in games. Firstly, the potential existence of cycles (recurrent dynamics) implies there are no convergence guarantees, see example 1 below and Mertikopoulos et al. (2018). Secondly, even when gradient descent converges, the rate of convergence may be too slow in practice because 'rotational forces' necessitate extremely small learning rates, see Figure 4. Finally, since there is no single objective, there is no way to measure progress. Concretely, the losses obtained by the generator and the discriminator in GANs are not useful guides to the quality of the images generated. Application-specific proxies have been proposed, for example the inception score for GANs (Salimans et al., 2016), but these are of little help during training. The inception score is domain specific and is no substitute for looking at samples. This paper tackles the first two problems.

1. Source code is available at `https://github.com/deepmind/symplectic-gradient-adjustment`.

©2019 Alistair Letcher, David Balduzzi, Sébastien Racanière, James Martens, Jakob Foerster, Karl Tuyls, Thore Graepel.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v20/19-008.html`.

1.2. Outline and Summary of Main Contributions

1.2.1. The Infinitesimal Structure of Games

We start with the basic case of a zero-sum bimatrix game: example 1. It turns out that the dynamics under simultaneous gradient descent can be reformulated in terms of Hamilton's equations. The cyclic behavior arises because the dynamics live on the level sets of the Hamiltonian. More directly useful, gradient descent on the Hamiltonian converges to a Nash equilibrium.

Lemma 1 shows that the Jacobian of any game decomposes into symmetric and antisymmetric components. There are thus two 'pure' cases corresponding to when the Jacobian is symmetric and anti-symmetric. The first case, known as potential games (Monderer and Shapley, 1996), have been intensively studied in the game-theory literature because they are exactly the games where gradient descent *does* converge.

The second case, Hamiltonian[2] games, were not studied previously, probably because they coincide with zero-sum games in the bimatrix case (or constant-sum, depending on the constraints). Zero-sum and Hamiltonian games differ when the losses are not bilinear or when there are more than two players. Hamiltonian games are important because (i) they are easy to solve and (ii) general games combine potential-like and Hamiltonian-like dynamics. Unfortunately, the concept of a zero-sum game is too loose to be useful when there are many players: any n-player game can be reformulated as a zero-sum $(n+1)$-player game where $\ell_{n+1} = -\sum_{i=1}^{n} \ell_i$. In this respect, zero-sum games are as complicated as general-sum games. In contrast, Hamiltonian games are much simpler than general-sum games. Theorem 4 shows that Hamiltonian games obey a conservation law – which also provides the key to solving them, by gradient descent on the conserved quantity.

1.2.2. Algorithms

The general case, neither potential nor Hamiltonian, is more difficult and is therefore the focus of the remainder of the paper. Section 3 proposes *symplectic gradient adjustment (SGA)*, a gradient-based method for finding stable fixed points in general games. Appendix A contains TensorFlow code to compute the adjustment. The algorithm computes two Jacobian-vector products, at a cost of two iterations of backprop. SGA satisfies a few natural desiderata explained in Section 3.1: ($D1$) it is compatible with the original dynamics; and it is guaranteed to find stable equilibria in ($D2$) potential and ($D3$) Hamiltonian games.

For general games, correctly picking the *sign* of the adjustment (whether to add or subtract) is critical since it determines the behavior near stable and unstable equilibria. Section 2.4 defines stable equilibria and contrasts them with local Nash equilibria. Theorem 10 proves that SGA converges locally to stable fixed points for sufficiently small parameters (which we quantify via the notion of an additive condition number). While strong, this may be impractical or slow down convergence significantly. Accordingly, Lemma 11 shows how to set the sign so as to be attracted towards stable equilibria and repelled from unstable ones. Correctly aligning SGA allows higher learning rates and faster, more robust convergence, see Theorem 15. Finally, Theorem 17 tackles the remaining class of saddle fixed points by proving that SGA locally avoids strict saddles for appropriate parameters.

2. Lu (1992) defined an unrelated notion of Hamiltonian game.

1.2.3. Experiments

We investigate the empirical performance of SGA in four basic experiments. The first experiment shows how increasing alignment allows higher learning rates and faster convergence, Figure 4. The second set of experiments compares SGA with optimistic mirror descent on two-player and four-player games. We find that SGA converges over a much wider range of learning rates.

The last two sets of experiments investigate mode collapse, mode hopping and the related, less well-known problem of boundary distortion identified in Santurkar et al. (2018). Mode collapse and mode hopping are investigated in a setup involving a two-dimensional mixture of 16 Gaussians that is somewhat more challenging than the original problem introduced in Metz et al. (2017). Whereas simultaneous gradient descent completely fails, our symplectic adjustment leads to rapid convergence – slightly improved by correctly choosing the sign of the adjustment.

Finally, boundary distortion is studied using a 75-dimensional spherical Gaussian. Mode collapse is not an issue since there the data distribution is unimodal. However, as shown in Figure 10, a vanilla GAN with RMSProp learns only one of the eigenvalues in the spectrum of the covariance matrix, whereas SGA approximately learns all of them.

The appendix provides some background information on differential and symplectic geometry, which motivated the developments in the paper. The appendix also explores what happens when the analogy with classical mechanics is pushed further than perhaps seems reasonable. We experiment with assigning units (in the sense of masses and velocities) to quantities in games, and find that type-consistency yields unexpected benefits.

1.3. Related Work

Nash (1950) was only concerned with existence of equilibria. Convergence in two-player games was studied in Singh et al. (2000). WoLF (Win or Learn Fast) converges to Nash equilibria in two-player two-action games (Bowling and Veloso, 2002). Extensions include weighted policy learning (Abdallah and Lesser, 2008) and GIGA-WoLF (Bowling, 2004). Infinitesimal Gradient Ascent (IGA) is a gradient-based approach that is shown to converge to pure Nash equilibria in two-player two-action games. Cyclic behaviour may occur in case of mixed equilibria. Zinkevich (2003) generalised the algorithm to n-action games called GIGA. Optimistic mirror descent approximately converges in two-player bilinear zero-sum games (Daskalakis et al., 2018), a special case of Hamiltonian games. In more general settings it converges to coarse correlated equilibria.

Convergence has also been studied in various n-player settings, see Rosen (1965); Scutari et al. (2010); Facchinei and Kanzow (2010); Mertikopoulos and Zhou (2016). However, the recent success of GANs, where the players are neural networks, has focused attention on a much larger class of *nonconvex* games where comparatively little is known, especially in the n-player case. Heusel et al. (2017) propose a two-time scale methods to find Nash equilibria. However, it likely scales badly with the number of players. Nagarajan and Kolter (2017) prove convergence for some algorithms, but under very strong assumptions (Mescheder et al., 2018). Consensus optimization (Mescheder et al., 2017) is closely related to our proposad algorithm, and is extensively discussed in Section 3. A variety of game-theoretically or

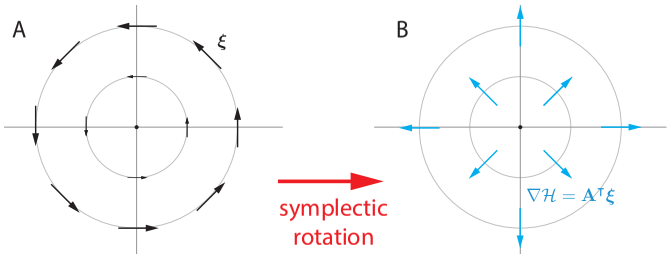


Figure 1: *A minimal example of Hamiltonian mechanics.* Consider a game where $\ell_1(x, y) = xy$, $\ell_2(x, y) = -xy$, and the dynamics are given by $\boldsymbol{\xi}(x, y) = (y, -x)$. The game is a special case of example 1. **(A)** The dynamics $\boldsymbol{\xi}$ cycle around the origin since they live on the level sets of the Hamiltonian $\mathcal{H}(x, y) = \frac{1}{2}(x^2 + y^2)$. **(B)** Gradient descent on the Hamiltonian $\mathcal{H}$ converges to the Nash equilibrium of the game, at the origin $(0, 0)$. Note that $\mathbf{A}^\intercal\boldsymbol{\xi} = (x, y) = \nabla\mathcal{H}$.

minimax motivated modifications to vanilla gradient descent have been investigated in the context of GANs, see Mertikopoulos et al. (2019); Gidel et al. (2018).

Learning with opponent-learning awareness (LOLA) infinitesimally modifies the objectives of players to take into account their opponents' goals (Foerster et al., 2018). However, Letcher et al. (2019) recently showed that LOLA modifies fixed points and thus fails to find stable equilibria in general games.

Symplectic gradient adjustment was independently discovered by Gemp and Mahadevan (2018), who refer to it as "crossing-the-curl". Their analysis draws on powerful techniques from variational inequalities and monotone optimization that are complementary to those developed here—see for example Gemp and Mahadevan (2016, 2017); Gidel et al. (2019). Using techniques from monotone optimization, Gemp and Mahadevan (2018) obtained more detailed and stronger results than ours, in the more particular case of Wasserstein LQ-GANs, where the generator is linear and the discriminator is quadratic (Feizi et al., 2017; Nagarajan and Kolter, 2017).

Network zero-sum games are shown to be Hamiltonian systems in Bailey and Piliouras (2019). The implications of the existence of invariant functions for games is just beginning to be understood and explored.

1.3.1. Notation

Dot products are written as $\mathbf{v}^\intercal\mathbf{w}$ or $\langle \mathbf{v}, \mathbf{w}\rangle$. The angle between two vectors is $\theta(\mathbf{v}, \mathbf{w})$. Positive definiteness is denoted $\mathbf{S} \succ 0$.

2. The Infinitesimal Structure of Games

In contrast to the classical formulation of games, we do not constrain the parameter sets to the probability simplex or require losses to be convex in the corresponding players' parameters. Our motivation is that we are primarily interested in use cases where players are interacting neural nets such as GANs (Goodfellow et al., 2014), a situation in which results from classical game theory do not straightforwardly apply.

Definition 1 (differentiable game)
A differentiable game consists in a set of players $[n] = \{1, \ldots, n\}$ *and corresponding twice continuously differentiable losses* $\{\ell_i : \mathbb{R}^d \to \mathbb{R}\}_{i=1}^n$. *Parameters are* $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_n) \in \mathbb{R}^d$ *where* $\sum_{i=1}^n d_i = d$. *Player* i *controls* $\mathbf{w}_i \in \mathbb{R}^{d_i}$, *and aims to minimize its loss.*

It is sometimes convenient to write $\mathbf{w} = (\mathbf{w}_i, \mathbf{w}_{-i})$ where $\mathbf{w}_{-i}$ concatenates the parameters of all the players other than the i^{th}, which is placed out of order by abuse of notation.

The *simultaneous gradient* is the gradient of the losses with respect to the parameters of the respective players:

$$\boldsymbol{\xi}(\mathbf{w}) = (\nabla_{\mathbf{w}_1}\ell_1, \ldots, \nabla_{\mathbf{w}_n}\ell_n) \in \mathbb{R}^d.$$

By the **dynamics** of the game, we mean following the *negative* of the vector field, $-\boldsymbol{\xi}$, with infinitesimal steps. There is no reason to expect $\boldsymbol{\xi}$ to be the gradient of a *single* function in general, and therefore no reason to expect the dynamics to converge to a fixed point.

2.1. Hamiltonian Mechanics

Hamiltonian mechanics is a formalism for describing the dynamics in classical physical systems, see Arnold (1989); Guillemin and Sternberg (1990). The system is described via canonical coordinates $(\mathbf{q}, \mathbf{p})$. For example, $\mathbf{q}$ often refers to position and $\mathbf{p}$ to momentum of a particle or particles.

The Hamiltonian of the system $\mathcal{H}(\mathbf{q}, \mathbf{p})$ is a function that specifies the total energy as a function of the generalized coordinates. For example, in a closed system the Hamiltonian is given by the sum of the potential and kinetic energies of the particles. The time evolution of the system is given by Hamilton's equations:

$$\frac{d\mathbf{q}}{dt} = \frac{\partial\mathcal{H}}{\partial\mathbf{p}} \quad \text{and} \quad \frac{d\mathbf{p}}{dt} = -\frac{\partial\mathcal{H}}{\partial\mathbf{q}}.$$

An importance consequence of the Hamiltonian formalism is that the dynamics of the physical system—that is, the trajectories followed by the particles in phase space—live on the level sets of the Hamiltonian. In other words, the total energy is conserved.

2.2. Hamiltonian Mechanics in Games

The next example illustrates the essential problem with gradients in games and the key insight motivating our approach.

Example 1 (Conservation of energy in a zero-sum unconstrained bimatrix game)
Zero-sum games, where $\sum_{i=1}^n \ell_i \equiv 0$, *are well-studied. The zero-sum game*

$$\ell_1(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\intercal\mathbf{A}\mathbf{y} \quad \textit{and} \quad \ell_2(\mathbf{x}, \mathbf{y}) = -\mathbf{x}^\intercal\mathbf{A}\mathbf{y}$$

has a Nash equilibrium at $(\mathbf{x}, \mathbf{y}) = (\mathbf{0}, \mathbf{0})$. *The simultaneous gradient* $\boldsymbol{\xi}(\mathbf{x}, \mathbf{y}) = (\mathbf{A}\mathbf{y}, -\mathbf{A}^\intercal\mathbf{x})$ *rotates around the Nash, see Figure 1.*

The matrix $\mathbf{A}$ *admits singular value decomposition (SVD)* $\mathbf{A} = \mathbf{U}^\intercal\mathbf{D}\mathbf{V}$. *Changing to coordinates* $\mathbf{u} = \mathbf{D}^{\frac{1}{2}}\mathbf{U}\mathbf{x}$ *and* $\mathbf{v} = \mathbf{D}^{\frac{1}{2}}\mathbf{V}\mathbf{y}$ *gives* $\ell_1(\mathbf{u},\mathbf{v}) = \mathbf{u}^\intercal\mathbf{v}$ *and* $\ell_2(\mathbf{u},\mathbf{v}) = -\mathbf{u}^\intercal\mathbf{v}$. *Introduce the* Hamiltonian

$$\mathcal{H}(\mathbf{u},\mathbf{v}) = \frac{1}{2}\left(\|\mathbf{u}\|_2^2 + \|\mathbf{v}\|_2^2\right) = \frac{1}{2}\left(\mathbf{x}^\intercal\mathbf{U}^\intercal\mathbf{D}\mathbf{U}\mathbf{x} + \mathbf{y}^\intercal\mathbf{V}^\intercal\mathbf{D}\mathbf{V}\mathbf{y}\right).$$

Remarkably, the dynamics can be reformulated via Hamilton's equations in the coordinates given by the SVD of $\mathbf{A}$*:*

$$\boldsymbol{\xi}(\mathbf{u},\mathbf{v}) = \left(\frac{\partial\mathcal{H}}{\partial\mathbf{v}}, -\frac{\partial\mathcal{H}}{\partial\mathbf{u}}\right).$$

The vector field $\boldsymbol{\xi}$ *cycles around the equilibrium because* $\boldsymbol{\xi}$ *conserves the Hamiltonian's level sets (i.e.* $\langle\boldsymbol{\xi}, \nabla\mathcal{H}\rangle = 0$*). However,* **gradient descent on the Hamiltonian converges to the Nash equilibrium.** *The remainder of the paper explores the implications and limitations of this insight.*

Papadimitriou and Piliouras (2016) recently analyzed the dynamics of Matching Pennies (essentially, the above example) and showed that the cyclic behavior covers the entire parameter space. The Hamiltonian reformulation directly explains the cyclic behavior via a conservation law.

2.3. The Generalized Helmholtz Decomposition

The **Jacobian** of a game with dynamics $\boldsymbol{\xi}$ is the $(d \times d)$-matrix of second-derivatives $\mathbf{J}(\mathbf{w}) := \nabla_{\mathbf{w}} \cdot \boldsymbol{\xi}(\mathbf{w})^\intercal = \left(\frac{\partial\xi_\alpha(\mathbf{w})}{\partial w_\beta}\right)_{\alpha,\beta=1}^{d}$, where $\boldsymbol{\xi}_\alpha(\mathbf{w})$ is the α^{th} entry of the d-dimensional vector $\boldsymbol{\xi}(\mathbf{w})$. Concretely, the Jacobian can be written as

$$\mathbf{J}(\mathbf{w}) = \begin{pmatrix} \nabla^2_{\mathbf{w}_1}\ell_1 & \nabla^2_{\mathbf{w}_1,\mathbf{w}_2}\ell_1 & \cdots & \nabla^2_{\mathbf{w}_1,\mathbf{w}_n}\ell_1 \\ \nabla^2_{\mathbf{w}_2,\mathbf{w}_1}\ell_2 & \nabla^2_{\mathbf{w}_2}\ell_2 & \cdots & \nabla^2_{\mathbf{w}_2,\mathbf{w}_n}\ell_2 \\ \vdots & & & \vdots \\ \nabla^2_{\mathbf{w}_n,\mathbf{w}_1}\ell_n & \nabla^2_{\mathbf{w}_n,\mathbf{w}_2}\ell_n & \cdots & \nabla^2_{\mathbf{w}_n}\ell_n \end{pmatrix}$$

where $\nabla^2_{\mathbf{w}_i,\mathbf{w}_j}\ell_k$ is the $(d_i \times d_j)$-block of 2^{nd}-order derivatives. The Jacobian of a game is a square matrix, but not necessarily symmetric. Note: Greek indices α, β run over d parameter dimensions whereas Roman indices i, j run over n players.

Lemma 1 (generalized Helmholtz decomposition)
The Jacobian of any vector field decomposes uniquely into two components $\mathbf{J}(\mathbf{w}) = \mathbf{S}(\mathbf{w}) + \mathbf{A}(\mathbf{w})$ *where* $\mathbf{S} \equiv \mathbf{S}^\intercal$ *is symmetric and* $\mathbf{A} + \mathbf{A}^\intercal \equiv 0$ *is antisymmetric.*

Proof Any matrix decomposes uniquely as $\mathbf{M} = \mathbf{S} + \mathbf{A}$ where $\mathbf{S} = \frac{1}{2}(\mathbf{M} + \mathbf{M}^\intercal)$ and $\mathbf{A} = \frac{1}{2}(\mathbf{M} - \mathbf{M}^\intercal)$ are symmetric and antisymmetric. The decomposition is preserved by orthogonal change-of-coordinates: given orthogonal matrix $\mathbf{P}$, we have $\mathbf{P}^\intercal\mathbf{M}\mathbf{P} = \mathbf{P}^\intercal\mathbf{S}\mathbf{P} + \mathbf{P}^\intercal\mathbf{A}\mathbf{P}$ since the terms remain symmetric and antisymmetric. Applying the decomposition to the Jacobian yields the result. ■

The connection to the classical Helmholtz decomposition in calculus is sketched in appendix B. Two natural classes of games arise from the decomposition:

Definition 2 *A game is a* ***potential game*** *if the Jacobian is symmetric, i.e. if* $\mathbf{A}(\mathbf{w}) \equiv 0$. *It is a* ***Hamiltonian game*** *if the Jacobian is antisymmetric, i.e. if* $\mathbf{S}(\mathbf{w}) \equiv 0$.

Potential games are well-studied and easy to solve. Hamiltonian games are a new class of games that are also easy to solve. The general case is more difficult, see Section 3.

2.4. Stable Fixed Points (SFPs) vs Local Nash Equilibria (LNEs)

There are (at least) two possible solution concepts in general differentiable games: stable fixed points and local Nash equilibria.

Definition 3 *A point* $\mathbf{w}$ *is a* ***local Nash equilibrium*** *if, for all* i*, there exists a neighborhood* U_i *of* $\mathbf{w}_i$ *such that* $\ell_i(\mathbf{w}_i', \mathbf{w}_{-i}) \geq \ell_i(\mathbf{w}_i, \mathbf{w}_{-i})$ *for* $\mathbf{w}_i' \in U_i$.

We introduce *local* Nash equilibria because finding *global* Nash equilibria is unrealistic in games involving neural nets. Gradient-based methods can reliably find local—but not global—optima of nonconvex objective functions (Lee et al., 2016, 2017). Similarly, gradient-based methods cannot be expected to find global Nash equilibria in nonconvex games.

Definition 4 *A fixed point* $\mathbf{w}^*$ *with* $\boldsymbol{\xi}(\mathbf{w}^*) = 0$ *is* **stable** *if* $\mathbf{J}(\mathbf{w}^*) \succeq 0$ *and* $\mathbf{J}(\mathbf{w}^*)$ *is invertible,* **unstable** *if* $\mathbf{J}(\mathbf{w}^*) \prec 0$ *and a* **strict saddle** *if* $\mathbf{J}(\mathbf{w}^*)$ *has an eigenvalue with negative real part. Strict saddles are a subset of unstable fixed points.*

The definition is adapted from Letcher et al. (2019), where conditions on the Jacobian hold *at* the fixed point; in contrast, Balduzzi et al. (2018a) imposed conditions on the Jacobian in a *neighborhood* of the fixed point. We motivate this concept as follows.

Positive semidefiniteness, $\mathbf{J}(\mathbf{w}^*) \succeq 0$, is a minimal condition for any reasonable notion of stable fixed point. In the case of a single loss ℓ, the Jacobian of $\boldsymbol{\xi} = \nabla\ell$ is the Hessian of ℓ, i.e. $\mathbf{J} = \nabla^2\ell$. Local convergence of gradient descent on single functions cannot be guaranteed if $\mathbf{J}(\mathbf{w}^*) \not\succeq 0$, since such points are strict saddles. These are almost always avoided by Lee et al. (2017), so this semidefinite condition must hold.

Another viewpoint is that invertibility and positive semidefiniteness of the Hessian together imply *positive definiteness*, and the notion of stable fixed point specializes, in a one-player game, to local minima that are detected by the second partial derivative test. These minima are precisely those which gradient-like methods provably converge to. Stable fixed points are defined by analogy, though note that invertibility and semidefiniteness do *not* imply positive definiteness in n-player games since $\mathbf{J}$ may not be symmetric.

Finally, it is important to impose only positive *semi*-definiteness to keep the class as large as possible. Imposing strict positivity would imply that the origin is not an SFP in the cyclic game $\ell_1 = xy = -\ell_2$ from Example 1, while clearly deserving of being so.

Remark 1 *The conditions* $\mathbf{J}(\mathbf{w}^*) \succeq 0$ *and* $\mathbf{J}(\mathbf{w}^*) \prec 0$ *are equivalent to the conditions on the symmetric component* $\mathbf{S}(\mathbf{w}^*) \succeq 0$ *and* $\mathbf{S}(\mathbf{w}^*) \prec 0$ *respectively, since*

$$\mathbf{u}^\intercal\mathbf{J}\mathbf{u} = \mathbf{u}^\intercal\mathbf{S}\mathbf{u} + \mathbf{u}^\intercal\mathbf{A}\mathbf{u} = \mathbf{u}^\intercal\mathbf{S}\mathbf{u}$$

for all $\mathbf{u}$*, by antisymmetry of* $\mathbf{A}$. *This equivalence will be used throughout.*

Stable fixed points and local Nash equilibria are both appealing solution concepts, one from the viewpoint of optimisation by analogy with single objectives, and the other from game theory. Unfortunately, neither is a subset of the other:

Example 2 (stable $\not\Longrightarrow$ local Nash)
Let $\ell_1(x,y) = x^3 + xy$ *and* $\ell_2(x,y) = -xy$. *Then*

$$\boldsymbol{\xi}(x,y) = \begin{pmatrix} 3x^2 + y \\ -x \end{pmatrix} \quad \textit{and} \quad \mathbf{J}(x,y) = \begin{pmatrix} 6x & 1 \\ -1 & 0 \end{pmatrix}.$$

There is a stable fixed point with invertible Hessian at $(x,y) = (0,0)$, *since* $\boldsymbol{\xi}(0,0) = 0$ *and* $\mathbf{J}(0,0) \succeq 0$ *invertible. However any neighbourhood of* $x = 0$ *contains some small* $\epsilon > 0$ *for which* $\ell_1(-\epsilon, 0) = -\epsilon^3 < 0 = \ell_1(0,0)$, *so the origin is not a local Nash equilibrium.*

Example 3 (local Nash $\not\Longrightarrow$ stable)
Let $\ell_1(x,y) = \ell_2(x,y) = xy$. *Then*

$$\boldsymbol{\xi}(x,y) = \begin{pmatrix} y \\ x \end{pmatrix} \quad \textit{and} \quad \mathbf{J}(x,y) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

There is a fixed point at $(x,y) = (0,0)$ *which is a local (in fact, global) Nash equilibrium since* $\ell_1(0,y) = 0 \geq \ell_1(0,0)$ *and* $\ell_2(x,0) = 0 \geq \ell_2(0,0)$ *for all* $x, y \in \mathbb{R}$. *However* $\mathbf{J} = \mathbf{S}$ *has eigenvalues* $\lambda_1 = 1$ *and* $\lambda_2 = -1 < 0$, *so* $(0,0)$ *is not a stable fixed point.*

In Example 3, the Nash equilibrium is a *saddle point* of the common loss $\ell = xy$. Any algorithm that converges to Nash equilibria will thus converge to an undesirable saddle point. This rules out local Nash equilibrium as a solution concept for our purposes. Conversely, Example 2 emphasises the better notion of stability whereby player 1 may have a local incentive to deviate from the origin *immediately*, but would later be punished for doing so since the game is locally dominated by the $\pm xy$ terms, whose only 'resolution' or 'stable minimum' is the origin (see Example 1).

2.5. Potential Games

Potential games were introduced by Monderer and Shapley (1996). It turns out that our definition of potential game above coincides with a special case of the potential games of Monderer and Shapley (1996), which they refer to as exact potential games.

Definition 5 (classical definition of potential game)
A game is a potential game if there is a single potential function $\phi : \mathbb{R}^d \to \mathbb{R}$ *and positive numbers* $\{\alpha_i > 0\}_{i=1}^n$ *such that*

$$\phi(\mathbf{w}_i', \mathbf{w}_{-i}) - \phi(\mathbf{w}_i'', \mathbf{w}_{-i}) = \alpha_i \Big(\ell_i(\mathbf{w}_i', \mathbf{w}_{-i}) - \ell_i(\mathbf{w}_i'', \mathbf{w}_{-i}) \Big)$$

for all i *and all* $\mathbf{w}_i', \mathbf{w}_i'', \mathbf{w}_{-i}$, *see Monderer and Shapley (1996).*

Lemma 2 *A game is a potential game iff* $\alpha_i \nabla_{\mathbf{w}_i} \ell_i = \nabla_{\mathbf{w}_i} \phi$ *for all* i, *which is equivalent to*

$$\alpha_i \nabla^2_{\mathbf{w}_i \mathbf{w}_j} \ell_i = \alpha_j \nabla^2_{\mathbf{w}_i \mathbf{w}_j} \ell_j = \alpha_j \left(\nabla^2_{\mathbf{w}_j \mathbf{w}_i} \ell_j \right)^\intercal \quad \forall i, j. \tag{1}$$

Proof See Monderer and Shapley (1996). ■

Corollary 3 *If* $\alpha_i = 1$ *for all* i *then Equation* (1) *is equivalent to requiring that the Jacobian of the game is symmetric.*

Proof In an exact potential game, the Jacobian coincides with the Hessian of the potential function ϕ, which is necessarily symmetric. ■

Monderer and Shapley (1996) refer to the special case where $\alpha_i = 1$ for all i as an ***exact potential game***. We use the shorthand 'potential game' to refer to exact potential games in what follows.

Potential games have been extensively studied since they are one of the few classes of games for which Nash equilibria can be computed (Rosenthal, 1973). For our purposes, they are games where simultaneous gradient descent on the losses corresponds to gradient descent on a single function. It follows that descent on $\boldsymbol{\xi}$ converges to a fixed point that is a local minimum of ϕ or a saddle.

2.6. Hamiltonian Games

Hamiltonian games, where the Jacobian is antisymmetric, are a new class games. They are related to the harmonic games introduced in Candogan et al. (2011), see Section B.4. An example from Balduzzi et al. (2018b) may help develop intuition for antisymmetric matrices:

Example 4 (antisymmetric structure of tournaments)
Suppose n *competitors play one-on-one and that the probability of player* i *beating player* j *is* p_{ij}. *Then, assuming there are no draws, the probabilities satisfy* $p_{ij} + p_{ji} = 1$ *and* $p_{ii} = \frac{1}{2}$. *The matrix* $\mathbf{A} = \left(\log \frac{p_{ij}}{1 - p_{ij}} \right)_{i,j=1}^n$ *of logits is then antisymmetric. Intuitively, antisymmetry reflects a* hyperadversarial *setting where all pairwise interactions between players are zero-sum.*

Hamiltonian games are closely related to zero-sum games.

Example 5 (an unconstrained bimatrix game is zero-sum iff it is Hamiltonian)
Consider bimatrix game with $\ell_1(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\intercal \mathbf{P} \mathbf{y}$ *and* $\ell_2(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\intercal \mathbf{Q} \mathbf{y}$, *but where the parameters are* not *constrained to the probability simplex. Then* $\boldsymbol{\xi} = (\mathbf{P}\mathbf{y}, \mathbf{Q}^\intercal \mathbf{x})$ *and the Jacobian components have block structure*

$$\mathbf{A} = \frac{1}{2} \begin{pmatrix} 0 & \mathbf{P} - \mathbf{Q} \\ (\mathbf{Q} - \mathbf{P})^\intercal & 0 \end{pmatrix} \quad \textit{and} \quad \mathbf{S} = \frac{1}{2} \begin{pmatrix} 0 & \mathbf{P} + \mathbf{Q} \\ (\mathbf{P} + \mathbf{Q})^\intercal & 0 \end{pmatrix}$$

The game is Hamiltonian iff $\mathbf{S} = 0$ *iff* $\mathbf{P} + \mathbf{Q} = 0$ *iff* $\ell_1 + \ell_2 = 0$.

However, in general there are Hamiltonian games that are *not* zero-sum and vice versa.

Example 6 (Hamiltonian game that is not zero-sum)
Fix constants a and b and suppose players 1 and 2 minimize losses

$$\ell_1(x, y) = x(y - b) \;\; \textit{and} \;\; \ell_2(x, y) = -(x - a)y$$

with respect to x and y respectively.

Example 7 (zero-sum game that is not Hamiltonian)
Players 1 and 2 minimize

$$\ell_1(x, y) = x^2 + y^2 \qquad \ell_2(x, y) = -(x^2 + y^2).$$

The game actually has potential function $\phi(x, y) = x^2 - y^2$.

Hamiltonian games are quite different from potential games. In a Hamiltonian game there is a Hamiltonian function $\mathcal{H}$ that specifies a conserved quantity. In potential games the dynamics *equal* $\nabla\phi$; in Hamiltonian games the dynamics are *orthogonal* to $\nabla\mathcal{H}$. The orthogonality implies the conservation law that underlies the cyclic behavior in example 1.

Theorem 4 (conservation law for Hamiltonian games)
Let $\mathcal{H}(\mathbf{w}) := \frac{1}{2}\|\boldsymbol{\xi}(\mathbf{w})\|_2^2$. If the game is Hamiltonian then

i) $\nabla\mathcal{H} = \mathbf{A}^\intercal\boldsymbol{\xi}$ and

*ii) $\boldsymbol{\xi}$ **preserves the level sets of** $\mathcal{H}$ since $\langle\boldsymbol{\xi}, \nabla\mathcal{H}\rangle = 0$.*

iii) If the Jacobian is invertible and $\lim_{\|\mathbf{w}\|\to\infty}\mathcal{H}(\mathbf{w}) = \infty$ then gradient descent on $\mathcal{H}$ converges to a stable fixed point.

Proof Direct computation shows $\nabla\mathcal{H} = \mathbf{J}^\intercal\boldsymbol{\xi}$ for any game. The first statement follows since $\mathbf{J} = \mathbf{A}$ in Hamiltonian games.

For the second statement, the directional derivative is $D_{\boldsymbol{\xi}}\mathcal{H} = \langle\boldsymbol{\xi}, \nabla\mathcal{H}\rangle = \boldsymbol{\xi}^\intercal\mathbf{A}^\intercal\boldsymbol{\xi}$ where $\boldsymbol{\xi}^\intercal\mathbf{A}^\intercal\boldsymbol{\xi} = (\boldsymbol{\xi}^\intercal\mathbf{A}^\intercal\boldsymbol{\xi})^\intercal = \boldsymbol{\xi}^\intercal\mathbf{A}\boldsymbol{\xi} = -(\boldsymbol{\xi}^\intercal\mathbf{A}^\intercal\boldsymbol{\xi})$ since $\mathbf{A} = -\mathbf{A}^\intercal$ by anti-symmetry. It follows that $\boldsymbol{\xi}^\intercal\mathbf{A}^\intercal\boldsymbol{\xi} = 0$.

For the third statement, gradient descent on $\mathcal{H}$ will converge to a point where $\nabla\mathcal{H} = \mathbf{J}^\intercal\boldsymbol{\xi}(\mathbf{w}) = 0$. If the Jacobian is invertible then clearly $\boldsymbol{\xi}(\mathbf{w}) = 0$. The fixed-point is stable since $0 \equiv \mathbf{S} \succeq 0$ in a Hamiltonian game, recall remark 1. ■

In fact, $\mathcal{H}$ is a Hamiltonian function for the game dynamics, see appendix B for a concise explanation. We use the notation $\mathcal{H}(\mathbf{w}) = \frac{1}{2}\|\boldsymbol{\xi}(\mathbf{w})\|^2$ throughout the paper. However, $\mathcal{H}$ can only be interpreted as a Hamiltonian function for $\boldsymbol{\xi}$ when the game is Hamiltonian.

There is a precise mapping from Hamiltonian games to symplectic geometry, see appendix B. Symplectic geometry is the modern formulation of classical mechanics (Arnold, 1989; Guillemin and Sternberg, 1990). Recall that periodic behaviors (e.g. orbits) often arise in classical mechanics. The orbits lie on the level sets of the Hamiltonian, which expresses the total energy of the system.

3. Algorithms

We have seen that fixed points of potential and Hamiltonian games can be found by descent on $\boldsymbol{\xi}$ and $\nabla\mathcal{H}$ respectively. This Section tackles finding stable fixed points in general games.

3.1. Finding Stable Fixed Points

There are two classes of games where we know how to find stable fixed points: potential games where $\boldsymbol{\xi}$ converges to a local minimum and Hamiltonian games where $\nabla\mathcal{H}$, which is orthogonal to $\boldsymbol{\xi}$, finds stable fixed points.

In the general case, the following desiderata provide a set of reasonable properties for an adjustment $\boldsymbol{\xi}_\lambda$ of the game dynamics. Recall that $\theta(\mathbf{u}, \mathbf{v})$ is the angle between the vectors $\mathbf{u}$ and $\mathbf{v}$.

3.1.1. Desiderata

To find stable fixed points, an adjustment $\boldsymbol{\xi}_\lambda$ to the game dynamics should satisfy

D1. *compatible*[3] *with game dynamics:* $\langle\boldsymbol{\xi}_\lambda, \boldsymbol{\xi}\rangle = \alpha_1 \cdot \|\boldsymbol{\xi}\|^2$;

D2. *compatible with potential dynamics:*
if the game is a potential game then $\langle\boldsymbol{\xi}_\lambda, \nabla\phi\rangle = \alpha_2 \cdot \|\nabla\phi\|^2$;

D3. *compatible with Hamiltonian dynamics:*
If the game is Hamiltonian then $\langle\boldsymbol{\xi}_\lambda, \nabla\mathcal{H}\rangle = \alpha_3 \cdot \|\nabla\mathcal{H}\|^2$;

D4. *attracted to stable equilibria:*
in neighborhoods where $\mathbf{S} \succ 0$, require $\theta(\boldsymbol{\xi}_\lambda, \nabla\mathcal{H}) \leq \theta(\boldsymbol{\xi}, \nabla\mathcal{H})$;

D5. *repelled by unstable equilibria:*
in neighborhoods where $\mathbf{S} \prec 0$, require $\theta(\boldsymbol{\xi}_\lambda, \nabla\mathcal{H}) \geq \theta(\boldsymbol{\xi}, \nabla\mathcal{H})$.

for some $\alpha_1, \alpha_2, \alpha_3 > 0$.

Desideratum $D1$ does not guarantee that players act in their own self-interest—this requires a stronger positivity condition on dot-products with subvectors of $\boldsymbol{\xi}$, see Balduzzi (2017). Desiderata $D2$ and $D3$ imply that the adjustment behaves correctly in potential and Hamiltonian games respectively.

To understand desiderata $D4$ and $D5$, observe that gradient descent on $\mathcal{H} = \frac{1}{2}\|\boldsymbol{\xi}\|^2$ will find local minima that are fixed points of the dynamics. However, we specifically wish to converge to stable fixed points. Desideratum $D4$ and $D5$ require that the adjustment improves the rate of convergence to stable fixed points (by finding a steeper angle of descent), and avoids unstable fixed points.

More concretely, desiderata $D4$ can be interpreted as follows. If $\boldsymbol{\xi}$ points at a stable equilibrium then we require that $\boldsymbol{\xi}_\lambda$ points *more* towards the equilibrium (i.e. has smaller angle). Conversely, desiderata $D5$ requires that if $\boldsymbol{\xi}$ points away then the adjustment should point *further* away.

The unadjusted dynamics $\boldsymbol{\xi}$ satisfies all the desiderata except $D3$.

3. Two nonzero vectors are compatible if they have positive inner product.

3.2. Consensus Optimization

Since gradient descent on the function $\mathcal{H}(\mathbf{w}) = \frac{1}{2}\|\boldsymbol{\xi}\|^2$ finds stable fixed points in Hamiltonian games, it is natural to ask how it performs in general games. If the Jacobian $\mathbf{J}(\mathbf{w})$ is invertible, then $\nabla\mathcal{H} = \mathbf{J}^\intercal\boldsymbol{\xi} = 0$ iff $\boldsymbol{\xi} = 0$. Thus, gradient descent on $\mathcal{H}$ converges to fixed points of $\boldsymbol{\xi}$.

However, there is no guarantee that descent on $\mathcal{H}$ will find a *stable* fixed point. Mescheder et al. (2017) propose *consensus optimization*, a gradient adjustment of the form

$$\boldsymbol{\xi} + \lambda \cdot \mathbf{J}^\intercal\boldsymbol{\xi} = \boldsymbol{\xi} + \lambda \cdot \nabla\mathcal{H}.$$

Unfortunately, consensus optimization can converge to unstable fixed points even in simple cases where the 'game' is to minimize a single function:

Example 8 (consensus optimization can converge to a global maximum) *Consider a potential game with losses $\ell_1(x, y) = \ell_2(x, y) = -\frac{\kappa}{2}(x^2 + y^2)$ with $\kappa \gg 0$. Then*

$$\boldsymbol{\xi} = -\kappa \cdot \begin{pmatrix} x \\ y \end{pmatrix} \quad \textit{and} \quad \mathbf{J} = -\begin{pmatrix} \kappa & 0 \\ 0 & \kappa \end{pmatrix}$$

Note that $\|\boldsymbol{\xi}\|^2 = \kappa^2(x^2 + y^2)$ and

$$\boldsymbol{\xi} + \lambda \cdot \mathbf{J}^\intercal\boldsymbol{\xi} = \kappa(\lambda\kappa - 1) \cdot \begin{pmatrix} x \\ y \end{pmatrix}.$$

Descent on $\boldsymbol{\xi} + \lambda \cdot \mathbf{J}^\intercal\boldsymbol{\xi}$ converges to the global maximum $(x, y) = (0, 0)$ unless $\lambda < \frac{1}{\kappa}$.

Although consensus optimization works well in two-player zero-sum, it cannot be considered a candidate algorithm for finding stable fixed points in general games since it fails in the basic case of potential games. Consensus optimization only satisfies desiderata $D3$ and $D4$.

3.3. Symplectic Gradient Adjustment

The problem with consensus optimization is that it can perform worse than gradient descent on potential games. Intuitively, it makes bad use of the symmetric component of the Jacobian. Motivated by the analysis in Section 2, we propose symplectic gradient adjustment, which takes care to only use the antisymmetric component of the Jacobian when adjusting the dynamics.

Proposition 5 *The **symplectic gradient adjustment (SGA)***

$$\boldsymbol{\xi}_\lambda := \boldsymbol{\xi} + \lambda \cdot \mathbf{A}^\intercal\boldsymbol{\xi}.$$

satisfies $D1$—$D3$ for $\lambda > 0$, with $\alpha_1 = 1 = \alpha_2$ and $\alpha_3 = \lambda$.

Proof First claim: $\lambda \cdot \boldsymbol{\xi}^\intercal\mathbf{A}^\intercal\boldsymbol{\xi} = 0$ by anti-symmetry of $\mathbf{A}$. Second claim: $\mathbf{A} \equiv 0$ in a potential game, so $\boldsymbol{\xi}_\lambda = \boldsymbol{\xi} = \nabla\phi$. Third claim: $\langle\boldsymbol{\xi}_\lambda, \nabla\mathcal{H}\rangle = \langle\boldsymbol{\xi}_\lambda, \mathbf{J}^\intercal\boldsymbol{\xi}\rangle = \langle\boldsymbol{\xi}_\lambda, \mathbf{A}^\intercal\boldsymbol{\xi}\rangle = \lambda \cdot \boldsymbol{\xi}^\intercal\mathbf{A}\mathbf{A}^\intercal\boldsymbol{\xi} = \lambda \cdot \|\nabla\mathcal{H}\|^2$ since $\mathbf{J} = \mathbf{A}$ by assumption. ■

Note that desiderata $D1$ and $D2$ are true even when $\lambda < 0$. This will prove useful, since example 9 shows that it may be necessary to pick negative λ near $\mathbf{S} \prec 0$. Section 3.5 shows how to also satisfy desiderata $D4$ and $D5$.

3.4. Convergence

We begin by analysing convergence of SGA near stable equilibria. The following lemma highlights that the interaction between the symmetric and antisymmetric components is important for convergence. Recall that two matrices $\mathbf{A}$ and $\mathbf{S}$ *commute* iff $[\mathbf{A}, \mathbf{S}] := \mathbf{AS} - \mathbf{SA} = \mathbf{0}$. That is, $\mathbf{A}$ and $\mathbf{S}$ commute iff $\mathbf{AS} = \mathbf{SA}$. Intuitively, two matrices commute if they have the same preferred coordinate system.

Lemma 6 *If $\mathbf{S} \succeq 0$ is symmetric positive semidefinite and $\mathbf{S}$ commutes with $\mathbf{A}$ then $\boldsymbol{\xi}_\lambda$ points towards stable fixed points for non-negative λ:*

$$\langle\boldsymbol{\xi}_\lambda, \nabla\mathcal{H}\rangle \geq 0 \textit{ for all } \lambda \geq 0.$$

Proof First observe that $\boldsymbol{\xi}^\intercal\mathbf{AS}\boldsymbol{\xi} = \boldsymbol{\xi}^\intercal\mathbf{S}^\intercal\mathbf{A}^\intercal\boldsymbol{\xi} = -\boldsymbol{\xi}^\intercal\mathbf{SA}\boldsymbol{\xi}$, where the first equality holds since the expression is a scalar, and the second holds since $\mathbf{S} = \mathbf{S}^\intercal$ and $\mathbf{A} = -\mathbf{A}^\intercal$. It follows that $\boldsymbol{\xi}^\intercal\mathbf{AS}\boldsymbol{\xi} = 0$ if $\mathbf{SA} = \mathbf{AS}$. Finally rewrite the inequality as

$$\langle\boldsymbol{\xi}_\lambda, \nabla\mathcal{H}\rangle = \langle\boldsymbol{\xi} + \lambda \cdot \mathbf{A}^\intercal\boldsymbol{\xi}, \mathbf{S}\boldsymbol{\xi} + \mathbf{A}^\intercal\boldsymbol{\xi}\rangle = \boldsymbol{\xi}^\intercal\mathbf{S}\boldsymbol{\xi} + \lambda\boldsymbol{\xi}^\intercal\mathbf{A}\mathbf{A}^\intercal\boldsymbol{\xi} \geq 0$$

since $\boldsymbol{\xi}^\intercal\mathbf{AS}\boldsymbol{\xi} = 0$ and by positivity of $\mathbf{S}$, λ and $\mathbf{AA}^\intercal$. ■

The lemma suggests that in general the *failure* of $\mathbf{A}$ and $\mathbf{S}$ to commute should be important for understanding the dynamics of $\boldsymbol{\xi}_\lambda$. We therefore introduce the **additive condition number** κ to upper-bound the worst-case noncommutativity of $\mathbf{S}$, which allows to quantify the relationship between $\boldsymbol{\xi}_\lambda$ and $\nabla\mathcal{H}$. If $\kappa = 0$, then $\mathbf{S} = \sigma \cdot \mathbf{I}$ commutes with *all* matrices. The larger the additive condition number κ, the larger the *potential* failure of $\mathbf{S}$ to commute with other matrices.

Theorem 7 *Let $\mathbf{S}$ be a symmetric matrix with eigenvalues $\sigma_{max} \geq \cdots \geq \sigma_{min}$. The **additive condition number**[4] of $\mathbf{S}$ is $\kappa := \sigma_{max} - \sigma_{min}$. If $\mathbf{S} \succeq 0$ is positive semidefinite with additive condition number κ then $\lambda \in (0, \frac{4}{\kappa})$ implies*

$$\langle\boldsymbol{\xi}_\lambda, \nabla\mathcal{H}\rangle \geq 0.$$

If $\mathbf{S}$ is negative semidefinite, then $\lambda \in (0, \frac{4}{\kappa})$ implies

$$\langle\boldsymbol{\xi}_{-\lambda}, \nabla\mathcal{H}\rangle \leq 0.$$

The inequalities are strict if $\mathbf{J}$ is invertible.

Proof We prove the case $\mathbf{S} \succeq 0$; the case $\mathbf{S} \preceq 0$ is similar. Rewrite the inequality as

$$\begin{aligned}\langle\boldsymbol{\xi} + \lambda \cdot \mathbf{A}^\intercal\boldsymbol{\xi}, \nabla\mathcal{H}\rangle &= (\boldsymbol{\xi} + \lambda \cdot \mathbf{A}^\intercal\boldsymbol{\xi})^\intercal \cdot (\mathbf{S} + \mathbf{A}^\intercal)\boldsymbol{\xi} \\ &= \boldsymbol{\xi}^\intercal\mathbf{S}\boldsymbol{\xi} + \lambda\boldsymbol{\xi}^\intercal\mathbf{AS}\boldsymbol{\xi} + \lambda\boldsymbol{\xi}^\intercal\mathbf{A}\mathbf{A}^\intercal\boldsymbol{\xi}\end{aligned}$$

4. The condition number of a positive definite matrix is $\frac{\sigma_{\max}}{\sigma_{\min}}$.

Let $\beta = \|A^\intercal \boldsymbol{\xi}\|$ and $\tilde{\mathbf{S}} = \mathbf{S} - \sigma_{\min} \cdot \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Then

$$\boldsymbol{\xi}^\intercal \mathbf{S}\boldsymbol{\xi} + \lambda \boldsymbol{\xi}^\intercal \mathbf{A}\mathbf{S}\boldsymbol{\xi} + \lambda \cdot \beta^2 \geq \boldsymbol{\xi}^\intercal \tilde{\mathbf{S}}\boldsymbol{\xi} + \lambda \boldsymbol{\xi}^\intercal \mathbf{A}\tilde{\mathbf{S}}\boldsymbol{\xi} + \lambda \cdot \beta^2$$

since $\boldsymbol{\xi}^\intercal \mathbf{S}\boldsymbol{\xi} \geq \boldsymbol{\xi}^\intercal \tilde{\mathbf{S}}\boldsymbol{\xi}$ by construction and $\boldsymbol{\xi}^\intercal \mathbf{A}\tilde{\mathbf{S}}\boldsymbol{\xi} = \boldsymbol{\xi}^\intercal \mathbf{A}\mathbf{S}\boldsymbol{\xi} - \sigma_{\min}\boldsymbol{\xi}^\intercal \mathbf{A}\boldsymbol{\xi} = \boldsymbol{\xi}^\intercal \mathbf{A}\mathbf{S}\boldsymbol{\xi}$ because $\boldsymbol{\xi}^\intercal \mathbf{A}\boldsymbol{\xi} = 0$ by the anti-symmetry of $\mathbf{A}$. It therefore suffices to show that the inequality holds when $\sigma_{\min} = 0$ and $\kappa = \sigma_{\max}$.

Since $\mathbf{S}$ is positive semidefinite, there exists an upper-triangular square-root matrix T such that $\mathbf{T}^\intercal \mathbf{T} = \mathbf{S}$ and so $\boldsymbol{\xi}^\intercal \mathbf{S}\boldsymbol{\xi} = \|\mathbf{T}\boldsymbol{\xi}\|^2$. Further,

$$|\boldsymbol{\xi}^\intercal \mathbf{A}\mathbf{S}\boldsymbol{\xi}| \leq \|\mathbf{A}^\intercal \boldsymbol{\xi}\| \cdot \|\mathbf{T}^\intercal \mathbf{T}\boldsymbol{\xi}\| \leq \sqrt{\sigma_{\max}} \cdot \|\mathbf{A}^\intercal \boldsymbol{\xi}\| \cdot \|\mathbf{T}\boldsymbol{\xi}\|.$$

since $\|\mathbf{T}\|_2 = \sqrt{\sigma_{\max}}$. Putting the observations together obtains

$$\begin{aligned}
\|\mathbf{T}\boldsymbol{\xi}\|^2 + \lambda(\|\mathbf{A}\boldsymbol{\xi}\|^2 - \langle \mathbf{A}\boldsymbol{\xi}, \mathbf{S}\boldsymbol{\xi}\rangle) &\geq \|\mathbf{T}\boldsymbol{\xi}\|^2 + \lambda(\|\mathbf{A}\boldsymbol{\xi}\|^2 - \|\mathbf{A}\boldsymbol{\xi}\| \, \|\mathbf{S}\boldsymbol{\xi}\| \\
&\geq \|\mathbf{T}\boldsymbol{\xi}\|^2 + \lambda\|\mathbf{A}\boldsymbol{\xi}\|(\|\mathbf{A}\boldsymbol{\xi}\| - \|\mathbf{S}\boldsymbol{\xi}\|) \\
&\geq \|\mathbf{T}\boldsymbol{\xi}\|^2 + \lambda\|\mathbf{A}\boldsymbol{\xi}\|(\|\mathbf{A}\boldsymbol{\xi}\| - \sqrt{\sigma_{max}}\|\mathbf{T}\boldsymbol{\xi}\|)
\end{aligned}$$

Set $\alpha = \sqrt{\lambda}$ and $\eta = \sqrt{\sigma_{max}}$. We can continue the above computation

$$\begin{aligned}
\|\mathbf{T}\boldsymbol{\xi}\|^2 + \lambda(\|\mathbf{A}\boldsymbol{\xi}\|^2 - \langle \mathbf{A}\boldsymbol{\xi}, \mathbf{S}\boldsymbol{\xi}\rangle) &\geq \|\mathbf{T}\boldsymbol{\xi}\|^2 + \alpha^2\|\mathbf{A}\boldsymbol{\xi}\|(\|\mathbf{A}\boldsymbol{\xi}\| - \eta\|\mathbf{T}\boldsymbol{\xi}\|) \\
&= \|\mathbf{T}\boldsymbol{\xi}\|^2 + \alpha^2\|\mathbf{A}\boldsymbol{\xi}\|^2 - \alpha^2\|\mathbf{A}\boldsymbol{\xi}\|\eta\|\mathbf{T}\boldsymbol{\xi}\| \\
&= (\|\mathbf{T}\boldsymbol{\xi}\| - \alpha\|\mathbf{A}\boldsymbol{\xi}\|)^2 + 2\alpha\|\mathbf{A}\boldsymbol{\xi}\| \, \|\mathbf{T}\boldsymbol{\xi}\| - \alpha^2\eta\|A\boldsymbol{\xi}\| \, \|\mathbf{T}\boldsymbol{\xi}\| \\
&= (\|\mathbf{T}\boldsymbol{\xi}\| - \alpha\|\mathbf{A}\boldsymbol{\xi}\|)^2 + \|\mathbf{A}\boldsymbol{\xi}\| \, \|\mathbf{T}\boldsymbol{\xi}\|(2\alpha - \alpha^2\eta)
\end{aligned}$$

Finally, $2\alpha - \alpha^2\eta > 0$ for any α in the range $(0, \frac{2}{\eta})$, which is to say, for any $0 < \lambda < \frac{4}{\sigma_{max}}$. The kernel of $\mathbf{S}$ and the kernel of $\mathbf{T}$ coincide. If $\boldsymbol{\xi}$ is in the kernel of $\mathbf{A}$, resp. $\mathbf{T}$, it cannot be in the kernel of $\mathbf{T}$, resp. $\mathbf{A}$ and the term $(\|\mathbf{T}\boldsymbol{\xi}\| - \alpha\|\mathbf{A}\boldsymbol{\xi}\|)^2$ is positive. Otherwise, the term $\|\mathbf{A}\boldsymbol{\xi}\|\|\mathbf{T}\boldsymbol{\xi}\|$ is positive. ■

The theorem above guarantees that SGA always points in the direction of stable fixed points for λ sufficiently small. This does not technically guarantee convergence; we use Ostrowski's theorem to strengthen this formally. Applying Ostrowski's theorem will require taking a more abstract perspective by encoding the adjusted dynamics into a differentiable map $F : \Omega \to \mathbb{R}^d$ of the form $F(\mathbf{w}) = \mathbf{w} - \alpha\boldsymbol{\xi}_\lambda(\mathbf{w})$.

Theorem 8 (Ostrowski) *Let $F : \Omega \to \mathbb{R}^d$ be a continuously differentiable map on an open subset $\Omega \subseteq \mathbb{R}^d$, and assume $\mathbf{w}^* \in \Omega$ is a fixed point. If all eigenvalues of $\nabla F(\mathbf{w}^*)$ are strictly in the unit circle of $\mathbb{C}$, then there is an open neighbourhood U of $\mathbf{w}^*$ such that for all $\mathbf{w}_0 \in U$, the sequence $F^k(\mathbf{w}_0)$ of iterates of F converges to $\mathbf{w}^*$. Moreover, the rate of convergence is at least linear in k.*

Proof This is a standard result on fixed-point iterations, adapted from Ortega and Rheinboldt (2000, 10.1.3). ■

Corollary 9 *A matrix $\mathbf{M}$ is called positive stable if all its eigenvalues have positive real part. Assume $\mathbf{w}^*$ is a fixed point of a differentiable game such that $(\mathbf{I} + \lambda\mathbf{A}^\intercal)\mathbf{J}(\mathbf{w}^*)$ is positive stable for λ in some set Λ. Then SGA converges locally to $\mathbf{w}^*$ for $\lambda \in \Lambda$ and $\alpha > 0$ sufficiently small.*

Proof Let $X = (\mathbf{I} + \lambda\mathbf{A}^\intercal)$. By definition of fixed points, $\boldsymbol{\xi}(\mathbf{w}^*) = 0$ and so

$$\nabla[X\boldsymbol{\xi}](\mathbf{w}^*) = \nabla X(\mathbf{w}^*)\boldsymbol{\xi}(\mathbf{w}^*) + X(\mathbf{w}^*)\nabla\boldsymbol{\xi}(\mathbf{w}^*) = X\mathbf{J}(\mathbf{w}^*)$$

is positive stable by assumption, namely has eigenvalues $a_k + ib_k$ with $a_k > 0$. Writing $F(\mathbf{w}) = \mathbf{w} - \alpha X\boldsymbol{\xi}(\mathbf{w})$ for the iterative procedure given by SGA, it follows that

$$\nabla F(\mathbf{w}^*) = \mathbf{I} - \alpha\nabla[X\boldsymbol{\xi}](\mathbf{w}^*)$$

has eigenvalues $1 - \alpha a_k - i\alpha b_k$, which are in the unit circle for small α. More precisely,

$$\begin{aligned}
|1 - \alpha a_k - i\alpha b_k|^2 < 1 \quad &\iff \quad 1 - 2\alpha a_k + \alpha^2 a_k^2 + \alpha^2 b_k^2 < 1 \\
&\iff \quad 0 < \alpha < \frac{2a_k}{a_k^2 + b_k^2}
\end{aligned}$$

which is always possible for $a_k > 0$. Hence $\nabla F(\mathbf{w}^*)$ has eigenvalues in the unit circle for $0 < \alpha < \min_k 2a_k/(a_k^2 + b_k^2)$, and we are done by Ostrowski's Theorem since $\mathbf{w}^*$ is a fixed point of F. ■

Theorem 10 *Let $\mathbf{w}^*$ be a stable fixed point and κ the additive condition number of $\mathbf{S}(\mathbf{w}^*)$. Then SGA converges locally to $\mathbf{w}^*$ for all $\lambda \in (0, \frac{4}{\kappa})$ and $\alpha > 0$ sufficiently small.*

Proof By Theorem 5 and the assumption that $\mathbf{w}^*$ is a stable fixed point with invertible Jacobian, we know that

$$\langle \boldsymbol{\xi}_\lambda, \nabla\mathcal{H}\rangle = \langle (\mathbf{I} + \lambda\mathbf{A}^\intercal)\boldsymbol{\xi}, \mathbf{J}^\intercal\boldsymbol{\xi}\rangle > 0$$

for $\lambda \in (0, \frac{4}{\kappa})$. The proof does not rely on any particular property of $\boldsymbol{\xi}$, and can trivially be extended to the claim that

$$\langle (\mathbf{I} + \lambda\mathbf{A}^\intercal)\mathbf{u}, \mathbf{J}^\intercal\mathbf{u}\rangle > 0$$

for all non-zero vectors $\mathbf{u}$. In particular this can be rewritten as

$$\mathbf{u}^\intercal\mathbf{J}(\mathbf{I} + \lambda\mathbf{A}^\intercal)\mathbf{u}\rangle > 0\,,$$

which implies positive definiteness of $\mathbf{J}(\mathbf{I} + \lambda\mathbf{A}^\intercal)$. A positive definite matrix is positive stable, and any matrices AB and BA have identical spectrum. This implies also that $(\mathbf{I} + \lambda\mathbf{A}^\intercal)\mathbf{J}$ is positive stable, and we are done by the corollary above. ■

We conclude that SGA converges to an SFP if λ is small enough, where 'small enough' depends on the additive condition number.

3.5. Picking $\text{sign}(\lambda)$

This section explains desiderata $D4$—$D5$ and shows how to pick $\text{sign}(\lambda)$ to speed up convergence towards stable and away from unstable fixed points. In the example below, almost any choice of positive λ results in convergence to an unstable equilibrium. The problem arises from the combination of a weak repellor with a strong rotational force.

Example 9 (failure case for $\lambda > 0$)
Suppose $\epsilon > 0$ is small and

$$\ell_1(x, y) = -\frac{\epsilon}{2}x^2 - xy \;\; \textit{and} \;\; \ell_2(x, y) = -\frac{\epsilon}{2}y^2 + xy$$

with an unstable equilibrium at $(0, 0)$. The dynamics are

$$\boldsymbol{\xi} = \epsilon \cdot \begin{pmatrix} -x \\ -y \end{pmatrix} + \begin{pmatrix} -y \\ x \end{pmatrix} \quad \textit{with} \quad \mathbf{A} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

and

$$\mathbf{A}^\intercal \boldsymbol{\xi} = \begin{pmatrix} x \\ y \end{pmatrix} + \epsilon \begin{pmatrix} -y \\ x \end{pmatrix}$$

Finally observe that

$$\boldsymbol{\xi} + \lambda \cdot \mathbf{A}^\intercal \boldsymbol{\xi} = (\lambda - \epsilon) \cdot \begin{pmatrix} x \\ y \end{pmatrix} + (1 + \epsilon\lambda) \cdot \begin{pmatrix} -y \\ x \end{pmatrix}$$

which converges to the unstable equilibrium if $\lambda > \epsilon$.

We now show how to pick the sign of λ to avoid unstable equilibria. First, observe that $\langle \boldsymbol{\xi}, \nabla\mathcal{H} \rangle = \boldsymbol{\xi}^\intercal(\mathbf{S} + \mathbf{A})^\intercal \boldsymbol{\xi} = \boldsymbol{\xi}^\intercal \mathbf{S} \boldsymbol{\xi}$. It follows that for $\boldsymbol{\xi} \neq 0$:

$$\begin{cases} \text{if} \quad \mathbf{S} \succeq 0 \quad \text{then } \langle \boldsymbol{\xi}, \nabla\mathcal{H} \rangle \geq 0; \\ \text{if} \quad \mathbf{S} \prec 0 \quad \text{then } \langle \boldsymbol{\xi}, \nabla\mathcal{H} \rangle < 0. \end{cases} \tag{2}$$

A criterion to probe the positive/negative definiteness of $\mathbf{S}$ is thus to check the sign of $\langle \boldsymbol{\xi}, \nabla\mathcal{H} \rangle$. The dot product can take any value if $\mathbf{S}$ is neither positive nor negative (semi-)definite. The behavior near saddle points will be explored in Section 3.7.

Recall that desiderata $D4$ requires that, if $\boldsymbol{\xi}$ points at a stable equilibrium then we require that $\boldsymbol{\xi}_\lambda$ points *more* towards the equilibrium (i.e. has smaller angle). Conversely, desiderata $D5$ requires that, if $\boldsymbol{\xi}$ points away then the adjustment should point *further* away. More formally,

Definition 6 *Let $\mathbf{u}$ and $\mathbf{v}$ be two vectors. The **infinitesimal alignment** of $\boldsymbol{\xi}_\lambda := \mathbf{u} + \lambda \cdot \mathbf{v}$ with a third vector $\mathbf{w}$ is*

$$\text{align}(\boldsymbol{\xi}_\lambda, \mathbf{w}) := \frac{d}{d\lambda} \left\{ \cos^2 \theta_\lambda \right\}_{|\lambda=0} \;\; \textit{for} \;\; \theta_\lambda := \theta(\boldsymbol{\xi}_\lambda, \mathbf{w}).$$

If $\mathbf{u}$ and $\mathbf{w}$ point the same way, $\mathbf{u}^\intercal \mathbf{w} > 0$, then align > 0 when $\mathbf{v}$ bends $\mathbf{u}$ further toward $\mathbf{w}$, see Figure 2A. Otherwise align > 0 when $\mathbf{v}$ bends $\mathbf{u}$ away from $\mathbf{w}$, see Figure 2B.

The following lemma allows us to rewrite the infinitesimal alignment in terms of known (computable) quantities, from which we can deduce the correct choice of λ.

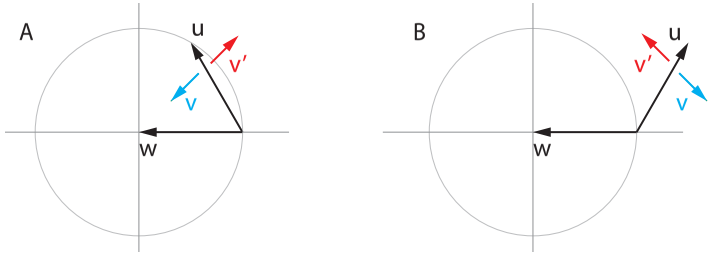


Figure 2: *Infinitesimal alignment* between $\mathbf{u} + \lambda\mathbf{v}$ and $\mathbf{w}$ is positive (cyan) when small positive λ either: **(A)** pulls $\mathbf{u}$ toward $\mathbf{w}$, if $\mathbf{w}$ and $\mathbf{u}$ have angle $< 90°$; or **(B)** pushes $\mathbf{u}$ away from $\mathbf{w}$ if their angle is $> 90°$. Conversely, the infinitesimal alignment is negative (red) when small positive λ either: **(A)** pushes $\mathbf{u}$ away from $\mathbf{w}$ when their angle is acute or **(B)** pulls $\mathbf{u}$ toward $\mathbf{w}$ when their angle is obtuse.

Algorithm 1 Symplectic Gradient Adjustment

Input: losses $\mathcal{L} = \{\ell_i\}_{i=1}^n$, weights $\mathcal{W} = \{\mathbf{w}_i\}_{i=1}^n$
$\boldsymbol{\xi} \leftarrow \left[\texttt{gradient}(\ell_i, \mathbf{w}_i) \;\; \texttt{for} \;\; (\ell_i, \mathbf{w}_i) \in (\mathcal{L}, \mathcal{W})\right]$
$\mathbf{A}^\intercal\boldsymbol{\xi} \leftarrow \texttt{get_sym_adj}(\mathcal{L}, \mathcal{W})$ // appendix A
if `align` **then**
  $\nabla\mathcal{H} \leftarrow \left[\texttt{gradient}(\frac{1}{2}\|\boldsymbol{\xi}\|^2, \mathbf{w}) \;\; \texttt{for} \;\; \mathbf{w} \in \mathcal{W})\right]$
  $\lambda \leftarrow \texttt{sign}\Big(\frac{1}{d}\langle \boldsymbol{\xi}, \nabla\mathcal{H} \rangle \langle \mathbf{A}^\intercal\boldsymbol{\xi}, \nabla\mathcal{H} \rangle + \epsilon\Big)$ // $\epsilon = \frac{1}{10}$
else
  $\lambda \leftarrow 1$
end if
Output: $\boldsymbol{\xi} + \lambda \cdot \mathbf{A}^\intercal\boldsymbol{\xi}$ // plug into any optimizer

Lemma 11 *When $\boldsymbol{\xi}_\lambda$ is the symplectic gradient adjustment,*

$$\text{sign}\left(\text{align}(\boldsymbol{\xi}_\lambda, \nabla\mathcal{H})\right) = \text{sign}\left(\langle \boldsymbol{\xi}, \nabla\mathcal{H} \rangle \cdot \langle \mathbf{A}^\intercal\boldsymbol{\xi}, \nabla\mathcal{H} \rangle\right).$$

Proof Observe that

$$\cos^2\theta_\lambda = \left(\frac{\langle \boldsymbol{\xi}_\lambda, \nabla\mathcal{H} \rangle}{\|\boldsymbol{\xi}_\lambda\| \cdot \|\nabla\mathcal{H}\|}\right)^2 = \frac{\langle \boldsymbol{\xi}, \nabla\mathcal{H} \rangle + 2\lambda \langle \boldsymbol{\xi}, \nabla\mathcal{H} \rangle \langle \mathbf{A}^\intercal\boldsymbol{\xi}, \nabla\mathcal{H} \rangle + O(\lambda^2)}{\left(\|\boldsymbol{\xi}\|^2 + O(\lambda^2)\right) \cdot \|\nabla\mathcal{H}\|^2}$$

where the denominator has no linear term in λ because $\boldsymbol{\xi} \perp \mathbf{A}^\intercal\boldsymbol{\xi}$. It follows that the sign of the infinitesimal alignment is

$$\text{sign}\left\{\frac{d}{d\lambda}\cos^2\theta_\lambda\right\} = \text{sign}\left\{\langle \boldsymbol{\xi}, \nabla\mathcal{H} \rangle \langle \mathbf{A}^\intercal\boldsymbol{\xi}, \nabla\mathcal{H} \rangle\right\}$$

as required. ■

Intuitively, computing the sign of $\langle \boldsymbol{\xi}, \nabla\mathcal{H}\rangle$ provides a check for stable and unstable fixed points. Computing the sign of $\langle \mathbf{A}^\intercal\boldsymbol{\xi}, \nabla\mathcal{H}\rangle$ checks whether the adjustment term points towards or away from the nearby fixed point. Putting the two checks together yields a prescription for the sign of λ, as follows.

Proposition 12 *Desiderata D4—D5 are satisfied for λ such that $\lambda \cdot \langle \boldsymbol{\xi}, \nabla\mathcal{H}\rangle \cdot \langle \mathbf{A}^\intercal\boldsymbol{\xi}, \nabla\mathcal{H}\rangle \geq 0$.*

Proof If we are in a neighborhood of a stable fixed point then $\langle \boldsymbol{\xi}, \nabla\mathcal{H}\rangle \geq 0$. It follows by Lemma 11 that $\text{sign}\left(\text{align}(\boldsymbol{\xi}_\lambda), \nabla\mathcal{H}\right) = \text{sign}\left(\langle \mathbf{A}^\intercal\boldsymbol{\xi}, \nabla\mathcal{H}\rangle\right)$ and so choosing $\text{sign}(\lambda) = \text{sign}\left(\langle \mathbf{A}^\intercal\boldsymbol{\xi}, \nabla\mathcal{H}\rangle\right)$ leads to the angle between $\boldsymbol{\xi}_\lambda$ and $\nabla\mathcal{H}$ being smaller than the angle between $\boldsymbol{\xi}$ and $\nabla\mathcal{H}$, satisfying desideratum $D4$. The proof for the unstable case is similar. ■

3.5.1. Alignment and Convergence Rates

Gradient descent is also known as the method of steepest descent. In general games, however, $\boldsymbol{\xi}$ does not follow the steepest path to fixed points due to the 'rotational force', which forces lower learning rates and slows down convergence.

The following lemma provides some intuition about alignment. The idea is that, the smaller the cosine between the 'correct direction' $\mathbf{w}$ and the 'update direction' $\boldsymbol{\xi}$, the smaller the learning rate needs to be for the update to stay in a unit ball, see Figure 3.

Lemma 13 (Alignment Lemma)
If $\mathbf{w}$ and $\boldsymbol{\xi}$ are unit vectors with $0 < \mathbf{w}^\intercal\boldsymbol{\xi}$ then $\|\mathbf{w} - \eta\cdot\boldsymbol{\xi}\| \leq 1$ for $0 \leq \eta \leq 2\mathbf{w}^\intercal\boldsymbol{\xi} = 2\cos\theta(\mathbf{w}, \boldsymbol{\xi})$. In other words, ensuring that $\mathbf{w} - \eta\boldsymbol{\xi}$ is closer to the origin than $\mathbf{w}$ requires smaller learning rates η as the angle between $\mathbf{w}$ and $\boldsymbol{\xi}$ gets larger.

Proof Check $\|\mathbf{w} - \eta\cdot\boldsymbol{\xi}\|^2 = 1 + \eta^2 - 2\eta\cdot\mathbf{w}^\intercal\boldsymbol{\xi} \leq 1$ iff $\eta^2 \leq 2\eta\cdot\mathbf{w}^\intercal\boldsymbol{\xi}$. The result follows. ■

The next lemma is a standard technical result from the convex optimization literature.

Lemma 14 *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex Lipschitz smooth function satisfying $\|\nabla f(\mathbf{y}) - \nabla f(\mathbf{x})\| \leq L\cdot\|\mathbf{y} - \mathbf{x}\|$ for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$. Then*

$$|f(\mathbf{y}) - f(\mathbf{x}) - \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x}\rangle| \leq \frac{L}{2}\cdot\|\mathbf{y} - \mathbf{x}\|^2$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$.

Proof See Nesterov (2004). ■

Finally, we show that increasing alignment helps speed convergence:

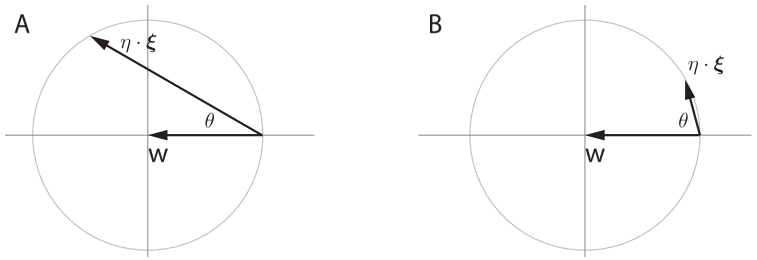


Figure 3: *Alignment and learning rates.* The larger $\cos\theta$, the larger the learning rate η that can be applied to unit vector $\boldsymbol{\xi}$ without $\mathbf{w} + \eta\cdot\boldsymbol{\xi}$ leaving the unit circle.

Theorem 15 *Suppose f is convex and Lipschitz smooth with $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\cdot\|\mathbf{x} - \mathbf{y}\|$. Let $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\cdot\mathbf{v}$ where $\|\mathbf{v}\| = \|\nabla f(\mathbf{w}_t)\|$. Then the optimal step size is $\eta^* = \frac{\cos\theta}{L}$ where $\theta := \theta(\nabla f(\mathbf{w}_t), \mathbf{v})$, with*

$$f(\mathbf{w}_{t+1}) \leq f(\mathbf{w}_t) - \frac{\cos^2\theta}{2L}\cdot\|\nabla f(\mathbf{w}_t)\|^2.$$

The proof of Theorem 15 adapts Lemma 14 to handle the angle arising from the 'rotational force'.

Proof By the Lemma 14,

$$\begin{aligned} f(\mathbf{y}) &\leq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x}\rangle + \frac{L}{2}\|\mathbf{y} - \mathbf{x}\|^2 \\ &= f(\mathbf{x}) - \eta\cdot\langle \nabla f, \boldsymbol{\xi}\rangle + \eta^2\frac{L}{2}\cdot\|\boldsymbol{\xi}\|^2 \\ &= f(\mathbf{x}) - \eta\cdot\langle \nabla f, \boldsymbol{\xi}\rangle + \eta^2\frac{L}{2}\cdot\|\nabla f\|^2 \\ &= f(\mathbf{x}) - \eta(\alpha - \frac{\eta}{2}L)\cdot\|\nabla f\|^2 \end{aligned}$$

where $\alpha := \cos\theta$. Solve

$$\min_\eta \Delta(\eta) = \min_\eta \left\{-\eta(\alpha - \frac{\eta}{2}L)\right\}$$

to obtain $\eta^* = \frac{\alpha}{L}$ and $\Delta(\eta^*) = -\frac{\alpha^2}{2}L$ as required. ■

Increasing the cosine with the steepest direction improves convergence. The alignment computation in algorithm 1 chooses λ to be positive or negative such that $\boldsymbol{\xi}_\lambda$ is bent towards stable (increasing the cosine) and away from unstable fixed points. Adding a small $\epsilon > 0$ to the computation introduces a weak bias towards stable fixed points.

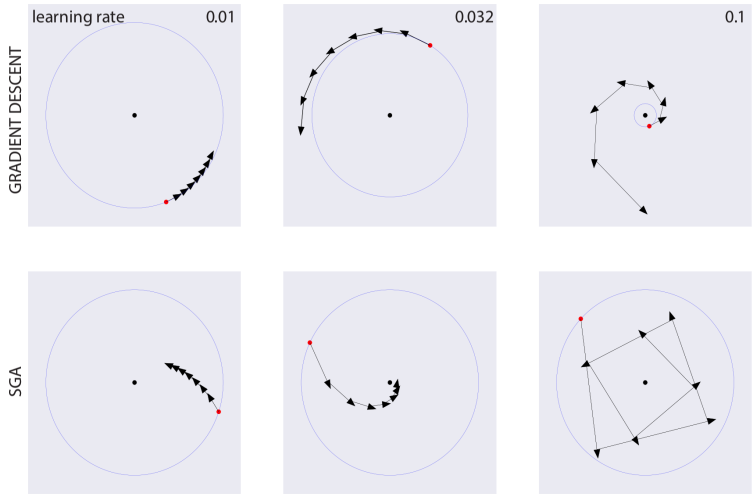


Figure 4: SGA allows faster and more robust convergence to stable fixed points than vanilla gradient descent in the presence of 'rotational forces', by bending the direction of descent towards the fixed point. Note the gradient descent diverges extremely rapidly in the top-right panel, which has a different scale from the other panels.

3.6. Aligned Consensus Optimization

The stability criterion in Equation (2) also provides a simple way to prevent consensus optimization from converging to unstable equilibria. **Aligned consensus optimization** is

$$\boldsymbol{\xi} + |\lambda| \cdot \operatorname{sign}\left(\langle \boldsymbol{\xi}, \nabla \mathcal{H} \rangle\right) \cdot \mathbf{J}^\intercal \boldsymbol{\xi}, \tag{3}$$

where in practice we set $\lambda = 1$. Aligned consensus optimization satisfies desiderata *D3*—*D5*. However, it behaves strangely in potential games. Multiplying by the Jacobian is the 'inverse' of Newton's method since for potential games the Jacobian of $\boldsymbol{\xi}$ is the Hessian of the potential function. Multiplying by the Hessian increases the gap between small and large eigenvalues, increasing the (usual, multiplicative) condition number and slows down convergence. Nevertheless, consensus optimization works well in GANs (Mescheder et al., 2017), and aligned consensus may improve performance, see experiments below.

Dropping the first term $\boldsymbol{\xi}$ from Equation (3) yields a simpler update that also satisfies *D3*—*D5*. However, the resulting algorithm performs poorly in experiments (not shown), perhaps because it is attracted to saddles.

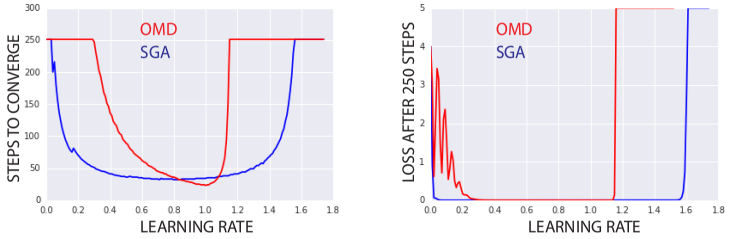


Figure 5: Comparison of SGA with optimistic mirror descent. The plots sweep over learning rates in range $[0.01, 1.75]$, with $\lambda = 1$ throughout for SGA. **(Left):** iterations to convergence, with maximum value of 250 after which the run was interrupted. **(Right):** average absolute value of losses over the last 10 iterations, 240-250, with a cutoff at 5.

3.7. Avoiding Strict Saddles

How does SGA behave near saddles? We show that Symplectic Gradient Adjustment locally avoids strict saddles, provided that λ and α are small and parameters are initialized with (arbitrarily small) noise. More precisely, let $\mathbf{F}(\mathbf{w}) = \mathbf{w} - \alpha \boldsymbol{\xi}_\lambda(\mathbf{w})$ be the iterative optimization procedure given by SGA. Then every strict saddle $\mathbf{w}^*$ has a neighbourhood U such that $\{\mathbf{w} \in U \mid \mathbf{F}^n(\mathbf{w}) \to \mathbf{w}^* \text{ as } n \to \infty\}$ has measure zero for small $\alpha > 0$ and λ.

Intuitively, the Taylor expansion around a strict saddle $\mathbf{w}^*$ is locally dominated by the Jacobian at $\mathbf{w}^*$, which has a negative eigenvalue. This prevents convergence to $\mathbf{w}^*$ for random initializations of $\mathbf{w}$ near $\mathbf{w}^*$. The argument is made rigorous using the Stable Manifold Theorem following Lee et al. (2017).

Theorem 16 (Stable Manifold Theorem)
Let $\mathbf{w}^*$ be a fixed point for the C^1 local diffeomorphism $F : U \to \mathbb{R}^d$, where U is a neighbourhood of $\mathbf{w}^*$ in $\mathbb{R}^d$. Let $E^s \oplus E^u$ be the generalized eigenspaces of $\nabla F(\mathbf{w}^*)$ corresponding to eigenvalues with $|\sigma| \leq 1$ and $|\sigma| > 1$ respectively. Then there exists a local stable center manifold W with tangent space E^s at $\mathbf{w}^*$ and a neighbourhood B of $\mathbf{w}^*$ such that $F(W) \cap B \subset W$ and $\cap_{n=0}^{\infty} F^{-n}(B) \subset W$.

Proof See Shub (2000). ■

It follows that if $\nabla F(\mathbf{w}^*)$ has at least one eigenvalue $|\sigma| > 1$ then E^u has dimension at least 1. Since W has tangent space E^s at $\mathbf{w}^*$ with codimension at least one, we conclude that W has measure zero. This is central to proving that the set of nearby initial points which converge to a given strict saddle $\mathbf{w}^*$ has measure zero. Since $\mathbf{w}$ is initialized randomly, the following theorem is obtained.

Theorem 17 *SGA locally avoids strict saddles almost surely, for $\alpha > 0$ and λ small.*

Proof Let $\mathbf{w}^*$ a strict saddle and recall that SGA is given by

$$F(\mathbf{w}) = \mathbf{w} - \alpha(\mathbf{I} - \alpha\mathbf{J})\boldsymbol{\xi}(\mathbf{w})\,.$$

All terms involved are continuously differentiable and we have

$$\nabla F(\mathbf{w}^*) = \mathbf{I} - \alpha(\mathbf{I} - \alpha\mathbf{J})\mathbf{J}(\mathbf{w}^*)$$

by assumption that $\boldsymbol{\xi}(\mathbf{w}^*) = 0$. Since all terms except $\mathbf{I}$ are of order at least α, $\nabla F(\mathbf{w}^*)$ is invertible for all α sufficiently small. By the inverse function theorem, there exists a neighbourhood U of $\mathbf{w}^*$ such that F is has a continuously differentiable inverse on U. Hence F restricted to U is a C^1 diffeomorphism with fixed point $\mathbf{w}^*$.

By definition of strict saddles, $\mathbf{J}(\mathbf{w}^*)$ has an eigenvalue with negative real part. It follows by continuity that $(\mathbf{I} - \alpha\mathbf{J})\mathbf{J}(\mathbf{w}^*)$ also has an eigenvalue $a + ib$ with $a < 0$ for α sufficiently small. Finally,

$$\nabla F(\mathbf{w}^*) = \mathbf{I} - \alpha(\mathbf{I} - \alpha\mathbf{J})\mathbf{J}(\mathbf{w}^*)$$

has an eigenvalue $\sigma = 1 - \alpha a - i\alpha b$ with

$$|\sigma| = 1 - 2\alpha a + \alpha^2(a^2 + b^2) \geq 1 - 2\alpha a > 1\,.$$

It follows that E^s has codimension at least one, implying in turn that the local stable set W has measure zero. We can now prove that

$$Z = \{\mathbf{w} \in U \mid \lim_{n\to\infty} F^n(\mathbf{w}) = \mathbf{w}^*\}$$

has measure zero, or in other words, that local convergence to $\mathbf{w}^*$ occurs with zero probability. Let B the neighbourhood guaranteed by the Stable Manifold Theorem, and take any $\mathbf{w} \in Z$. By definition of convergence there exists $N \in \mathbb{N}$ such that $F^{N+n}(\mathbf{w}) \in B$ for all $n \in \mathbb{N}$, so that

$$F^N(\mathbf{w}) \in \cap_{n\in\mathbb{N}}^{\infty} F^{-n}(B) \subset W$$

by the Stable Manifold Theorem. This implies that $\mathbf{w} \in F^{-N}(W)$, and by extension $\mathbf{w} \in \cup_{n\in\mathbb{N}} F^{-n}(W)$. Since $\mathbf{w}$ was arbitrary, we obtain the inclusion

$$Z \subseteq \cup_{n\in\mathbb{N}} F^{-n}(W)\,.$$

Now F^{-1} is C^1, hence locally Lipschitz and thus preserves sets of measure zero, so that $F^{-n}(W)$ has measure zero for each n. Countable unions of measure zero sets are still measure zero, so we conclude that Z also has measure zero. In other words, SGA converges to $\mathbf{w}^*$ with zero probability upon random initialization of $\mathbf{w}$ in U. ■

Unlike stable and unstable fixed points, it is unclear how to avoid strict saddles using only alignment, that is, independently from the size of λ.

4. Experiments

We compare SGA with simultaneous gradient descent, optimistic mirror descent (Daskalakis et al., 2018) and consensus optimization (Mescheder et al., 2017) in basic settings.

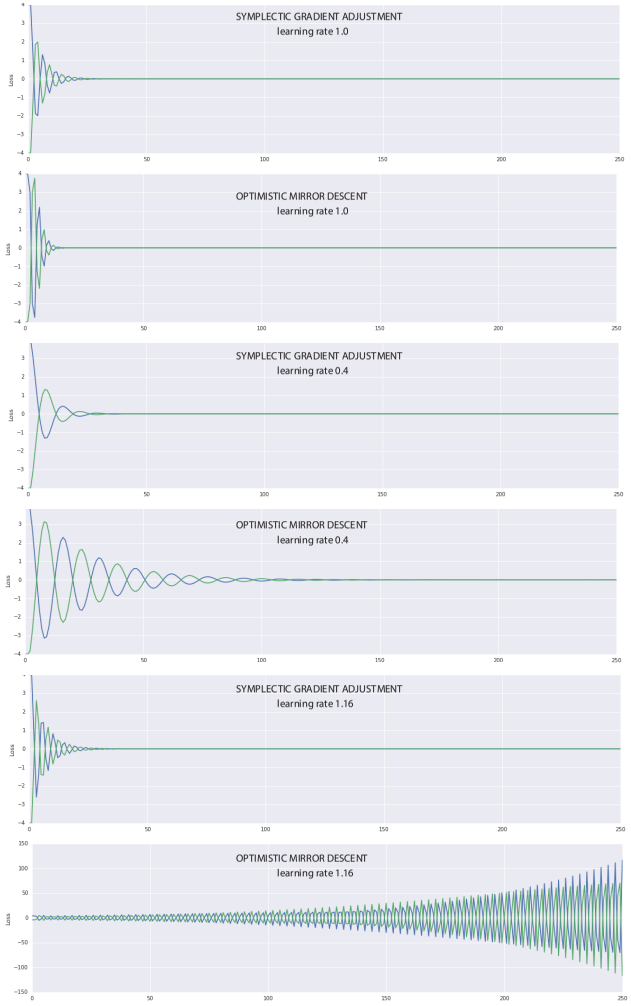


Figure 6: Individual runs on zero-sum bimatrix game in Section 4.2.

4.1. Learning rates and alignment

We investigate the effect of SGA when a weak attractor is coupled to a strong rotational force:

$$\ell_1(x,y) = \frac{1}{2}x^2 + 10xy \quad \text{and} \quad \ell_2(x,y) = \frac{1}{2}y^2 - 10xy$$

Gradient descent is extremely sensitive to the choice of learning rate η, top row of Figure 4. As η increases through $\{0.01, 0.032, 0.1\}$ gradient descent goes from converging extremely slowly, to diverging slowly, to diverging rapidly. SGA yields faster, more robust convergence. SGA converges faster with learning rates $\eta = 0.01$ and $\eta = 0.032$, and only starts overshooting the fixed point for $\eta = 0.1$.

4.2. Basic adversarial games

Optimistic mirror descent is a family of algorithms that has nice convergence properties in games (Rakhlin and Sridharan, 2013; Syrgkanis et al., 2015). In the special case of optimistic *gradient* descent the updates are

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta \cdot \boldsymbol{\xi}_t - \eta \cdot (\boldsymbol{\xi}_t - \boldsymbol{\xi}_{t-1}).$$

Figure 5 compares SGA with optimistic gradient descent (OMD) on a zero-sum bimatrix game with $\ell_{1/2}(\mathbf{w}_1, \mathbf{w}_2) = \pm \mathbf{w}_1^\intercal \mathbf{w}_2$. The example is modified from Daskalakis et al. (2018) who also consider a linear offset that makes no difference. A run is taken to have converged if the average absolute value of losses on the last 10 iterations is < 0.01; we end each experiment after 250 steps.

The left panel shows the number of steps to convergence (when convergence occurs) over a range of learning rates. OMD's peak performance is better than SGA, where the red curve dips below the blue. Howwever, we find that SGA converges—and does so faster—for a much wider range of learning rates. OMD diverges for learning rates not in the range [0.3, 1.2]. Simultaneous gradient descent oscillates without converging (not shown). The right panel shows the average performance of OMD and SGA on the last 10 steps. Once again, here SGA consistently performs better over a wider range of learning rates. Individual runs are shown in Figure 6.

4.2.1. OMD and SGA on a Four-Player Game

Figure 7 shows time to convergence (using the same convergence criterion as above) for optimistic mirror descent and SGA. The games are constructed with four players, each of which controls one parameter. The losses are

$$\ell_1(w,x,y,z) = \frac{\epsilon}{2}w^2 + wx + wy + wz$$
$$\ell_2(w,x,y,z) = -wx + \frac{\epsilon}{2}x^2 + xy + xz$$

$$\ell_3(w,x,y,z) = -wy - xy + \frac{\epsilon}{2}y^2 + yz$$
$$\ell_4(w,x,y,z) = -wz - xz - yz + \frac{\epsilon}{2}z^2,$$

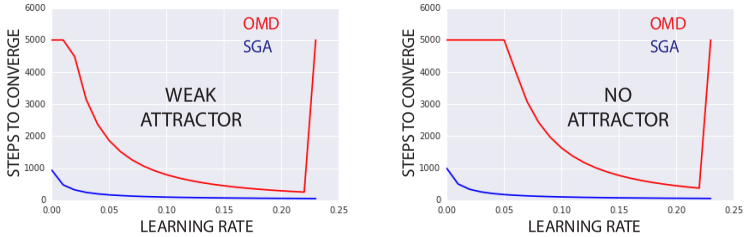


Figure 7: Time to convergence of OMD and SGA on two 4-player games. Times are cutoff after 5000 iterations. **Left panel:** Weakly positive definite $\mathbf{S}$ with $\epsilon = \frac{1}{100}$. **Right panel:** Symmetric component is identically zero.

where $\epsilon = \frac{1}{100}$ in the left panel and $\epsilon = 0$ in the right panel. The antisymmetric component of the game Jacobian is

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ -1 & 0 & 1 & 1 \\ -1 & -1 & 0 & 1 \\ -1 & -1 & -1 & 0 \end{pmatrix}$$

and the symmetric component is

$$\mathbf{S} = \epsilon \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

OMD converges considerably slower than SGA across the full range of learning rates. It also diverges for learning rates > 0.22. In contrast, SGA converges more quickly and robustly.

4.3. Learning a two-dimensional mixture of Gaussians

We apply SGA to a basic Generative Adversarial Network setup adapted from Metz et al. (2017). Data is sampled from a highly multimodal distribution designed to probe the tendency of GANs to collapse onto a subset of modes during training. The distribution is a mixture of 16 Gaussians arranged in a 4×4 grid. Figure 8 shows the probability distribution that is sampled to train the generator and discriminator. The generator and discriminator networks both have 6 ReLU layers of 384 neurons. The generator has two output neurons; the discriminator has one.

Figure 9 shows results after $\{2000, 4000, 6000, 8000\}$ iterations. The networks are trained under RMSProp. Learning rates were chosen by visual inspection of grid search results at iteration 8000. More precisely, grid search was over learning rates {1e-5, 2e-5,5e-5, 8e-5, 1e-4, 2e-4, 5e-4} and then a more refined linear search over [8e-5, 2e-4]. Simultaneous gradient descent and SGA are shown in the figure.

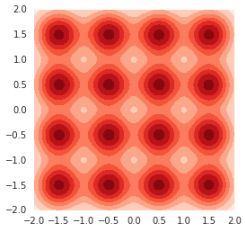


Figure 8: Ground truth for GAN experiments on a two-dimensional mixture of 16 Gaussians.

The last two rows of Figure 9 show the performance of consensus optimization without and with alignment. Introducing alignment slightly improves speed of convergence (second column) and final result (fourth column), although intermediate results in third column are ambiguous.

Simultaneous gradient descent exhibits mode collapse followed by mode hopping in later iterations (not shown). Mode hopping is analogous to the cycles in example 1. Unaligned SGA converges to the correct distribution; alignment speeds up convergence slightly. Consensus optimization performs similarly in this GAN example. However, consensus optimization can converge to local maxima even in potential games, recall example 8.

4.4. Learning a high-dimensional unimodal Gaussian

Mode collapse is a well-known phenomenon in GANs. A more subtle phenomenon, termed boundary distortion, was identified in Santurkar et al. (2018). Boundary distortion is a form of covariate shift where the generator fails to model the true data distribution.

Santurkar *et al* demonstrate boundary distortion using data sampled from a 75-dimensional unimodal Gaussian with spherical covariate matrix. Mode collapse is not a problem in this setting because the data distribution is unimodal. Nevertheless, they show that vanilla GANs fail to learn most of the spectrum of the covariate matrix.

Figure 10 reproduces their result. Panel A shows the ground truth: all 75 eigenvalues are equal to 1.0. Panel B shows the spectrum of the covariance matrix of the data generated by a GAN trained with RMSProp. The GAN concentrates on a single eigenvalue and essentially ignores the remaining 74 eigenvalues. This is similar to, but more extreme than, the empirical results obtained in Santurkar et al. (2018). We emphasize that the problem is not mode collapse, since the data is unimodal (although, it's worth noting that most of the mass of a high-dimensional Gaussian lies on the "shell").

Finally, panel C shows the spectrum of the covariance matrix of the data sampled from a GAN trained via SGA. The GAN approximately learns all the eigenvalues, with values ranging between 0.6 and 1.5.

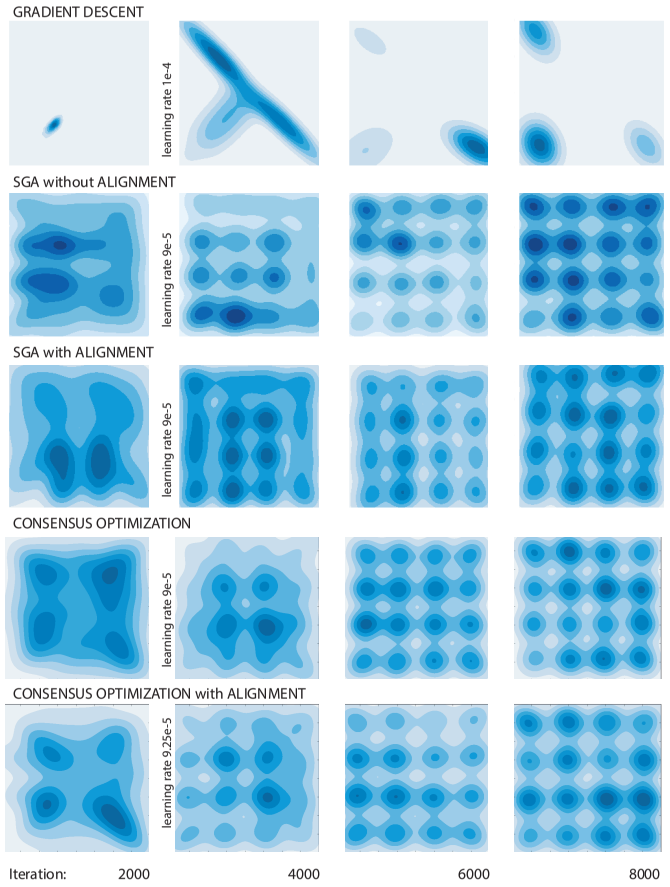


Figure 9: **First row:** Simultaneous gradient descent suffers from mode collapse and in later iterations (not shown) mode hopping. **Second and third rows:** vanilla SGA converges smoothly to the ground truth (Figure 8). SGA with alignment converges slightly faster. **Fourth and fifth rows:** Consensus optimization without and with alignment.

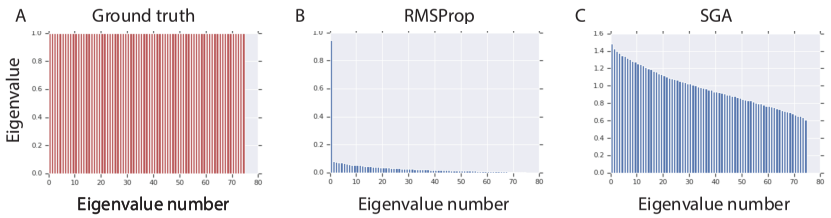


Figure 10: **Panel A:** The ground truth is a 75 dimensional spherical Gaussian whose covariance matrix has all eigenvalues equal to 1.0. **Panel B:** A vanilla GAN trained with RMSProp approximately learns the first eigenvalue, but essentially ignores all the rest. **Panel C:** Applying SGA results in the GAN approximately learning all 75 eigenvalues, although the range varies from 0.6 to 1.5.

5. Discussion

Modern deep learning treats differentiable modules like plug-and-play lego blocks. For this to work, at the very least, we need to know that gradient descent will find local minima. Unfortunately, gradient descent does *not* necessarily find local minima when optimizing multiple interacting objectives. With the recent proliferation of algorithms that optimize more than one loss, it is becoming increasingly urgent to understand and control the dynamics of interacting losses. Although there is interesting recent work on two-player adversarial games such as GANs, there is essentially no work on finding stable fixed points in more general games played by interacting neural nets.

The generalized Helmholtz decomposition provides a powerful new perspective on game dynamics. A key feature is that the analysis is indifferent to the number of players. Instead, it is the interplay between the simultaneous gradient $\boldsymbol{\xi}$ on the losses and the symmetric and antisymmetric matrices of second-order terms that guides algorithm design and governs the dynamics under gradient adjustments.

Symplectic gradient adjustment is a straightforward application of the generalized Helmholtz decomposition. It is unlikely that SGA is the best approach to finding stable fixed points. A deeper understanding of the interaction between the potential and Hamiltonian components will lead to more effective algorithms. Reinforcement learning algorithms that optimize multiple objectives are increasingly common, and second-order terms are difficult to estimate in practice. Thus, first-order methods that do not use Jacobian-vector products are of particular interest.

5.0.1. Gamification

Finally, it is worth raising a philosophical point. In this paper we are concerned with finding stable fixed points (because, for example, they yield pleasing samples in GANs). We are not concerned with the losses of the players *per se*. The gradient adjustments may lead to a

player acting against its own self-interest by increasing its loss. We consider this acceptable insofar as it encourages convergence to a stable fixed point. The players are but a means to an end.

We have argued that stable fixed points are a more useful solution concept than local Nash equilibria for our purposes. However, neither is entirely satisfactory, and the question "What is the right solution concept for neural games?" remains open. In fact, it likely has many answers. The intrinsic curiosity module introduced by Pathak et al. (2017) plays two objectives against one another to drive agents to search for novel experiences. In this case, converging to a fixed point is precisely what is to be avoided.

It is remarkable—to give a few examples sampled from many—that curiosity, generating photorealistic images, and image-to-image translation (Zhu et al., 2017) can be formulated as games. What else can games do?

Acknowledgments

We thank Guillaume Desjardins and Csaba Szepesvari for useful comments.

Appendix A. TensorFlow Code to Compute SGA

Source code is available at `https://github.com/deepmind/symplectic-gradient-adjustment`. Since computing the symplectic adjustment is quite simple, we include an explicit description here for completeness.

The code requires a list of n losses, `Ls`, and a list of variables for the n players, `xs`. The function `fwd_gradients` which implements forward mode auto-differentiation is in the module `tf.contrib.kfac.utils`.

```
% compute Jacobian-vector product Jv
def jac_vec(ys, xs, vs):
    return fwd_gradients(ys, xs, grad_xs = vs, stop_gradients = xs)

% compute Jacobian^T-vector product J^T v
def jac_tran_vec(ys, xs, vs):
  dydxs = tf.gradients(ys, xs, grad_ys = vs, stop_gradients = xs)
  return [tf.zeros_like(x) if dydx is None
          else dydx for (x, dydx) in zip(xs, dydxs)]

% compute Symplectic Gradient Adjustment A^T ξ
def get_sym_adj(Ls, xs):
  % compute game dynamics ξ
  xi = [tf.gradients(ℓ, x)[0] for (ℓ, x) in zip(Ls, xs)]
  J_xi = jac_vec(xi, xs, xi)
  Jt_xi = jac_tran_vec(xi, xs, xi)
  % compute A^T ξ = 1/2 (J^T ξ − J ξ)
```

```
At_xi = [(jt-j)/2 for (j, jt) in zip(J_xi, Jt_xi)]
return At_xi
```

Appendix B. Helmholtz, Hamilton, Hodge, and Harmonic Games

This section explains the mathematical connections with the Helmholtz decomposition, symplectic geometry and the Hodge decomposition. The discussion is *not* necessary to understand the main text. It is also not self-contained. The details can be found in textbooks covering differential and symplectic geometry (Arnold, 1989; Guillemin and Sternberg, 1990; Bott and Tu, 1995).

B.1. The Helmholtz Decomposition

The classical Helmholtz decomposition states that any vector field $\boldsymbol{\xi}$ in 3-dimensions is the sum of curl-free (gradient) and divergence-free (infinitesimal rotation) components:

$$\boldsymbol{\xi} = \underbrace{\nabla\phi}_{\text{gradient component}} + \underbrace{\text{curl}(\mathbf{B})}_{\text{rotational component}} \qquad \Big[\ \text{curl}(\bullet) := \nabla \times (\bullet)\Big]$$

We explain the link between curl and the antisymmetric component of the game Jacobian. Recall that gradients of functions are actually differential 1-forms, not vector fields. Differential 1-forms and vector fields on a manifold are canonically isomorphic once a Riemannian metric has been chosen. In our case, we are implicitly using the Euclidean metric. The antisymmetric matrix $\mathbf{A}$ is the differential 2-form obtained by applying the exterior derivative d to the 1-form $\boldsymbol{\xi}$.

In 3-dimensions, the Hodge star operator is an isormorphism from differential 2-forms to vector fields, and the curl can be reformulated as $\text{curl}(\bullet) = *d(\bullet)$. In claiming $\mathbf{A}$ is analogous to curl, we are simply dropping the Hodge-star operator.

Finally, recall that the Lie algebra of infinitesimal rotations in d-dimensions is given by antisymmetric matrices. When $d = 3$, the Lie algebra can be represented as vectors (three numbers specify a 3×3 antisymmetric matrix) with the $\times$-product as Lie bracket. In general, the antisymmetric matrix $\mathbf{A}$ captures the infinitesimal tendency of $\boldsymbol{\xi}$ to rotate at each point in the parameter space.

B.2. Hamiltonian Mechanics

A symplectic form ω is a closed nondegenerate differential 2-form. Given a manifold with a symplectic form, a vector field $\boldsymbol{\xi}$ is **Hamiltonian vector field** if there exists a function $\mathcal{H} : M \to \mathbb{R}$ satisfying

$$\omega(\boldsymbol{\xi}, \bullet) = d\mathcal{H}(\bullet) = \langle \nabla\mathcal{H}, \bullet \rangle. \tag{4}$$

The function is then referred to as the Hamiltonian function of the vector field. In our case, the antisymmetric matrix $\mathbf{A}$ is a closed 2-form because $\mathbf{A} = d\boldsymbol{\xi}$ and $d \circ d = 0$. It may however be degenerate. It is therefore a presymplectic form (Bottacin, 2005).

Setting $\omega = \mathbf{A}$, Equation (4) can be rewritten in our notation as

$$\underbrace{\omega(\boldsymbol{\xi}, \bullet)}_{\mathbf{A}^\intercal\boldsymbol{\xi}} = \underbrace{d\mathcal{H}(\bullet)}_{\nabla\mathcal{H}},$$

justifying the terminology 'Hamiltonian'.

B.3. The Hodge Decomposition

The exterior derivative $d_k : \Omega^k(M) \to \Omega^{k+1}(M)$ is a linear operator that takes differential k-forms on a manifold M, $\Omega^k(M)$, to differential $k+1$-forms, $\Omega^{k+1}(M)$. In the case $k = 0$, the exterior derivative is the gradient, which takes 0-forms (that is, functions) to 1-forms. Given a Riemannian metric, the adjoint of the exterior derivative δ goes in the opposite direction. Hodge's Theorem states that k-forms on a compact manifold decompose into a direct sum over three types:

$$\Omega^k(M) = d\Omega^{k-1}(M) \oplus \text{Harmonic}^k(M) \oplus \delta\Omega^{k+1}(M).$$

Setting $k = 1$, we recover a decomposition that closely resembles the generalized Helmholtz decomposition:

$$\underbrace{\Omega^1(M)}_{\text{1-forms}} = \underbrace{d\Omega^0(M)}_{\text{gradients of functions}} \oplus \text{Harm}^k(M) \oplus \underbrace{\delta\Omega^2(M)}_{\text{antisymmetric component}}$$

The harmonic component is isomorphic to the de Rham cohomology of the manifold—which is zero when $k = 1$ and $M = \mathbb{R}^n$.

Unfortunately, the Hodge decomposition does not straightforwardly apply to the case when $M = \mathbb{R}^n$, since $\mathbb{R}^n$ is not compact. It is thus unclear how to relate the generalized Helmholtz decomposition to the Hodge decomposition.

B.4. Harmonic and Potential Games

Candogan et al. (2011) derive a Hodge decomposition for games that is closely related in spirit to our generalized Helmholtz decomposition—although the details are quite different. Candogan et al. (2011) work with classical games (probability distributions on finite strategy sets). Their losses are multilinear, which is easier than our setting, but they have constrained solution sets, which is harder in many ways. Their approach is based on combinatorial Hodge theory (Jiang et al., 2011) rather than differential and symplectic geometry. Finding a best-of-both-worlds approach that encompasses both settings is an open problem.

Appendix C. Type Consistency

The next two sections carefully work through the units in classical mechanics and two-player games respectively. The third section briefly describes a use-case for type consistency.

C.1. Units in Classical Mechanics

Consider the well-known Hamiltonian

$$\mathcal{H}(p, q) = \frac{1}{2}\left(\kappa \cdot q^2 + \frac{1}{\mu} \cdot p^2\right)$$

where q is position, $p = \mu \cdot \dot{q}$ is momentum, μ is mass, κ is surface tension and $\mathcal{H}$ measures energy. The units (denoted by τ) are

$$\tau(q) = m \qquad \tau(p) = \tfrac{kg \cdot m}{s}$$

$$\tau(\kappa) = \tfrac{kg}{s^2} \qquad \tau(\mu) = kg$$

where m is meters, kg is kilograms and s is seconds. Energy is measured in joules, and indeed it is easy to check that $\tau(\mathcal{H}) = \frac{kg \cdot m^2}{s^2}$.

Note that the units for differentation by x are $\tau(\frac{\partial}{\partial x}) = \frac{1}{\tau(x)}$. For example, differentiating by time has units $\frac{1}{s}$. Hamilton's equations state that $\dot{q} = \frac{\partial H}{\partial p} = \frac{1}{\mu} \cdot p$ and $\dot{p} = -\frac{\partial \mathcal{H}}{\partial q} = -\kappa \cdot q$ where

$$\tau(\dot{q}) = \tfrac{m}{s} \qquad \tau(\dot{p}) = \tfrac{kg \cdot m}{s^2}$$

$$\tau\left(\tfrac{\partial}{\partial q}\right) = \tfrac{1}{m} \qquad \tau\left(\tfrac{\partial}{\partial p}\right) = \tfrac{s}{kg \cdot m}$$

The resulting flow describing the dynamics of the system is

$$\boldsymbol{\xi} = \dot{q} \cdot \frac{\partial}{\partial q} + \dot{p} \cdot \frac{\partial}{\partial p} = \frac{1}{\mu} p \cdot \frac{\partial}{\partial q} - \kappa q \cdot \frac{\partial}{\partial p}$$

with units $\tau(\boldsymbol{\xi}) = \frac{1}{s}$. Hamilton's equations can be reformulated more abstractly via symplectic geometry. Introduce the symplectic form

$$\omega = dq \wedge dp \quad \text{with units} \quad \tau(\omega) = \frac{kg \cdot m^2}{s}.$$

Observe that contracting the flow with the Hamiltonian obtains

$$\iota_{\boldsymbol{\xi}} \omega = \omega(\boldsymbol{\xi}, \bullet) = dH = \frac{\partial \mathcal{H}}{\partial q} \cdot dq + \frac{\partial \mathcal{H}}{\partial p} \cdot dp$$

with units $\tau(d\mathcal{H}) = \tau(\mathcal{H}) = \frac{kg \cdot m^2}{s^2}$.

C.1.1. Losses in Classical Mechanics

Although there is no notion of "loss" in classical mechanics, it is useful (for the next section) to keep pushing the formal analogy. Define the "losses"

$$\ell_1(q,p) = \frac{1}{\mu} \cdot qp \quad \text{and} \quad \ell_2(q,p) = -\kappa \cdot qp \tag{5}$$

with units $\tau(\ell_1) = \frac{m^2}{s}$ and $\tau(\ell_2) = \frac{kg^2 \cdot m^2}{s^3}$. The Hamiltonian dynamics can then be recovered game-theoretically by differentiating ℓ_1 and ℓ_2 with respect to q and p respectively. It is easy to check that

$$\boldsymbol{\xi} = \frac{\partial \mathcal{H}}{\partial p}\frac{\partial}{\partial q} - \frac{\partial \mathcal{H}}{\partial q}\frac{\partial}{\partial p} = \frac{\partial \ell_1}{\partial q}\frac{\partial}{\partial q} + \frac{\partial \ell_2}{\partial p}\frac{\partial}{\partial p}.$$

C.1.2. The Duality between Vector Fields and Differential Forms

Finally recall that the symplectic form in games was not "pulled out of thin air" as $\omega = dq \wedge dp$, but rather derived as $\omega = d\boldsymbol{\xi}^\flat$, where $\boldsymbol{\xi}^\flat$ is the differential form corresponding to the vector field $\boldsymbol{\xi}$ under the musical isomorphism $\flat : TM \to T^*M$.

It is instructive to compute $\boldsymbol{\xi}^\flat$ in the case of a classical mechanical system and see what happens. Naively, we would guess that the musical isomorphism is $\left(\frac{\partial}{\partial q}\right)^\flat = dq$ and $\left(\frac{\partial}{\partial p}\right)^\flat = dp$. However, applying the naive musical isomorphism to $\boldsymbol{\xi}$ to get

$$\boldsymbol{\xi}^\flat = \frac{\partial \ell_1}{\partial q} \cdot dq + \frac{\partial \ell_2}{\partial p} \cdot dp$$

results in a *type violation* because

$$\tau\left(\frac{\partial \ell_1}{\partial q} \cdot dq\right) = \tau(\ell_1) = \frac{m^2}{s}$$

whereas

$$\tau\left(\frac{\partial \ell_2}{\partial p} \cdot dp\right) = \tau(\ell_2) = \frac{kg^2 \cdot m^2}{s^3}$$

and we cannot add objects with different types.

To correct the type inconsistency, define the musical isomorphism as

$$\left(\frac{\partial}{\partial q}\right)^\flat = \frac{\mu}{2} \cdot dq \quad \text{and} \quad \left(\frac{\partial}{\partial p}\right)^\flat = \frac{1}{2\kappa} \cdot dp$$

with inverse

$$(dq)^\sharp = \frac{2}{\mu} \cdot \frac{\partial}{\partial q} \quad \text{and} \quad (dp)^\sharp = 2\kappa \cdot \frac{\partial}{\partial p}.$$

The correction terms in the direction $\flat : TM \to T^*M$ invert the coupling terms κ and $\frac{1}{\mu}$ that were originally introduced into the Hamiltonian for physical reasons. Applying the corrected musical isomorphism to $\boldsymbol{\xi}$ yields

$$\boldsymbol{\xi}^\flat = \frac{\mu}{2} \cdot \frac{\partial f}{\partial q} \cdot dq + \frac{1}{2\kappa} \cdot \frac{\partial g}{\partial p} \cdot dp = \frac{1}{2}\left(p \cdot dq - q \cdot dp\right).$$

The two terms of $\boldsymbol{\xi}^\flat$ then have coherent types

$$\tau\left(\tfrac{\partial \ell_1}{\partial q} \cdot \mu \cdot dq\right) = \tfrac{m}{s} \cdot kg \cdot m = \tfrac{kg \cdot m^2}{s}$$
$$\tau\left(\tfrac{\partial \ell_2}{\partial p} \cdot \tfrac{1}{\kappa} \cdot dp\right) = \tfrac{kg \cdot m}{s^2} \cdot \tfrac{s^2}{kg} \cdot \tfrac{kg \cdot m}{s} = \tfrac{kg \cdot m^2}{s}$$

as required. The associated two form is

$$\omega := d\boldsymbol{\xi}^\flat = -\left(\mu \cdot \frac{\partial^2 f}{\partial q \partial p} - \frac{1}{\kappa} \cdot \frac{\partial^2 g}{\partial q \partial p}\right) dq \wedge dp = -dq \wedge dp$$

which recovers the symplectic form (up to sign) with units $\tau(\omega) = \frac{kg \cdot m^2}{s}$ as required. Finally, observe that

$$\begin{aligned}\langle \boldsymbol{\xi}, \boldsymbol{\xi}^\flat \rangle &= \frac{1}{2}\left\langle \frac{p}{\mu}\cdot\frac{\partial}{\partial q} - \kappa q \cdot \frac{\partial}{\partial p}, p\cdot dq - q \cdot dp \right\rangle \\ &= \frac{1}{2}\left(\kappa \cdot q^2 + \frac{1}{\mu}\cdot p^2\right) = \mathcal{H}(p,q)\end{aligned}$$

recovering the Hamiltonian.

C.2. Units in Two-Player Games

Without loss of generality let $\mathbf{w} = (\mathbf{x};\mathbf{y})$ where we refer to $\mathbf{x}$ as position and $\mathbf{y}$ as momentum so that $\tau(\mathbf{x}) = m$ and $\tau(\mathbf{y}) = \frac{kg\cdot m}{s}$. The aim of this section is to check type-consistency under these, rather arbitrarily assigned, units. Since we are considering a game, we do not require that $\mathbf{x}$ and $\mathbf{y}$ have the same dimension—even though this would necessarily be the case for a physical system. The goal is to verify that units can be consistently assigned to games.

Consider a quadratic two player game of the form

$$\ell_1(\mathbf{w}) = \frac{1}{2}\begin{pmatrix}\mathbf{x}^\intercal & \mathbf{y}^\intercal\end{pmatrix}\begin{pmatrix}\mathbf{A}_{11} & \mathbf{A}_{12}\\ \mathbf{A}_{21} & \mathbf{A}_{22}\end{pmatrix}\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix} + \begin{pmatrix}\mathbf{x}^\intercal & \mathbf{y}^\intercal\end{pmatrix}\begin{pmatrix}\mathbf{b}_1\\ \mathbf{b}_2\end{pmatrix}$$

and

$$\ell_2(\mathbf{w}) = \frac{1}{2}\begin{pmatrix}\mathbf{x}^\intercal & \mathbf{y}^\intercal\end{pmatrix}\begin{pmatrix}\mathbf{C}_{11} & \mathbf{C}_{12}\\ \mathbf{C}_{21} & \mathbf{C}_{22}\end{pmatrix}\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix} + \begin{pmatrix}\mathbf{x}^\intercal & \mathbf{y}^\intercal\end{pmatrix}\begin{pmatrix}\mathbf{d}_1\\ \mathbf{d}_2\end{pmatrix}$$

We restrict to quadratic games since our methods only involve first and second derivatives. We assume the matrices $\mathbf{A}$ and $\mathbf{C}$ are symmetric without loss of generality so that, for example, $\mathbf{A}_{12} = \mathbf{A}_{21}^\intercal$. Adding constant terms to ℓ_1 and ℓ_2 makes no difference to the analysis so they are omitted.

By Equation (5), the units for ℓ_1 and ℓ_2 should be $\frac{m^2}{s}$ and $\frac{kg\cdot m^2}{s^3}$ respectively. We can therefore derive the correct units for each of the components of the quadratic losses as

$$\underbrace{\begin{pmatrix}m & \frac{kg\cdot m}{s}\end{pmatrix}\begin{pmatrix}\frac{1}{s} & \frac{1}{kg}\\ \frac{1}{kg} & \frac{s}{kg^2}\end{pmatrix}\begin{pmatrix}m\\ \frac{kg\cdot m}{s}\end{pmatrix}}_{\mathbf{w}^\intercal\mathbf{A}\mathbf{w}} + \underbrace{\begin{pmatrix}m & \frac{kg\cdot m}{s}\end{pmatrix}\begin{pmatrix}\frac{m}{s}\\ \frac{m}{kg}\end{pmatrix}}_{\mathbf{w}^\intercal\mathbf{b}}$$

for ℓ_1 and

$$\underbrace{\begin{pmatrix}m & \frac{kg\cdot m}{s}\end{pmatrix}\begin{pmatrix}\frac{kg^2}{s^3} & \frac{kg}{s^2}\\ \frac{kg}{s^2} & \frac{1}{s}\end{pmatrix}\begin{pmatrix}m\\ \frac{kg\cdot m}{s}\end{pmatrix}}_{\mathbf{w}^\intercal\mathbf{C}\mathbf{w}} + \underbrace{\begin{pmatrix}m & \frac{kg\cdot m}{s}\end{pmatrix}\begin{pmatrix}\frac{kg^2\cdot m}{s^3}\\ \frac{kg\cdot m}{s^2}\end{pmatrix}}_{\mathbf{w}^\intercal\mathbf{d}}$$

for ℓ_2. It follows from a straightforward computation that the vector field $\boldsymbol{\xi} = \frac{\partial \ell_1}{\partial \mathbf{x}}\frac{\partial}{\partial \mathbf{x}} + \frac{\partial \ell_2}{\partial \mathbf{y}}\frac{\partial}{\partial \mathbf{y}}$ has type $\tau(\boldsymbol{\xi}) = \frac{1}{s}$ as required.

The presymplectic form $\omega = d\boldsymbol{\xi}^\flat$ makes use of the musical isomorphism $\flat : T^M \to T^*M$. As in Section C.1, if we naively define $(\frac{\partial}{\partial \mathbf{x}})^\flat = d\mathbf{x}$ and $(\frac{\partial}{\partial \mathbf{y}})^\flat = d\mathbf{y}$ then

$$\boldsymbol{\xi}^\flat = \frac{\partial \ell_1}{\partial \mathbf{x}}\cdot d\mathbf{x} + \frac{\partial \ell_2}{\partial \mathbf{y}}\cdot d\mathbf{y}$$

which is type inconsistent because $\tau(\frac{\partial \ell_1}{\partial \mathbf{x}}\cdot d\mathbf{x}) = \frac{m^2}{s}$ and $\tau(\frac{\partial \ell_2}{\partial \mathbf{y}}\cdot d\mathbf{y}) = \frac{kg^2\cdot m^2}{s^3}$.

C.2.1. Type-Consistency via SVD

It is necessary, as in Section C.1, to correct the naive musical isomorphism by taking into account the coupling constants for the mixed position-momentum terms. In the classical setup the coupling constants were the scalars $\frac{1}{\mu}$ and κ, whereas in a game they are the off-diagonal blocks $\mathbf{A}_{12}$ and $\mathbf{C}_{12}$.

Apply singular value decomposition to factorize

$$\mathbf{A}_{12} = \mathbf{U}_\mathbf{A}^\intercal \mathbf{D}_\mathbf{A}\mathbf{V}_\mathbf{A} \quad \text{and} \quad \mathbf{C}_{12} = \mathbf{U}_\mathbf{C}^\intercal \mathbf{D}_\mathbf{C}\mathbf{V}_\mathbf{C}$$

where the entries of the diagonal matrices have types $\tau(\mathbf{D}_\mathbf{A}) = \frac{1}{kg}$ and $\tau(\mathbf{D}_\mathbf{C}) = \frac{kg}{s^2}$, and *the types of the orthogonal matrices* $\mathbf{U}$ *and* $\mathbf{V}$ *are pure scalars.* The diagonal matrices $\mathbf{D}_\mathbf{A}$ and $\mathbf{D}_\mathbf{C}$ have the same types as $\frac{1}{\mu}$ and κ in the classical system since they play the same coupling role.

Extending the procedure adopted in the Section C.1, fix the type-inconsistency by defining the musical isomorphisms as

$$\left(\frac{\partial}{\partial \mathbf{x}}\right)^\flat = \mathbf{U}_\mathbf{A}^\intercal \mathbf{D}_\mathbf{A}^{-1}\mathbf{U}_\mathbf{A}\cdot d\mathbf{x}$$

and

$$\left(\frac{\partial}{\partial \mathbf{y}}\right)^\flat = \mathbf{V}_\mathbf{C}^\intercal \mathbf{D}_\mathbf{C}^{-1}\mathbf{V}_\mathbf{C}\cdot d\mathbf{y}.$$

Alternatively, the isomorphisms can be computed by noting that $\mathbf{U}_\mathbf{A}^\intercal \mathbf{D}_\mathbf{A}^{-1}\mathbf{U}_\mathbf{A} = (\sqrt{\mathbf{A}_{12}\mathbf{A}_{21}})^{-1}$ and $\mathbf{V}_\mathbf{C}^\intercal \mathbf{D}_\mathbf{C}^{-1}\mathbf{V}_\mathbf{C} = (\sqrt{\mathbf{C}_{21}\mathbf{C}_{12}})^{-1}$.

The dual isomorphism $\sharp : T^*M \to TM$ is then

$$(d\mathbf{x})^\sharp = \mathbf{U}_\mathbf{A}^\intercal \mathbf{D}_\mathbf{A}\mathbf{U}_\mathbf{A}\cdot\frac{\partial}{\partial \mathbf{x}} \quad \text{and} \quad (d\mathbf{y})^\sharp = \mathbf{V}_\mathbf{C}^\intercal \mathbf{D}_\mathbf{C}\mathbf{V}_\mathbf{C}\cdot\frac{\partial}{\partial \mathbf{y}}$$

If

$$\boldsymbol{\xi} = \begin{pmatrix}\mathbf{A}_{12}\mathbf{y} + \mathbf{b}_1\\ \mathbf{C}_{21}\mathbf{x} + \mathbf{d}_2\end{pmatrix}$$

then it follows that

$$\boldsymbol{\xi}^\flat = \begin{pmatrix}\mathbf{U}_\mathbf{A}^\intercal \mathbf{D}_\mathbf{A}^{-1}\mathbf{U}_\mathbf{A}\mathbf{b}_1 + \mathbf{U}_\mathbf{A}^\intercal \mathbf{U}_\mathbf{A}\mathbf{y}\\ \mathbf{V}_\mathbf{C}^\intercal \mathbf{D}_\mathbf{C}^{-1}\mathbf{V}_\mathbf{C}\mathbf{d}_2 + \mathbf{V}_\mathbf{C}^\intercal \mathbf{V}_\mathbf{C}\mathbf{x}\end{pmatrix}$$

with associated closed two form

$$\omega_\tau = d\boldsymbol{\xi}^\flat = -\left(\mathbf{U}_\mathbf{A}^\intercal \mathbf{V}_\mathbf{A} - \mathbf{U}_\mathbf{C}^\intercal \mathbf{V}_\mathbf{C}\right) d\mathbf{x}\wedge d\mathbf{y}.$$

where the notation ω_τ emphasizes that the two-form is type-consistent.

C.3. What Does Type-Consistency Buy?

Example 10 *Consider the loss functions*

$$f(x, y) = xy \quad \textit{and} \quad g(x, y) = 2xy,$$

with $\boldsymbol{\xi} = (y, 2x)$*. There is no function* $\phi : \mathbb{R}^2 \to \mathbb{R}$ *such that* $\nabla\phi = \boldsymbol{\xi}$*. However, there is a family of functions* $\phi_\alpha(x, y) = \alpha \cdot xy$ *which satisfies*

$$\langle \boldsymbol{\xi}, \nabla\phi_\alpha \rangle = \alpha \cdot (x^2 + 2y^2) \geq 0 \quad \textit{for all } \alpha > 0.$$

Although $\boldsymbol{\xi}$ is not a potential field, there is a family of functions on which $\boldsymbol{\xi}$ performs gradient descent—albeit with coordinate-wise learning rates that may not be optimal. The vector field $\boldsymbol{\xi}$ arguably does not require adjustment. This kind of situation often arises when the learning rates of different parameters are set adaptively during training of neural nets, by rescaling them by positive numbers.

The vanilla and type-consistent 1-forms corresponding to $\boldsymbol{\xi}$ are, respectively,

$$\boldsymbol{\xi}^\flat = y \cdot dx + 2x \cdot dy \quad \text{and} \quad \boldsymbol{\xi}^\flat_\tau = y \cdot dx + x \cdot dy$$

with

$$\omega = d\boldsymbol{\xi}^\flat_{\text{non}} = dx \wedge dy \quad \text{and} \quad \omega_\tau = d\boldsymbol{\xi}^\flat_\tau = 0.$$

It follows that the *type-consistent* symplectic gradient adjustment is zero. Type-consistency 'detects' that no gradient adjustment is needed in example 10.

References

S Abdallah and V R Lesser. A multiagent reinforcement learning algorithm with non-linear dynamics. *J. Artif. Intell. Res.*, 33:521–549, 2008.

V Arnold. *Mathematical Methods of Classical Mechanics*. Springer, 1989.

J Bailey and G Piliouras. Multiagent learning in network zero-sum games is a Hamiltonian system. In *AAMAS*, 2019.

D Balduzzi. Strongly-typed agents are guaranteed to interact safely. In *ICML*, 2017.

D Balduzzi, S Racanière, J Martens, J Foerster, K Tuyls, and T Graepel. The mechanics of n-player differentiable games. In *ICML*, 2018a.

D Balduzzi, K Tuyls, J Perolat, and T Graepel. Re-evaluating evaluation. In *NeurIPS*, 2018b.

R Bott and L Tu. *Differential Forms in Algebraic Topology*. Springer, 1995.

F Bottacin. A Marsden-Weinstein reduction theorem for presymplectic manifolds. 2005.

M Bowling and M Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136:215–250, 2002.

M Bowling. Convergence and no-regret in multiagent learning. In *NeurIPS*, pages 209–216, 2004.

Y Burda, H Edwards, D Pathak, A Storkey, T Darrell, and A Efros. Large-scale study of curiosity-driven learning. In *ICLR*, 2019.

O Candogan, I Menache, A Ozdaglar, and P A Parrilo. Flows and decompositions of games: harmonic and potential games. *Mathematics of Operations Research*, 36(3):474–503, 2011.

C Daskalakis, P W Goldberg, and C Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM J. Computing*, 39(1):195–259, 2009.

C Daskalakis, A Ilyas, V Syrgkanis, and H Zeng. Training GANs with optimism. In *ICLR*, 2018.

F Facchinei and C Kanzow. Generalized Nash equilibrium problems. *Annals of Operations Research*, 175(1):177–211, 2010.

S Feizi, C Suh, F Xia, and D Tse. Understanding GANs: the LQG setting. In *arXiv:1710.10793*, 2017.

J N Foerster, R Y Chen, M Al-Shedivat, S Whiteson, P Abbeel, and I Mordatch. Learning with opponent-learning awareness. In *AAMAS*, 2018.

I Gemp and S Mahadevan. Online monotone optimization. In *arXiv:1608.07888*, 2016.

I Gemp and S Mahadevan. Online monotone games. In *arXiv:1710.07328*, 2017.

I Gemp and S Mahadevan. Global convergence to the equilibrium of GANs using variational inequalities. In *arXiv:1808.01531*, 2018.

G Gidel, R A Hemmat, M Pezeshki, R Lepriol, G Huang, S Lacoste-Julien, and I Mitliagkas. Negative momentum for improved game dynamics. In *arXiv:1807.04740*, 2018.

G Gidel, H Berard, G Vignoud, P Vincent, and Lacoste-Julien. A variational inequality perspective on generative adversarial networks. In *ICLR*, 2019.

I J Goodfellow, J Pouget-Abadie, M Mirza, B Xu, D Warde-Farley, S Ozair, A Courville, and Y Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

V Guillemin and S Sternberg. *Symplectic Techniques in Physics*. Cambridge University Press, 1990.

S Hart and A Mas-Colell. *Simple Adaptive Strategies: From Regret-Matching to Uncoupled Dynamics*. World Scientific, 2013.

M Heusel, H Ramsauer, T Unterthiner, B Nessler, G Klambauer, and S Hochreiter. GANs trained by a two time-scale update rule converge to a Nash equilibrium. In *NeurIPS*, 2017.

M Jaderberg, W M Czarnecki, S Osindero, O Vinyals, A Graves, and K Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *ICML*, 2017.

X Jiang, L Lim, Y Yao, and Y Ye. Statistical ranking and combinatorial Hodge theory. *Math. Program., Ser. B*, 127:203–244, 2011.

J D Lee, M Simchowitz, M I Jordan, and B Recht. Gradient descent converges to minimizers. In *COLT*, 2016.

JD Lee, I Panageas, G Piliouras, M Simchowitz, MI Jordan, and B Recht. First-order methods almost always avoid saddle points. In *arXiv:1710.07406*, 2017.

A Letcher, J Foerster, D Balduzzi, T Rocktäschel, and S Whiteson. Stable opponent shaping in differentiable games. In *ICLR*, 2019.

B Liu, J Liu, M Ghavamzadeh, S Mahadevan, and M Petrik. Proximal gradient temporal difference learning algorithms. In *IJCAI*, 2016.

X Lu. Hamiltonian games. *Journal of Combinatorial Theory, Series B*, 55:18–32, 1992.

P Mertikopoulos and Z Zhou. Learning in games with continuous action sets and unknown payoff functions. In *arXiv:1608.07310*, 2016.

P Mertikopoulos, C Papadimitriou, and G Piliouras. Cycles in adversarial regularized learning. In *SODA*, 2018.

P Mertikopoulos, H Zenati, B Lecouat, C Foo, V Chandrasekhar, and G Piliouras. Mirror descent in saddle-point problems: Going the extra (gradient) mile. In *ICLR*, 2019.

L Mescheder, S Nowozin, and A Geiger. The numerics of GANs. In *NeurIPS*. 2017.

L Mescheder, A Geiger, and S Nowozin. Which training methods for GANs do actually converge? In *ICML*, 2018.

L Metz, B Poole, D Pfau, and J Sohl-Dickstein. Unrolled generative adversarial networks. In *ICLR*, 2017.

D Monderer and L S Shapley. Potential games. *Games and Economic Behavior*, 14:124–143, 1996.

V Nagarajan and J Z Kolter. Gradient descent GAN optimization is locally stable. In *NeurIPS*, 2017.

J Nash. Equilibrium points in n-person games. *PNAS*, 36(1):48–49, 1950.

Y Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer, 2004.

J. Ortega and W. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Society for Industrial and Applied Mathematics, 2000.

C Papadimitriou and G Piliouras. From Nash equilibria to chain recurrent sets: solution concepts and topology. In *ITCS*, 2016.

C Papadimitriou and G Piliouras. From Nash equilibria to chain recurrent sets: an algorithmic solution concept for game theory. *Entropy*, 20, 2018.

D Pathak, P Agrawal, A A Efros, and T Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017.

David Pfau and Oriol Vinyals. Connecting generative adversarial networks and actor-critic methods. In *arXiv:1610.01945*, 2016.

S Racanière, T Weber, D P Reichert, L Buesing, A Guez, D J Rezende, A P Badia, O Vinyals, N Heess, Y Li, R Pascanu, P Battaglia, D Hassabis, D Silver, and D Wierstra. Imagination-augmented agents for deep reinforcement learning. In *NeurIPS*, 2017.

S Rakhlin and K Sridharan. Optimization, learning, and games with predictable sequences. In *NeurIPS*, 2013.

J B Rosen. Existence and uniqueness of equilibrium points for concave N-person games. *Econometrica*, 33(3):520–534, 1965.

R W Rosenthal. A class of games possessing pure-strategy Nash equilibria. *Int J Game Theory*, 2:65–67, 1973.

T Salimans, I Goodfellow, W Zaremba, V Cheung, A Radford, and X Chen. Improved techniques for training GANs. In *NeurIPS*, 2016.

S Santurkar, L Schmidt, and A Madry. A classification-based study of covariate shift in GAN distributions. In *ICML*, 2018.

G Scutari, D P Palomar, F Facchinei, and J Pang. Convex optimization, game theory, and variational inequality theory. *IEEE Signal Processing Magazine*, pages 35–49, 2010.

Y Shoham and K Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2008.

M Shub. *Global Stability of Dynamical Systems*. Springer, 2000.

S Singh, M Kearns, and Y Mansour. Nash convergence of gradient dynamics in general-sum games. In *UAI*, 2000.

G Stoltz and G Lugosi. Learning correlated equilibria in games with compact sets of strategies. *Games and Economic Behavior*, 59:187–208, 2007.

V Syrgkanis, A Agarwal, H Luo, and R E Schapire. Fast convergence of regularized learning in games. In *NeurIPS*, 2015.

A Vezhnevets, S Osindero, T Schaul, N Heess, M Jaderberg, D Silver, and K Kavukcuoglu. FeUdal networks for hierarchical reinforcement learning. In *ICML*, 2017.

G Wayne and L F Abbott. Hierarchical control using networks trained with higher-level forward models. *Neural Computation*, (26), 2014.

J Zhu, T Park, P Isola, and A A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *CVPR*, 2017.

M Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, pages 928–936, 2003.

Journal of Machine Learning Research 20 (2019) 1-35 Submitted 12/16; Revised 9/18; Published 5/19

Bayesian Space-Time Partitioning by Sampling and Pruning Spanning Trees

Leonardo V. Teixeira LTEIXEIR@PURDUE.EDU
Department of Computer Science
Purdue University
West Lafayette, IN 47907, USA

Renato M. Assunção ASSUNCAO@DCC.UFMG.BR
Departamento de Cieência da Computação
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil

Rosangela H. Loschi LOSCHI@EST.UFMG.BR
Departamento de Estatística
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil

Editor: Jennifer Dy

Abstract

A typical problem in spatial data analysis is regionalization or spatially constrained clustering, which consists of aggregating small geographical areas into larger regions. A major challenge when partitioning a map is the huge number of possible partitions that compose the search space. This is compounded if we are partitioning spatio-temporal data rather than purely spatial data. We introduce a spatio-temporal product partition model that deals with the regionalization problem in a probabilistic way. Random spanning trees are used as a tool to tackle the problem of searching the space of possible partitions making feasible this exploration. Based on this framework, we propose an efficient Gibbs sampler algorithm to sample from the posterior distribution of the parameters, specially the random partition. The proposed Gibbs sampler scheme carries out a random walk on the space of the spanning trees and the partitions induced by deleting tree edges. In the purely spatial situation, we compare our proposed model with other state-of-art regionalization techniques to partition maps using simulated and real social and health data. To illustrate how the temporal component is handled by the algorithm and to show how the spatial clusters vary along the time we presented an application using human development index data. The analysis shows that our proposed model is better than state-of-art alternatives. Another appealing feature of the method is that the prior distribution for the partition is interpretable with a trivial coin flipping mechanism allowing its easy elicitation.

Keywords: Spatial Clustering, Product Partition Models, Random Spanning Trees, Bayesian Clustering

1. Introduction

Traditional cluster analysis aims at partitioning a set of n objects into k clusters such that the clusters are composed by objects with similar valued attributes while objects from different clusters tend to be dissimilar. When the objects are spatially located, inducing a neighborhood structure, the clustering can be constrained by their spatial contiguity. For instance, in mapping problems, it is common to aggregate neighboring small areas to partition the map into larger regions, named *spatial clusters*. The small areas within a given spatial cluster are relatively homogeneous with respect to attributes such as, for example, ecological characteristics (Sayre et al., 2014) or crime rates (Mennis and Harris, 2013). This spatial clustering problem is called *regionalization* or *spatially constrained clustering*.

Regionalization serves to a range of purposes, such as to facilitate the visualization and understanding of the geographical information, to reduce the noise introduced by outliers and inaccurate data, to make data analysis tractable, or to provide a better statistical handling of the data by reducing the effect of different populations (Wise et al., 1997). It has been used in widely different applied problems such as ecoregion delimitation (Sayre et al., 2014; McKenney et al., 2007), climate zoning (Zhang et al., 2016), environmental planning (Bernetti et al., 2011), image segmentation (Ribeiro et al., 2013), communication protocols in geo-sensor networks (Reis et al., 2007), map generalization (Ruas, 2008), zone design for health studies (Cockings and Martin, 2005; Ricketts, 1997), and enhanced sampling procedures (Martin, 1998).

There are two possible ways to carry out this aggregation. One way is through an artificial clustering, where the constructed regions are predetermined using official or normative designations (such as states, districts and counties). That is, the regionalization of small units of interest, such as counties, are arbitrarily specified as larger units such as states, defined for administrative or political reasons. This kind of aggregation is usually the expression of political will and may not take into account the information specific to the domain being studied. Another way is to perform the aggregation based on the analysis of data characteristics related to the phenomena under study. An outcome of a regionalization method can be found in the left hand side of Figure 1, which shows the three spatial clusters based on the values of lung cancer mortality rates in municipalities in the South of Brazil for the period 2008-2012.

Many different methods have been proposed to deal with the problem of spatial regionalization as a non-stochastic optimization problem. These previous works are presented in Section 1.1. Most of these regionalization techniques consider data as fixed, static values and prefix the number of clusters *a priori*. Frequently, these assumptions are inappropriate, as they do not allow for measurement error or uncertainty on the areas' measures nor the evaluation of the uncertainty of the obtained clusters. For example, consider the bladder cancer mortality rate of a small town in a given year. This value should not be considered as the most representative value for the true mortality rate since it can be severely impacted by small differences from one year to another, particularly if the area has a small population. The measured value can show a natural variability, expressed in this widely different bladder cancer mortality rate variation in two successive years in small population areas. Traditional regionalization techniques that emphasizes the similarity between the observations will be sensitive to this variability and may output an regionalization that discloses an undesirably high number of clusters. To illustrate this problem, the right hand side map in Figure 1 shows an outcome of the *Automatic Zoning Procedure* (AZP) regionalization technique proposed by Openshaw (1977) considering the bladder cancer mortality rate in municipalities in the South of Brazil for the period 2008-2012. This type of cancer is rare, generating small

©2019 Leonardo V. Teixeira, Renato M. Assunção, Rosangela H. Loschi.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/16-615.html.

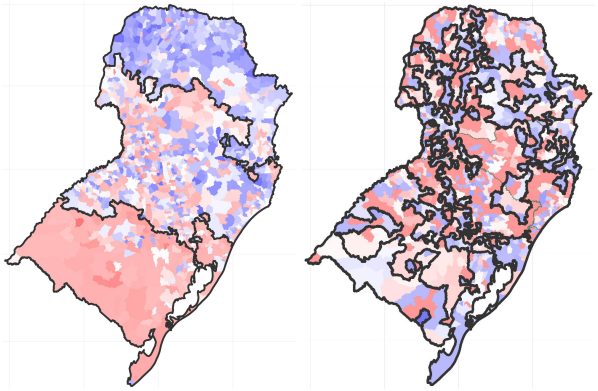


Figure 1: Example of a good (left) and a bad (right) regionalization using data from lung (left) and bladder (right) cancer mortality rates in South Brazilian municipalities for the period 2008-2012.

death counts in low population areas and thus extreme rates (high or low) in sharp contrast with their neighbors.

It is necessary to use an explicit stochastic framework to perform the regionalization if we want to take into account this natural variability in the data. An appropriate modeling approach should also allow us to quantify the uncertainty about the geographic partition. A stochastic framework accounting for such characteristics defines a random partition model. Consequently, our inference becomes more complex as there is a huge number of possible partitions that compose the parametric space.

Some attempts to achieve this goal in the spatial context have been already considered in the literature and they are reviewed in Section 1.1. In Teixeira et al. (2015), we introduce a random partition model that deals with purely spatial regionalization problems in a probabilistic way. In this article, in addition to presenting the model, we discuss in detail the elicitation of prior distributions for the partition and for the number of clusters. Such model is an alternative to the spatial product partition model (PPM) introduced by Hegarty and Barry (2008) and by Page and Quintana (2016). We also extend these models to space-time data by also accounting for spatio-temporal regionalization and provide extensive simulation results and new examples. We run a simulation study comparing the proposed model with well-known stochastic and non-stochastic regionalization techniques including Bayesian modeling alternatives. To illustrate the purely spatial case, we partition the Brazilian map based on the Human Development Index (HDI) and the Brazilian South region based on bladder and lung cancer mortality rates. The space-time case is illustrated with HDI in three decades.

The spatial and the spatio-temporal structure are both represented by a graph. The partition of graphs is not a new subject (Green and Thomas, 2013; Bornn and Caron, 2011).

Our contribution is the introduction of random spanning trees into the random partition model as a tool to handle the problem of searching in the space of possible partitions. We propose an efficient Gibbs sampler algorithm to sample from the posterior probability distribution, specially that for the partition. Conditionally on the spanning tree, the random partition has a product distribution thus defining a spatio-temporal/spatial product partition model. This strategy also greatly facilitates the scheme for sampling from the posterior distributions. By conditioning on the partitions resulting from spanning trees pruning, we substantially reduce the space of partitions or clusters to be explored. It is important to emphasize that the partition space is the set of all possible spatio-temporal/spatial partitions. The way we search this space is by selecting a new random spanning tree from the set of all possible spanning trees at each Gibbs step and then proceed with its partitioning. This guarantees that there is a positive probability that any partition will be reached in a finite time starting from any other partition in our MCMC sampling scheme.

In Section 2, we briefly review some basic concepts of graph theory we need in the construction of the proposed model. In Section 3 we introduce our proposed model and describe how we incorporate the spanning trees as a tool to drive through the partition space. In Section 4 we propose an efficient algorithm to sample from the posterior distribution. Section 5 presents a simulation study comparing the proposed model with some other well-known methods for regionalization in the spatial context. The analysis of some real data from Brazil is carried out in Section 6. Section 7 closes the paper with some final comments and the main conclusions.

1.1. Related Work

Openshaw (Openshaw, 1977) was a pioneer when he proposed *Automatic Zoning Procedure* (AZP), a heuristic method to aggregate areas that swapped regions locally improving an initial rough partitioning. Later, AZP variants were introduced using simulated annealing and tabu search (Openshaw and Rao, 1995). A modification of AZP named *Automatic Regionalization with Initial Seed Location* (ARiSeL) was presented by Duque and Church (2004). In ARiSeL, the construction of an initial feasible solution is repeated several times before running a tabu search which, according to the author, is less expensive than performing a local search. The *Self Organizing Maps* (SOM) algorithm, proposed by Kohonen (1990), is an unsupervised neural network which adjusts its weights to represent a data set distribution on a regular lattice. Although used to perform regionalization, the spatial contiguity desirable in a regionalization is not guaranteed. SOM variants have been proposed by Bação et al. (2004) and Bação et al. (2005) considering different procedures to explore the neighborhood structure. The heuristic devised by Aldstadt and Getis (2006), called AMOEBA (*A Multidirectional Optimum Ecotope-Based Algorithm*), starts with an initial area and grows it by adding neighboring areas until a local spatial autocorrelation statistic stops increasing. This process is repeated to all areas and a final step resolves overlaps. The *Max-p-regions* technique (Duque et al., 2012) does not require the previous setting of the number of spatial clusters and enforces the contiguity constraint. Clusters are formed in a such way that a regional attribute is always above certain threshold such as a minimum population or cluster area size.

The *Spatial 'K'luster Analysis by Tree Edge Removal* (SKATER) proposed by Assunção et al. (2006) is a graph based method that uses a minimal spanning tree to reduce the search space. The regions are then defined by the removal of edges from the minimal spanning tree. The removed edges are chosen to minimize a dissimilarity measure. Inspired by SKATER, Guo (2008) proposed REDCAP (*Regionalization with Dynamically Constrained Agglomerative clustering and Partitioning*) which considers six other methods to explore different connection strategies using the underlying graph structure. The model introduced in the present manuscript is also inspired by SKATER by using spanning trees as a tool to summarize the spatial connection and the variability of the data. It adopts a Bayesian approach putting prior distributions on all unknown aspects of the regionalization, including the partition itself. As a consequence, we are able to make inference on unknown parameters and, most importantly, to infer about the partition in a probabilistic way. Our method outputs a posterior distribution over all possible partitions and we can infer which ones are the most probable.

Knorr-Held and Raßer (2000) presented a Bayesian approach, named *Bayesian Detection of Clusters and Discontinuities* (BDCD), to aggregate small contiguous areas into larger regions to form spatial clusters. Focusing on the random partition, they assume a prior distribution $p(c)$ for the number c of spatial clusters. Conditioned on c, cluster centers are uniformly selected among the n available centroids. Each area is assigned to that cluster whose center is the nearest according to the number of boundaries that need to be crossed. In the Bayesian partition model (BPM) introduced by Denison and Holmes (2001), the prior distribution $p(c)$ is an uniform distribution and they use a Voronoi tessellation to determine the spatial clusters. One of the main differences between the two of them is the computational strategy to sample from the posterior distribution. While Knorr-Held and Raßer (2000) use an explicit reversible jump Markov chain Monte Carlo (MCMC) procedure (Green, 1995; Richardson and Green, 1997), Denison and Holmes (2001) proceed using a standard MCMC after integrating out some parameters.

Gangnon and Clayton (2000) proposed a different Bayesian approach for this problem by assuming a prior distribution for the number of clusters that depends on the geometry of the clusters. Such prior distribution gives more flexibility for spatial cluster analysis since, depending on our prior knowledge, it can be elicited penalizing some particular cluster aspects such as large cluster sizes or odd-shaped clusters. The algorithm proposed to approximate the posterior distribution has two components. The first is a window of plausibility, an adaptation of the Occam's window approach to model selection (Madigan and Raftery, 1994). In the second, given a window of plausibility, they use a randomized search algorithm similar to the backward elimination method used for variable selection in regression problems.

Lu and Carlin (2005) and Banerjee and Gelfand (2006) worked on a dual problem proposing a method known as boundary (or wobbling) analysis. Rather than aggregating similar areas into homogeneous regions, their method aims at identifying sharp boundaries between pairs of areas in such a way that homogeneous regions can be obtained as a byproduct. Wakefield and Kim (2013) proposed a Bayesian method for the detection of a small number of high risk zones, rather than providing a map partition. It requires the pre-specification of the maximum number of clusters, the clusters found tend to have a circular shape.

Anderson et al. (2014) proposed a two-step regionalization method. First, a hierarchical clustering method is used to define a set of possible partitions. In the second step, the

models defined by each partition are compared using a model selection tool and the best partition is thus selected.

The product partition model (PPM) introduced by Hartigan (1990) has been used for different purposes due to its flexibility in modeling heterogeneous data. However, only recently it has been considered to analyze spatial data. PPMs assume that the partition $\boldsymbol{\pi} = \{\mathcal{G}_1, \ldots, \mathcal{G}_c\}$ of a set of data is a random quantity and its main feature is to assume that the $\boldsymbol{\pi}$ distribution is a product of subjective non-negative functions $\kappa(\mathcal{G}_k)$ called prior cohesions, for $k = 1, \ldots, c$. The cohesion functions measure how likely elements in $\mathcal{G}_k$ are clustered *a priori*. The structure adopted to such cohesions defines the type of PPM we have in mind. Hegarty and Barry (2008) were the first to propose a spatial approach to PPM. They assumed that the prior cohesions of a component $\mathcal{G}_i$ of $\boldsymbol{\pi}$ is a function of the summation (over all areas in $\mathcal{G}_i$) of the number of neighboring areas not in $\mathcal{G}_i$. This cohesion may encourage maps with few contiguous clusters and discourage maps with large number of disconnected ones, which is desirable in regionalization problems. In the spatial PPM introduced by Page and Quintana (2016) the spatial dependence among the neighboring areas is incorporated into the model through both the likelihood and the prior for $\boldsymbol{\pi}$. Four prior cohesions are introduced. Differently from what was considered by Hegarty and Barry (2008), all of them are location dependent and usually dependent on the distance between areas.

2. Preliminary Concepts

Consider n contiguous geographical regions such as those in Figure 2. The map is identified with an undirected graph $\mathcal{G} = (V, E)$, where V is the set of vertices or nodes representing the areas and E is the set of edges connecting pairs of vertices and representing the adjacency relationship among regions (see Figure 2, map on the top left). If there is an edge between vertices i and j we say that the corresponding areas are neighboring areas. A *path* from node v_1 to node v_m is a sequence of nodes $v_1, v_2, \ldots, v_m$ which are connected by edges $(v_1, v_2), \ldots, (v_{m-1}, v_m)$ and with $v_i \neq v_{i+1}$ for $i = 1, \ldots, n$. All vertices are distinct except, possibly, the initial and final vertices v_1 and v_m. This exceptional case, a path with $v_1 = v_m$, is a *circuit*. A graph is said to be *connected* if, for any pair of nodes v_i and v_j, there is at least one path connecting them.

A *spatial cluster* is defined as any subset of nodes forming a connected subgraph. The graph $\mathcal{G}$ is partitioned into c spatial clusters $\mathcal{G}_1, \ldots, \mathcal{G}_c$ if the clusters are disjoint and $\mathcal{G} = \bigcup_i \mathcal{G}_i$, where $1 \leq c \leq n$. A spatial partition $\pi = \{\mathcal{G}_1, \ldots, \mathcal{G}_c\}$ can be viewed as a function from the set $\{1, \ldots, n\}$ that labels the areas into the set $\{1, \ldots, c\}$ of spatial cluster labels.

A *spanning tree* $\mathcal{T}$ of a graph $\mathcal{G}$ is a fundamental concept in our work. It is a connected subgraph with no circuits containing all nodes of $\mathcal{G}$. The second and third maps on the top row in Figure 2 show two spanning trees associated with the first graph. In a spanning tree, any two nodes of $\mathcal{G}$ are connected by a unique path and the number of edges in $\mathcal{T}$ is $n - 1$. This implies that the removal of any $c - 1$ edges from $\mathcal{T}$ partitions the graph $\mathcal{G}$ into c spatial clusters. The last property makes the spanning tree a useful tool for spatial clustering problems. This can be seen in the bottom row where three different partitions are shown. The last two are based on the removal of only four edges from the corresponding spanning trees in the top row.

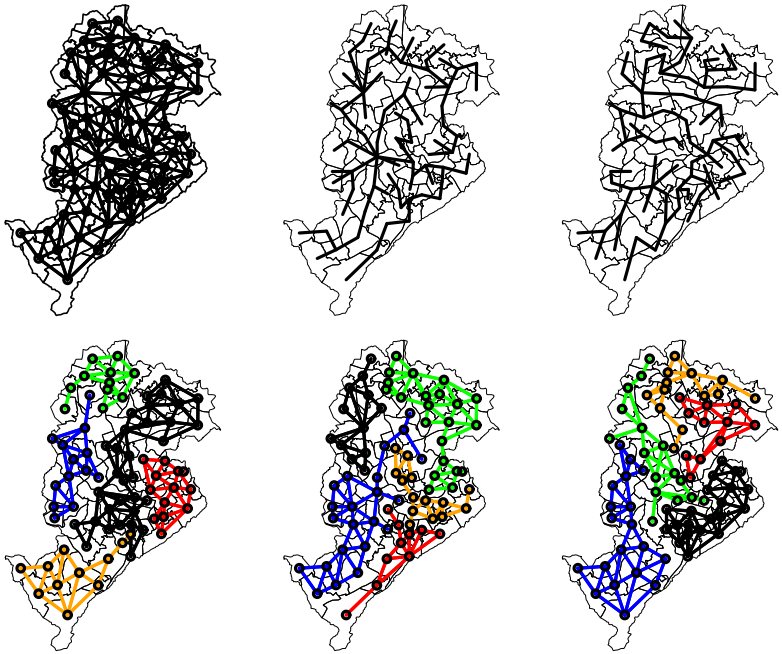


Figure 2: Map of Belo Horizonte city partitioned into 81 administrative planning units superimposed by the adjacency neighborhood graph (first map in the top row) and two spanning trees (other two maps in the top row). The bottom row shows three partitions into five spatial clusters.

Usually, there are many possible spanning trees associated with a given graph. A special kind of spanning tree can be obtained when we associate a cost or weight to each edge. The *cost of a graph* is the sum of its weights. A *minimum spanning tree* is a spanning tree with minimum cost. The minimum spanning tree is not necessarily unique. A sufficient condition for uniqueness is that the pairwise costs are distinct numbers.

Consider a spanning tree $\mathcal{T}$ and a partition π of $\mathcal{G}$ into c disjoint spatial clusters $\mathcal{G}_1, \ldots, \mathcal{G}_c$. We say that π is *compatible* with $\mathcal{T}$ if π can be obtained by pruning $c-1$ edges from $\mathcal{T}$, and we denote this by $\pi \prec \mathcal{T}$. Otherwise, they are incompatible and we write $\pi \not\prec \mathcal{T}$. The extremely large number of possible partitions of a graph is reduced tremendously by considering only those compatible with a given spanning tree. For example, there

are only $n-1$ possible partitions of a particular spanning tree into two spatial clusters while in the original graph this number is of order $O(2^n)$.

These purely spatial graph concepts can be easily extended to the space-time situation. We assume that the same map is observed for T time periods, that is, areas are not created or deleted during the observation period. Hence, we can stack the sequence of maps creating a three-dimensional lattice with nodes indexed by (t, i) where i denotes the geographical unit and t, the time. Edges between nodes (t, i) and (t, j) at the same time t are specified based only on the adjacency between the areas, as described before. Additionally, each node (t, i) is connected to itself and to all its adjacent neighboring regions at time $t+1$. Since the remaining concepts of spanning trees, paths and others are defined for general graphs, they are also valid for this extended three-dimensional graph $\mathcal{G} = (V, E)$.

3. Spatio-Temporal PPM Driven by Spanning Trees

Let $\boldsymbol{Y} = (\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_T)$ and $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_T)$ where $\boldsymbol{Y}_t = (Y_{t1}, \ldots, Y_{tn})$ and $\boldsymbol{\theta}_t = (\boldsymbol{\theta}_{t1}, \ldots, \boldsymbol{\theta}_{tn})$, respectively, denote the observable variables and the vector of parameters for the n regions of a map, at time t, $t = 1, \ldots, T$. We associate $(Y_{tr}, \boldsymbol{\theta}_{tr})$ with the node (t, r) in the graph. If the interest lies only on a purely spatial regionalization we take $T = 1$. Assume that $Y_{tj} \mid \boldsymbol{\theta}_{tj} \overset{ind}{\sim} f(Y_{tj} \mid \boldsymbol{\theta}_{tj})$, $j = 1, \ldots, n$ and $t = 1, \ldots, T$. Let $I = \{1, \ldots, nT\}$ be the set of labels for the nodes.

A major problem when partitioning a data set or a graph is the huge number of possible partitions that compose the search space. To introduce the cluster structure tackling this problem and making feasible the exploration of this space of possible partitions, let us assume a random spanning tree $\mathcal{T}$ selected from the set of all possible spanning trees associated with the graph. Denote by $\boldsymbol{\pi}$ a partition of I compatible with the selected $\mathcal{T}$ and assume that, given $\mathcal{T}$ and the partition $\boldsymbol{\pi} = \{\mathcal{G}_1, \ldots, \mathcal{G}_c\}$, there are common parameters $\boldsymbol{\theta}_{\mathcal{G}_k}$, $k = 1, \ldots, c$, such that, for all nodes with labels belonging to $\mathcal{G}_k$, we have that $\boldsymbol{\theta}_i = \boldsymbol{\theta}_{\mathcal{G}_k}, \quad i \in \mathcal{G}_k$. To establish notation, denote by $\boldsymbol{Y}_{\mathcal{G}_k}$ the set of observations associated with the nodes in $\mathcal{G}_k$. A spatio-temporal PPM induced by random spanning trees is the joint distribution of $(\boldsymbol{Y}, \boldsymbol{\theta}, \boldsymbol{\pi}, \mathcal{T})$, with $\boldsymbol{\pi} \prec \mathcal{T}$, and denoted by $(\boldsymbol{Y}, \boldsymbol{\theta}, \boldsymbol{\pi}, \mathcal{T}) \sim \text{SPPM}$, satisfying the following conditions:

(i) Given $\mathcal{T}$, $\boldsymbol{\pi} = \{\mathcal{G}_1, \ldots, \mathcal{G}_c\} \prec \mathcal{T}$ and $\boldsymbol{\theta}_{\mathcal{G}_1}, \ldots, \boldsymbol{\theta}_{\mathcal{G}_c}$, the observations $\boldsymbol{Y}_{\mathcal{G}_1}, \ldots, \boldsymbol{Y}_{\mathcal{G}_c}$ are independent and such that

$$Y_i \mid \boldsymbol{\theta}_{\mathcal{G}_k} \overset{iid}{\sim} f(Y_i \mid \boldsymbol{\theta}_{\mathcal{G}_k}),\ \forall\, i \in \mathcal{G}_k\,;$$

(ii) Given $\mathcal{T}$ and $\boldsymbol{\pi} = \{\mathcal{G}_1, \ldots, \mathcal{G}_c\} \prec \mathcal{T}$, the common parameters $\boldsymbol{\theta}_{\mathcal{G}_1}, \ldots, \boldsymbol{\theta}_{\mathcal{G}_c}$ are independent with joint distribution given by

$$\boldsymbol{\theta}_{\mathcal{G}_1}, \ldots, \boldsymbol{\theta}_{\mathcal{G}_c} \mid \boldsymbol{\pi}, \mathcal{T} \sim \prod_{k=1}^{c} f(\boldsymbol{\theta}_{\mathcal{G}_k})\,;$$

(iii) Given $\mathcal{T}$, the prior distribution of $\boldsymbol{\pi} \prec \mathcal{T}$ is a product distribution such that, for each partition $\{\mathcal{G}_1, \ldots, \mathcal{G}_c\}$ for $c \in \{1, \ldots, nT\}$, **we have**

$$\mathbb{P}(\boldsymbol{\pi} = \{\mathcal{G}_1, \ldots, \mathcal{G}_c\} \mid \mathcal{T}) = \frac{\prod_{k=1}^{c} \kappa(\mathcal{G}_k)}{\sum_{\mathcal{G}'_k \in \mathcal{C}(\mathcal{T})} \prod_{k=1}^{c} \kappa(\mathcal{G}'_k)}\,, \tag{1}$$

where $\kappa(\mathcal{G}_k) \geq 0$ denotes the prior cohesion associated to the subgraph $\mathcal{G}_k$ and represents how likely elements in $\mathcal{G}_k$ are clustered a priori. The summation in the denominator is over all the elements in the set $\mathcal{C}(\mathcal{T})$ which represents all the $2^{nT-1}-1$ partitions compatible with the specific spanning tree $\mathcal{T}$ in which we are conditioning.

(iv) The prior distribution of the spanning tree $\mathcal{T}$ is $\mathbb{P}(\mathcal{T})$ with the support on the set of all possible spanning trees of $\mathcal{G}$.

Usually, there is very little prior information to guide our choice of this last distribution. Hence, we assume that $\mathcal{T}$ is uniformly distributed over the space of spanning trees of the original graph.

The partition of a graph consists simply of the removal of a set of edges resulting in disjoint connected subgraphs. By conditioning on a spanning tree, this task is substantially simplified. We have only $nT-1$ edges and the removal of any $c-1$ of them immediately partition the original graph into c space-times clusters. We want to emphasize that we do not prune a single and fixed spanning tree. Rather, we randomly walk on the space of all spanning trees and, at each step, we generate a new partition. The final inference about the partition is obtained by integrating out over the spanning tree space.

With this motivation, we specify the prior cohesions as functions of ρ_e, the probability of removing a given edge e. For instance, if all edges have a common probability ρ, the prior cohesion related to group $\mathcal{G}_k$ becomes

$$\kappa(\mathcal{G}_k) = \begin{cases} (1-\rho)^{n_{\mathcal{G}_k}-1}\rho, & \text{if } k < c \\ (1-\rho)^{n_{\mathcal{G}_k}-1}, & \text{if } k = c, \end{cases} \tag{2}$$

where $n_{\mathcal{G}_k}$ is the number of remaining edges in $\mathcal{G}_k$. This prior cohesion is similar to that one considered by Barry and Hartigan (1993) to analyze problems involving the identification of contiguous clusters, such as in change point analysis in time series. In that context, this structure is a consequence of the Markovian behavior usually assumed for the change points.

Considering the prior cohesions in (2), given $\mathcal{T}$ and ρ, the prior probability of $\boldsymbol{\pi} = \{\mathcal{G}_1, \ldots, \mathcal{G}_c\}$ is given by

$$\mathbb{P}(\boldsymbol{\pi} \mid \mathcal{T},\ \rho) = \begin{cases} \rho^{(c-1)}(1-\rho)^{(nT-c)}, & \text{if } \boldsymbol{\pi} \prec \mathcal{T} \\ 0, & \text{otherwise.} \end{cases}$$

It is more relevant to obtain the unconditional prior distributions for the partition $\boldsymbol{\pi}$ and for the number C of clusters. Let $N_{\mathcal{T}}$ be the total number of spanning trees associated with $\mathcal{G}$ and $N_{\mathcal{T}}(\boldsymbol{\pi})$ the total number of spanning trees compatible with a partition π. It follows that, given ρ, the prior distributions for $\boldsymbol{\pi}$ and C are given respectively by

$$\begin{aligned} \mathbb{P}(\boldsymbol{\pi} \mid \rho) &= \sum_{\mathcal{T}} \mathbb{P}(\boldsymbol{\pi} \mid \mathcal{T},\ \rho)\mathbb{P}(\mathcal{T}) \\ &= \sum_{\mathcal{T}} \rho^{(c-1)}(1-\rho)^{(nT-c)}\mathbb{P}(\mathcal{T})\ I[\boldsymbol{\pi} \prec \mathcal{T}] \\ &= \rho^{(c-1)}(1-\rho)^{(nT-c)}\frac{N_{\mathcal{T}}(\boldsymbol{\pi})}{N_{\mathcal{T}}} \end{aligned}$$

and

$$\mathbb{P}(C = c \mid \rho) = \sum_{\boldsymbol{\pi}} I[\boldsymbol{\pi}, c]\mathbb{P}(\boldsymbol{\pi} \mid \rho) = \binom{nT-1}{c-1}\rho^{(c-1)}(1-\rho)^{(nT-c)}\frac{N_{\mathcal{T}}(\boldsymbol{\pi})}{N_{\mathcal{T}}}$$

where the last sum is over all partitions $\boldsymbol{\pi}$ and $I[\boldsymbol{\pi}, c]$ is an indicator function assuming 1, if the partition $\boldsymbol{\pi}$ has c clusters, and 0, otherwise.

Assuming *a priori* that $\mathbb{P}(\rho = 1/2) = 1$, it becomes clear the dependence of the prior distribution for $\boldsymbol{\pi}$ on the graph topology since $\mathbb{P}(\pi) = 2^{-nT+1}N_{\mathcal{T}}(\boldsymbol{\pi})N_{\mathcal{T}}^{-1}$. As $N_{\mathcal{T}}(\boldsymbol{\pi})$ depends on $\boldsymbol{\pi}$, we have that $\mathbb{P}(\boldsymbol{\pi})$ is not uniform over all possible partitions. As a consequence, the prior distribution of the number of clusters C is also influenced by the graph structure. Figure 3 shows a simple graph with three different partitions. The left hand side partition is given by $\pi_1 = \{\mathcal{G}_1, \mathcal{G}_2\} = \{\{1,2,3\},\{4,5,6\}\}$ and $N_{\mathcal{T}}(\boldsymbol{\pi}_1) = 9$. The middle partition is $\pi_2 = \{\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3\} = \{\{1,2,3\},\{4,5\},\{6\}\}$ and $N_{\mathcal{T}}(\boldsymbol{\pi}_2) = 6$. The number $N_{\mathcal{T}}(\boldsymbol{\pi})$ does not depend only on the number c of spatial clusters in $\boldsymbol{\pi}$. The right hand side partition is $\pi_3 = \{\mathcal{G}_1, \mathcal{G}_2\} = \{\{1,2,3,4,5\},\{6\}\}$, with the same number of clusters as $\boldsymbol{\pi}_1$ but with $N_{\mathcal{T}}(\boldsymbol{\pi}_3) = 6$.

It will be more common to assume a prior distribution for ρ such as Beta(r, s). Then

$$\begin{aligned} \mathbb{P}(\boldsymbol{\pi}) &= \frac{N_{\mathcal{T}}(\boldsymbol{\pi})}{N_{\mathcal{T}}}\frac{\Gamma(r+s)\Gamma(r+nT-c)\Gamma(s+c-1)}{\Gamma(r)\Gamma(s)\Gamma(nT+r+s-1)} \\ \mathbb{P}(C = c) &= \binom{nT-1}{c-1}\frac{N_{\mathcal{T}}(\boldsymbol{\pi})}{N_{\mathcal{T}}}\frac{\Gamma(r+s)\Gamma(r+nT-c)\Gamma(s+c-1)}{\Gamma(r)\Gamma(s)\Gamma(nT+r+s-1)}\,. \end{aligned}$$

Although the prior distribution for C is dependent on the graph topology, the expected number of clusters *a priori* is not. Given $\mathcal{T}$ and ρ, the number of removed edges from the

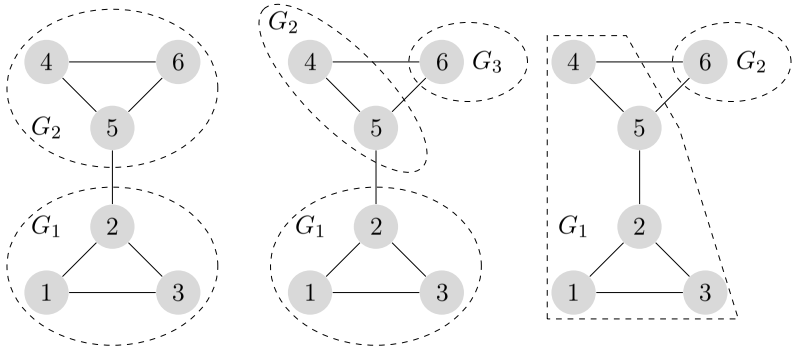


Figure 3: A graph composed by six nodes and with three different partitions $\boldsymbol{\pi}_1$ (left), $\boldsymbol{\pi}_2$ (middle), and $\boldsymbol{\pi}_3$ (right). The number of trees compatible with each partition is $N_{\mathcal{T}}(\boldsymbol{\pi}_1) = 9$, and $N_{\mathcal{T}}(\boldsymbol{\pi}_2) = N_{\mathcal{T}}(\boldsymbol{\pi}_3) = 6$.

tree compatible with the partition has a Binomial distribution with parameters $nT-1$ and ρ. Consequently, the expected number C of clusters is

$$\mathbb{E}(C) = \mathbb{E}\left(\mathbb{E}(C \mid \mathcal{T}, \rho)\right) = \mathbb{E}\left((nT-1)\rho + 1\right) = (nT-1)E(\rho) + 1\,. \tag{3}$$

As (3) reveals, the prior distribution of ρ determines how many clusters we expect *a priori*. A larger value for $\mathbb{E}(\rho)$ would indicate that we expect a high number of clusters, while a smaller value has the opposite effect. Note that the choice of a non informative uniform prior for ρ would stimulate the partition of the graph into a large number of clusters, around $\mathbb{E}(C) = (nT+1)/2$.

4. Sampling Partitions Using Spanning Trees

The great advantage of removing edges from spanning trees for regionalization is the reduction of the search space of partitions. Originally, we have to move through the huge space of all partitions of the graph avoiding those that do not respect the spatial constraint. By pruning the spanning trees, the search space of partitions is drastically reduced and all partitions generated in this way will, by definition, respect the spatial constraint. After all, the spanning tree is built from the neighborhood structure and, therefore, the clusters formed will be spatially or spatio-temporally connected.

To sample from the posterior distribution of $(\boldsymbol{\theta}, \boldsymbol{\pi}, \mathcal{T}, \rho \mid \boldsymbol{Y})$, we introduce an MCMC algorithm. The full conditional distributions of $\boldsymbol{\theta}_i$, $i = 1, \ldots, nT$, ρ are, respectively, given by:

$$\begin{aligned}
\boldsymbol{\theta}_i \mid \boldsymbol{\pi}, \mathcal{T}, \rho, \boldsymbol{Y} &\sim \mathbb{P}(\boldsymbol{\theta}_{\mathcal{G}^*} \mid \boldsymbol{Y}_{\mathcal{G}^*}), \;\; \text{if} \;\; i \in \mathcal{G}^*, \\
\rho \mid \boldsymbol{\theta}, \boldsymbol{\pi}, \mathcal{T}, \boldsymbol{Y} &\sim \text{Beta}(r + c - 1, s + nT - c)\,.
\end{aligned}$$

The full conditional distribution for $\mathcal{T}$ assuming a general prior cohesion is given by

$$\mathbb{P}(\mathcal{T} \mid \boldsymbol{\theta}, \boldsymbol{\pi}, \mathcal{T}, \boldsymbol{Y}) \propto \begin{cases} \left(\sum_{\mathcal{C}(\mathcal{T})} \prod_{k=1}^{c} \kappa(\mathcal{G}_k)\right)^{-1}, & \text{if } \boldsymbol{\pi} \prec \mathcal{T} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Assuming the prior cohesion specified in (2), this full conditional distribution assumes the simple uniform distribution

$$\mathbb{P}(\mathcal{T} \mid \boldsymbol{\theta}, \boldsymbol{\pi}, \mathcal{T}, \boldsymbol{Y}) \propto \begin{cases} 1, & \text{if } \boldsymbol{\pi} \prec \mathcal{T} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

because, in this case, the denominator in (1) is constant and equal to 1 for any spanning tree $\mathcal{T}$:

$$\sum_{\mathcal{C}(\mathcal{T})} \prod_{k=1}^{c} \kappa(\mathcal{G}_k) = \sum_{c=1}^{nT} \binom{nT-1}{c-1} \rho^{(c-1)} (1-\rho)^{(nT-c)} = 1\,.$$

A more efficient algorithm that has a fast convergence is obtained when we integrate out $\boldsymbol{\theta}$ and ρ from the full conditional distribution $\mathbb{P}(\boldsymbol{\pi} \mid \boldsymbol{\theta}, \mathcal{T}, \rho, \boldsymbol{Y})$ thus obtaining

$$\mathbb{P}(\boldsymbol{\pi} \mid \boldsymbol{Y}, \mathcal{T}) = \begin{cases} f_{\mathcal{T}}(\boldsymbol{Y}) \frac{\Gamma(r+s)\Gamma(s+nT-c)\Gamma(r+c-1)}{\Gamma(r+s+nT-1)\Gamma(r)\Gamma(s)}, & \text{for } \pi \prec \mathcal{T} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $f_{\mathcal{T}}(\boldsymbol{Y}) = \prod_{k=1}^{c} \int_{\Theta} f(\boldsymbol{Y}_{\mathcal{G}_k} \mid \theta_{\mathcal{G}_k}) f(\theta_{\mathcal{G}_k})\, \mathrm{d}\theta_{\mathcal{G}_k}$. We reinforce that $f_{\mathcal{T}}(\boldsymbol{Y})$ is calculated only for partitions $\boldsymbol{\pi}$ compatible with a given spanning tree $\mathcal{T}$. The posterior distribution for the partition $\boldsymbol{\pi}$ should be obtained by marginalizing over all spanning trees compatible with a given partition: $\mathbb{P}(\boldsymbol{\pi} \mid \boldsymbol{Y}) = \sum_{\mathcal{T} \prec \boldsymbol{\pi}} \mathbb{P}(\boldsymbol{\pi} \mid \boldsymbol{Y}, \mathcal{T})\, \mathbb{P}(\mathcal{T} \mid \boldsymbol{Y})$. This would imply the need to additionally derive the posterior $\mathbb{P}(\mathcal{T} \mid \boldsymbol{Y})$. Rather than following this approach, we decide to adopt a different strategy, by sampling from the joint posterior of compatible pairs of $\mathcal{T}$ and $\boldsymbol{\pi}$. The challenge is to sample partitions and the spanning trees that are compatible to each other. The proposed strategy to achieve this is presented in the following.

4.1. Sampling a Partition Compatible with the Current Tree

The transformation suggested by Barry and Hartigan (1993) inspired the Gibbs sampler we describe here. A partition $\boldsymbol{\pi}$ of the graph can be transformed into a vector $\boldsymbol{U}$ of $nT-1$ dependent binary variables, given a compatible spanning tree $\mathcal{T}$. The coordinate U_i of $\boldsymbol{U}$ is 1 if the i-th edge is not removed from the tree to form $\boldsymbol{\pi}$, and 0, otherwise. With this multidimensional random variable, we can sample from the posterior of $\boldsymbol{\pi}$ using Gibbs sampler. Let $\boldsymbol{U}_{-i} = \{U_1, \cdots, U_{i-1}, U_{i+1}, \cdots, U_{nT-1}\}$. To decide if each edge in the tree should be removed or not, it is sufficient to know the ratio between the $\mathbb{P}(U_i = 1 \mid \boldsymbol{U}_{-i}, \mathcal{T}, \boldsymbol{Y})$ and $\mathbb{P}(U_i = 0 \mid \boldsymbol{U}_{-i}, \mathcal{T}, \boldsymbol{Y})$. For the model assumed here such ratio is given by

$$R_i = \frac{f_{\mathcal{T}}^{(1)}\left(\boldsymbol{Y}_{\mathcal{G}_k}\right)}{f_{\mathcal{T}}^{(0)}\left(\boldsymbol{Y}_{\mathcal{G}_k}^{(L)}\right) f_{\mathcal{T}}^{(0)}\left(\boldsymbol{Y}_{\mathcal{G}_k}^{(R)}\right)} \frac{(r+c-2)}{(s+nT-c)}\,,$$

where $f_{\mathcal{T}}^{(1)}\left(Y_{\mathcal{G}_k}\right)$ is the prior predictive of the group formed when the edge i is present, whereas $f_{\mathcal{T}}^{(0)}\left(\boldsymbol{Y}_{\mathcal{G}_k}^{(L)}\right)$ and $f_{\mathcal{T}}^{(0)}\left(\boldsymbol{Y}_{\mathcal{G}_k}^{(R)}\right)$ are the predictive distributions for the observations of the two groups formed when the edge i is removed from the tree $\mathcal{T}$ pruned by $\boldsymbol{U}_{-i}$.

Thus, we can sample from the distribution of U_i simply by sampling a uniform value $u \sim \text{Uniform}(0,1)$ and using the following accept/reject criterion:

$$U_i = \begin{cases} 1, & \text{if } R_i \geq \frac{u}{1-u} \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

4.2. Sampling a tree compatible with the current partition

We need to sample trees that are compatible with the current partition. The valid trees are uniformly distributed over the subset of trees compatible with the partition as shown in (5). The generation of uniformly distributed spanning trees **without constraints** is a subject of study since 1989 (Broder, 1989). Wilson algorithm (Wilson, 1996) is the main algorithm in the literature and it is based on a certain random walk on the adjacency graph. Under the constraint that the generated trees must be compatible with the current partition at each step of the Gibbs sampler, Wilson's algorithm could be used using a rejection sampling approach. Uniformly select a spanning tree from the adjacency graph $\mathcal{G}$ and reject it if it is not compatible with the current partition. Otherwise, accept it. This is a very expensive way to generate uniformly from the set of spanning trees compatible with $\boldsymbol{\pi}$. For example, suppose that $\mathcal{G}$ is a regular 26×26 rectangular lattice split into two spatial clusters, the

upper half and the lower half. We did not obtain a single compatible spanning tree after uniformly generating 3.8 million spanning trees. Another example is the Brazilian map partitioned into 5564 municipalities (see Figure 7a), again split only into two groups, one being the municipalities within the three most Southern states of Brazil (see Figure 13 for the states' boundaries). This partitioning of the map has 67 edges connecting municipalities of different groups. After 1.5 million simulations by Wilson's method, none was compatible with that partition.

Another possibility is to adapt Wilson's algorithm to obey the constraints imposed by the current partition. A reviewer suggested the following procedure. Suppose a clustering as in Figure 3 (middle). First sample a uniform spanning tree by Wilson's algorithm within each of $\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3$. Next, define a collapsed graph where the nodes are the three clusters and we set an edges between $\mathcal{G}_i$ and $\mathcal{G}_j$ if there is at least one edge between the two subgraphs. Sample a uniform spanning tree on that collapsed graph. Finally, for each edge sampled in the collapsed graph, sample one of the edge uniformly in the original bipartite graph.

While this procedure may sound promising at first, it, unfortunately fails to generate trees uniformly. What is problematic in this approach is that the number of edges between clusters can be imbalanced, which could lead to trees being sampled with different probability.

Consider the graph in Figure 4. We would first sample a uniform tree inside of each of the tree groups. In this example, there are 3 independent options for each group, so in this first step a choice is made with probability $\left(\frac{1}{3}\right)^3 = \frac{1}{27}$. Now, the collapsed graph is a complete graph of size 3 (a triangle). Once again, there is a choice between three spanning trees. Assume we uniformly select a spanning tree which contains an edge between Group A and Group B and between Group B and Group C. This choice has probability $\frac{1}{3}$. Finally, we need to uniformly choose among the corresponding edges between the clusters. For the connection between Group B and Group C, there is only one option, while for Group A and Group B, there are three choices. Uniformly selecting the edges, thus, will have a probability of $\frac{1}{3} \cdot 1 = \frac{1}{3}$. The final tree, then, will have been selected with probability $\frac{1}{27}\frac{1}{3}\frac{1}{3} = \frac{1}{247}$.

Consider now another possible combination of choices. For the first step, we select the same trees inside the groups, with probability $\frac{1}{27}$. For the second step, we choose the spanning tree of the collapsed graph which contains a connection between Group A and Group C and between Group C and Group B. This step, again, has probability $\frac{1}{3}$. Finally, to select the corresponding trees in the original bipartite graph, there is only one choice corresponding to each connection, namely (B2, C3) and (A2, C1). Therefore, the choice in this final step was made with probability 1. The final tree, then, was chosen with probability $\frac{1}{27}\frac{1}{3}1 = \frac{1}{81}$. Therefore, this method will sample the trees with a probability which is not uniform.

Instead of Wilson's algorithm, we employ another procedure to sample trees using a minimum spanning tree (MST) algorithm. We sample a new tree $\mathcal{T}$ compatible with the current partition π by assigning weights to the edges in the graph respecting this current partition in the following way. The edges that connect vertices belonging to the same cluster receive a low weight, obtained from a uniform distribution which generates low values (e.g. between 0 and 1). The edges that connect vertices belonging to different clusters receive a high weight, obtained from a uniform distribution which generates higher values (e.g.

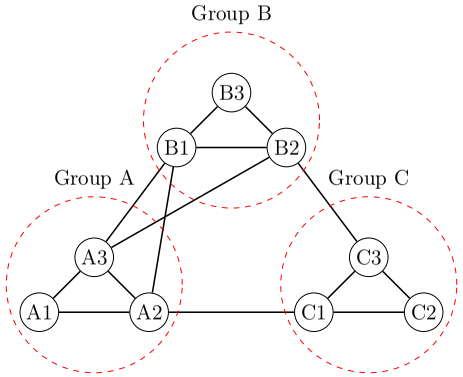


Figure 4: Counter example of graph where the approach of adapting Wilson's algorithm as a multi-step procedure to obey the constraints imposed by the partition fails to sample trees uniformly.

between 10 and 20). Once the weights are assigned, the minimum spanning tree is obtained and it is the new sampled tree, compatible with the current partition.

The reason for using these two sets of values is that the algorithm computes the spanning tree with minimum sum of weights. When we use weights in this way, we ensure that the tree will be compatible with the partition, since the edges with higher weights are added to the tree only when all possible connections through edges with a lower weight are already explored. So, it only adds a connection between clusters when all the possible connections inside a cluster have been visited.

The proposed algorithm works because the MST algorithm, either by Prim algorithm or by the Kruskal algorithm (Cormen et al., 2009), selects the edges according to their weights. Since all edges separating clusters have higher weights, they are selected only after the lower weight edges connecting vertices inside a cluster. The random attribution of weights ensures that the tree will respect the partition and, since we randomly assign weights each time we sample a tree, we get a random tree, even though the algorithm is deterministic.

This procedure does not sample exactly from a uniform distribution. To see this, consider the following example. Consider the graph in the left hand side of Figure 5 where we disregard any partition. There are 5 spanning trees that do not include the $(3-4)$ edge, each one obtained by the removal of $(3-4)$ and one of the remaining edges. There are 6 other spanning trees including the $(3-4)$ edge: we need to prune one of the three edges located at the left-hand side of $(3-4)$ and one of the two located at its right-hand side. Our method would assign independent $U(0,1)$ random variables $W_1, \ldots, W_6$ to the six edges and then select its minimum spanning tree. This is equivalent to substitute the W_i's by their rank statistics $R_1, \ldots, R_6$ where $R_i = \sum_{j=1}^{6} I[W_j \leq W_i]$ and to select the minimum spanning tree using these ranks. The total number of possible ranks assignment to the

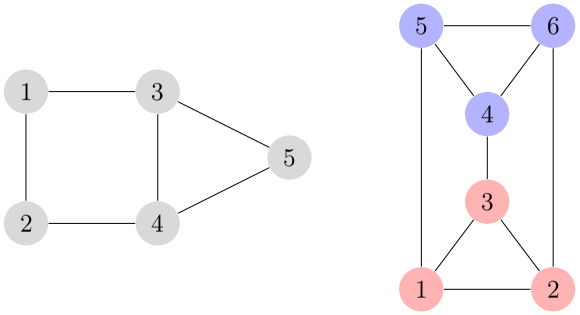


Figure 5: Two graphs to discuss different possibilities to randomly generate spanning trees.

edges is equal to the number of permutations of the six edges, that is, $6! = 720$. Each rank statistic configuration is equally likely and each one of the 11 spanning trees is associated with a subset of these 720 ranking patterns. The cardinality of these subsets should be the same if we have a uniform sampling procedure. However, 11 is not a divisor of 720 and therefore some spanning trees are selected with higher frequency than others.

However, this simple and fast procedure is a good approximation for the uniform sampling of spanning trees and we show this by means of an example. Consider the graph in the right-hand side of Figure 5 with 6 nodes divided into 2 groups (nodes 1-3 and nodes 4-6). A complete enumeration shows that there are 27 spanning trees compatible with this partition, which, if uniformly sampled, should have a selection probability equal to $1/27 \approx 0.03704$. Using Wilson's algorithm, we selected 100 thousand random spanning trees. The observed frequencies of the 27 spanning trees varied from 0.03588 to 0.03831. Using our method based on 100 thousand simulations of $U(0, 1)$ random variables for the edges within cluster and $U(10, 20)$ random variables for the edges between clusters, the observed frequencies of the spanning trees varied from 0.03588 to 0.03811. Additionally, Wilson's algorithm requires generating about 3 spanning trees before one is accepted (compatible with the partition), even in this small example, while for our approach each generated tree is guaranteed to be compatible with the partition.

5. Analysis of Simulated Data Sets

In this section, we evaluate the performance of our model for regionalization in spatial settings only. The main goal is to compare it with the following state-of-art regionalization techniques: *SKATER*, through its implementation on the `spdep` R package[1], the original Automatic Zoning Procedure (*AZP*), and its variations with Simulated Annealing (*AZP_SA*), Tabu (*AZP_TABU*), Reactive tabu (*AZP_RTABU*), the Automatic Regionalization with Initial Seed Location (*ARISEL*), the Max-p-regions Tabu model (*MAXP*) and the "A Multidirectional Optimum Ecotope-Base Algorithm" (*AMOEBA*), all implemented

1. `http://cran.r-project.org/package=spdep`

in the Python ClusterPy[2] library (Duque et al., 2011). For Poisson data sets, the proposed spatial PPM is also compared to the BDCD model, implemented in C++ code (available at `http://www.statistik.lmu.de/sfb386/software/bdcd/download.html`) and to BPM, implemented in R software by ourselves.

To define the spatial structure of the simulated data (coordinates, shapes and spatial adjacency), we consider two geographical neighborhood structures of Brazilian regions. In the Gaussian data sets, inspired by the human development index map shown in Section 6, we considered 853 Brazilian regions grouped into three clusters: two large clusters and a third small cluster, composed by 10 regions and islanded in the middle of one of the other two clusters. The small cluster mean is very different from that in surrounding regions. In Scenario 1, the observations within each cluster are *iid* generated from normal distributions. To mimic the observed data, we choose means $\mu_i = 0.73$ and precision $\tau_i = 4000$, for the small cluster, $\mu_i = 0.63$ and $\tau_i = 754.20$, for the large cluster containing the small one, and $\mu_i = 0.70$ and $\tau_i = 878.35$, for the other large cluster. In Scenario 2, the spatial structure also presents three clusters but inside each cluster, the distribution varies slightly from a region to another. The regions means within the large cluster containing the smaller one vary from 0.612 up to 0.685 and precision varies between 435 and 484. The other large cluster have $\mu_i \in (0.663, 0.726)$ and $\tau_i \in (471, 497)$. Finally, the small cluster is composed by three areas with common mean equal to 0.780 and precision 10000.

For the Poisson data sets, we were inspired by the epidemiological studies presented in Section 6. We considered smaller areas in the southern region of Brazil, the 1188 Brazilian municipalities divided into clusters. Each observation y_i simulates a cancer death count and the expected count E_i is extracted from the real data analysis and based on true population in each area. Motivated by the high mortality rates observed in the lung cancer data, we simulated Poisson data with high rates in Scenarios 3 and 4, while in Scenarios 5 and 6, we assumed low rates similar to those observed in the bladder cancer data. Rates assumed in high rate scenarios are, on average, 8 times higher than those for the low rate scenarios. Similarly to the normal case, we assume areas with the same rate (Scenarios 3 and 5) and with distinct but similar rates (Scenarios 4 and 6) within each cluster.

In Scenarios 3 and 4, the map is divided into 10 clusters with the common rates in Scenario 3 ranging from 0.45 to 1.40. The distinct rates for the 1188 regions considered in Scenario 4 range from 0.45 to 1.75. In bladder cancer inspired low rate scenarios, the map is divided into 13 clusters. The common rates in Scenario 5 has values varying from 0.24 to 1.74, while for Scenario 6, the distinct parameters for each region vary from 0.77 to 1.42.

Concerning the inference, we assume that $Y_i \mid \mu_{\mathcal{G}_k}, \tau_{\mathcal{G}_k} \overset{iid}{\sim} \text{Normal}\left(\mu_{\mathcal{G}_k}, \tau_{\mathcal{G}_k}^{-1}\right)$, for all $i \in \mathcal{G}_k$, in the normal case. We also consider the normal-gamma prior distribution with $\mu_{\mathcal{G}_k} \mid \tau_{\mathcal{G}_k} \sim \text{Normal}\left(m, [v\tau_{\mathcal{G}_k}]^{-1}\right)$ and $\tau_{\mathcal{G}_k} \sim \text{Gamma}\,(a, b)$, where $m = 0.65$, $v = 1$, $a = 400$ and $b = 1$. For the Poisson data, $Y_i \mid \phi_{\mathcal{G}_k} \overset{iid}{\sim} \text{Poisson}\,(E_i \cdot \phi_{\mathcal{G}_k})$ and $\phi_{\mathcal{G}_k} \sim \text{Gamma}(a, b)$, where $a = b = 2$. This prior distribution for $\phi_{\mathcal{G}_k}$ puts probability mass around 1, as we usually expect for the relative risk. Furthermore, 90% of its probability mass is concentrated between 0.18 and 2.37, which is a reasonable range for many human disease relative risks. As usually done in real problems, in our analysis, the expected count E_i is assumed to

2. `http://www.rise-group.org/`

be known. Besides, we assume that each edge is removed from the spanning tree with probability $\rho \sim \text{Beta}(5, 1000)$.

Our algorithm was implemented in C++ and is available upon request. For the MCMC we run a chain of 5000 iterations, skipping the first 500 samples as a *burn-in* period and, to avoid correlation, we use a thinning of 5 simulated values.

5.1. Evaluation Metrics

We consider two groups of metrics. The first group is based on the estimates of the parameters that index the data distributions. We consider the mean absolute error (MAE), the mean relative error (MRE) and the mean squared error (MSE) based on the distance between the true parameters and their estimates. Under BPM, BDCD and the proposed PPM the parameter estimates in each region are the posterior means. Under the usual regionalization methods, such estimates are obtained by averaging the observations Y_i into the cluster. For normal data, these are given by $\hat{\theta}_i = \sum_{j \in \mathcal{G}_k} Y_j / n_k \quad \forall i \in \mathcal{G}_k$ and, in the Poisson case, they are $\hat{\theta}_i = [\sum_{j \in \mathcal{G}_k} Y_j][\sum_{j \in \mathcal{G}_k} E_j]^{-1}$, $\forall i \in \mathcal{G}_k$.

The other set of metrics is based on the difference between the true clusters used to generate the data and the clusters obtained by the different methods. Each pair of areas is labeled as positive if they are in the same cluster and as negative if they are in different clusters. We looked at the number of true positives (TP) classifications, the number of pairs that are in the same cluster both, in the true partition as well as in the estimated one. False positive classifications (FP) is the number of pairs assigned to distinct clusters but that actually are in the same cluster in the true partition. True negatives (TN) classification is the number of pairs correctly assigned to distinct clusters. Finally, false negatives (FN) classification is the number of pairs incorrectly assigned to distinct clusters. In our evaluation, we consider the rand measure (RAND), the F_1 score (F1), the Jaccard index (JI) and the Fowlkes-Mallows index (FM) which are given, respectively, by

$$RAND = [TP + TN] \cdot [TP + FP + FN + TN]^{-1},$$
$$F1 = 2 \cdot P \cdot R \cdot [P + R]^{-1},$$
$$JI = TP \cdot [TP + FP + FN]^{-1},$$
$$FM = \sqrt{P \cdot R},$$

where $P = TP \cdot [TP + FP]^{-1}$ and $R = TP \cdot [TP + FN]^{-1}$.

Most of the traditional regionalization techniques considered in our analysis requires the pre-definition of the number of clusters to be generated as an input. In all cases, we use three different values: the true number c of clusters as well as 3 more and 3 less (1 less for the normal case) clusters than the true number. The max-p-regions model clusters a set of geographic areas into the maximum number of homogeneous regions such that the value of a spatially extensive regional attribute is above a predefined threshold value. In clusterPy we measure heterogeneity as the within-cluster sum of squares from each area to the attribute centroid of its cluster. The parameter values shown in the tables for this method correspond to this threshold.

To summarize the posterior information about the random partition provided by our method, we take the underlying graph of the spatial structure of the data and, for each edge

connecting neighboring areas, we compute how often they are in the same cluster based on the generated partitions. This percentage is assigned to each edge. Then, we trim the edges by removing all those which are below a certain threshold. Once the infrequent edges are removed, the remaining components of the graph define the clusters. The reasoning is that the removed edges are exactly those which are frequently crossing the borders between clusters in the sampled partitions. By removing them, the bulk of the clusters frequently present in the sampled partitions remain connected in the graph. We consider three thresholds: 70%, 80% and 90%. The same strategy is used for the BPM and BDCD methods.

5.2. Results

Tables 1 to 3 show the model fit measures for the proposed model and the competitor methods for normal and Poisson data. For each evaluation measure, the best outcome is underlined and shown in bold.

In almost all the simulated data sets, our model outperformed all the other methods. The only scenarios where our method had inferior results were in the normal data set with common parameter, where the MAE for the *SPPM* was the second best and the Poisson data set with low rate and common parameter where *SKATER* had the lowest MRE. However, in both cases, the *SPPM* had better performance according to all other metrics we consider to evaluate the models. Furthermore, in these two exceptional cases, even in those losing metrics, *SPPM* had a very close value to the better competitors.

In all data sets, particularly in the Poisson data sets, the error metrics (MAE, MSE, MRE) for our method were from 1.5 to 5 times smaller than the other methods. For *SKATER*, *ARISEL* and the *AZP*-type methods, we notice that the bias in the parameter estimates (either the normal mean or the Poisson rate) depends on the number of clusters we assume to implement the method as well as on the type of scenario. We notice, for instance, that *SKATER* produced less biased estimates for such parameters in all scenarios if the assumed number of cluster is $c - 3$ (and $c - 1$ for the normal case), the exception occurring in Scenario 1. Moreover, in Scenario 3, less biased estimates are obtained by all such methods, except *SKATER*, if the number of clusters is correctly assumed. This bias may be explained by the poor capacity of these methods to identify correctly the clusters. The non-stochastic methods were favored in the sense that they received good information about the number of clusters. We either provided the exact number or a close one. In some cases, even with the correct number of clusters, the *AZP*-type methods had a worse performance than with the wrong number of clusters.

Our method also achieved better results if compared to the Bayesian models *BPM* and *BDCD* (only implemented for Poisson models). *BPM* had a very poor performance for both, cluster identification and parameter estimation, in all four Poisson scenarios. *BDCD* and the proposed *SPPM* are comparable with respect to the parameter estimation, with proposed model usually providing only slightly better estimates. However, the proposed model has much better performance for cluster identification, except for Scenario 3 and thresholds 70% and 80%, and Scenarios 4 and 6 and thresholds 70%.

Table 1: Model fit summaries for all methods, Normal data, Scenarios 1 and 2.

Method	Parameter	MAE	MSE	MRE	RAND	F1	JI	FM
				Common parameters				
SPPM	70%	0.00466	**0.00010**	**0.00701**	**96.50**	**96.51**	**93.26**	**96.51**
	80%				96.30	96.26	92.79	96.27
	90%				94.74	94.58	89.72	94.65
SKATER	$c-1$	0.00745	0.00024	0.01132	92.08	92.22	85.56	92.22
	c	0.00503	0.00016	0.00771	94.10	94.08	88.82	94.09
	$c+3$	0.00909	0.00027	0.01389	89.24	88.56	79.47	88.79
AZP	$c-1$	0.03526	0.00125	0.05304	50.34	66.92	50.28	70.87
	c	0.00488	0.00017	0.00751	92.52	92.34	85.77	92.39
	$c+3$	0.01158	0.00035	0.01799	84.98	83.11	71.10	83.85
AZP_SA	$c-1$	0.00488	0.00015	0.00751	96.03	96.07	92.44	96.07
	c	0.01902	0.00071	0.02865	67.15	62.10	45.04	62.93
	$c+3$	0.00849	0.00028	0.01304	81.07	77.92	63.82	79.14
AZP_TABU	$c-1$	0.02972	0.00104	0.04536	60.17	67.43	50.86	68.50
	c	0.00702	0.00021	0.01093	87.63	86.55	76.29	86.95
	$c+3$	0.01171	0.00037	0.01805	86.35	84.84	73.67	85.45
AZP_RTABU	$c-1$	0.02680	0.00108	0.03943	60.49	63.33	46.34	63.47
	c	0.00683	0.00023	0.01056	93.50	93.59	87.96	93.60
	$c+3$	0.00530	0.00022	0.00807	92.39	92.17	85.49	92.23
ARISEL	$c-1$	**0.00456**	0.00015	0.00701	96.02	96.07	92.43	96.07
	c	0.00945	0.00024	0.01481	88.77	87.92	78.45	88.23
	$c+3$	0.00542	0.00025	0.00824	86.91	85.72	75.01	86.14
AMOEBA	*None*	0.02497	0.00074	0.03770	70.89	64.90	48.04	66.45
MAXP	10	0.01243	0.00036	0.01879	51.45	7.89	4.10	19.15
	100	0.01551	0.00058	0.02317	65.74	55.54	38.44	58.36
				Distinct parameters				
SPPM	70%	**0.00705**	**0.00015**	**0.01065**	50.37	66.58	49.90	70.32
	80%				**90.97**	**90.84**	**83.22**	**90.86**
	90%				90.33	89.63	81.21	89.92
SKATER	$c-1$	0.01086	0.00037	0.01624	87.45	87.55	77.85	87.55
	c	0.01131	0.00039	0.01702	86.06	85.93	75.33	85.94
	$c+3$	0.01263	0.00041	0.01927	82.92	81.75	69.14	81.99
AZP	$c-1$	0.01213	0.00046	0.01822	85.08	85.40	74.51	85.40
	c	0.03308	0.00130	0.04894	52.96	59.88	42.74	60.49
	$c+3$	0.01162	0.00043	0.01741	86.41	86.51	76.23	86.51
AZP_SA	$c-1$	0.01102	0.00037	0.01651	87.88	88.07	78.69	88.08
	c	0.01256	0.00049	0.01891	73.41	69.46	53.21	70.33
	$c+3$	0.01281	0.00053	0.01931	76.11	74.31	59.12	74.57
AZP_TABU	$c-1$	0.02277	0.00113	0.03506	65.63	69.11	52.80	69.42
	c	0.01512	0.00066	0.02280	78.56	78.98	65.27	78.99
	$c+3$	0.01566	0.00055	0.02374	73.43	68.39	51.96	69.77
AZP_RTABU	$c-1$	0.01386	0.00054	0.02068	83.51	83.73	72.02	83.73
	c	0.01317	0.00050	0.01978	84.15	84.15	72.63	84.15
	$c+3$	0.01394	0.00051	0.02076	79.09	77.26	62.95	77.61
ARISEL	$c-1$	0.01206	0.00044	0.01808	85.87	86.12	75.62	86.12
	c	0.01149	0.00040	0.01722	87.01	87.17	77.26	87.17
	$c+3$	0.01477	0.00047	0.02199	77.31	73.55	58.16	74.69
AMOEBA	*None*	0.03450	0.00144	0.05156	61.44	52.00	35.14	53.76
MAXP	10	0.01180	0.00034	0.01751	51.58	8.63	4.51	19.86
	100	0.01774	0.00053	0.02647	60.15	43.46	27.76	48.12

Table 2: Model fit summaries for all methods, Poisson data with high rate, Scenarios 3 and 4.

Method	Parameter	MAE	MSE	MRE	RAND	F1	JI	FM
				Common parameters				
SPPM	70%	**0.03054**	**0.00422**	**0.03797**	49.14	42.08	26.65	51.24
	80%				76.54	59.65	42.50	63.71
	90%				**90.59**	**74.28**	**59.09**	**74.32**
SKATER	$c-3$	0.11544	0.02046	0.12527	74.76	54.92	37.86	58.11
	c	0.12579	0.03044	0.13527	75.50	54.12	37.10	56.62
	$c+3$	0.12618	0.03409	0.12778	77.03	54.06	37.04	55.77
AZP	$c-3$	0.14481	0.03003	0.15500	69.35	43.77	28.01	46.01
	c	0.10819	0.02484	0.11362	74.69	57.25	40.11	61.45
	$c+3$	0.13351	0.02751	0.14128	79.76	51.97	35.11	52.26
AZP_SA	$c-3$	0.17650	0.04653	0.21710	51.07	42.05	26.62	50.49
	c	0.07600	0.01620	0.08459	76.36	51.36	34.55	52.70
	$c+3$	0.11834	0.03139	0.12526	74.29	47.72	31.34	49.07
AZP_TABU	$c-3$	0.19176	0.06519	0.23379	49.84	36.52	22.34	42.84
	c	0.10617	0.02122	0.11570	73.60	51.01	34.23	53.47
	$c+3$	0.13359	0.02907	0.14364	81.11	48.89	32.35	48.90
AZP_RTABU	$c-3$	0.18137	0.05388	0.22075	53.29	41.53	26.20	48.90
	c	0.10348	0.02615	0.11161	78.00	47.37	31.03	47.60
	$c+3$	0.13224	0.03324	0.14001	78.60	50.97	34.20	51.45
ARISEL	$c-3$	0.13210	0.02655	0.14694	66.90	47.90	31.49	52.43
	c	0.10389	0.02524	0.11519	73.19	55.76	38.66	60.23
	$c+3$	0.12787	0.03623	0.13810	77.63	46.42	30.22	46.64
AMOEBA	*None*	0.16781	0.06013	0.18080	69.82	37.53	23.10	38.48
MAXP	10	0.08963	0.01851	0.10448	81.91	10.06	5.30	20.76
	100	0.12921	0.02821	0.15403	82.93	46.76	30.52	47.50
BPM	70%	0.35841	0.30151	0.53348	49.90	41.00	25.79	49.33
	80%				49.90	41.00	25.79	49.33
	90%				49.90	41.00	25.79	49.33
BDCD	70%	0.03637	0.00528	0.04562	74.61	57.73	40.58	62.21
	80%				83.63	61.93	44.85	62.41
	90%				82.33	14.34	7.72	25.57
				Distinct parameters				
SPPM	70%	**0.04759**	**0.00544**	**0.05465**	51.13	42.04	26.62	50.46
	80%				**89.02**	**70.40**	**54.32**	**70.41**
	90%				88.95	65.78	49.01	66.73
SKATER	$c-3$	0.09821	0.02133	0.10367	82.90	62.00	44.93	62.83
	c	0.10061	0.02474	0.10698	84.04	61.96	44.89	62.28
	$c+3$	0.10650	0.02619	0.11220	85.39	62.13	45.06	62.15
AZP	$c-3$	0.11725	0.02275	0.12573	72.38	46.57	30.35	48.37
	c	0.09930	0.01941	0.10935	71.55	50.88	34.12	54.29
	$c+3$	0.11223	0.02456	0.12804	74.61	43.48	27.78	44.05
AZP_SA	$c-3$	0.12101	0.02373	0.14110	70.97	39.75	24.81	40.73
	c	0.09564	0.01968	0.10301	76.76	56.78	39.65	59.50
	$c+3$	0.07958	0.01855	0.08658	83.88	56.34	39.22	56.35
AZP_TABU	$c-3$	0.10113	0.01917	0.11041	69.77	50.18	33.49	54.22
	c	0.14293	0.03248	0.17143	64.21	36.72	22.49	38.95
	$c+3$	0.09638	0.01827	0.10530	76.34	46.68	30.45	47.22
AZP_RTABU	$c-3$	0.11068	0.01971	0.12723	67.98	48.34	31.87	52.53
	c	0.10050	0.02104	0.11083	68.54	49.37	32.78	53.70
	$c+3$	0.12937	0.02724	0.14326	78.26	44.22	28.39	44.25
ARISEL	$c-3$	0.09152	0.01625	0.09966	76.59	48.94	32.40	49.74
	c	0.11073	0.02656	0.11981	79.64	51.77	34.93	52.07
	$c+3$	0.09866	0.02000	0.10814	76.19	51.87	35.01	53.40
AMOEBA	*None*	0.17683	0.05720	0.19835	67.94	35.11	21.29	36.13
MAXP	10	0.09528	0.01730	0.10998	81.93	10.68	5.64	21.19
	100	0.13384	0.02775	0.16170	80.19	41.26	26.00	41.56
BPM	70%	0.35298	0.26530	0.51927	50.58	37.93	23.40	44.61
	80%				50.58	37.93	23.40	44.61
	90%				50.61	37.87	23.36	44.52
BDCD	70%	0.05641	0.00676	0.06367	67.69	46.47	30.27	50.13
	80%				83.00	29.55	17.34	35.75
	90%				82.76	16.59	9.04	29.32

Table 3: Model fit summaries for all methods, Poisson data with low rate, Scenarios 5 and 6.

Method	Parameter	MAE	MSE	MRE	RAND	F1	JI	FM
				Common parameters				
SPPM	70%	**0.07669**	**0.01905**	0.49370	65.43	78.99	65.27	80.78
	80%				65.98	79.17	65.52	80.87
	90%				**79.50**	**82.46**	**70.16**	**82.99**
SKATER	$c-3$	0.22172	0.12678	**0.36023**	49.73	52.20	35.32	53.72
	c	0.26417	0.16666	0.41019	46.72	47.16	30.85	49.27
	$c+3$	0.29476	0.23519	0.44595	45.43	44.73	28.81	47.16
AZP	$c-3$	0.24460	0.14382	0.74205	44.84	47.85	31.45	49.16
	c	0.23630	0.15310	0.60011	56.83	58.58	41.42	60.46
	$c+3$	0.27275	0.21656	0.78907	43.98	43.27	27.61	45.62
AZP_SA	$c-3$	0.22391	0.11542	0.74414	45.51	47.05	30.76	48.77
	c	0.18481	0.14430	0.66085	49.64	55.47	38.38	56.10
	$c+3$	0.17373	0.14279	0.67033	44.04	41.78	26.40	44.65
AZP_TABU	$c-3$	0.18362	0.12962	0.63674	51.98	60.42	43.29	60.58
	c	0.30429	0.18741	0.46918	45.31	38.46	23.81	43.40
	$c+3$	0.27051	0.19779	0.85028	47.51	45.23	29.23	48.42
AZP_RTABU	$c-3$	0.27233	0.17007	0.78498	44.19	44.95	28.99	46.86
	c	0.30857	0.20849	0.81854	43.09	43.67	27.93	45.59
	$c+3$	0.30699	0.21135	0.86227	43.84	35.09	21.28	40.60
ARISEL	$c-3$	0.29914	0.19268	0.81843	43.80	43.72	27.97	45.86
	c	0.28959	0.18852	0.43265	45.82	43.18	27.53	46.35
	$c+3$	0.30599	0.22116	0.88723	41.76	33.93	20.43	38.56
AMOEBA	*None*	0.53662	0.56779	0.66286	42.83	33.89	20.40	39.23
MAXP	10	0.16617	0.05139	0.60488	36.08	3.69	1.88	13.04
	100	0.12539	0.03611	0.68845	39.68	20.78	11.59	29.41
BPM	70%	0.18454	0.05587	0.67587	65.02	78.80	65.02	80.63
	80%				64.61	78.47	64.56	80.24
	90%				46.72	58.23	41.08	58.24
BDCD	70%	0.09166	0.01986	0.51715	63.98	77.84	63.72	79.44
	80%				36.77	5.67	2.92	16.60
	90%				35.02	0.12	0.06	2.44
				Distinct parameters				
SPPM	70%	0.05172	0.00532	0.05553	65.02	78.80	65.02	80.63
	80%				71.80	81.23	68.39	81.97
	90%				**82.34**	**84.77**	**73.57**	**85.40**
SKATER	$c-3$	0.17415	0.14741	0.19465	58.07	72.63	57.02	73.47
	c	0.21371	0.17941	0.24005	53.24	67.56	51.02	67.89
	$c+3$	0.24092	0.24516	0.27006	49.64	63.36	46.37	63.45
AZP	$c-3$	0.23537	0.14779	0.26733	43.52	53.24	36.28	53.39
	c	0.24854	0.17059	0.28081	42.97	38.19	23.60	41.85
	$c+3$	0.18657	0.19945	0.20594	48.30	51.36	34.56	52.69
AZP_SA	$c-3$	0.19384	0.11809	0.21444	45.09	38.37	23.74	43.20
	c	0.14145	0.11201	0.15493	48.34	59.72	42.57	59.72
	$c+3$	0.22061	0.22617	0.24595	39.32	29.60	17.37	34.37
AZP_TABU	$c-3$	0.20738	0.16752	0.22732	46.57	48.66	32.16	50.25
	c	0.21042	0.21353	0.22851	54.15	57.56	40.41	58.79
	$c+3$	0.30981	0.25766	0.34688	40.37	30.94	18.30	35.87
AZP_RTABU	$c-3$	0.24173	0.11225	0.26920	49.53	57.09	39.95	57.41
	c	0.26007	0.22601	0.29338	41.96	40.90	25.71	43.23
	$c+3$	0.28364	0.24032	0.32242	42.29	51.31	34.51	51.56
ARISEL	$c-3$	0.26916	0.18284	0.30273	42.89	46.43	30.23	47.59
	c	0.26754	0.19789	0.30433	40.72	48.16	31.72	48.62
	$c+3$	0.30933	0.26337	0.34726	42.22	37.86	23.35	41.28
AMOEBA	*None*	0.50147	0.55240	0.56333	42.28	34.78	21.05	39.38
MAXP	10	0.14022	0.03872	0.15519	35.97	3.46	1.76	12.44
	100	0.08065	0.01122	0.08758	40.55	23.89	13.57	31.97
BPM	70%	0.22147	0.06551	0.24927	52.61	58.85	41.70	59.35
	80%				52.61	58.85	41.70	59.35
	90%				41.58	43.90	28.13	45.33
BDCD	70%	**0.04487**	**0.00426**	**0.04738**	65.12	78.77	64.98	80.55
	80%				40.61	16.44	8.95	29.42
	90%				35.04	0.17	0.09	2.88

6. Case Studies

The analysis described in this section illustrates the usefulness of our method in three relevant applications. The first one is a purely spatial case with Gaussian data and aims at the regionalization of the Brazilian municipalities according to their Human Development Index (HDI) in 2010. The second application is also purely spatial but assumes a Poisson distribution for death counts. It aims at the regionalization of the Southern Brazilian municipalities according to lung and bladder cancer mortality. We restrict the analysis to the Brazilian region where the mortality data is not affected by under reporting. The third application is a spatio-temporal situation with Gaussian data. We analysed the HDI as in the first application but now we describe its evolution from 1991 to 2010. In the three analysis, for the MCMC, we ran chains of size 10000, skipping the first 1000 samples as the *burn-in* period and take a lag of 10 to avoid correlation.

6.1. HDI Data: A Spatial Regionalization with Gaussian SPPM

The HDI is an index combining life expectancy, education and income measures developed by the United Nations Development Programme (UNDP) and it is often used to rank countries. Together with UNDP, the *Instituto de Pesquisa Econômica e Aplicada* (IPEA) and *Fundação João Pinheiro* developed a version of the HDI incorporating additional variables extracted from the demographic census to evaluate the Brazilian municipalities.

The data set is composed of the HDI of 5564 municipalities of Brazil (see Figure 7) in 2010. which is assumed to be normally distributed with mean and variance changing over the space. We consider the neighborhood structure computed through the geographic adjacency. As prior distribution for the cluster parameters we assume a conjugate Normal-Gamma distribution such that $\mu_{\mathcal{G}_k} \mid \tau_{\mathcal{G}_k} \sim N\left(0.65, (0.04\tau_{\mathcal{G}_k})^{-1}\right)$ and $\tau_{\mathcal{G}_k} \sim \text{Gamma}\,(100, 1)$. This prior distribution for the precision concentrates its probability mass around 100, which yields a standard deviation of about 0.1 for the observations of the clusters. Consequently, the cluster means are most probably around 0.65, with a deviation of 0.5. This spans most of the range the HDI can take, which is from 0.0 to 1.0. *A priori*, we assume that each edge is removed from the tree with probability $\rho \sim \text{Beta}(2, 7)$. As a result, the expected number of clusters in the map is around 1236. This is likely to be much larger than one would expect. However, we selected these values for two reasons. First, we want to allow a possible large variability in the parameters to be expressed by the prior. Second, we want to verify if the data provide enough evidence to shrink the distribution of the number of clusters.

The posterior distribution for the number c of clusters has mode equal to 12 and varies between 10 and 16, the 5% and 95% quantiles, respectively. The posterior probability of edge removal ρ has mean 0.0025 and median 0.0022.

In Figure 6 we show a random sample of partitions generated by our algorithm. The main difference between samples is on the frontier between the clusters. As expected, the boundaries are hard to be established because, for this type of data, we do not expect a clearly defined and sharp transition in the data.

To summarize the posterior partition distribution, we present Figure 7b. It is constructed by deleting the boundary between neighboring municipalities that belong to the same cluster in at least 80% of the sampled partitions. Therefore, the boundaries that remain indicates

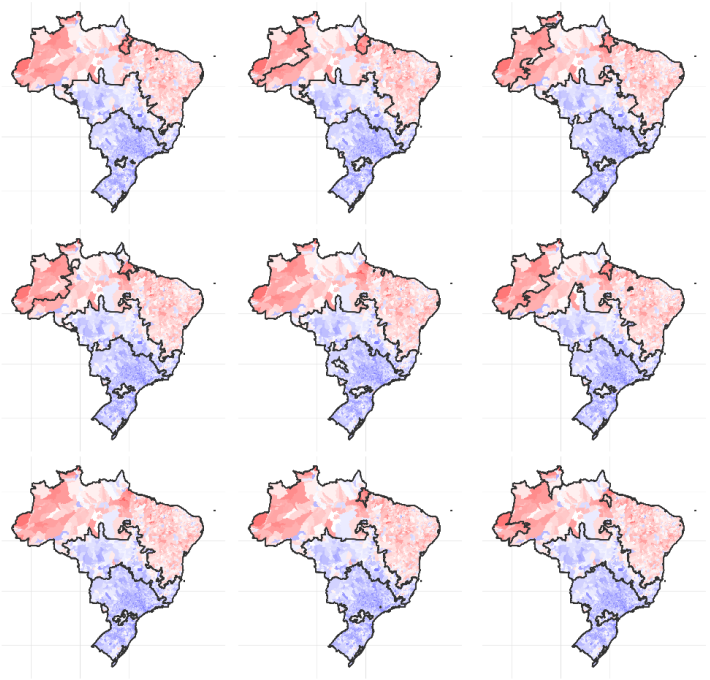
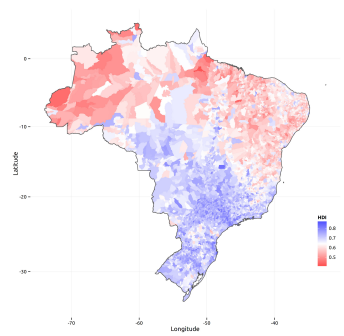


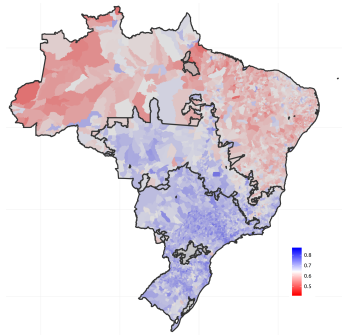
Figure 6: Some of the sampled partitions (HDI data)

areas that cannot be sharply allocated together into a cluster with its neighbor. It does not mean that the area necessarily stays isolated from their neighbors most of the time. It means that it is often allocated to clusters with different area components. Lines clearly splitting large portions of the map indicate sharp transition zones. Another posterior partition summary is given in Figure 7c . Each area is colored according to the size of cluster to which it belongs. Darker colors indicate that the area belong to a small cluster.

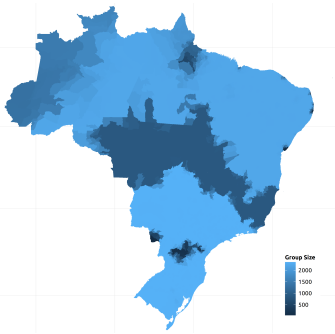
These figures show the presence of three large regions in the country. Additionally, we can see a number of small groups in the frontier and some tiny clusters located on the Northeast coast, hard to visualize on the figure. We also have two small clusters isolated in the middle of the big clusters, both with average HDI. The Southern one is surrounded by better HDI areas while the Northern one is relatively better than its neighbors. Although the main groups are well defined, the separation between them is not, causing a certain level of noise. In fact, this discloses the natural characteristic of this kind of data where there is a transition between two distinct groups but frontiers are not well defined. Another result is related to the tiny clusters found on the Northeast coast. These areas are frequently assigned to small clusters, with size within 9000 squared kilometers in at least 90% of the



(a) Map of Brazilian municipalities and their HDI data



(b) Summary of the sampled partitions for Brazilian municipalities (HDI data)



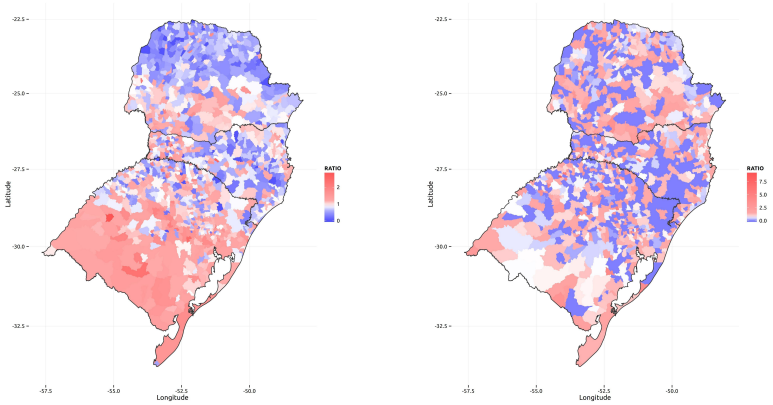
(c) Average group size for each municipality in the sampled partitions

Figure 7: Posterior regionalization of Brazilian municipalities according to their HDI

sampled partitions. Such areas are the capital of states in the northeast region and their neighboring towns. The HDI in the northeast region is generally very low, as can be seen in Figure 7a, but the capital cities and their surroundings are more developed and have a stronger economy than other regions on the State.

6.2. Lung and Bladder Cancer Data: A Spatial Regionalization with Poisson SPPM

To illustrate the use of the Poisson SPPM we consider the number of deaths by bladder and lung cancer in the south region of Brazil. These two types of cancers were selected due to their different incidence rate. Bladder cancer is almost an order of magnitude rarer than lung cancer. Thus, the incidence rate for bladder cancer is more affected by small variations.



(a) Lung cancer (b) Bladder cancer

Figure 8: Ratio between the actual and expected number of deaths by lung and bladder cancer in Brazilian south region

This scenario is a good example where the benefits of using a stochastic method instead of traditional approaches may be more easily perceived.

For both data sets, we obtained the total number of deaths by age group and gender, in each municipality, in the years 2008 - 2012. Data are available in the DATASUS website (`http://datasus.saude.gov.br/`). We also obtained demographic information of the same years, for the same age groups and gender, from IBGE. We summed over the years the number of deaths in each area to obtain the total number of events in the period generating one single count Y_i for each municipality. We assume that, conditional on a parameter θ_i, the random variable Y_i follows a Poisson distribution with mean $\mu_i = E_i\theta_i$. The values of E_i represent the expected number of deaths in area i if the risk was spatially constant in each age and sex class. That is, after obtaining the age-sex-specific mortality rates for the entire map, we applied them on each municipality respecting their demographic distribution by age and sex. We end up with the expected number of events in each municipality under this spatially homogeneous hypothesis. The θ_i parameters are called relative risks and they represent multiplicative factors with respect to the baseline E_i. In Figure 8 we show the ratio between the observed and the expected number of deaths by cancer.

We assume that, a priori, the cluster mortality rate has a Gamma distribution with parameters $a = 1.1$, and $b = 1.1$. This distribution concentrates its mass around 1.1, with a variance of 0.91, so the relative risk is concentrated in values mostly between 0 and 2. This seems reasonable as we expect the incidence rate to deviate from the expected value by a factor smaller than 2. We also assume that $\rho \sim \text{Beta}(5, 1000)$ and thus the expected number of cluster *a priori* is 6.9. The proposed model is compared to ARISEL and SKATER.

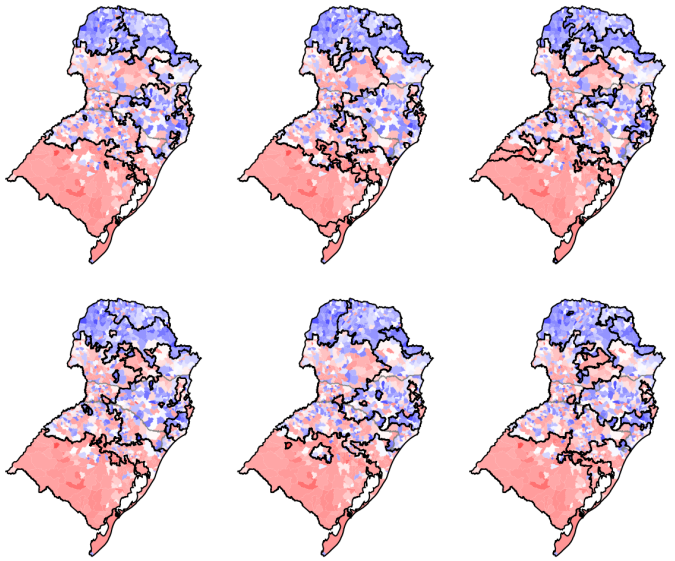


Figure 9: Some of the sampled partitions (lung cancer)

Inspired by our findings by applying the proposed model, for such approaches, we fixed the number the clusters equal to 4.

The posterior distribution for the number of clusters c has mode 29 (lung) and 13 (bladder) with 5% and 95% quantiles given by 18 and 34 (lung) and 7 and 22 (bladder). The edge removal parameter ρ has posterior mean and median around 0.013 and 0.008 for the lung and bladder examples, respectively.

In Figures 9 and 10 we show some of the partitions sampled by our algorithm for the lung and bladder cancer data sets, respectively.

As in the normal case, we summarize the posterior distribution of these partitions in Figure 11. Maps in (a) and (b) that are constructed taking into consideration the neighboring municipalities belonging to the same cluster in at least 85% of the sampled partitions. Maps in (c) and (d) represent the average size of the cluster to which each region belongs.

Our method is able to find the easily spotted clusters in the extreme North and South regions in Figure 8a for the lung cancer. However, it goes beyond that by also finding additional clusters that divide the central area into West and East. In the bladder cancer case, our model finds evidence in favor of the existence of clusters in the West and South regions. These findings are hardly identified by visual inspection of Figure 8b or by using some of the traditional approaches.

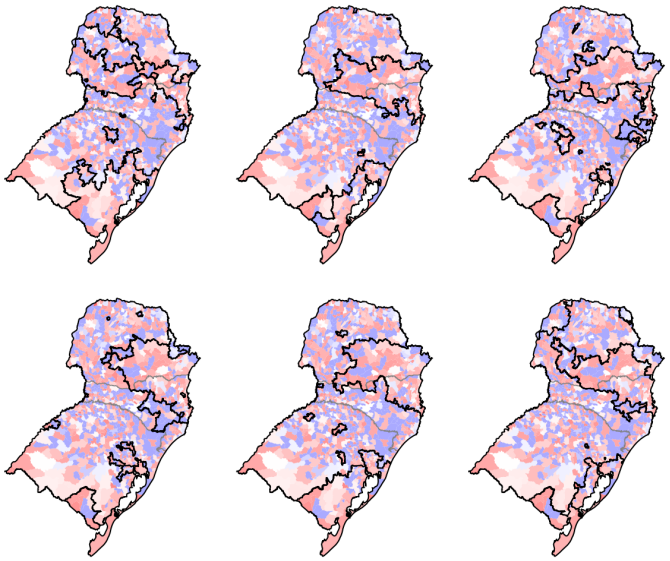
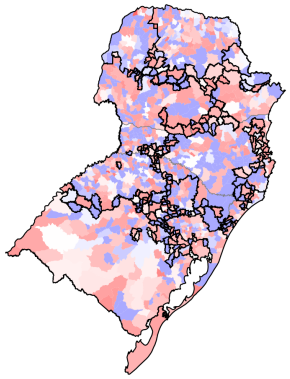


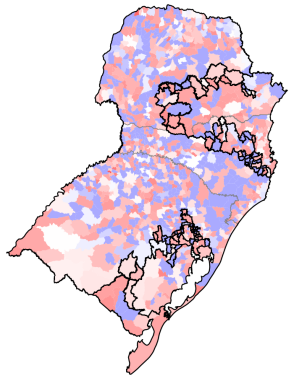
Figure 10: Some of the sampled partitions (bladder cancer)

As we can see in Figure 12, the *ARiSEL* and *SKATER* techniques result in regionalizations that capture only the most obvious visual aspects of the raw data maps. These deterministic methods were run with 4 groups. The *SKATER* method was still able to separate the top from the bottom of the lung cancer map, while *ARiSEL* detected only the southern boundary and with a more jagged line. For the bladder cancer, however, neither method was able to detect the region in the northeast of the map. They seemed to be more sensitive to local variations in the rate. This is the practical exemplification of what we expected. Given that these methods do not use a statistical model, they are more susceptible to this kind of problem where the population incidence rate is small.

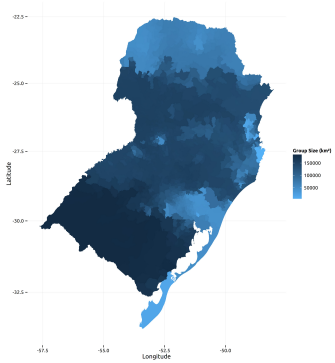
Lung cancer is linked to smoking and the rate increases going south, which could be related to the colder climate. This explains the clusters in the two extreme parts of the North-South gradient in Figure 11a. With the help of hindsight, we can see that the right portion of the middle region in this figure is where some of the large cities are located, and presents a lower rate than its western counterpart. The deterministic methods either separates out only the extreme south region and fail to identify the regions in the center. A more visible difference between the methods is in Figure 11b. SPPM identified a cluster in the northeast of the map, where Curitiba (capital of Paraná state) and the most populated cities of the state of Santa Catarina are located. All the other methods failed to find this



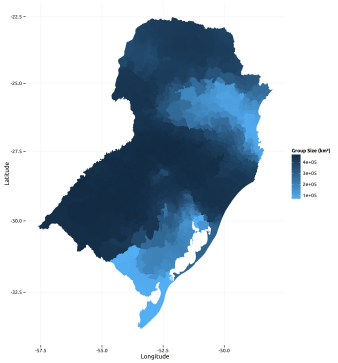
(a) Summary of the sampled partitions for each region, Lung cancer



(b) Summary of the sampled partitions for each region, Bladder cancer

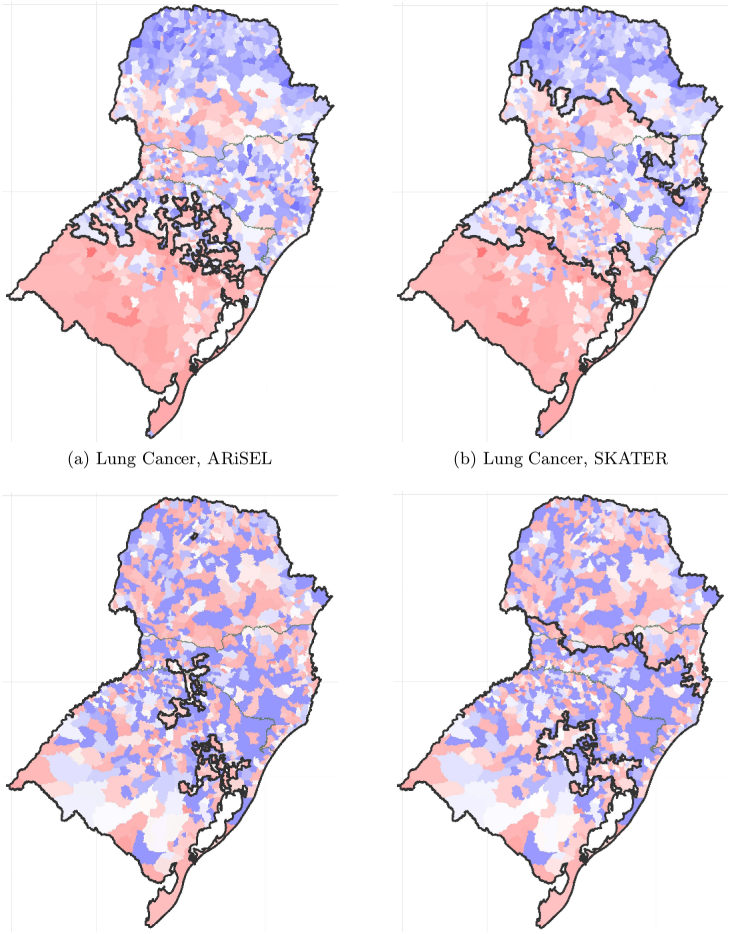


(c) Average group size for each region, Lung cancer



(d) Average group size for each region, Bladder cancer

Figure 11: Posterior regionalization of municipalities in Brazilian south region according to the number of deaths by lung and bladder cancer, proposed model



(a) Lung Cancer, ARiSEL (b) Lung Cancer, SKATER

(c) Bladder Cancer, ARiSEL (d) Bladder Cancer, SKATER

Figure 12: Regionalization of municipalities in the Brazilian south region according to the number of deaths by lung and bladder cancer, ARiSEL and SKATER methods.

cluster and instead identified noisy small regions, which is a demonstration of how these methods are sensitive to the variations of the data.

6.3. HDI Data: A Space-Time Regionalization

In this section, we provide a short example of our space-time method using the state-level HDI for the 26 Brazilian states and the Federal district. We have data for three different time moments, the 1991, 2000, and 2010 Census years, which implies in 81 graph nodes. We used the state-level graph to simplify the analysis and the interpretation of the results. We considered the same prior choices as in Section 6.1: $\mu_{\mathcal{G}_k} \mid \tau_{\mathcal{G}_k} \sim N\left(0.65, (0.04\tau_{\mathcal{G}_k})^{-1}\right)$, $\tau_{\mathcal{G}_k} \sim \text{Gamma}\,(100, 1)$, and $\rho \sim \text{Beta}(2, 7)$.

Figure 13 shows the HDI in each year in the top row and the summary of our posterior distribution in the bottom row. We delete the boundary between neighboring states that belong to the same cluster in at least 70% of the sampled partitions. It is clear the improvement of the Brazilian socio-economic condition over the three decades. In the first decade, the small Federal District was isolated from the other regions presenting the highest HDI level. In the second decade, this cluster is enlarged including the states in southern part of the map. Such regions, in fact, presented an improvement in their socio-economic conditions if compared with the first decade under analysis, getting closer of the Federal District. Another small cluster was identified separating Amapá state, located in the North, from the other regions. After three decades, all Brazilian states are assembled in a single cluster.

7. Conclusion

In this work we dealt with the problem of regionalization, an important type of clustering problem which arises in many areas. We proposed a new product partition model that accounts for spatial and spatio-temporal clustering. The innovative aspect of this paper is the use of random spanning trees as a tool to reduce the search space of partitions. This random spanning tree trick allowed us to represent the random partitions as fixed dimensional vector with binary elements. We presented a posterior sampling algorithm for the proposed model that keeps fixed the dimension of the parametric space and resorts to a regular Gibbs sampling procedure.

We evaluated of our method using simulated and real data. In the simulated study, we compared our approach to available implementations of non-stochastic optimization methods and Bayesian proposals and showed how our results were consistently superior. This was particularly noticed in low rate Poisson generated data. We applied our technique to perform the regionalization of Brazilian municipalities based on the human development index and on bladder and lung cancer mortality data. We discussed how the results we obtained were suitable for the domain subjects and how the use of our model obtained better and more meaningful results.

The model proposed has some limitations. The temporal component is not dealt with typical time series techniques and this prevents us to make inference in a prospective way. More importantly, the prior distribution on the set of spanning trees is an artificial tool to make inference on the partitions and these spanning trees do not have an obvious interpretation making difficult a prior distribution elicitation for $\mathcal{T}$. However, the spanning tree plays

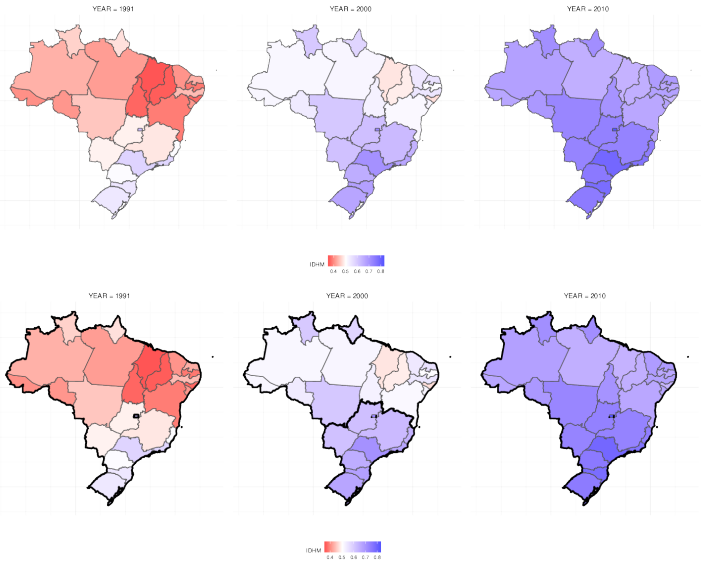


Figure 13: Map of Brazilian states and their HDI data (first row) and the evolution of the posterior summaries for cut point 0.7 (second row), over the years 1991, 2000, and 2010.

an important role in the prior elicitation for the partition. Coupled with the simplification of the computational search for a good partition by restricting this search to n edges, it allowed us to write the prior cohesion as functions of ρ, the prior probability of removing edges from the tree.

In conclusion, we proposed a stochastic model for the problem of space-time regionalization that captures much of the prior reasoning one could have for its formation. We introduced the use of spanning trees to provide an effective sampling algorithm. Our model is flexible enough to accommodate different types of data and provides good results.

Acknowledgments

The authors would like to thank CNPq, CAPES and FAPEMIG, three Brazilian funding agencies, for partial support. We also thank the comments provided by Pedro O. S. Vaz de Melo and Fábio Cozman on a first version of this paper and the comments provided by the editor and reviewers, which contributed to improve the paper. We also want to thank Vinícius Fernandes dos Santos for suggesting the counter-example illustrated in the left hand-side of Figure 4. The research for this project was conducted while the first author was a student at the Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, where he obtained his Master of Science degree in Computer Science.

References

Jared Aldstadt and Arthur Getis. Using amoeba to create a spatial weights matrix and identify spatial clusters. *Geographical Analysis*, 38(4):327–343, 2006.

Craig Anderson, Duncan Lee, and Nema Dean. Identifying clusters in Bayesian disease mapping. *Biostatistics*, 15(3):457–469, 2014.

Renato M. Assunção, Marcos C. Neves, Gilberto Câmara, and Corina da Costa Freitas. Efficient regionalization techniques for socio-economic geographical units using minimum spanning trees. *International Journal of Geographical Information Science*, 20(7):797–811, 2006.

Fernando Bação, Victor Lobo, and Marco Painho. Geo-self-organizing map (geo-som) for building and exploring homogeneous regions. In *Geographic Information Science*, pages 22–37. Springer, 2004.

Fernando Bação, Victor Lobo, and Marco Painho. Applying genetic algorithms to zone design. *Soft Computing*, 9(5):341–348, 2005.

Sudipto Banerjee and Alan E. Gelfand. Bayesian wombling: Curvilinear gradient assessment under spatial process models. *Journal of the American Statistical Association*, 101(476): 1487–1501, 2006.

Daniel Barry and John A. Hartigan. A Bayesian analysis for change point problems. *Journal of the American Statistical Association*, 88(421):309–319, 1993.

Iacopo Bernetti, Christian Ciampi, and Sandro Sacchelli. Minimizing carbon footprint of biomass energy supply chain in the province of florence. *Italian Journal of Forest and Mountain Environments*, 66(4):321–329, 2011.

Luke Bornn and Francois Caron. Bayesian clustering in decomposable graphs. *Bayesian Analysis*, 6(4):829–846, 12 2011. doi: 10.1214/11-BA630.

Andrei Broder. Generating random spanning trees. In *30th Annual Symposium on Foundations of Computer Science, 1989*, pages 442–447. IEEE, Oct 1989. doi: 10.1109/SFCS.1989.63516.

Samantha Cockings and David Martin. Zone design for environment and health studies using pre-aggregated data. *Social Science & Medicine*, 60(12):2729 – 2742, 2005. doi: http://dx.doi.org/10.1016/j.socscimed.2004.11.005.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009. ISBN 0262033844, 9780262033848.

David G.T. Denison and Christofer C. Holmes. Bayesian partitioning for estimating disease risk. *Biometrics*, 57(1):143–149, 2001.

Juan C. Duque and Richard L. Church. A new heuristic model for designing analytical regions. In *North American Meeting of the International Regional Science Association, Seattle*, 2004.

Juan C. Duque, Boris Dev, Alejandro Betancourt, and Jose L. Franco. *ClusterPy: Library of spatially constrained clustering algorithms, Version 0.9.9*. RiSE-group (Research in Spatial Economics). EAFIT University., Colombia, 2011. URL `http://www.rise-group.org`.

Juan C. Duque, Luc Anselin, and Sergio J. Rey. The max-p-regions problem. *Journal of Regional Science*, 52(3):397–419, 2012.

Ronald E. Gangnon and Murray K. Clayton. Bayesian detection and modeling of spatial disease clustering. *Biometrics*, 56(3):922–935, 2000.

Peter J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

Peter J. Green and Alun Thomas. Sampling decomposable graphs using a Markov chain on junction trees. *Biometrika*, 100(1):91–110, 2013. doi: 10.1093/biomet/ass052.

Diansheng Guo. Regionalization with dynamically constrained agglomerative clustering and partitioning (REDCAP). *International Journal of Geographical Information Science*, 22 (7):801–823, 2008.

John A. Hartigan. Partition models. *Communications in Statistics-Theory and Methods*, 19 (8):2745–2756, 1990.

Avril Hegarty and Daniel Barry. Bayesian disease mapping using product partition models. *Statistics in Medicine*, 27(19):3868–3893, 2008.

Leonhard Knorr-Held and Günter Raßer. Bayesian detection of clusters and discontinuities in disease maps. *Biometrics*, 56(1):13–21, 2000.

Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

Haolan Lu and Bradley P. Carlin. Bayesian areal wombling for geographical boundary analysis. *Geographical Analysis*, 37(3):265–285, 2005.

David Madigan and Adrian E. Raftery. Model selection and accounting for model uncertainty in graphical models using Occam's window. *Journal of the American Statistical Association*, 89(428):1535–1546, 1994.

David Martin. Optimizing census geography: the separation of collection and output geographies. *International Journal of Geographical Information Science*, 12(7):673–685, 1998.

Daniel W. McKenney, John H. Pedlar, Kevin Lawrence, Kathy Campbell, and Michael F. Hutchinson. Beyond traditional hardiness zones: Using climate envelopes to map plant range limits. *BioScience*, 57(11):929–937, 2007. doi: 10.1641/B571105.

Jeremy Mennis and Philip Harris. Spatial contagion of male juvenile drug offending across socioeconomically homogeneous neighborhoods. In *Crime Modeling and Mapping Using Geospatial Technologies*, pages 227–248. Springer, 2013.

Stan Openshaw. A geographical solution to scale and aggregation problems in region-building, partitioning and spatial modelling. *Transactions of the Institute of British Geographers*, pages 459–472, 1977.

Stan Openshaw and Liang Rao. Algorithms for reengineering 1991 census geography. *Environment and planning A*, 27(3):425–446, 1995.

Garritt L. Page and Fernando A. Quintana. Spatial product partition models. *Bayesian Analysis*, 11(1):265–298, 03 2016. doi: 10.1214/15-BA971.

Ilka A. Reis, Gilberto Câmara, Renato Assunção, and Antônio M. V. Monteiro. Data-aware clustering for geosensor networks data collection. In *Anais XIII Simpósio Brasileiro de Sensoriamento Remoto*, pages 6059–6066, 2007.

Patricia B. Ribeiro, Roseli A.F. Romero, Patrícia R. Oliveira, Homero Schiabel, and Luciana B. Verçosa. Automatic segmentation of breast masses using enhanced ica mixture model. *Neurocomputing*, 120:61–71, 2013.

Sylvia Richardson and Peter J. Green. On Bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 59(4):731–792, 1997.

Thomas C. Ricketts. *Using geographic methods to understand health issues*. Agency for Health Care Policy and Research, Dept. of Health and Human Services, US Public Health Service, 1997.

Anne Ruas. Map generalization. In *Encyclopedia of GIS*, pages 631–632. Springer, 2008.

Roger Sayre, Jack Dangermond, Charlie Frye, Randy Vaughan, Peter Aniello, Sean Breyer, Douglas Cribbs, Dabney Hopkins, Richard Nauman, William Derrenbacher, et al. A new map of global ecological land units—an ecophysiographic stratification approach. *Washington, DC: Association of American Geographers*, 2014.

Leonardo V. Teixeira, Renato M. Assunção, and Rosangela H. Loschi. A generative spatial clustering model for random data through spanning trees. In *2015 IEEE International Conference on Data Mining*, pages 997–1002, Nov 2015. doi: 10.1109/ICDM.2015.106.

Jonathan Wakefield and Albert Kim. A Bayesian model for cluster detection. *Biostatistics*, 14(4):752–765, 2013.

David B. Wilson. Generating random spanning trees more quickly than the cover time. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 296–303, New York, NY, USA, 1996. ACM. ISBN 0-89791-785-5. doi: 10.1145/237814.237880.

Steve Wise, Robert Haining, and Jingsheng Ma. Regionalisation tools for the exploratory spatial analysis of health data. In Manfred M. Fischer and Arthur Getis, editors, *Recent Developments in Spatial Analysis*, Advances in Spatial Science, pages 83–100. Springer Berlin Heidelberg, 1997.

Ying Zhang, Semu Moges, and Paul Block. Optimal cluster analysis for objective regionalization of seasonal precipitation in regions of high spatial-temporal variability: Application to Western Ethiopia. *Journal of Climate*, 29(10):3697–3717, 2016.

Journal of Machine Learning Research 20 (2019) 1-62 Submitted 12/16; Revised 2/19; Published 2/19

Streaming Principal Component Analysis From Incomplete Data

Armin Eftekhari ARMIN.EFTEKHARI@EPFL.CH
Institute of Electrical Engineering
École Polytechnique Fédérale de Lausanne
Lausanne, VD 1015, Switzerland

Gregory Ongie GONGIE@UCHICAGO.EDU
Department of Statistics
University of Chicago
Chicago, IL 60637, USA

Laura Balzano GIRASOLE@UMICH.EDU
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, MI 48109, USA

Michael B. Wakin MWAKIN@MINES.EDU
Department of Electrical Engineering
Colorado School of Mines
Golden, CO 80401, USA

Editor: Tong Zhang

Abstract

Linear subspace models are pervasive in computational sciences and particularly used for large datasets which are often incomplete due to privacy issues or sampling constraints. Therefore, a critical problem is developing an efficient algorithm for detecting low-dimensional linear structure from incomplete data efficiently, in terms of both computational complexity and storage.

In this paper we propose a streaming subspace estimation algorithm called Subspace Navigation via Interpolation from Partial Entries (SNIPE) that efficiently processes blocks of incomplete data to estimate the underlying subspace model. In every iteration, SNIPE finds the subspace that best fits the new data block but remains close to the previous estimate. We show that SNIPE is a streaming solver for the underlying nonconvex matrix completion problem, that it converges globally to a stationary point of this program regardless of initialization, and that the convergence is locally linear with high probability. We also find that SNIPE shows state-of-the-art performance in our numerical simulations.

Keywords: Principal component analysis, Subspace identification, Matrix completion, Streaming algorithms, Nonconvex optimization, Global convergence

1. Introduction

Linear models are the backbone of computational science, and Principal Component Analysis (PCA) in particular is an indispensable tool for detecting linear structure in collected data (van Overschee and de Moor, 2012; Ardekani et al., 1999; Krim and Viberg, 1996; Tong and Perreau, 1998). Principal components of a dataset are used, for example, to perform linear dimensionality reduction, which is in turn at the heart of classification, regression and other learning tasks that often suffer from the "curse of dimensionality", where having a small number of training samples in relation to the number of features typically leads to overfitting (Hastie et al., 2013).

In this work, we are particularly interested in applying PCA to data that is presented sequentially to a user, with limited processing time available for each item. Moreover, due to hardware limitations, we assume the user can only store small amounts of data. Finally, we also consider the possibility that the incoming data is incomplete, either due to physical sampling constraints, or deliberately subsampled to facilitate faster processing times or to address privacy concerns.

As one example, consider monitoring network traffic over time, where acquiring complete network measurements at fine time-scales is impractical and subsampling is necessary (Lakhina et al., 2004; Gershenfeld et al., 2010). As another example, suppose we have a network of cheap, battery-powered sensors that must relay summary statistics of their measurements, say their principal components, to a central node on a daily basis. Each sensor cannot store or process all its daily measurements locally, nor does it have the power to relay all the raw data to the central node. Moreover, many measurements are not reliable and can be treated as missing. It is in this and similar contexts that we hope to develop a *streaming* algorithm for PCA from incomplete data.

More formally, we consider the following problem: Let $\mathcal{S}$ be an r-dimensional subspace with orthonormal basis $S \in \mathbb{R}^{n\times r}$. For an integer T, let the coefficient vectors $\{q_t\}_{t=1}^T \subset \mathbb{R}^r$ be independent copies of a random vector $q \in \mathbb{R}^r$ with bounded expectation, namely, $\mathbb{E}\|q\|_2 < \infty$. Consider the sequence of vectors $\{Sq_t\}_{t=1}^T \subset \mathcal{S}$ and set $s_t := Sq_t$ for short. At each time $t \in [1:T] := \{1, 2, \cdots, T\}$, we observe each entry of s_t independently with a probability of p and collect the observed entries in $y_t \in \mathbb{R}^n$. Formally, we let $\omega_t \subseteq [1:n]$ be the random index set over which s_t is observed and write this measurement process as $y_t = P_{\omega_t} \cdot s_t$, where $P_{\omega_t} \in \mathbb{R}^{n\times n}$ is the projection onto the coordinate set ω_t, namely, it equals one on its diagonal entries corresponding to the index set ω_t and is zero elsewhere.

©2019 Armin Eftekhari, Gregory Ongie, Laura Balzano, Michael B. Wakin.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/16-627.html.



Figure 1: The sequence of generic vectors $\{s_t\}_{t=1}^T$ drawn from an unknown r-dimensional subspace $\mathcal{S}$ in panel (a) is only partially observed on random index sets $\{\omega_t\}_{t=1}^T$. That is, we only have access to incomplete data vectors $\{y_t\}_{t=1}^T$ in panel (b), where the white entries are missing. Our objective is to estimate the subspace $\mathcal{S}$ from the incomplete data vectors, when limited storage and processing resources are available. See Section 1 for the detailed setup.

Our objective in this paper is to design a streaming algorithm to identify the subspace $\mathcal{S}$ from the incomplete data $\{y_t\}_{t=1}^T$ supported on the index sets $\{\omega_t\}_{t=1}^T$. Put differently, our objective is to design a streaming algorithm to compute leading r principal components of the full (but hidden) data matrix $[s_1\, s_2 \cdots s_T]$ from the incomplete observations $[y_1\, y_2 \cdots y_T]$, see Figure 1. By the Eckart-Young-Mirsky Theorem, this task is equivalent to computing leading r left singular vectors of the full data matrix from its partial observations (Eckart and Young, 1936; Mirsky, 1966).

Assuming that $r = \dim(\mathcal{S})$ is known *a priori* (or estimated from data by other means), we present the SNIPE algorithm for this task in Section 2. SNIPE is designed based on the *principle of least-change* and, in every iteration, finds the subspace that best fits the new data block but remains close to the previous estimate. SNIPE requires $O(n)$ bits of memory and performs $O(n)$ flops in every iteration, which is optimal in its dependence on the ambient dimension n. As discussed in Section 3, SNIPE has a natural interpretation as a streaming algorithm for low-rank matrix completion (Davenport and Romberg, 2016).

Section 4 discusses the global and local convergence of SNIPE. In particular, the local convergence rate is linear near the true subspace, namely, the estimation error of SNIPE reduces by a factor of $1-cp$ in every iteration, for a certain factor c and with high probability. This local convergence guarantee for SNIPE is a key technical contribution of this paper which is absent in its close competitors, see Section 5.

Even though we limit ourselves to the "noise-free" case of $y_t = P_{\omega_t} s_t$ in this paper, SNIPE can also be applied (after minimal changes) when $y_t = P_{\omega_t}(s_t + n_t)$, where we might think of $n_t \in \mathbb{R}^n$ as measurement noise from a signal processing viewpoint. Alternatively from a statistical viewpoint, n_t represents the tail of the covariance matrix of the generative model, from which $\{s_t\}_t$ are drawn. Moreover, SNIPE can be easily adapted to the *dynamic* case where the underlying subspace $\mathcal{S} = \mathcal{S}(t)$ changes over time. We leave the convergence analysis of SNIPE under a noisy time-variant model to a future work. Similarly, entries of incoming vectors are observed uniformly at random in our theoretical analysis but SNIPE also applies to any incomplete data.

A review of prior art is presented in Section 5, and the performance of SNIPE and rival algorithms are examined numerically in Section 6, where we find that SNIPE shows the state-of-the-art performance. Technical proofs appear in Section 7 and in the appendices, with Appendix A (Toolbox) collecting some of the frequently-used mathematical tools. Finally, Appendix L offers an alternative initialization for SNIPE.

2. SNIPE

In this section, we present Subspace Navigation via Interpolation from Partial Entries (SNIPE), a streaming algorithm for subspace identification from incomplete data, received sequentially.

Let us first introduce some additional notation. Recall that we denote the incoming sequence of incomplete vectors by $\{y_t\}_{t=1}^T \subset \mathbb{R}^n$, which are supported on index sets $\{\omega_t\}_{t=1}^T \subseteq [1:n]$. For a block size $b \geq r$, we concatenate every b consecutive vectors into a data block, thereby partitioning the incoming data into $K = T/b$ non-overlapping blocks $\{Y_k\}_{k=1}^K$, where $Y_k \in \mathbb{R}^{n\times b}$ for every k. We assume for convenience that K is an integer. We also often take $b = O(r)$ to maximize the efficiency of SNIPE, as discussed below.

At a high level, SNIPE processes the first incomplete block Y_1 to produce an estimate $\widehat{\mathcal{S}}_1$ of the true subspace $\mathcal{S}$. This estimate is then iteratively updated after receiving each of the new incomplete blocks $\{Y_k\}_{k=2}^K$, thereby producing a sequence of estimates $\{\widehat{\mathcal{S}}_k\}_{k=2}^K$, see Figure 2. Every $\widehat{\mathcal{S}}_k$ is an r-dimensional subspace of $\mathbb{R}^n$ with orthonormal basis $\widehat{S}_k \in \mathbb{R}^{n\times r}$; the particular choice of orthonormal basis is inconsequential throughout the paper.

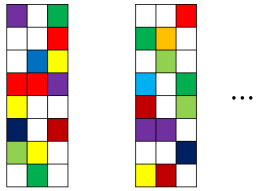


Figure 2: SNIPE concatenates every b incoming vectors into a block and iteratively updates its estimate of the true subspace $\mathcal{S}$ after receiving each new block. That is, SNIPE updates its estimate of $\mathcal{S}$, from $\widehat{\mathcal{S}}_{k-1}$ to $\widehat{\mathcal{S}}_k$, after receiving the incomplete data block $Y_k \in \mathbb{R}^{n\times b}$, see Section 2 for the details.

More concretely, SNIPE sets $\widehat{\mathcal{S}}_1$ to be the span of leading r left singular vectors of Y_1, namely, the left singular vectors corresponding to largest r singular values of Y_1, with ties broken arbitrarily. Then, at iteration $k \in [2:K]$ and given the previous estimate $\widehat{\mathcal{S}}_{k-1} = \mathrm{span}(\widehat{S}_{k-1})$, SNIPE processes the columns of the kth incomplete block Y_k one by one and forms the matrix

$$R_k = \left[\cdots \quad y_t + P_{\omega_t^C} \widehat{S}_{k-1} \left(\widehat{S}_{k-1}^* P_{\omega_t} \widehat{S}_{k-1} + \lambda I_r \right)^\dagger \widehat{S}_{k-1}^* y_t \quad \cdots \right] \in \mathbb{R}^{n\times b}, \qquad t \in [(k-1)b+1 : kb], \tag{1}$$

where $\dagger$ denotes the pseudo-inverse and $\lambda \geq 0$ is a parameter. Above, $P_{\omega_t^C} = I_n - P_{\omega_t} \in \mathbb{R}^{n\times n}$ projects a vector onto the complement of the index set ω_t. The motivation for the particular choice of R_k above will become clear in Section 3. SNIPE then updates its estimate by setting $\widehat{\mathcal{S}}_k$ to be the span of leading r left singular vectors of R_k. Algorithm 1 summarizes these steps. Note that Algorithm 1 rejects ill-conditioned updates in Step 3 for the convenience of analysis and that similar reject options have precedence in the literature (Balzano and Wright, 2015). We however found implementing this reject option to be unnecessary in numerical simulations.

Remark 1 [Computational complexity of SNIPE] We measure the *per-iteration* algorithmic complexity of SNIPE by calculating the average number of floating-point operations (flops) performed on an incoming vector. Every iteration of SNIPE involves finding leading r left singular vectors of an $n \times b$ matrix. Assuming that $b = O(r)$, this could be done with $O(nr^2)$ flops. At the kth iteration with $k \geq 2$, SNIPE also requires finding the pseudo-inverse of $P_{\omega_j} \widehat{S}_{k-1} \in \mathbb{R}^{n\times r}$ for each incoming vector which costs $O(nr^2)$ flops. Therefore the overall computational complexity of SNIPE is $O(nr^2)$ flops per vector. As further discussed in Section 5, this matches the complexity of other algorithms for streaming PCA even though here the received data is highly incomplete.

Remark 2 [Storage requirements of SNIPE] We measure the storage required by SNIPE by calculating the number of memory elements stored by SNIPE at any given instant. At the kth iteration, SNIPE must store the current estimate $\widehat{S}_{k-1} \in \mathbb{R}^{n\times r}$ (if available) and the new incomplete block $Y_k \in \mathbb{R}^{n\times b}$. Assuming that $b = O(r)$, this translates into $O(nr) + O(pnr) = O(nr)$ memory elements. SNIPE therefore requires $O(nr)$ bits of storage, which is optimal up to a constant factor.

Algorithm 1 SNIPE for streaming PCA from incomplete data

Input:

- Dimension r,
- Received data $\{y_t\}_{t=1}^T \subset \mathbb{R}^n$ supported on index sets $\{\omega_t\}_{t=1}^T \subseteq [1:n]$, presented sequentially in $K = T/b$ blocks of size $b \geq r$,
- Tuning parameter $\lambda \geq 0$,
- Reject thresholds $\sigma_{\min}, \tau \geq 0$.

Output:

- r-dimensional subspace $\widehat{\mathcal{S}}_K$.

Body:

- Form $Y_1 \in \mathbb{R}^{n\times b}$ by concatenating the first b received vectors $\{y_t\}_{t=1}^b$. Let $\widehat{\mathcal{S}}_1$, with orthonormal basis $\widehat{S}_1 \in \mathbb{R}^{n\times r}$, be the span of leading r left singular vectors of Y_1, namely, those corresponding to r largest singular values. Ties are broken arbitrarily.
- For $k \in [2:K]$, repeat:
  1. Set $R_k \leftarrow \{\}$.
  2. For $t \in [(k-1)b+1 : kb]$, repeat
     - Set
     $$R_k \leftarrow \begin{bmatrix} R_k & y_t + P_{\omega_t^C}\widehat{S}_{k-1}\left(\widehat{S}_{k-1}^* P_{\omega_t}\widehat{S}_{k-1} + \lambda I_r\right)^\dagger \widehat{S}_{k-1}^* y_t \end{bmatrix},$$
     where $P_{\omega_t} \in \mathbb{R}^{n\times n}$ equals one on its diagonal entries corresponding to the index set ω_t, and is zero elsewhere. Likewise, $P_{\omega_t^C}$ projects a vector onto the complement of the index set ω_t.
  3. If $\sigma_r(R_k) < \sigma_{\min}$ or $\sigma_r(R_k) \leq (1+\tau)\cdot\sigma_{r+1}(R_k)$, then set $\widehat{\mathcal{S}}_k \leftarrow \widehat{\mathcal{S}}_{k-1}$. Otherwise, let $\widehat{\mathcal{S}}_k$, with orthonormal basis $\widehat{S}_k \in \mathbb{R}^{n\times r}$, be the span of leading r left singular vectors of R_k. Ties are broken arbitrarily. Here, $\sigma_i(R_k)$ is the ith largest singular value of R_k.
- Return $\widehat{\mathcal{S}}_K$.

3. Interpretation of SNIPE

SNIPE has a natural interpretation as a streaming algorithm for low-rank matrix completion, which we now discuss. First let us enrich our notation. Recall the incomplete data blocks $\{Y_k\}_{k=1}^K \subset \mathbb{R}^{n\times b}$ and let the random index set $\Omega_k \subseteq [1:n]\times[1:b]$ be the support of Y_k for every k. We write that $Y_k = P_{\Omega_k}(S_k)$ for every k, where the complete (but hidden) data block $S_k \in \mathbb{R}^{n\times b}$ is formed by concatenating $\{s_t\}_{t=(k-1)b+1}^{kb}$. Here, $P_{\Omega_k}(S_k)$ retains only the entries of S_k on the index set Ω_k, setting the rest to zero. By design, $s_t = Sq_t$ for every t and we may therefore write that $S_k = S\cdot Q_k$ for the coefficient matrix $Q_k \in \mathbb{R}^{r\times b}$ formed by concatenating $\{q_t\}_{t=(k-1)b+1}^{kb}$. To summarize, $\{S_k\}_{k=1}^K$ is formed by partitioning $\{s_t\}_{t=1}^T$ into K blocks. Likewise, $\{Q_k, Y_k, \Omega_k\}_{k=1}^K$ are formed by partitioning $\{q_t, y_t, \omega_t\}_{t=1}^T$, respectively.

With this introduction, let us form $Y \in \mathbb{R}^{n\times T}$ by concatenating the incomplete blocks $\{Y_k\}_{k=1}^K \subset \mathbb{R}^{n\times b}$, supported on the index sets $\Omega \subseteq [1:n]\times[1:T]$. To find the true subspace $\mathcal{S}$, one might consider solving

$$\begin{cases} \min\limits_{X,\mathcal{U}} \ \|P_{\mathcal{U}^\perp}X\|_F^2 + \lambda\|P_{\Omega^C}(X)\|_F^2 \\ P_\Omega(X) = Y, \end{cases} \tag{2}$$

where the minimization is over a matrix $X \in \mathbb{R}^{n\times T}$ and r-dimensional subspace $\mathcal{U} \subset \mathbb{R}^n$. Above, $P_{\mathcal{U}^\perp} \in \mathbb{R}^{n\times n}$ is the orthogonal projection onto the orthogonal complement of subspace $\mathcal{U}$ and $P_\Omega(X)$ retains only the entries of X on the index set Ω, setting the rest to zero. Note that Program (2) encourages its solution(s) to be low-rank while matching the observations Y on the index set Ω. The term $\lambda\|P_{\Omega^C}(X)\|_F^2$ for $\lambda \geq 0$ is the *Tikhonov regularizer* that controls the energy of solution(s) on the complement of index set Ω.

With complete data, namely, when $\Omega = [1:n]\times[1:T]$, Program (2) reduces to PCA, as it returns $X = Y$ and searches for an r-dimensional subspace that captures most of the energy of Y. That is, Program (2) reduces to $\min_{\mathcal{U}} \|P_{\mathcal{U}^\perp}Y\|_F^2$ when $\Omega = [1:n]\times[1:T]$, solution of which is the span of leading r left singular vectors of Y in light of the Eckart-Young-Mirsky Theorem (Eckart and Young, 1936; Mirsky, 1966). In this sense then, Program (2) performs PCA from incomplete data. Note crucially that Program (2) is a nonconvex problem because the Grassmannian $\mathbb{G}(n,r)$, the set of all r-dimensional subspaces in $\mathbb{R}^n$, is a nonconvex set.[1] However, given a fixed subspace $\mathcal{U} \in \mathbb{G}(n,r)$, Program (2) reduces to the simple least-squares program

$$\begin{cases} \min\limits_{X} \ \|P_{\mathcal{U}^\perp}X\|_F^2 + \lambda\|P_{\Omega^C}(X)\|_F^2 \\ P_\Omega(X) = Y, \end{cases} \tag{3}$$

where the minimization is over $X \in \mathbb{R}^{n\times T}$. If in addition λ is positive, then Program (3) is strongly convex and has a unique minimizer. Given a fixed feasible $X \in \mathbb{R}^{n\times T}$, Program (2) has the same minimizers as

$$\min_{\mathcal{U}\in\mathbb{G}(n,r)} \|P_{\mathcal{U}^\perp}X\|_F^2. \tag{4}$$

1. The Grassmannian can be embedded in $\mathbb{R}^{n\times n}$ via the map that takes $\mathcal{U} \in \mathbb{G}(n,r)$ to the corresponding orthogonal projection $P_{\mathcal{U}} \in \mathbb{R}^{n\times n}$. The resulting submanifold of $\mathbb{R}^{n\times n}$ is a nonconvex set.

That is, for a fixed feasible X, Program (2) simply performs PCA on X. We might also view Program (2) from a matrix completion perspective. More specifically, let

$$\rho_r^2(X) = \sum_{i \geq r+1} \sigma_i^2(X), \tag{5}$$

be the *residual* of X, namely, the energy of its trailing singular values $\sigma_{r+1}(X) \geq \sigma_{r+2}(X) \geq \cdots$. Like the popular nuclear norm $\|X\|_* = \sum_{i \geq 1} \sigma_i(X)$ in (Davenport and Romberg, 2016), the residual $\rho_r(X)$ gauges the rank of X. In particular, $\rho_r(X) = 0$ if and only if $\text{rank}(X) \leq r$. Unlike the nuclear norm, however, the residual is still a nonconvex function of X. We now rewrite Program (2) as

$$\begin{aligned} \begin{cases} \min\limits_{X, \mathcal{U}} \ \|P_{\mathcal{U}^\perp} X\|_F^2 + \lambda \|P_{\Omega^C}(X)\|_F^2 \\ P_\Omega(X) = Y \end{cases} &= \begin{cases} \min\limits_{X} \ \min\limits_{\mathcal{U} \in \mathbb{G}(n,r)} \ \|P_{\mathcal{U}^\perp} X\|_F^2 + \lambda \|P_{\Omega^C}(X)\|_F^2 \\ P_\Omega(X) = Y \end{cases} \\ &= \begin{cases} \min\limits_{X} \ \rho_r^2(X) + \lambda \|P_{\Omega^C}(X)\|_F^2 \\ P_\Omega(X) = Y. \end{cases} \end{aligned} \tag{6}$$

That is, if we ignore the regularization term $\lambda \|P_{\Omega^C}(X)\|_F^2$, Program (2) searches for a matrix with the least residual, as a proxy for least rank, that matches the observations Y. In this sense then, Program (2) is a "relaxation" of the low-rank matrix completion problem. Several other formulations for the matrix completion problem are reviewed in (Davenport and Romberg, 2016; Eftekhari et al., 2018b,a). We can also rewrite Program (2) in terms of its data blocks by considering the equivalent program

$$\begin{cases} \min \ \sum_{k=1}^K \|P_{\mathcal{U}}^\perp X_k\|_F^2 + \lambda \|P_{\Omega_k^C}(X_k)\|_F^2 \\ P_{\Omega_k}(X_k) = Y_k \qquad k \in [1:K], \end{cases} \tag{7}$$

where the minimization is over matrices $\{X_k\}_{k=1}^K \subset \mathbb{R}^{n \times b}$ and subspace $\mathcal{U} \in \mathbb{G}(n,r)$. Let us additionally introduce a number of auxiliary variables into Program (7) by considering the equivalent program

$$\begin{cases} \min \ \sum_{k=1}^K \|P_{\mathcal{U}_k}^\perp X_k\|_F^2 + \lambda \|P_{\Omega_k^C}(X_k)\|_F^2 \\ P_{\Omega_k}(X_k) = Y_k \qquad k \in [1:K] \\ \mathcal{U}_1 = \mathcal{U}_2 = \cdots = \mathcal{U}_K, \end{cases} \tag{8}$$

where the minimization is over matrices $\{X_k\}_{k=1}^K \subset \mathbb{R}^{n \times b}$ and subspaces $\{\mathcal{U}_k\}_{k=1}^K \subset \mathbb{G}(n,r)$. Indeed, Programs (2,7,8) are all equivalent and all nonconvex. Now consider the following approximate solver for Program (8) that alternatively solves for matrices and subspaces:

- Setting $X_1 = Y_1$ in Program (8), we minimize $\|P_{\mathcal{U}_1}^\perp Y_1\|_F^2$ over $\mathcal{U}_1 \in \mathbb{G}(n,r)$ and, by the Eckart-Young-Mirsky Theorem, find a minimizer to be the span of leading r left singular vectors of Y_1, which coincides with $\widehat{\mathcal{S}}_1$ in SNIPE.
- For $k \in [2:K]$, repeat:
  - Setting $\mathcal{U}_k = \widehat{\mathcal{S}}_{k-1}$ in Program (8), we solve

$$\begin{cases} \min_{X_k} \ \|P_{\widehat{\mathcal{S}}_{k-1}^\perp} X_k\|_F^2 + \lambda \|P_{\Omega_k^C}(X_k)\|_F^2 \\ P_{\Omega_k}(X_k) = Y_k, \end{cases} \tag{9}$$

    over matrix $X_k \in \mathbb{R}^{n \times b}$. We verify in Appendix B that the minimizer of Program (9) coincides with R_k in SNIPE, see (1).
  - If $\sigma_r(R_k) < \sigma_{\min}$ or $\sigma_r(R_k) \leq (1+\tau)\sigma_{r+1}(R_k)$, then no update is made, namely, we set $\widehat{\mathcal{S}}_k = \widehat{\mathcal{S}}_{k-1}$. Otherwise, setting $X_k = R_k$ in Program (8), we solve $\min \|P_{\mathcal{U}_k}^\perp R_k\|_F^2$ over $\mathcal{U}_k \in \mathbb{G}(n,r)$ to find $\widehat{\mathcal{S}}_k$. That is, by the Eckart-Young-Mirsky Theorem again, $\widehat{\mathcal{S}}_k$ is the span of leading r left singular vectors of R_k. The output of this step matches $\widehat{\mathcal{S}}_k$ produced in SNIPE.

To summarize, following the above procedure produces $\{R_k\}_{k=1}^K$ and $\{\widehat{\mathcal{S}}_k\}_{k=1}^K$ in SNIPE. In other words, we might think of SNIPE as an approximate solver for Program (2), namely, SNIPE is a streaming algorithm for low-rank matrix completion. In fact, the output of SNIPE always converges to a stationary point of Program (2) in the sense described in Section 4.

Another insight about the choice of R_k in (1) is as follows. Let us set $\lambda = 0$ for simplicity. At the beginning of the kth iteration of SNIPE with $k \geq 2$, the available estimate of the true subspace is $\widehat{\mathcal{S}}_{k-1}$ with orthonormal basis $\widehat{S}_{k-1}$. Given a new incomplete vector $y \in \mathbb{R}^n$, supported on the index set $\omega \subseteq [1:n]$, $z = \widehat{S}_{k-1}(P_\omega \widehat{S}_{k-1})^\dagger y$ best approximates y in $\widehat{\mathcal{S}}_{k-1}$ in ℓ_2 sense. In order to agree with the measurements, we minimally adjust this to $y + P_{\omega^C} z$, where P_{ω^C} projects onto the complement of index set ω. This indeed matches the expression for the columns of R_k in SNIPE. We note that this type of least-change strategy has been successfully used in the development of quasi-Newton methods for optimization (Nocedal and Wright, 2006, Chapter 6).

4. Performance of SNIPE

To measure the performance of SNIPE—whose output is a subspace—we naturally use principal angles as an error metric. More specifically, recall that $\mathcal{S}$ and $\widehat{\mathcal{S}}_K$ denote the true subspace and the output of SNIPE, respectively. Then the ith largest singular value of $P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_K}$ equals $\sin(\theta_i(\mathcal{S}, \widehat{\mathcal{S}}_K))$, where

$$\theta_1(\mathcal{S}, \widehat{\mathcal{S}}_K) \geq \theta_2(\mathcal{S}, \widehat{\mathcal{S}}_K) \geq \cdots \geq \theta_r(\mathcal{S}, \widehat{\mathcal{S}}_K)$$

denote the principal angles between the two r-dimensional subspaces $\mathcal{S}, \widehat{\mathcal{S}}_K$ (Golub and Van Loan, 2013). The estimation error of SNIPE is then

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_K) := \sqrt{\frac{1}{r} \sum_{i=1}^r \sin^2\left(\theta_i(\mathcal{S}, \widehat{\mathcal{S}}_K)\right)} = \frac{\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_K}\|_F}{\sqrt{r}}, \tag{10}$$

which also naturally induces a metric topology on the Grassmannian $\mathbb{G}(n,r)$.[2] Note also that we will always reserve calligraphic letters for subspaces and capital letters for their orthonormal bases, for example subspace $\mathcal{S}$ and its orthonormal basis S.

Our first result loosely speaking states that a subsequence of SNIPE converges to a stationary point of the nonconvex Program (2) as T goes to infinity, see Section 7.1 for the proof.

Theorem 3 [Global convergence] *Consider an r-dimensional subspace $\mathcal{S}$ with orthonormal basis $S \in \mathbb{R}^{n\times r}$. For an integer T, let the coefficient vectors $\{q_t\}_{t=1}^T \subset \mathbb{R}^r$ be independent copies of a random vector $q \in \mathbb{R}^r$ with bounded expectation, namely, $\mathbb{E}\|q\|_2 < \infty$. For every $t \in [1:T]$, we observe each coordinate of $s_t = Sq_t \in \mathcal{S}$ independently with a probability of p and collect the observations in $y_t \in \mathbb{R}^n$, supported on a random index set $\omega_t \subseteq [1:n]$. Fix positive λ, block size $b \geq r$, positive reject thresholds $\sigma_{\min}, \tau$, and consider the output sequence of* SNIPE *in Algorithm 1, namely, $\{(R_k, \widehat{\mathcal{S}}_k)\}_k$. Also by partitioning $\{q_t, s_t, y_t, \omega_t\}_t$, form the coefficient blocks $\{Q_k\}_k \subset \mathbb{R}^{r\times b}$, data blocks $\{S_k\}_k \subset \mathbb{R}^{n\times b}$, and incomplete data blocks $\{Y_k\}_k \subset \mathbb{R}^{n\times b}$ supported on index sets $\{\Omega_k\}_k \subseteq [1:n]\times[1:b]$, as described in Section 3.*

For every integer l, there exists an integer k_l, for which the following asymptotic statement is almost surely true as $T \to \infty$. Consider the restriction of Program (2) to iteration k_l, namely,

$$\begin{cases} \min_{X,\mathcal{U}} \ \|P_{\mathcal{U}^\perp} X\|_F^2 + \lambda \|P_{\Omega_{k_l}^C}(X)\|_F^2 \\ P_{\Omega_{k_l}}(X) = Y_{k_l}, \end{cases} \tag{11}$$

where the minimization is over matrix $X \in \mathbb{R}^{n\times b}$ and r-dimensional subspace $\mathcal{U}$. Then there exists $R \in \mathbb{R}^{n\times b}$ and r-dimensional subspace $\widehat{\mathcal{S}}$ such that

- *$\widehat{S}$ is the span of leading r left singular vectors of R,*
- *$(R, \widehat{\mathcal{S}})$ is a stationary pair of Program (11), namely, it satisfies the first-order optimality conditions of Program (11),*
- *$\lim_{l\to\infty} \|R_{k_l} - R\|_F = 0$,*
- *$\lim_{l\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_{k_l}, \mathcal{S}) = 0$.*

Remark 4 [Discussion of Theorem 3] Theorem 3 roughly speaking states that there is a subsequence of SNIPE that converges to a stationary point of Program (2), which was the program designed in Section 3 for PCA from incomplete data or, from a different perspective, for low-rank matrix completion. Theorem 3 is however silent about the nature of this stationary point, whether it is a local or global minimizer/maximizer, or a saddle point. To some extent, this question is addressed below in Proposition 6.

More generally, we have been able to show that this stationary point is in fact rank-r. When the block size of SNIPE is sufficiently large, namely when $b = \Omega(n)$, we can further establish that the limit point of SNIPE is indeed a global minimizer of Program (2)

2. Another possible error metric is simply the largest principal angle $\theta_1(\mathcal{S}, \widehat{\mathcal{S}}_K)$. The two metrics are very closely related: $\theta_1(\mathcal{S}, \widehat{\mathcal{S}}_K)/\sqrt{r} \leq d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_K) \leq \theta_1(\mathcal{S}, \widehat{\mathcal{S}}_K)$. However, we find that $\theta_1(\mathcal{S}, \widehat{\mathcal{S}}_K)$ is not amenable for analysis of our problem, as opposed to $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_K)$.

and moreover SNIPE recovers the true subspace, namely, $\lim_{l\to\infty} d(\widehat{\mathcal{S}}_{k_l}, \mathcal{S}) = 0$, with high probability and under certain standard conditions on the *coherence* of the true subspace $\mathcal{S}$ and sampling probability p. We have not included these results here because SNIPE is intended as a streaming algorithm and we are therefore more interested in the setting where $b = O(r)$, see Remarks 1 and 2 about the implementation of SNIPE. It is not currently clear to us when SNIPE converges in general but, as suggested by Proposition 6 below, if SNIPE converges, it does indeed converge to the true subspace $\mathcal{S}$.

Remark 5 [Technical point about Theorem 3] Note that Theorem 3 is proved for positive (but possibly arbitrarily small) $\sigma_{\min}$ and τ. In particular, an update $(R_k, \widehat{\mathcal{S}}_k)$ is rejected in Algorithm 1 if

$$\frac{\sigma_r(R_k)}{\sigma_{r+1}(R_k)} \leq 1 + \tau, \tag{12}$$

for an (otherwise arbitrary) positive τ and whenever the ratio is well-defined. Here, $\sigma_i(R_k)$ is the ith largest singular value of R_k. This is merely a technical nuance, limited to the rth singular value gap, that helps prevent the the output subspace $\widehat{\mathcal{S}}_k$ from oscillating in the limit. Likewise, Theorem 3 does not address the case $\lambda = 0$ in Program (11), even though λ can be made arbitrarily small in Theorem 3. This is again for technical convenience and in fact the numerical simulations in Section 6 are all conducted with $\lambda = 0$.

Our second result establishes that, if SNIPE converges, then it converges to the true subspace $\mathcal{S}$, see Section 7.2 for the proof.

Proposition 6 [Convergence] *Consider the setup in the first paragraph of Theorem 3. Suppose that r independent copies of random coefficient vector $q \in \mathbb{R}^r$ almost surely form a basis for $\mathbb{R}^r$.*[3] *Suppose also that the output of* SNIPE *converges to an r-dimensional subspace $\widehat{\mathcal{S}}$, namely,*

$$\lim_{k\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_k, \widehat{\mathcal{S}}) = 0. \tag{13}$$

Then almost surely it must hold that $\widehat{\mathcal{S}} = \mathcal{S}$.

Remark 7 [Discussion of Proposition 6] Proposition 6 does not specify the conditions under which SNIPE converges. Indeed, if the sampling probability p is too small, namely, if very few of the entries of incoming vectors are observed, then SNIPE might not converge at all as the numerical evidence suggests, see also Remark 4. However, if SNIPE converges, then it converges to the true subspace $\mathcal{S}$. The local rate of convergence is specified below.

The concept of *coherence* is critical in specifying the local convergence rate, since we consider entrywise subsampling. The coherence of an r-dimensional subspace $\mathcal{S}$ with orthonormal basis $S \in \mathbb{R}^{n\times r}$ is defined as

$$\eta(\mathcal{S}) := \frac{n}{r} \max \|S[i,:]\|_2^2, \tag{14}$$

3. For example, this requirement is met if entries of q are independent Gaussian random variables with zero-mean and unit variance.

where $S[i,:]$ is the ith row of S. It is easy to verify that $\eta(\mathcal{S})$ is independent of the choice of orthonormal basis S and that

$$1 \le \eta(\mathcal{S}) \le \frac{n}{r}. \tag{15}$$

It is also common to say that $\mathcal{S}$ is *coherent* (*incoherent*) when $\eta(\mathcal{S})$ is large (small). Loosely speaking, when $\mathcal{S}$ is coherent, its orthonormal basis S is "spiky." An example is when $\mathcal{S}$ is the span of a column-subset of the identity matrix. In contrast, when $\mathcal{S}$ is incoherent, entries of S tend to be "diffuse." Not surprisingly, identifying a coherent subspace from subsampled data may require many more samples (Balzano and Wright, 2015; Mitliagkas et al., 2014; Chen, 2015).

We will also use $\lesssim$ and $\gtrsim$ below to suppress (most of) the universal constants. Moreover, throughout C represents a universal constant, the value of which is subject to change in every appearance.

Our next results specify the local convergence rate of SNIPE. Indeed, the convergence speed near the true subspace $\mathcal{S}$ is linear as detailed in Theorems 8 and 10, and proved in Section 7.3. In particular, Theorem 8 states that, when sufficiently small, the expected estimation error of SNIPE reduces by a factor of $1 - p/32$ in every iteration.

Theorem 8 [Locally linear convergence of SNIPE **in expectation]** *Consider the setup in the first paragraph of Theorem 3. Fix a positive tuning parameter α, iteration $k \in [2:K]$, and let $\mathfrak{E}_{k-1}$ be the event where*

$$\frac{1}{\sqrt{nb}} \gtrsim p \gtrsim \log n \log^2\left(p\sqrt{r} d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1})\right) \frac{\eta(\mathcal{S}) r}{n}, \tag{16}$$

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) \log\left(\frac{16}{p\sqrt{r} d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1})}\right) \lesssim \frac{p^{\frac{3}{2}}}{\sqrt{\log n}}, \tag{17}$$

$$\|Q_k\| \le \frac{\sigma_{\min}}{\sqrt{1 - p/4}}, \tag{18}$$

where $\sigma_{\min}$ is the reject threshold in SNIPE*. Let also $\mathcal{A}_k$ be the event where the kth iteration of* SNIPE *is not rejected (see Step 3 of Algorithm 1) and let $1_{\mathcal{A}_k}$ be the indicator for this event, taking one if the event happens and zero otherwise. Then it holds that*

$$\mathbb{E}\left[1_{\mathcal{A}_k} \cdot d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k) \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}\right] \le \left(1 - \frac{p}{32}\right) d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}). \tag{19}$$

Remark 9 [Discussion of Theorem 8] When the sampling probability p is large enough and SNIPE is near the true subspace $\mathcal{S}$, Theorem 8 states that the expected estimation error of SNIPE reduces by a factor of $1 - p/32$, if the iterate of SNIPE is not rejected. Note that: ① The lower bound on the sampling probability p in (16) matches the one in the low-rank matrix completion literature up to a logarithmic factor (Davenport and Romberg, 2016). Indeed, SNIPE can be interpreted as a streaming matrix completion algorithm as discussed in Section 3. The upper bound on p in (16) is merely for technical convenience and a tidier presentation in the most interesting regime for p. Indeed, since we often take $b = O(r) \ll n$, one might loosely read (16) as

$$\frac{1}{\sqrt{nr}} \gtrsim p \gtrsim \frac{\eta(\mathcal{S}) r}{n}, \tag{20}$$

in which the upper bound hardly poses a restriction even for moderately large data dimension n, as it forces $r = O(n^{\frac{1}{3}})$. ② Ignoring the logarithmic factors for simplicity, we may read (17) as $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) \lesssim p^{3/2}$, which "activates" (19). In other words, the basin of attraction of the true subspace $\mathcal{S}$ as a (possibly local) minimizer of the (nonconvex) Program (2) has a radius of $O(p^{3/2})$. ③ The indicator $1_{\mathcal{A}_k}$ in (19) removes the rejected iterates and similar conditions implicitly exist in the analysis of other streaming PCA algorithms (Balzano and Wright, 2015).

Note that Theorem 8 *cannot* tell us what the local convergence rate of SNIPE is, even in expectation. Indeed, the expected reduction in the estimation error of SNIPE, specified in (19), is not enough to activate (17) for the next iteration (namely, with k instead of $k-1$). That is, we cannot apply Theorem 8 iteratively and find the expected convergence rate of SNIPE. A key technical contribution of this paper is specifying the local behaviour of SNIPE below. With high probability, the estimation error does not *increase* by much in every iteration near the true subspace. However, only in some of these iterations does the error reduce. Overall, on a long enough interval, the estimation error of SNIPE near the true subspace indeed reduces substantially and with high probability as detailed in Theorem 10 and proved in Section 7.3. Performance guarantees for stochastic algorithms on long intervals is not uncommon, see for example (Cartis and Scheinberg, 2017).

Theorem 10 [Locally linear convergence of SNIPE**]** *Consider the setup in the first paragraph of Theorem 3. Suppose that the output $\widehat{\mathcal{S}}_{K_0}$ of* SNIPE *at iteration $K_0 \in [2:K]$ satisfies*

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0}) \lesssim \frac{e^{-\frac{Cp^3 nb}{\widetilde{\eta}}} p^{\frac{7}{2}} nb}{\widetilde{\eta} \log b \log\left(C\widetilde{\eta} n\right) \log^2(K - K_0)}, \tag{21}$$

and that

$$K - K_0 \gtrsim \frac{\widetilde{\eta} \log b \log(K - K_0)}{p^2 nb}. \tag{22}$$

Above, C is a universal constant that might change in every appearance. Then it holds that

$$\prod_{k=K_0+1}^{K} 1_{\mathcal{A}_k} \cdot d(\mathcal{S}, \widehat{\mathcal{S}}_K) \lesssim \left(1 - \frac{Cp^3 nb}{\widetilde{\eta} \log b \log(K - K_0)}\right)^{K-K_0} d(\mathcal{S}, \widehat{\mathcal{S}}_{K_0}), \tag{23}$$

except with a probability of at most

$$b^{-C\log(K-K_0)} + \sum_{k=K_0+1}^{K} \Pr\left[\|Q_k\| > \left(1 + \frac{Cp^3 nb}{\widetilde{\eta} \log b}\right) \sigma_{\min}\right] \tag{24}$$

and provided that

$$\frac{1}{\sqrt{nb}} \gtrsim p \gtrsim \log^2 b \log n \frac{\eta(\mathcal{S}) r}{n}. \tag{25}$$

Above, $\sigma_{\min}$ is the reject threshold of SNIPE *and*

$$\widetilde{\eta} = \max_{k \in [K_0:K]} \widetilde{\eta}_k, \tag{26}$$

$$\widetilde{\eta}_k := nb \cdot \frac{\|P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} S_k\|_\infty^2}{\|P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} S_k\|_F^2}. \tag{27}$$

Remark 11 [Discussion of Theorem 10] Loosely speaking, Theorem 10 states that, with high probability, the estimation error of SNIPE over $O(\widetilde{\eta} n)$ iterations reduces linearly (i.e., exponentially fast) when SNIPE is near the true subspace and the sampling probability p is large enough. Most of the remarks about Theorem 8 are also valid here. Let us also point out that the dependence on the coefficient matrix Q_k in (24) is mild but necessary. As an example, consider the case where the coefficient vectors $\{q_t\}_t$ are standard random Gaussian vectors so that the coefficient matrices $\{Q_k\}_k$ are standard random Gaussian matrices, namely, populated with independent zero-mean Gaussian random variables with unit variance. Then by taking

$$\sigma_{\min} = C\sqrt{b} / \left(1 + \frac{Cp^3 nb}{\widetilde{\eta} \log b}\right),$$

we find that

$$\Pr\left[\|Q_k\| > \left(1 + \frac{Cp^3 nb}{\widetilde{\eta} \log b}\right) \sigma_{\min}\right] = \Pr\left[\|Q_k\| \gtrsim \sqrt{b}\right] \leq e^{-Cb} \tag{28}$$

and consequently the failure probability in (24) becomes $b^{-C\log(K-K_0)} + (K-K_0)e^{-Cb}$, which can be made arbitrarily small by modestly increasing the block size b. For the reader's convenience, Appendix K collects the relevant spectral properties of a standard random Gaussian matrix. The dependence on $\|Q_k\|$ in Theorem 8 is *not* an artifact of our proof techniques. Indeed, when $\|Q_k\| \gg 1$, it is likely that certain directions in $\mathcal{S}$ are over represented which will skew the estimate of SNIPE in their favor.

Remark 12 [Coherence factor $\widetilde{\eta}$] A key quantity in Theorem 10 is the new "coherence" factor $\widetilde{\eta}$ which is absent in the expected behavior of SNIPE in Theorem 8. Somewhat similar to the coherence $\eta(\cdot)$ in (14), $\widetilde{\eta}_k$ measures how "spiky" the matrix $P_{\mathcal{S}_{k-1}^{\perp}} S_k \in \mathbb{R}^{n\times b}$ is. In fact, one may easily verify that

$$\widetilde{\eta}_k \leq \operatorname{rank}(P_{\mathcal{S}_{k-1}^{\perp}} S_k) \cdot \nu(P_{\mathcal{S}_{k-1}^{\perp}} S_k)^2 \cdot \eta(\operatorname{span}(P_{\mathcal{S}_{k-1}^{\perp}} S_k)) \cdot \eta(\operatorname{span}(S_k^* P_{\mathcal{S}_{k-1}^{\perp}})), \tag{29}$$

where $\nu(P_{\mathcal{S}_{k-1}^{\perp}} S_k)$ is the condition number of $P_{\mathcal{S}_{k-1}^{\perp}} S_k$, namely, the ratio of largest to smallest nonzero singular values. The number of iterations needed to see a reduction in estimation error in (22) and the convergence rate of SNIPE in (23) both prefer $\widetilde{\eta}$ to be small, namely, prefer that $\{P_{\mathcal{S}_{k-1}^{\perp}} S_k\}_k$ are all incoherent as measured by $\widetilde{\eta}_k$.

When SNIPE is close enough to true subspace $\mathcal{S}$ as required in (21), one would expect that iterates of SNIPE would be nearly as coherent as $\mathcal{S}$ itself in the sense that $\eta(\widehat{\mathcal{S}}_k) \approx \eta(\mathcal{S})$. This intuition is indeed correct and also utilized in our analysis. However, even when $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1})$ is small and $\eta(\widehat{\mathcal{S}}_{k-1}), \eta(\mathcal{S})$ are both small, $\widetilde{\eta}_k$ might be very large, namely, $P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} S_k$ might be a spiky matrix. Indeed, when $b = r$, $(\widehat{\mathcal{S}}_{k-1}^{\perp})^* S_k$ is approximately the (horizontal) tangent at $\widehat{\mathcal{S}}_{k-1}$ to the geodesic on the Grassmannian $\mathbb{G}(n, r)$ that connects $\widehat{\mathcal{S}}_{k-1}$

to $\mathcal{S}$. Even though both $\widehat{\mathcal{S}}_{k-1}$ and $\mathcal{S}$ are incoherent subspaces, namely, $\eta(\widehat{\mathcal{S}}_{k-1}), \eta(\mathcal{S})$ are both small, the tangent direction connecting the two is not necessarily incoherent. Despite the dependence of our results on $\widetilde{\eta}$, it is entirely possible that SNIPE with high probability approaches the true subspace $\mathcal{S}$ from an incoherent tangent direction, of which there are many. Such a result has remained beyond our reach. In fact, similar questions arise in matrix completion. Iterative Hard Thresholding (IHT) is a powerful algorithm for matrix completion with excellent empirical performance (Tanner and Wei, 2013), the convergence rate of which has remained unknown, to the best of our knowledge. With $\{M_k\}_k$ denoting the iterates of IHT, it is not difficult to see that if the differences $\{M_k - M_{k-1}\}_k$ are incoherent matrices (i.e., not spiky), then the linear convergence rate of IHT would follow from rather standard arguments.

5. Related Work

In this paper, we presented SNIPE for streaming PCA from incomplete data and, from a different perspective, SNIPE might be considered as a streaming matrix completion algorithm, see Section 3. In other words, SNIPE is a "subspace tracking" algorithm that identifies the linear structure of data as it arrives. Note also that t in our framework need not correspond to time, see Figure 1. For example, only a small portion of a large data matrix Y can be stored in the fast access memory of the processing unit, which could instead use SNIPE to fetch and process the data in small chunks and iteratively update the principal components. Moreover, SNIPE can be easily adapted to the dynamic case where the distribution of data changes over time. In dynamic subspace tracking for example, the (hidden) data vector s_t is drawn from the subspace $\mathcal{S}(t) \in \mathbb{G}(n, r)$ that varies with time. Likewise, it is easy to slightly modify SNIPE to handle noisy observations or equivalently to the case where $\{s_t\}_t$ are generated from a distribution with possibly full-rank covariance matrix. We leave investigating both of these directions to a future work.

Among several algorithms that have been proposed for tracking low-dimensional structure in a dataset from partially observed streaming data (Mitliagkas et al., 2014; Chi et al., 2013; Mardani et al., 2015; Xie et al., 2013; Eftekhari et al., 2017), SNIPE might be most closely related to GROUSE (Balzano et al., 2010; Balzano and Wright, 2013). GROUSE performs streaming PCA from incomplete data using stochastic gradient projection on the Grassmannian, updating its estimate of the true subspace with each new incomplete vector. Both GROUSE and SNIPE were designed based on the principle of least-change, discussed in Section 3. In fact, when GROUSE is sufficiently close to the true subspace and with a specific choice of its step length, both algorithms have nearly identical updates, see (Balzano and Wright, 2015, Equation 1.9). A weaker analogue of Theorem 8 for GROUSE was recently established in (Balzano and Wright, 2015). More specifically, (Balzano and Wright, 2015) stipulates that, if the current estimate $\widehat{\mathcal{S}}_k$ is sufficiently close to the true subspace $\mathcal{S}$, then $\widehat{\mathcal{S}}_{k+1}$ will be even closer to $\mathcal{S}$ in expectation. Such a result however *cannot* tell us what the local convergence rate of SNIPE is, even in expectation, see the discussion right before Theorem 10 above. In this sense, a key technical contribution of our work is establishing the local linear convergence of SNIPE, see Theorem 10, which is missing from its close competitor GROUSE. In fact, the global convergence guarantees listed in Theorem 3

and Proposition 6 are also unique to SNIPE; such theoretical guarantees are not available for GROUSE.

It might be interesting to add that our proposed update in SNIPE was inspired by that of GROUSE when we found zero-filled updates were unreliable (Eftekhari et al., 2017). However, GROUSE was derived as a purely streaming algorithm, and it therefore is not designed to leverage common low-rank structure that may be revealed when a block of vectors is processed at once. Therefore, for each block SNIPE often achieves a more significant reduction in error than is possible with GROUSE.

Lastly, both SNIPE and GROUSE have a computational complexity of $O(nr^2)$ flops per incoming vector, see Remark 1. Also, SNIPE and GROUSE both require $O(nr)$ memory elements of storage, see Remark 2. With complete data, namely, when no entries are missing, a close relative of both SNIPE and GROUSE are incremental SVD algorithms, a class of algorithms that efficiently compute the SVD of streaming data (Bunch and Nielsen, 1978; Balsubramani et al., 2013; Oja and Karhunen, 1985; Watanabe and Pakvasa, 1973; Balsubramani et al., 2013; Brand, 2002).

A streaming PCA algorithm might also be interpreted as a stochastic algorithm for PCA (Arora et al., 2012). Stochastic projected gradient ascent in this context is closely related to the classical power method. In particular, the algorithm in (Mitliagkas et al., 2014) extends the power method to handle missing data, in part by improving the main result of (Lounici, 2014). With high probability, this algorithm converges globally and linearly to the true subspace and, most notably, succeeds for arbitrarily small sampling probability p, if the scope of the algorithm T is large enough. Additionally, this algorithm too has a computational complexity of $O(nr^2)$ operations per vector and a storage requirement of $O(nr)$ memory elements. In practice, SNIPE substantially outperforms the power method, as we will see in Section 6. A disadvantage of the power method is that it updates its estimate of the true subspace with every $O(n)$ incoming vectors; the waiting time might be prohibitively long if n is large. In contrast, SNIPE frequently updates its estimate with every $b = O(r)$ incoming vectors. As we will see in Section 6, SNIPE substantially outperforms the power method in practice. Let us add that POPCA (Gonen et al., 2016) is closely related to the power method, for which the authors provide lower bounds on the achievable sample complexity. However, POPCA has substantially greater memory demand than SNIPE, since it maintains an estimate of the possibly dense $n \times n$ sample covariance matrix of incoming data.

The PETRELS algorithm (Chi et al., 2013) operates on one column at a time (rather than blocks) and global convergence for PETRELS, namely, convergence to a stationary point of the underlying nonconvex program, is known. Designed for streaming matrix completion, the algorithm in (Mardani et al., 2015) also operates on one column at a time and asymptotic onvergence to the true subspace is established, see Propositions 2 and 3 therein. This framework is also extended to tensors. MOUSSE in (Xie et al., 2013) tracks a union of subspaces rather than just one; SNIPE would function more like an ingredient of this algorithm. Asymptotic consistency of MOUSSE is also established there. The theoretical guarantees of SNIPE are more comprehensive in the sense that we also offer local convergence rate for SNIPE, see Theorems 8 and 10. ReProcs, introduced in (Lois and Vaswani, 2015), tracks a slowly changing subspace when initialized sufficiently close.

In the next section, we compare the performance of several of these algorithms in practice and find that SNIPE competes empirically with state-of-the-art algorithms.

Even though we consider uniform random sampling of the entries of incoming vectors, SNIPE can be applied to any incomplete data. For example, instead of uniform sampling analyzed here, one can perhaps sample the entries of every incoming vector based on their estimated importance. More specifically, in iteration k, one might observe each entry of the incoming vector with a probability proportional to the *leverage score* of the corresponding row of the current estimate $\widehat{\mathcal{S}}_{k-1}$. In batch or offline matrix completion, using the idea of *leveraged sampling* (as opposed to uniform sampling) alleviates the dependence on the coherence factor $\eta(\mathcal{S})$ in (16) (Eftekhari et al., 2018a; Chen, 2015). While interesting, we have not pursued this direction in the current work.

6. Simulations

This section consists of two parts: first, we empirically study the dependence of SNIPE on various parameters, and second we compare SNIPE with existing algorithms for streaming subspace estimation with missing data. In all simulations, we consider an r-dimensional subspace $\mathcal{S} \subset \mathbb{R}^n$ and a sequence of generic vectors $\{s_t\}_{t=1}^T \subset \mathcal{S}$. Each entry of these vectors is observed with probability $p \in (0, 1]$ and collected in vectors $\{y_t\}_{t=1}^T \subset \mathbb{R}^n$. Our objective is to estimate $\mathcal{S}$ from $\{y_t\}$, as described in Section 1.

Sampling probability We first set $n = 100$, $r = 5$, and let $\mathcal{S}$ be a generic r-dimensional subspace, namely, the span of an $n \times r$ standard random Gaussian matrix. For various values of probability p, we run SNIPE with block size $b = 2r = 10$ and scope of $T = 500r = 2500$, recording the average estimation error $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_K)$ over 50 trials, see (10). The average error versus probability is plotted in Figure 3a.

Subspace dimension With the same setting as the previous paragraph, we now set $p = 3r/n = 0.15$ and vary the subspace dimension r, block size $b = 2r$, and scope $T = 500r$. The average error versus subspace dimension is plotted in Figure 3b.

Ambient dimension This time, we set $r = 5$, $p = 3r/n$, $b = 2r$, $T = 500r$, and vary the ambient dimension n. In other words, we vary n while keeping the number of samples per vector fixed at about $pn = 3r$. The average error versus ambient dimension is plotted in Figure 3c. Observe that the performance of SNIPE steadily degrades as n increases. This is in agreement with Theorem 8 by substituting $p = 3r/n$ there, which states that the error reduces by a factor of $1 - Cr/n$, in expectation. A similar behavior is observed for our close competitor, namely, GROUSE (Balzano and Wright, 2015).

Block size Next we set $n = 100$, $r = 5$, $p = 3r/n$, $T = 500r$, and vary the block size b. The average error versus block size in both cases is depicted in Figure 3d. From Step 3 of Algorithm 1, a block size of $b \geq r$ is necessary for the success of SNIPE and qualitatively speaking larger values of b lead to better stability in face of missing data, which might explain the poor performance of SNIPE for very small values of b. However, as b increases, the number of blocks $K = T/b$ reduces because the scope T is held fixed. As the estimation error of SNIPE scales like $(1 - cp)^{-K}$ in Theorem 10 for a certain factor c, the performance

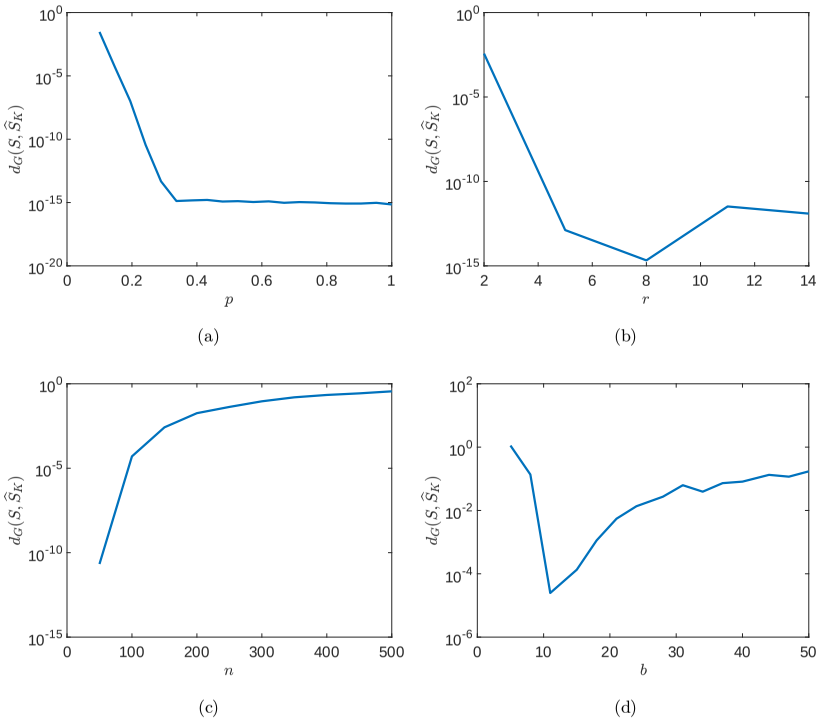


Figure 3: Performance of SNIPE as (a) sampling probability p, (b) subspace dimension r, (c) ambient dimension n, (d) block size b vary. $\widehat{\mathcal{S}}_K$ is the output of SNIPE and $d_G(\mathcal{S},\widehat{\mathcal{S}}_K)$ is its distance to the true subspace $\mathcal{S}$, which generated the input of SNIPE. See Section 6 for details and note that each panel is generated with a different and random subspace $\mathcal{S}$.

suffers in Figure 3d. It appears that the choice of $b = 2r$ in SNIPE guarantees the best empirical performance.

Coherence Lastly, we set $n = 300$, $r = 10$, $p = 3r/n$, $b = 2r$, and $T = 500r$. We then test the performance of SNIPE as the coherence of $\mathcal{S}$ varies, see (14). To that end, let $\mathcal{S} \subset \mathbb{R}^n$ be a generic subspace with orthonormal basis $S \in \mathbb{R}^{n\times r}$. In particular S is obtained by orthogonalizing the columns of a standard $n \times n$ random Gaussian matrix. Then, the average coherence of $\mathcal{S}$ over 50 trials was $3.3334 \ll n/r$ and the average estimation error of SNIPE was $2.795 \cdot 10^{-5}$. On the other hand, let $D \in \mathbb{R}^{n\times n}$ be a diagonal matrix with entries $D[i,i] = i^{-1}$ and consider $S' = DS$. Unlike $\mathcal{S}$, the new subspace $\mathcal{S}' := \text{span}(S')$ is

	$p=0.15$		$p=0.30$		$p=0.45$		$p=0.60$		$p=0.75$	
	$d_{\mathbb{G}}<10^{-3}$	$d_{\mathbb{G}}<10^{-7}$	$d_{\mathbb{G}}<10^{-3}$	$d_{\mathbb{G}}<10^{-7}$	$d_{\mathbb{G}}<10^{-3}$	$d_{\mathbb{G}}<10^{-7}$	$d_{\mathbb{G}}<10^{-3}$	$d_{\mathbb{G}}<10^{-7}$	$d_{\mathbb{G}}<10^{-3}$	$d_{\mathbb{G}}<10^{-7}$
GROUSE	**878.0** (76.1)	**1852.4** (85.2)	**294.9** (20.6)	**646.1** (26.8)	181.6 (13.8)	391.2 (18.0)	130.2 (10.3)	277.2 (12.8)	105.1 (11.7)	213.5 (12.8)
PETRELS	1689.0 (1394.1)	2853.7 (916.9)	421.8 (31.3)	1100.5 (64.2)	262.1 (25.3)	802.0 (31.1)	181.8 (20.2)	671.4 (22.8)	133.3 (21.3)	599.1 (24.0)
SNIPE	1815.7 (137.9)	3946.9 (182.4)	537.4 (39.9)	1236.4 (62.5)	282.4 (25.8)	649.6 (41.5)	171.4 (17.2)	391.4 (25.4)	105.5 (9.1)	241.6 (15.6)
SNIPE-overlap	1588.3 (183.8)	3319.5 (232.9)	318.7 (27.1)	704.6 (36.8)	**131.8** (11.6)	**296.4** (17.4)	**71.3** (5.7)	**155.6** (9.9)	**44.2** (4.2)	**91.8** (6.8)

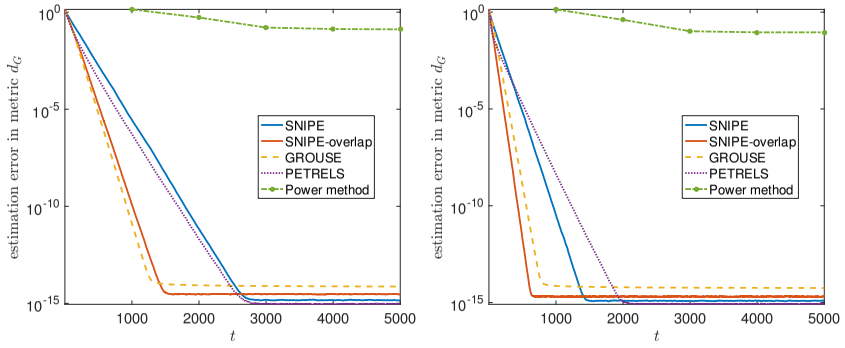
Table 1: Average number of revealed data columns needed to reach the indicated subspace recovery error for various sampling probabilities p over 100 random trials. Standard deviations are given in parenthesis.

typically coherent since the energy of its orthonormal basis S' is mostly concentrated along its first few rows. This time, the average coherence of $\mathcal{S}'$ over 50 trials was $19.1773 \approx n/r$ and the average estimation error of SNIPE was substantially worse at 0.4286.

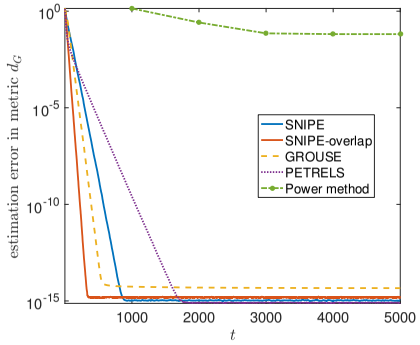
Comparisons Next we empirically compare SNIPE with GROUSE (Balzano et al., 2010; Zhang and Balzano, 2016), PETRELS (Chi et al., 2013), and the modified power method in (Mitliagkas et al., 2014). In addition to the version of SNIPE given in Algorithm 1, we also include comparisons with a simple variant of SNIPE, which we call SNIPE-overlap. Unlike SNIPE which processes disjoint blocks, SNIPE-overlap processes all overlapping blocks of data. More precisely, for a block size b, SNIPE-overlap first processes data columns $t = 1, 2, ..., b$, followed by columns $t = 2, 3, ..., b+1$, and so on, whereas regular SNIPE processes columns $t = 1, 2, ..., b$ followed by $t = b+1, b+2, ..., 2b$, etc. The theory developed in this paper does not hold for SNIPE-overlap because of lack of statistical independence between iterations, but we include the algorithm in the comparisons since it represents a minor modifications of the SNIPE-overlap framework and appears to have some empirical benefits, as detailed below.

In these experiments, we set $n = 100$, $r = 5$, $T = 5000$, and take $\mathcal{S} \subset \mathbb{R}^n$ to be a generic r-dimensional subspace and simulate noiseless data samples as before. In Figure 4 we compare the algorithms for three values of sampling probability p, which shows the average over 100 trials of the estimation error of algorithms (with respect to the metric $d_{\mathbb{G}}$) relative to the number of revealed data columns. For SNIPE, we used the block size of $b = 2r$. Having tried to get the best performance from GROUSE, we used the "greedy" step-size as proposed in (Zhang and Balzano, 2016). For (Mitliagkas et al., 2014), we set the block size as $b = 1000$ which was found empirically to yield the lowest subspace error after $T = 5000$ iterations.

In Table 1 we also compare the average number of revealed columns needed to reach a given error tolerance for each algorithm (as measured by error metric $d_{\mathbb{G}}$) for various values of the sampling probability p. We omit the modified power method from the results since it was unable to reach the given error tolerances in all cases. For the medium/high sampling rates $p = 0.45, 0.60, 0.75$, SNIPE-overlap is fastest to converge, while regular SNIPE is competitive with GROUSE and PETRELS. For the lower sampling rates $p = 0.15, 0.30$ we find GROUSE yields the fastest convergence, although SNIPE-overlap is also competitive with GROUSE for $p = 0.30$.



(a) $p = 0.30$ (b) $p = 0.45$



(c) $p = 0.60$

Figure 4: Average subspace estimation error versus number of revealed data columns at the specified sampling probability p, see Section 6 for details.

7. Theory

In this section, we prove the technical results presented in Section 4. A short word on notation is in order first. We will frequently use MATLAB's matrix notation so that, for example, $A[i,j]$ is the $[i,j]$th entry of A, and the row-vector $A[i,:]$ corresponds to the ith row of A. By $\{\epsilon_i\}_i \overset{\text{ind.}}{\sim} \text{Bernoulli}(p)$, we mean that $\{\epsilon_i\}_i$ are independent Bernoulli random variables taking one with probability of p and zero otherwise. Throughout, $E_{i,j}$ stands for the $[i,j]$th canonical matrix so that $E_{i,j}[i,j] = 1$ is its only nonzero entry. The size of $E_{i,j}$ may be inferred from the context. As usual, $\|\cdot\|$ and $\|\cdot\|_F$ stand for the spectral and Frobenius norms. In addition, $\|A\|_\infty$ and $\|A\|_{2\to\infty}$ return the largest entry of a matrix A (in magnitude) and the largest ℓ_2 norm of the rows of A, respectively. Singular values of

a matrix A are denoted by $\sigma_1(A) \geq \sigma_2(A) \geq \cdots$. For purely aesthetic reasons, we will occasionally use the notation $a \vee b = \max(a,b)$ and $a \wedge b = \min(a,b)$.

7.1. Convergence of SNIPE to a Stationary Point (Proof of Theorem 3)

Consider Program (2), namely,

$$\begin{cases} \min \;\; f_\Omega(X,\mathcal{U}) := \|P_{\mathcal{U}}^\perp X\|_F^2 + \lambda \|P_{\Omega^C}(X)\|_F^2, \\ P_\Omega(X) = Y, \end{cases} \tag{30}$$

where the minimization is over matrix $X \in \mathbb{R}^{n\times T}$ and subspace $\mathcal{U} \in \mathbb{G}(n,r)$. Before proceeding with the rest of the proof, let us for future reference record the partial derivatives of f_Ω below. Consider a small perturbation to X in the form of $X + \Delta$, where $\Delta \in \mathbb{R}^{n\times T}$. Let $U \in \mathbb{R}^{n\times r}$ and $U^\perp$ be orthonormal bases for $\mathcal{U}$ and its orthogonal complement $\mathcal{U}^\perp$, respectively. Consider also a small perturbation to U in the form of $U + U^\perp \Delta' \in \mathbb{R}^{n\times r}$, where $\Delta' \in \mathbb{R}^{(n-r)\times r}$. The perturbation to $f_\Omega(X,\mathcal{U}) = f_\Omega(X,U)$ can be written as

$$\begin{aligned} f_\Omega\left(X+\Delta, U+U^\perp\Delta'\right) &= f_\Omega(X,\mathcal{U}) + \langle \Delta, \partial_X f_\Omega(X,\mathcal{U})\rangle \\ &\quad + \left\langle \Delta', \partial_{\mathcal{U}} f_\Omega(X,\mathcal{U})\right\rangle + o(\|\Delta\|_F) + o(\|\Delta'\|_F), \end{aligned} \tag{31}$$

where $o(\cdot)$ is the standard little-o notation. The partial derivatives of f_Ω are listed below and derived in Appendix D.

Lemma 13 *For f_Ω in Program (30), the first-order partial derivatives at $(X,\mathcal{U}) \in \mathbb{R}^{n\times T} \times \mathbb{G}(n,r)$ are*

$$\begin{aligned} \partial_X f_\Omega(X,\mathcal{U}) &= 2P_{\mathcal{U}^\perp} X + 2\lambda P_{\Omega^C}(X) \in \mathbb{R}^{n\times T}, \\ \partial_{\mathcal{U}} f_\Omega(X,\mathcal{U}) &= -2(U^\perp)^* X X^* U \in \mathbb{R}^{(n-r)\times r}, \end{aligned} \tag{32}$$

where $U \in \mathbb{R}^{n\times r}$ and $U^\perp \in \mathbb{R}^{n\times(n-r)}$ are orthonormal bases for $\mathcal{U}$ and its orthogonal complement, respectively.

Recall that $Q_k \in \mathbb{R}^{r\times b}$ is a random matrix with bounded expectation, namely, $\mathbb{E}\|Q_k\|_F < \infty$. As $K \to \infty$, $\{Q_k\}_{k=1}^K$ therefore has a bounded subsequence. To keep the notation simple and without any loss of generality, we assume that in fact the sequence $\{Q_k\}_k$ is itself bounded. As $K \to \infty$ and for an integer l, we can always find an interval of length l over which the same index set and nearly the same coefficient matrix repeats. More specifically, consider an index set $\widehat{\Omega} \subseteq [1:n] \times [1:b]$ and a matrix $\widehat{Q} \in \mathbb{R}^{r\times b}$ in the support of the distributions from which $\{\Omega_k\}_{k=1}^K$ and $\{Q_k\}_{k=1}^K$ are drawn. For every integer l, as a result of the second Borel-Cantelli lemma (Durrett, 2010, pg. 64), almost surely there exists a contiguous interval

$$\kappa_l := [k_l - l + 1 : k_l], \tag{33}$$

such that

$$\Omega_k = \widehat{\Omega}, \qquad k \in \kappa_l, \tag{34}$$

$$\max_{k\in\kappa_l} \|Q_k - \widehat{Q}\|_F \leq \frac{1}{l}. \tag{35}$$

As $l \to \infty$, the measurements corresponding to the interval κ_l converge. To be specific, let $\widehat{Y} := P_{\widehat{\Omega}}(S\widehat{Q})$ and note that

$$\begin{aligned}\lim_{l\to\infty} \max_{k\in\kappa_l} \ \|Y_k - \widehat{Y}\|_F &= \lim_{l\to\infty} \max_{k\in\kappa_l} \ \|P_{\Omega_k}(SQ_k) - \widehat{Y}\|_F \\ &= \lim_{l\to\infty} \max_{k\in\kappa_l} \ \|P_{\widehat{\Omega}}(SQ_k) - \widehat{Y}\|_F \\ &= \lim_{l\to\infty} \max_{k\in\kappa_l} \ \|P_{\widehat{\Omega}}(S(Q_k - \widehat{Q}))\|_F \\ &\le \lim_{l\to\infty} \max_{k\in\kappa_l} \ \|Q_k - \widehat{Q}\|_F \\ &= 0. \end{aligned} \tag{36}$$

The above observation encourages us to exchange (Ω_k, Y_k) with $(\widehat{\Omega}, \widehat{Y})$ on the interval κ_l. Let us therefore study the program

$$\begin{cases} \min \ \ f_{\widehat{\Omega}}(X, \mathcal{U}), \\ P_{\widehat{\Omega}}(X) = \widehat{Y}, \end{cases} \tag{37}$$

where the minimization is over all matrices $X \in \mathbb{R}^{n\times b}$ and subspaces $\mathcal{U} \in \mathbb{G}(n, r)$. From a technical viewpoint, it is in fact more convenient to relax the equality constraint above as

$$\begin{cases} \min \ \ f_{\widehat{\Omega}}(X, \mathcal{U}), \\ \|P_{\widehat{\Omega}}(X) - \widehat{Y}\|_F \le \epsilon, \end{cases} \tag{38}$$

for $\epsilon > 0$. We fix ϵ for now. Let us next use alternative minimization to solve Program (38). More specifically, recall (33) and consider the initialization $\widehat{\mathcal{S}}_{k_l-l,\epsilon} := \widehat{\mathcal{S}}_{k_l-l}$, where $\widehat{\mathcal{S}}_{k_l-l}$ is the output of SNIPE at iteration $k_l - l$, see Algorithm 1. For every $k \in \kappa_l$, consider the program

$$\begin{cases} \min \ \ f_{\widehat{\Omega}}(X, \widehat{\mathcal{S}}_{k-1,\epsilon}), \\ \|P_{\widehat{\Omega}}(X) - \widehat{Y}\|_F \le \epsilon, \end{cases} \tag{39}$$

and let $R_{k,\epsilon}$ be a minimizer of Program (39). We then update the subspace by solving

$$\min_{\mathcal{U}\in\mathbb{G}(n,r)} \ f_{\widehat{\Omega}}(R_{k,\epsilon}, \mathcal{U}), \tag{40}$$

and setting $\widehat{\mathcal{S}}_{k,\epsilon}$ to be a minimizer of Program (40). Recalling the definition of $f_{\widehat{\Omega}}$ in Program (30) and in light of the Eckart-Young-Mirsky Theorem, Program (40) can be solved by computing top r left singular vectors of $R_{k,\epsilon}$ (Eckart and Young, 1936; Mirsky, 1966). For future reference, note that the optimality and hence stationarity of $\widehat{\mathcal{S}}_{k,\epsilon}$ in Program (40) dictates that

$$\partial_{\mathcal{U}} f_{\widehat{\Omega}}(R_{k,\epsilon}, \widehat{\mathcal{S}}_{k,\epsilon}) = 0, \qquad k \in \kappa_l, \tag{41}$$

where $\partial_{\mathcal{U}} f$ was specified in Lemma 13. From the above construction of the sequence $\{(R_{k,\epsilon}, \widehat{\mathcal{S}}_{k,\epsilon})\}_{k\in\kappa_l}$, we also observe that

$$0 \le f_{\widehat{\Omega}}(R_{k,\epsilon}, \widehat{\mathcal{S}}_{k,\epsilon}) \le f_{\widehat{\Omega}}(R_{k,\epsilon}, \widehat{\mathcal{S}}_{k-1,\epsilon}) \le f_{\widehat{\Omega}}(R_{k-1,\epsilon}, \widehat{\mathcal{S}}_{k-1,\epsilon}), \tag{42}$$

for every $k \in [k_l - l + 2 : k_l] \subset \kappa_l$, see (33). That is, $\{f_{\widehat{\Omega}}(R_{k,\epsilon}, \widehat{\mathcal{S}}_{k,\epsilon})\}_{k\in\kappa_l}$ is a nonincreasing and nonnegative sequence. It therefore holds that

$$\lim_{l\to\infty} \left| f_{\widehat{\Omega}}(R_{k_l-1,\epsilon}, \widehat{\mathcal{S}}_{k_l-1,\epsilon}) - f_{\widehat{\Omega}}(R_{k_l,\epsilon}, \widehat{\mathcal{S}}_{k_l-1,\epsilon}) \right| = 0. \tag{43}$$

By the feasibility of $R_{k_l-1,\epsilon}$ in Program (39) and by the continuity of $f_{\widehat{\Omega}}(X, \mathcal{U})$ in X, we conclude in light of (43) that $R_{k_l-1,\epsilon}$ too is a minimizer (and hence also a stationary point) of Program (39) and in the limit of $l \to \infty$. We therefore find by writing the stationarity conditions of Program (39) at $R_{k_l,\epsilon}$ that

$$\|P_{\widehat{\Omega}}(R_{k_l,\epsilon}) - \widehat{Y}\|_F \le \epsilon, \tag{44}$$

$$\lim_{l\to\infty} \left\| \partial_X f_{\widehat{\Omega}}(R_{k_l,\epsilon}, \widehat{\mathcal{S}}_{k_l,\epsilon}) + \lambda_{k_l,\epsilon}(P_{\widehat{\Omega}}(R_{k_l,\epsilon}) - \widehat{Y}) \right\|_F = 0. \tag{45}$$

for nonnegative $\lambda_{k_l,\epsilon}$. Recalling the definition of $f_{\widehat{\Omega}}$ and that $\lambda > 0$ by assumption, we observe that Program (39) is strongly convex in $P_{\widehat{\Omega}^C}(X)$ and consequently any pair of minimizers of Program (39) must agree on the index set $\widehat{\Omega}^C$. Optimality of $R_{k_l,\epsilon}$ and limit optimality of $R_{k_l-1,\epsilon}$ in Program (39) therefore imply that

$$\lim_{l\to\infty} \left\| P_{\widehat{\Omega}^C}(R_{k_l-1,\epsilon} - R_{k_l,\epsilon}) \right\|_F = 0. \tag{46}$$

On the index set $\widehat{\Omega}$, on the other hand, the feasibility of both $R_{k_l-1,\epsilon}$ and $R_{k_l,\epsilon}$ in Program (39) implies that

$$\begin{aligned} \|P_{\widehat{\Omega}}(R_{k_l-1,\epsilon} - R_{k_l,\epsilon})\|_F &\le \|P_{\widehat{\Omega}}(R_{k_l-1,\epsilon}) - \widehat{Y}\|_F + \|P_{\widehat{\Omega}}(R_{k_l,\epsilon}) - \widehat{Y}\|_F \qquad \text{(triangle inequality)} \\ &\le 2\epsilon. \end{aligned} \tag{47}$$

Combining (46) and (47) yields that

$$\begin{aligned} &\lim_{l\to\infty} \|R_{k_l-1,\epsilon} - R_{k_l,\epsilon}\|_F \\ &\le \lim_{l\to\infty} \|P_{\widehat{\Omega}}(R_{k_l-1,\epsilon} - R_{k_l,\epsilon})\|_F + \lim_{l\to\infty} \|P_{\widehat{\Omega}^C}(R_{k_l-1,\epsilon} - R_{k_l,\epsilon})\|_F \qquad \text{(triangle inequality)} \\ &\le 2\epsilon, \qquad \text{(see (46,47))} \end{aligned} \tag{48}$$

In light of (39), $\{R_{k_l,\epsilon}\}_l$ is bounded and consequently has a convergent subsequence. Without loss of generality and to simplify the notation, we assume that $\{R_{k_l,\epsilon}\}_l$ is itself convergent, namely, that there exists $R_\epsilon \in \mathbb{R}^{n\times b}$ for which

$$\lim_{l\to\infty} \|R_{k_l,\epsilon} - R_\epsilon\|_F \le 2\epsilon. \tag{49}$$

Let us now send ϵ to zero in (49) to obtain that

$$\lim_{\epsilon\to 0} \lim_{l\to\infty} \|R_{k_l,\epsilon} - R_\epsilon\|_F \le \lim_{\epsilon\to 0} 2\epsilon = 0. \tag{50}$$

We next show that it is possible to essentially change the order of limits above and also conclude that $(R, \widehat{\mathcal{S}})$ coincides with the output of SNIPE in limit. The following result is proved in Appendix E.

Lemma 14 *With the setup above, there exist a sequence* $\{\epsilon_i\}_i$ *with* $\lim_{i\to\infty}\epsilon_i = 0$ *and a matrix* $R \in \mathbb{R}^{n\times b}$ *such that*

$$\lim_{l,i\to\infty} \|R_{k_l,\epsilon_i} - R\|_F = \lim_{l\to\infty}\lim_{i\to\infty} \|R_{k_l,\epsilon_i} - R\|_F = \lim_{i\to\infty}\lim_{l\to\infty} \|R_{k_l,\epsilon_i} - R\|_F = 0. \tag{51}$$

Moreover, suppose that the output of SNIPE *in every iteration has a spectral gap in the sense that there exists* $\tau > 0$ *such that*

$$\frac{\sigma_r(R_k)}{\sigma_{r+1}(R_k)} \geq 1 + \tau, \tag{52}$$

for every k. *Let* $\widehat{\mathcal{S}}_{k_l,\epsilon_i}$ *and* $\widehat{\mathcal{S}}$ *be the span of top* r *left singular vectors of* R_{k_l,ϵ_i} *and* R, *respectively. Then it holds that*

$$\lim_{l,i\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_{k_l,\epsilon_i}, \mathcal{S}) = 0. \tag{53}$$

Lastly, in the limit of $l \to \infty$, SNIPE *produces* $(R, \widehat{\mathcal{S}})$ *in every iteration, namely,*

$$\lim_{l\to\infty} \|R_{k_l} - R\|_F = \lim_{l\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_{k_l}, \widehat{\mathcal{S}}) = 0, \tag{54}$$

where $(R_{k_l}, \widehat{\mathcal{S}}_{k_l})$ *is the output of* SNIPE *in iteration* k_l, *see Algorithm 1.*

In fact, the pair $(R, \widehat{S})$ from Lemma 14 is stationary in limit in the sense described next and proved in Appendix F.

Lemma 15 *The pair* $(R, \widehat{S})$ *in Lemma 14 is a stationary point of the program*

$$\begin{cases} \min \ \ f_{\Omega_{k_l}}(X,\mathcal{U}), \\ P_{\Omega_{k_l}}(X) = Y_{k_l}, \end{cases} \tag{55}$$

as $l \to \infty$. *The minimization above is over all matrices* $X \in \mathbb{R}^{n\times b}$ *and subspaces* $\mathcal{U} \in \mathbb{G}(n, r)$. *More specifically, it holds that*

$$\lim_{l\to\infty} \|\partial_{\mathcal{U}} f_{\Omega_{k_l}}(R, \widehat{\mathcal{S}})\|_F = 0, \tag{56}$$

$$\lim_{l\to\infty} \|P_{\Omega_{k_l}}(R) - Y_{k_l}\|_F = 0, \tag{57}$$

$$\lim_{l\to\infty} \|P_{\Omega^C_{k_l}}(\partial_X f_{\Omega_{k_l}}(R, \widehat{\mathcal{S}}))\|_F = 0. \tag{58}$$

In words, Lemmas 14 and 15 together imply that the output of SNIPE in limit is a stationary point of Program (55). This completes the proof of Theorem 3.

7.2. Convergence of SNIPE (Proof of Proposition 6)

In iteration k of SNIPE, we partially observe the data block $SQ_k \in \mathbb{R}^{n\times b}$ on a random index set $\Omega_k \subset [1:n] \times [1:b]$, where $Q_k \in \mathbb{R}^{r\times b}$ is a random coefficient matrix. We collect the observations in $Y_k = P_{\Omega_k}(SQ_k) \in \mathbb{R}^{n\times b}$, see Sections 2 and 3 for the detailed setup. Note that R_k in (1) can be written as

$$R_k = Y_k + P_{\Omega_k^C}(\widehat{S}_{k-1}Q'_k) = P_{\Omega_k}(SQ_k) + P_{\Omega_k^C}(\widehat{S}_{k-1}Q'_k), \tag{59}$$

where

$$Q'_k := \begin{bmatrix} \cdots & \left(\widehat{S}^*_{k-1}P_{\omega_t}\widehat{S}_{k-1} + \lambda I_r\right)^\dagger y_t & \cdots \end{bmatrix} \in \mathbb{R}^{r\times b}. \tag{60}$$

By (13), there exists $Q''_k \in \mathbb{R}^{r\times b}$ and

$$R'_k := P_{\Omega_k}(SQ_k) + P_{\Omega_k^C}(\widehat{S}Q''_k), \tag{61}$$

such that

$$\lim_{k\to\infty} \|R_k - R'_k\|_F = 0. \tag{62}$$

In (61) above, $\widehat{S} \in \mathbb{R}^{n\times r}$ is an orthonormal basis for the subspace $\widehat{\mathcal{S}}$. In Algorithm 1, the rank-r truncated SVD of R_k spans $\widehat{\mathcal{S}}_k \in \mathbb{G}(n, r)$, namely, the output of SNIPE in iteration k. Let also $\widehat{\mathcal{S}}'_k \in \mathbb{G}(n, r)$ denote the span of rank-r truncated SVD of R'_k. The existence of the reject option in Algorithm 1 with positive τ implies that R_k has a spectral gap and therefore $\widehat{\mathcal{S}}_k$ is uniquely defined. Combining this with (62), we find that $\widehat{\mathcal{S}}'_k$ too is uniquely defined in the limit of $k \to \infty$. Therefore another consequence of (62) is that

$$\lim_{k\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_k, \widehat{\mathcal{S}}'_k) = 0. \tag{63}$$

Then we have that

$$\lim_{k\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}'_k, \widehat{\mathcal{S}}) \leq \lim_{k\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}'_k, \widehat{\mathcal{S}}_k) + \lim_{k\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_k, \widehat{\mathcal{S}}) \leq 0 + 0, \qquad \text{(see (63,13))} \tag{64}$$

namely, $\widehat{\mathcal{S}}'_k$ converges to $\widehat{\mathcal{S}}$ in the limit too. Let us now rewrite R'_k as

$$R'_k = P_{\Omega_k}(SQ_k - \widehat{S}Q''_k) + \widehat{S}Q''_k, \qquad \text{(see (61))} \tag{65}$$

which, together with (64), implies that

$$\lim_{k\to\infty} \|\widehat{S}^* P_{\Omega_k}(SQ_k - \widehat{S}Q''_k)\|_F = 0. \tag{66}$$

We can rewrite the above limit in terms of the data vectors (rather than data blocks) to obtain that

$$\lim_{t\to\infty} \|\widehat{S}^* P_{\omega_t}(Sq_t - \widehat{S}q''_t)\|_F = 0, \tag{67}$$

where $\{q_t, q''_t\}_t$ form the columns of the blocks $\{Q_k, Q''_k\}_k$, and the index sets $\{\omega_t\}_t \subseteq [1:n]$ form $\{\Omega_k\}_k$. There almost surely exists a subsequence $\{t_i\}_i$ over which $\omega_{t_i} = \{1\}$, namely, there is a subsequence where we only observe the first entry of the incoming data vector. Consider a vector $q^1 \in \mathbb{R}^r$ in the support of the distribution from which $\{q_t\}_t$

are drawn. Then there also exists a subsequence of $\{t_i\}_i$, denoted by $\{t_{i_j}\}_{i_j}$, such that $\lim_{j\to\infty}\|q_{t_{i_j}} - q^1\|_2 = 0$. Restricted to the subsequence $\{t_{i_j}\}_j$, (67) reads as

$$0 = \lim_{j\to\infty}\|\widehat{S}^* P_{\omega_{t_{i_j}}}(Sq_{t_{i_j}} - \widehat{S}q''_{t_{i_j}})\|_F = \lim_{j\to\infty}\|\widehat{S}^* P_{\{1\}}(Sq_{t_{i_j}} - \widehat{S}q''_{t_{i_j}})\|_F = \|\widehat{S}^* P_{\{1\}}(Sq^1 - \widehat{S}q''^1)\|_F, \tag{68}$$

where we set $q''^1 := \lim_{j\to\infty} q''_{t_{i_j}}$; the limit exists by (60). Likewise, we can show that

$$v^l := P_{\{l\}}(Sq^l - \widehat{S}q''^l) \in \mathcal{S}^\perp, \qquad l \in [1:n], \tag{69}$$

where $\{q''^l\}_l$ are defined similarly. Because $\dim(\mathcal{S}^\perp) = n - r$, at most $n - r$ of the vectors $\{v^l\}_{l=1}^n$ are linearly independent. Because the supports of $\{v^l\}_l$ are disjoint, it follows that there are at most $n - r$ of the vectors $\{v^l\}_{l=1}^n$ are nonzero. Put differently, there exists an index set $I \subset [1:n]$ of size at least r such that

$$Sq^l = \widehat{S}q''^l, \qquad l \in I. \tag{70}$$

Almost surely, $\{q^l\}_{l\in I} \subset \mathbb{R}^r$ form a basis for $\mathbb{R}^r$, and therefore $\mathcal{S} \subseteq \widehat{\mathcal{S}}$. Because $\widehat{\mathcal{S}} \in \mathbb{G}(n,r)$ by assumption, it follows that $\widehat{\mathcal{S}} = \mathcal{S}$, which completes the proof of Proposition 6.

7.3. Locally Linear Convergence of SNIPE (Proof of Theorems 8 and 10)

At iteration $k \in [2:K]$, SNIPE uses the current estimate $\widehat{\mathcal{S}}_{k-1}$ and the new incomplete block Y_k to produce a new estimate $\widehat{\mathcal{S}}_k$ of the true subspace $\mathcal{S}$. The main challenge here is to compare the new and old principal angles with $\mathcal{S}$, namely, compare $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k)$ and $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1})$. Lemma 16 below, proved in Appendix G, loosely speaking states that $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k)$ reduces by a factor of $1 - O(p)$ in expectation in every iteration, when $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k) \lesssim p^{\frac{5}{2}}$ and ignoring all other parameters in this qualitative discussion. In other words, when sufficiently small, the estimation error of SNIPE reduces in every iteration, but in expectation. The actual behavior of SNIPE is more nuanced. Indeed, Lemma 16 below also adds that the estimation error $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k)$ in fact contracts in *some* iterations by a factor of $1 - Cp$, namely,

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k) \lesssim (1 - Cp) \cdot d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}),$$

provided that $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) \lesssim p^{\frac{5}{2}}$. That is, when sufficiently small, the estimation error of SNIPE reduces in some but not all iterations. In the rest of iterations, the error does not increase by much, namely,

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k) \approx d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}),$$

with high probability and provided that $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) \lesssim p^{\frac{5}{2}}$.

Lemma 16 *Fix $k \in [2:K]$, $\alpha, \nu \geq 1$, and $c > 0$. Let $\mathfrak{E}_{k-1}$ be the event where*

$$p \gtrsim \alpha^2 \log^2 b \log n \frac{\eta(\widehat{\mathcal{S}}_{k-1})r}{n}, \tag{71}$$

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) \lesssim \frac{p^{\frac{7}{2}} nb}{\alpha c \log b \sqrt{r \log n}}, \tag{72}$$

and let $\mathfrak{E}'_k$ be the event where $\|Q_k\| \leq \nu \cdot \sigma_{\min}$, where $\sigma_{\min}$ is the reject threshold in SNIPE, *see Algorithm 1. Then it holds that*

$$\mathbb{E}\left[d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k) \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k\right] \leq \nu\left(1 - \frac{p}{2} + \frac{p^3 nb}{c}\right) d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) + \frac{b^{-C\alpha}}{\sqrt{r}}. \tag{73}$$

Moreover, conditioned on $\widehat{\mathcal{S}}_{k-1}$ and the event $\mathfrak{E}_{k-1} \cap \mathfrak{E}'_k$, it holds that

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k) \leq \nu\left(1 + \frac{p^3 nb}{c}\right) d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}), \tag{74}$$

except with a probability of at most $b^{-C\alpha}$. Lastly, a stronger bound holds conditioned on $\widehat{\mathcal{S}}_{k-1}$ and the event $\mathfrak{E}_{k-1} \cap \mathfrak{E}'_k$, namely,

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k) \leq \nu\left(1 - \frac{p}{4} + \frac{p^3 nb}{c}\right) d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) \tag{75}$$

with a probability of at least

$$\phi_k(\alpha) := 1 - \exp\left(-\frac{C_1 p^2 nb}{\widetilde{\eta}_{k-1}}\right) - b^{-C\alpha}, \tag{76}$$

where

$$\widetilde{\eta}_k = \widetilde{\eta}(P_{\widehat{\mathcal{S}}_{k-1}^\perp} S_k) := nb \cdot \frac{\|P_{\widehat{\mathcal{S}}_{k-1}^\perp} S_k\|_\infty^2}{\|P_{\widehat{\mathcal{S}}_{k-1}^\perp} S_k\|_F^2}. \tag{77}$$

Let us now use Lemma 16 to complete the proofs of Theorems 8 and 10.

7.3.1. Proof of Theorem 8

With the choice of $c = 4p^2 nb$ and $\nu = 1/\sqrt{1 - p/4}$, (73) reads as

$$\begin{aligned}
\mathbb{E}\left[d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k) \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k\right] &\leq \nu\left(1 - \frac{p}{2} + \frac{p^3 nb}{c}\right) d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) + \frac{b^{-C\alpha}}{\sqrt{r}} \qquad \text{(see (73))} \\
&= \sqrt{1 - \frac{p}{4}}\, d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) + \frac{b^{-C\alpha}}{\sqrt{r}} \\
&\leq \left(1 - \frac{p}{8}\right) d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) + \frac{b^{-C\alpha}}{\sqrt{r}}.
\end{aligned} \tag{78}$$

With the choice of

$$\alpha = -\frac{C \log\left(p\sqrt{r}\, d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1})/16\right)}{\log b}, \tag{79}$$

for an appropriate constant C above, the bound in (78) simplifies to

$$\begin{aligned}
\mathbb{E}\left[d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k) \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k\right] &\leq \left(1 - \frac{p}{8}\right) d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) + \frac{b^{-C\alpha}}{\sqrt{r}} \qquad \text{(see (78))} \\
&\leq \left(1 - \frac{p}{8}\right) d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1}) + \frac{p}{16} d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{k-1})
\end{aligned}$$

$$\leq \left(1-\frac{p}{16}\right) d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{k-1}). \tag{80}$$

Lastly we remove the conditioning on $\mathfrak{E}'_k$ above. Using the law of total expectation, we write that

$$\begin{aligned}
&\mathbb{E}\left[d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k)\,|\,\widehat{\mathcal{S}}_{k-1},\mathfrak{E}_{k-1}\right]\\
&=\mathbb{E}\left[d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k)\,|\,\widehat{\mathcal{S}}_{k-1},\mathfrak{E}_{k-1},\mathfrak{E}'_k\right]\cdot\Pr[\mathfrak{E}'_k]+\mathbb{E}\left[d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k)\,|\,\widehat{\mathcal{S}}_{k-1},\mathfrak{E}_{k-1},\mathfrak{E}'^C_k\right]\cdot\Pr[\mathfrak{E}'^C_k]\\
&\leq\mathbb{E}\left[d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k)\,|\,\widehat{\mathcal{S}}_{k-1},\mathfrak{E}_{k-1},\mathfrak{E}'_k\right]+\Pr[\mathfrak{E}'^C_k]\qquad\text{(see (10))}\\
&\leq\left(1-\frac{p}{16}\right)d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{k-1})+\Pr[\mathfrak{E}'^C_k]\qquad\text{(see (80))}\\
&\leq\left(1-\frac{p}{32}\right)d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{k-1}),
\end{aligned}\tag{81}$$

where the last line holds if

$$\Pr[\mathfrak{E}'^C_k]\leq\frac{p}{32}d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{k-1}).$$

With the choice of c,ν,α above, let us also rewrite the event $\mathfrak{E}_{k-1}$ in Lemma 16. First, we rewrite (72) as

$$\begin{aligned}
d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{k-1})&\lesssim\frac{p^{\frac{7}{2}}nb}{\alpha c\log b\sqrt{r\log n}}=\frac{Cp^{\frac{3}{2}}}{\alpha\log b\sqrt{r\log n}}\\
&=-\frac{Cp^{\frac{3}{2}}}{\log\left(p\sqrt{r}d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{k-1})/16\right)\sqrt{r\log n}}.\qquad\text{(see (79))}
\end{aligned}\tag{82}$$

Second, we replace the coherence $\eta(\widehat{\mathcal{S}}_{k-1})$ in (71) with the simpler quantity $\eta(\mathcal{S})$. We can do so thanks to Lemma 20 which roughly speaking states that a pair of subspaces $\mathcal{A}$ and $\mathcal{B}$ with a small principal angle have similar coherences, namely, $\theta_1(\mathcal{A},\mathcal{B})\approx 0\Longrightarrow\eta(\mathcal{A})\approx\eta(\mathcal{B})$. More concretely, note that

$$\begin{aligned}
\sqrt{\eta(\widehat{\mathcal{S}}_{k-1})}&\leq\sqrt{\eta\left(\mathcal{S}\right)}+d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{k-1})\sqrt{n}\qquad\text{(see Lemma 20)}\\
&\leq\sqrt{\eta(\mathcal{S})}+Cp^{\frac{3}{2}}\sqrt{\frac{n}{r\log n}}\qquad\text{(see (82))}\\
&\leq\sqrt{\eta(\mathcal{S})}+1\qquad\left(\text{if }p\lesssim\frac{1}{\sqrt{nb}}\right)\\
&\leq 2\sqrt{\eta(\mathcal{S})}.\qquad\text{(see (15))}
\end{aligned}\tag{83}$$

This completes the proof of Theorem 8.

7.3.2. Proof of Theorem 10

For $K_0\in[1:K]$, we condition on $\widehat{\mathcal{S}}_{K_0}$. For positive c to be set later, suppose that

$$d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0})\lesssim\frac{e^{-\frac{Cp^3nb}{\eta}}p^{\frac{7}{2}}nb}{\alpha c\log b\sqrt{\log n}}.\tag{84}$$

In particular, (84) implies that the error at iteration K_0 is small enough to activate Lemma 16, see (72). For $\nu\geq 1$ to be set later, we condition for now on the event

$$\mathfrak{E}':=\cap_{k=K_0+1}^{K}\mathfrak{E}'_k,\tag{85}$$

where the event $\mathfrak{E}'_k$ was defined in Lemma 16. Suppose also that (71) holds for every $k\in[K_0+1:K]$, namely,

$$p\gtrsim\max_{k\in[K_0:K-1]}\eta(\widehat{\mathcal{S}}_k)\cdot\frac{r\log^2 b\log n}{n},\tag{86}$$

which will next allow us to apply Lemma 16 repeatedly to all iterations in the interval $[K_0+1:K]$. With the success probability $\phi_k(\alpha)$ defined in Lemma 16, let us also define

$$\phi(\alpha):=\min_{k\in[K_0+1:K]}\phi_k(\alpha)=1-\exp\left(-\frac{C_1p^2nb}{\widetilde{\eta}}\right)-b^{-C\alpha},\qquad\widetilde{\eta}:=\max_{k\in[K_0+1:K]}\widetilde{\eta}_k\geq 1,\tag{87}$$

where the inequality above follows because $\widetilde{\eta}_k\geq 1$ for every k, see (77). We now partition $[K_0+1:K]$ into (non-overlapping) intervals $\{I_i\}_i$, each with the length

$$l=\frac{C_2\log b\log(K-K_0)}{\phi(\alpha)},\tag{88}$$

except possibly the last interval which might be shorter. Consider one of these intervals, say I_1. Then by Lemma 16 and the union bound, (74) holds for every iteration $k\in I_1$ except with a probability of at most $l\cdot b^{-C\alpha}$ because the length of I_1 is l. That is, the estimation error does not increase by much in every iteration in the interval I_1. In some of these iterations, the error in fact reduces. More specifically, (75) holds in iteration k with a probability of at least $\phi_k(\alpha)$, see (76). While $b^{-C\alpha}$ in (76) can be made arbitrary small by increasing the tuning parameter α, this of course would not necessarily make $\phi_k(\alpha)$ arbitrary close to one. That is, there is a sizable chance that the estimation error does not contract in iteration k. However, (75) holds at least in one iteration in the interval I_1 except with a probability of at most

$$(1-\phi(\alpha))^l\leq e^{-\phi(\alpha)l}=b^{-C_2\log(K-K_0)}.\qquad(1+a\leq e^a)$$

Therefore, except with a probabilty of at most $lb^{-C\alpha}+b^{-C_2\log(K-K_0)}$, (75) holds at least once and (74) holds for all iterations in the interval I_1. It immediately follows that

$$\begin{aligned}
&\frac{d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0+l})}{d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0})}\\
&\leq\nu\left(1-\frac{p}{4}+\frac{p^3nb}{c}\right)\cdot\nu^{l-1}\left(1+\frac{p^3nb}{c}\right)^{l-1}\qquad\text{(see (74,75))}\\
&\leq\nu^l\exp\left(-\frac{p}{4}+\frac{p^3np}{c}+(l-1)\frac{p^3nb}{c}\right)\qquad(1+a\leq e^a)\\
&=\nu^l\exp\left(-\frac{p}{4}+\frac{lp^3nb}{c}\right),
\end{aligned}\tag{89}$$

except with a probability of at most

$$lb^{-C\alpha} + b^{-C_2 \log(K-K_0)}. \tag{90}$$

In particular, suppose that

$$c \geq 4lp^2nb, \tag{91}$$

so that the exponent in the last line of (89) is negative. Let

$$i_{\max} := \left\lfloor \frac{K-K_0}{l} \right\rfloor,$$

and note that

$$\frac{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_K)}{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0})} = \prod_{i=1}^{i_{\max}} \frac{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0+il})}{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0+(i-1)l})} \cdot \frac{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_K)}{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0+i_{\max}l})}. \tag{92}$$

By applying the bound in (89) to all intervals $\{I_i\}_i$, we then conclude that

$$\begin{aligned}
&\frac{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_K)}{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0})} \\
&= \prod_{i=1}^{i_{\max}} \frac{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0+il})}{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0+(i-1)l})} \cdot \frac{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_K)}{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0+i_{\max}l})} \qquad \text{(see (92))} \\
&\leq \left(\nu^l \exp\left(-\frac{p}{4} + \frac{lp^3nb}{c}\right)\right)^{\left\lfloor \frac{K-K_0}{l} \right\rfloor} \cdot \nu^l \left(1 + \frac{p^3nb}{c}\right)^l \qquad \text{(see (89,74))} \\
&\leq \left(\nu^l \exp\left(-\frac{p}{4} + \frac{lp^3nb}{c}\right)\right)^{\left\lfloor \frac{K-K_0}{l} \right\rfloor} \cdot \nu^l \exp\left(\frac{lp^3nb}{c}\right) \qquad (1+a \leq e^a) \\
&\leq \nu^{K-K_0} \exp\left(\left\lfloor \frac{K-K_0}{2l} \right\rfloor \left(-\frac{p}{4} + \frac{lp^3nb}{c}\right)\right) \cdot \exp\left(\frac{lp^3nb}{c}\right) \\
&\leq \nu^{K-K_0} \exp\left(\frac{K-K_0}{2l} \left(-\frac{p}{4} + \frac{lp^3nb}{c}\right)\right) \cdot \exp\left(\frac{lp^3nb}{c}\right) \quad \text{(if } K-K_0 \geq l \text{ and (91) holds)} \\
&\leq \nu^{K-K_0} \exp\left(\frac{K-K_0}{2l} \left(-\frac{p}{4} + \frac{3lp^3nb}{c}\right)\right), \qquad \text{(if } K-K_0 \geq l\text{)}
\end{aligned} \tag{93}$$

except with a probability of at most

$$\begin{aligned}
&\left\lceil \frac{K-K_0}{l} \right\rceil \left(lb^{-C\alpha} + b^{-C_2 \log(K-K_0)}\right) \\
&\quad \leq \frac{2(K-K_0)}{l} \left(lb^{-C\alpha} + b^{-C_2 \log(K-K_0)}\right) \qquad \text{(if } K-K_0 \geq l\text{)} \\
&\quad \leq (K-K_0)e^{-C\alpha} + b^{-CC_2 \log(K-K_0)},
\end{aligned} \tag{94}$$

which follows from an application of the union bound to the failure probability in (90). With the choice of $\alpha = \alpha' \log b \log(K-K_0)$ with sufficiently large α', the failure probability in (94) simplifies to

$$(K-K_0)e^{-C\alpha} + b^{-C\log(K-K_0)} = (K-K_0) \cdot (K-K_0)^{-C\alpha' \log b} + b^{-C\log(K-K_0)}$$

$$\begin{aligned}
&\leq (K-K_0)^{-C\alpha' \log b} + b^{-C\log(K-K_0)} \\
&= b^{-C\alpha' \log(K-K_0)} + b^{-C\log(K-K_0)} \\
&\leq b^{-C\log(K-K_0)}.
\end{aligned} \tag{95}$$

The next step involves elementary bookkeeping to upper-bound the last line of (93). Suppose that

$$\alpha \gtrsim \frac{\log\left(\frac{C\widetilde{\eta}}{p^2nb}\right)}{\log b}, \tag{96}$$

$$p \lesssim \frac{1}{\sqrt{nb}}. \tag{97}$$

Using (96) and (97) with appropriate constants replacing $\gtrsim$ and $\lesssim$ above, we may verify that

$$b^{-C\alpha} \leq \frac{C_1p^2nb}{4\widetilde{\eta}}, \qquad \text{(see (96))} \tag{98}$$

$$\frac{C_1p^2nb}{\widetilde{\eta}} \lesssim 2, \qquad \text{((97) and } \widetilde{\eta} \geq 1\text{)} \tag{99}$$

$$\begin{aligned}
\phi(\alpha) &= 1 - \exp\left(-\frac{C_1p^2nb}{\widetilde{\eta}}\right) - b^{-C\alpha} \qquad \text{(see (87))} \\
&\geq \frac{1}{2} \cdot \frac{C_1p^2nb}{\widetilde{\eta}} - b^{-C\alpha} \qquad \left(\text{(99) and } e^{-a} \leq 1 - \frac{a}{2} \text{ for } a \lesssim 2\right) \\
&\geq \frac{C_1p^2nb}{4\widetilde{\eta}}, \qquad \text{(see (98))}
\end{aligned} \tag{100}$$

$$\begin{aligned}
l &= \frac{C_2 \log b \log(K-K_0)}{\phi(\alpha)} \qquad \text{(see (88))} \\
&\leq \frac{4C_2\widetilde{\eta} \log b \log(K-K_0)}{C_1p^2nb}. \qquad \text{(see (100))}
\end{aligned} \tag{101}$$

Now with the choice of

$$c = \frac{96C_2}{C_1}\widetilde{\eta} \log b \log(K-K_0), \tag{102}$$

we may verify that

$$-\frac{p}{4} + \frac{3lp^3nb}{c} \leq -\frac{p}{8}, \qquad \text{(see (101,102))} \tag{103}$$

and, revisiting (93), we find that

$$\begin{aligned}
&\frac{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_K)}{d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_{K_0})} \\
&\leq \nu^{K-K_0} \exp\left(\frac{K-K_0}{2l} \left(-\frac{p}{4} + \frac{3lp^3nb}{c}\right)\right) \qquad \text{(see (93))}
\end{aligned}$$

$$
\begin{aligned}
&\le \nu^{K-K_0}\exp\left(-\frac{(K-K_0)p}{16l}\right) && \text{(see (103))}\\
&= \left(\nu\exp\left(-\frac{p}{16l}\right)\right)^{K-K_0}\\
&\le \left(\nu\exp\left(-\frac{Cp^3nb}{\widetilde{\eta}\log b\log(K-K_0)}\right)\right)^{K-K_0} && \text{(see (101))}\\
&\le \nu^{K-K_0}\left(1-\frac{Cp^3nb}{\widetilde{\eta}\log b\log(K-K_0)}\right)^{K-K_0} && \text{((97) and } \widetilde{\eta}\ge 1)\\
&\le \left(1-\frac{Cp^3nb}{\widetilde{\eta}\log b\log(K-K_0)}\right)^{K-K_0}, && (104)
\end{aligned}
$$

where we set

$$\nu = 1+\frac{Cp^3nb}{\widetilde{\eta}\log b\log(K-K_0)}, \tag{105}$$

for an appropriate choice of constant C. To reiterate, conditioned on the event $\mathfrak{E}'$ in (85), (104) is valid provided that (84,86,96,97) hold and except with the probability of at most $b^{-C\log(K-K_0)}$, see (95). In particular, to better interpret (86), we next replace the coherence $\eta(\widehat{\mathcal{S}}_k)$ therein with the simpler quantity $\eta(\mathcal{S})$. We can do so thanks to Lemma 20 which roughly speaking states that a pair of subspaces $\mathcal{A}$ and $\mathcal{B}$ with a small principal angle have similar coherences, namely, $\theta_1(\mathcal{A},\mathcal{B})\approx 0 \implies \eta(\mathcal{A})\approx\eta(\mathcal{B})$. More concretely, Lemma 20 implies that

$$\sqrt{\eta(\widehat{\mathcal{S}}_k)} \le \sqrt{\eta(\mathcal{S})} + d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k)\sqrt{n}. \tag{106}$$

for every k. To bound the distance in the last line above, we observe that (104) holds also after replacing K with any $k\in[K_0+l:K]$, implying in particular that

$$d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k)\le d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0}),\qquad k\in[K_0+l:K]. \tag{107}$$

When $k\in[K_0+1:K_0+l-1]$ however, we cannot guarantee that the error reduces and all we can say is that that the error does not increase by much. That is, for every $k\in[K_0+1:K_0+l-1]$, we have that

$$
\begin{aligned}
d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k) &\le \nu^{k-K_0}\left(1+\frac{p^3nb}{c}\right)^{k-K_0} d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0}) && \text{(see (74))}\\
&\le \nu^{l}\left(1+\frac{p^3nb}{c}\right)^{l} d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0})\\
&= \nu^{l}\left(1+\frac{Cp^3nb}{\widetilde{\eta}\log b\log(K-K_0)}\right)^{l} d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0})\\
&\le \left(1+\frac{Cp^3nb}{\widetilde{\eta}\log b\log(K-K_0)}\right)^{2l} d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0}), && \text{(see (105))} \qquad (108)
\end{aligned}
$$

with an appropriate choice of C in (105). We continue by writing that

$$d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k)\le\left(1+\frac{Cp^3nb}{\widetilde{\eta}\log b\log(K-K_0)}\right)^{2l} d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0}) \qquad \text{(see (108))}$$

$$
\begin{aligned}
&\le \exp\left(\frac{Cp^3nb}{\widetilde{\eta}\log b\log(K-K_0)}\cdot l\right) d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0}) && (1+a\le e^a)\\
&\le e^{Cp} d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0}) && \text{(see (101))}\\
&\lesssim d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0}). && (p\le 1) \qquad (109)
\end{aligned}
$$

Combining (107) and (109), we arrive at

$$d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k)\lesssim d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0}), \tag{110}$$

for every $k\in[K_0+1:K]$, provided that (84,86,96,97) hold and except with a probability of at most $b^{-C\log(K-K_0)}$, see (95). Substituting the above bound into (106) yields that

$$
\begin{aligned}
\sqrt{\eta(\widehat{\mathcal{S}}_k)} &\le \sqrt{\eta(\mathcal{S})} + d_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_k)\sqrt{n} && \text{(see (106))}\\
&\le \sqrt{\eta(\mathcal{S})} + Cd_{\mathbb{G}}(\mathcal{S},\widehat{\mathcal{S}}_{K_0})\sqrt{n} && \text{(see (110))}\\
&\le \sqrt{\eta(\mathcal{S})} + \frac{Cp^{\frac{7}{2}}n^{\frac{3}{2}}b}{\alpha c\log b\sqrt{\log n}} && \text{(see (84))}\\
&\le \sqrt{\eta(\mathcal{S})} + \frac{Cp^{\frac{7}{2}}n^{\frac{3}{2}}b}{\alpha\widetilde{\eta}\log^2 b\sqrt{\log n}\log(K-K_0)} && \text{(see (102))}\\
&\le \sqrt{\eta(\mathcal{S})} + 1. && (\text{(97) and } \widetilde{\eta}\ge 1)\\
&\le 2\sqrt{\eta(\mathcal{S})}. && (\eta(\mathcal{S})\ge 1) \qquad (111)
\end{aligned}
$$

Plugging back the bound above into (86) yields that

$$p\gtrsim \log^2 b\log n\frac{\eta(\mathcal{S})r}{n}. \tag{112}$$

To summarize, conditioned on the event $\mathfrak{E}'$, we have established that (104) is valid under (84,96,97,112) and except with a probability of at most $b^{-C\log(K-K_0)}$. The event $\mathfrak{E}'=\cap_{k=K_0+1}^{K}\mathfrak{E}'_k$ itself holds except with a probability of at most $\sum_{k=K_0+1}^{K}\Pr[\mathfrak{E}'^C_k]$ by the union bound. With an application of the law of total probability, (104) is therefore valid except with a probability of at most $b^{-C\log(K-K_0)}+\sum_{k=K_0+1}^{K}\Pr[\mathfrak{E}'^C_k]$. This completes the proof of Theorem 10.

Acknowledgments

AE would like to thank Anand Vidyashankar and Chris Williams for separately pointing out the possibility of a statistical interpretation of SNIPE, as discussed at the end of Section 3. AE is also extremely grateful to Raphael Hauser for his helpful insights. Lastly, the authors would like to acknowledge and thank Dehui Yang for his involvement in the early phases of this project. For this project, AE was supported by the Alan Turing Institute under the EPSRC grant EP/N510129/1 and partially by the Turing Seed Funding grant SF019. GO and LB were supported by ARO Grant W911NF-14-1-0634. LB was also supported by DARPA grant 16-43-D3M-FP-037. MBW was partially supported by NSF grant CCF-1409258 and NSF CAREER grant CCF-1149225.

Appendix A. Toolbox

This section collects a number of standard results for the reader's convenience. We begin with the following large-deviation bounds that are repeatedly used in the rest of the appendices (Gross, 2011; Tropp, 2012). Throughout, C is a universal constant the value of which might change in every appearance.

Lemma 17 [Hoeffding inequality] *Let $\{z_i\}_i$ be a finite sequence of zero-mean independent random variables and assume that almost surely every z_i belongs to a compact interval of length l_i on the real line. Then, for positive α and except with a probability of at most $e^{-C\alpha^2/\sum_i l_i^2}$, it holds that $\sum_i z_i \le \alpha$.*

Lemma 18 [Matrix Bernstein inequality for spectral norm] *Let $\{Z_i\}_i \subset \mathbb{R}^{n\times b}$ be a finite sequence of zero-mean independent random matrices, and set*

$$\beta := \max_i \|Z_i\|,$$

$$\sigma^2 := \left\|\sum_i \mathbb{E}\left[Z_i^* Z_i\right]\right\| \vee \left\|\sum_i \mathbb{E}\left[Z_i Z_i^*\right]\right\|.$$

Then, for $\alpha \ge 1$ and except with a probability of at most $e^{-C\alpha}$, it holds that

$$\left\|\sum_i Z_i\right\| \lesssim \alpha \cdot \max\left(\log(n\vee b)\cdot\beta, \sqrt{\log(n\vee b)}\cdot\sigma\right).$$

For two r-dimensional subspaces $\mathcal{A}$ and $\mathcal{B}$ with principal angles $\theta_1(\mathcal{A},\mathcal{B}) \ge \theta_2(\mathcal{A},\mathcal{B}) \ge \cdots \ge \theta_r(\mathcal{A},\mathcal{B})$, recall the following useful identities about the principal angles between them:

$$\sin\left(\theta_1\left(\mathcal{A},\mathcal{B}\right)\right) = \|P_{\mathcal{A}^\perp} P_{\mathcal{B}}\| = \|P_{\mathcal{A}} - P_{\mathcal{B}}\|, \tag{113}$$

$$\sqrt{\sum_{i=1}^r \sin^2\left(\theta_i\left(\mathcal{A},\mathcal{B}\right)\right)} = \|P_{\mathcal{A}^\perp} P_B\|_F = \frac{1}{\sqrt{2}}\|P_{\mathcal{A}} - P_{\mathcal{B}}\|_F. \tag{114}$$

Note also the following perturbation bound that is slightly stronger than the standard ones, but proved similarly nonetheless (Wedin, 1972).

Lemma 19 [Perturbation bound] *Fix a rank-r matrix A and let $\mathcal{A} = \mathrm{span}(A)$. For matrix B, let B_r be a rank-r truncation of B obtained via SVD and set $\mathcal{B}_r = \mathrm{span}(B_r)$. Then, it holds that*

$$\|P_{\mathcal{A}} - P_{\mathcal{B}_r}\| = \|P_{\mathcal{A}^\perp} P_{\mathcal{B}_r}\| \le \frac{\|P_{\mathcal{A}^\perp} B\|}{\sigma_r(B)} \le \frac{\|B-A\|}{\sigma_r(B)} \le \frac{\|B-A\|}{\sigma_r(A) - \|B-A\|},$$

$$\frac{1}{\sqrt{2}}\|P_{\mathcal{A}} - P_{\mathcal{B}_r}\|_F = \|P_{\mathcal{A}^\perp} P_{\mathcal{B}_r}\|_F \le \frac{\|P_{\mathcal{A}^\perp} B\|_F}{\sigma_r(B)} \le \frac{\|B-A\|_F}{\sigma_r(B)},$$

where $\sigma_r(A)$ is the r largest singular value of A.

Proof Let $B_{r^+} := B - B_r$ denote the residual and note that

$$\begin{aligned}
\|P_{\mathcal{A}^\perp} P_{\mathcal{B}_r}\| &= \left\|P_{\mathcal{A}^\perp} B_r B_r^\dagger\right\| \qquad (\mathcal{B}_r = \mathrm{span}(B_r)) \\
&= \left\|P_{\mathcal{A}^\perp}\left(B - B_{r^+}\right) B_r^\dagger\right\| \qquad (B = B_r + B_{r^+}) \\
&= \left\|P_{\mathcal{A}^\perp} B B_r^\dagger\right\| \qquad \left(\mathrm{span}\left(B_{r^+}^*\right) \perp \mathrm{span}\left(B_r^*\right) = \mathrm{span}\left(B_r^\dagger\right)\right) \\
&\le \|P_{\mathcal{A}^\perp} B\| \cdot \|B_r^\dagger\| \\
&= \frac{\|P_{\mathcal{A}^\perp} B\|}{\sigma_r(B_r)} \\
&= \frac{\|P_{\mathcal{A}^\perp}(B-A)\|}{\sigma_r(B_r)} \\
&\le \frac{\|B-A\|}{\sigma_r(B_r)} \\
&\le \frac{\|B-A\|}{\sigma_r(A) - \|B-A\|}. \qquad \text{(Weyl's inequality)}
\end{aligned}$$

The proof is identical for the claim with the Frobenius norm and is therefore omitted. ■

Lastly, let us record what happens to the coherence of a subspace under a small perturbation, see (14).

Lemma 20 [Coherence under perturbation] *Let $\mathcal{A}, \mathcal{B}$ be two r-dimensional subspaces in $\mathbb{R}^n$, and let $d_{\mathbb{G}}(\mathcal{A},\mathcal{B})$ denote their distance, see (10). Then their coherences are related as*

$$\sqrt{\eta(\mathcal{B})} \le \sqrt{\eta(\mathcal{A})} + d_{\mathbb{G}}(\mathcal{A},\mathcal{B})\sqrt{n}.$$

Proof Let $\theta_i = \theta_i(\mathcal{A},\mathcal{B})$ be the shorthand for the ith principal angle between the subspaces $\mathcal{A}$ and $\mathcal{B}$. It is well-known (Golub and Van Loan, 2013) that there exist orthonormal bases $A, B \in \mathbb{R}^{n\times r}$ for the subspaces $\mathcal{A}$ and $\mathcal{B}$, respectively, such that

$$A^* B = \mathrm{diag}\left(\begin{bmatrix}\cos\theta_1 & \cos\theta_2 & \cdots & \cos\theta_r\end{bmatrix}\right) =: \Gamma \in \mathbb{R}^{r\times r}, \tag{115}$$

where $\mathrm{diag}(a)$ is the diagonal matrix formed from vector a. There also exists $A' \in \mathbb{R}^{n\times r}$ with orthonormal columns such that

$$\left(A'\right)^* B = \mathrm{diag}\left(\begin{bmatrix}\sin\theta_1 & \sin\theta_2 & \cdots & \sin\theta_r\end{bmatrix}\right) =: \Sigma \in \mathbb{R}^{r\times r}, \qquad \left(A'\right)^* A = 0, \tag{116}$$

and, moreover,

$$\mathrm{span}\left(\begin{bmatrix} A & B\end{bmatrix}\right) = \mathrm{span}\left(\begin{bmatrix} A & A'\end{bmatrix}\right). \tag{117}$$

With $\mathcal{A}' = \mathrm{span}(A')$, it follows that

$$B = P_{\mathcal{A}} B + P_{\mathcal{A}'} B = A A^* B + A'(A')^* B = A\Gamma + A'\Sigma. \qquad \text{(see (115) and (116))} \tag{118}$$

Consequently,

$$\sqrt{\eta(\mathcal{B})} = \sqrt{\frac{n}{r}}\max_i \|B[i,:]\|_2 \qquad \text{(see (14))}$$

$$
\begin{aligned}
&\le \sqrt{\frac{n}{r}} \max_i \|A[i,:]\cdot\Gamma\|_2 + \sqrt{\frac{n}{r}} \max_i \left\|A'[i,:]\cdot\Sigma\right\|_2 \quad \text{((118) and triangle inequality)}\\
&\le \sqrt{\frac{n}{r}} \max_i \|A[i,:]\|_2 \|\Gamma\| + \sqrt{\frac{n}{r}} \max_i \left\|A'[i,:]\right\|_2 \|\Sigma\|\\
&= \sqrt{\eta(\mathcal{A})}\,\|\Gamma\| + \sqrt{\eta(\mathcal{A}')}\,\|\Sigma\| \qquad \text{(see (14))}\\
&\le \sqrt{\eta(\mathcal{A})}\,\|\Gamma\| + \sqrt{\frac{n}{r}}\,\|\Sigma\| \qquad \left(\eta(\mathcal{A}') \le \frac{n}{r}\right)\\
&\le \sqrt{\eta(\mathcal{A})} + \sqrt{\frac{n}{r}}\sin\theta_1 \qquad \text{(see (115) and (116))}\\
&= \sqrt{\eta(\mathcal{A})} + \sqrt{\frac{n}{r}}\,\|P_{\mathcal{A}^\perp} P_{\mathcal{B}}\| \qquad \text{(see (113))}\\
&\le \sqrt{\eta(\mathcal{A})} + \sqrt{\frac{n}{r}}\,\|P_{\mathcal{A}^\perp} P_{\mathcal{B}}\|_F\\
&= \sqrt{\eta(\mathcal{A})} + d_{\mathbb{G}}(\mathcal{A},\mathcal{B})\sqrt{n}, \qquad \text{(see (10))} \qquad (119)
\end{aligned}
$$

which completes the proof of Lemma 20. ■

Appendix B. Supplement to Section 3

In this section, we verify that

$$
R_k = \begin{cases} \arg\min \ \ \|P_{\widehat{\mathcal{S}}_{k-1}^\perp} X_k\|_F^2 + \lambda \|P_{\Omega_k^C}(X_k)\|_F^2 \\ P_{\Omega_k}(X_k) = Y_k, \end{cases} \qquad (120)
$$

when $k \ge 2$. The optimization above is over $X_k \in \mathbb{R}^{n\times b}$. First note that Program (120) is separable and equivalent to the following b programs:

$$
R_k[:,j] = \begin{cases} \arg\min \ \ \|P_{\widehat{\mathcal{S}}_{k-1}^\perp} x\|_2^2 + \lambda \|P_{\omega_t^C} x\|_F^2 \\ P_{\omega_t}\cdot x = y_t, \end{cases} \qquad t = (k-1)b + j, \ \ j \in [1:b]. \qquad (121)
$$

Above, $R_k[:,j] \in \mathbb{R}^n$ is the jth column of R_k in MATLAB's matrix notation and the optimization is over $x \in \mathbb{R}^n$. To solve the jth program in (121), we make the change of variables $x = y_t + W_{\omega_t^C}\cdot z$. Here, $z \in \mathbb{R}^{n-m}$ and $W_{\omega_t^C} \in \{0,1\}^{n\times(n-m)}$ is defined naturally so that $P_{\omega_t^C} = W_{\omega_t^C} W_{\omega_t^C}^*$. We now rewrite (121) as the following b unconstrained programs:

$$
\begin{aligned}
z_j &:= \arg\min \ \ \|P_{\widehat{\mathcal{S}}_{k-1}^\perp} y_t + P_{\widehat{\mathcal{S}}_{k-1}^\perp} W_{\omega_t^C} z\|_2^2 + \lambda \|P_{\omega_t^C} W_{\omega_t^C} z\|_F^2\\
&= \arg\min \ \ \|(\widehat{S}_{k-1}^\perp)^* y_t + (\widehat{S}_{k-1}^\perp)^* W_{\omega_t^C} z\|_2^2 + \lambda\|z\|_F^2, \qquad t = (k-1)b + j, \ \ j \in [1:b].
\end{aligned} \qquad (122)
$$

Above, $\widehat{S}_{k-1}^\perp \in \mathbb{R}^{n\times(n-r)}$ is an orthonormal basis for the subspace $\widehat{\mathcal{S}}_{k-1}^\perp \in \mathbb{G}(n, n-r)$ and in particular $P_{\widehat{\mathcal{S}}_{k-1}^\perp} = \widehat{S}_{k-1}^\perp (\widehat{S}_{k-1}^\perp)^*$. The optimization above is over $z \in \mathbb{R}^{n-m}$. Note that

$$
z_j = -\left(W_{\omega_t^C}^* P_{\widehat{\mathcal{S}}_{k-1}^\perp} W_{\omega_t^C} + \lambda I_{n-m}\right)^{-1} W_{\omega_t^C}^* P_{\widehat{\mathcal{S}}_{k-1}^\perp} y_t, \qquad j \in [1:b], \qquad (123)
$$

are solutions of the least squares programs in (122) when m is large enough. For fixed j, we simplify the expression for z_j as follows:

$$
\begin{aligned}
z_j &= -\left(\lambda I_{n-m} + W_{\omega_t^C}^* P_{\widehat{\mathcal{S}}_{k-1}^\perp} W_{\omega_t^C}\right)^{-1} W_{\omega_t^C}^* P_{\widehat{\mathcal{S}}_{k-1}^\perp} y_t\\
&= -\left((1+\lambda) I_{n-m} - W_{\omega_t^C}^* P_{\widehat{\mathcal{S}}_{k-1}} W_{\omega_t^C}\right)^{-1} W_{\omega_t^C}^* P_{\widehat{\mathcal{S}}_{k-1}^\perp} y_t \qquad \left(P_{\widehat{\mathcal{S}}_{k-1}^\perp} = I_n - P_{\widehat{\mathcal{S}}_{k-1}}\right)\\
&= \frac{-1}{1+\lambda}\left(I_{n-m} + W_{\omega_t^C}^* \widehat{S}_{k-1}\left((1+\lambda) I_r - \widehat{S}_{k-1}^* P_{\omega_t^C} \widehat{S}_{k-1}\right)^{-1} \widehat{S}_{k-1}^* W_{\omega_t^C}\right) W_{\omega_t^C}^* P_{\widehat{\mathcal{S}}_{k-1}^\perp} y_t\\
&\qquad \text{(inversion lemma)}\\
&= \frac{-1}{1+\lambda}\left(I_{n-m} + W_{\omega_t^C}^* \widehat{S}_{k-1}\left((1+\lambda) I_r - \widehat{S}_{k-1}^* P_{\omega_t^C} \widehat{S}_{k-1}\right)^{-1} \widehat{S}_{k-1}^* W_{\omega_t^C}\right) W_{\omega_t^C}^* P_{\widehat{\mathcal{S}}_{k-1}^\perp} P_{\omega_t} y_t\\
&\qquad (y_t = P_{\omega_t} y_t)\\
&= \frac{1}{1+\lambda}\left(I_{n-m} + W_{\omega_t^C}^* \widehat{S}_{k-1}\left((1+\lambda) I_r - \widehat{S}_{k-1}^* P_{\omega_t^C} \widehat{S}_{k-1}\right)^{-1} \widehat{S}_{k-1}^* W_{\omega_t^C}\right) W_{\omega_t^C}^* P_{\widehat{\mathcal{S}}_{k-1}} y_t\\
&\qquad \left(W_{\omega_t^C} P_{\omega_t} = 0\right)\\
&= \frac{W_{\omega_t^C} P_{\widehat{\mathcal{S}}_{k-1}} y_t}{1+\lambda} + \frac{W_{\omega_t^C}^* \widehat{S}_{k-1}}{1+\lambda}\left((1+\lambda) I_r - \widehat{S}_{k-1}^* P_{\omega_t^C} \widehat{S}_{k-1}\right)^{-1} \widehat{S}_{k-1}^* P_{\omega_t^C} P_{\widehat{\mathcal{S}}_{k-1}} y_t\\
&\qquad \left(P_{\omega_t^C} = W_{\omega_t^C} W_{\omega_t^C}^*\right)\\
&= \frac{W_{\omega_t^C} P_{\widehat{\mathcal{S}}_{k-1}} y_t}{1+\lambda}\\
&\quad + \frac{W_{\omega_t^C}^* \widehat{S}_{k-1}}{1+\lambda}\left((1+\lambda) I_r - \widehat{S}_{k-1}^* P_{\omega_t^C} \widehat{S}_{k-1}\right)^{-1}\left(\widehat{S}_{k-1}^* P_{\omega_t^C} \widehat{S}_{k-1} - (1+\lambda) I_r\right) \widehat{S}_{k-1}^* y_t\\
&\quad + W_{\omega_t^C}^* \widehat{S}_{k-1}\left((1+\lambda) I_r - \widehat{S}_{k-1}^* P_{\omega_t^C} \widehat{S}_{k-1}\right)^{-1} \widehat{S}_{k-1}^* y_t\\
&= W_{\omega_t^C}^* \widehat{S}_{k-1}\left((1+\lambda) I_r - \widehat{S}_{k-1}^* P_{\omega_t^C} \widehat{S}_{k-1}\right)^{-1} \widehat{S}_{k-1}^* y_t\\
&= W_{\omega_t^C}^* \widehat{S}_{k-1}\left(\lambda I_r + \widehat{S}_{k-1}^* P_{\omega_t} \widehat{S}_{k-1}\right)^{-1} \widehat{S}_{k-1}^* y_t, \qquad (124)
\end{aligned}
$$

which means that

$$
y_t + W_{\omega_t^C} z_j = y_t + P_{\omega_t^C} \widehat{S}_{k-1}\left(\lambda I_r + \widehat{S}_{k-1}^* P_{\omega_t} \widehat{S}_{k-1}\right)^{-1} \widehat{S}_{k-1}^* y_t, \qquad (125)
$$

is a solution of the jth program in (121) which indeed matches the jth column of R_k defined in (1).

Appendix C. Proof of Proposition 24

Let us form the blocks $Q_1 \in \mathbb{R}^{b_1\times r}$, $S_1 \in \mathbb{R}^{n\times b_1}$, and $\Omega_1 \subseteq [1:n]\times[1:b_1]$ as usual, see Section 3. As in that section, we also write the measurement process as $Y_1 = P_{\Omega_1}(S_1)$, where $P_{\Omega_1}(\cdot)$ projects onto the index set Ω_1. Let us fix Q_1 for now. Also let $Y_{1,r} \in \mathbb{R}^{n\times b_1}$ be a

rank-r truncation of Y_1 obtained via SVD. SNIPE then sets $\widehat{\mathcal{S}}_1 = \mathrm{span}(Y_{1,r})$. Our objective here is to control $\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_1}\|_F$. Since

$$\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_1}\|_F \le \sqrt{r}\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_1}\|, \qquad \left(\widehat{\mathcal{S}}_1 \in \mathbb{G}(n,r)\right) \tag{126}$$

it suffices to bound the spectral norm. Conditioned on Q_1, it is easy to verify that $\mathbb{E}[Y_1] = \mathbb{E}[P_{\Omega_1}(S_1)] = p \cdot S_1$, suggesting that we might consider Y_1 as a perturbed copy of $p \cdot S_1$ and perhaps consider $\widehat{\mathcal{S}}_1 = \mathrm{span}(Y_{1,r})$ as a perturbation of $\mathcal{S} = \mathrm{span}(p \cdot S_1)$. Indeed, the perturbation bound in Lemma 19 dictates that

$$\begin{aligned}
\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_1}\| &\le \frac{\|Y_1 - pS_1\|}{p \cdot \sigma_r(S_1) - \|Y_1 - pS_1\|} \\
&= \frac{\|Y_1 - pS_1\|}{p \cdot \sigma_r(Q_1) - \|Y_1 - pS_1\|} \qquad (S_1 = SQ_1,\ S^*S = I_r) \\
&\le \frac{2}{p} \cdot \frac{\|Y_1 - pS_1\|}{\sigma_r(Q_1)}. \qquad \left(\text{if } \|Y_1 - pS_{Q_1}\| \le \frac{p}{2} \cdot \sigma_r(Q_1)\right)
\end{aligned} \tag{127}$$

It remains to bound the norm in the last line above. To that end, we study the concentration of Y_1 about its expectation by writing that

$$Y_1 - pS_1 = P_{\Omega_1}(S_1) - pS_1 = \sum_{i,j} (\epsilon_{i,j} - p)\, S_1[i,j] \cdot E_{i,j} =: \sum_{i,j} Z_{i,j}, \tag{128}$$

where $\{\epsilon_{i,j}\}_{i,j} \overset{\text{ind.}}{\sim} \text{Bernoulli}(p)$ and $E_{i,j} \in \mathbb{R}^{n \times b_1}$ is the $[i,j]$th canonical matrix. Additionally, $\{Z_{i,j}\}_{i,j}$ are independent zero-mean random matrices. In order to appeal to the matrix Bernstein inequality (Lemma 18), we compute the β and σ parameters below, starting with β:

$$\begin{aligned}
\|Z_{i,j}\| &= \|(\epsilon_{i,j} - p)\, S_1[i,j] \cdot E_{i,j}\| \\
&= |(\epsilon_{i,j} - p)\, S_1[i,j]| \qquad (\|E_{i,j}\| = 1) \\
&\le |S_1[i,j]| \qquad (\epsilon_{i,j} \in \{0,1\}) \\
&\le \|S_1\|_\infty \\
&\le \|S\|_{2\to\infty} \|Q_1\|_{2\to\infty} \qquad (S_1 = SQ_1,\ \|AB^*\|_\infty \le \|A\|_{2\to\infty}\|B\|_{2\to\infty}) \\
&\le \sqrt{\frac{\eta(\mathcal{S})\, r}{n}} \cdot \sqrt{\frac{\eta(\mathcal{Q}_1)\, r}{b_1}} \cdot \|Q_1\| \qquad (\text{see (14)}) \\
&=: \beta.
\end{aligned} \tag{129}$$

Above, $\|A\|_\infty$ and $\|A\|_{2\to\infty}$ return the largest entry of A in magnitude and the largest ℓ_2 norm of the rows of matrix A, respectively. As for σ, we write that

$$\begin{aligned}
\left\|\mathbb{E}\left[\sum_{i,j} Z_{i,j} Z_{i,j}^*\right]\right\| &= \left\|\sum_{i,j} \mathbb{E}\left[(\epsilon_{i,j} - p)^2\right] S_1[i,j]^2 \cdot E_{i,i}\right\| \\
&= \left\|\sum_{i,j} p(1-p) S_1[i,j]^2 \cdot E_{i,i}\right\| \qquad (\epsilon_{i,j} \sim \text{Bernoulli}(p))
\end{aligned}$$

$$\begin{aligned}
&\le p \left\|\sum_{i,j} S_1[i,j]^2 \cdot E_{i,j}\right\| \\
&= p \left\|\sum_{i} \|S_1[i,:]\|_2^2 \cdot E_{i,i}\right\| \\
&= p \max_i \|S_1[i,:]\|_2^2 \\
&= p\, \|S_1\|_{2\to\infty}^2 \\
&\le p\, \|S\|_{2\to\infty}^2 \cdot \|Q_1\|^2 \qquad (S_1 = SQ_1,\ \|AB\|_{2\to\infty} \le \|A\|_{2\to\infty}\|B\|) \\
&= p \cdot \frac{\eta(\mathcal{S})\, r}{n} \cdot \|Q_1\|^2. \qquad (\text{see (14)})
\end{aligned} \tag{130}$$

In a similar fashion, we find that

$$\begin{aligned}
\left\|\mathbb{E}\left[\sum_{i,j} Z_{i,j}^* Z_{i,j}\right]\right\| &\le p \left\|\sum_{j} \|S_1[:,j]\|_2^2\, E_{j,j}\right\| \\
&= p\, \|S_1^*\|_{2\to\infty}^2 \\
&\le p \cdot \|S\|^2 \cdot \|Q_1\|_{2\to\infty}^2 \qquad (S_1 = SQ_1,\ \|AB\|_{2\to\infty} \le \|A\|_{2\to\infty}\|B\|) \\
&\le p \cdot \frac{\eta(\mathcal{Q}_1)\, r}{b_1} \cdot \|Q_1\|^2, \qquad (\|S\| = 1,\ \text{see (14)})
\end{aligned} \tag{131}$$

and eventually

$$\begin{aligned}
\sigma^2 &= \left\|\mathbb{E}\left[\sum_{i,j} Z_{i,j}^* Z_{i,j}\right]\right\| \vee \left\|\mathbb{E}\left[\sum_{i,j} Z_{i,j} Z_{i,j}^*\right]\right\| \\
&\le \frac{pr}{n}\left(1 \vee \frac{n}{b_1}\right)\left(\eta(\mathcal{S}) \vee \eta(\mathcal{Q}_1)\right)\|Q_1\|^2. \qquad (\text{see (130) and (131)})
\end{aligned} \tag{132}$$

Lastly,

$$\begin{aligned}
&\max\left(\log(n \vee b_1) \cdot \beta, \sqrt{\log(n \vee b_1)} \cdot \sigma\right) \\
&\lesssim \max\left(\log(n \vee b_1) \cdot \frac{r}{n}, \sqrt{\log(n \vee b_1)} \cdot \sqrt{\frac{pr}{n}}\right) \sqrt{1 \vee \frac{n}{b_1}} \cdot \sqrt{\eta(\mathcal{S}) \vee \eta(\mathcal{Q}_1)} \cdot \|Q_1\| \\
&\qquad (\text{see (129) and (132)}) \\
&\le \sqrt{\log(n \vee b_1)} \cdot \sqrt{\frac{pr}{n}} \sqrt{1 \vee \frac{n}{b_1}} \cdot \sqrt{\eta(\mathcal{S}) \vee \eta(\mathcal{Q}_1)} \cdot \|Q_1\|. \qquad \left(\text{if } p \ge \frac{\log(n \vee b_1)\, r}{n}\right)
\end{aligned} \tag{133}$$

The Bernstein inequality now dictates that

$$\|Y_1 - pS_1\| = \left\|\sum_{i,j} Z_{i,j}\right\| \qquad (\text{see (128)})$$

$$\lesssim \alpha \max\left(\log(n \vee b_1)\cdot\beta, \sqrt{\log(n\vee b_1)}\cdot\sigma\right) \qquad \text{(see Lemma 18)}$$
$$\lesssim \alpha\sqrt{\log(n\vee b_1)}\cdot\sqrt{\frac{rp}{n}}\sqrt{1\vee\frac{n}{b_1}}\cdot\sqrt{\eta(\mathcal{S})\vee\eta(\mathcal{Q}_1)}\cdot\|Q_1\|, \qquad \text{(see (133))} \tag{134}$$

except with a probability of at most $e^{-\alpha}$. In particular, suppose that

$$p \gtrsim \alpha^2\nu(Q_1)^2\left(1\vee\frac{n}{b_1}\right)\frac{(\eta(\mathcal{S})\vee\eta(\mathcal{Q}_1))\, r\log(n\vee b_1)}{n}, \qquad \left(\nu(Q_1)=\frac{\|Q_1\|}{\sigma_r(Q_1)}\right) \tag{135}$$

so that (127) holds. Then, by substituting (134) back into (127) and then applying (126), we find that

$$\begin{aligned}\frac{\|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_1}\|_F}{\sqrt{r}} &\le \|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_1}\| \qquad \text{(see (126))}\\ &\le \frac{2}{p}\cdot\frac{\|Y_1-pS_1\|}{\sigma_r(Q_1)} \qquad \text{(see (127))}\\ &\lesssim \alpha\sqrt{\log(n\vee b_1)\cdot\frac{r}{pn}\left(1\vee\frac{n}{b_1}\right)(\eta(\mathcal{S})\vee\eta(\mathcal{Q}_1))}\cdot\nu(Q_1) \qquad \text{(see (134))}\\ &=: \delta_1(\nu(Q_1),\eta(Q_1)),\end{aligned} \tag{136}$$

except with a probability of at most $e^{-\alpha}$ and for fixed Q_1. In order to remove the conditioning on Q_1, fix $\nu\ge 1$, $1\le\eta_1\le\frac{b_1}{r}$, and recall the following inequality for events $\mathcal{A}$ and $\mathcal{B}$:

$$\Pr[\mathcal{A}] = \Pr[\mathcal{A}|\mathcal{B}]\cdot\Pr[\mathcal{B}] + \Pr\left[\mathcal{A}|\mathcal{B}^C\right]\cdot\Pr\left[\mathcal{B}^C\right] \le \Pr[\mathcal{A}|\mathcal{B}] + \Pr\left[\mathcal{B}^C\right]. \tag{137}$$

Set $\mathcal{Q}_1=\operatorname{span}(Q_1^*)$ and let $\mathcal{E}$ be the event where both $\nu(Q_1)\le\nu$ and $\eta(\mathcal{Q}_1)\le\eta_1$. Thanks to the inequality above, we find that

$$\begin{aligned}&\Pr\left[\frac{\|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_1}\|_F}{\sqrt{r}}\gtrsim\delta_1(\nu,\eta_1)\right]\\ &\le \Pr\left[\frac{\|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_1}\|_F}{\sqrt{r}}\gtrsim\delta(\nu,\eta_1)\mid\mathcal{E}\right]+\Pr\left[\mathcal{E}^C\right] \qquad \text{(see (137))}\\ &\le e^{-\alpha}+\Pr[\nu(Q_1)>\nu]+\Pr[\eta(Q_1)>\eta_1], \qquad \text{(see (136))}\end{aligned} \tag{138}$$

which completes the proof of Proposition 24.

Appendix D. Proof of Lemma 13

We conveniently define the orthonormal basis

$$B=\begin{bmatrix} U & U^\perp\end{bmatrix}\in\mathbb{R}^{n\times n}.$$

Then the perturbation from $\mathcal{U}$ can be written more compactly as

$$U+U^\perp\Delta' = B\begin{bmatrix} I_r\\ \Delta'\end{bmatrix}\in\mathbb{R}^{n\times r}.$$

In particular, orthogonal projection onto $\operatorname{span}(U+U^\perp\Delta')$ is

$$\begin{aligned}(U+U^\perp\Delta')(U+U^\perp\Delta')^\dagger &= B\begin{bmatrix} I_r\\ \Delta'\end{bmatrix}\cdot\begin{bmatrix} I_r\\ \Delta'\end{bmatrix}^\dagger B^*\\ &= B\begin{bmatrix} I_r\\ \Delta'\end{bmatrix}\left(I_r+\Delta'^*\Delta'\right)^{-1}\begin{bmatrix} I_r & \Delta'^*\end{bmatrix}O^*\\ &= B\begin{bmatrix} I_r & \Delta'^*\\ \Delta' & 0_{n-r}\end{bmatrix}B^* + o(\|\Delta'\|_F),\end{aligned} \tag{139}$$

where $0_{n-r}\in\mathbb{R}^{(n-r)\times(n-r)}$ is the matrix of zeros of size $n-r$. From (30), note also that

$$f_\Omega(X,\mathcal{U}) = \langle P_{\mathcal{U}^\perp}, XX^*\rangle + \lambda\|P_{\Omega^C}(X)\|_F^2 = \left\langle I_n - UU^\dagger, XX^*\right\rangle + \lambda\|P_{\Omega^C}(X)\|_F^2.$$

We can now write that

$$\begin{aligned}&f_\Omega\left(X+\Delta, U+U^\perp\Delta'\right)\\ &= \left\langle I_n - B\begin{bmatrix} I_r & \Delta'^*\\ \Delta' & 0_{n-r}\end{bmatrix}B^*, (X+\Delta)(X+\Delta)^*\right\rangle + 2\lambda P_{\Omega^C}(X) + o(\|\Delta\|_F) + o(\|\Delta'\|_F)\\ &\qquad \text{(see (139))}\\ &= \left\langle B\begin{bmatrix} 0_r & -\Delta'^*\\ -\Delta' & I_{n-r}\end{bmatrix}B^*, (X+\Delta)(X+\Delta)^*\right\rangle + 2\lambda P_{\Omega^C}(X) + o(\|\Delta\|_F) + o(\|\Delta'\|_F)\\ &= \left\langle \begin{bmatrix} 0_r & -\Delta'^*\\ -\Delta' & I_{n-r}\end{bmatrix}, B^*\left(XX^*+\Delta X^*+X\Delta^*\right)B\right\rangle + 2\lambda P_{\Omega^C}(X) + o(\|\Delta\|_F) + o(\|\Delta'\|_F)\\ &= \left\langle \begin{bmatrix} 0_r & 0_{r\times(n-r)}\\ 0_{(n-r)\times r} & I_{n-r}\end{bmatrix}, B^*XX^*B\right\rangle\\ &\quad + \left\langle \begin{bmatrix} 0_r & 0_{r\times(n-r)}\\ 0_{(n-r)\times r} & I_{n-r}\end{bmatrix}, B^*\Delta X^*B + B^*X\Delta^*B\right\rangle\\ &\quad - \left\langle \begin{bmatrix} 0_r & \Delta'^*\\ \Delta' & 0_{n-r}\end{bmatrix}, B^*XX^*B\right\rangle + 2\lambda P_{\Omega^C}(X) + o(\|\Delta\|_F) + o(\|\Delta'\|_F).\end{aligned} \tag{140}$$

We can further simplify the above expansion as

$$\begin{aligned}&f_\Omega\left(X+\Delta, U+U^\perp\Delta'\right)\\ &= f_\Omega(X,\mathcal{U}) + 2\langle\Delta, P_{\mathcal{U}^\perp}X\rangle - 2\left\langle\Delta', (U^\perp)^*XX^*U\right\rangle + 2\lambda P_{\Omega^C}(X) + o(\|\Delta\|_F) + o(\|\Delta'\|_F).\end{aligned} \tag{141}$$

Therefore the partial derivatives of f are

$$\begin{aligned}\partial_X f_\Omega(X,\mathcal{U}) &= 2P_{\mathcal{U}^\perp}X + 2\lambda P_{\Omega^C}(X)\in\mathbb{R}^{n\times T},\\ \partial_{\mathcal{U}} f_\Omega(X,\mathcal{U}) &= -2(U^\perp)^*XX^*U\in\mathbb{R}^{(n-r)\times r},\end{aligned} \tag{142}$$

which completes the proof of Lemma 13.

Appendix E. Proof of Lemma 14

By definition in (39), $\{R_{\epsilon'}\}_{\epsilon' \le \epsilon}$ is a bounded set, see also Program (30) for the definition of $f_{\widehat{\Omega}}$. Therefore there exist a subsequence $\{R_{\epsilon_i}\}_i$ and a matrix $R \in \mathbb{R}^{n \times b}$ such that

$$\lim_{i \to \infty} \epsilon_i = 0, \tag{143}$$

$$\lim_{i \to \infty} \|R_{\epsilon_i} - R\|_F = 0. \tag{144}$$

That is, $\{R_{\epsilon_i}\}_i$ converges to R. On the other hand, for every $\delta \ge 0$, (49) implies that there exists an integer l_δ that depneds on δ and

$$\|R_{k_l,\epsilon} - R_\epsilon\|_F \le 2\epsilon l' + \delta, \qquad l \ge l_\delta. \tag{145}$$

Restricted to the sequence $\{\epsilon_i\}_i$ above, (145) reads as

$$\|R_{k_l,\epsilon_i} - R_{\epsilon_i}\|_F \le 2\epsilon_i l' + \delta, \qquad l \ge l_\delta, \tag{146}$$

which, in the limit of $i \to \infty$, yields that

$$\lim_{i \to \infty} \|R_{k_l,\epsilon_i} - R_{\epsilon_i}\|_F \le \lim_{i \to \infty} 2\epsilon_i + \delta = \delta, \qquad l \ge l_\delta. \tag{147}$$

We used (143) to obtain the identity above. An immediate consequence of (147) is that

$$\lim_{l \to \infty} \lim_{i \to \infty} \|R_{k_l,\epsilon_i} - R_{\epsilon_i}\|_F = 0. \tag{148}$$

Invoking (144), it then follows that

$$\begin{aligned} \lim_{l \to \infty} \lim_{i \to \infty} \|R_{k_l,\epsilon_i} - R\|_F &= \lim_{l \to \infty} \lim_{i \to \infty} \|R_{k_l,\epsilon_i} - R_{\epsilon_i}\|_F \qquad \text{(see (144))} \\ &= 0. \qquad \text{(see (148))} \end{aligned} \tag{149}$$

Exchanging the order of limits above yields that

$$\begin{aligned} &\lim_{i \to \infty} \lim_{l \to \infty} \|R_{k_l,\epsilon_i} - R\|_F \\ &\le \lim_{i \to \infty} \lim_{l \to \infty} \|R_{k_l,\epsilon_i} - R_{\epsilon_i}\|_F + \lim_{i \to \infty} \|R_{\epsilon_i} - R\|_F \qquad \text{(triangle inequality)} \\ &= \lim_{i \to \infty} \lim_{l \to \infty} \|R_{k_l,\epsilon_i} - R_{\epsilon_i}\|_F \qquad \text{(see (144))} \\ &= 0. \qquad \text{(see (50))} \end{aligned} \tag{150}$$

Therefore, (149) and (150) together state that R_{k,ϵ_i} converges to R as $l, i \to \infty$, namely,

$$\lim_{l,i \to \infty} \|R_{k_l,\epsilon_i} - R\|_F = 0, \tag{151}$$

thereby proving the first claim in Lemma 14, see (51). In order to prove the claim about subspaces in Lemma 14, we proceed as follows. Recall the output of SNIPE, namely, $\{(R_k, \widehat{\mathcal{S}}_k)\}_k$ constructed in Algorithm 1. In light of Section 3, R_k is also the unique minimizer of Program

(9). Recall that $\kappa_l = [k_l - l + 1 : k_l]$ from (33). Recall also the construction of sequence $\{(R_{k,\epsilon}, \widehat{\mathcal{S}}_{k,\epsilon})\}_{k \in \kappa_l}$ in the beginning of Section 7.1 and note that both procedures are initialized identically at the beginning of interval κ_l, namely, $\widehat{\mathcal{S}}_{k_l - l,\epsilon} = \widehat{\mathcal{S}}_{k_l - 1}$. Therefore, for fixed l, observe that[4]

$$\lim_{\epsilon \to 0} \|R_{k,\epsilon} - R_k\|_F = 0, \qquad k \in \kappa_l, \tag{152}$$

which, when restricted to $\{\epsilon_i\}_i$, reads as

$$\lim_{i \to \infty} \|R_{k,\epsilon_i} - R_k\|_F = 0, \qquad k \in \kappa_l. \tag{153}$$

By design, every R_k has a spectral gap in the sense that there exists $\tau > 0$ such that

$$\frac{\sigma_r(R_k)}{\sigma_{r+1}(R_k)} \ge 1 + \tau, \tag{154}$$

for every k. Recall that $\widehat{\mathcal{S}}_k$ is the span of top r left singular vectors of R_k. An immediate consequence of (154) is that $\widehat{\mathcal{S}}_k$ is uniquely defined, namely, there are no ties in the spectrum of R_k. By (153), there are no ties in the spectrum of R_{k,ϵ_i} as well for sufficiently large i, namely,

$$\lim_{i \to \infty} \frac{\sigma_r(R_{k,\epsilon_i})}{\sigma_{r+1}(R_{k,\epsilon_i})} \ge 1 + \tau, \qquad k \in \kappa_l. \tag{155}$$

By sending l to infinity above, we find that

$$\begin{aligned} 1 + \tau &\le \lim_{l \to \infty} \lim_{i \to \infty} \frac{\sigma_r(R_{k_l,\epsilon_i})}{\sigma_{r+1}(R_{k_l,\epsilon_i})} \qquad \text{(see (155))} \\ &= \lim_{l,i \to \infty} \frac{\sigma_r(R_{k_l,\epsilon_i})}{\sigma_{r+1}(R_{k_l,\epsilon_i})} \qquad \text{(see (151))} \\ &= \frac{\sigma_r(R)}{\sigma_{r+1}(R)}. \qquad \text{(see (151))} \end{aligned} \tag{156}$$

Recall that $\widehat{\mathcal{S}}_{k,\epsilon_i}$ is by definition the span of leading r left singular vectors of R_{k,ϵ_i}. Likewise, let $\widehat{\mathcal{S}}$ be the span of leading r left singular vectors of R. An immediate consequence of the second line of (156) is that $\widehat{\mathcal{S}}_{k_l,\epsilon_i}$ is uniquely defined, namely, no ties in the spectrum of R_{k_l,ϵ_i} in the limit of $l, i \to \infty$. The third line of (156) similarly implies that $\widehat{\mathcal{S}}$ is uniquely defined. Given the uniqueness of these subspaces, another implication of (151) is that

$$\lim_{l,i \to \infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_{k_l,\epsilon_i}, \widehat{\mathcal{S}}) = 0, \tag{157}$$

where $d_{\mathbb{G}}$ is the metric on Grassmannian defined in (10). Lastly we show that SNIPE in the limit produces copies of $(R, \widehat{\mathcal{S}})$ on the interval $\kappa_{l,l'}$. This is done by simply noting that

$$\lim_{l \to \infty} \|R_{k_l} - R\|_F = \lim_{l \to \infty} \lim_{i \to \infty} \|R_{k_l,\epsilon_i} - R\|_F \qquad \text{(see (153))}$$

4. To verify (152), note that for every feasible X_ϵ in Program (38), $X = Y + P_{\widehat{\Omega}^C}(X_\epsilon)$ is feasible for Program (37). Moreover, as $\epsilon \to 0$, $\|X_\epsilon - X\|_F \to 0$ and consequently, by continuity, $|f_{\widehat{\Omega}}(X_\epsilon, \mathcal{U}) - f_{\widehat{\Omega}}(X, \mathcal{U})| \to 0$. On the other hand, by definition, R_k is the unique minimizer of Program (37), namely, $f_{\widehat{\Omega}}(R_k, \mathcal{U}) < f_{\widehat{\Omega}}(X, \mathcal{U})$ for any $X \neq R_k$ feasible for Program (37). For sufficiently small ϵ, it follows that $f_{\widehat{\Omega}}(R_k, \mathcal{U}) < f_{\widehat{\Omega}}(X_\epsilon, \mathcal{U})$ for any X_ϵ that is feasible for Program (38) and $\liminf_{\epsilon \to 0} \|X_\epsilon - R_k\| > 0$. That is, the unique minimizer of Program (38) approaches R_k in the limit, namely, $\lim_{\epsilon \to 0} \|R_{k,\epsilon} - R_k\|_F = 0$.

$$
\begin{aligned}
&= \lim_{l,i\to\infty} \|R_{k_l,\epsilon_i} - R\|_F \qquad \text{(see (151))} \\
&= 0, \qquad \text{(see (151))}
\end{aligned} \tag{158}
$$

$$
\begin{aligned}
\lim_{l\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_{k_l}, \widehat{\mathcal{S}}) &= \lim_{l\to\infty} \lim_{i\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_{k_l,\epsilon_i}, \widehat{\mathcal{S}}) \qquad \text{(see (153))} \\
&= \lim_{l,i\to\infty} \lim_{i\to\infty} d_{\mathbb{G}}(\widehat{\mathcal{S}}_{k_l,\epsilon_i}, \widehat{\mathcal{S}}) \qquad \text{(see (157))} \\
&= 0, \qquad \text{(see (157))}
\end{aligned} \tag{159}
$$

which completes the proof of Lemma 14.

Appendix F. Proof of Lemma 15

By restricting (41) to $\{\epsilon_i\}_i$, we find that

$$
\begin{aligned}
0 &= \partial_{\mathcal{U}} f_{\widehat{\Omega}}(R_{k_l,\epsilon_i}, \widehat{\mathcal{S}}_{k,\epsilon_i}) \qquad \text{(see (41))} \\
&= \partial_{\mathcal{U}} f_{\Omega_{k_l}}(R_{k_l,\epsilon_i}, \widehat{\mathcal{S}}_{k,\epsilon_i}), \qquad \text{(see (34))}
\end{aligned} \tag{160}
$$

for every l, i. By the joint continuity of $\partial_{\mathcal{U}} f_{\Omega_{k_l}}$ in Lemma 13, it follows that

$$
\begin{aligned}
0 &= \lim_{l,i\to\infty} \left\| \partial_{\mathcal{U}} f_{\Omega_{k_l}}(R_{k_l,\epsilon_i}, \widehat{\mathcal{S}}_{k_l,\epsilon_i}) \right\|_F \\
&= \lim_{l\to\infty} \left\| \partial_{\mathcal{U}} f_{\Omega_{k_l}}(R, \widehat{\mathcal{S}}) \right\|_F, \qquad \text{(see Lemma 14)}
\end{aligned} \tag{161}
$$

which establishes (56). To establish (57), we restrict (44) to $\{\epsilon_i\}_i$ and then send i, l to infinity to find that

$$
\begin{aligned}
0 &= \lim_{i\to\infty} \lim_{l\to\infty} \|P_{\widehat{\Omega}}(R_{k_l,\epsilon_i}) - \widehat{Y}\|_F \qquad \text{(see (44))} \\
&= \lim_{i\to\infty} \lim_{l\to\infty} \|P_{\Omega_k}(R_{k_l,\epsilon_i}) - Y_{k_l}\|_F \qquad \text{(see (34,36))} \\
&= \lim_{l\to\infty} \|P_{\Omega_{k_l}}(R) - Y_{k_l}\|_F. \qquad \text{(see Lemma 14)}
\end{aligned} \tag{162}
$$

To establish (58), we restrict (45) to $\{\epsilon_i\}_i$ and then send i to infinity to find that

$$
\begin{aligned}
0 &= \lim_{i\to\infty} \lim_{l\to\infty} \left\| \partial_X f_{\widehat{\Omega}}(R_{k_l,\epsilon_i}, \widehat{\mathcal{S}}_{k_l,\epsilon_i}) + \lambda_{k_l,\epsilon_i}(P_{\widehat{\Omega}}(R_{k_l,\epsilon_i}) - \widehat{Y}) \right\|_F^2 \\
&= \lim_{i\to\infty} \lim_{l\to\infty} \left\| P_{\widehat{\Omega}^C}\left(\partial_X f_{\widehat{\Omega}}(R_{k_l,\epsilon_i}, \widehat{\mathcal{S}}_{k_l,\epsilon_i}) \right) \right\|_F^2 \\
&\quad + \lim_{i\to\infty} \lim_{l\to\infty} \left\| P_{\widehat{\Omega}}\left(\partial_X f_{\widehat{\Omega}}(R_{k_l,\epsilon}, \widehat{\mathcal{S}}_{k_l,\epsilon_i}) + \lambda_{k,\epsilon_i}(R_{k_l,\epsilon_i} - \widehat{Y}) \right) \right\|_F^2 \\
&\geq \lim_{i\to\infty} \lim_{l\to\infty} \left\| P_{\widehat{\Omega}^C}\left(\partial_X f_{\widehat{\Omega}}(R_{k_l,\epsilon_i}, \widehat{\mathcal{S}}_{k_l,\epsilon_i}) \right) \right\|_F^2 \\
&= \lim_{l\to\infty} \left\| P_{\Omega_{k_l}^C}\left(\partial_X f_{\Omega_{k_l}}(R, \widehat{\mathcal{S}}) \right) \right\|_F^2. \qquad \text{(see (34) and Lemma 14)}
\end{aligned} \tag{163}
$$

This completes the proof of Lemma 15.

Appendix G. Proof of Lemma 16

Recall that by construction in Section 3, the rows of the coefficient matrix $Q_k \in \mathbb{R}^{b\times r}$ are independent copies of the random vector $q \in \mathbb{R}^r$. Setting $S_k = S \cdot Q_k^* \in \mathbb{R}^{n\times b}$, we observe in iteration k each entry of the data block S_k independently with a probability of p, collect the observed entries in $Y_k \in \mathbb{R}^{n\times b}$, supported on the index set $\Omega_k \subseteq [1:n]\times[1:b]$. We write this as $Y_k = P_{\Omega_k}(S_k)$, where the linear operator P_{Ω_k} retains the entries on the index set Ω_k and sets the rest to zero. Recall also that Q_k is obtained by concatenating the coefficient vectors $\{q_t\}_{t=(k-1)b+1}^{kb} \subset \mathbb{R}^r$. To unburden the notation, we enumerate these vectors as $\{q_j\}_{j=1}^b$. Likewise, we use the indexing $\{s_j, y_j, \omega_j\}_{j=1}^b$ for the data vectors, incomplete data vectors, and their supports, respectively.

Given the new incomplete block Y_k at iteration k, we update our estimate of the true subspace $\mathcal{S}$ from the old $\widehat{\mathcal{S}}_{k-1}$ as follows. We calculate the random matrix

$$
\mathbb{R}^{n\times b} \ni R_k := Y_k + P_{\Omega_k^C}\left(O\left(Y_k\right)\right), \tag{164}
$$

where the linear operator $P_{\Omega_k^C}$ projects onto the complement of index set Ω_k, and

$$
O(Y_k) = \left[\cdots \widehat{S}_{k-1}(P_{\omega_j}\widehat{S}_{k-1})^\dagger y_j \cdots \right] \in \mathbb{R}^{n\times b}. \tag{165}
$$

Above, $\widehat{S}_{k-1} \in \mathbb{R}^{n\times r}$ is an orthonormal basis for the r-dimensional subspace $\widehat{\mathcal{S}}_{k-1}$. If $\sigma_r(R_k) < \sigma_{\min}$, reject this iteration, see Algorithm 1. Otherwise, let $R_{k,r}$ denote a rank-r truncation of R_k obtained via SVD. Then our updated estimate is $\widehat{\mathcal{S}}_k = \text{span}(R_{k,r})$.

We condition on the subspace $\widehat{\mathcal{S}}_{k-1}$ and the coefficient matrix Q_k for now. To control the estimation error $d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_k)$, our strategy is to treat R_k as a perturbed copy of $S_k = SQ_k^*$ and in turn treat $\widehat{\mathcal{S}}_k = \text{span}(R_{k,r})$ as a perturbed copy of $\mathcal{S} = \text{span}(S_k)$. Indeed, an application of the perturbation bound in Lemma 19 yields that

$$
\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_k}\|_F \leq \frac{\|P_{\mathcal{S}^\perp} R_k\|_F}{\sigma_r\left(R_k\right)}. \tag{166}
$$

To control the numerator above, we begin with some preparation. First, recalling the definition of $O(Y_k)$ from (165), we observe that

$$
\begin{aligned}
O\left(Y_k\right) &= O\left(P_{\Omega_k}\left(S_k\right)\right) \qquad \left(Y_k = P_{\Omega_k}\left(S_k\right)\right) \\
&= O(P_{\Omega_k}(P_{\widehat{\mathcal{S}}_{k-1}} S_k)) + O(P_{\Omega_k}(P_{\widehat{\mathcal{S}}_{k-1}^\perp} S_k)) \qquad \text{(linearity of } O\text{)} \\
&= P_{\widehat{\mathcal{S}}_{k-1}} S_k + O(P_{\Omega_k}(P_{\widehat{\mathcal{S}}_{k-1}^\perp} S_k)). \qquad \text{(see (165))}
\end{aligned} \tag{167}
$$

The above decomposition allows us to rewrite R in (164) as

$$
\begin{aligned}
R_k &= Y_k + P_{\Omega_k^C}\left(O\left(Y_k\right)\right) \qquad \text{(see (164))} \\
&= P_{\Omega_k}(S_k) + P_{\Omega_k^C}\left(O\left(Y_k\right)\right) \qquad \left(Y_k = P_{\Omega_k}(S_k)\right) \\
&= P_{\Omega_k}\left(S_k\right) + P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}_{k-1}} S_k) + [P_{\Omega_k^C} \circ O \circ P_{\Omega_k}](P_{\widehat{\mathcal{S}}_{k-1}^\perp} S_k) \\
&\qquad \text{(see (167), } f \circ g(x) := f(g(x))\text{)}
\end{aligned}
$$

$$= S_k - P_{\Omega_k^C}(S_k) + P_{\Omega_k^C}(P_{\widehat{S}_{k-1}} S_k) + [P_{\Omega_k^C} \circ O \circ P_{\Omega_k}](P_{\widehat{S}_{k-1}^\perp} S_k)$$
$$= S_k - P_{\Omega_k^C}(P_{\widehat{S}_{k-1}^\perp} S_k) + [P_{\Omega_k^C} \circ O \circ P_{\Omega_k}](P_{\widehat{S}_{k-1}} S_k)$$
$$= S_k - P_{\Omega_k^C}(P_{\widehat{S}_{k-1}^\perp} S_k) - [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}} S_k\right) + [O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}^\perp} S_k\right). \tag{168}$$

Since $P_{\mathcal{S}^\perp} S_k = P_{\mathcal{S}^\perp} S Q_k^* = 0$, it immediately follows that

$$P_{\mathcal{S}^\perp} R_k = -P_{\mathcal{S}^\perp} \cdot P_{\Omega_k^C}(P_{\widehat{S}_{k-1}^\perp} S_k) - P_{\mathcal{S}^\perp} \cdot [P_{\Omega_k} \circ O \circ P_{\Omega_k}](P_{\widehat{S}_{k-1}^\perp} S_k) + P_{\mathcal{S}^\perp} \cdot [O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}^\perp} S_k\right).$$

In particular, with an application of the triangle inequality above, we find that

$$\|P_{\mathcal{S}^\perp} R_k\|_F$$
$$\leq \left\|P_{\Omega_k^C}(P_{\widehat{S}_{k-1}^\perp} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}](P_{\widehat{S}_{k-1}^\perp} S_k)\right\|_F + \left\|P_{\mathcal{S}^\perp} \cdot [O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}^\perp} S_k\right)\right\|_F. \tag{169}$$

We proceed by controlling each norm on the right-hand side above using the next two technical lemmas, proved in Appendices H and I, respectively.

Lemma 21 *It holds that*

$$\mathbb{E}\left[\left\|P_{\Omega_k^C}(P_{\widehat{S}_{k-1}^\perp} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}^\perp} S_k\right)\right\|_F \,|\, \widehat{\mathcal{S}}_{k-1}, Q_k\right]$$
$$\leq \sqrt{1-p} \cdot \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|_F \cdot \|Q_k\|. \tag{170}$$

For fixed $\widehat{\mathcal{S}}_{k-1}$ and Q_k, we also have

$$\left\|P_{\Omega_k^C}(P_{\widehat{S}_{k-1}^\perp} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}^\perp} S_k\right)\right\|_F \leq \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|_F \cdot \|Q_k\|. \tag{171}$$

and the stronger bound

$$\left\|P_{\Omega_k^C}(P_{\widehat{S}_{k-1}^\perp} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}^\perp} S_k\right)\right\|_F \leq \sqrt{1-p/2} \cdot \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|_F \cdot \|Q_k\|, \tag{172}$$

except with a probability of at most

$$\exp\left(-\frac{C_1 p^2 nb}{\widetilde{\eta}_k}\right), \tag{173}$$

where

$$\widetilde{\eta}_k = \widetilde{\eta}(P_{\widehat{S}_{k-1}^\perp} S_k) = nb \cdot \frac{\|P_{\widehat{S}_{k-1}^\perp} S_k\|_\infty^2}{\|P_{\widehat{S}_{k-1}^\perp} S_k\|_F^2}. \tag{174}$$

Lemma 22 *For fixed $\widehat{\mathcal{S}}_{k-1}, Q_k$ and $\alpha \geq 1$, it holds that*

$$\left\|P_{\mathcal{S}^\perp} \cdot [O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}^\perp} S_k\right)\right\|_F \lesssim \alpha \log b \sqrt{\frac{\log n}{p}} \cdot \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\| \cdot \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|_F \cdot \|Q_k\|, \tag{175}$$

except with a probability of at most $b^{-C\alpha}$ and provided that $p \gtrsim \alpha^2 \log^2 b \log n \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r/n$.

We next use Lemmas 21 and 22 to derive two bounds for the numerator of (166), the weaker bound holds with high probability but the stronger bound holds with only some probability. More specifically, substituting (171) and (175) into (169) yields that

$$\|P_{\mathcal{S}^\perp} R_k\|_F$$
$$\leq \left\|P_{\Omega_k^C}(P_{\widehat{S}_{k-1}^\perp} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}^\perp} S_k\right)\right\|_F + \left\|P_{\mathcal{S}^\perp} \cdot [O \circ P_{\Omega_k}]\left(P_{\widehat{S}_{k-1}^\perp} S_k\right)\right\|_F$$
(see (169))
$$\leq \left(1 + C\alpha \log b \sqrt{\frac{\log n}{p}} \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|\right) \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|_F \|Q_k\|, \tag{176}$$

except with a probability of at most $b^{-C\alpha}$ and provided that $p \gtrsim \alpha^2 \log^2 b \log n \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r/n$. For positive c to be set later, let us further assume that

$$\|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\| \leq \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|_F \lesssim \frac{p^{\frac{7}{2}} nb}{c\alpha \log b \sqrt{\log n}}. \tag{177}$$

With an appropriate constant replacing $\lesssim$ above, (176) simplifies to

$$\|P_{\mathcal{S}^\perp} R_k\|_F \lesssim \left(1 + \frac{p^3 nb}{c}\right) \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|_F \|Q_k\|. \tag{178}$$

A stronger bound is obtained by substituting (172, 175) into (169), namely,

$$\|P_{\mathcal{S}^\perp} R_k\|_F$$
$$\leq \left\|P_{\Omega_k^C}(P_{\widehat{S}_{k-1}^\perp} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}](P_{\widehat{S}_{k-1}^\perp} S_k)\right\|_F + \left\|P_{\mathcal{S}^\perp} \cdot [O \circ P_{\Omega_k}](P_{\widehat{S}_{k-1}^\perp} S_k)\right\|_F$$
(see (169))
$$\leq \left(\sqrt{1 - \frac{p}{2}} + \frac{p^3 nb}{c}\right) \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|_F \|Q_k\| \qquad \text{(see (172,175,177))}$$
$$\leq \left(1 - \frac{p}{4} + \frac{p^3 nb}{c}\right) \|P_{\mathcal{S}^\perp} P_{\widehat{S}_{k-1}}\|_F \|Q_k\|, \tag{179}$$

provided that $p \gtrsim \alpha^2 \log^2 b \log n \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r/n$ and except with a probability of at most

$$\exp\left(-\frac{C_1 p^2 nb}{\widetilde{\eta}_k}\right) + b^{-C\alpha}.$$

Note that (178) and (179) offer two alternative bounds for the numerator in the last line (166), which we will next use to complete the proof of Lemma 16.

Fix the subspace $\widehat{\mathcal{S}}_{k-1}$ for now. Let $\mathfrak{E}_{k-1}$ be the event where $p \gtrsim \alpha^2 \log^2 b \log n \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r/n$ and (177) holds. For $\nu \geq 1$ to be set later, let $\mathfrak{E}'_k$ be the event where

$$\|Q_k\| \leq \nu \cdot \sigma_{\min}. \tag{180}$$

Conditioned on the event $\mathfrak{E}_{k-1} \cap \mathfrak{E}'_k$, we write that

$$\|P_{\mathcal{S}^\perp} P_{\widehat{S}_k}\|_F \leq \frac{\|P_{\mathcal{S}^\perp} R_k\|_F}{\sigma_r(R_k)} \qquad \text{(see (166))}$$

$$\leq \frac{\|P_{\mathcal{S}^\perp} R_k\|_F}{\sigma_{\min}}$$
$$\lesssim \nu \left(1 + \frac{p^3 nb}{c}\right) \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F, \qquad \text{(see (178))} \tag{181}$$

except with a probability of at most $b^{-C\alpha}$. A stronger bound is obtained from (179), namely,

$$\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_k}\|_F \leq \frac{\|P_{\mathcal{S}^\perp} R_k\|_F}{\sigma_r(R_k)} \qquad \text{(see (166))}$$
$$\leq \nu \left(1 - \frac{p}{4} + \frac{p^3 nb}{c}\right) \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F, \qquad \text{(see (179))} \tag{182}$$

conditioned on the event $\mathfrak{E}_{k-1} \cap \mathfrak{E}'_k$ and except with a probability of at most

$$\exp\left(-\frac{C_1 p^2 nb}{\widetilde{\eta}_k}\right) + b^{-C\alpha}. \tag{183}$$

This completes the proof of the probabilistic claims in Lemma 16, namely, (74) and (75). To complete the proof of Lemma 16, we next derive a bound for the conditional expectation of $\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_k}\|_F$. Let $\mathfrak{E}''_k$ be the event where $p \gtrsim \alpha^2 \log^2 b \log n \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r/n$ and

$$\left\| P_{\mathcal{S}^\perp} \cdot [O \circ P_\Omega] \left(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k\right) \right\|_F \lesssim \frac{p^3 nb}{c} \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F \|Q_k\|. \tag{184}$$

In light of Lemma 22, we have that

$$\Pr[\mathfrak{E}''_k | \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_k, \mathfrak{E}'_k] \geq 1 - b^{-C\alpha}. \tag{185}$$

Using the law of total expectation, we now write that

$$\begin{aligned}
&\mathbb{E}\left[\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_k}\|_F \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k\right] \\
&= \mathbb{E}\left[\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_k}\|_F \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k\right] \cdot \Pr[\mathfrak{E}''_k | \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k] \\
&\quad + \mathbb{E}\left[\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_k}\|_F \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''^C_k\right] \cdot \Pr[\mathfrak{E}''^C_k | \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k] \\
&\leq \mathbb{E}\left[\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_k}\|_F \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k\right] + \sqrt{r} \Pr[\mathfrak{E}''^C_k | \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k] \qquad \left(\widehat{\mathcal{S}}_k \in \mathbb{G}(n,r)\right) \\
&\leq \mathbb{E}\left[\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_k}\|_F \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k\right] + \sqrt{r} b^{-C\alpha} \qquad \text{(see (185))} \\
&\leq \mathbb{E}\left[\frac{\|P_{\mathcal{S}^\perp} R_k\|_F}{\sigma_r(R_k)} \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k\right] + b^{-C\alpha} \qquad \text{((166) and } b \geq r) \\
&\leq \sigma_{\min}^{-1} \mathbb{E}\left[\|P_{\mathcal{S}^\perp} R_k\|_F \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k\right] + b^{-C\alpha}.
\end{aligned} \tag{186}$$

We next bound the remaining expectation above by writing that

$$\begin{aligned}
&\mathbb{E}\left[\|P_{\mathcal{S}^\perp} R_k\|_F | \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k\right] \\
&\leq \mathbb{E}\left[\left\| P_{\Omega_k^C} (P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}] (P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F \right. \\
&\quad \left. + \left\| P_{\mathcal{S}^\perp} \cdot [O \circ P_\Omega] (P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k \right] \qquad \text{(see (169))} \\
&\leq \mathbb{E}\left[\left\| P_{\Omega_k^C} (P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}] (P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F \right. \\
&\quad \left. + \frac{p^3 nb}{c} \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F \|Q_k\| \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k \right] \qquad \text{(see (184))} \\
&= \mathbb{E}\left[\mathbb{E}\left[\left\| P_{\Omega_k^C} (P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}] (P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F | \widehat{\mathcal{S}}_{k-1}, Q_k \right] \right. \\
&\quad \left. + \frac{p^3 nb}{c} \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F \|Q_k\| | \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k \right] \\
&\leq \mathbb{E}\left[\sqrt{1-p} \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F \|Q_k\| + \frac{p^3 nb}{c} \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F \|Q_k\| | \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k \right] \\
&\qquad \text{(see (170))} \\
&= \mathbb{E}\left[\nu \cdot \sigma_{\min} \left(\sqrt{1-p} + \frac{p^3 nb}{c}\right) \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F | \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k \right] \qquad \text{(see (180))} \\
&= \nu \cdot \sigma_{\min} \left(\sqrt{1-p} + \frac{p^3 nb}{c}\right) \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F \\
&\leq \nu \cdot \sigma_{\min} \left(1 - \frac{p}{2} + \frac{p^3 nb}{c}\right) \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F.
\end{aligned} \tag{187}$$

Plugging the bound above back into (186) yields that

$$\begin{aligned}
&\mathbb{E}\left[\|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_k}\|_F | \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k\right] \\
&\leq \sigma_{\min}^{-1} \mathbb{E}\left[\|P_{\mathcal{S}^\perp} R_k\|_F \,|\, \widehat{\mathcal{S}}_{k-1}, \mathfrak{E}_{k-1}, \mathfrak{E}'_k, \mathfrak{E}''_k\right] + b^{-C\alpha} \qquad \text{(see (186))} \\
&= \nu \left(1 - \frac{p}{2} + \frac{p^3 nb}{c}\right) \|P_{\mathcal{S}^\perp} P_{\widehat{\mathcal{S}}_{k-1}}\|_F + b^{-C\alpha},
\end{aligned} \tag{188}$$

which proves (73) and completes the proof of Lemma 16.

Appendix H. Proof of Lemma 21

Throughout, $\widehat{\mathcal{S}}_{k-1}, Q_k$ is fixed. Recalling the definition of operator $O(\cdot)$ from (165), we write that

$$\begin{aligned}
&[P_{\Omega_k} \circ O \circ P_{\Omega_k}] (P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \\
&= \left[\begin{array}{cccc} \cdots & \left(P_{\omega_j} \widehat{S}_{k-1}\right) \left(P_{\omega_j} \widehat{S}_{k-1}\right) \cdot P_{\omega_j} P_{\widehat{\mathcal{S}}^\perp_{k-1}} S q_j & \cdots \end{array}\right] \qquad \text{(see (165))} \\
&= \left[\begin{array}{cccc} \cdots & \left(P_{\omega_j} \widehat{S}_{k-1}\right) \left(P_{\omega_j} \widehat{S}_{k-1}\right) \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} S q_j & \cdots \end{array}\right] \\
&= \left[\begin{array}{cccc} \cdots & P_{\widehat{\mathcal{S}}_{k-1,j}} \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} S q_j & \cdots \end{array}\right]. \qquad \left(\widehat{\mathcal{S}}_{k-1,j} := \text{span}(P_{\omega_j} \widehat{S}_{k-1})\right)
\end{aligned} \tag{189}$$

Let also $\widehat{\mathcal{S}}^C_{k-1,j} := \text{span}(P_{\omega_j^C} \widehat{S}_{k-1})$. Note that $\widehat{S}_{k-1} = P_{\omega_j} \widehat{S}_{k-1} + P_{\omega_j^C} \widehat{S}_{k-1}$ and that $(P_{\omega_j} \widehat{S}_{k-1})^* (P_{\omega_j^C} \widehat{S}_{k-1}) = 0$. Consequently, $\widehat{\mathcal{S}}_{k-1,j} \perp \widehat{\mathcal{S}}^C_{k-1,j}$ and then $P_{\widehat{\mathcal{S}}_{k-1}} = P_{\widehat{\mathcal{S}}_{k-1,j}} +$

$P_{\widehat{\mathcal{S}}^C_{k-1,j}}$. Put differently, $\widehat{\mathcal{S}}_{k-1} = \widehat{\mathcal{S}}_{k-1,j} \oplus \widehat{\mathcal{S}}^C_{k-1,j}$. Using this decomposition, we simplify (189) as

$$\begin{aligned}
&[P_{\Omega_k} \circ O \circ P_{\Omega_k}]\,(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \\
&= \begin{bmatrix} \cdots & P_{\widehat{\mathcal{S}}_{k-1,j}} \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} Sq_j & \cdots \end{bmatrix} \qquad \text{(see (189))} \\
&= \begin{bmatrix} \cdots & (P_{\widehat{\mathcal{S}}_{k-1}} - P_{\widehat{\mathcal{S}}^C_{k-1,j}}) \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} Sq_j & \cdots \end{bmatrix} \qquad \left(\widehat{\mathcal{S}}_{k-1} = \widehat{\mathcal{S}}_{k-1,j} \oplus \widehat{\mathcal{S}}^C_{k-1,j}\right) \\
&= -\begin{bmatrix} \cdots & P_{\widehat{\mathcal{S}}^C_{k-1,j}} \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} Sq_j & \cdots \end{bmatrix}. \qquad (190)
\end{aligned}$$

It immediately follows that

$$\begin{aligned}
&P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\,(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \\
&= \begin{bmatrix} \cdots & P_{\omega_j^C} \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} Sq_j - P_{\widehat{\mathcal{S}}^C_{k-1,j}} \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} Sq_j & \cdots \end{bmatrix} \\
&= \begin{bmatrix} \cdots & P_{\omega_j^C}(I_n - P_{\widehat{\mathcal{S}}^C_{k-1,j}}) P_{\omega_j^C} \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} Sq_j & \cdots \end{bmatrix}, \qquad \left(\widehat{\mathcal{S}}^C_{k-1,j} = \operatorname{span}(P_{\omega_j^C} \widehat{S}_{k-1})\right) \qquad (191)
\end{aligned}$$

and consequently

$$\begin{aligned}
&\left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\,(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F^2 \\
&= \sum_{j=1}^{b} \left\| P_{\omega_j^C}(I_n - P_{\widehat{\mathcal{S}}^C_{k-1,j}}) P_{\omega_j^C} \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} Sq_j \right\|_2^2 \qquad \text{(see (191))} \\
&\le \sum_{j=1}^{b} \left\| P_{\omega_j^C} \cdot P_{\widehat{\mathcal{S}}^\perp_{k-1}} Sq_j \right\|_2^2 \\
&= \left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} SQ_k) \right\|_F^2 \\
&= \left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F^2. \qquad (S_k = SQ_k) \qquad (192)
\end{aligned}$$

Note that

$$\begin{aligned}
&\mathbb{E}\left[\left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\,(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F \mid \widehat{\mathcal{S}}_{k-1}, Q_k \right] \\
&\le \sqrt{\mathbb{E}\left[\left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\,(P_{\mathcal{S}^\perp_{k-1}} S_k) \right\|_F^2 \mid \widehat{\mathcal{S}}_{k-1}, Q_k \right]} \qquad \text{(Jensen's inequality)} \\
&\le \sqrt{\mathbb{E}\left[\left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F^2 \mid \widehat{\mathcal{S}}_{k-1}, Q_k \right]} \qquad \text{(see (192))} \\
&= \sqrt{\mathbb{E}\left[\sum_{i,j} (1 - \epsilon_{i,j}) \left| (P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k)[i,j] \right|^2 \mid \widehat{\mathcal{S}}_{k-1}, Q_k \right]} \\
&= \sqrt{1-p}\, \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_F
\end{aligned}$$

$$\le \sqrt{1-p}\, \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S \|_F \| Q_k \| \qquad (S_k = SQ_k^*, \qquad \|AB\|_F \le \|A\|_F \cdot \|B\|) \qquad (193)$$

where $\{\epsilon_{i,j}\}_{i,j} \overset{\text{ind.}}{\sim} \text{Bernoulli}(p)$. We thus proved the first claim in Lemma 21, namely, (170). Then note that

$$\begin{aligned}
&\left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\,(P_{\mathcal{S}^\perp_{k-1}} S_k) \right\|_F^2 \\
&\le \left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F^2 \qquad \text{(see (192))} \\
&\le \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_F^2 \\
&\le \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S \|_F^2 \| Q_k \|^2, \qquad (S_k = SQ_k) \qquad (194)
\end{aligned}$$

which proves the second claim, namely, (171). In fact, with some probability, a stronger bound can be derived by controlling the deviation from the expectation in (193) using the Hoeffding inequality (Lemma 17). With $\alpha = \frac{p}{2} \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_F^2$ in Lemma 17 and recalling that $\widehat{\mathcal{S}}_{k-1}, Q_k$ are fixed for now, we find that

$$\begin{aligned}
\left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F^2 &\le \mathbb{E}\left[\left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F^2 \mid \widehat{\mathcal{S}}_{k-1}, Q_k \right] + \alpha \\
&= (1 - p/2) \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_F^2 \qquad \text{(see (193))} \\
&= (1 - p/2) \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} SQ_k \|_F^2 \qquad (S_k = SQ_k) \\
&\le (1 - p/2) \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S \|_F^2 \, \| Q_k \|^2, \qquad (195)
\end{aligned}$$

except with a probability of at most

$$\begin{aligned}
\exp\left(-\frac{C_1 p^2 \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_F^4}{\sum_{i,j} |(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k)[i,j]|^4} \right) &\le \exp\left(-\frac{C_1 p^2 \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_F^4}{\| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_\infty^2 \cdot \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_F^2} \right) \\
&= \exp\left(-\frac{C_1 p^2 \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_F^2}{\| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k \|_\infty^2} \right) \\
&=: \exp\left(\frac{-C_1 p^2 nb}{\widetilde{\eta}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k)} \right), \qquad (196)
\end{aligned}$$

where $\|A\|_\infty$ returns the largest entry of A in magnitude. Substituting (195) back into (194) yields that

$$\begin{aligned}
&\left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) + [P_{\Omega_k} \circ O \circ P_{\Omega_k}]\,(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F \\
&\le \left\| P_{\Omega_k^C}(P_{\widehat{\mathcal{S}}^\perp_{k-1}} S_k) \right\|_F \qquad \text{(see (194))} \\
&\le \sqrt{1 - p/2} \cdot \| P_{\widehat{\mathcal{S}}^\perp_{k-1}} S \|_F \| Q_k \|, \qquad (197)
\end{aligned}$$

which proves the last claim in Lemma 21, namely, (172).

Appendix I. Proof of Lemma 22

We fix $\widehat{\mathcal{S}}_{k-1}, Q_k$ throughout. We begin by bounding the target quantity as

$$\begin{aligned}
&\left\|P_{\mathcal{S}^\perp}\cdot[O\circ P_{\Omega_k}]\left(P_{\widehat{\mathcal{S}}_{k-1}^\perp}S_k\right)\right\|_F\\
&=\left\|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_{k-1}}\cdot[O\circ P_{\Omega_k}]\left(P_{\widehat{\mathcal{S}}_{k-1}^\perp}S_k\right)\right\|_F \qquad \text{(see (165))}\\
&\le\|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_{k-1}}\|\cdot\left\|[O\circ P_{\Omega_k}]\left(P_{\widehat{\mathcal{S}}_{k-1}^\perp}S_k\right)\right\|_F \qquad (\|AB\|_F\le\|A\|\cdot\|B\|_F). \qquad (198)
\end{aligned}$$

We bound the random norm in the last line above in Appendix J.

Lemma 23 *For $\alpha\ge 1$ and except with a probability of at most $b^{-C\alpha}$, it holds that*

$$\left\|[O\circ P_{\Omega_k}]\left(P_{\widehat{\mathcal{S}}_{k-1}^\perp}S_k\right)\right\|_F\lesssim\alpha\log b\sqrt{\frac{\log n}{p}}\|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_{k-1}}\|_F\cdot\|Q_k\|, \qquad (199)$$

provided that $p\gtrsim\alpha^2\log^2 b\log n\cdot\eta(\widehat{\mathcal{S}}_{k-1})r/n$.

In light of the above lemma, we conclude that

$$\begin{aligned}
&\left\|P_{\mathcal{S}^\perp}\cdot[O\circ P_{\Omega_k}]\left(P_{\widehat{\mathcal{S}}_{k-1}^\perp}S_k\right)\right\|_F\\
&\le\|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_{k-1}}\|\cdot\left\|[O\circ P_{\Omega_k}]\left(P_{\widehat{\mathcal{S}}_{k-1}^\perp}S_k\right)\right\|_F \qquad \text{(see (198))}\\
&\lesssim\alpha\log b\sqrt{\frac{\log n}{p}}\|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_{k-1}}\|\cdot\|P_{\mathcal{S}^\perp}P_{\widehat{\mathcal{S}}_{k-1}}\|_F\cdot\|Q_k\|, \qquad \text{(see (199))} \qquad (200)
\end{aligned}$$

except with a probability of at most $b^{-C\alpha}$ and provided that $p\gtrsim\alpha^2\log^2 b\log n\cdot\eta(\widehat{\mathcal{S}}_{k-1})r/n$. This completes the proof of Lemma 22.

Appendix J. Proof of Lemma 23

Using the definition of operator O in (165), we write that

$$\begin{aligned}
&\left\|[O\circ P_{\Omega_k}]\left(P_{\widehat{\mathcal{S}}_{k-1}^\perp}S_k\right)\right\|_F^2\\
&=\sum_{j=1}^b\left\|\widehat{S}_{k-1}(P_{\omega_j}\widehat{S}_{k-1})^\dagger P_{\omega_j}\cdot P_{\widehat{\mathcal{S}}_{k-1}^\perp}Sq_j\right\|_2^2 \qquad (\text{(165) and } S_k=SQ_k)\\
&=\sum_{j=1}^b\left\|\widehat{S}_{k-1}(P_{\omega_j}\widehat{S}_{k-1})^\dagger P_{\widehat{\mathcal{S}}_{k-1}^\perp}Sq_j\right\|_2^2\\
&=\sum_{j=1}^b\left\|(P_{\omega_j}\widehat{S}_{k-1})^\dagger P_{\widehat{\mathcal{S}}_{k-1}^\perp}Sq_j\right\|_2^2. \qquad \left(\widehat{S}_{k-1}^*\widehat{S}_{k-1}=I_r\right) \qquad (201)
\end{aligned}$$

For fixed $j\in[1:b]$, consider the summand in the last line above:

$$\begin{aligned}
\left\|(P_{\omega_j}\widehat{S}_{k-1})^\dagger P_{\widehat{\mathcal{S}}_{k-1}^\perp}S\cdot q_j\right\|_2&\le\left\|(P_{\omega_j}\widehat{S}_{k-1})^\dagger P_{\widehat{\mathcal{S}}_{k-1}^\perp}\right\|\cdot\|P_{\widehat{\mathcal{S}}_{k-1}^\perp}Sq_j\|_2\\
&=\left\|(P_{\omega_j}\widehat{S}_{k-1})^\dagger\widehat{S}_{k-1}^\perp\right\|\cdot\|P_{\widehat{\mathcal{S}}_{k-1}^\perp}Sq_j\|_2 \qquad \left(P_{\widehat{\mathcal{S}}_{k-1}^\perp}=\widehat{S}_{k-1}^\perp(\widehat{S}_{k-1}^\perp)^*\right)\\
&=:\|\widehat{Z}_j\|\cdot\|P_{\widehat{\mathcal{S}}_{k-1}^\perp}Sq_j\|_2. \qquad (202)
\end{aligned}$$

Above, $\widehat{S}_{k-1}^\perp$ is as usual an orthonormal basis for the subspace $\widehat{\mathcal{S}}_{k-1}^\perp$. We can now revisit (201) and write that

$$\begin{aligned}
\left\|O(P_{\widehat{\mathcal{S}}_{k-1}^\perp}S_k)\right\|_F^2&=\sum_{j=1}^b\left\|(P_{\omega_j}\widehat{S}_{k-1})^\dagger P_{\widehat{\mathcal{S}}_{k-1}^\perp}Sq_j)\right\|_2^2 \qquad \text{(see (201))}\\
&\le\max_j\|\widehat{Z}_j\|^2\cdot\sum_{j=1}^b\|P_{\widehat{\mathcal{S}}_{k-1}^\perp}Sq_j\|_2^2 \qquad \text{(see (202))}\\
&=\max_j\|\widehat{Z}_j\|^2\cdot\|P_{\widehat{\mathcal{S}}_{k-1}^\perp}SQ_k\|_F^2\\
&\le\max_j\|\widehat{Z}_j\|^2\cdot\|P_{\widehat{\mathcal{S}}_{k-1}^\perp}S\|_F^2\|Q_k\|^2. \qquad (\|AB\|_F\le\|A\|_F\cdot\|B\|) \qquad (203)
\end{aligned}$$

It remains to control the maximum in the last line above. We first focus on controlling $\|\widehat{Z}_j\|$ for fixed $j\in[1:b]$. Observe that $\widehat{Z}_j$ is a solution of the least-squares problem

$$\min_{Z\in\mathbb{R}^{n\times(n-r)}}\left\|\widehat{S}_{k-1}^\perp-(P_{\omega_j}\widehat{S}_{k-1})Z\right\|_F^2,$$

and therefore satisfies the *normal equation*

$$(P_{\omega_j}\widehat{S}_{k-1})^*\left((P_{\omega_j}\widehat{S}_{k-1})\widehat{Z}_j-\widehat{S}_{k-1}^\perp\right)=0,$$

which is itself equivalent to

$$(\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1})\widehat{Z}_j=\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1}^\perp. \qquad \left(P_{\omega_j}^2=P_{\omega_j}\right) \qquad (204)$$

In fact, since

$$\begin{aligned}
\mathbb{E}\left[\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1}^\perp\right]&=p\cdot\widehat{S}_{k-1}^*\widehat{S}_{k-1}^\perp=0,\\
\mathbb{E}\left[\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1}\right]&=p\cdot I_r, \qquad \left(\widehat{S}_{k-1}^*\widehat{S}_{k-1}=I_r\right)
\end{aligned}$$

we can rewrite (204) as

$$\left(\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1}-\mathbb{E}\left[\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1}\right]\right)\widehat{Z}_j+p\cdot\widehat{Z}_j=\widehat{S}_{k-1}^TP_{\omega_j}\widehat{S}_{k-1}^\perp-\mathbb{E}\left[\widehat{S}_{k-1}^TP_{\omega_j}\widehat{S}_{k-1}^\perp\right].$$

An application of the triangle inequality above immediately implies that

$$p\|\widehat{Z}_j\|\le\left\|\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1}-\mathbb{E}\left[\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1}\right]\right\|\cdot\|\widehat{Z}_j\|+\left\|\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1}^\perp-\mathbb{E}\left[\widehat{S}_{k-1}^*P_{\omega_j}\widehat{S}_{k-1}^\perp\right]\right\|. \qquad (205)$$

To control $\|\widehat{Z}_j\|$, we therefore need to derive large devation bounds for the two remaining norms on the right-hand side above. For the first spectral norm, we write that

$$\left\|\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1} - \mathbb{E}\left[\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1}\right]\right\| = \left\|\sum_i (\epsilon_i - p) \cdot \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}\right\| =: \left\|\sum_i A_i\right\|, \quad (206)$$

where $\{\epsilon_i\}_i \overset{\text{ind.}}{\sim} \text{Bernoulli}(p)$ and $E_{i,i} \in \mathbb{R}^{n\times n}$ is the $[i,i]$th canonical matrix. Furthermore, $\{A_i\}_i \subset \mathbb{R}^{r\times r}$ above are independent and zero-mean random matrices. To apply the Bernstein inequality (Lemma 18), we first compute the parameter β as

$$\begin{aligned}
\|A_i\| &= \left\|(\epsilon_i - p) \cdot S_{k-1}^* E_{i,i} S_{k-1}\right\| \qquad \text{(see (206))} \\
&\le \|\widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}\| \qquad (\epsilon_i \in \{0,1\}) \\
&= \|\widehat{S}_{k-1}[i,:]\|_2^2 \\
&\le \frac{\eta(\widehat{\mathcal{S}}_{k-1})r}{n} =: \beta. \qquad \text{(see (14))}
\end{aligned} \quad (207)$$

To compute the weak variance σ, we write that

$$\begin{aligned}
\left\|\mathbb{E}\left[\sum_i A_i^2\right]\right\| &= \left\|\sum_i \mathbb{E}\left[(\epsilon_i - p)^2\right] (\widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1})^2\right\| \qquad \text{(see (206))} \\
&= \left\|\sum_i p(1-p)(\widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1})^2\right\| \qquad (\epsilon_i \sim \text{Bernoulli}(p)) \\
&\le p \left\|\sum_i (\widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1})^2\right\| \\
&= p \left\|\sum_i \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1} \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}\right\| \\
&= p \left\|\sum_i \|\widehat{S}_{k-1}[i,:]\|_2^2 \cdot \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}\right\| \\
&\le p \cdot \max_i \|\widehat{S}_{k-1}[i,:]\|_2^2 \cdot \left\|\sum_i \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}\right\| \\
&= p \left\|\sum_i \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}\right\| \\
&= p \cdot \frac{\eta(\widehat{\mathcal{S}}_{k-1})r}{n} \cdot \left\|\sum_i \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}\right\| \qquad \text{(see (14))} \\
&= p \cdot \frac{\eta(\widehat{\mathcal{S}}_{k-1})r}{n} \qquad \left(\sum_i E_{i,i} = I_n,\ \widehat{S}_{k-1}^* \widehat{S}_{k-1} = I_r\right) \\
&=: \sigma^2.
\end{aligned} \quad (208)$$

It also follows that

$$\begin{aligned}
&\max\left(\log r \cdot \beta, \sqrt{\log r} \cdot \sigma\right) \\
&= \max\left(\frac{\log r \cdot \eta(\widehat{\mathcal{S}}_{k-1})r}{n}, \sqrt{\frac{\log r \cdot p \cdot \eta(\widehat{\mathcal{S}}_{k-1})r}{n}}\right) \qquad \text{(see (207) and (208))} \\
&\le \sqrt{\frac{\log r \cdot p \cdot \eta(\widehat{\mathcal{S}}_{k-1})r}{n}}. \qquad \left(\text{if } p \ge \frac{\log r \cdot \eta(\widehat{\mathcal{S}}_{k-1})r}{n}\right)
\end{aligned} \quad (209)$$

As a result, for $\alpha \ge 1$ and except with a probability of at most $e^{-C\alpha}$, it holds that

$$\begin{aligned}
\left\|\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1} - \mathbb{E}\left[\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1}\right]\right\| &\lesssim \alpha \max\left(\log r \cdot \beta, \sqrt{\log r} \cdot \sigma\right) \qquad \text{(see Lemma 18)} \\
&\le \alpha \sqrt{\frac{\log r \cdot p \cdot \eta(\widehat{\mathcal{S}}_{k-1})r}{n}}.
\end{aligned} \quad (210)$$

On the other hand, in order to apply the Bernstein inequality to the second spectral norm in (205), we write that

$$\left\|\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1}^\perp - \mathbb{E}\left[\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1}^\perp\right]\right\| = \left\|\sum_i (\epsilon_i - p)\, \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}^\perp\right\| =: \left\|\sum_i A_i\right\|, \quad (211)$$

where $\{\epsilon_i\}_i \overset{\text{ind.}}{\sim} \text{Bernoulli}(p)$, $E_{i,i} \in \mathbb{R}^{n\times n}$ is the ith canonical matrix, and $\{A_i\}_i \subset \mathbb{R}^{r\times(n-r)}$ are zero-mean and independent random matrices. To compute the parameter β here, we write that

$$\begin{aligned}
\|A_i\| &= \left\|(\epsilon_i - p)\, \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}^\perp\right\| \qquad \text{(see (211))} \\
&\le \|\widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}^\perp\| \qquad (\epsilon_i \in \{0,1\}) \\
&\le \|\widehat{S}_{k-1}^* E_{i,i}\| \qquad \left((\widehat{S}_{k-1}^\perp)^* \widehat{S}_{k-1}^\perp = I_{n-r}\right) \\
&= \|\widehat{S}_{k-1}[i,:]\|_2 \\
&\le \sqrt{\frac{\eta(\widehat{\mathcal{S}}_{k-1})r}{n}} =: \beta. \qquad \text{(see (14))}
\end{aligned} \quad (212)$$

To compute the weak variance σ, we notice that

$$\begin{aligned}
\left\|\mathbb{E}\left[\sum_i A_i A_i^*\right]\right\| &= \left\|\sum_i \mathbb{E}\left[(\epsilon_i - p)^2\right] \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}^\perp (\widehat{S}_{k-1}^\perp)^* E_{i,i} \widehat{S}_{k-1}\right\| \qquad \text{(see (211))} \\
&= \left\|\sum_i p(1-p) \cdot \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}^\perp (\widehat{S}_{k-1}^\perp)^* E_{i,i} \widehat{S}_{k-1}\right\| \qquad (\epsilon_i \sim \text{Benoulli}(p)) \\
&\le p \left\|\sum_i \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}^\perp (\widehat{S}_{k-1}^\perp)^* E_{i,i} \widehat{S}_{k-1}\right\|
\end{aligned}$$

$$\begin{aligned}
&\le p\left\|\sum_i \widehat{S}_{k-1}^* E_{i,i} E_{i,i} \widehat{S}_{k-1}\right\| \qquad \left(\widehat{S}_{k-1}^{\perp}(\widehat{S}_{k-1}^{\perp})^* \preceq I_n\right)\\
&= p\left\|\sum_i \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}\right\|\\
&= p. \qquad \left(\sum_i E_{i,i} = I_n,\ \widehat{S}_{k-1}^* \widehat{S}_{k-1} = I_r\right)
\end{aligned} \tag{213}$$

In a similar fashion, we find that

$$\begin{aligned}
\left\|\mathbb{E}\left[\sum_i A_i^* A_i\right]\right\| &\le p\left\|\sum_i (\widehat{S}_{k-1}^{\perp})^* E_{i,i} \widehat{S}_{k-1} \widehat{S}_{k-1}^* E_{i,i} \widehat{S}_{k-1}^{\perp}\right\|\\
&= p\left\|\sum_i \|\widehat{S}_{k-1}[i,:]\|_2^2 \cdot (\widehat{S}_{k-1}^{\perp})^* E_{i,i} \widehat{S}_{k-1}^{\perp}\right\|\\
&\le p \cdot \max_i \|\widehat{S}_{k-1}[i,:]\|_2^2 \cdot \left\|\sum_i (\widehat{S}_{k-1}^{\perp})^* E_{i,i} \widehat{S}_{k-1}^{\perp}\right\|\\
&= p \cdot \max_i \|\widehat{S}_{k-1}[i,:]\|_2^2 \qquad \left(\sum_i E_{i,i} = I_n,\ (\widehat{S}_{k-1}^{\perp})^* \widehat{S}_{k-1}^{\perp} = I_{n-r}\right)\\
&= p \cdot \frac{\eta(\widehat{\mathcal{S}}_{k-1}) r}{n}, \qquad \text{(see (14))}
\end{aligned} \tag{214}$$

and finally

$$\begin{aligned}
\sigma &= \max\left(\left\|\mathbb{E}\sum_i A_i A_i^*\right\|, \left\|\mathbb{E}\sum_i A_i^* A_i\right\|\right)\\
&= \max\left(\sqrt{p}, \sqrt{p} \cdot \sqrt{\frac{\eta(\widehat{\mathcal{S}}_{k-1}) r}{n}}\right) \qquad \text{(see (213) and (214))}\\
&= \sqrt{p}. \qquad \left(\eta(\widehat{\mathcal{S}}_{k-1}) \le \frac{n}{r}\right)
\end{aligned} \tag{215}$$

We now compute

$$\begin{aligned}
\max\left(\log n \cdot \beta, \sqrt{\log n} \cdot \sigma\right) &= \max\left(\log n \sqrt{\frac{\eta(\widehat{\mathcal{S}}_{k-1}}{n}}, \sqrt{\log n \cdot p}\right) \qquad \text{(see (212) and (215))}\\
&= \sqrt{\log n \cdot p}. \qquad \left(\text{if } p \ge \frac{\log n \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r}{n}\right)
\end{aligned} \tag{216}$$

Therefore, for $\alpha \ge 1$ and except with a probability of at most $e^{-C\alpha}$, it holds that

$$\left\|\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1}^{\perp} - \mathbb{E}\left[\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1}^{\perp}\right]\right\| \lesssim \alpha \max\left(\log n \cdot \beta, \sqrt{\log n} \cdot \sigma\right) \qquad \text{(see Lemma 18)}$$

$$= \alpha\sqrt{\log n \cdot p}. \tag{217}$$

Overall, by substituting the large deviation bounds (210) and (217) into (205), we find that

$$\begin{aligned}
p\|\widehat{Z}_j\| &\le \left\|\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1} - \mathbb{E}\left[\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1}\right]\right\| \cdot \|\widehat{Z}_j\|\\
&\quad + \left\|\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1}^{\perp} - \mathbb{E}\left[\widehat{S}_{k-1}^* P_{\omega_j} \widehat{S}_{k-1}^{\perp}\right]\right\| \qquad \text{(see (205))}\\
&\lesssim \alpha\sqrt{\frac{\log r \cdot p \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r}{n}} \cdot \|\widehat{Z}_j\| + \alpha\sqrt{\log n \cdot p}, \qquad \text{(see (210) and (217))}
\end{aligned}$$

except with a probability of at most $e^{-C\alpha}$ and under (209) and (216). It immediately follows that

$$\begin{aligned}
\|\widehat{Z}_j\| &\lesssim \frac{\alpha\sqrt{\frac{\log n}{p}}}{1 - \sqrt{\frac{\alpha^2 \log r \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r}{pn}}} \qquad \text{(see the next line)}\\
&\lesssim \alpha\sqrt{\frac{\log n}{p}}, \qquad \left(\text{if } \frac{\alpha^2 \log r \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r}{pn} \lesssim 1\right)
\end{aligned} \tag{218}$$

except with a probability of at most $e^{-C\alpha}$. In light of (209) and (216), we assume that $p \gtrsim \alpha^2 \log n \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r/n$. Then using the union bound and with the choice of $\alpha = \alpha' \log b$, it follows that

$$\max_{j \in [1:b]} \|\widehat{Z}_j\| \lesssim \alpha' \log b \sqrt{\frac{\log n}{p}},$$

provided that $p \gtrsim \alpha'^2 \log^2 b \cdot \log n \cdot \eta(\widehat{\mathcal{S}}_{k-1}) r/n$ and except with a probability of at most $be^{-C\alpha' \log b} = b^{-C\alpha'}$. Invoking (203), we finally conclude that

$$\begin{aligned}
\left\|O(P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} S_k)\right\|_F &\le \max_j \|\widehat{Z}_j\| \cdot \|P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} P_{\mathcal{S}}\|_F \|Q_k\| \qquad \text{(see (203))}\\
&\lesssim \alpha' \log b \sqrt{\frac{\log n}{p}} \cdot \|P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} P_{\mathcal{S}}\|_F \|Q_k\|.
\end{aligned}$$

A bound in expectation also easily follows: Let δ denote the factor of δ' in last line above. Then we have that

$$\begin{aligned}
\mathbb{E}\left\|P_{\mathcal{S}^{\perp}} \cdot [O \circ P_{\Omega_k}] (P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} S_k)\right\|_F &= \delta \int_0^{\infty} \Pr\left[\left\|P_{\mathcal{S}^{\perp}} \cdot [O \circ P_{\Omega}] (P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} S_k)\right\|_F > \alpha'\delta\right] d\alpha'\\
&\le \delta\left(1 + \int_1^{\infty} \Pr\left[\left\|P_{\mathcal{S}^{\perp}} \cdot [O \circ P_{\Omega}] (P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} S_k)\right\|_F > \alpha'\delta\right] d\alpha'\right)\\
&\le \delta\left(1 + \int_1^{\infty} b^{-\alpha'} d\alpha'\right)\\
&\le 2\delta\\
&= 2\log b \sqrt{\frac{\log n}{p}} \cdot \|P_{\widehat{\mathcal{S}}_{k-1}^{\perp}} P_{\mathcal{S}}\|_F \|Q_k\|.
\end{aligned} \tag{219}$$

This completes the proof of Lemma 23.

Appendix K. Properties of a Standard Random Gaussian Matrix

As a supplement to Remark 25, we show here that a standard random Gaussian matrix $G \in \mathbb{R}^{b\times r}$ is well-conditioned and incoherent when b is sufficiently large. From (Vershynin, 2012a, Corollary 5.35) and for fixed $\alpha \geq 1$, recall that

$$\sqrt{b} - C_3\alpha\sqrt{r} \leq \sigma_r(G) \leq \sigma_1(G) \leq \sqrt{b} + C_3\alpha\sqrt{r}, \tag{220}$$

except with a probability of at most $e^{-\alpha^2 r}$. It follows that

$$\nu(G) = \frac{\sigma_1(G)}{\sigma_r(G)} \leq \frac{\sqrt{b} + C_3\alpha\sqrt{r}}{\sqrt{b} - C_3\alpha\sqrt{r}}, \tag{221}$$

which can be made close to one by choosing $b \gtrsim \alpha^2 r$.

For the coherence, note that $G(G^*G)^{-\frac{1}{2}} \in \mathbb{R}^{b\times r}$ is an orthonormal basis for $\mathrm{span}(G)$. Using the definition of coherence in (14), we then write that

$$\begin{aligned}
\eta\left(\mathrm{span}(G)\right) &= \frac{b}{r}\max_{i\in[1:b]} \left\| G[i,:]\left(G^*G\right)^{-\frac{1}{2}} \right\|_2^2 \qquad \text{(see (14))} \\
&\leq \frac{b}{r}\max_i \|G[i,:]\|_2^2 \cdot \| \left(G^*G\right)^{-\frac{1}{2}} \|^2 \\
&= \frac{b}{r}\max_i \|G[i,:]\|_2^2 \cdot \left(\sigma_r(G)\right)^{-2} \\
&\leq \frac{b}{r}\max_i \|G[i,:]\|_2^2 \cdot \left(\sqrt{b} - C_3\alpha\sqrt{r}\right)^{-2} \qquad \text{(see (220))} \\
&\leq \frac{b}{r}\max_i \|G[i,:]\|_2^2 \cdot \left(\frac{b}{2} - C_3^2\alpha^2 r\right)^{-1} \qquad \left((a-b)^2 \geq \frac{a^2}{2} - b^2\right) \\
&\lesssim \frac{b}{r}\max_i \|G[i,:]\|_2^2 \cdot \left(b - C_4\alpha^2 r\right)^{-1},
\end{aligned} \tag{222}$$

except with a probability of at most $e^{-\alpha^2 r}$. For fixed i, $\|G[i,:]\|_2^2$ is a chi-squared random variable with r degrees of freedom so that

$$\Pr\left[\|G[i,:]\|_2^2 \gtrsim \beta\cdot r\right] \leq e^{-\beta}, \tag{223}$$

for $\beta \geq 1$. An application of the union bound and the choice of $\beta = C\alpha\log b$ then leads us to

$$\Pr\left[\max_{i\in[1:b]} \|G[i,:]\|_2^2 \gtrsim \alpha\log b\cdot r\right] \leq b\cdot b^{-C\alpha} = b^{-C\alpha}. \tag{224}$$

Substituting the bound above back into (222) yields that

$$\begin{aligned}
\eta\left(\mathrm{span}(G)\right) &\lesssim \frac{b}{r}\max_i \|G[i,:]\|_2^2 \cdot (b - C_4 r)^{-1} \qquad \text{(see (222))} \\
&\lesssim \frac{b}{r}\cdot r\alpha\log b\cdot (b - C_4 r)^{-1} \qquad \text{(see (224))} \\
&\lesssim \frac{\alpha b\log b}{b - C_4\alpha^2 r},
\end{aligned} \tag{225}$$

except with a probability of at most $e^{-\alpha r} + b^{-C\alpha}$. In particular, when $b \geq 2C_4\alpha^2 r$, we find that $\eta(\mathrm{span}(G)) \lesssim \alpha\log b$ except with a probability of at most $e^{-\alpha r} + b^{-C\alpha}$.

Appendix L. Alternative Initialization

SNIPE in Algorithm 1 is initialized by truncating the SVD of the first incomplete block $Y_1 \in \mathbb{R}^{n\times b}$, where we often take $b = O(r)$ to keep the computational and storage requirements of SNIPE minimal, see Remarks 1 and 2. Put differently, with the notation of Section 3, even though our end goal is to compute rank-r truncated SVD of the full (but hidden) data block $S_1 \in \mathbb{R}^{n\times b}$, SNIPE is initialized with truncated SVD of the incomplete (but available) block $Y_1 = P_{\Omega_1}(S_1) \in \mathbb{R}^{n\times b}$, which fills in the missing entries with zeros. Indeed, when the first incomplete block Y_1 arrives, there is no prior knowledge to leverage and zero-filling the missing entries in Y_1 is a sensible strategy. In contrast, for the rest of blocks $k \geq 2$, SNIPE uses its previous estimate $\widehat{\mathcal{S}}_{k-1}$ to fill out the erased entries in Y_k before updating its estimate to $\widehat{\mathcal{S}}_k$, see Algorithm 1.

One might instead initialize SNIPE with a larger block. More specifically, suppose that we change the first block size to $b_1 \geq b$ while keeping the rest of the blocks at the same size b. Then we set $\widehat{\mathcal{S}}_1$ to be the span of leading r left singular vectors of the first incomplete block $Y_1 \in \mathbb{R}^{n\times b_1}$, while the rest of steps in Algorithm 1 do not change. As the size of the first block b_1 increases, $\widehat{\mathcal{S}}_1$ increasingly better approximates the true subspace $\mathcal{S}$. Indeed, one might consider $Y_1 = P_{\Omega_1}(S_1) = S_1 + (P_{\Omega_1}(S_1) - S_1)$ as a "noisy" copy of S_1, where the noise is due to the erasures. Roughly speaking then, as b_1 increases, the energy of the "signal" part, namely, $\|S_1\|_F$, grows faster than and eventually dominates the energy of the random noise $\|P_{\Omega_1}(S_1) - S_1\|_F$. This intuition is made precise by the following result which loosely speaking states that

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_1) \lesssim \sqrt{\frac{r}{pn}},$$

when $b_1 = \Omega(n)$. This result is proved in Appendix C with the aid of standard large deviation bounds.

Proposition 24 *Consider an r-dimensional subspace $\mathcal{S}$ with orthonormal basis $S \in \mathbb{R}^{n\times r}$. For an integer $b_1 \geq r$, let the coefficient vectors $\{q_t\}_{t=1}^{b_1} \subset \mathbb{R}^r$ be independent copies of a random vector $q \in \mathbb{R}^r$. For every $t \in [1:b_1]$, we observe each coordinate of $s_t = Sq_t \in \mathcal{S}$ independently with a probability of p and collect the observed entries in $y_t \in \mathbb{R}^n$, supported on the random index set $\omega_t \subseteq [1:n]$. We set $Q_1 = [q_1\, q_2 \cdots q_{b_1}] \in \mathbb{R}^{r\times b_1}$ and $Y_1 = [y_1\, y_2 \cdots y_{b_1}] \in \mathbb{R}^{n\times b_1}$ for short. Let also $\widehat{\mathcal{S}}_1$ be the span of leading r left singular vectors of Y_1.*

Then, for fixed $\alpha, \nu \geq 1$ and $1 \leq \eta \leq b_1/r$, it holds that

$$d_{\mathbb{G}}(\mathcal{S}, \widehat{\mathcal{S}}_1) \lesssim \alpha\cdot\nu\sqrt{\left(1 \vee \frac{n}{b_1}\right)\frac{\left(\eta \vee \eta\left(\mathcal{S}\right)\right) r\log(n \vee b_1)}{pn}}, \tag{226}$$

except with a probability of at most $e^{-\alpha} + \Pr[\nu(Q_1) > \nu] + \Pr[\eta(\mathcal{Q}_1) > \eta]$. Above, $\nu(Q_1)$ is the condition number of Q_1, $\eta(\mathcal{Q}_1)$ is the coherence of $\mathcal{Q}_1 = \mathrm{span}(Q_1^*)$ (see (14)), and $a \vee b := \max\{a, b\}$.

Remark 25 [Discussion of Proposition 24] As (226) suggests, for $\widehat{\mathcal{S}}_1$ to be close to $\mathcal{S}_1$, the first block should be a wide matrix, namely, $b_1 = O(n)$. This dependence on the block size was anticipated. Indeed, it is well-understood that one needs $O(n)$ samples in order

for the sample covariance matrix to closely approximate the covariance matrix of a random vector in $\mathbb{R}^n$ (Vershynin, 2012b). As an example, consider the case where the coefficient vectors $\{q_t\}_{t=1}^{b_1}$ are standard random Gaussian vectors and so $Q_1 \in \mathbb{R}^{n\times b_1}$ is a standard random Gaussian matrix, namely, populated with zero-mean independent Gaussian random variables with unit variance. Then both probabilities above are small when b_1 is sufficiently large. More specifically, we show in Appendix K that

$$\Pr[\nu(Q_1) > \nu] \leq \exp\left(-C\frac{\nu-1}{\nu+1}\right), \qquad \text{when } b \gtrsim r, \tag{227}$$

$$\Pr[\eta(\mathcal{Q}_1) > \eta] \leq \exp(-C\eta/\log b), \qquad \text{when } b\log^2 b \gtrsim \eta^2 r, \tag{228}$$

for a Gaussian coefficient matrix Q_1.

It is also worth noting that initializing SNIPE with a large first block can be done *without* increasing the storage requirement or computational complexity of SNIPE, namely, replacing the block size $b = O(r)$ in first step of Algorithm 1 with $b_1 = O(n)$ can be done without losing the streaming nature of SNIPE. More specifically, with the alternative initialization, naively computing the truncated SVD of the first block requires $O(b_1 n) = O(n^2)$ bits of storage and $O(rb_1 n) = O(rn^2)$ flops. These requirements can be significantly lowered by implementing a state-of-the-art streaming PCA algorithm such as the "power method" in (Mitliagkas et al., 2013). This suggests a two-phase algorithm. In the first phase, the power method is applied to the incoming data, where the missing entries are filled with zeros. This phase produces the estimate $\widehat{\mathcal{S}}_1$ in SNIPE, which serves as an initialization for the second phase in which the main loop of SNIPE is applied to the incoming blocks, producing the estimates $\{\widehat{\mathcal{S}}_k\}_{k\geq 2}$. If b_1 is sufficiently large, the first phase brings us within the basin of attraction of the true subspace $\mathcal{S}$ and activates the locally linear convergence of SNIPE to $\mathcal{S}$ in the second phase, see Theorems 8 and 10.

References

B. A. Ardekani, J. Kershaw, K. Kashikura, and I. Kanno. Activation detection in functional MRI using subspace modeling and maximum likelihood estimation. *IEEE Transactions on Medical Imaging*, 18(2):101–114, 1999.

R. Arora, A. Cotter, K. Livescu, and N. Srebro. Stochastic optimization for PCA and PLS. In *Annual Allerton Conference on Communication, Control, and Computing*, pages 861–868. IEEE, 2012.

A. Balsubramani, S. Dasgupta, and Y. Freund. The fast convergence of incremental PCA. In *Advances in Neural Information Processing Systems*, pages 3174–3182, 2013.

L. Balzano and S. J Wright. On GROUSE and incremental SVD. In *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 1–4. IEEE, 2013.

L. Balzano and S. J. Wright. Local convergence of an algorithm for subspace identification from partial data. *Foundations of Computational Mathematics*, 15(5):1279–1314, 2015.

L. Balzano, R. Nowak, and B. Recht. Online identification and tracking of subspaces from highly incomplete information. In *Annual Allerton Conference on Communication, Control, and Computing*, pages 704–711. IEEE, 2010.

M. Brand. Incremental singular value decomposition of uncertain data with missing values. In *European Conference on Computer Vision*, pages 707–720. Springer, 2002.

J. R. Bunch and C. P. Nielsen. Updating the singular value decomposition. *Numerische Mathematik*, 31(2):111–129, 1978.

C. Cartis and K. Scheinberg. Global convergence rate analysis of unconstrained optimization methods based on probabilistic models. *Mathematical Programming*, pages 1–39, 2017.

Y. Chen. Incoherence-optimal matrix completion. *IEEE Transactions on Information Theory*, 61(5):2909–2923, 2015.

Y. Chi, Y. C. Eldar, and R. Calderbank. PETRELS: Parallel subspace estimation and tracking by recursive least squares from partial observations. *IEEE Transactions on Signal Processing*, 61(23):5947–5959, 2013.

M. A. Davenport and J. Romberg. An overview of low-rank matrix recovery from incomplete observations. *IEEE Journal of Selected Topics in Signal Processing*, 10(4):608–622, 2016.

R. Durrett. *Probability: Theory and examples*. Cambridge university press, 2010.

C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1:211–218, 1936. doi: 10.1007/BF02288367.

A. Eftekhari, L. Balzano, and M. B. Wakin. What to expect when you are expecting on the Grassmannian. *IEEE Signal Processing Letters*, 24(6):872–876, 2017.

A. Eftekhari, M. B. Wakin, and R. A. Ward. MC^2: A two-phase algorithm for leveraged matrix completion. *Information and Inference: A Journal of the IMA*, 7(3):581–604, 2018a.

A. Eftekhari, D. Yang, and M. B. Wakin. Weighted matrix completion and recovery with prior subspace information. *IEEE Transactions on Information Theory*, 64(6):4044–4071, 2018b.

N. Gershenfeld, S. Samouhos, and B. Nordman. Intelligent infrastructure for energy efficiency. *Science*, 327(5969):1086–1088, 2010.

G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2013.

A. Gonen, D. Rosenbaum, Y. C. Eldar, and S. Shalev-Shwartz. Subspace learning with partial information. *Journal of Machine Learning Research*, 17(52):1–21, 2016.

D. Gross. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, 57(3):1548–1566, 2011.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer New York, 2013.

H. Krim and M. Viberg. Two decades of array signal processing research: The parametric approach. *IEEE Signal Processing Magazine*, 13(4):67–94, 1996.

A. Lakhina, M. Crovella, and C. Diot. Diagnosing network-wide traffic anomalies. In *ACM SIGCOMM Computer Communication Review*, volume 34, pages 219–230. ACM, 2004.

B. Lois and N. Vaswani. Online matrix completion and online robust PCA. In *IEEE International Symposium on Information Theory (ISIT)*, pages 1826–1830. IEEE, 2015.

K. Lounici. High-dimensional covariance matrix estimation with missing observations. *Bernoulli*, 20(3):1029–1058, 2014.

M. Mardani, G. Mateos, and G. B. Giannakis. Subspace learning and imputation for streaming big data matrices and tensors. *IEEE Transactions on Signal Processing*, 63(10):2663–2677, 2015.

L. Mirsky. Symmetric gauge functions and unitarily invariant norms. *Quart. J. Math. Oxford*, pages 1156–1159, 1966.

I. Mitliagkas, C. Caramanis, and P. Jain. Memory limited, streaming PCA. In *Advances in Neural Information Processing Systems*, pages 2886–2894, 2013.

I. Mitliagkas, C. Caramanis, and P. Jain. Streaming PCA with many missing entries. *Preprint*, 2014.

J. Nocedal and S. J. Wright. *Numerical optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006.

E. Oja and J. Karhunen. On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of Mathematical Analysis and Applications*, 106(1):69–84, 1985.

J. Tanner and K. Wei. Normalized iterative hard thresholding for matrix completion. *SIAM Journal on Scientific Computing*, 35(5):S104–S125, 2013.

L. Tong and S. Perreau. Multichannel blind identification: From subspace to maximum likelihood methods. *Proceedings of IEEE*, 86:1951–1968, 1998.

J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

P. van Overschee and B. L. de Moor. *Subspace identification for linear systems: Theory, implementation, applications*. Springer US, 2012.

R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. In Y. C. Eldar and G. Kutyniok, editors, *Compressed Sensing: Theory and Applications*, pages 95–110. Cambridge University Press, 2012a.

R. Vershynin. How close is the sample covariance matrix to the actual covariance matrix? *Journal of Theoretical Probability*, 25(3):655–686, 2012b.

S. Watanabe and N. Pakvasa. Subspace method of pattern recognition. In *Proc. 1st. IJCPR*, pages 25–32, 1973.

P. Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

Y. Xie, J. Huang, and R. Willett. Change-point detection for high-dimensional time series with missing data. *IEEE Journal of Selected Topics in Signal Processing*, 7(1):12–27, 2013.

D. Zhang and L. Balzano. Global convergence of a grassmannian gradient descent algorithm for subspace estimation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, page 1460Ð1468, 2016.

Journal of Machine Learning Research 20 (2019) 1-30 Submitted 11/17; Revised 10/18; Published 6/19

An asymptotic analysis of distributed nonparametric methods

Botond Szabó B.T.SZABO@MATH.LEIDENUNIV.NL
Mathematical Institute
Leiden University
2333 CA Leiden
The Netherlands

Harry van Zanten HVZANTEN@UVA.NL
Korteweg-de Vries Institute for Mathematics
University of Amsterdam
Science Park 105-107
1098 XG Amsterdam
The Netherlands

Editor: Manfred Opper

Abstract

We investigate and compare the fundamental performance of several distributed learning methods that have been proposed recently. We do this in the context of a distributed version of the classical signal-in-Gaussian-white-noise model, which serves as a benchmark model for studying performance in this setting. The results show how the design and tuning of a distributed method can have great impact on convergence rates and validity of uncertainty quantification. Moreover, we highlight the difficulty of designing nonparametric distributed procedures that automatically adapt to smoothness.

Keywords: distributed learning, nonparametric models, high-dimensional models, Gaussian processes, convergence rates

1. Introduction

Both in statistics and machine learning there has been substantial interest in the design and study of distributed statistical or learning methods in recent years. One driving reason is the fact that in certain applications datasets have become so large that it is often unfeasible, or computationally undesirable, to carry out the analysis on a single machine. In a distributed method the data are divided over a cluster consisting of several machines and/or cores. The machines in the cluster then process their data locally, after which the local results are somehow aggregated on a central machine to finally produce the overall outcome of the statistical analysis. Distributed methods are not only used for computational reasons, but are for instance also of interest in situations where privacy is important and it is undesirable that all data are handled at a single location. Moreover, there are applications in which data are by construction gathered at multiple locations and first processed locally, before being combined at a central location.

©2019 Botond Szabó and Harry van Zanten.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-666.html.

Over the last years a variety of distributed methods have been proposed. Recent examples include Consensus Monte Carlo (Scott et al. (2016)), WASP (Srivastava et al. (2015)), distributed GP's (Deisenroth and Ng (2015)), and methods proposed in Shang and Cheng (2015), Jordan et al. (2016), Lee et al. (2015), Volgushev et al. (2017), to mention but a few. Most papers on distributed methods do extensive experiments on simulated, benchmark and real data to numerically assess and compare the performance of the various methods. Some papers also derive a number of theoretical properties. Theoretical results on the performance of distributed methods are not yet widely available however and there is certainly no common theoretical framework in place that allows a clear theoretical comparison of methods and the development of an understanding of fundamental performance guarantees and limitations. Clearly we can not consider the complete list of all existing methods in this paper. We limit ourselves to a number of representative methods that are Bayesian in nature, allowing for a meaningful comparison.

Since a better theoretical understanding of distributed methods can help to pinpoint fundamental difficulties and opportunities, we develop a framework in this paper which allows us to study and compare the performance of various methods. We are in particular interested in high-dimensional, or nonparametric problems. It is by now well known that the performance of learning or statistical methods in such settings depends crucially on wether or not a method succeeds in realizing the correct bias-variance trade-off, or, in different terminology, succeeds in balancing under- and overfitting. For classical, non-distributed settings we have a rather well-developed understanding of how methods should be tuned to achieve a proper bias-variance trade-off. For distributed methods however, such theory is currently not yet available.

To be able to develop relevant theory we study an idealized model, which is a distributed version of the canonical "signal-in-white-noise" model that serves as an important benchmark model in mathematical statistics (see for instance Tsybakov (2009); Johnstone (2017); Giné and Nickl (2016)). The model is on the one hand rich enough to be interesting, in the sense that it is really distributed in nature and the unknown object that needs to be learned is truly infinite-dimensional. On the other hand it is tractable enough to allow detailed mathematical analysis. In the non-distributed case the signal-in-white-noise model is well known to be very closely related to other nonparametric models, such as nonparametric regression and density estimation. (This can be made very precise in the context of Le Cam's theory of limits of experiments (e.g. Le Cam (2012), Brown and Low (1996), Nussbaum (1996)).) Similarly, the distributed signal-in-white-noise model that we consider in this paper provides a unified framework to compare methods that were originally introduced in different settings. Therefore, although our theoretical results are all obtained in the context of this relatively simple "benchmark model" and we consider only a number of representative distributed methods, we believe that the conclusions that we draw are relevant more generally. We introduce the model in Section 2.

It is not difficult to see that if the number of machines m is relatively large with respect to the total sample size, or signal-to-noise-ratio n, then doing things completely naively in the distributed case leads to a sub-optimal bias-variance trade-off (see also the simulation example in Section 2). In particular, just computing the "usual" estimators on every local machine and then averaging them on the central machine typically leads to a global estimator with a bias that is too large. To achieve good performance, the trade-off has to be

adjusted somehow. This can in principle be done in various ways. For instance by locally choosing the "wrong" settings for tuning parameters on purpose, or, in a Bayesian setting, by adjusting the likelihood (e.g. raising it to some power) or by adjusting the prior. In Section 3 we study to what degree various methods that have been proposed in the literature succeed in ultimately achieving the right trade-off. We will see that some are more successful than others in this respect.

An important observation that we make is that the methods that are shown to work well in Section 3 all use information on aspects of the true signal that are in principle unknown, such as its degree of regularity. A key question is whether in distributed settings it is fundamentally possible to set tuning parameters correctly in a purely data-driven way, without using such information. In the non-distributed setting it is well known that such adaptive methods indeed exist (e.g. Tsybakov (2009) or Giné and Nickl (2016)). In the distributed case that we study here however, this is much less clear. In Section 4 we show that using a distributed version of a standard adaptation method that is known to work in the non-distributed case, such as maximum marginal likelihood empirical Bayes, can lead to sub-optimal results in the distributed setting. We will argue that this seems to be a fundamental issue and that we expect that correct automatic setting of tuning parameters in distributed methods is fundamentally more challenging than in the classical, non-distributed case. We believe this is an important issue and want to highlight it as an important and interesting topic for future research.

To further study the fundamental potential and limitations of distributed methods, one should also take into account that there are typically computational and/or communication cost restrictions in such settings. In fact, without such restrictions the matter of obtaining good strategies is essentially non-existent, since we could simply communicate all data to a central machine and then apply an existing optimal non-distributed method. The main goal of this paper, however, is to study and compare the theoretical performance of a number of *existing* distributed methods. The methods we consider are all divide-and-conquer algorithms in which computational or communication limitations are not explicitly imposed, however such restrictions do motivate the methods and all methods implicitly or explicitly consider situations in which simply aggregating and handling all data in one machine is not an option. Formally taking into account communication restrictions in the theoretical analysis is an important next step, but turns out to be rather delicate. We recently obtained some first results in the follow-up paper Szabo and van Zanten (2019), see also Zhu and Lafferty (2018) for related work.

The remainder of the paper is organized as follows. In the next section we introduce the distributed version of the signal-in-white-noise model and provide a simple simulation example to show that in a distributed setting, naively combining inferences from local machines into a global estimator may produce misleading results. In Section 3 we study the performance of a number of Bayesian procedures for signal reconstruction in the distributed signal-in-white-noise model introduced in Section 2. We include a number of methods that have recently been proposed in the literature. We show that some succeed in obtaining the appropriate bias-variance trade-off, but others do not. Moreover, the ones that do produce good results are all non-adaptive, in the sense that they use knowledge of the smoothness of the unkown signal to set their tuning parameters. In the final Section 4 we consider the more realistic setting in which this smoothness is unknown. We study a distributed method

that has been proposed for data-driven tuning of the hyperparameters and show that there exist "difficult signals", which this method can not recover in the distributed model at an optimal rate. We argue that this appears to be a fundamental issue, and that designing procedures that automatically adapt to smoothness is fundamentally more challenging in the distributed framework. Mathematical proofs are collected in appendix Sections A and B.

2. Distributed signal-in-white-noise model

Consider the problem of estimating a signal in Gaussian white noise. This is the continuous regression-type problem in which we observe a signal $X = (X_t : t \in [0,1])$ satisfying a stochastic differential equation

$$X_t = \int_0^t f(s)\,ds + \frac{\sigma}{\sqrt{n}} W_t,$$

where W is a Brownian motion, modelling the "white noise", and f is an unknown signal, modelled by a square integrable function, that needs to be recovered from the data. The natural number n is the signal-to-noise ratio. Its size affects the difficulty of the problem and n can be seen as playing the role of sample size in this problem.

By expanding the data in a fixed orthonormal basis of $L^2[0,1]$ (for instance the classical Fourier basis), it is seen that the statistical problem of recovering the signal f from the data X is equivalent to the problem of recovering the sequence of (Fourier) coefficients $\theta \in \ell^2$ from noisy observations $Y_1, Y_2, \ldots$, satisfying

$$Y_i = \theta_i + \sqrt{\frac{\sigma^2}{n}} Z_i, \qquad i = 1, 2, \ldots, \tag{2.1}$$

where the Z_i are independent standard normal variables. This is the usual setting in which there is a single observer that observes every coefficient θ_i with additive Gaussian noise with variance σ^2/n. See for instance Tsybakov (2009); Johnstone (2017); Giné and Nickl (2016) for more details on this classical model.

In the distributed version of the model we divide the "precision budget" n over m different observers, so that each one observes the signal in Gaussian noise with variance $\sigma^2 m/n$, independent of the others. In other words, observer j has data $Y_1^j, Y_2^j, \ldots$ satisfying

$$Y_i^j = \theta_i + \sqrt{\frac{\sigma^2 m}{n}} Z_i^j, \qquad i = 1, 2, \ldots, \tag{2.2}$$

where the Z_i^j are independent, standard Gaussian random variables. We call the m independent sub-problems in which the signal-to-noise ratio is $\sigma^2 m/n$ the "local" problems.

Note that the classical, non-distributed signal-in-white-noise model is obtained again from the distributed model by aggregating all the local data. Indeed, if for $j = 1, 2, \ldots$ we define $Y_i = m^{-1} \sum_{j=1}^m Y_i^j$, then (2.1) holds, with $Z_i = m^{-1/2} \sum_{j=1}^m Z_i^j$ independent standard normal variables. This model has been studied extensively in the literature, serving as a canonical model for understanding the performance of high-dimensional or nonparametric

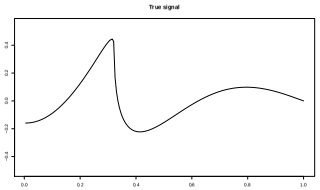


Figure 1: True signal.

statistical procedures. It is well known for instance that if the true signal θ belongs to an ellipsoid or a hyper rectangle of the form

$$\{\theta \in \ell^2 : \sum i^{2\beta}\theta_i^2 \leq M^2\} \text{ or } \{\theta \in \ell^2 : \sup_i(i^{1+2\beta}\theta_i^2) \leq M^2\}$$

for some $\beta, M > 0$, then the optimal rate of convergence of estimators (relative to the ℓ^2-norm) is of the order $n^{-\beta/(1+2\beta)}$. Moreover, there exist so-called adaptive estimators, which achieve this rate without using knowledge about the parameters β or M that describe the complexity, or regularity of the true signal. See, for instance, Tsybakov (2009) or Giné and Nickl (2016). Our central question is whether or not the same results can be obtained in the distributed setting in which each of the m different observers first separately make inference about the signal, and then the local estimates are aggregated into one joint estimator. For technical convenience we will quantify regularity using hyper rectangles in the remainder of the paper, but the analoguous results can be obtained for ellipsoids.

The specific examples of distributed procedures that we consider in this paper are about distributed Bayesian methods. These methods have in common that each local observer first chooses a prior distribution and computes the corresponding local posterior distribution using the local data (or an appropriate modification). In the next step the m local posteriors are somehow aggregated into a global posterior-type distribution, which is then used to produce an estimate of the signal and/or a quantification of the associated uncertainty. In general there is no guarantee that this "aggregated posterior" resembles the posterior distribution that would be obtained in the non-distributed setting, using all the data at once. In particular, it is not clear beforehand how a distributed Bayes method should be constructed in order to have good theoretical properties, like optimal convergence rates, reliable uncertainty quantification or adaptation properties. In this paper we investigate various distributed methods that have been proposed from this point of view.

To see that interesting things can happen it is exemplifying to compare the results of a distributed and a non-distributed (Bayesian) analysis of simulated data. Concretely, we consider a true signal θ consisting of the Fourier coefficients of the function shown in Figure 1. For this signal we simulate data according to (2.2), with $\sigma = 1$, $m = 40$ and $n = 120 \times 40 = 4800$. Then for every local observer a Bayesian procedure is carried out with a Gaussian prior on θ, postulating that the coordinates θ_i are independent and $N(0, i^{-1-2\alpha})$-distributed. The hyperparameter α, which describes the regularity of the prior,

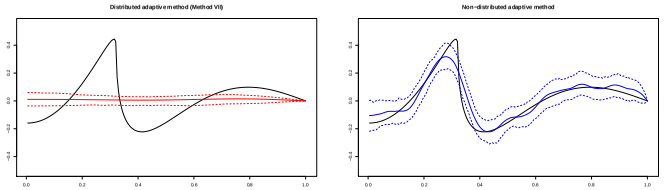


Figure 2: Signal reconstruction using the distributed method (left) and the non-distributed method (right).

is determined using a distributed version of maximum marginal likelihood, as described in Section 4. This analysis leads to $m = 40$ local posterior distributions. These are then combined to produce an overall posterior distribution for the signal. The precise procedure is described in Section 4. The resulting estimator for the signal, together with pointwise 95% credible intervals, is shown in the left plot in Figure 2. The corresponding non-distributed result is obtained by first aggregating all local data as in (2.1) and then carrying out the same Bayesian procedure on these complete data. The resulting non-distributed reconstruction of the signal is shown on the right in Figure 2.

The non-distributed version of this method was studied theoretically for instance in Knapik et al. (2016) and Szabó et al. (2015), where it was shown that the method is adaptive and rate-optimal. The simulation suggests however that an apparently reasonable distributed analogue of the method does not necessarily inherit these favourable properties. The procedure seems to be underfitting and the credible intervals appear to be too narrow. We will argue that this is in some sense a fundamental issue and in the next sections we will study various proposed distributed methods to investigate to what degree they succeed in avoiding or solving these problems.

3. Results for non-adaptive procedures

In this section we study the performance of a number of Bayesian procedures for signal reconstruction in the distributed signal-in-white-noise model introduced in Section 2. All methods involve putting a prior distribution on the unknown signal $\theta \in \ell^2$ in each local problem and then combining the resulting local posteriors into one global posterior-type distribution. To be able to compare the various methods we consider the same Gaussian process (GP) prior in every case, namely the prior

$$\Pi(\cdot|\alpha) = \bigotimes_{i=1}^{\infty} N(0, i^{-1-2\alpha}), \tag{3.1}$$

which postulates that the coefficients θ_i of the signal θ are independent and $N(0, i^{-1-2\alpha})$-distributed. The hyper parameter $\alpha > 0$ essentially controls the regularity of the prior, since

it basically controls how fast the Fourier coefficients decrease. (Some of the methods we consider use exactly this prior, others modify it in a certain way with the aim of achieving better performance.) The global posterior-type distribution depends on all the data $\mathbf{Y} = (Y_i^j : j = 1, \ldots, m; i = 1, 2, \ldots)$ and is denoted by $\Pi(\cdot \,|\, \mathbf{Y})$. It is generally some type of average of the local posteriors, but its precise construction differs between proposed methods. We will see that this can have a significant effect on the performance.

We take an asymptotic perspective and investigate in every case the rate at which the global posterior contracts around the true signal as $n \to \infty$ relative to the ℓ^2-norm, which is as usual defined by $\|\theta\|_2^2 = \sum \theta_i^2$. For a sequence of positive numbers $\varepsilon_n \to 0$ we say that the global posterior contracts at the rate ε_n around the true signal θ_0 if for all sequences $M_n \to \infty$,

$$\mathrm{E}_{\theta_0} \Pi(\theta \in \ell^2 : \|\theta - \theta_0\|_2 > M_n \varepsilon_n \,|\, \mathbf{Y}) \to 0$$

as $n \to \infty$. This means that asymptotically, all posterior mass is concentrated in balls around the true signal θ_0 with ℓ^2-radius of the order ε_n.

Additionally, we study how well the posterior quantifies the remaining uncertainty. Specifically, we consider the coverage probabilities of credible balls around the global posterior mean. These credible sets are constructed by first computing the mean $\hat{\theta}$ of the global "posterior" $\Pi(\cdot \,|\, \mathbf{Y})$. Then for a level $\gamma \in (0, 1)$, the posterior is used to determine the radius r_γ such that the ball around $\hat{\theta}$ with radius r_γ receives $1 - \gamma$ posterior mass, i.e.

$$\Pi(\theta : \|\theta - \hat{\theta}\|_2 \leq r_\gamma | \mathbf{Y}) = 1 - \gamma.$$

For $L > 0$, the credible set $\hat{C}(L)$ is subsequently defined by

$$\hat{C}(L) = \{\theta : \|\theta - \hat{\theta}_n\|_2 \leq L r_\gamma\}. \tag{3.2}$$

(The extra constant L gives some added flexibility, for $L = 1$ we obtain an exact $1 - \gamma$ credible set.) We are interested in the coverage probabilities $\mathrm{P}_{\theta_0}(\theta_0 \in \hat{C}(L))$. If this tends to 0 as $n \to \infty$, the credible sets are asymptotically not frequentist confidence sets, hence give a misleading quantification of the uncertainty. Ideally, the coverage probabilities stay bounded away from 0 as $n \to \infty$.

In the non-distributed case $m = 1$ it is well known that both the rate at which the posterior contracts around the truth and the behaviour of the coverage probabilities of credible sets depend crucially on how the hyper parameter α is tuned. The correct bias-variance trade-off is achieved if α is in accordance with the regularity of the unknown signal. To make this precise, we will consider signals belonging to hyper rectangles of the form

$$H^\beta(M) = \left\{\theta \in \ell^2 : \sup_i (i^{1+2\beta} \theta_i^2) \leq M^2\right\} \tag{3.3}$$

for some $\beta, M > 0$. It is shown for instance in Knapik et al. (2011) for the non-distributed case that if $\theta_0 \in H^\beta(M)$ and we set $\alpha = \beta$, then the posterior contracts around θ_0 at the optimal rate $n^{-\beta/(1+2\beta)}$. Moreover, for L large enough it then holds that $\mathrm{P}_{\theta_0}(\theta_0 \notin \hat{C}(L)) \leq \gamma$. Hence, in the non-distributed case it is optimal to choose the hyper parameter α in such a way that the regularity α of the prior matches the regularity β of the true signal.

In the remainder of this section we investigate distributed methods from this point of view. We will see that the different proposed methods lead to different behaviours in terms

of contraction rates and coverage. We stress that the results in this section are non-adaptive, in the sense that we allow the tuning parameter α and other aspects of the constructions to use knowledge of the regularity β of the true signal. This is of course not realistic. It is however important to first understand for every method whether ideally, if the value of β is given to us by an oracle, it is possible to tune the method optimally. Whether this is also possible adaptively, without knowing β, is then the next natural question, which we address in Section 4.

3.1. Naive averaging of local posterior draws

Recall that we have m local observers that each have a dataset $\mathbf{Y}^j = (Y_1^j, Y_2^j, \ldots)$ of noisy coefficients satisfying (2.2). The aim is to recover the true sequence of coefficients θ.

As a starting point, and to have a baseline case to compare the other methods to, we analyse the naive distributed approach in which in every local problem we simply use the prior $\Pi(\cdot \,|\, \alpha)$ defined by (3.1), with $\alpha = \beta$ equal to the regularity of the true sequence θ, in the sense of (3.3). Every local observer then computes its corresponding local posterior, $\Pi^j(\cdot \,|\, \mathbf{Y}^j)$. By Bayes' formula this is given by

$$d\Pi^j(\theta \,|\, \mathbf{Y}^j) \propto p(\mathbf{Y}^j \,|\, \theta)\, d\Pi(\theta \,|\, \beta),$$

where the likelihood for the jth local problem is given by

$$p(\mathbf{Y}^j \,|\, \theta) \propto \prod e^{-\frac{1}{2} \frac{n (Y_i^j - \theta_i)^2}{\sigma^2 m}}. \tag{3.4}$$

Finally these local posteriors are combined into a global, "average posterior" $\mathbf{\Pi}_I(\cdot \,|\, \mathbf{Y})$ by postulating that a draw from this global posterior is generated by first drawing once from each local posterior and then averaging these m independent draws. (Formally, this means that the global "posterior" $\mathbf{\Pi}_I(\cdot \,|\, \mathbf{Y})$ is the convolution of the rescaled local posteriors $\Pi^1(m \times \cdot \,|\, \mathbf{Y}^1), \ldots, \Pi^m(m \times \cdot \,|\, \mathbf{Y}^m)$).

This distributed method is conceptually very simple, but it turns out that neither from the point of view of contraction rates, nor from the point of view of uncertainty quantification it performs very well. The reason is basically that although the choice $\alpha = \beta$ of the tuning parameter of the prior correctly matches squared bias, variance and posterior spread in the local problems, the averaging procedure results in a global "posterior" for which the spread and the variance of the mean are too small relative to the squared bias. The following theorem asserts that for every smoothness level $\beta > 0$ there exist β-regular truths for which the contraction rate of the posterior deteriorates substantially and for which the uncertainty quantification by the credible sets (3.2) constructed from the global posterior $\mathbf{\Pi}_I(\cdot \,|\, \mathbf{Y})$ is useless, no matter how far they are blown up by a constant $L > 0$.

Theorem 1 (naive averaging) *For every $\beta, M > 0$ there exists a $\theta_0 \in H^\beta(M)$ such that for small enough $c > 0$,*

$$E_{\theta_0} \mathbf{\Pi}_I(\theta : \|\theta - \theta_0\|_2 \leq c m^{\frac{\beta}{1+2\beta}} n^{-\frac{\beta}{1+2\beta}} | \beta, \mathbf{Y}) \to 0$$

as $m \to \infty$ and $n/m \to \infty$. Furthermore, for all $L > 0$ it holds that

$$P_{\theta_0}\big(\theta_0 \in \hat{C}(L)\big) \to 0.$$

Proof The proof of the theorem is given in Section A.1. ■

In the literature several less naive distributed strategies have been proposed. These methods either change the local likelihoods in a certain way, and/or the priors that are locally used, and/or the way that the local posteriors are aggregated. In the next few sections we investigate whether such strategies can improve the bad asymptotic performance of the naive averaging method.

3.2. Adjusted local likelihoods and averaging

One perspective on the bad performance of the naive method is to say that since the "sample size" n/m in the local problems is too small, the influence of the data on the local posterior is too small, resulting in a variance (and spread) that is too small relative to the squared bias in the global posterior. A possible way to remedy this that has been proposed in several papers is to raise the local likelihoods to the power m, in order to mimic the situation that we have sample size n in the local problems. This generalized Bayesian approach for the local problems has for instance been considered in the distributed context by Srivastava et al. (2015). They combine it with a different aggregation method however, which we consider in Section 3.4. In this section we still consider the simple averaging scheme, in order to isolate the effect of adjusting the local likelihoods.

So in method II all local observers use the prior $\Pi(\cdot\,|\,\alpha)$ again, with $\alpha = \beta$ equal to the regularity of the truth. They now each compute a generalized local posterior $\tilde{\Pi}^j(\cdot\,|\,\mathbf{Y}^j)$, defined by

$$d\tilde{\Pi}^j(\theta\,|\,\mathbf{Y}^j) \propto \Big(p(\mathbf{Y}^j\,|\,\theta)\Big)^m d\Pi(\theta\,|\,\beta).$$

As before the global "posterior" $\mathbf{\Pi}_{II}(\cdot\,|\,\mathbf{Y})$ is defined by postulating that a draw from this global posterior is generated by first drawing once from each local generalized posterior and then averaging these m independent draws.

The following theorem states that this method indeed improves the naive approach of Section 3.1. The global posterior now contracts at the optimal rate for every β-regular truth. Unfortunately, the bad behaviour of the credible sets has not been remedied. For this approach the uncertainty quantification is in fact misleading for all β-regular truths.

Theorem 2 (adjusted likelihoods + averaging) *For all $\beta, M > 0$ and all sequences $M_n \to \infty$,*

$$\sup_{\theta_0 \in H^\beta(M)} \mathrm{E}_{\theta_0} \mathbf{\Pi}_{II}(\theta : \|\theta - \theta_0\|_2 \geq M_n n^{-\frac{\beta}{1+2\beta}} | \mathbf{Y}) \to 0$$

as $n, m \to \infty$. However, for all $\theta_0 \in H^\beta(M)$ and all $L > 0$ it holds that

$$\mathrm{P}_{\theta_0}\big(\theta_0 \in \hat{C}(L)\big) \to 0.$$

Proof The proof is given in Section A.2. ■

3.3. Adjusted priors and averaging

Adjusting the likelihood as in the preceding section resulted in a correct trade-off between the bias and the variance of the global posterior mean, yielding an optimal posterior contraction rate. The spread of the posterior remained too small in comparison however, resulting in credible sets with zero asymptotic coverage. Instead of raising the local posteriors to the power m, as considered in the preceding section, we could alternatively raise the prior density to the power $1/m$. This has for instance been proposed in the context of the "Consensus Monte Carlo" approach by Scott et al. (2016), in combination with simple averaging of the local posteriors. In this section we investigate the performance of this method in terms of posterior contraction and uncertainty quantification in our distributed signal-in-white-noise model.

The prior $\Pi(\cdot\,|\,\alpha)$ that we use in the local problems is again a product of centered Gaussians with variance $i^{-1-2\alpha}$. Raising the corresponding densities to the power $1/m$ has the effect of multiplying the ith prior variance by m. Hence, in our case raising the prior density to the power $1/m$ is the same as multiplicative rescaling, postulating that θ is a-priori distributed according to $\Pi(\cdot\,|\,\alpha, m)$, where

$$\Pi(\cdot|\alpha, \tau) = \bigotimes_{i=1}^{\infty} N(0, \tau i^{-1-2\alpha}) \tag{3.5}$$

for $\alpha, \tau > 0$. Rescaled GPs have also been considered by Shang and Cheng (2015), who have used them in the distributed setting to construct global credible sets from local ones.

Using rescaling we can actually obtain good results if the prior regularity α is not exactly equal to the true regularity β. By using a scaling different from $\tau = m$ we can somehow compensate for the mismatch between α and β, at least in the range $\beta \leq 1 + 2\alpha$. In the non-distributed setting this is a well-known phenomenon, see for instance van der Vaart and van Zanten (2007); Knapik et al. (2011); Szabó et al. (2013).

The distributed procedure that we consider in this section then takes the following form. Every local observer uses the rescaled prior $\Pi(\cdot|\alpha, \tau)$ defined by (3.5), with $\alpha > 0$ and

$$\tau = m n^{\frac{2(\alpha-\beta)}{1+2\beta}},$$

where β is the regularity of the truth. Next the (normal, unadjusted) corresponding posteriors are computed and they are averaged into a global "posterior" $\mathbf{\Pi}_{III}(\cdot\,|\,\mathbf{Y})$ as in the preceding sections. (Note that if in the local problems the prior regularity $\alpha = \beta$ is used, then $\tau = m$, so the method corresponds to raising the prior density to the power $1/m$.)

The following theorem gives the posterior contraction and coverage results for this method.

Theorem 3 (adjusted priors + averaging) *Suppose $\beta, M > 0$ and $\beta \leq 1 + 2\alpha$. Then for all sequences $M_n \to \infty$,*

$$\sup_{\theta_0 \in H^\beta(M)} \mathrm{E}_{\theta_0} \mathbf{\Pi}_{III}(\theta : \|\theta - \theta_0\|_2 > M_n n^{-\frac{\beta}{1+2\beta}} | \mathbf{Y}) \to 0$$

as $n \to \infty$. *Moreover, for all* $\gamma \in (0,1)$ *it holds that*

$$\sup_{\theta_0 \in H^\beta(M)} \mathrm{P}_{\theta_0}\Big(\theta_0 \notin \hat{C}(L)\Big) \leq \gamma$$

for large enough $L > 0$.

Proof See Section A.3. ■

So adjusting the prior in this way actually works better than adjusting the likelihood. Not only do we get optimal contraction rates, but the credible sets that this method produces have asymptotic frequentist coverage too. The proof shows that the credible sets have optimal radius of the order $n^{-\beta/(1+2\beta)}$ as well.

3.4. Adjusted local likelihoods and Wasserstein barycenters

In Section 3.2 we saw that raising the local likelihoods to the power m and then averaging the corresponding generalized posteriors yields optimal contraction rates, but can produce badly performing credible sets. In this section we study the approach considered by Minsker et al. (2014); Srivastava et al. (2015) in the context of their "WASP" method, which consists in aggregating the local posteriors not by simple averaging, but by computing their Wasserstein barycenter.

The generalized local posteriors $\tilde{\Pi}^j(\cdot \,|\, \mathbf{Y}^j)$, as defined in Section 3.2, are (Gaussian) measures on ℓ^2. The 2-Wasserstein distance $W_2(\mu,\nu)$ between two probability measures μ and ν on ℓ^2 is defined by

$$W_2^2(\mu,\nu) = \inf_\gamma \iint \|x-y\|_2^2 \, \gamma(dx,dy),$$

where the infimum is over all measures γ on $\ell^2 \times \ell^2$ with marginals μ and ν. The corresponding 2-Wasserstein barycenter of m probability measures $\mu_1, \dots, \mu_m$ on ℓ^2 is then defined by

$$\bar{\mu} = \operatorname*{argmin}_\mu \frac{1}{m} \sum_{j=1}^m W_2^2(\mu, \mu_j),$$

where the minimum is over all probability measures on ℓ^2 with finite second moments. There exist effective algorithms to compute Wasserstein barycenters in many cases, see for instance Cuturi and Doucet (2014) and the references therein.

Having this notion at our disposal the distributed method we consider in this section proceeds as follows. In every local problem the prior $\Pi(\cdot \,|\, \alpha)$ is used, with $\alpha = \beta$ equal to the regularity of the truth. Next, the corresponding generalized posteriors $\tilde{\Pi}^j(\cdot \,|\, \mathbf{Y}^j)$ are computed locally, which involves raising the likelihood to the power m as described in Section 3.2. Finally, the global "posterior" $\mathbf{\Pi}_{IV}(\cdot \,|\, \mathbf{Y})$ is constructed as the 2-Wasserstein barycenter of the local measures $\tilde{\Pi}^1(\cdot \,|\, \mathbf{Y}^1), \dots, \tilde{\Pi}^m(\cdot \,|\, \mathbf{Y}^m)$.

The following theorem asserts that this method results in optimal posterior contraction rates and correct quantification of uncertainty.

Theorem 4 (adjusted likelihoods + barycenters) *For all* $\beta, M > 0$ *and all sequences* $M_n \to \infty$,

$$\sup_{\theta_0 \in H^\beta(M)} \mathrm{E}_{\theta_0} \mathbf{\Pi}_{IV}(\theta : \|\theta - \theta_0\|_2 > M_n n^{-\frac{\beta}{1+2\beta}} \,|\, \mathbf{Y}) \to 0$$

as $n \to \infty$. *Moreover, for all* $\gamma \in (0,1)$ *it holds that*

$$\sup_{\theta_0 \in H^\beta(M)} \mathrm{P}_{\theta_0}\Big(\theta_0 \notin \hat{C}(L)\Big) \leq \gamma$$

for large enough $L > 0$.

Proof See Section A.4. ■

3.5. Product of Gaussian process experts

The proofs of the theorems presented so far show that since in our context the global "posterior" is always a Gaussian measure, the behaviour of the procedure can be understood by analyzing three central quantities: the bias of the posterior mean, the variance of the posterior mean, and the spread of the posterior. Depending on how these quantities are related we have found different behaviours: sub-optimal posterior contraction and bad coverage of credible sets (Section 3.1), optimal posterior contraction but bad coverage of credible sets (Section 3.2), and optimal posterior contraction and also good coverage of credible sets (Sections 3.3 and 3.4).

In principle it is now straightforward to analyze different methods as well, provided the three central quantities can be controlled. As an illustration we consider in this section the single-layer version of the product-of-Gaussian-process-expert (PoE) model, introduced in Ng and Deisenroth (2014) and a generalization proposed in Cao and Fleet (2014). An interesting fact is that we will encounter a combination of behaviours that we have not seen yet: sub-optimal contraction rates, but good coverage of credible sets. These methods were introduced to deal with the distributed non-parametric regression model, but for the sake of comparison we analyze them in the context of our distributed signal-in-white-noise model, which can be thought of as an idealized version of the regression model.

The idea of the basic version of the Gaussian PoE model is to employ a Gaussian prior in every local machine, compute the corresponding posterior densities and approximate the global posterior density by multiplying and normalizing these. In our infinite-dimensional setting this does not make sense strictly speaking, since we can not express priors and posteriors on ℓ^2 in terms of densities with respect to some generic dominating measure. We could remedy this by considering a truncated version of our distributed model, where we assume we only observe the first n noisy coefficients Y_i^j in every machine, say, and focus on making inference about the first n true coefficients θ_i. This would make the setting finite-dimensional, allowing us to write prior and posterior densities with respect to the Lebesgue measure. Alternatively, we can stay in the infinite-dimensional setting of the paper and just reason formally and still arrive at a well-defined global PoE "posterior". This is the approach we follow here.

Indeed, say that as before we use the prior $\Pi(\cdot \,|\, \alpha)$ given by (3.1) in every local machine, with $\alpha = \beta$ equal to the regularity of the true signal. This prior has formal "density" proportional to

$$\theta \mapsto \prod e^{-\frac{1}{2}\frac{\theta_i^2}{i^{-1-2\beta}}}.$$

By completing the square we see that the product of this expression with the local likelihood given by (3.4) is, still formally, proportional to

$$\theta \mapsto \prod e^{-\frac{1}{2}\frac{\theta_i^2}{i^{-1-2\beta}}} e^{-\frac{1}{2}\frac{n(Y_i^j-\theta_i)^2}{\sigma^2 m}} \propto \prod e^{-\frac{1}{2}\theta_i^2(i^{1+2\beta}+\frac{n}{m\sigma^2})+\theta_i\frac{nY_i^j}{m\sigma^2}}.$$

Taking the product over j we then obtain the formal density of the PoE posterior, which is proportional to

$$\theta \mapsto \prod e^{-\frac{1}{2}\theta_i^2(mi^{1+2\beta}+\frac{n}{\sigma^2})+\theta_i\frac{n\sum_{j=1}^m Y_i^j}{m\sigma^2}}.$$

Now this last expression is, up to a constant, the density of a product of Gaussians with means $\hat{\theta}_i$ and variances t_i^2 given by

$$\hat{\theta}_i = \frac{nm^{-1}\sum Y_i^j}{n+\sigma^2 mi^{1+2\beta}}, \qquad t_i^2 = \frac{\sigma^2}{n+\sigma^2 mi^{1+2\beta}}.$$

The latter is in fact a well-defined Gaussian measure on ℓ^2, so we can now simply define the global PoE "posterior" $\mathbf{\Pi}_V(\cdot \,|\, \mathbf{Y})$ as the latter measure.

We see that the expressions for the global mean and spread are in fact the same as what we found in Section A.1 for the naive averaging method. As a consequence, the negative result of Theorem 1 holds for the basic version of the Gaussian PoE model as well.

Theorem 5 (product of Gaussian experts) *For every $\beta, M > 0$ there exists a $\theta_0 \in H^\beta(M)$ such that for small enough $c > 0$,*

$$\mathrm{E}_{\theta_0}\mathbf{\Pi}_V(\theta : \|\theta-\theta_0\|_2 \leq cm^{\frac{\beta}{1+2\beta}}n^{-\frac{\beta}{1+2\beta}} \,|\, \mathbf{Y}) \to 0$$

as $m \to \infty$ and $n/m \to \infty$. Furthermore, for all $L > 0$ it holds that

$$\mathrm{P}_{\theta_0}\big(\theta_0 \in \hat{C}(L)\big) \to 0.$$

One can generalize the PoE model by raising the local posterior densities to some power before multiplying and normalizing them, as proposed in Cao and Fleet (2014). In the subsequent analysis we consider the choice $1/m$ for the power, as suggested in Deisenroth and Ng (2015). Adapting the preceding analysis for the ordinary PoE model we see that for this generalized PoE model the global "posterior" $\mathbf{\Pi}_{VI}(\cdot \,|\, \mathbf{Y})$ is in our setting again a product of Gaussians, but now with means and variances given by

$$\hat{\theta}_i = \frac{nm^{-1}\sum Y_i^j}{n+\sigma^2 mi^{1+2\beta}}, \qquad t_i^2 = \frac{\sigma^2 m}{n+\sigma^2 mi^{1+2\beta}}.$$

So the global posterior mean is unaltered compared to the basic PoE model, but the global posterior spread has been blown up by a factor m. As a result, there still exists the same

Method	Description	Optimal rate	Coverage
I	naive averaging	no	no
II	adjusted likelihoods, averaging	yes	no
III	adjusted priors, averaging	yes	yes
IV	adjusted likelihoods, barycenter	yes	yes
V	product of experts	no	no
VI	generalized product of experts	no	yes

Table 1: Performance of the various non-adaptive methods.

class of truths as in Section A.1 for which the squared bias and the variance of the posterior mean will be incorrectly balanced, resulting in a sub-optimal rate of posterior contraction. However, the larger posterior spread ensures that we do have asymptotic coverage of credible sets. It should be noted however that these sets have a diameter that is sub-optimal, i.e. they are too conservative.

Theorem 6 (generalized product of Gaussian experts) *For every $\beta, M > 0$ there exists a $\theta_0 \in H^\beta(M)$ such that for small enough $c > 0$,*

$$\mathrm{E}_{\theta_0}\mathbf{\Pi}_{VI}(\theta : \|\theta-\theta_0\|_2 \leq cm^{\frac{\beta}{1+2\beta}}n^{-\frac{\beta}{1+2\beta}} \,|\, \mathbf{Y}) \to 0$$

as $m \to \infty$ and $n/m \to \infty$. However, for all $\gamma \in (0,1)$ it holds that

$$\sup_{\theta_0 \in H^\beta(M)} \mathrm{P}_{\theta_0}\Big(\theta_0 \notin \hat{C}(L)\Big) \leq \gamma$$

for large enough $L > 0$.

Proof The proof of the theorem can be found in Section A.5. ■

3.6. Summary of results for non-adaptive methods

We have seen that the various methods for aggregation of the local posteriors can give quite different results. The methods we considered produce different global "posterior" measures. Depending on the relation between the bias and variance of the global posterior mean and the spread of this global posterior, the posterior contraction rate and coverage probabilities of credible sets can have different behaviours. We summarize our findings in Table 1. This is certainly not meant to be an exhaustive list of methods, but rather an illustration of how the design of distributed procedures can affect their fundamental performance.

Simulations further illustrate the theoretical results. We have considered a true signal θ consisting of the Fourier coefficients of the function shown in the left panel of Figure 3. This is a signal which has regularity $\beta = 1$ in the sense of (3.3). For this signal we simulated data according to (2.2), with $\sigma = 1$, $n = 4800$ and $m = 40$, i.e. we considered a distributed setting with $m = 40$ machines. For the sake of comparison, the right panel of Figure 3 shows the

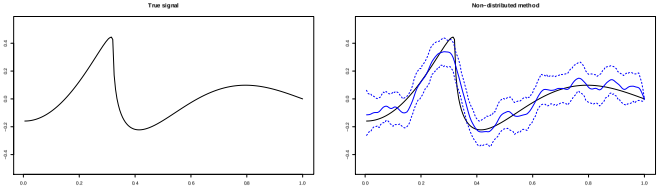


Figure 3: Left: true signal. Right: posterior mean (blue solid curve) and 95% pointwise credible bands (dashed blue curves) for the non-distributed method.

signal reconstruction and uncertainty quantification for the non-distributed method which first aggregates all data in a single machine and then computes the posterior corresponding to the prior $\Pi(\cdot\,|\,\alpha)$ defined by (3.1), with $\alpha = \beta$. This is a method which is known to have an optimal convergence rate and correct quantification of uncertainty. This classical, non-distributed result should be compared to Figure 4, which visualizes the "posteriors" generated by each of the distributed methods I–VI.

In accordance with our theoretical results, we see that the results of methods III and IV are comparable with the non-distributed method. Methods I, V and VI have worse signal reconstruction. The posterior mean of Method II is comparable to that of the optimal methods, but the uncertainty is underestimated.

An important observation to make is that the methods that achieve the same optimal performance as non-distributed methods, all use information about the regularity β of the unkown signal, mostly through the setting of tuning parameters in the priors. In that sense, they are non-adaptive. They serve as useful results that indicate what is possible in principle if we have certain oracle knowledge about the truth we are trying to learn. To understand what realistic procedures can achieve this has to be combined with insight into what can be learned about this oracle knowledge from the data. In the next section we address this issue in the context of our distributed signal-in-white-noise model.

4. Results for adaptive procedures

In the non-distributed case it is well known that there exist adaptive methods that achieve the same optimal performance as non-adaptive procedures, without using knowledge of the regularity β of the unkown signal. These methods somehow succeed in correctly trading off bias, variance (and spread in Bayesian methods) in a purely data-driven manner. For several such result in the context of the signal-in-white-noise model, see, for instance, Giné and Nickl (2016) and the references therein. For distributed methods the issue of adaptation appears to be a lot more subtle. In this paper we only have a first, negative result on adaptive properties of distributed methods.

So now we do not assume that we know the true regularity β of the unknown signal. As before we employ the prior $\Pi(\cdot\,|\,\alpha)$ in the local machines. To tune the regularity parameter α

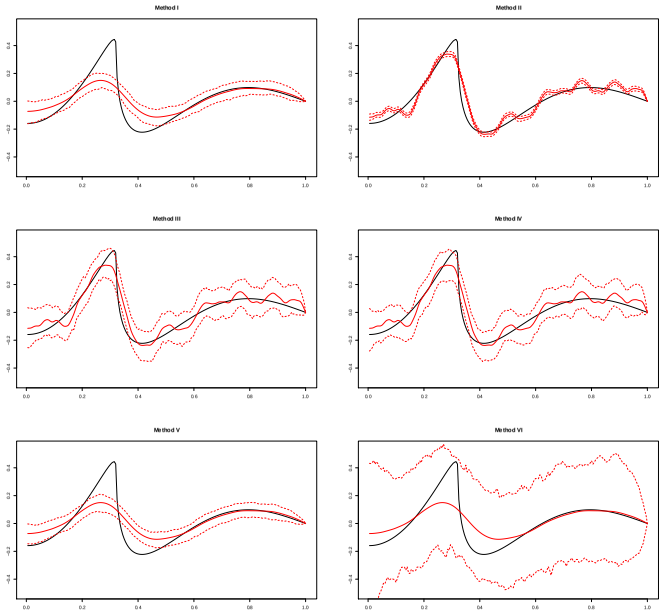


Figure 4: Global posterior mean (solid red curve) and 95% pointwise credible bands (dashed red curves) for each of the methods I–VI.

of the prior we consider a distributed version of maximum marginal likelihood estimator, as proposed by Deisenroth and Ng (2015). The usual, non-distributed version of that method would use the maximizer of the map

$$\alpha \mapsto \log \int \Big(\prod_{j=1}^{m} p(\mathbf{Y}^j \,|\, \theta) \Big) \, \Pi(d\theta \,|\, \alpha)$$

as tuning parameter. Maximizing this function however requires having all data available in a central machine. In the distributed setting, Deisenroth and Ng (2015) argue that this map is well approximated by the map

$$\alpha \mapsto \sum_{j=1}^{m} \log \Big(\int p(\mathbf{Y}^j \,|\, \theta) \, \Pi(d\theta \,|\, \alpha) \Big).$$

Now every term in the sum just depends on one of the local machines and this function can be maximized on the central machine by repeatedly asking the local machines for function evaluations and gradients of the local log-marginal likelihoods

$$\log \int p(\mathbf{Y}^j \,|\, \theta) \, \Pi(d\theta \,|\, \alpha).$$

The resulting estimator is denoted by $\hat{\alpha}$, i.e.

$$\hat{\alpha} = \operatorname*{argmax}_{\alpha \in [0, \log n]} \sum_{j=1}^{m} \log \Big(\int p(\mathbf{Y}^j \,|\, \theta) \, \Pi(d\theta \,|\, \alpha) \Big).$$

(We maximize over a compact interval to ensure that the maximizer exists.)

It turns out that in the distributed setting, the local machines are in general not able to learn enough about the true signal regularity β. The following lemma asserts that there exist "difficult" signals for which the estimator $\hat{\alpha}$ overestimates the regularity.

Lemma 7 *For $\beta, M > 0$, consider a signal $\theta_0 \in \ell^2$ such that*

$$\theta_{0,i}^2 = \begin{cases} M^2 i^{-1-2\beta} & \text{if } i \geq (n/(\sigma^2 \sqrt{m}))^{1/(1+2\beta)}, \\ 0 & \text{else.} \end{cases} \tag{4.1}$$

Then $\theta_0 \in H^\beta(M)$ and if M is small enough, then

$$\mathrm{P}_{\theta_0}(\hat{\alpha} \geq \beta + 1/2) \to 1 \tag{4.2}$$

if $n/m \to \infty$ and $m \to \infty$.

Proof The proof is given in Section B.1. ∎

In view of Lemma 7 it is perhaps not surprising that if the approximated maximum marginal likelihood estimator $\hat{\alpha}$ is used to tune the local prior that is used in every machine, sub-optimal performance is obtained for certain truths. Intuitively this is because due to the smaller signal-to-noise ratio, or "sample size" in the local machines, certain truths may appear more regular than they really are. It turns out that using the estimator $\hat{\alpha}$ in combination with any of the methods considered in the preceding section indeed leads to sub-optimal rates and bad coverage probabilities for certain truths. As an illustration we present a rigorous statement for the method of Section 3.4, but similar results can be derived for the others methods as well.

So suppose that in every local problem the prior $\Pi(\cdot \,|\, \alpha)$ is used, the corresponding generalized posterior $\tilde{\Pi}^j(\cdot \,|\, \mathbf{Y}^j)$ is computed locally (which involves raising the local likelihood to the power m), and then the tuning parameter α is substituted by the estimator $\hat{\alpha}$ defined above. In the central machine, the global "posterior" $\mathbf{\Pi}_{VII}(\cdot \,|\, \mathbf{Y})$ is constructed as the 2-Wasserstein barycenter of the local "posterior" measures $\tilde{\Pi}^1(\cdot \,|\, \hat{\alpha}, \mathbf{Y}^1), \ldots, \tilde{\Pi}^m(\cdot \,|\, \hat{\alpha}, \mathbf{Y}^m)$.

Theorem 8 *For $\beta, M > 0$ and θ_0 as in Lemma 7 we have, for some $c > 0$,*

$$\mathrm{E}_{\theta_0} \mathbf{\Pi}_{VII}(\theta : \, \|\theta - \theta_0\|_2 \leq c(n/\sqrt{m})^{-\frac{\beta}{1+2\beta}} | \mathbf{Y}) \to 0$$

as $m \to \infty$ and $n/m \to \infty$. Furthermore, for all $L > 0$ it holds that

$$\mathrm{P}_{\theta_0}\big(\theta_0 \in \hat{C}(L)\big) \to 0.$$

Proof See Section B.2. ∎

A simulation illustrating the theoretical result of the theorem is given in Figure 2. The left panel visualizes the "posterior" generated by method VII, in the same distributed setting, and using the same simulated data as considered in Section 3.6.

So when combined with a data-driven tuning method like the distributed version of maximum marginal likelihood considered here, even the distributed methods that perform well in the non-adaptive setting loose their favourable properties. None of the methods yields a procedure that automatically adapts to regularity and achieves the optimal non-distributed rate. This does not imply of course that such an adaptive method does not exist. We expect however that the matter is delicate and that fundamental limitations exist.

The issue appears to be similar to that of the existence of adaptive confidence sets. To achieve adaptation in our distributed setting the local machines must be able to learn the "global" regularity of the signal from the limited local data that they have available. Analogous to the adaptive confidence problem we expect that this is in general only possible under additional assumptions on the true signal, like the self-similarity or polished tail conditions proposed for instance in Giné and Nickl (2010), Bull (2012), Szabó et al. (2015), Nickl and Szabó (2016), Belitser et al. (2017). Making these admittedly somewhat loose claims mathematically precise takes considerably more effort. Recent work shows that in a distributed setting with communication restrictions, some degree of adaptation to smoothness is in principle possible, but requires different kinds of algorithms (see Szabo and van Zanten (2019)). Many open questions remain at the moment however. It is for instance unclear how the possibility of adaptation, or purely data-driven tuning, is related the degree of communication or the amount of central computation allowed. It would in particular be interesting to better understand the theoretical performance of distributed methods which allow multiple rounds of communication (e.g. Shamir et al. (2014), Heinze et al. (2016), Wang et al. (2017a,b), Lu et al. (2016)).

Appendix A. Proofs for Section 3

A.1. Proof of Theorem 1

By completing the square we see that under the local posterior $\Pi^j(\cdot\,|\,\mathbf{Y}^j)$ the coefficients θ_i are independent and Gaussian, with mean $\hat\theta_i^j$ and variance s_i^2 given by

$$\hat\theta_i^j = \frac{n}{n+\sigma^2 m i^{1+2\beta}} Y_i^j, \qquad s_i^2 = \frac{\sigma^2 m}{n+\sigma^2 m i^{1+2\beta}}.$$

Hence the global "posterior" $\mathbf{\Pi}_I(\cdot\,|\,\mathbf{Y})$ is Gaussian as well, and under that measure the coefficients θ_i are independent and have mean $\hat\theta_i$ and variance t_i^2 given by

$$\hat\theta_i = \frac{1}{m}\sum_{j=1}^m \hat\theta_i^j, \qquad t_i^2 = \frac{s_i^2}{m}.$$

For the global posterior mean we have, for every $\theta_0 \in \ell^2$,

$$\mathrm{E}_{\theta_0}\hat\theta_i - \theta_{0,i} = \frac{-\sigma^2 m i^{1+2\beta}}{n+\sigma^2 m i^{1+2\beta}}\theta_{0,i}, \qquad \mathrm{Var}_{\theta_0}\hat\theta_i = \frac{\sigma^2 n}{(n+\sigma^2 m i^{1+2\beta})^2},$$

and hence,

$$\mathrm{E}_{\theta_0}\|\hat\theta-\theta_0\|_2^2 = \sum \frac{\sigma^4 m^2 i^{2+4\beta}}{(n+\sigma^2 m i^{1+2\beta})^2}\theta_{0,i}^2 + \sum \frac{\sigma^2 n}{(n+\sigma^2 m i^{1+2\beta})^2}.$$

By Lemma 9 the second, variance term is of the order

$$m^{-1/(1+2\beta))} n^{-2\beta/(1+2\beta)},$$

as $n/m \to \infty$. For $\theta_{0,i}^2 = M i^{-1-2\beta}$, by the same lemma, the first, squared bias term is proportional to $(n/m)^{-2\beta/(1+2\beta)}$. For the global spread, again in view of Lemma 9, we have

$$\sum t_i^2 = \sum \frac{\sigma^2}{n+\sigma^2 m i^{1+2\beta}} \asymp m^{-1/(1+2\beta))} n^{-2\beta/(1+2\beta)}. \tag{A.1}$$

By the triangle inequality we have

$$\begin{aligned}&\mathbf{\Pi}_I(\|\theta-\theta_0\|_2 \le c m^{\frac{\beta}{1+2\beta}} n^{-\frac{\beta}{1+2\beta}}\,|\mathbf{Y})\\ &\le \mathbf{\Pi}_I(\theta:\ \|\mathrm{E}_{\theta_0}\hat\theta-\theta_0\|_2 - c m^{\frac{\beta}{1+2\beta}} n^{-\frac{\beta}{1+2\beta}} - \|\hat\theta-\mathrm{E}_{\theta_0}\hat\theta\|_2 \le \|\theta-\hat\theta\|_2|\mathbf{Y}).\end{aligned}$$

It follows from the bounds on the variance and squared bias of the posterior mean that for θ_0 as chosen above, the quantity

$$\|\mathrm{E}_{\theta_0}\hat\theta-\theta_0\|_2 - c m^{\frac{\beta}{1+2\beta}} n^{-\frac{\beta}{1+2\beta}} - \|\hat\theta-\mathrm{E}_{\theta_0}\hat\theta\|_2$$

appearing in the posterior probability is with P_{θ_0}-probability tending to one bounded from below by $c m^{\frac{\beta}{1+2\beta}} n^{-\frac{\beta}{1+2\beta}}$ for $c>0$ small enough. Then by the upper bound for the posterior spread and Chebyshev's inequality we obtain the first statement of the theorem.

For the coverage we note that the radius r_γ of the credible set is a multiple of $m^{-1/(2+4\beta)} n^{-\beta/(1+2\beta)}$, which follows from the Gaussianity of the posterior and (A.1). Then by similar computations as above we get that for the same truth θ_0,

$$\begin{aligned}\mathrm{P}_{\theta_0}(\theta_0 \in \hat C(L)) &= \mathrm{P}_{\theta_0}(\|\hat\theta-\theta_0\|_2 \le L r_\gamma)\\ &\le \mathrm{P}_{\theta_0}\big(\|\hat\theta-\mathrm{E}_{\theta_0}\hat\theta\|_2 \ge \|\mathrm{E}_{\theta_0}\hat\theta-\theta_0\|_2 - L r_\gamma\big)\\ &\le \mathrm{P}_{\theta_0}\big(\|\hat\theta-\mathrm{E}_{\theta_0}\hat\theta\|_2 \ge c m^{\frac{\beta}{1+2\beta}} n^{-\frac{\beta}{1+2\beta}}\big)\\ &\lesssim m^{\frac{-2\beta}{1+2\beta}} n^{\frac{2\beta}{1+2\beta}} \mathrm{E}_{\theta_0}\|\hat\theta-\mathrm{E}_{\theta_0}\hat\theta\|_2^2 \lesssim m^{-1} \to 0.\end{aligned}$$

This completes the proof of the theorem.

A.2. Proof of Theorem 2

Raising the local likelihood (3.4) to the power m makes it proportional to

$$\prod e^{-\frac{1}{2}\frac{n(Y_i^j-\theta_i)^2}{\sigma^2}},$$

which is the likelihood for the case $m=1$. It follows that under the generalized local posterior $\tilde\Pi^j(\cdot\,|\,\mathbf{Y}^j)$ the coefficients θ_i are independent and Gaussian, with mean $\hat\theta_i^j$ and variance s_i^2 given by

$$\hat\theta_i^j = \frac{n}{n+\sigma^2 i^{1+2\beta}} Y_i^j, \qquad s_i^2 = \frac{\sigma^2}{n+\sigma^2 i^{1+2\beta}}.$$

Hence the global "posterior" $\mathbf{\Pi}_{II}(\cdot\,|\,\mathbf{Y})$ is again Gaussian, and under this global measure the coefficients θ_i are independent and have mean $\hat\theta_i$ and variance t_i^2 given by

$$\hat\theta_i = \frac{1}{m}\sum_{j=1}^m \hat\theta_i^j, \qquad t_i^2 = \frac{s_i^2}{m}.$$

For the global posterior mean we have in this case, for every $\theta_0 \in \ell^2$,

$$\mathrm{E}_{\theta_0}\hat\theta_i - \theta_{0,i} = \frac{-\sigma^2 i^{1+2\beta}}{n+\sigma^2 i^{1+2\beta}}\theta_{0,i}, \qquad \mathrm{Var}_{\theta_0}\hat\theta_i = \frac{\sigma^2 n}{(n+\sigma^2 i^{1+2\beta})^2},$$

and hence,

$$\mathrm{E}_{\theta_0}\|\hat\theta-\theta_0\|_2^2 = \sum \frac{\sigma^4 i^{2+4\beta}}{(n+\sigma^2 i^{1+2\beta})^2}\theta_{0,i}^2 + \sum \frac{\sigma^2 n}{(n+\sigma^2 i^{1+2\beta})^2}.$$

For all $\theta_0 \in H^\beta(M)$, in view of Lemma 9, the squared bias term is bounded by

$$M^2 \sum \frac{\sigma^4 i^{1+2\beta}}{(n+\sigma^2 i^{1+2\beta})^2} \lesssim M^2 n^{-2\beta/(1+2\beta)}$$

for large n, and the variance term behaves like a constant times $n^{-2\beta/(1+2\beta)}$ as well. The global spread $\sum t_i^2$ is of the order $m^{-1} n^{-2\beta/(1+2\beta)}$ for large n, again following from Lemma 9.

For $M_n \to \infty$ and $\theta_0 \in H^\beta(M)$ we now have, by the triangle inequality,

$$\begin{aligned}&\mathbf{\Pi}_{II}(\theta:\ \|\theta-\theta_0\|_2 \geq M_n n^{-\beta/(1+2\beta)} \,|\, \mathbf{Y})\\ &\leq \mathbf{\Pi}_{II}(\theta:\ \|\theta-\hat\theta\|_2 \geq M_n n^{-\beta/(1+2\beta)} - \|\hat\theta - \mathrm{E}_{\theta_0}\hat\theta\|_2 - \|\theta_0 - \mathrm{E}_{\theta_0}\hat\theta\|_2 \,|\, \mathbf{Y}).\end{aligned}$$

By the bounds on the bias and the variance of the posterior mean derived above the quantity on the right of the inequality in the last posterior probability is bounded from below by $(M_n/2)n^{-\beta/(1+2\beta)}$ with P_{θ_0}-probability tending to one as $n, m \to \infty$, uniformly in $\theta_0 \in H^\beta(M)$. By Chebychev's inequality, and the bound on the posterior spread, we conclude that the first statement of the theorem holds.

For the second statement we first note that by Chebychev's inequality and by the upper bound on the posterior spread the radius r_γ of the credible set is for large n bounded by $Cm^{-1/2}n^{-\beta/(1+2\beta)}$ for some $C > 0$. Hence, since the posterior mean is Gaussian and the Gaussian measure of a ball of a fixed size is maximal if the ball is centered at the mean (a consequence of Anderson's inequality, e.g. Lifshits (1995), Section 11), we have

$$\begin{aligned}\mathrm{P}_{\theta_0}\big(\theta_0 \in \hat C(L)\big) &\leq \mathrm{P}_{\theta_0}\big(\|\hat\theta - \theta_0\|_2 \leq CLm^{-1/2}n^{-\beta/(1+2\beta)}\big)\\ &\leq \mathrm{P}_{\theta_0}\big(\|\hat\theta - \mathrm{E}_{\theta_0}\hat\theta\|_2 \leq CLm^{-1/2}n^{-\beta/(1+2\beta)}\big).\end{aligned}$$

By Chebychev's inequality,

$$\mathrm{P}_{\theta_0}\big(\|\hat\theta - \mathrm{E}_{\theta_0}\hat\theta\|_2^2 \leq \sum \sigma_i^2 - a\sqrt{2\sum \sigma_i^4}\big) \leq \frac{1}{a^2}$$

for all $a > 0$, where $\sigma_i^2 = \mathrm{Var}_{\theta_0}\hat\theta_i$. Above we saw that $\sum \sigma_i^2 \asymp n^{-2\beta/(1+2\beta)}$. Similarly, it is easily seen that $\sum \sigma_i^4 \asymp n^{(-1-4\beta)/(1+2\beta)}$. Hence by taking $a = n^{(1/4)/(1+2\beta)}$, for instance, we see that for $c > 0$ small enough,

$$\mathrm{P}_{\theta_0}\big(\|\hat\theta - \mathrm{E}_{\theta_0}\hat\theta\|_2 \leq cn^{-\beta/(1+2\beta)}\big) \to 0$$

as $n \to \infty$. But then also

$$\mathrm{P}_{\theta_0}\big(\|\hat\theta - \mathrm{E}_{\theta_0}\hat\theta\|_2 \leq CLm^{-1/2}n^{-\beta/(1+2\beta)}\big) \to 0$$

as $m, n \to \infty$.

A.3. Proof of Theorem 3

In this case the jth local posterior is a product of Gaussians with means and variances given by

$$\hat\theta_i^j = \frac{n}{n + \sigma^2 m\tau^{-1}i^{1+2\alpha}}Y_i^j, \qquad s_i^2 = \frac{\sigma^2 m}{n + \sigma^2 m\tau^{-1}i^{1+2\alpha}}.$$

As before the global "posterior" is Gaussian as well, and under that measure the coefficients θ_i are independent and have mean $\hat\theta_i$ and variance t_i^2 given by

$$\hat\theta_i = \frac{1}{m}\sum_{j=1}^m \hat\theta_i^j, \qquad t_i^2 = \frac{s_i^2}{m}.$$

For the global posterior mean we have, for every $\theta_0 \in \ell^2$,

$$\mathrm{E}_{\theta_0}\hat\theta_i - \theta_{0,i} = \frac{-\sigma^2 m\tau^{-1}i^{1+2\alpha}}{n + \sigma^2 m\tau^{-1}i^{1+2\alpha}}\theta_{0,i}, \qquad \mathrm{Var}_{\theta_0}\hat\theta_i = \frac{\sigma^2 n}{(n + \sigma^2 m\tau^{-1}i^{1+2\alpha})^2},$$

and hence

$$\mathrm{E}_{\theta_0}\|\hat\theta - \theta_0\|_2^2 = \sum \frac{\sigma^4 m^2\tau^{-2}i^{2+4\alpha}}{(n + \sigma^2 m\tau^{-1}i^{1+2\alpha})^2}\theta_{0,i}^2 + \sum \frac{\sigma^2 n}{(n + \sigma^2 m\tau^{-1}i^{1+2\alpha})^2}.$$

Then in view of Lemma 9, we see that for $\beta < 1 + 2\alpha$ and uniformly for $\theta_0 \in H^\beta(M)$, the squared bias term is bounded by a constant times

$$M^2(\tau n/m)^{-2\beta/(1+2\alpha)}.$$

Similarly, the variance term and the posterior spread $\sum t_i^2$ both behave like a constant times

$$(\tau/m)^{1/(1+2\alpha)}n^{-2\alpha/(1+2\alpha)}$$

as $n \to \infty$. The choice $\tau = mn^{2(\alpha-\beta)/(1+2\beta)}$ balances these quantities, so that all three are of the order $n^{-2\beta/(1+2\beta)}$.

By exactly the same reasoning as in Section A.2, the fact that the squared bias bound and the variance and spread are of the same order implies the first statement of the theorem. For the coverage statement we first note that the squared credible set radius r_γ^2 is the $1-\gamma$ quantile of the distribution of $\sum t_i^2 Z_i^2$, with t_i^2 as above and Z_i independent standard normals. This distribution has mean $\sum t_i^2 \asymp n^{-2\beta/(1+2\beta)}$ and variance

$$2\sum t_i^4 \asymp \frac{1}{n}n^{-2\beta/(1+2\beta)},$$

following from Lemma 9. As the standard deviation is of smaller order than the mean, it follows from Chebychev's inequality that $r_\gamma \geq cn^{-\beta/(1+2\beta)}$ for some $c > 0$. For the coverage probability we then have

$$\mathrm{P}_{\theta_0}\Big(\theta_0 \notin \hat C(L)\Big) \leq \mathrm{P}_{\theta_0}\Big(\|\hat\theta - \theta_0\|_2 \geq cLn^{-\beta/(1+2\beta)}\Big) \leq \frac{n^{2\beta/(1+2\beta)}}{c^2L^2}\mathrm{E}_{\theta_0}\|\hat\theta - \theta_0\|_2^2.$$

By the bounds on the bias and variance of the posterior mean the right-hand side is smaller than γ for L large enough, uniformly for $\theta_0 \in H^\beta(L)$.

A.4. Proof of Theorem 4

As we saw in Section A.2, the jth local generalized posterior is a product of Gaussians with means $\hat\theta_i^j$ and variances s_i^2 given by

$$\hat\theta_i^j = \frac{n}{n + \sigma^2 i^{1+2\beta}}Y_i^j, \qquad s_i^2 = \frac{\sigma^2}{n + \sigma^2 i^{1+2\beta}}.$$

In other words, the jth local measure is a Gaussian measure on ℓ^2 with mean $\hat\theta^j = (\hat\theta_i^j)_i$ and (diagonal) covariance operator $R : \ell^2 \to \ell^2$ given by $(Rx)_i = s_i^2 x_i$, which is the same for

every local machine. The Wasserstein barycenter of a finite collection of Gaussian measures is a Gaussian measure again (e.g. Agueh and Carlier (2011)). By Theorem 3.5 of Gelbrich (1990) the squared 2-Wasserstein distance between the jth local measure and a Gaussian measure on ℓ^2 with mean μ and covariance operator K is given by

$$\|\hat\theta^j-\mu\|_2^2+\mathrm{tr}(R)+\mathrm{tr}(K)-2\mathrm{tr}\sqrt{R^{1/2}KR^{1/2}}.$$

It follows that the barycenter $\mathbf{\Pi}_{IV}(\cdot\,|\,\mathbf{Y})$ of the local generalized posteriors is the Gaussian measure on ℓ^2 with mean $\hat\theta$ equal to the average of the local means $\hat\theta^j$ and covariance operator equal to R. In other words, the global "posterior" is a product of Gaussians with means and variances given by

$$\hat\theta_i=\frac{1}{m}\sum_{j=1}^m\hat\theta_i^j,\qquad t_i^2=s_i^2.$$

So the global posterior mean is the same as in Section A.2 and the posterior spread $\sum t_i^2$ is a factor m larger. It then follows from the considerations in Section A.2 that the squared bias of the global posterior mean is bounded by a constant times $M^2n^{-2\beta/(1+2\beta)}$, uniformly for $\theta_0\in H^\beta(M)$. Moreover, the variance term $\sum s_i^2$ and the posterior spread $\sum t_i^2$ behave like a multiple of $n^{-2\beta/(1+2\beta)}$ as well. As was explained in Section A.3, this leads to the statement of the theorem.

A.5. Proof of Theorem 6

The proof of the first statement is the same as in Section A.1, since the mean of the global "posterior" is the same as for the naive averaging method.

For the second statement, we observe that for $\theta_0\in H^\beta(M)$, the squared bias term for the posterior mean satisfies

$$\sum\frac{\sigma^4m^2i^{2+4\beta}}{(n+\sigma^2mi^{1+2\beta})^2}\theta_{0,i}^2\le M^2\sum\frac{\sigma^4m^2i^{1+2\beta}}{(n+\sigma^2mi^{1+2\beta})^2}\lesssim M^2(n/m)^{-2\beta/(1+2\beta)}.$$

for $n/m\to\infty$. As was shown in Section A.1 the variance of the posterior mean behaves as $m^{-1/(1+2\beta)}n^{-2\beta/(1+2\beta)}$. Since the spread $\sum t_i^2$ of the posterior is a factor m larger than in Section A.1, it is of the same order $(n/m)^{-2\beta/(1+2\beta)}$ as the squared bias term. Since squared bias and spread are of the same order, the variance is of smaller order, and

$$\sqrt{\sum t_i^4}\asymp\Big(\frac{n}{m}\Big)^{\frac{-1/2}{1+2\beta}}\sum t_i^2$$

is of lower order than $\sum t_i^2$, the coverage statement can be proved as in Section A.3.

A.6. Technical Lemma

Lemma 9 *For $s,t>0$ with $st>1$ and $r<st-1$ consider the function $f(x)=x^r(x^s+1)^{-t}$ and set $f_\nu=\sum_{k=1}^\infty\nu^{-1}f(k/\nu)$. Then as $\nu\to\infty$,*

(i) If $r>-1$, then $f_\nu\asymp\int_0^\infty f(x)dx$.

(ii) If $r=-1$, then $f_\nu\asymp\log\nu$.

(iii) If $r<-1$, then $f_\nu\asymp\nu^{r-1}\sum_{k=1}^\infty k^{-r}$.

Proof Assertions (ii) and (iii), along with (i) for $-1<r\le0$ are proved in Lemma A.1 of Szabó et al. (2013), hence it remains to verify assertion (i) for $0<r<st-1$. Note that

$$\sum_{k=1}^{N\nu}\nu^{-1}f(k/\nu)<f_\nu<\sum_{k=1}^{N\nu}\nu^{-1}f(k/\nu)+\sum_{k=N\nu+1}^{\infty}\nu^{-1}f(k/\nu).\tag{A.2}$$

Since the function $f(x)$ is continuous on $[0,N]$ it is Riemann integrable (see for instance Theorem 6.8 of Rudin (1976)), hence the right Riemann sum converges to the integral, i.e. $\sum_{k=1}^{N\nu}\nu^{-1}f(k/\nu)\to\int_0^Nf(x)dx$ as $\nu\to\infty$ (for simplicity assume that $\nu\in\mathbb{N}$). Furthermore, for every $\nu>0$,

$$\sum_{k=N\nu+1}^{\infty}\nu^{-1}f(k/\nu)\le\nu^{st-r-1}\sum_{k=N\nu+1}^{\infty}k^{r-st}\le\nu^{st-r-1}\int_{N\nu}^\infty x^{r-st}dx\le\frac{1}{st-r-1}N^{r+1-st}$$

and $\int_{N+1}^\infty f(x)dx\le\int_{N+1}^\infty x^{r-st}dx\le N^{-st+r+1}/(st-r-1)$. Therefore by choosing N sufficiently large, both sides of (A.2) gets arbitrarily close to $\int_0^\infty f(x)dx$ as $\nu\to\infty$, concluding the proof of the statement. ∎

Appendix B. Proofs for Section 4

B.1. Proof of Lemma 7

The estimator $\hat\alpha$ is the maximizer of the random map $\alpha\mapsto\sum_j\ell_j(\alpha)$, where

$$\ell_j(\alpha)=\log\int p(\mathbf{Y}^j\,|\,\theta)\,\Pi(d\theta\,|\,\alpha).$$

The asymptotic behaviour of the local log-marginal likelihood ℓ_j has been studied in Knapik et al. (2016). Denote the derivative of ℓ_j with respect to α by $\dot\ell_j$ and let $k=n/(\sigma^2m)$ be the local "sample size". Moreover, for $l>0$, define

$$\underline\alpha=\inf\{\alpha>0:h_k(\alpha)>l\}\wedge\sqrt{\log k},$$

where

$$h_k(\alpha)=\frac{1+2\alpha}{k^{1/(1+2\alpha)}\log k}\sum_i\frac{k^2i^{1+2\alpha}\theta_{0,i}^2\log i}{(k+i^{1+2\alpha})^2}.$$

Note that the expectation $\mathrm{E}_{\theta_0}\dot\ell_j(\alpha)$ does not depend on j. It is proved in Section 5.3 of Knapik et al. (2016) that if l is smaller than some universal threshold, then for every j

$$\liminf_{k\to\infty}\inf_{\alpha\le\underline\alpha}\frac{1+2\alpha}{k^{1/(1+2\alpha)}\log k}\mathrm{E}_{\theta_0}\dot\ell_j(\alpha)=\delta>0,$$

$$\mathrm{E}_{\theta_0} \sup_{\alpha \le \underline{\alpha}} \frac{1+2\alpha}{k^{1/(1+2\alpha)} \log k} |\dot{\ell}_j(\alpha) - \mathrm{E}_{\theta_0} \dot{\ell}_j(\alpha)| \lesssim e^{-C\sqrt{\log k}}$$

for constants $\delta, C > 0$. But then we also have

$$\liminf_{k\to\infty} \inf_{\alpha \le \underline{\alpha}} \frac{1+2\alpha}{k^{1/(1+2\alpha)} \log k} \mathrm{E}_{\theta_0} \sum_j \dot{\ell}_j(\alpha) > m\delta$$

and

$$\mathrm{E}_{\theta_0} \sup_{\alpha \le \underline{\alpha}} \frac{1+2\alpha}{k^{1/(1+2\alpha)} \log k} \Big| \sum_j \dot{\ell}_j(\alpha) - \mathrm{E}_{\theta_0} \sum_j \dot{\ell}_j(\alpha) \Big| \lesssim m e^{-C\sqrt{\log k}}.$$

By Markov's inequality, it follows that with probability at least $1 - C_1 \exp(-C_2\sqrt{\log k})$ the map $\alpha \mapsto \sum_j \ell_j(\alpha)$ is strictly increasing on the interval $[0, \underline{\alpha}]$. Hence, on that event we have $\hat{\alpha} \ge \underline{\alpha}$.

It remains to show that $\underline{\alpha} \ge \beta + 1/2$. To that end it suffices to prove that $h_k(\alpha) \le l$ for all $\alpha \le \beta + 1/2$. To see this, suppose first that $\alpha < \beta$. Define $N_\beta = (n/(\sigma^2\sqrt{m}))^{1/(1+2\beta)}$ and $M_\alpha = k^{1/(1+2\alpha)}$. By definition of θ_0 we then have

$$\begin{aligned}
h_k(\alpha) &= \frac{M^2}{M_\alpha \log M_\alpha} \sum_{i=N_\beta}^{\infty} \frac{k^2 i^{2\alpha-2\beta} \log i}{(i^{1+2\alpha}+k)^2} \\
&\le \frac{M^2}{M_\alpha \log M_\alpha} \sum_{i=N_\beta}^{M_\alpha} i^{2\alpha-2\beta} \log i + \frac{M^2 k^2}{M_\alpha \log M_\alpha} \sum_{i=M_\alpha}^{\infty} i^{-2-2\alpha-2\beta} \log i \\
&\le M^2 \frac{M_\alpha N_\beta^{2\alpha-2\beta} \log M_\alpha}{M_\alpha \log M_\alpha} + M^2 \frac{k^2 M_\alpha^{-1-2\alpha-2\beta} \log M_\alpha}{M_\alpha \log M_\alpha} \\
&\lesssim M^2
\end{aligned}$$

for n, m large enough. Hence, if M is small enough, then $h_k \le l$ for $\alpha < \beta$. For $\beta \le \alpha \le \beta + 1/2$ we have

$$\begin{aligned}
h_k(\alpha) &\le \frac{M^2 k^2}{M_\alpha \log M_\alpha} \sum_{i=N_\beta}^{\infty} i^{-2-2\alpha-2\beta} \log i \\
&\le \frac{M^2 k^2 N_\beta^{-1-2\alpha-2\beta} \log N_\beta}{M_\alpha \log M_\alpha} \\
&= M^2 (n/\sigma^2)^{\frac{2\alpha}{1+2\alpha} - \frac{2\alpha}{1+2\beta}} m^{-2+\frac{1}{1+2\alpha}+\frac{1+2\alpha+2\beta}{2(1+2\beta)}} \frac{\log N_\beta}{\log M_\alpha} \lesssim m^{-\frac{2\alpha}{1+2\alpha}} \log m
\end{aligned}$$

for n/m large enough. Together, this shows that if both n/m and m are large enough, then indeed $h_k(\alpha) \le l$ for all $\alpha \le \beta + 1/2$.

B.2. Proof of Theorem 8

In view of the proof of Theorem 4 the jth local generalized posterior is a product of Gaussians with means $\hat{\theta}_i^j$ and variances s_i^2 given by

$$\hat{\theta}_i^j = \frac{n}{n + \sigma^2 i^{1+2\hat{\alpha}}} Y_i^j, \qquad s_i^2 = \frac{\sigma^2}{n + \sigma^2 i^{1+2\hat{\alpha}}}.$$

Using again that the Wasserstein barycenter of a finite collection of Gaussian measures is a Gaussian measure in combination with the explicit expression for the 2-Wasserstein distance between Gaussians (see Section A.4) we see that the global "posterior" is a product of Gaussians with means $\hat{\theta}_i$ and variances t_i^2 given by

$$\hat{\theta}_i = \frac{1}{m} \sum_{j=1}^m \hat{\theta}_i^j, \qquad t_i^2 = s_i^2.$$

The posterior mean can be written as $\hat{\theta} = \hat{\theta}(\hat{\alpha})$, where $\hat{\theta}(\alpha)$ is the estimator with a fixed choice α for the hyperparameter, i.e.

$$\hat{\theta}_i(\alpha) = \frac{1}{m} \sum_{j=1}^m \frac{n}{n + \sigma^2 i^{1+2\alpha}} Y_i^j.$$

For fixed α we also define the corresponding expectation $E(\alpha) = \mathrm{E}_{\theta_0} \hat{\theta}(\alpha)$. Then by the triangle inequality,

$$\|\hat{\theta} - \theta_0\|_2 \ge \|E(\hat{\alpha}) - \theta_0\|_2 - \|E(\hat{\alpha}) - \hat{\theta}(\hat{\alpha})\|_2.$$

We have the explicit expressions

$$\|E(\alpha) - \theta_0\|_2^2 = \sum_i \frac{\sigma^4 i^{2+4\alpha} \theta_{0,i}^2}{(n + \sigma^2 i^{1+2\alpha})^2}$$

and

$$\|E(\alpha) - \hat{\theta}(\alpha)\|_2^2 = \sum_i \frac{\sigma^2 n}{(n + \sigma^2 i^{1+2\alpha})^2} \Big(\frac{1}{\sqrt{m}} \sum_{j=1}^m Z_i^j \Big)^2.$$

Since the first expression is increasing in α and the second one is decreasing, we see that on the event $A = \{\hat{\alpha} \ge \beta + 1/2\}$ it holds that

$$\|\hat{\theta} - \theta_0\|_2 \ge \sqrt{\sum_i \frac{\sigma^4 \theta_{0,i}^2 i^{4+4\beta}}{(n + \sigma^2 i^{2+2\beta})^2}} - \sqrt{\sum_i \frac{\sigma^2 n}{(n + \sigma^2 i^{2+2\beta})^2} \Big(\frac{1}{\sqrt{m}} \sum_{j=1}^m Z_i^j \Big)^2}.$$

By definition of θ_0, the square of the first term on the right is bounded from below by

$$M^2 \sum_{i \ge \left((n/(\sigma^2\sqrt{m}))\right)^{1/(1+2\beta)}} \frac{\sigma^4 i^{3+2\beta}}{(n + \sigma^2 i^{2+2\beta})^2}.$$

In view of Lemma 9 we see that it is of the order $M^2 (n/\sqrt{m})^{-2\beta/(1+2\beta)}$. The square of the second term can be written as

$$\sum_i \frac{\sigma^2 n}{(n + \sigma^2 i^{2+2\beta})^2} U_i^2,$$

with the U_i independent and standard normal under P_{θ_0}. Again in view of Lemma 9 it is easily seen that the mean and variance of this sum behave as $n^{-(1+2\beta)/(2+2\beta)}$ and $n^{-(3+4\beta)/(2+2\beta)}$, respectively. Hence the standard deviation is of smaller order than the mean for large n, so that by Chebychev's inequality the square of the second term is of stochastic order $n^{-(1+2\beta)/(2+2\beta)}$. Since this is of smaller order than $(n/\sqrt{m})^{-2\beta/(1+2\beta)}$, we conclude that for the global "posterior" mean we have, for some constant $c > 0$,

$$\mathrm{P}_{\theta_0}(\|\hat{\theta} - \theta_0\|_2 \geq c(n/\sqrt{m})^{-\beta/(1+2\beta)}) \to 1$$

as $n/m \to \infty$ and $m \to \infty$. The spread $\sum t_i^2$ of the global posterior is on the event A bounded by

$$\sum_i \frac{1}{n + i^{2+2\beta}},$$

which is of the order $n^{-(1+2\beta)/(2+2\beta)} \ll (n/\sqrt{m})^{-2\beta/(1+2\beta)}$ as well, see Lemma 9. The conclusions of the theorem now follow.

Acknowledgments

The research leading to these results has received funding from the Netherlands Science foundation NWO and from the European Research Council under ERC Grant Agreement 320637.

References

Martial Agueh and Guillaume Carlier. Barycenters in the Wasserstein space. *SIAM Journal on Mathematical Analysis*, 43(2):904–924, 2011.

Eduard Belitser et al. On coverage and local radial rates of credible sets. *The Annals of Statistics*, 45(3):1124–1151, 2017.

Lawrence D. Brown and Mark G. Low. Asymptotic equivalence of nonparametric regression and white noise. *Ann. Statist.*, 24(6):2384–2398, 1996.

Adam D. Bull. Honest adaptive confidence bands and self-similar functions. *Electron. J. Statist.*, 6:1490–1516, 2012.

Y. Cao and D. J. Fleet. Generalized Product of Experts for Automatic and Principled Fusion of Gaussian Process Predictions. *ArXiv e-prints*, October 2014.

Marco Cuturi and Arnaud Doucet. Fast computation of Wasserstein barycenters. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 685–693, Bejing, China, 22–24 Jun 2014. PMLR.

Marc Deisenroth and Jun Wei Ng. Distributed Gaussian processes. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1481–1490, Lille, France, 07–09 Jul 2015. PMLR.

Matthias Gelbrich. On a formula for the L^2 Wasserstein metric between measures on Euclidean and Hilbert spaces. *Mathematische Nachrichten*, 147(1):185–203, 1990.

Evarist Giné and Richard Nickl. Confidence bands in density estimation. *Ann. Statist.*, 38 (2):1122–1170, 04 2010. doi: 10.1214/09-AOS738.

Evarist Giné and Richard Nickl. *Mathematical foundations of infinite-dimensional statistical models*. Cambridge University Press. 2016.

Christina Heinze, Brian McWilliams, and Nicolai Meinshausen. Dual-loco: Distributing statistical estimation using random projections. In Arthur Gretton and Christian C. Robert, editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 875–883, Cadiz, Spain, 09–11 May 2016. PMLR.

I. M. Johnstone. *Gaussian estimation: Sequence and wavelet models*. Book draft, 2017.

M. I. Jordan, J. D. Lee, and Y. Yang. Communication-Efficient Distributed Statistical Inference. *ArXiv e-prints*, May 2016.

B. T. Knapik, A. W. van der Vaart, and J. H. van Zanten. Bayesian inverse problems with Gaussian priors. *Ann. Statist.*, 39(5):2626–2657, 10 2011.

B. T. Knapik, B. T. Szabó, A. W. Vaart, and J. H. Zanten. Bayes procedures for adaptive inference in inverse problems for the white noise model. *Probability Theory and Related Fields*, 164(3):771–813, 2016.

Lucien Le Cam. *Asymptotic methods in statistical decision theory*. Springer, 2012.

J. D. Lee, Y. Sun, Q. Liu, and J. E. Taylor. Communication-efficient sparse regression: a one-shot approach. *ArXiv e-prints*, March 2015.

Mikhail Anatolevich Lifshits. *Gaussian random functions*. Springer, 1995.

J. Lu, G. Cheng, and H. Liu. Nonparametric Heterogeneity Testing For Massive Data. *ArXiv e-prints*, January 2016.

S. Minsker, S. Srivastava, L. Lin, and D. B. Dunson. Robust and scalable Bayes via a median of subset posterior measures. *ArXiv e-prints*, March 2014.

J. W. Ng and M. P. Deisenroth. Hierarchical Mixture-of-Experts Model for Large-Scale Gaussian Process Regression. *ArXiv e-prints*, December 2014.

Richard Nickl and Botond Szabó. A sharp adaptive confidence ball for self-similar functions. *Stochastic Processes and their Applications*, 126(12):3913–3934, 2016.

Michael Nussbaum. Asymptotic equivalence of density estimation and Gaussian white noise. *Ann. Statist.*, 24(6):2399–2430, 1996.

W. Rudin. *Principles of Mathematical Analysis*. International series in pure and applied mathematics. McGraw-Hill, 1976.

Steven L. Scott, Alexander W. Blocker, Fernando V. Bonassi, Hugh A. Chipman, Edward I. George, and Robert E. McCulloch. Bayes and big data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management*, 11:78–88, 2016.

Ohad Shamir, Nathan Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1000–II–1008. JMLR.org, 2014.

Z. Shang and G. Cheng. A Bayesian splitotic theory for nonparametric models. *ArXiv e-prints*, August 2015.

Sanvesh Srivastava, Volkan Cevher, Quoc Dinh, and David Dunson. WASP: Scalable Bayes via barycenters of subset posteriors. In Guy Lebanon and S. V. N. Vishwanathan, editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pages 912–920, San Diego, California, USA, 09–12 May 2015. PMLR.

B. Szabo and H. van Zanten. Adaptive distributed methods under communication constraints. *ArXiv e-prints*, 2019.

B. T. Szabó, A. W. van der Vaart, and J. H. van Zanten. Empirical Bayes scaling of Gaussian priors in the white noise model. *Electron. J. Statist.*, 7:991–1018, 2013.

Botond Szabó, A. W. van der Vaart, and J. H. van Zanten. Frequentist coverage of adaptive nonparametric Bayesian credible sets. *Ann. Statist.*, 43(4):1391–1428, 08 2015.

Alexandre B. Tsybakov. *Introduction to nonparametric estimation*. Springer, New York, 2009.

Aad van der Vaart and J. H. van Zanten. Bayesian inference with rescaled Gaussian process priors. *Electron. J. Statist.*, 1:433–448, 2007.

S. Volgushev, S.-K. Chao, and G. Cheng. Distributed inference for quantile regression processes. *ArXiv e-prints*, January 2017.

Jialei Wang, Mladen Kolar, Nathan Srebro, and Tong Zhang. Efficient distributed learning with sparsity. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3636–3645, International Convention Centre, Sydney, Australia, 06–11 Aug 2017a. PMLR.

Jialei Wang, Jason Lee, Mehrdad Mahdavi, Mladen Kolar, and Nati Srebro. Sketching Meets Random Projection in the Dual: A Provable Recovery Algorithm for Big and High-dimensional Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1150–1158, Fort Lauderdale, FL, USA, 20–22 Apr 2017b. PMLR.

Y. Zhu and J. Lafferty. Distributed Nonparametric Regression under Communication Constraints. *ArXiv e-prints*, March 2018.

Journal of Machine Learning Research 20 (2019) 1-26 Submitted 11/17; Revised 3/19; Published 5/19

Model Selection via the VC Dimension

Merlin Mpoudeu MERLIN.MPOUDEU@HUSKERS.UNL.EDU
Bank of America
Atlanta, GA, USA

Bertrand Clarke BCLARKE3@UNL.EDU
Department of Statistics
University of Nebraska-Lincoln
Lincoln, NE 68503, USA

Editor: John Shawe-Taylor

Abstract

We derive an objective function that can be optimized to give an estimator for the Vapnik-Chervonenkis dimension for use in model selection in regression problems. We verify our estimator is consistent. Then, we verify it performs well compared to seven other model selection techniques. We do this for a variety of types of data sets.

Keywords: Vapnik-Chervonenkis dimension, model selection, Bayesian information criterion, sparsity methods, empirical risk minimization, multi-type data.

1. Complexity and Model Selection

Model selection is often the first problem that must be addressed when analyzing data. In M-closed problems, see Bernardo and Smith (2000), the analyst posits a list of models and assumes one of them is true. In such cases, model selection is any procedure that uses data to identify one of the models on the model list. There is a vast literature on model selection in this context including information based methods such as the Aikaikie Information Criterion (AIC), the Bayes information criterion (BIC), residual based methods such as Mallows C_p or branch and bound, and code length methods such as the two-stage coding proposed by Barron and Cover (1991). We also have computational search methods such as simulated annealing and genetic algorithms. In addition, cross-validation (CV) is often used with non-parametric methods such as recursive partitioning, neural networks (aka deep learning) and kernel methods. A less well developed approach to model selection is via complexity as assessed by the Vapnik-Chervonenkis (VC) dimension, here denoted by d_{VC}. Its earliest usage seems to be in Vapnik and Chervonenkis (1968). A translation into English was published as Vapnik and Chervonenkis (1971).

Although, the VC dimension goes back to 1968, it wasn't until Vapnik et al. (1994) that a method for estimating d_{VC} was proposed in the classification context. Specifically, given a collection $\mathbb{C}$ of classifiers, Vapnik et al. (1994) tried to estimate the VC dimension of $\mathbb{C}$ by deriving an objective function based on the expected value of the maximum difference between two empirical evaluations of a single loss function, here denoted by Δ. The two empirical values come from dividing a given data set into a first and second part. The objective function proposed by Vapnik et al. (1994) depends on d_{VC}, the sample size n, and several constants that had to be determined. Using their objective function, they derived an estimator $\hat{d}_{VC}$ for d_{VC} given a class $\mathbb{C}$ of classifiers. This algorithm treated possible sample sizes as design points $n_1, n_2, \cdots, n_L$ and requires one level of bootstrapping. Despite the remarkable contribution of Vapnik et al. (1994), the objective function was over-complex and the algorithm did not give a tight enough bound on Δ. Later, Vapnik and his collaborators suggested a fix to tighten the bound on Δ. We do not use this here; it is unclear if this 'fix' will work in classification, let alone regression.

Choosing the design points is a nontrivial source of variability in the estimate of d_{VC}. So, Shao et al. (2000) proposed an algorithm, based on extensive simulations, to generate optimal values of $n_1, n_2, \cdots, n_L$, given L. They argued that non-uniform values of the n_l's gave better results than the uniform n_l's used in Vapnik et al. (1994).

More recently, in a pioneering paper that deserves more recognition that it has received, McDonald et al. (2011) established the consistency of the Vapnik (1998) estimator $\hat{d}_{VC}$ for d_{VC} in the classification context.

The main reason the estimator for d_{VC} of Vapnik et al. (1994) did not become more widely used, despite the result in McDonald et al. (2011), is, we suggest, that it was too unstable because the objective function did not bound Δ tightly enough in terms of d_{VC}. In addition, the form of the objective function in Vapnik et al. (1994) is more complicated and less well-motivated than our result Theorem 2. The reason is that the derivation in Vapnik et al. (1994) uses conditional probabilities, one of which goes to zero quite quickly (with n). So, it contributes negligibly to the upper bound. Our derivation ignores the conditioning and bounds a CV form of Δ that is typically larger than that used in Vapnik et al. (1994).

Our consistency proof is a simplification of the proof of the main result McDonald et al. (2011). Accordingly, we obtain a slower rate of consistency, but the probability of correct model selection still goes to one.

Our overall strategy is to derive an objective function for estimating d_{VC} in the regression setting that provides, we think, a tighter bound on a modified form of Δ. To convert from classification to regression, we discretize the loss used for regression into m intervals (the case $m = 1$ would then apply to classification). To get a tighter bound, we change the form of Δ from what Vapnik et al. (1994) used and we optimize over the leading factor in our upper bound. To use our estimator, we use an extra layer of bootstrapping so the quantity we empirically optimize represents the quantity we derive theoretically more accurately. The extra layer of bootstrapping stabilizes our estimator of d_{VC} and appears to reduce its dependency on the n_l's. If the models are nested in order of increasing VC dimension, it is straightforward to choose the model with VC dimension closest to our estimate $\hat{d}_{VC}$. Otherwise, we can convert a non-nested problem to the nested case by ordering the inclusion of the covariates using a shrinkage method such as the 'smoothly clipped absolute deviation' (SCAD, Fan and Li (2001)), or correlation (see Fan and Lv (2008)), and use our $\hat{d}_{VC}$ as before. Even when we force a model list to be nested, our model selection method performs well compared to a range of competitors including Vapnik et al. (1994)'s original method, two forms of penalized empirical risk minimization (denoted $\widehat{PERM}_1$ and $\widehat{PERM}_2$), AIC, BIC, CV (10-fold), SCAD, and adaptive LASSO (ALASSO, Zou (2006)). Our general findings indicate that in realistic settings, model selection via estimated VC dimension,

©2019 Merlin Mpoudeu and Bertrand Clarke.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-669.html.

when properly done, is fully competitive with existing methods and, unlike them, rarely gives abberant results.

This manuscript is structured as follows. In Sec. 2 we present the main theory justifying our estimator. In Subsec. 2.1 we discretize bounded loss functions so that upper bounds for the distinct regions involved in the definition of Δ can be derived and in Subsec. 2.2 we define our estimator of the VC dimension and give an algorithm for how to compute it. In Sec. 3 we use McDonald et al. (2011)'s consistency theorem to motivate our consistency theorem for $\hat{d}_{VC}$. In Sec. 4 we present our studies using simulated, benchmark, and real data. We compare our method for model selection to AIC, BIC, CV, $\widehat{PERM}_1$, and $\widehat{PERM}_2$. In this context, we suggest criteria to guide the selection of design points. Our comparisons also include simplifying non-nested model lists by using correlation, SCAD, and ALASSO. In Sec. 5 we discuss our overall findings.

2. Deriving an optimality criterion for estimating VC dimension

This section concerns Δ, the expected supremal difference between two evaluations of a bounded loss function, formally defined in (18) and (19). These bounds will enable us to derive an estimator for the VC dimension. In Sec. 2.1, we present our alternative version of the Vapnik et al. (1994) bounds and in Sec. 2.2, we present our estimator of d_{VC}.

2.1. Extension of the Vapnik et al. (1994) bounds to regression

Let $Z = (X, Y)$ be a random variable with outcomes $z = (x, y)$ assuming values on $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. The first entry, $X = x$, is regarded as an explanatory variable leading to $Y = y$. Let $P \in \mathcal{M}(\mathcal{Z})$ be the distribution of Z, where $\mathcal{M}(\mathcal{Z})$ is the collection of probability measures on $\mathcal{Z}$, and let $Z_{1:2n} = (Z_1, \ldots, Z_n, Z_{n+1}, \ldots, \ldots, Z_{2n})$ be a data set of size $2m$ of independently and identically distributed (IID) copies of Z. Write $D^1 = \{Z_1, \ldots, Z_n\}$ for the first half and $D^2 = \{Z_{n+1}, \ldots, Z_{2n}\}$ for the second half. Writing $Z_i = (X_i, Y_i)$ for $i = 1 \ldots 2n$, let

$$Q(Z_i, \alpha) = L(Y_i, f(X_i, \alpha)),$$

for a bounded real valued loss function L and $\alpha \in \Lambda$. We assume that Λ is a compact set in a finite dimensional real space, that the interior of Λ, $\mathrm{Int}(\Lambda)$, is non-void and convex, and that $\Lambda = \overline{\mathrm{Int}(\Lambda)}$. Also, we assume the continuous functions $f(\cdot \mid \alpha)$ are parametrized by α continuously and one-to-one. Thus, in our examples, Λ will be the parameter space for a class of regression functions $f(\cdot \mid \alpha)$. For ease of exposition we assume L, and hence Q, are also continuous.

For a fixed $\alpha \in \Lambda$, discretize $Q(z, \alpha)$ using m disjoint intervals (with union $[0, B)$):

$$Q^m(z, \alpha) = \sum_{j=0}^{m-1} \frac{(2j+1)B}{2m} \mathbb{I}\left[Q(Z, \alpha) \in I_j^m\right]. \tag{1}$$

The discretization is based on the uniform left-closed, right-open partition of $[0, B)$ into m subintervals, here denoted I_j^m and the numbers $((2j+1)B)/(2m)$ are the midpoints. In (1), $\mathbb{I}[\cdot]$ is an indicator function taking value 1 when its argument is true and value 0 when it is false. We use losses of the form (1) to define a cross-validation form for Δ.

Start by letting $\alpha_1, \alpha_2 \in \Lambda$, with $\alpha_1 \neq \alpha_2$, and let

$$\nu(D^2, \alpha_1) = \frac{1}{n} \sum_{i=1}^{n} Q(Z_{n+i}, \alpha_1) \quad \text{and} \quad \nu(D^1, \alpha_2) = \frac{1}{n} \sum_{i=1}^{n} Q(Z_i, \alpha_2). \tag{2}$$

These are the empirical risks of model α_1 on the second half of the data and of model α_2 on the first half of the sample, respectively. Observe that the empirical counts of the data points whose losses land in I_j^m are

$$N_j^m(D^2, \alpha_1) = \sum_{i=1}^{n} \mathbb{I}\left[Q(Z_{n+i}, \alpha_1) \in I_j^m\right] \quad \text{and} \quad N_j^m(D^1, \alpha_2) = \sum_{i=1}^{n} \mathbb{I}\left[Q(Z_i, \alpha_2) \in I_j^m\right]. \tag{3}$$

This means we are counting the errors of the α_1 model on the second half of the data and the erors of the α_2 model on the first half of the data. This begins the set up of the cross-validation form of the error that we use and leads to the following expressions for the empirical losses of the discretized loss functions:

$$\nu^m\left(D^2, \alpha_1\right) = \frac{1}{n} \sum_{j=0}^{m-1} N_j^m(D^2, \alpha_1) \frac{(2j+1)B}{2m}$$

$$\text{and} \quad \nu^m\left(D^1, \alpha_2\right) = \frac{1}{n} \sum_{j=0}^{m-1} N_j^m(D^1, \alpha_2) \frac{(2j+1)B}{2m}. \tag{4}$$

It is seen that the expressions in (4) are formed from the counts within each of the intervals. Let these be denoted by

$$\nu_j^m(D^2, \alpha_1) = \frac{1}{n} N_j^m(D^2, \alpha_1) \frac{(2j+1)B}{2m} \quad \text{and} \quad \nu_j^m(D^1, \alpha_2) = \frac{1}{n} N_j^m(D^1, \alpha_2) \frac{(2j+1)B}{2m}. \tag{5}$$

The first step in bounding Δ is to bound the probability of the 'bad set' where $\nu^m(D^2, \alpha_1)$ and $\nu^m(D^1, \alpha_2)$ are not close. Let $\epsilon > 0$ and, using the discretization into m intervals, define the set A_ϵ by the union:

$$A_\epsilon = \bigcup_{m=0}^{m-1} A_{\epsilon,m}, \tag{6}$$

where

$$A_{\epsilon,m} = \left\{ Z_{1:2n} \middle| \sup_{\alpha_1, \alpha_2 \in \Lambda} \left[\nu^m(Z_{n+1:2n}, \alpha_1) - \nu^m(Z_{1:n}, \alpha_2)\right] \geq \epsilon \right\}. \tag{7}$$

The only way a $Z_{1:2n} = z_{1:2n}$ can be in $A_{\epsilon,m}$ is that at least one value of j satisfies

$$\sup_{\alpha_1,\ \alpha_2 \in\ \Lambda} \left(\nu_j^m(z_{n+1:2n}, \alpha_1) - \nu_j^m(z_{1:n}, \alpha_2)\right) \geq \frac{\epsilon}{m}.$$

Since A_ϵ is defined on the entire range of our loss function, and we want to partition the range into m disjoint intervals, write

$$A_{\epsilon,m} \subseteq \{ Z_{1:2n} \mid \exists\ j \sup_{\alpha_1,\ \alpha_2 \in \Lambda} \left(\nu_j^m(Z_{n+1:2n}, \alpha_1) - \nu_j^m(Z_{1:n}, \alpha_2)\right) \geq \frac{\epsilon}{m} \} \subseteq \bigcup_{j=0}^{m-1} A_{\epsilon,m,j},$$

where $A_{\epsilon,m,j} = \{Z_{1:2n} \Big| \sup_{\alpha_1,\ \alpha_2 \in \Lambda} (\nu_j^m(Z_{n+1:2n}, \alpha_1) - \nu_j^m(Z_{1:n}, \alpha_2)) \geq \frac{\epsilon}{m}\}$.

Next, fix any value $j \in \{0, 1, \ldots, m-1\}$. For any fixed $z_{1:2n}$, and any given $\alpha_1, \alpha_2 \in \Lambda$, define the vector of length $2n$

$$(Q_j^m(z_{n+1}, \alpha_1), \ldots, Q_j^m(z_{2n}, \alpha_1), Q_j^m(z_1, \alpha_2), \ldots, Q_j^m(z_n, \alpha_2)), \tag{8}$$

where $Q_j^m(z, \alpha) = \mathbb{I}(Q(z, \alpha) \in I_j^m)$. Now define $(\alpha_1, \alpha_2) \sim (\alpha_1', \alpha_2')$ when the corresponding $2n$-tuples are equal (for the given $z_{1:2n}$). It is seen that $\sim$ is an equivalence relation on $\Lambda \times \Lambda$ and therefore partitions $\Lambda \times \Lambda$ into disjoint equivalence classes. Let $K_j = K_j(z_{1:2n})$ be the number of equivalence classes for given j and $z_{1:2n}$ and for given $(\alpha_1, \alpha_2) \in \Lambda \times \Lambda$, write $[(\alpha_1, \alpha_2)]$ for the equivalence class that contains it. Now, for $k = 1 \ldots, K_j$ let

$$\left(\alpha_{1jk}^*, \alpha_{2jk}^*\right) = \arg \sup_{\alpha_1, \alpha_2 \in (\Lambda \times \Lambda)_k} \left(\nu_j^m\left(z_{n+1:2n}, \alpha_1\right) - \nu_j^m\left(z_{1:n}, \alpha_2\right)\right), \tag{9}$$

where $(\Lambda \times \Lambda)_k$ is the k-th equivalence class. Now, $\bigcup_{k=1}^{K_j} \left[(\alpha_{1jk}^*, \alpha_{2jk}^*)\right] = \Lambda \times \Lambda$ and $\left[(\alpha_{1jk}^*, \alpha_{2jk}^*)\right] \cap \left[(\alpha_{1jk}^*, \alpha_{2jk'}^*)\right] = \phi$ unless $k = k'$.

Any permutation π of $\{1, \ldots, 2n\}$ induces a permutation map $T_\pi : \mathcal{Z}^{2n} \to \mathcal{Z}^{2n}$ which acts by shuffling coordinates according to the indices permuted by π. There are $(2n)!$ such maps that can be denoted T_i for $i = 1, \ldots, 2n$. The IID assumption implies that the distribution of any $T_i(Z_{1:2n})$ is the same as the distribution of $Z_{1:2n}$. So, if any function $f : \mathcal{Z}^{2n} \to \mathbb{R}$ satisfies the symmetry condition $f(T_i(z_{1:2n})) = f(z_{1:2n})$ and is integrable, its integral satisfies

$$\int_{\mathcal{Z}^{2n}} f\left(Z_{1:2n}\right) \mathrm{d}P^{2n}\left(z_{1:2n}\right) = \int_{\mathcal{Z}^{2n}} f\left(T_i Z_{1:2n}\right) \mathrm{d}P^{2n}\left(z_{1:2n}\right), \tag{10}$$

in which $\mathrm{d}P^{2n}(z_{1:2n}) = \mathrm{d}P(z_1) \cdots \mathrm{d}P(z_{2n})$ and $P \in \mathcal{M}(\mathcal{Z})$.

One of the quantities that will be essential to getting a tight enough bound on $P^{2n}(A_\epsilon)$ is the annealed entropy $H_{ann}^{\Lambda}(2n)$. Given a sample, say $z_{1:2n}$, let $N^{\Lambda}(z_{1:2n})$ be the number of different separations of $z_{1:2n}$ by a given set of functions. In the proof we will choose all the functions in (8) for a given j. Since $N^{\Lambda}(z_{1:2n}) \leq 2^{2n}$ and the $N^{\Lambda}(z_{1:2n})$'s are measurable, $\mathbb{E}N^{\Lambda}(Z_{1:2n})$ exists. The annealed entropy is the natural logarithm (base e) of this, $H_{ann}^{\Lambda}(2n) = \log \mathbb{E}N^{\Lambda}(Z_{1:2n})$. As is customary, $\mathbb{E}$ means expectation in the true distribution, $P \in \mathcal{M}(\mathcal{Z})$.

Our first main result is similar to the corrresponding result in Vapnik et al. (1994). However, there are numerous differences are in the details. For instance, our equivalence class is defined on $\Lambda \times \Lambda$, we use a cross-validation form of the error, we discretized the loss function, and our result leads to Theorem 2 that only has one term, whereas the corresponding result in Vapnik et al. (1994) has three terms.

Theorem 1 : *Let $\epsilon \geq 0$ and $m \in \mathbb{N}$. If $d_{VC} = VC\left(\{Q\left(\cdot, \alpha\right) : \alpha \in \Lambda\}\right)$ is finite, then*

$$P^{2n}\left(A_\epsilon\right) \leq 2m \left(\frac{2ne}{d_{VC}}\right)^{d_{VC}} \exp\left\{-\frac{n\epsilon^2}{m^2}\right\}. \tag{11}$$

Remark: The technique used to prove (11) is similar to the proof of Theorem 4.1 in Vapnik (1998) giving bounds for the uniform convergence of the empirical risk. The hypotheses of Theorem 4.1 in Vapnik (1998) require only the existence of the key quantities e.g the annealed entropy, and the growth function. Our only extra condition is that d_{VC} be finite.

Proof : Let $m \in \mathbb{N}$ and $j \in \{0, 1, \ldots, m-1\}$. For any given $z_{1:2n}$, α_1, and α_2 write $\Delta_j^m\left(z_{1:2n}, \alpha_1, \alpha_2\right) = \nu_j^m\left(z_{n+1:2n}, \alpha_1\right) - \nu_j^m\left(z_{1:n}, \alpha_2\right)$. Also, denote

$$\begin{aligned} \left(\alpha_{1j}^*, \alpha_{2j}^*\right) &= \arg \sup_{(\alpha_1, \alpha_2) \in \Lambda \times \Lambda} \Delta_j^m\left(z_{1:2n}, \alpha_1, \alpha_2\right) \\ &= \arg \sup_{(\alpha_1, \alpha_2) \in \Lambda \times \Lambda} \left[\nu_j^m(z_{n+1:2n}, \alpha_1) - \nu_j^m(z_{1:n}, \alpha_2)\right] \end{aligned} \tag{12}$$

It is seen that $(\alpha_{1j}^*, \alpha_{2j}^*)$ are estimated using D^2 and D^1 respectively; this reversal of the estimators with respect to the data is the essence of the cross-validation form of the error that we use. Using some manipulations, we have by dropping the superscript $2n$ on P:

$$\begin{aligned} P\left(A_\epsilon\right) &\leq P\left(\bigcup_{j=0}^{m-1} A_{\epsilon,m,j}\right) \leq \sum_{j=0}^{m-1} P\left(A_{\epsilon,m,j}\right) \\ &= \sum_{j=0}^{m-1} P\left(\left\{z_{1:2n} : \sup_{\alpha_1,\ \alpha_2 \in \Lambda} \left(\nu_j^m(z_{n+1:2n}, \alpha_1) - \nu_j^m(z_{1:n}, \alpha_2)\right) \geq \frac{\epsilon}{m}\right\}\right). \\ &= \sum_{j=0}^{m-1} P\left(\left\{z_{1:2n} : \sup_{\alpha_1,\ \alpha_2 \in \Lambda} \Delta_j^m(z_{1:2n}, \alpha_1, \alpha_2) \geq \frac{\epsilon}{m}\right\}\right) \\ &= \sum_{j=0}^{m-1} P\left(\left\{z_{1:2n} : \Delta_j^m(z_{1:2n}, \alpha_{1j}^*, \alpha_{2j}^*) \geq \frac{\epsilon}{m}\right\}\right). \end{aligned}$$

Now, for each T_i write $T_i(z_{1:2n})$ for the permuted sample and correspondingly write $D^1_{T_i(z_{1:2n})}$ and $D^2_{T_i(z_{1:2n})}$ for the first and second halves of the permuted sample. This implies

$$\Delta_j^m(T_i(z_{1:2n}), \alpha_1, \alpha_2) = \nu_j^m(D^2_{T_i(z_{1:2n})}, \alpha_1) - \nu_j^m(D^1_{T_i(z_{1:2n})}, \alpha_2).$$

By symmetry of this function and (10) we can write

$$P\left(\left\{z_{1:2n} : \Delta_j^m(z_{1:2n}, \alpha_{1j}^*, \alpha_{2j}^*) \geq \frac{\epsilon}{m}\right\}\right) = \frac{1}{(2n)!} \sum_{i=1}^{(2n)!} P\left(\left\{z_{1:2n} : \Delta_j^m(T_i(z_{1:2n}), \alpha_{1j}^*, \alpha_{2j}^*) \geq \frac{\epsilon}{m}\right\}\right)$$

and therefore $P(A_\epsilon)$ is bounded from above by

$$\frac{1}{(2n)!} \sum_{j=0}^{m-1} \sum_{i=1}^{(2n)!} P\left(\left\{z_{1:2n} : \Delta_j^m(T_i(z_{1:2n}), \alpha_{1j}^*, \alpha_{2j}^*) \geq \frac{\epsilon}{m}\right\}\right). \tag{13}$$

Using the properties of the equivalence relation $\sim$ and letting Z' denote a dummy variable with the same distribution as Z, we have that for each fixed i, j and $z_{1:2n}$

$$\begin{aligned} I_{\left\{Z'_{1:2n}:\Delta_j^m\left(T_iZ'_{1:2n},\alpha^*_{1j},\alpha^*_{2j}\right)\geq\frac{\epsilon}{m}\right\}}(\cdot) &\leq I_{\left\{Z'_{1:2n}:\Delta_j^m\left(T_iZ'_{1:2n},\alpha^*_{1j1},\alpha^*_{2j1}\right)\geq\frac{\epsilon}{m}\right\}}(\cdot) \\ &+ \cdots + I_{\left\{Z'_{2n}:\Delta_j^m\left(T_iZ'_{1:2n},\alpha^*_{1jK_j(z_{1:2n})}\alpha^*_{2jK_j(z_{1:2n})}\right)\geq\frac{\epsilon}{m}\right\}}(\cdot) \\ &= \sum_{k=1}^{K_j(z_{1:2n})} I_{\left\{Z'_{1:2n}:\Delta_j^m\left(T_iZ'_{1:2n},\alpha^*_{1jk}\alpha^*_{2jk}\right)\geq\frac{\epsilon}{m}\right\}}(\cdot)\,. \end{aligned} \tag{14}$$

The inequality in (14) follows because each $z'_{1:2n}$ making the indicator function on the left side 1, must make at least one of the indicators on the right 1. This follows from the fact that $(\alpha^*_{1j},\alpha^*_{2j})$ is a global maximum and each $(\alpha^*_{1jk},\alpha^*_{2jk})$ is a local maximum for n equivalence class, see (12) and (9). Note that in (14) a T_i appears. Formally, this necessitates choosing $(\alpha^*_{1j},\alpha^*_{2j})$ and each $(\alpha^*_{1jk},\alpha^*_{2jk})$ for given k to be dependent on the i in T_i also; this extra step is suppressed in the notation since i has been dropped for ease of exposition.

Now, using (14), (13) is bounded by

$$\begin{aligned} P\left(A_\epsilon\right) &\leq \frac{1}{(2n)!}\sum_{j=0}^{m-1}\sum_{i=1}^{(2n)!}\int\sum_{k=1}^{K_j(z_{1:2n})} I_{\left\{Z'_{1:2n}:\Delta_j^m(T_iZ'_{1:2n},\alpha^*_{1jk}\alpha^*_{2jk})\geq\frac{\epsilon}{m}\right\}}(z_{1:2n})\,\mathrm{d}P\left(z_{1:2n}\right) \\ &= \int\sum_{j=0}^{m-1}\sum_{k=1}^{K_j(z_{1:2n})}\left[\frac{1}{(2n)!}\sum_{i=1}^{(2n)!} I_{\left\{Z'_{1:2n}:\Delta_j^m(T_iZ'_{1:2n},\alpha^*_{1jk},\alpha^*_{2jk})\geq\frac{\epsilon}{m}\right\}}(z_{1:2n})\right]\mathrm{d}P\left(z_{1:2n}\right). \end{aligned}$$

To bound the summation in square brackets, we follow Vapnik (1998), Chap. 4. Let

$$A_{\epsilon,m,j,k} = \left\{Z_{1:2n} : \Delta_j^m\left(T_iZ_{1:2n},\alpha^*_{1jk}\alpha^*_{2jk}\right)\geq\frac{\epsilon}{m}\right\}$$

for fixed j and each k, where $\left[(\alpha^*_{1jk},\alpha^*_{2jk})\right] = (\Lambda\times\Lambda)_k$. Now, the summation in square brackets is the fraction of the number of the $(2n)!$ permutations T_i of $Z_{1:2n}$ for which $A_{\epsilon,m,j,k}$ is closed under T_i for any fixed equivalence class $(\Lambda\times\Lambda)_k$. As proved in Vapnik (1998) Sec. 4.13, it equals

$$\Gamma_k = \sum_\ell \frac{\binom{b}{\ell}\binom{2n-b}{n-\ell}}{\binom{2n}{n}}$$

where $b = b(z_{1:2n})$ is the number of z_i's in $z_{1:2n}$ that satisfy $Q(z_i,\alpha^*_{1jk}) = 1$ (for $i = 1,\ldots,n$) or $Q(z_i,\alpha^*_{2jk}) = 1$ (for $i = n+1,\ldots,2n$), see Vapnik (1998), p. 136 or 143. The summation is over ℓ's in the set

$$\left\{\ell : \left|\frac{\ell}{n} - \frac{b-\ell}{n}\right| \geq \frac{\epsilon}{m}\right\}.$$

From Sec. 4.13 in Vapnik (1998), we have $\Gamma_k \leq 2\exp\left(-\frac{n\epsilon^2}{m^2}\right)$ uniformly in k.

So, using this in the last bound on $P(A_\epsilon)$ gives that $P\left(A_{\epsilon,m}\right)$ is bounded from above by

$$\begin{aligned} &\int\sum_{j=0}^{m-1}\sum_{k=1}^{K_j(z_{1:2n})} 2\exp\left(-\frac{n\epsilon^2}{m^2}\right)dP(z_{1:2n}) = 2\exp\left(-\frac{n\epsilon^2}{m^2}\right)\int\sum_{j=0}^{m-1}\sum_{k=1}^{K_j(z_{1:2n})} dP(z_{1:2n}) \\ &= 2\exp\left(-\frac{n\epsilon^2}{m^2}\right)\sum_{j=0}^{m-1}\int\sum_{k=1}^{K_j(z_{1:2n})} dP(z_{1:2n}) = 2\exp\left(-\frac{n\epsilon^2}{m^2}\right)\sum_{j=0}^{m-1}\int K_j(z_{1:2n})dP(z_{1:2n}) \\ &= 2\exp\left(-\frac{n\epsilon^2}{m^2}\right)\sum_{j=0}^{m-1} E\left(K_j(Z_{1:2n})\right). \end{aligned} \tag{15}$$

Since $K_j(z_{1:2n})$ is the number of equivalence classes given α_1, α_2, j, and $z_{1:2n}$ and N^Λ is the number of separations of $z_{1:2n}$ given by the functions in (8) i.e., over all $\alpha_1,\alpha_2\in\Lambda$, we have that

$$K_j(z_{1:2n}) \leq N^\Lambda(z_{1:2n}).$$

The reasoning is as follows and simply makes the reasoning behind the statement at the top of p. 136 in Vapnik (1998) explicit. Recall $K_j(z_{1:2n})$ is the number of equivalence classes in $\Lambda\times\Lambda$ for fixed j and $z_{1:2n}$. If (α_1,α_2) and (α'_1,α'_2) are in different equivalence classes then

$$\exists u \quad Q_j^m(z_u,\alpha_1)\neq Q_j^m(z_u,\alpha'_1) \quad \text{or} \quad Q_j^m(z_u,\alpha_2)\neq Q_j^m(z_u,\alpha'_2).$$

Without loss of generality, suppose the first inequality holds for some u. Then the two functions $Q_j^m(z_u,\alpha_1), Q_j^m(z_u,\alpha'_1)$ must assume values $(0,1)$ or $(1,0)$. Again, without loss of generality suppose the first holds. Then, these two functions can separate $z_{1:2n}$ into two disjoint subsets $\{z_v\big|Q_j^m(z_v,\alpha_1)=0\}$ and $\{z_v\big|Q_j^m(z_v,\alpha_1)=0\}$ and this is one of the separations counted by $N^\Lambda(z_{1:2n})$. Taking into account all such separations we have

$$\mathbb{E}K_j(z_{1:2n}) \leq \mathbb{E}N^\Lambda(z_{1:2n}). \tag{16}$$

The growth function is defined to be

$$G^\Lambda(2n) = \log\sup_{z_1,z_2,\ldots,z_{2n}} N^\Lambda\left(z_1,\ldots,z_{2n}\right) \geq \mathbb{E}N^\Lambda(Z_{1:2n}) = H_{ann}(2n).$$

So it is easy to see that $H^\Lambda_{ann}(2n)\leq G^\Lambda(2n)$. Now, Theorem 4.3 from Vapnik (1998) p.145 gives that

$$G(2n)\leq d_{VC}\log\left(\frac{2ne}{d_{VC}}\right)\Rightarrow \mathbb{E}\left(N^\Lambda(z_{1:2n})\right)\leq\left(\frac{2ne}{d_{VC}}\right)^{d_{VC}}. \tag{17}$$

Using (17) and (16) m times in (15) gives the theorem. ■

Next, we use use Theorem 1 to identify an objective function that can be minimized to give an estimator for d_{VC}. Formally, let

$$\Delta_m = \mathbb{E}\left(\sup_{\alpha_1,\alpha_2\in\Lambda}\left|\nu^m(D^2,\alpha_1)-\nu^m(D^1,\alpha_2)\right|\right) \tag{18}$$

and

$$\Delta = \mathbb{E}\left(\sup_{\alpha_1,\alpha_2\in\Lambda}\left|\nu\left(D^2,\alpha_1\right)-\nu\left(D^1,\alpha_2\right)\right|\right). \tag{19}$$

Obviously, $\Delta_m \approx \Delta$ provided that m, n, and $d_{VC} \to \infty$ at appropriate rates and the argument of Δ_m satisfies appropriate uniform integrability conditions. In fact, we do not use $\Delta_m \to \Delta$. For our purpose, the following bounds are sufficient. They are important to our methodology because they bound the expected maximum difference between two values of the empirical losses by an expression that can be used to estimate the VC dimension.

Theorem 2 :

1. *If $d_{VC} < \infty$, we have*

$$\Delta_m \leq m\sqrt{\frac{1}{n}\log\left(2m^3\left(\frac{2ne}{d_{VC}}\right)^{d_{VC}}\right)} + \frac{1}{m\sqrt{n\log\left(2m^3\left(\frac{2ne}{d_{VC}}\right)^{d_{VC}}\right)}} \tag{20}$$

2. *If $d_{VC} < \infty$, and*

$$D_p(\alpha) = \int_0^\infty \sqrt[p]{P\left\{Q\left(z,\alpha\right)\geq c\right\}}dc \leq \infty$$

where $1 < p \leq 2$ is some fixed parameter, we have

$$\Delta \leq \frac{D_p(\alpha^*)2^{2.5+\frac{1}{p}}\sqrt{d_{VC}\log\left(\frac{ne}{d_{VC}}\right)}}{n^{1-\frac{1}{p}}} + \frac{16D_p(\alpha^*)2^{2.5+\frac{1}{p}}}{n^{1-\frac{1}{p}}\sqrt{d_{VC}\log\left(\frac{ne}{d_{VC}}\right)}}. \tag{21}$$

3. *Assume that $d_{VC} \to \infty$, $\frac{n}{d_{VC}} \to \infty$, $m \to \infty$, $\log(m) = o(n)$, and*

$$D_p(\alpha) = \int_0^\infty \sqrt[p]{P\left\{Q(z,\alpha)\geq c\right\}}dc \leq \infty$$

where $p = 2$. Then we have that

$$\Delta \leq \min\left(1, 8D_p(\alpha^*)\right)\sqrt{\frac{d_{VC}}{n}\log\left(\frac{2ne}{d_{VC}}\right)}. \tag{22}$$

Proof : Proofs of the three clause of Theorem 2 can be found in Mpoudeu (2017) in Appendices A1–A3. They rest on using the integral of probabilities identity and then bounding the probabilities as in Theorem 1. ■

We can also use Theorem 1 to obtain an upper bound on the unknown true risk via the following propositions. Let $Q(\alpha_k)$ be the true unknown risk at α_k and $Q_{emp}(\alpha_k)$ be the empirical risk at α_k. Assume that K values of α_k have been fixed. These correspond to a set of points in the parameter space; in our examples below we use estimates. In effect, we are assuming that given an estimate, there is an α_k so close to it that the approximation error is negligible.

Proposition 3 : *For any $\eta \in (0,1)$, with probability at least $1-\eta$, the inequality*

$$Q(\alpha_k) \leq Q_{emp}(\alpha_k) + m\sqrt{\frac{1}{n}\log\left(\left(\frac{2m}{\eta}\right)\left(\frac{2ne}{d_{VC}}\right)^{d_{VC}}\right)} \tag{23}$$

holds simultaneously for all functions $Q(z,\alpha_k)$, $k = 1, 2, \cdots, K$.

Remark: This inequality follows from the additive Chernoff bounds (see, e.g., Vapnik (1998), formulae (4.4) and (4.5)) and suggests that the best model will be the one that minimizes the RHS of (23). The use of (23) in model selection as a form of risk minimization because as d_{VC} increases the second term on the right increases. This limits the size of d_{VC}; we denoted this technique by $PERM_1$ since a penalized empirical risk is being minimized.

Proof : To obtain inequality (23), we equate the RHS of Theorem 1 to a positive number $0 \leq \eta \leq 1$. Thus:

$$\eta = 2m\left(\frac{2ne}{d_{VC}}\right)^{d_{VC}}\exp\left(-\frac{n\epsilon^2}{m^2}\right).$$

Solving for ϵ gives

$$\epsilon \;=\; m\sqrt{\frac{1}{n}\log\left(\left(\frac{2m}{\eta}\right)\left(\frac{2ne}{d_{VC}}\right)^{d_{VC}}\right)}. \tag{24}$$

Proposition 3 can be obtained from the additive Chernoff bounds, expression 4.4 in Vapnik (1998) as follows

$$Q(\alpha_k) \;\leq\; Q_{emp}(\alpha_k) + \epsilon. \tag{25}$$

Using (24) in inequality (25), completes the proof. ■

Parallel to Prop. 3, we have the following for the multiplicative case.

Proposition 4 : *For any $\eta \in (0,1)$, with probability $1-\eta$, the inequality*

$$Q(\alpha_k) \leq Q_{emp}(\alpha_k) + \frac{m^2}{2n}\log\left(\frac{2m}{\eta}\left(\frac{2ne}{d_{VC}}\right)^{d_{VC}}\right)\left(1+\sqrt{1+\frac{4nQ_{emp}(\alpha_k)}{m^2\log\left(\frac{2m}{\eta}\left(\frac{2ne}{d_{VC}}\right)^{d_{VC}}\right)}}\right) \tag{26}$$

holds simultaneously for all K functions in the set $Q(z,\alpha_k)$, $k = 1, 2, \ldots, K$.

Remark: This follows from the multiplicative Chernoff bounds (see e.g., Vapnik (1998) formulae (4.17) and (4.18)) and suggests that the best model will be the one that minimizes the right hand side (RHS) of (26). Analogous to (23) we refer to the use of (26) in model selection as $PERM_2$.

Proof : Let $\epsilon, \eta > 0$. Then, inequality (4.18) in Vapnik (1998) gives, with probability at least $1-\eta$, that

$$\frac{Q(\alpha_k) - Q_{emp}(\alpha_k)}{\sqrt{Q(\alpha_k)}} \leq \epsilon.$$

Routine algebraic manipulations and completing the square give

$$\left(Q(\alpha_k) - 0.5\left(\epsilon^2 + 2Q_{emp}(\alpha_k)\right)\right)^2 - 0.25\left(\epsilon^2 + 2Q_{emp}(\alpha_k)\right)^2 \leq -Q^2_{emp}(\alpha_k).$$

Taking the square root on both sides and re-arranging gives

$$Q(\alpha_k) \leq Q_{emp}(\alpha_k) + 0.5\epsilon^2\left(1 + \sqrt{1 + \frac{4Q_{emp}(\alpha_k)}{\epsilon^2}}\right)$$

Using (24) in the last inequality completes the proof of the Proposition. ■

More details on the use of Propositions 3 and 4 can be found in Vapnik (1998) and Mpoudeu (2017).

2.2. An Estimator of the VC Dimension

The upper bound from Theorem 2 can be written as

$$\Phi_{d_{VC}}(n) = \min(1, 8D_p(\alpha^*))\sqrt{\frac{d_{VC}}{n}\log\left(\frac{2ne}{d_{VC}}\right)}. \tag{27}$$

This expression is meaningfully different from the form derived in Vapnik et al. (1994) and studied in McDonald et al. (2011). Moreover, although $\min(1, 8D_p(\alpha^*))$ does not affect the optimization, it might not be the best constant for the inequality in (22). So, we replace it with an arbitrary constant c over which we optimize to make our upper bound as tight as possible. In our computations, we let c vary from 0.01 to 100 in steps of size 0.01. However, we have observed in practice that the best value of $\hat{c}$ is usually between 1 and 8. The technique that we use to estimate $\hat{d}_{VC}$ is also different from that in Vapnik et al. (1994). Indeed, our Algorithm #1 below accurately encapsulates the way the LHS of (22) is formed unlike the algorithm in Vapnik et al. (1994).

In particular, we use two bootstrapping procedures, one as a proxy for calculating expectations and the second as a proxy for calculating a maximum. Moreover, we split the data set into two subsets. Using the first data set, we fit model I and using the second we fit model II. To explain how we find our estimate of the RHS of (22) from Theorem 2, we start by replacing the sample size n in (27) with a specified value of design point, so that the only unknown is d_{VC}. Thus, formally, we replace (27) by

$$\Phi^*_{d_{VC}}(n_l) = \widehat{c}\sqrt{\frac{d_{VC}}{n_l}\log\left(\frac{2n_le}{d_{VC}}\right)},$$

where $\hat{c}$ is the optimal data driven constant. If we knew the left hand side (LHS) of (22), even computationally, we could use it to estimate d_{VC}. However, in general we don't know the LHS of (22). Instead, we generate one observation of the form

$$\xi(n_l) = \Phi^*_{d_{VC}}(n_l) + \epsilon(n_l) \tag{28}$$

for each design point n_l by bootstrapping and denoted the realized values by $\hat{\xi}(n_l)$. In (28), we assume $\epsilon(n_l)$ has a mean zero, but an otherwise unknown, distribution. We can therefore obtain a list of values of $\widehat{\xi}(n_l)$ for the elements of N_L. In effect, we are assuming that $\Phi^*_{d_{VC}}(n_l)$ provides a tight bound on Δ, and hence Δ_m as suggested by Theorem 2. Our algorithm is as follows.

Algorithm #1

Inputs:

- A collection of regression models $\mathcal{G} = \{g_\beta\}$,
- A data set,
- Two integers b_1 and b_2 for the number of bootstrap samples,
- An integer m for the number of subintervals to discretize the losses,
- A set of design points $N_L = \{n_1, n_2, \ldots, n_L\}$.

For each $l = 1, 2, \ldots, L$ do:

1. Take a bootstrap sample of size $2n_l$ (with replacement) form the data set;
2. Randomly subdivide the bootstrap data into two groups G^1 and G^2 of size n_l each;
3. Fit two models, one for G^1 and one for G^2;
4. Compute the squared error for each model on the covariates and responses that the other model was trained on. Thus:

$$SE_1 = (\text{predict}(Model_1, x_2) - y_2)^2 \quad \text{and} \quad SE_2 = (\text{predict}(Model_2, x_1) - y_1)^2$$

   where (x_1, y_1) ranges over G^1 and (x_2, y_2) ranges over G^2. So, there are n_l values of SE_1 and n_l values of SE_2.
5. Discretize the loss function, i.e. put each SE_1 and SE_2 in one of the m disjoint intervals;
6. Estimate $\nu^m_j(G^2, \alpha_1)$ and $\nu^m_j(G^1, \alpha_2)$ using the SE_1's and SE_2's respectively in the intervals I^m_j for $j = 0, 1, \ldots, m-1$;
7. Compute the differences $|\nu^m_j(G^2, \alpha_1) - \nu^m_j(G^1, \alpha_2)|$ for $j = 0, 1, \ldots, m-1$;
8. Repeat Steps 1-7 b_1 times, take the mean interval-wise and sum it across all intervals, so we have:

$$r_{b_1}(n_l) = \sum_{j=0}^{m-1} \text{mean}\,|\nu^m_j(G^2, \alpha_1) - \nu^m_j(G^1, \alpha_2)|\,;$$

9. Repeat Steps 1-8 b_2 times to get $r_{b_1,i}$ for $i = 1, 2, \ldots, b_2$ and form

$$\hat{\xi}(n_l) = \frac{1}{b_2} \sum_{i=1}^{b_2} r_{b_1,i}(n_l) \,.$$

It is seen that Step 9 uses a mean even though the definition of Δ_m and Δ (see (18) and (19)) has a supremum inside the expectation. This is intentional because using a supremum within each interval gave a worse estimator. We suggest that summing the mean over the intervals performs well because it is not too far from the supermum and is more stable.

Note that this algorithm is parallelizable because different n_l can be sent to different nodes to speed the process of estimating $\hat{\xi}(\cdot)$ for all n_l. After obtaining $\hat{\xi}(n_l)$ for each value of n_l, we estimate d_{VC} by minimizing the squared distance between $\hat{\xi}(n_l)$ and $\Phi^*_{d_{VC}}(n_l)$. Our objective function is

$$f_{n_l}(d_{VC}) = \sum_{l=1}^{L} \left(\hat{\xi}(n_l) - \hat{c} \sqrt{\frac{d_{VC}}{n_l} \log\left(\frac{2n_l e}{d_{VC}}\right)} \right)^2 , \tag{29}$$

where L is the number of design points. Optimizing (29) usually only leads to numerical solutions and in our work below, we set $b_1 = b_2 = W$ for convenience.

3. Proof of Consistency

In this section, we provide a proof of consistency for the estimator $\hat{d}_{VC}$ for d_{VC} that we presented in Subsec. 2.2. In many respects, the structure of this proof should be credited to McDonald et al. (2011). Our contribution is to adapt McDonald et al. (2011) to our stable estimator for the regression context. We begin with some notation and definitions.

Let $\Phi = \{\phi_{d_{VC},c}\}$ be a collection of real valued functions parametrized by $d_{VC} \in \mathbb{H} = [1, M]$ and $c \in I = [a, b] \subset \mathbb{R}$ with $M \in \mathbb{N}$ large enough and $0 < a < b < \infty$ so that $b - a > 0$ is also large enough. Elements of this collection are of the form

$$\phi_{d_{VC},c}(n_l) = c \sqrt{\frac{d_{VC}}{n_l} \log\left(\frac{2n_l e}{d_{VC}}\right)} \tag{30}$$

as derived in Subsec. 2.2 (see expression (27)). In expression (30), we assume L values $n_1, \ldots, n_L$ have been pre-specified. Fix a value of c and let $\Phi_c \subset \Phi$ be the section of elements corresponding to the fixed c. The proof holds for each fixed c and if we optimize over c to obtain $\hat{c}$ as explained in Subsec. 2.2, the convergence of $\hat{d}_{VC}$ to the true value d_{VC} will only be faster.

The collection of functions Φ is the continuous image of a compact set and hence is compact. Now, without loss of generality, we can choose $R > \sup_{d_{VC}} \|\phi_{d_{VC}}\|_L$ where the norm $\|\cdot\|_L$ is derived from the inner product

$$\langle f, g \rangle_L = \frac{1}{L} \sum_{l=1}^{L} f(n_l)\, g(n_l)$$

for real valued functions of a real variable. Thus $\phi_{d_{VC}} = (\phi_{d_{VC}}(n_1), \ldots, \phi_{d_{VC}}(n_L))$ (where the subscript c on the $\phi_{d_{VC},c}(n_L)$'s in expression (30) have been dropped for ease of notation). Fix a value of c and consider the compact subclass of Φ given by

$$\Phi_c(R) = \left\{ \phi \in \Phi_c : \|\phi - \phi_{d_{VC}}\|_L < R \right\}, \tag{31}$$

where $\phi_{d_{VC}}$ is the element of Φ_c corresponding to the correct value of d_{VC}. For a given n_l, we have

$$\hat{\xi}(n_l) = \frac{1}{b_2} \sum_{i=1}^{b_2} r_{b_1,i}(n_l) \tag{32}$$

where $r_{b_1,i}(n_l)$ is the i bootstrapped value of the integrand of Δ_m for each n_l, $i = 1, \ldots, W$ and $l = 1, \ldots, L$. In vector form, write $\hat{\xi} = \left(\hat{\xi}(n_1), \ldots, \hat{\xi}(n_L)\right)$. Using (28), each $\hat{\xi}(n_l)$ can be represented as

$$\hat{\xi}(n_l) = \phi_{d_{VC}}(n_l) + \epsilon(n_l) \,. \tag{33}$$

We have the following result.

Theorem 5 : *Suppose the true $d_{VC} \in [1, M]$ and that $\forall i = 1, \ldots, W$, $\forall l = 1, \ldots, L$, $r_{b_1,i}(n_l) \sim N\left(\phi_{d_{VC}}(n_l), \sigma^2\right)$ and independent, $\mathbb{E}(\epsilon(n_l)) = 0$, $Var(\epsilon(n_l)) = \sigma^2$. Then, on $\Phi_c(R)$, as $n \to \infty$, $m \to \infty$ and $W = W(n) \to \infty$ at suitable rates we have that*

$$P\left(\left\| \phi_{\hat{d}_{VC}} - \phi_{d_{VC}} \right\|_L \geq \delta \right) = \mathcal{O}\left(\frac{1}{W}\right). \tag{34}$$

Remark: In fact, the $r_{b_1,i}(n_l)$'s are only approximately independent $N\left(\phi_{d_{VC}}(n_l), \sigma^2\right)$. However, as n increases they become closer and closer to being independent $N\left(\phi_{d_{VC}}(n_l), \sigma^2\right)$, assuming $\phi_{d_{VC}}(n_l)$ is a tight enough upper bound, as $n, m \to \infty$ at appropriate rates. Also, it is seen that if $L = L(n)$ is increasing then $\|\cdot\|_L$ averages the evaluations of more and more components of, say, $\phi_{\hat{d}_{VC}}$. In the limit, this can be exihibited as an integral, i.e. as a quadratic norm. So, $\|\cdot\|_L$ can be regarded as an approximation of a L^2-space norm that strengthens as a norm (or inner product) as $n \to \infty$. In Theorem 5, if we controlled the distance between $\|\cdot\|_L$ and its limit, we could get a stronger mode of consistency.

Proof : By definition of $\phi_{d_{VC}}$, we have

$$\sum_{l=1}^{L} \left(\hat{\xi}(n_l) - c \sqrt{\frac{\hat{d}_{VC}}{n_l} \log\left(\frac{2n_l e}{\hat{d}_{VC}}\right)} \right)^2 \leq \sum_{l=1}^{L} \left(\hat{\xi}(n_l) - c \sqrt{\frac{d_{VC}}{n_l} \log\left(\frac{2n_l e}{d_{VC}}\right)} \right)^2 \tag{35}$$

or more compactly $\left\| \hat{\xi} - \phi_{\hat{d}_{VC}} \right\|_L^2 \leq \left\| \hat{\xi} - \phi_{d_{VC}} \right\|_L^2$. Expanding both sides of (35) gives

$$\sum_{l=1}^{L} \left(\phi^2_{\hat{d}_{VC}}(n_l) - \phi^2_{d_{VC}}(n_l) \right) \leq 2 \sum_{l=1}^{L} \hat{\xi}(n_l) \left(\phi_{\hat{d}_{VC}} - \phi_{d_{VC}}(n_l) \right)$$

and hence

$$\begin{aligned}\left\|\phi_{\hat{d}_{VC}}\right\|_L^2 - \|\phi_{d_{VC}}\|_L^2 &\leq \frac{2}{L}\sum_{l=1}^{L}\left(\phi_{d_{VC}}(n_l)+\epsilon(n_l)\right)\left(\phi_{\hat{d}_{VC}}(n_l)-\phi_{d_{VC}}(n_l)\right)\\ &= \frac{2}{L}\sum_{l=1}^{L}\Big(\phi_{d_{VC}}(n_l)\,\phi_{\hat{d}_{VC}}(n_l)-\phi_{d_{VC}}^2(n_l)\\ &\qquad +\epsilon(n_l)\left(\phi_{\hat{d}_{VC}}(n_l)-\phi_{d_{VC}}(n_l)\right)\Big).\end{aligned}$$

Rearranging gives

$$\left\|\phi_{\hat{d}_{VC}}\right\|_L^2 - 2\langle\epsilon,\phi_{\hat{d}_{VC}}\rangle + \|\phi_{d_{VC}}\|_L^2 \leq 2\langle\epsilon,\phi_{d_{VC}}-\phi_{\hat{d}_{VC}}\rangle_L,$$

where $\epsilon=(\epsilon(n_1),\ldots,\epsilon(n_L))$, i.e.

$$\left\|\phi_{\hat{d}_{VC}}-\phi_{d_{VC}}\right\|_L^2 \leq 2\langle\epsilon,\phi_{\hat{d}_{VC}}-\phi_{d_{VC}}\rangle_L. \tag{36}$$

It is seen that the LHS is the main quantity we want to control. We have

$$\begin{aligned}P\left(\left\|\phi_{\hat{d}_{VC}}-\phi_{d_{VC}}\right\|_L>\delta\right) &\leq P\left(\langle\epsilon,\phi_{d_{VC}}-\phi_{\hat{d}_{VC}}\rangle\geq\frac{\delta^2}{2}\right)\\ &\leq P\left(\|\epsilon\|_L\left\|\phi_{d_{VC}}-\phi_{\hat{d}_{VC}}\right\|_L>\frac{\delta^2}{2}\right)\\ &\leq \frac{2R^2}{\delta^2}\mathbb{E}\|\epsilon\|_L^2,\end{aligned} \tag{37}$$

using the Cauchy-Schwarz inequality, the bound in (31), and Markov's inequality.

By construction, we have that

$$\begin{aligned}\mathbb{E}\|\epsilon\|_L^2 &= \frac{1}{L}\sum_{l=1}^{L}\mathbb{E}\left(\epsilon^2(n_l)\right)\\ &= \frac{1}{L}\sum_{l=1}^{L}\mathbb{E}\left[\left(\frac{1}{W}\sum_{i=1}^{W}r_{b_1,i}(n_l)\right)-\phi(n_l)\right]^2\\ &= \frac{1}{L}\sum_{l=1}^{L}\mathbb{E}\left[\frac{1}{W}\sum_{i=1}^{W}\left(r_{b_1,i}(n_l)-\phi(n_l)\right)\right]^2\\ &= \frac{1}{LW^2}\sum_{l=1}^{L}\sum_{i=1}^{W}\mathbb{E}\left(r_{b_1,1}(n_l)-\phi(n_l)\right)^2\\ &= \frac{1}{LW^2}\sum_{l=1}^{L}\sum_{i=1}^{W}Var\left(r_{b_1,1}(n_l)\right)=\frac{\sigma^2}{W}.\end{aligned} \tag{38}$$

Using (38) in (37) gives

$$P\left(\left\|\phi_{\hat{d}_{VC}}-\phi_{d_{VC}}\right\|_L\geq\delta\right)\leq\frac{2R^2\sigma^2}{\delta^2W} \tag{39}$$

in which the upper bound decreases as n increases because $W(n)$ is increasing, thereby giving (34). ■

A notable difference between (34) and the corresponding theorem in McDonald et al. (2011) is that our simplified result effectively only gives

$$P\left(\left\|\phi_{\hat{d}_{VC}}-\phi_{d_{VC}}\right\|_L\geq\delta\right)=\mathcal{O}\left(\frac{1}{W}\right) \tag{40}$$

rather than $\mathcal{O}\left(e^{-\gamma W}\right)$ for some $\gamma>0$, a much faster rate. We conjecture that the more sophisticated techniques used in McDonald et al. (2011) could be adapted to our setting and thereby give an exponentially fast rate of convergence of $\hat{d}_{VC}$ to d_{VC} in probability. However, as yet, we have not been able to show this. Also, although it is suppressed in the notation, our result implicitly requires $m\to\infty$ to justify the use of $\phi_{d_{VC}}$.

Using Theorem 5, we can show that our $\hat{d_{VC}}$ is consistent. Suppose that $\phi_{d_{VC}}(\cdot)$ is κ-expansive, or simply expansive when κ is understood, i.e. $\forall n_l, \exists\kappa=\kappa(n_l)$ so that $\kappa(n_l)\left|d_{VC}-d_{VC}^{'}\right|\leq\left|\phi_{d_{VC}}(n_l)-\phi_{d_{VC}^{'}}(n_l)\right|$, where $\kappa(n)$, the expansion factor, is bounded on compact sets. Since the form of $\phi_{d_{VC}}(n_l)$ is known from (27), it is clear that the uniform expansivity condition we have assumed below actually holds, at least for appropriately chosen compact sets. We also observe that for $c\in I$ there exists a neighborhood $B(c,\epsilon_l),\eta>0$, on which (34) is true. Cover $I\times\mathbb{H}$ by sets of the form $B(c,\eta)\times\{d_{VC}\}$; finitely many will be enough since $I\times\mathbb{H}$ is compact.

Theorem 6 : *Given that the assumptions of Theorem 5 hold and that $\phi_{d_{VC}}(\cdot)$ is expansive, we have, as $n\to\infty$, that*

$$P\left(\left|\hat{d_{VC}}-d_{VC}\right|\geq\delta\right)\leq\frac{2R^2\sigma^2}{\delta^2\kappa W}=\mathcal{O}\left(\frac{1}{W}\right),\ \textit{as } W_n\to\infty, \tag{41}$$

where $\kappa=\sqrt{\frac{1}{L}\sum_{l=1}^{L}\kappa(n_l)}$ is the overall expansion factor.

Proof : Since all L of the $\phi_{d_{VC}}(n_l)$'s are at least locally expansive, their local expansivity inequalities can be summarized by an inequality of the form

$$\begin{aligned}\left|d_{VC}-d_{VC}^{'}\right|\sqrt{\frac{1}{L}\sum_{l=1}^{L}\kappa(n_l)} &\leq \sqrt{\frac{1}{L}\sum_{l=1}^{L}\left(\phi_{d_{VC}}(n_l)-\phi_{d_{VC}^{'}}(n_l)\right)^2}\\ &= \|\phi_{d_{VC}}(n_l)-\phi_{d_{VC}^{'}}(n_l)\|_L,\end{aligned} \tag{42}$$

where d_{VC} is the true value and $d_{VC}^{'}$ is any other value in $\mathbb{H}$, and any extra constant from the local expansion factors are assumed to have been absorbed into the $\kappa(n_l)$'s as needed. Let $\kappa=\sqrt{\frac{1}{L}\sum_{l=1}^{L}\kappa(n_l)}$. Using Theorem 5, and (42) we have

$$\begin{aligned}P\left(\left|\hat{d}_{VC}-d_{VC}\right|\geq\delta\right) &\leq P\left[\|\phi_{\hat{d}_{VC}}(n_l)-\phi_{d_{VC}}(n_l)\|_L\geq\delta\kappa\right]\\ &\leq\frac{2R^2\sigma^2}{\kappa\delta^2W},\end{aligned} \tag{43}$$

where the last upper bound decreases as $W = W_n \to \infty$ as $n \to \infty$, giving (41). ■

4. Numerical Comparisons

For any model, we can estimate the LHS of (22) from Theorem 2 by Algorithm #1 in Sec. 2.2. Then, we can use nonlinear regression in (29) to find $\hat{d}_{VC}$. So, it is seen that $\hat{d}_{VC}$ is a function of the conjectured model. In principle, for any given model class, the VC dimension can be found, so our method can be applied.

Since our goal is to estimate the true VC dimension, when a conjectured model $P(\cdot \mid \beta)$ is linear and correct, we expect $VC(P(\cdot \mid \beta)) \cong \hat{d}_{VC}$. By the same logic, if $P(\cdot \mid \beta)$ is far from the true model, we expect $VC(P(\cdot \mid \beta)) \gg \hat{d}_{VC}$ or $VC(P(\cdot \mid \beta)) \ll \hat{d}_{VC}$. This suggests we estimate d_{VC} by seeking

$$\hat{d}_{VC} = \arg\min_k \left| VC\left(P_k(\cdot \mid \beta)\right) - \hat{d}_{VC,k} \right|, \tag{44}$$

where $\{P_k(\cdot \mid \beta) \,|k = 1, 2, \cdots, K\}$ is some set of models and $\hat{d}_{VC,k}$ is calculated using model k, t is a positive and usually small number that such that $t \leq 2$. In the case of linear models, with $q = 1, 2, \cdots, Q$ explanatory variables, we get

$$\hat{d}_{VC} = \arg\min_q \left| q - \hat{d}_{VC,q} \right|, \tag{45}$$

where $\hat{d}_{VC,q}$ is the estimated VC dimension for model of size q. Note that (44) can identify a good model even when consistency fails. The reason is that (44) only requires a minimum at the VC dimension not convergence to the true VC dimension which may be any model under consideration. Here, to achieve uniqueness we use (45) and choose the smalest q achieving the smallest value of $|q - \hat{d}_{VC,q}|$, provided this makes sense in context.

Our numerical work uses linear models, since for these we know the VC dimension equals the number of explanatory variables, see Anthony and Bartlett (2009). To establish notation, we write the regression function as a linear combination of the fixed effect covariates x_j, $j = 0, 1, \cdots, p$,

$$y = f(x, \beta) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = \sum_{j=0}^{p} \beta_j x_j.$$

Given a data set, $\{(x_i, y_i),\ i = 1, 2, \ldots, n\}$, the matrix representation is $Y = X\beta + \epsilon$ where Y is the $n \times 1$ vector of response values, X is the $n \times (1 + p)$ matrix with rows $(1, x_{1,1}, x_{1,2}, \ldots, x_{1,p})$, $\beta = (\beta_0, \beta_1, \ldots, \beta_p)^T$ is the vector of model parameters, and $\epsilon = (\epsilon_1, \epsilon_2, \ldots, \epsilon_n)^T$ is a $n \times 1$ mean zero Gaussian random vector. The least squares estimator $\widehat{\beta}$, assuming it exists, is given by $\widehat{\beta} = \left(X'X\right)^{-1} X'Y$.

4.1. Simulated data

We simulate data from

$$Y = \beta_0 x_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon,$$

where $\epsilon \sim N(0, \sigma_\epsilon = 0.4)$ and

$$x_0 = 1,\ \beta_j \sim N(\mu = 5, \sigma_\beta = 3), x_j \sim N(\mu = 5, \sigma_x = 2),\ \text{for } j = 1, 2, \cdots p,$$

in which all the β's, x's and ϵ's are independently generated. We center and scale all our variables, including the response. For convenience, we use a nested sequence of model lists. If our covariates were highly correlated, before applying our method we could de-correlate them by sphering, i.e. transforming the covariates using their covariance matrix so they become approximately uncorrelated with variance one, see Murphy (2012) p. 144.

In Subsec. 4.1.1, we present a typical simulation result to verify our estimator for VC dimension is consistent for the VC dimension of the true model. In Subsec. 4.1.2, we discuss simulations we have done where results do not initially appear to be consistent with the theory. First, large values of n are needed to get good performance with large values of p. Second as p increases, we must choose n_l's that are properly spread out over $[0, n]$.

4.1.1. A first example

We implemented simulations for a range of model sizes $p = 15, 30, 40, 50, 60$ and 70 and applied six model selection techniques AIC, BIC, CV, $\widehat{PERM}_1$, $\widehat{PERM}_2$, and VC dimension (VCD), see Mpoudeu and Clarke (2018) for details. We tended to use larger sample sizes with larger values of p and spaced the design points uniformly over $[0, n]$, even though this may be suboptimal. We arbitrarily set $m = 10$ and $W = 50$. Our models were nested, including models that were too small and some that were too large, so that the estimate of d_{VC} would uniquely specify a model.

As a typical example, Fig. 1 shows the results for $n = 700$ and $p = 70$ with $L = 7$. When the size of the conjectured model is strictly less than the size of the true model, $\hat{d}_{VC}$ is equal to the smallest design point. However, when the conjectured model exactly matches the true model, $\hat{d}_{VC} \approx 61$, underestimating d_{VC}. Interestingly, if we simply look at the minimal VCD value it occurs at the conjectured model of size 70, the true values of p. When the conjectured model is more complex than the true model, the VCD value is visibly higher than the VCD value for the true model. Thus, using VCD favors parsimony more than the other methods do. In results not shown here, we increased n to 2000 and used good design points (as discussed in Subsec. 4.1.2) and found $\hat{d}_{VC} \approx 70$. Our observation for the other five model selection methods is that they are less affected by the small sample size, but decrease and 'flatline' in the sense that the AIC, BIC and CV values decrease very slowly (making it unclear which model to choose) while $\widehat{PERM}_1$ and $\widehat{PERM}_2$ routinely give models that are too small if one follows the usual rule of choosing the smallest local minimum. Overall, using VCD penalizes models that are too large more than other methods do, thereby giving a clearer statement about which model is true, at least in the limit with intelligently chosen design points. Other choices of n, p, the design points, and other inputs, gave results compatible with this interpretation.

Although the diagrams are not shown here (but see Mpoudeu and Clarke (2018), Sec. 4) the smallest discrepancy between the size d_{VC} of the model and $\hat{d}_{VC}$ usually occurs at the true model. This indicates that $\hat{d}_{VC}$ is consistent. In addition, even though the VCD values generally increase as the size of the conjectured model exceeds the size of the true model, in some cases, past a certain value d_{VC}, the VCD value may flatline as well. The

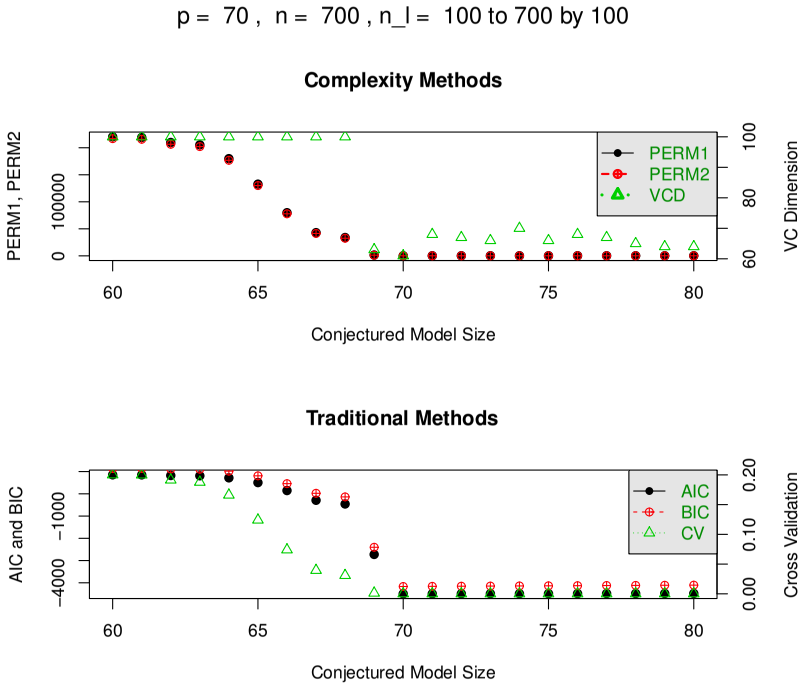


Figure 1: Upper: Values of $\widehat{PERM}_1$, $\widehat{PERM}_2$, and VC dimension. Lower: Values of AIC, BIC and CV for $p = 70$, $\sigma_\epsilon = 0.4$, $\sigma_\beta = 3$, $\sigma_x = 2$.

problem with $\hat{d}_{VC}$ flatlining (or decreasing) past a certain value of d_{VC} occurs mostly due to instability, e.g., when n is not large enough relative to p.

We argue that estimating VC dimension directly is better than using $\widehat{PERM}_1$ or $\widehat{PERM}_2$. There are several reasons. First, the computation of $\widehat{PERM}_1$ and $\widehat{PERM}_2$ requires $\hat{d}_{VC}$. It also requires a threshold η be chosen (see Propositions 3 and 4) and is more dependent on m than $\hat{d}_{VC}$ is. Being more complicated than $\hat{d}_{VC}$, $\widehat{PERM}_1$, $\widehat{PERM}_2$ will break down faster than $\hat{d}_{VC}$. This is seen, for instance in tables of Mpoudeu (2017) Chap. 3 and the discussion there. More generally, we argue that $\widehat{PERM}_1$, and $\widehat{PERM}_2$ break down faster than $\hat{d}_{VC}$ with increasing p, if the sample size is held constant. That is, $\widehat{PERM}_1$ and $\widehat{PERM}_2$ are less efficient than $\hat{d}_{VC}$.

Finally for this subsection, we reiterate our observation that in practice, when our VC dimension technique is used properly i.e., n is large enough relative to p, the σ used in Theorem 6 and the design points are adequately chosen, $\left|\hat{d}_VC - d_{VC}\right|$ has a well defined minimum at the true value of d_{VC}. Also, in contrast with other methods (including using shrinkage methods to nest models) our technique is generally more sensitive to over fit, thereby giving better parsimony.

4.1.2. Dependence of $\hat{d}_{CV}$ on n and N_L

It is no surprise that the higher n/p is the better the discrimination of $\hat{d}_{VC}$ over models is. However, our findings are more complicated because of the design points. The informal rule is that one wants $n \approx 10p$ for good parametric inference. However, this does not take into account model selection that often requires $n > 10p$. Indeed, for good model selection with d_{VC}, we find that $n \geq 15$ is usually sufficient, provided the design points are not badly chosen. In our examples choosing the n_l's uniformly over $[0, n]$ generally gives decent but not optimal performance and for typical ranges of model sizes $L \geq 5$ will suffice even though larger values of L are usually better, say $7 \leq L \leq 10$. Although we are unable at this point to chracterize the tradeoff between design points and sample size, we have noted that in some cases, good choice of design points can compensate for insufficient sample size. Indeed, relatively small changes in the n_l's can have a large numerical effect when n is small; possible due to instability in the nonlinear regression step, (29).

We leave the question of optimally choosing L and the n_l's as future work even though we make the following recommendations: 1) Good choices of n_l's are spread over $[0, n]$. 2. More n_l's should be in $[n/2, n]$ than in $[0, \frac{n}{2}]$, but neither should be empty. 3. Good choices for n_l's tend to remain good while poor choices of n_l's may not be as damaging to inference, as $n \to \infty$. 4. It is better to use fewer design points over a larger range than more design points over a smaller range. 5. As n increases the n_l's should shift upward, i.e., more in $[n/2, n]$. 6. If enough n_l's are large relative to p, $\hat{d}_{VC}$ may be accurate for smaller n, perhaps $n \geq 12p$ will suffice.

4.2. Analysis of a Benchmark Data Set

The goal of this section is to evaluate our method on a 'benchmark' data set Tour obtained from the Tour de France[1]. We start by giving some information about Tour, then in Sec.

1. Tour de France Data was compiled by B. Clarke. More information can be found at http://www.letour.fr/.

4.2.1 we analyze it using a model list based on two of the explanatory variables. The list is a sequence of models nested by SCAD. We evaluate our method by comparing $\hat{d}_{VC}$ to AIC, BIC, CV, $\widehat{PERM}_1$, and $\widehat{PERM}_2$. In Subsec. 4.2.2, we look at the effect of outliers in the estimates $\hat{d}_{VC}$, $\widehat{PERM}_1$, and $\widehat{PERM}_2$.

The full data set Tour has $n = 103$ data points. The data points are dependent (associated) because many cyclists competed in the Tour de France for more than one year. Here we ignore the dependence structure because the dependencies are small enough that the complication of accounting for them is not worthwhile. Each data point has a value of the response variable (Speed), the average speed in kilometers per hour (km/h) of the winner of the Tour. The explanatory variables are the Year (Y) of the Tour and its distance (D) in kilometers. Our data is from 1903 to 2016. However, during World Wars 1 and 2 there was no Tour, so we do not have data points for those years. The effect of World War I on the speed of the winner of the tour can be seen in a scatterplot of Speed vs. Y – the lowest winning speeds were in the years just after World War I, probably due to casualties. After World War II, there was also a decrease in average winning speed, but the decrease was less than after World War I. There is a curvilinear relationship between Speed and Year and a roughly linear relationship between Speed and D; the variability of Speed increases with D.

4.2.1. Analysis of Tour Using a Nested Collection of Models

We identify a nested model list using Y, D, Y^2, D^2 and the interaction between Year and Distance denoted $Y : D$ as covariates. Because the size of the data set is not large, we can only use a small model list.

We order the variables using SCAD because as a shrinkage method it perturbs parameter estimates the least and satisfies an oracle property. Under SCAD, the order of inclusion of variables is Y, D, D^2, Y^2, and $Y : D$. We therefore fit five different models. We use the six model selection techniques from Sec. 4.1 and include $\hat{d}_{VC}$ the original estimator in Vapnik et al. (1994) for the sake of comparison. It is seen that Vapnik et al. (1994)'s original

Model	$\hat{d}_V$	$\hat{d}_{VC}$	$\widehat{PERM}_1$	$\widehat{PERM}_2$	AIC	BIC	CV
Y	20	4	16.42	44.95	84	79.67	0.1294
$Y,\ D$	20	4	15.10	42.83	77	71.66	0.1209
$Y,\ D,\ D^2$	20	4	11.21	36.37	24	**26.21**	0.0727
$Y, D,\ D^2,\ Y^2$	20	4	11.09	36.16	**17**	28.77	**0.0681**
$Y, D,\ D^2,\ Y^2,\ Y : D$	20	4	**11.06**	**36.11**	19	32.96	0.0691

Table 1: Model selection for the Tour de France data using seven methods. The design points for $\hat{d}_V$ and $\hat{d}_{VC}$ are 20, 30, 40, 50, 60, 70, 80, 90, and 100 and $W = 50$. So $\hat{d}_V$ equals the smallest design point in all cases. This problem frequently occurs for $\hat{d}_V$.

method is helpful only if it is reasonable to surmise that there are 15 missing variables whereas our method uniquely identifies one of the models on the list. Even though there is likely no model for Tour de France data set that is accurate to infinite precision, our method is giving a useful result. Indeed, our method is choosing the fourth model list the same as

indicated by AIC and CV. The BIC drops the Y^2 term which is not unreasonable because the curvilinearity is less than quadratic. $\widehat{PERM}_1$ and $\widehat{PERM}_2$ include the interaction term (which can be seen to be zero by a simple t-test). It may also be the case that the derivation of the BIC rests heavily on the independent of data which is not the case here.

4.2.2. Analysis of The Tour de France data set with outliers removed

The observations just after World War I may be outliers. So, we consider the data set formed by deleting the points from 1919 to 1926. Let us see how the six model selection techniques now behave.

The process of analyzing this reduced data set is the same: We identify the nested model lists by SCAD and then find the models corresponding to $\hat{d}_{VC}$, AIC, BIC, CV, $\widehat{PERM}_1$, and $\widehat{PERM}_2$. The results are given in Table 2.

Model Size	$\hat{d}_{VC}$	$\widehat{PERM}_1$	$\widehat{PERM}_2$	AIC	BIC	CV
Y	4	12.87	40.72	67	75	0.1181
$Y,\ D^2$	4	12.01	39.26	69	79	0.1336
$Y,\ D^2,\ D$	4	11.66	38.36	46	59	0.0919
$Y,\ D^2,\ D,\ Y^2$	**4**	11.48	38.34	**28**	**43**	0.0742
$Y,\ D^2,\ D,\ Y^2,\ Y : D$	4	**11.35**	**38.13**	29	46	**0.0735**

Table 2: Model selection for the Tour de France data set with outliers removed.

Under SCAD, the order of inclusion of our covariates is: Y, D^2, D, Y^2 and $Y : D$. This order is different from when we used all data points. The outliers suggest $Y : D$ was more important than it probably is. Note also that when we used all the data points, D was included before D^2 and D^2 was included after Y^2. Again, we fit 5 nested models.

From Table 2, if we choose a model using $\hat{d}_{VC}$, we get the same answer as in Sec. 4.2.1, the model with four variables: Y, D^2, D, Y^2. AIC and BIC choose the same model probably because $(Y : D)$ has low correlation with Speed (-0.08). $\widehat{PERM}_1$, $\widehat{PERM}_2$ and CV choose the model of size 5, which we discount as before because $Y : D$ is only slightly correlated with Speed. That is, the reasoning in Subsec. 4.2.1 for why we think that the model chosen by $\hat{d}_{VC}$ is best continues to hold.

4.3. Application to a Real Data Set

To demonstrate the use of our technique, we re-analyze the Wheat data set presented and studied in Campbell et al. (2003), Dilbirligi et al. (2006), and Dhungana et al. (2007) from a non-complexity based standpoint. The Wheat data set has 2912 observations. More information concerning the data set and the design structure can be found in Campbell et al. (2003). The response variable is YIELD (MG/ha), the covariates that we used are 1000 kernel weight (TKWT), kernels per spike (KPS), Spikes per square meter (SPSM), height of the plant (HT), test weight (TSTWT(KG/hl)), and kernels per square meter (KPSM). Often, in agronomic data sets, there are several classes of explanatory variables, here we have phenotypic, single nucleotide polymorphisms (SNP's) and the variables defining the design. We compare VC dimension based model selection to the methods used in the last subsection and verify our method gives good results.

Our collection of explanatory variables can be grouped into three categories: phenotype, SNP, and design variables. For brevity, we fit only the phenotype variables in Subsec. 4.3.1 and phenotype plus design variables in Subsec. 4.3.2. Comparing these will show that the model selection is unaffected by the design. Also, for brevity, our only analysis is 'multilocation' because we pooleded the data over location-year pairs. A full analysis is in Mpoudeu and Clarke (2018).

4.3.1. Estimation of VC Dimension Using Phenotypic Covariates Only

Intuition suggests

$$YIELD = \beta_0 + \beta_1 \cdot TKWT \cdot KPSM + \epsilon \tag{46}$$

will be a good model because YIELD is essentially the product of the number of kernels and their average weight. Likewise,

$$YIELD = \beta_0 + \beta_1 \cdot TKWT \cdot KPS \cdot SPSM + \epsilon \tag{47}$$

should also be a good model. So, using only phenotpic variables does not lead to a *unique* good model. Both are over simplifications and we can be confident that other influences on YIELD must be considered. Indeed, a 3-dimensional plot of the vectors (YIELD, TKWT, KPSM) looks like a triangle that is bowed out to one side. The bowing means that (46) is only an approximation; other terms are required to explain YIELD. Henceforth, we focus on (46) rather than (47) because we have limited ourselves to second order models.

To implement our multilocation analysis, we first find the order of inclusion of the phenotypic variables in the model, using correlation with YIELD. Then, we find values for $\hat{d}_{VC}$, AIC, BIC, CV, $\widehat{PERM}_1$, $\widehat{PERM}_2$, and the models given by SCAD and ALASSO. Under absolute value of correlation with YIELD, the order of inclusion of the explanatory variables is: TKWT · KPSM, TSTWT · KPSM, KPSM, SPSM · KPS, KPSM^2, KPSM · HT, TKWT · SPSM, TSTWT^2, TSTWT, SPSM · KPSM, KPS · KPSM, TSTWT · SPSM, SPSM SPSM · HT, SPSM^2, TKWT · TSTWT, TKWT, TSTWT · HT, TKWT^2 TKWT · KPS, TSTWT · KPS, TKWT · HT, KPS · HT, HT, HT^2 KPS, KPS^2. So, we consider 27 nested models. This leads to Table 3.

First, we note there is no variability in $\widehat{PERM}_1$ and $\widehat{PERM}_2$ so following standard usage, they select a model with TKWT · KPSM as the only explanatory variable. Likewise, AIC, BIC, and CV suggest the one term model. However, $\hat{d}_{VC} = 14$ means the VC dimension chooses the model with the first 14 terms from the ordered list. SCAD and ALASSO both give the one term model

$$\widehat{YIELD} = 3.43 + 1.12 \cdot TKWT \cdot KPSM, \tag{48}$$

the same as $\widehat{PERM}_1$, $\widehat{PERM}_2$, AIC, BIC and CV. Thus the only reasonable model is the one chosen by $\hat{d}_{VC}$.

4.3.2. Analysis of Wheat Using Phenotypic Data and the Design Structure

Our objective in this subsection is to take the design structure into account and see its impact on the values of the VC dimension and hence on the chosen model. As before, we implement a multilocation analysis. Including design variables forces us to use a more

Size	$\hat{d}_{VC}$	$\widehat{PERM}_1$	$\widehat{PERM}_2$	AIC	BIC	CV
1	13	**7**	**11**	**-9160**	**-9142**	**0.001801809**
2	14	7	11	-9159	-9136	0.001802107
3	13	7	11	-9159	-9130	0.001802470
4	13	7	11	-9159	-9124	0.001803912
5	13	7	11	-9159	-9118	0.001803933
6	14	7	11	-9157	-9110	0.001805230
7	14	7	11	-9156	-9103	0.001805349
8	14	7	11	-9158	-9099	0.001806966
9	14	7	11	-9156	-9091	0.001807176
10	14	7	11	-9154	-9084	0.001808455
11	13	7	11	-9153	-9076	0.001808248
12	13	7	11	-9151	-9069	0.001808966
13	14	7	11	-9150	-9061	0.001810670
14	**14**	7	11	-9148	-9054	0.001812884
15	13	7	11	-9147	-9047	0.001813791

Table 3: The column labeled size gives the number of coefficients for each linear model. The 2^{nd} through 7^{th} columns give the corresponding estimates for $\hat{d}_{VC}$, $\widehat{ERM}_1$, $\widehat{ERM}_2$, AIC, BIC, and CV for Wheat.

complicated bootstrap procedure that would otherwise be sufficient. Thus, to implement our method here, we perform a *restricted* bootstrap. Specifically, we bootstrap in each level of the design variable (incomplete block) so that each half data set has all levels of the design structure. We do this to maintain the design structure and its effects. To include phenotypic variables in the models, we use the same order of inclusion as in Subsec. 4.3.1.

The natural comparison is between Tables 3 and 4. Apart from random variation, they are identical. Moreover, the sparsity methods give exactly the same results in both settings. Thus the conclusions here are the same as in Subsec. 4.3.1: The design variables have no impact on model selection and $\hat{d}_{VC}$ gives the only plausible model.

5. Conclusions

A concise summary of the contributions in this paper is as follows. Sec. 2.1 presents the derivation of the objective function we used to estimate the VC dimension. It is essentially an upper bound on the expected difference between two losses that we have defined as Δ_m or Δ, where the m indicates the discretization of the loss function for a regression problem. In Subsec. 2.2 we give an estimator for d_{VC} that uses our upper bound, nonlinear regression treating sample sizes as design points, a data driven estimator of Δ_m, and an optimization over an arbitrary constant. While this sounds complex, in practice the computations can usually be done in minutes on a regular laptop. Even though we only have an upper bound on Δ_m, in Sec. 3 we are able to give conditions under which our estimator is consistent. This is circumstantial evidence that our upper bound is tight.

Size	$\hat{d}_{VC}$	$\widehat{PERM}_1$	$\widehat{PERM}_2$	AIC	BIC	CV
1	13	**7**	**11**	**-9160**	**-9142**	**0.001801809**
2	13	7	11	-9159	-9136	0.001802107
3	13	7	11	-9159	-9130	0.001802470
4	13	7	11	-9159	-9124	0.001803912
5	13	7	11	-9159	-9118	0.001803933
6	13	7	11	-9157	-9110	0.001805230
7	13	7	11	-9156	-9103	0.001805349
8	13	7	11	-9158	-9099	0.001806966
9	13	7	11	-9156	-9091	0.001807176
10	13	7	11	-9154	-9084	0.001808455
11	13	7	11	-9153	-9076	0.001808248
12	13	7	11	-9151	-9069	0.001808966
13	**13**	7	11	-9150	-9061	0.001810670
14	13	7	11	-9148	-9054	0.001812884

Table 4: The column labeled size gives the number of coefficients for each linear model. The 2^{nd} through 7^{th} columns give the corresponding estimates for $\hat{d}_{VC}$, $\widehat{PERM}_1$, $\widehat{PERM}_2$, AIC, BIC, and CV for multi-location analysis with design structure.

We have done an extensive comparison of our estimator of VC dimension as a model selection method with seven established model selection methods, namely two forms of empirical risk minimization, AIC, BIC, CV, and two sparsity criteria. Other examples can be found in Mpoudeu and Clarke (2018). We did this for the special case of linear models but the same reasoning can be used for any class of nonlinear models e.g., trees, for which the VC dimension can be identified. We also gave one example of how our estimator for VC dimension performs better than the original estimator in Vapnik et al. (1994). As a generality, our method equals or outperforms these other methods.

Acknowledgments

The authors gratefully acknowledge support from NSF grant # DMS-1419754 and invaluable computational support from the Holland Computing Center. Data, code, and results for the full versions of the analyses presented here can be found at `https://github.com/poudas1981/Wheat_data_set.` We also express our gratitude to a sophisticated and hard-working referee and Editor who helped us improve our work immensely.

References

M. Anthony and P. Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University Press, 2009.

A. Barron and T. Cover. Minimum complexity density estimation. *IEEE Transactions on Information theory*, 37(4):1034–1054, 1991.

J. Bernardo and A. F. M. Smith. *Bayesian Theory*. Wiley Series in Probability and Statistics, 2000.

B. Campbell, P. Baenziger, P. Stephen, K. Gill, K. Eskridge, H. Budak, M. Erayman, I. Dweikat, and Y. Yen. Identification of qtls and environmental interactions associated with agronomic traits on chromosome 3a of wheat. *Crop Science*, 43:1493–1505, 2003.

P. Dhungana, K. Eskridge, P. Baenziger, B. Campbell, K. Gill, and I. Dweikat. Analysis of genotype-by-environment interaction in wheat using a structural equation model and chromosome substitution lines. *Crop Science*, 47:477–484, 2007.

M. Dilbirligi, M. Erayman, B. Campbell, H. Randhawa, P. Baenziger, I. Dweikat, and K. Gill. High-density mapping and comparative analysis of agronomically important traits on wheat chromosome 3a. *Genomics*, 88:74 – 87, 2006. ISSN 0888-7543. doi: https://doi.org/10.1016/j.ygeno.2006.02.001. URL `http://www.sciencedirect.com/science/article/pii/S0888754306000437`.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001. doi: 10.1198/016214501753382273. URL `http://dx.doi.org/10.1198/016214501753382273`.

J. Fan and J. Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):849–911, 2008.

D. McDonald, C. Shalizi, and M. Schervish. Estimated VC dimension for risk bounds. preprint arXiv:1111.3404, 2011.

M. Mpoudeu. *Use of Vapnik-Chervonenkis Dimension in Model Selection*. PhD thesis, University of Nebraska-Lincoln, 2017. See: `https://arxiv.org/pdf/1808.06684.pdf`.

M. Mpoudeu and B. Clarke. Model selection via the VC dimension. *See: `https://arxiv.org/abs/1808.05296`*, 2018.

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

X. Shao, V. Cherkassky, and W. Li. Measuring the VC dimension using optimized experimental design. *Neural computation*, 12(8):1969–1986, 2000.

V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Soviet Math. Dokl*, volume 9, pages 915–918, 1968.

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies to their probabilities. In *Soviet Math. Dokl*, volume 9, pages 915–918, 1971.

V. Vapnik, E. Levin, and Y. LeCun. Measuring the VC dimension of a learning machine. *Neural Computation*, 6(5):851–876, 1994.

H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

Journal of Machine Learning Research 20 (2019) 1-43 Submitted 12/17; Revised 1/19; Published 5/19

Dependent relevance determination for smooth and structured sparse regression

Anqi Wu ANQIW@PRINCETON.EDU
Princeton Neuroscience Institute
Princeton University
Princeton, NJ 08544, USA

Oluwasanmi Koyejo SANMI@ILLINOIS.EDU
Beckman Institute for Advanced Science and Technology
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois, 61801, USA

Jonathan Pillow PILLOW@PRINCETON.EDU
Princeton Neuroscience Institute
Princeton University
Princeton, NJ 08544, USA

Editor: David Wipf

Abstract

In many problem settings, parameter vectors are not merely sparse but dependent in such a way that non-zero coefficients tend to cluster together. We refer to this form of dependency as "region sparsity." Classical sparse regression methods, such as the lasso and automatic relevance determination (ARD), which model parameters as independent *a priori*, and therefore do not exploit such dependencies. Here we introduce a hierarchical model for smooth, region-sparse weight vectors and tensors in a linear regression setting. Our approach represents a hierarchical extension of the relevance determination framework, where we add a transformed Gaussian process to model the dependencies between the prior variances of regression weights. We combine this with a structured model of the prior variances of Fourier coefficients, which eliminates unnecessary high frequencies. The resulting prior encourages weights to be region-sparse in two different bases simultaneously. We develop Laplace approximation and Monte Carlo Markov Chain (MCMC) sampling to provide efficient inference for the posterior. Furthermore, a two-stage convex relaxation of the Laplace approximation approach is also provided to relax the inevitable non-convexity during the optimization. We finally show substantial improvements over comparable methods for both simulated and real datasets from brain imaging.

Keywords: Bayesian nonparametric, Sparsity, Structure learning, Gaussian Process, fMRI

©2019 Anqi Wu, Oluwasanmi Koyejo, and Jonathan W. Pillow.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/17-757.html.

1. Introduction

Recent work in statistics has focused on high-dimensional inference problems in which the number of parameters equals or exceeds the number of samples. We focus specifically on the linear regression setting: consider a scalar response $y_i \in \mathbb{R}$ generated from an input vector $\mathbf{x}_i \in \mathbb{R}^p$ via the linear model:

$$y_i = \mathbf{x}_i{}^\top \mathbf{w} + \epsilon_i, \quad \text{for} \quad i = 1, 2, \cdots, n, \tag{1}$$

with observation noise $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$. The regression (linear weight) vector $\mathbf{w} \in \mathbb{R}^p$ is the quantity of interest. This general problem is ill-posed when $n \leq p$. However, it is surprisingly tractable when $\mathbf{w}$ has special structure, such as sparsity in an appropriate basis. A large literature has provided theoretical guarantees about the solvability of such problems, as well as a suite of practical methods for solving them.

Methods based on simple sparsity such as the lasso (Tibshirani, 1996) typically treat regression weights as independent *a priori*. This neglects a statistical feature of many real-world problems, which is that non-zero weights tend to arise in local groups or clusters. In many problem settings, weights have an explicit geometric relationship, such as indexing in time (e.g., time series regression) or space (e.g., brain imaging data). If a single regression weight is non-zero, nearby weights in time or space are also likely to be non-zero. Conversely, in a region where most weights are zero, any particular coefficient is also likely to be zero. Thus, nearby weights exhibit dependencies that are not captured by independent priors. We refer to this form of dependency as *region sparsity*.

A variety of methods have been developed to incorporate local dependencies between regression weights, such as the group lasso (Yuan and Lin, 2006). However, these methods typically require the user to pre-specify the group size or to partition the weights into groups *a priori*. Such information is unavailable in many applications of interest, and hard partitioning into groups breaks dependencies between nearby coefficients that are assigned to different groups.

In this paper, we take a Bayesian approach to inferring regression weights with region-sparse structure. We introduce a hierarchical prior over $\mathbf{w}$ of the form:

$$\mathbf{u} \sim \mathcal{GP} \tag{2}$$

$$\mathbf{w}|\mathbf{u} \sim \mathcal{N}(0, C(\mathbf{u})), \tag{3}$$

where $\mathbf{u}$ is a latent vector that captures dependencies in the sparsity pattern of $\mathbf{w}$, and $\mathbf{w}|\mathbf{u}$ has a zero-mean Gaussian distribution with a diagonal covariance matrix $C(\mathbf{u})$, given by a deterministic function of $\mathbf{u}$. We use a Gaussian process (GP) prior over $\mathbf{u}$ to encode structural assumptions about region sparsity (e.g., the typical size of clusters of non-zero weights and the spacing between them). This model can be seen as an extension of automatic relevance determination (ARD), in which the elements of $\mathbf{u}$ are *a priori* independent (MacKay, 1992; Neal, 1995). We therefore refer to it as *dependent relevance determination* (DRD).

Note that region-sparsity refers only to the sparsity pattern of regression weights, i.e., the locations where they are non-zero, not to the particular values of the weights themselves.

This is reflected in the fact that we define the DRD prior covariance matrix $C(\mathbf{u})$ to be diagonal, making the weights conditionally independent given the pattern defined by $\mathbf{u}$. In many cases, however, we expect weights to be smooth as well as sparse due to the continuity of the input regressors in space or time. Most of the real datasets do exhibit spatial and temporal correlations. Coefficients usually possess contiguous regions and smoothness. Hence, we are aiming at developing a universal approach easily integrating both structured sparsity and smoothness concurrently. To incorporate smoothness, we combine the standard DRD prior with a squared exponential covariance function. The resulting prior has a non-diagonal covariance matrix that encourages smoothness as well as sparsity. We refer to this extension as *smooth dependent relevance determination* (smooth-DRD). Samples from the smooth-DRD prior have local islands of smooth and non-zero weights, surrounded by large regions of zeros. We will show that combining region-sparsity and smoothness together will significantly enhance the performance in a non-trivial way.

Unfortunately, exact inference under DRD and smooth-DRD priors is analytically intractable. We therefore introduce an approximate inference method based on a Laplace approximation to the posterior over $\mathbf{u}$, and a sampling-based inference method using Monte Carlo Markov Chain (MCMC) sampling. We also derive a two-stage convex relaxation of the Laplace approximation approach in order to overcome the effects of bad local optima.

We show experimental evaluations on 1D simulated datasets comparing the performance among different methods. In addition, the phase transition curve analyses are carried out against lasso to show the superiority of DRD and smooth-DRD in support recovery for group structure sparsity with or without smoothness. Furthermore, the DRD based priors are exploited for three brain imaging datasets. Domain expertise and current evidence in brain imaging suggest that discrimination performance is primarily driven by spatially smooth activation within spatially sparse regions, and several estimation algorithms have been proposed that exploit this structure (Grosenick et al., 2011; Michel et al., 2011; Baldassarre et al., 2012; Gramfort et al., 2013). We provide experimental comparisons to these methods, showing the superiority of DRD in practice. In particular, DRD provides spatial decoding weights for brain imaging data that are both more interpretable and achieve higher decoding performance.

Here we highlight our key contributions as follows:

- We introduce a new hierarchical model for smooth, region-sparse weight tensors. The model uses a Gaussian process to introduce dependencies between prior variances of regression weights governing localized sparsity in weights and simultaneously imposes smoothness by integrating a smoothness-inducing covariance function into the prior distribution of weights.

- We describe two methods for inferring the model parameters: one based on the Laplace approximation and a second based on MCMC. We propose a fast approximate inference method based on the Laplace approximation involving a novel two-stage convex relaxation of the log posterior in order to overcome the effects of bad local optima.

- We show phase transition curves governing the transition from imperfect to near-perfect recovery for lasso and DRD estimators, revealing that group structure and smoothness can have a major impact on the recoverability of sparse signals.

This paper is organized as follows. In Sec. 2, we review the related structured sparsity literature. In Sec. 3, we introduce our new region-sparsity and smoothness inducing priors. In Sec. 4, we propose two approaches to Bayesian inference for parameter estimation, the evidence optimization via Laplace approximation and the MCMC sampling. A two-stage convex relaxation of the Laplace approximation approach is also introduced to alleviate the non-convexity with a more robust two-stage convexity. Sec. 5 introduces a detailed analysis of the structured sparsity and smoothness properties of the DRD based priors and the other methods that can be used for this purpose. Sec. 6 presents the phase transition analysis for lasso and DRD estimators. Sec. 7 shows some experiments on three real brain imaging datasets, comparing different methods that can be used for structured sparsity. Finally, Sec. 8 presents the conclusion and discussion of this work.

2. Related work

The classic method for sparse variable selection is the lasso, introduced by Tibshirani (1996), which places an l_1 penalty on the regression weights. This method can be interpreted as a *maximum a posteriori* (MAP) estimate under a Laplace (or double-exponential) prior. A fully Bayesian treatment of this model was later developed by Park and Casella (2008). A variety of Bayesian methods based on other sparsity-inducing prior distributions have been developed, including the horseshoe prior (Carvalho et al., 2009), which uses a continuous density with an infinitely tall spike at the origin and heavy tails, and the spike-and-slab prior (Mitchell and Beauchamp, 1988) which consists of a weighted mixture of a delta function (the spike) and a broad Gaussian (the slab), both centered at the origin.

Another approach to sparse variable selection comes from empirical Bayes (also known as evidence optimization or "type-II" marginal likelihood). These methods rely on a two-step inference procedure: (1) optimize hyperparameters governing the sparsity pattern via ascent of the marginal likelihood; and then (2) compute MAP estimates of the parameters given the hyperparameters. The most popular such estimator is automatic relevance determination (ARD), which prunes unnecessary coefficients by optimizing the precision of each regression coefficient under a Gaussian model (MacKay, 1992; Neal, 1995). The relevance vector machine (RVM) was later formulated as a general Bayesian framework for obtaining sparse solutions to regression and classification tasks (Tipping, 2001). The RVM has an identical functional form to the support vector machine, but provides probabilistic analysis. Tipping and Faul (2003) then was proposed for RVM to scale up its training procedure.

All these methods can be interpreted as imposing a sparse and independent prior on the regression weights. The resulting posterior over weights has high concentration near the axes, so that many weights end up at zero unless forced away strongly by the likelihood.

In the field of structured sparsity learning, group lasso is the most straightforward extension of lasso to capture sparsity existing across collections of variables (Yuan and Lin, 2006).

They achieved the group sparse structure by introducing an l_1 penalty on the l_2 norms of each group. Moreover, Huang et al. (2011) generalized the group sparsity idea by using coding complexity regularization methods associated with the structure. A variety of other papers have proposed alternative approaches to correlated or structured regularization (Jacob et al., 2009; Liu et al., 2009; Kim and Xing, 2009; Friedman et al., 2010; Jenatton et al., 2011; Kowalski et al., 2013).

Previous literature has also explored Bayesian methods for structured sparse inference. A common strategy is to introduce a latent multivariate Gaussian that controls the correlation structure governing conditionally independent densities over coefficients. Gerven et al. (2009) extended the univariate Laplace prior to a novel multivariate Laplace distribution represented as a scale mixture that induces coupling. Hernández-Lobato and Hernández-Lobato (2013) described a similar approach that results in a marginally horseshoe prior. Several other papers have proposed dependent generalizations of the spike-and-slab prior. Hernández-Lobato et al. (2013) described a group spike-and-slab distribution using a multivariate Bernoulli distribution over the indicators of the spikes associated with a group specification. Subsequently, Andersen et al. (2014, 2015) relaxed the hard-coded group specification by encoding the structure with a generic covariance function. Meanwhile, Engelhardt and Adams (2014) introduced a Bayesian model for structured sparsity that uses a Gaussian process (GP) to control the mixing weights of the spike and slab prior in proportion to feature similarity. Apart from imposing the correlation structure on the independent spike and slab elements, Yu et al. (2012) put forward a hierarchical Bayesian framework with the mixing weights of the cluster patterns generated from Beta distributions. Our work is most similar to Engelhardt and Adams (2014) and Andersen et al. (2015), except that we use an ARD-like approach with a conditionally Gaussian density over coefficients instead of a spike and slab prior. Our work is also the first that we are aware of that simultaneously captures sparsity and smoothness.

3. Dependent relevance determination (DRD) priors

In this section, we introduce the DRD prior and the smooth-DRD prior, an extension to incorporate smoothness of regression weights. We focus on the linear regression setting with conditional responses distributed as:

$$\mathbf{y}|X, \mathbf{w}, \sigma^2 \sim \mathcal{N}(\mathbf{y}|X\mathbf{w}, \sigma^2 I), \tag{4}$$

where $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n\times p}$ denotes the design matrix, $\mathbf{y} = [y_1, \cdots, y_n]^\top \in \mathbb{R}^n$ is the observation vector, and σ^2 is the observation noise variance, where p is the dimension of the input vectors and n is the number of samples.

3.1. Automatic relevance determination

The relevance determination framework includes a family of estimators that rely on a zero-mean multivariate normal prior:

$$\mathbf{w}|\boldsymbol{\theta} \sim \mathcal{N}(0, C(\boldsymbol{\theta})), \tag{5}$$

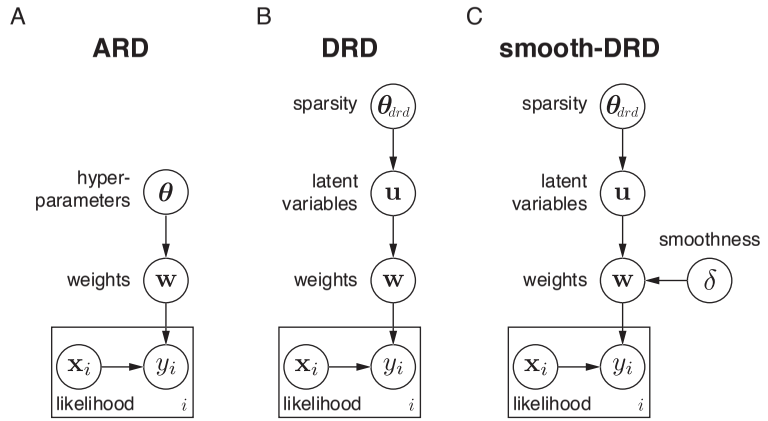


Figure 1: Graphical models for ARD, DRD and smooth-DRD.

where the prior covariance matrix $C(\boldsymbol{\theta})$ is a function of some hyperparameters $\boldsymbol{\theta}$. The form of the dependence of C on θ leads to different forms of assumed structure, including sparsity (Tipping, 2001; Faul and Tipping, 2002; Wipf and Nagarajan, 2008), smoothness (Sahani and Linden, 2003; Schmolck, 2008), or locality (Park and Pillow, 2011).

Automatic relevance determination (ARD) defines the prior covariance to be diagonal, $C_{ii} = \theta_i^{-1}$, where a distinct hyperparameter θ_i specifies the prior precision for the i'th regression coefficient. ARD places an independent improper gamma prior on each hyperparameter, $\theta_i \sim \text{gamma}(0, 0)$, and performs inference for $\{\theta_i\}$ by maximizing the marginal likelihood. This sends many θ_i to infinity, pruning the corresponding coefficients out of the model. A typical graphical model for ARD is presented in Fig. 1A. The independence assumption in the prior over hyperparameters means that there is no tendency for nearby coefficients to remain in or be pruned from the model. This is the primary shortcoming that our method seeks to overcome.

3.2. DRD: A hierarchical extension of ARD

We extend the standard ARD model by adding a level of hierarchy. Instead of directly optimizing hyperparameters that control sparsity of each weight, as in ARD, we introduce a latent vector governed by a GP prior to capture dependencies in the sparsity pattern over weights (see Fig. 1B). Let $\mathbf{u} \in \mathbb{R}^p$ denote a latent vector distributed according to a GP prior

$$\mathbf{u} \sim \mathcal{GP}(b\mathbf{1}, K), \tag{6}$$

where $b \in \mathbb{R}$ is the scalar mean, $\mathbf{1}$ is a length-p vector of ones, and covariance matrix K is determined by a squared exponential kernel. The i,j'th entry of K is given by

$$K_{ij} = \rho \exp\left(-\frac{||\chi_i - \chi_j||^2}{2l^2}\right), \tag{7}$$

where χ_i and χ_j are the spatial locations of weights w_i and w_j, respectively, and kernel hyperparameters are the marginal variance $\rho > 0$ and length scale $l > 0$. Samples from this GP on a grid of locations $\{\chi_i\}$ are smooth on the scale of l, and have mean b and marginal variance ρ.

To obtain a prior over region-sparse weight vectors, we transform $\mathbf{u}$ to the positive values via a nonlinear function f, and the transformed latent vector $\mathbf{g} = f(\mathbf{u})$ forms the diagonal of a diagonal covariance matrix for a zero-mean Gaussian prior over the weights:

$$C_{drd} = \text{diag}\Big[f(\mathbf{u})\Big], \tag{8}$$

where f is a monotonically increasing function that squashes negative values of $\mathbf{u}$ to near zero. Here we will mainly consider the exponential function $f(u) = \exp(u)$, but we will also consider "soft-rectification" function $f(u) = \log(1+\exp(u))$ in the experiment for numerical stability. When the GP mean b is very negative relative to the prior standard deviation $\sqrt{\rho}$, most elements of $\mathbf{g}$ will be close to zero, resulting in weights $\mathbf{w}$ with a high degree of sparsity (i.e., few weights far from zero). The length scale l determines the smoothness of samples $\mathbf{u}$ and thereby determines the typical width of bumps in the prior variance $\mathbf{g}$. We denote the set of hyperparameters governing the GP prior on $\mathbf{u}$ by $\boldsymbol{\theta}_{drd} = \{b, \rho, l\}$. Fig. 2A shows a depiction of sampling from the DRD generative model.

3.3. Smooth-DRD

The standard DRD model imposes smooth dependencies in the prior variances of the regression weights, but the weights themselves remain uncorrelated (as reflected by the fact that the covariance C_{drd} is diagonal). In many settings, however, we expect weights to exhibit smoothness in addition to region sparsity. To capture this property, we can augment DRD with a second Gaussian process, denoted as smooth-DRD, that induces smoothness, contributing off-diagonal structure to the prior covariance matrix while preserving the marginal variance pattern imposed by DRD (see Fig. 1C).

Let Σ denote a covariance matrix governed by a standard squared-exponential GP kernel:

$$\Sigma_{ij} = \exp\left(-\frac{||\chi_i - \chi_j||^2}{2\delta^2}\right), \tag{9}$$

with length scale δ and marginal variance set to 1. Then we define the *smooth-DRD* covariance as the "sandwich" matrix given by:

$$C_{smooth-DRD} = C_{drd}^{\frac{1}{2}} \, \Sigma \, C_{drd}^{\frac{1}{2}}, \tag{10}$$

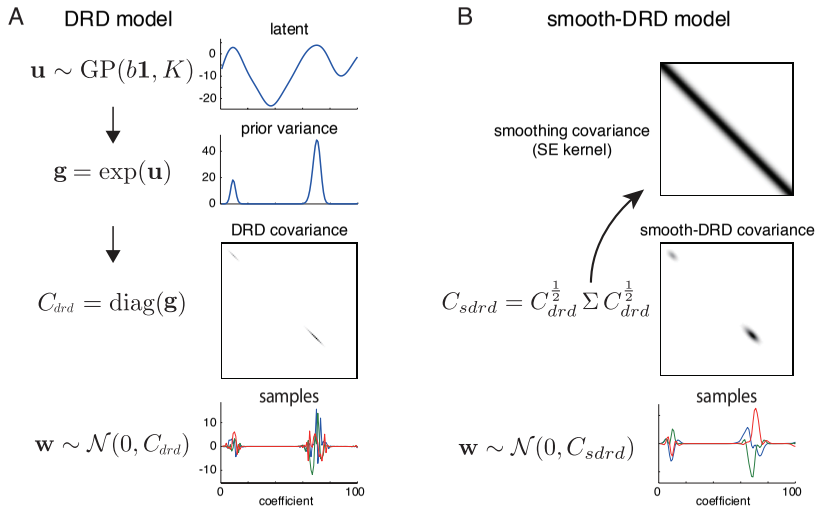


Figure 2: The sampling procedures for the generative models of DRD and smooth-DRD.

where $C_{drd}^{\frac{1}{2}}$ is simply the matrix square root of the diagonal covariance matrix C_{drd}. The resulting matrix has the same diagonal entries as C_{drd}, but has off-diagonal structure governed by Σ that induces smoothness. This matrix is positive semi-definite because, for all $\mathbf{x} \in \mathbb{R}^p$, $\mathbf{x}^\top C_{smooth-DRD}\mathbf{x} = (C_{drd}^{\frac{1}{2}}\mathbf{x})^\top \Sigma (C_{drd}^{\frac{1}{2}}\mathbf{x}) \geq 0$, due to the positive semi-definiteness of Σ. It is therefore a valid covariance matrix. Fig. 2B shows a depiction of sampling from the smooth-DRD generative model. In the following, we will let $\boldsymbol{\theta}$ denote the entire hyperparameter set for the smooth-DRD prior and the noise variance, where $\boldsymbol{\theta} = \{\boldsymbol{\theta}_{drd}, \delta, \sigma^2\}$.

4. Parameter estimation

In this section, we describe two methods for inference under the DRD and smooth-DRD priors: (1) empirical Bayesian inference via evidence optimization using the Laplace approximation; and (2) fully Bayesian inference via MCMC sampling. The first seeks to find the MAP estimate of the latent vector $\mathbf{u}$ governing region sparsity via optimization of the log marginal likelihood, and then provides a conditional MAP estimate of the weights $\mathbf{w}$. The second uses MCMC sampling to integrate over $\mathbf{u}$ and provides the posterior mean of $\mathbf{w}$ given the data via an average over samples.

4.1. Empirical Bayes inference with Laplace approximation

The likelihood $p(\mathbf{y}|X, \mathbf{w}, \sigma^2)$ (eq. 4) and the prior $p(\mathbf{w}|\mathbf{u}, \boldsymbol{\theta}_{drd}, \delta)$ (eq. 5) are both Gaussian given the latent variables $\mathbf{u}$ and hyperparameters $\boldsymbol{\theta}$, giving a conditionally Gaussian posterior over the regression weights:

$$p(\mathbf{w}|X, \mathbf{y}, \mathbf{u}, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\mu}_{\mathbf{w}}, \Lambda_{\mathbf{w}}), \tag{11}$$

with covariance and mean given by

$$\Lambda_{\mathbf{w}} = (\tfrac{1}{\sigma^2} X^\top X + C^{-1})^{-1}, \quad \boldsymbol{\mu}_{\mathbf{w}} = \tfrac{1}{\sigma^2} \Lambda_{\mathbf{w}} X^\top \mathbf{y}, \tag{12}$$

where prior covariance matrix C is a function of $\mathbf{u}$ and $\boldsymbol{\theta}$. The posterior mean $\boldsymbol{\mu}_{\mathbf{w}}$ is also the MAP estimate of $\mathbf{w}$ given latent vector $\mathbf{u}$ and hyperparameters $\boldsymbol{\theta}$.

Empirical Bayes inference involves setting the hyperparameters by maximizing the marginal likelihood or evidence, given by

$$p(\mathbf{y}|X, \boldsymbol{\theta}) = \iint p(\mathbf{y}|X, \mathbf{w}, \sigma^2) p(\mathbf{w}|\mathbf{u}, \delta) p(\mathbf{u}|\boldsymbol{\theta}_{drd})\, d\mathbf{w}\, d\mathbf{u}. \tag{13}$$

We can take the integral over $\mathbf{w}$ analytically due to the conditionally Gaussian prior and the likelihood, giving the simplified expression

$$p(\mathbf{y}|X, \boldsymbol{\theta}) = \int p(\mathbf{y}|X, \mathbf{u}, \sigma^2, \delta)\, p(\mathbf{u}|\boldsymbol{\theta}_{drd})\, d\mathbf{u}, \tag{14}$$

where the conditional evidence given $\mathbf{u}$ is a normal density evaluated at $\mathbf{y}$,

$$p(\mathbf{y}|X, \mathbf{u}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, XCX^\top + \sigma^2 I). \tag{15}$$

However, the integral over $\mathbf{u}$ has no analytic form. We therefore resort to the Laplace's method to approximate this integral.

4.1.1. Laplace approximation

Laplace's method provides a technique for approximating intractable integrals using a second-order Taylor expansion in $\mathbf{u}$ of the log of the integrand in (eq. 14). This method is equivalent to approximating the posterior over $\mathbf{u}$ given $\boldsymbol{\theta}$ by a Gaussian centered on its mode (MacKay (2003), chap. 27). The exact posterior is given by Bayes' rule:

$$p(\mathbf{u}|X, \mathbf{y}, \boldsymbol{\theta}) = \frac{1}{Z}\, p(\mathbf{y}|X, \mathbf{u}, \sigma^2, \delta) p(\mathbf{u}|\boldsymbol{\theta}_{drd}), \tag{16}$$

where the normalizing constant, $Z = p(\mathbf{y}|X, \boldsymbol{\theta})$, is the marginal likelihood we wish to compute. The Gaussian approximation to the posterior is

$$p(\mathbf{u}|X, \mathbf{y}, \boldsymbol{\theta}) \approx \mathcal{N}(\mathbf{m}_{\mathbf{u}}, \Lambda_{\mathbf{u}}), \tag{17}$$

where $\mathbf{m}_{\mathbf{u}}$ is the posterior mode and $\Lambda_{\mathbf{u}}$ is a local approximation to the posterior covariance. Substituting this approximation into (eq. 16), we can directly solve for Z:

$$Z \approx \frac{p(\mathbf{y}|X, \mathbf{u}, \sigma^2, \delta) p(\mathbf{u}|\boldsymbol{\theta}_{drd})}{\mathcal{N}(\mathbf{m}_{\mathbf{u}}, \Lambda_{\mathbf{u}})}. \tag{18}$$

The right-hand-side of this expression can be evaluated at any $\mathbf{u}$, but it is conventional to use the mode, $\mathbf{u} = \mathbf{m}_{\mathbf{u}}$, given that this is where the approximation is most accurate.

To compute the Laplace approximation, we first numerically optimize the log of the posterior (eq. 16) to find its mode:

$$\mathbf{m}_{\mathbf{u}} = \arg\max_{\mathbf{u}} \Big[\log p(\mathbf{y}|X, \mathbf{u}, \sigma^2, \delta) + \log p(\mathbf{u}|\boldsymbol{\theta}_{drd}) \Big], \tag{19}$$

where the first term is the log of the conditional evidence given $\mathbf{u}$ (eq. 15),

$$\log p(\mathbf{y}|X, \mathbf{u}, \sigma^2, \delta) = -\frac{1}{2} \log |XCX^\top + \sigma^2 I| - \frac{1}{2} \mathbf{y}^\top (XCX^\top + \sigma^2 I)^{-1} \mathbf{y} + const, \tag{20}$$

and the second is the log of the GP prior for $\mathbf{u}$,

$$\log p(\mathbf{u}|\boldsymbol{\theta}_{drd}) = -\frac{1}{2} (\mathbf{u} - b\mathbf{1})^\top K^{-1} (\mathbf{u} - b\mathbf{1}) - \frac{1}{2} \log |K| + const. \tag{21}$$

We use quasi-Newton methods to optimize this objective function because the fixed point methods developed for ARD (e.g., MacKay (1992); Tipping and Faul (2003)), which operate on one element of the prior precision vector at a time, are inefficient due to the strong dependencies induced by the GP prior. However, because this high-dimensional optimization problem is non-convex, we also formulate a novel approach for optimizing $\mathbf{u}$ using a two-stage convex relaxation inspired by Wipf and Nagarajan (2008). We will present the method in Sec. 4.1.2.

Given the mode of the log-posterior $\mathbf{m}_{\mathbf{u}}$, the second step to computing the Laplace-based approximation to the marginal likelihood is to compute the Hessian (2nd derivative matrix) of the log-posterior at $\mathbf{m}_{\mathbf{u}}$. The negative inverse of the Hessian gives us the posterior covariance for the Laplace approximation (eq. 17):

$$\Lambda_{\mathbf{u}} = \left(-\frac{\partial^2}{\partial \mathbf{u} \partial \mathbf{u}^\top} \Big[\log p(\mathbf{y}|X, \mathbf{u}, \sigma^2, \delta) + \log p(\mathbf{u}|\boldsymbol{\theta}_{drd}) \Big] \right)^{-1}. \tag{22}$$

See Appendix A for the explicit derivation of Hessian for the DRD model.

Given these ingredients, we can now write down the approximation to the log marginal likelihood (eq. 18):

$$\log p(\mathbf{y}|X, \boldsymbol{\theta}) \approx \log p(\mathbf{y}|X, \mathbf{m}_{\mathbf{u}}, \sigma^2, \delta) + \log p(\mathbf{m}_{\mathbf{u}}|\boldsymbol{\theta}_{drd}) + \tfrac{1}{2} \log |\Lambda_{\mathbf{u}}| + const, \tag{23}$$

where the first term is simply the log conditional evidence (eq. 20) with prior covariance C evaluated at $\mathbf{m}_{\mathbf{u}}$.

It is this log-marginal likelihood that we seek to optimize in order to learn hyperparameters $\boldsymbol{\theta}$. The key difficulty is that the Laplace approximation parameters $\mathbf{m}_{\mathbf{u}}$ and $\Lambda_{\mathbf{u}}$ depend

implicitly on $\boldsymbol{\theta}$ (since $\mathbf{m_u}$ is determined by numerical optimization at a fixed value of $\boldsymbol{\theta}$), making it impractical to evaluate their derivatives with respect to $\boldsymbol{\theta}$. To address this problem, we introduce a method for partially decoupling the Laplace approximation from the hyperparameters (Sec. 4.1.3).

4.1.2. A two-stage convex relaxation to Laplace Approximation

The optimization for $\mathbf{m_u}$ (eq. 19), the mode of the posterior over the latent vector $\mathbf{u}$, is a critical step for computing the Laplace approximation. However, the negative log-posterior is a non-convex function in $\mathbf{u}$, meaning that there is no guarantee of obtaining the global minimum. In this section, DRD resembles the original ARD model. Neither of the two most popular optimization methods for ARD, MacKay's fixed-point method (MacKay, 1992) and Tipping and Faul's fast-ARD (Tipping and Faul, 2003), are guaranteed to converge to a local minimum or even a fixed point of the log-posterior.

In this section, we introduce an alternative formulation of the cost function in (eq. 19) using an auxiliary function: this provides a tight convex upper bound that can be optimized more easily. The technique is similar to the iterative re-weighted l_1 formulation of ARD in Wipf and Nagarajan (2008).

Let $\mathcal{L}(\mathbf{u})$ denote the sum of terms in the negative log-posterior (eq. 19) that involve $\mathbf{u}$,

$$\mathcal{L}(\mathbf{u}) = \frac{1}{2}\log|XCX^\top + \sigma^2 I| + \frac{1}{2}\mathbf{y}^\top(XCX^\top + \sigma^2 I)^{-1}\mathbf{y} + \frac{1}{2}(\mathbf{u} - b\mathbf{1})^\top K^{-1}(\mathbf{u} - b\mathbf{1}), \quad (24)$$

where $C = \mathrm{diag}(e^{\mathbf{u}})$. We denote the three terms it contains as:

$$\mathcal{L}_1(\mathbf{u}) = \frac{1}{2}\log|X\mathrm{diag}(e^{\mathbf{u}})X^\top + \sigma^2 I| \quad (25)$$

$$\mathcal{L}_2(\mathbf{u}) = \frac{1}{2}\mathbf{y}^\top(X\mathrm{diag}(e^{\mathbf{u}})X^\top + \sigma^2 I)^{-1}\mathbf{y} \quad (26)$$

$$\mathcal{L}_3(\mathbf{u}) = \frac{1}{2}(\mathbf{u} - b\mathbf{1})^\top K^{-1}(\mathbf{u} - b\mathbf{1}). \quad (27)$$

Here $\mathcal{L}_1(\mathbf{u})$ and $\mathcal{L}_3(\mathbf{u})$ are both convex in $\mathbf{u}$ (see proof in Appendix B). We can derive a tight convex upper bound for $\mathcal{L}_2(\mathbf{u})$, thus providing a tight convex upper bound for $\mathcal{L}(\mathbf{u})$.

We know that $\mathcal{L}_2(\mathbf{u})$ is non-convex, but we are interested in rewriting it using concave duality. Let $\mathbf{h}(\mathbf{u}) : \mathbb{R}^p \to \Omega \subset \mathbb{R}^p$ be a mapping with range Ω, which may or may not be a one-to-one map. We assume that there exists a concave function $\Phi(\boldsymbol{\eta}) : \Omega \to \mathbb{R}, \forall \boldsymbol{\eta} \in \Omega$, such that $\mathcal{L}_2(\mathbf{u}) = \Phi(\mathbf{h}(\mathbf{u}))$ holds. To exploit this technique, we first rewrite $\mathcal{L}_2$ using the matrix inverse lemma (Higham, 2002) as:

$$\mathcal{L}_2(\mathbf{u}) = \frac{1}{2\sigma^2}\mathbf{y}^\top\mathbf{y} - \frac{1}{2\sigma^4}\mathbf{y}^\top X\left(\frac{1}{\sigma^2}X^\top X + \mathrm{diag}(e^{-\mathbf{u}})\right)^{-1}X^\top\mathbf{y}. \quad (28)$$

Then, setting $\mathbf{h}(\mathbf{u}) = e^{-\mathbf{u}}$, which is convex in $\mathbf{u}$, we have

$$\mathcal{L}_2(\mathbf{u}) = \Phi(\mathbf{h}(\mathbf{u})) = \frac{1}{2\sigma^2}\mathbf{y}^\top\mathbf{y} - \frac{1}{2\sigma^4}\mathbf{y}^\top X\left(\frac{1}{\sigma^2}X^\top X + \mathrm{diag}(\mathbf{h}(\mathbf{u}))\right)^{-1}X^\top\mathbf{y}. \quad (29)$$

Algorithm 1 A two-stage convex relaxation method for DRD Laplace approximation

Input: $X, \mathbf{y}, \boldsymbol{\theta} = \{\sigma^2, \delta, b, \rho, l\}$
Output: $\hat{\mathbf{u}}$
initialize dual variable $\hat{\mathbf{z}}_i = 1$, $\forall i = 1, 2, ..., p$
Repeat the following two steps until convergence:
1. Fix $\hat{\mathbf{z}}$, let $\hat{\mathbf{u}} = \mathrm{argmin}_{\mathbf{u}\in\mathbb{R}^p}\left[\mathbf{z}^\top\mathbf{h}(\mathbf{u}) + \mathcal{L}_1(\mathbf{u}) + \mathcal{L}_3(\mathbf{u})\right]$ in (eq. 33)
2. Fix $\hat{\mathbf{u}}$, let $\hat{\mathbf{z}} = \nabla_{\boldsymbol{\eta}}\Phi(\boldsymbol{\eta})|_{\boldsymbol{\eta}=\mathbf{h}(\hat{\mathbf{u}})}$ in (eq. 34)

This expression is concave in $\mathbf{h}(\mathbf{u})$ (inverse of a matrix is convex), and thus can be expressed as a minimum over upper-bounding hyperplanes via

$$\mathcal{L}_2(\mathbf{u}) = \Phi(\mathbf{h}(\mathbf{u})) = \inf_{\mathbf{z}\in\mathbb{R}^p}\left[\mathbf{z}^\top\mathbf{h}(\mathbf{u}) - \mathcal{L}^*_{\mathbf{h}}(\mathbf{z})\right], \quad (30)$$

where $\mathcal{L}^*_{\mathbf{h}}(\mathbf{z})$ is the concave conjugate of $\Phi(\boldsymbol{\eta})$ that is defined by the duality relationship

$$\mathcal{L}^*_{\mathbf{h}}(\mathbf{z}) = \inf_{\boldsymbol{\eta}\in\mathbb{R}^p}\left[\mathbf{z}^\top\boldsymbol{\eta} - \Phi(\boldsymbol{\eta})\right], \quad (31)$$

and $\mathbf{z}$ is the dual variable. Note, however, that for our purpose it is not necessary to ever explicitly compute $\mathcal{L}^*_{\mathbf{h}}(\mathbf{z})$. This leads to the following upper-bounding auxiliary cost function

$$\Phi(\mathbf{h}(\mathbf{u}), \mathbf{z}) = \mathbf{z}^\top\mathbf{h}(\mathbf{u}) - \mathcal{L}^*_{\mathbf{h}}(\mathbf{z}) \geq \Phi(\mathbf{h}(\mathbf{u})). \quad (32)$$

Thus, it naturally admits the tight convex upper bound for $\mathcal{L}(\mathbf{u})$,

$$\mathcal{L}(\mathbf{u}, \mathbf{z}) \triangleq \mathbf{z}^\top\mathbf{h}(\mathbf{u}) - \mathcal{L}^*_{\mathbf{h}}(\mathbf{z}) + \mathcal{L}_1(\mathbf{u}) + \mathcal{L}_3(\mathbf{u}) \geq \mathcal{L}(\mathbf{u}). \quad (33)$$

Moreover, for any fixed $\boldsymbol{\eta} = \mathbf{h}(\mathbf{u})$, it's well-known that the minimum of the right hand side of (eq. 31) is achieved at

$$\hat{\mathbf{z}} = \nabla_{\boldsymbol{\eta}}\Phi(\boldsymbol{\eta})|_{\boldsymbol{\eta}=\mathbf{h}(\mathbf{u})}. \quad (34)$$

This leads to the general optimization procedure presented in Algorithm 1. By repeatedly refining the dual parameter $\mathbf{z}$, we can obtain a repeatedly improved convex relaxation, leading to a solution superior to that of the initial convex relaxation.

Now we show the analysis of global convergence. According to the Zangwill's *Global Convergence Theorem* (Zangwill, 1969), let $\mathcal{A}(\cdot) : \mathcal{U} \to \mathcal{P}(\mathcal{U})$ be a point-to-set mapping to handle the multi-global minima case, which satisfies Steps 1 and 2 of the proposed algorithm, then

Theorem 1 *From any initialization point $\mathbf{u}^0 \in \mathbb{R}^p$, the sequence of parameter estimates $\{\mathbf{u}^k\}$ generated via $\mathbf{u}^{k+1} \in \mathcal{A}(\mathbf{u}^k)$ is guaranteed to converge monotonically to a local minimum (or saddle point) of $\mathcal{L}(\mathbf{u})$.*

Proof Let $\Gamma \in \mathcal{U}$ be a solution set. In order to use the global convergence theorem, we need to show that

1) all points $\{\mathbf{u}^k\}$ are contained in a compact set $S \in \mathcal{U}$, where $\mathcal{U}$ is $\mathbb{R}^p$ in our problem;
2) there is a continuous function Z on $\mathcal{U}$ such that
(a) if $x \notin \Gamma$, then $Z(y) < Z(x)$ for all $y \in \mathcal{A}(x)$;
(b) if $x \in \Gamma$, then $Z(y) \leq Z(x)$ for all $y \in \mathcal{A}(x)$;
3) the mapping $\mathcal{A}$ is closed at points outside Γ.

First, let's define the mapping $\mathcal{A}$ to be achieved by

$$\mathbf{u}^{k+1} \in \mathcal{A}(\mathbf{u}^k) = \text{argmin}_{\mathbf{u}\in\mathbb{R}^p}\, \mathcal{F}(\mathbf{u}, \mathbf{u}^k) = \text{argmin}_{\mathbf{u}\in\mathbb{R}^p}\, \mathbf{z}^{k\top}\mathbf{h}(\mathbf{u}) - \mathcal{L}^*_{\mathbf{h}}(\mathbf{z}) + \mathcal{L}_1(\mathbf{u}) + \mathcal{L}_3(\mathbf{u}), \quad (35)$$

where $\mathbf{z}^k = \nabla_{\boldsymbol{\eta}}\Phi(\boldsymbol{\eta})|_{\boldsymbol{\eta}=\mathbf{h}(\mathbf{u}^k)}$. We can prove that $\mathcal{F}$ is coercive, i.e., when $||\mathbf{u}|| \to \infty$, we have $\mathcal{F}(\mathbf{u}) \to \infty$ (proof in Appendix C). Therefore, the solution set of $\mathcal{F}(\mathbf{u})$ is bounded and nonempty. Accordingly, $\mathcal{A}(\mathbf{u})$ is nonempty. Using Proposition 7 in (Gunawardana and Byrne, 2005), we can further show that the point-to-set mapping $\mathcal{A}$ is closed at $\mathbf{u} \in \mathcal{U}$. Condition 3 is satisfied.

For each $\mathbf{u}^k$, $\mathbf{u}^{k+1}$ is the solution of $\mathcal{F}(\mathbf{u})$, and $\mathcal{A}(\mathbf{u})$ is a closed mapping; therefore each $\mathbf{u}^{k+1}$ belongs to a compact set. We know that the union of two compact sets is compact. Therefore, all points $\{\mathbf{u}^k\}$ are contained in a compact set $S \in \mathcal{U}$. Condition 1 is satisfied.

To prove condition 2, we must show that for any $\mathbf{u}^k$, $\mathcal{L}(\mathbf{u}^{k+1}) < \mathcal{L}(\mathbf{u}^k)$ for all $\mathbf{u}^{k+1} \in \mathcal{A}(\mathbf{u}^k)$ if $\mathbf{u}^k \notin \Gamma$; $\mathcal{L}(\mathbf{u}^{k+1}) \leq \mathcal{L}(\mathbf{u}^k)$ for all $\mathbf{u}^{k+1} \in \mathcal{A}(\mathbf{u}^k)$ if $\mathbf{u}^k \in \Gamma$. At any $\mathbf{u}^k$, the auxiliary cost function $\mathcal{F}(\mathbf{u})$ (eq. 35) is strictly tangent to $\mathcal{L}(\mathbf{u})$ at $\mathbf{u}^k$. Therefore, if $\mathbf{u}^k \notin \Gamma$, $\mathcal{L}(\mathbf{u}^k) = \mathcal{F}(\mathbf{u}^k) > \mathcal{F}(\mathbf{u}^{k+1}) \geq \mathcal{L}(\mathbf{u}^{k+1})$, thus $\mathcal{L}(\mathbf{u}^k) > \mathcal{L}(\mathbf{u}^{k+1})$; if $\mathbf{u}^k \in \Gamma$, $\mathcal{L}(\mathbf{u}^k) = \mathcal{F}(\mathbf{u}^k) = \mathcal{F}(\mathbf{u}^{k+1}) \geq \mathcal{L}(\mathbf{u}^{k+1})$, thus $\mathcal{L}(\mathbf{u}^k) \geq \mathcal{L}(\mathbf{u}^{k+1})$. Condition 2 is satisfied. ■

The algorithm could theoretically converge to a saddle point, but any minimal perturbation would easily lead to escape.

4.1.3. Decoupled Laplace approximation

To optimize the marginal likelihood for the DRD hyperparameters (eq. 23), we should ideally replace $\mathbf{m}_\mathbf{u}$ and $\Lambda_\mathbf{u}$ with explicit expressions in $\boldsymbol{\theta}$ in order to accurately compute derivatives with respect to $\boldsymbol{\theta}$. However, the deterministic formulation of such functions is intractable. We can nevertheless partially overcome this dependence by introducing a "decoupled" Laplace approximation that takes into account the dependence of $\Lambda_\mathbf{u}$ on the hyperparameters $\boldsymbol{\theta}_{drd}$. Wu et al. (2017) also proposed a conceptually similar decoupled Laplace approximation.

Specifically, we rewrite the inverse Laplace posterior covariance (eq. 22):

$$\Lambda_\mathbf{u} = (\Gamma + \Psi(\boldsymbol{\theta}_{drd}))^{-1} \quad (36)$$

where Γ is the negative Hessian of the log-likelihood (which is independent of $\boldsymbol{\theta}_{drd}$),

$$\Gamma = -\frac{\partial^2}{\partial\mathbf{u}\partial\mathbf{u}^\top}\log p(\mathbf{y}|X, \mathbf{u}, \sigma^2, \delta), \quad (37)$$

and $\Psi(\boldsymbol{\theta}_{drd})$ is the precision matrix of the prior distribution for $\mathbf{u}$,

$$\Psi(\boldsymbol{\theta}_{drd}) = -\frac{\partial^2}{\partial\mathbf{u}\partial\mathbf{u}^\top}\log p(\mathbf{u}|\boldsymbol{\theta}_{drd}) = K^{-1}, \quad (38)$$

Algorithm 2 Evidence optimization using decoupled Laplace approximation

Input: $X, \mathbf{y}$
Output: latents $\hat{\mathbf{u}}$, hyperparameters $\hat{\boldsymbol{\theta}} = \{\hat{\sigma^2}, \hat{\delta}, \hat{b}, \hat{\rho}, \hat{l}\}$.
At iteration t:
1. Numerically optimize log-posterior for latents $\mathbf{m}^t_\mathbf{u}$ using (eq. 19) or Algorithm 1.
2. Compute Γ^t using negative Hessian of the log conditional evidence (eq. 37).
3. Numerically optimize $p(\mathbf{y}|X, \boldsymbol{\theta}, \mathbf{m}^t_\mathbf{u}, \Gamma^t)$ (eq. 39) for $\boldsymbol{\theta}^t$.
Repeat step 1, 2 and 3 until $\{\mathbf{m}_\mathbf{u}, \Gamma\}$ and $\boldsymbol{\theta}$ converge.

which is the inverse of the GP prior covariance governing $\mathbf{u}$ (eq. 7). Substituting for $\Lambda_\mathbf{u}$ in (eq. 23), this gives:

$$\log p(\mathbf{y}|X, \boldsymbol{\theta}) \approx \log p(\mathbf{y}|X, \mathbf{m}_\mathbf{u}, \sigma^2, \delta) + \log p(\mathbf{m}_\mathbf{u}|\boldsymbol{\theta}_{drd}) - \tfrac{1}{2}\log|\Gamma + K^{-1}| + const. \quad (39)$$

This form decomposes the curvature at the posterior mode into the likelihood curvature and the prior curvature. In this way, the posterior curvature tracks the influence of the change in the prior curvature as we optimize the hyperparameters $\boldsymbol{\theta}$, while keeping the influence of the likelihood curvature fixed. This decoupling allows us to update the posterior without recomputing the Hessian. It will be accurate so long as the Hessian of the likelihood changes slowly over local regions in parameter space.

To optimize hyperparameters under the decoupled Laplace approximation, we fix $\mathbf{m}_\mathbf{u}$ and Γ using the current mode of the posterior, and optimize (eq. 39) directly for $\boldsymbol{\theta}$, incorporating the dependence of K on $\boldsymbol{\theta}_{drd}$. With this approach, the first term, $\log p(\mathbf{y}|X, \mathbf{m}_\mathbf{u}, \sigma^2, \delta)$, captures the dependence on σ^2 and δ; the second term, $\log p(\mathbf{m}_\mathbf{u}|\boldsymbol{\theta}_{drd})$, restricts $\boldsymbol{\theta}_{drd}$ around the current mode; and the third term $-\frac{1}{2}\log|\Gamma + K^{-1}|$ pushes $\boldsymbol{\theta}_{drd}$ along the second order curvature given the GP kernel. This decoupling weakens the strong dependency between $\boldsymbol{\theta}_{drd}$ and $\mathbf{m}_\mathbf{u}$, maintaining the accuracy of the Laplace approximation as we adjust $\boldsymbol{\theta}_{drd}$.

To ensure the accuracy of the Laplace approximation, in each iteration t, we optimize eq. (39) over a restricted region of the hyperparameter space around the previous hyperparameter setting θ^{t-1}, which allows varying within 20% of its current value on each iteration in our experiments. This prevents $\boldsymbol{\theta}$ from moving too far from the region where the current Laplace approximation ($\mathbf{m}_\mathbf{u}$ and Γ) is accurate. Then, based on a new hyperparameter setting $\boldsymbol{\theta}^t$, we update the Laplace approximation parameters $\mathbf{m}_\mathbf{u}$ and Γ. This procedure is summarized in Algorithm 2. The algorithm stops when $\{\mathbf{m}_\mathbf{u}, \Gamma\}$ and $\boldsymbol{\theta}$ converge. The empirical Bayes estimate is then given by the MAP estimate of the weights $\mathbf{w}_{map} = \boldsymbol{\mu}_\mathbf{w}$ (eq. 12) conditioned on the optimal latents $\hat{\mathbf{u}} = \mathbf{m}_\mathbf{u}$ and hyperparameters $\hat{\boldsymbol{\theta}}$.

4.2. Fully Bayesian inference with MCMC

An alternate approach to the empirical Bayesian inference procedure described above is to perform fully Bayesian inference using Markov Chain Monte Carlo (MCMC). Using sampling, we can compute the integrals over $\mathbf{u}$ and $\boldsymbol{\theta}$ in order to compute the posterior mean (Bayes' least squares estimates) for $\mathbf{w}$. The full posterior distribution over $\mathbf{w}$ can be

written as

$$p(\mathbf{w}|X,\mathbf{y}) = \iint p(\mathbf{w}|X,\mathbf{y},\mathbf{u},\boldsymbol{\theta})p(\mathbf{u},\boldsymbol{\theta}|X,\mathbf{y})\,d\mathbf{u}\,d\boldsymbol{\theta} \quad (40)$$

$$= \iint \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_{\mathbf{w}},\Lambda_{\mathbf{w}})p(\mathbf{u},\boldsymbol{\theta}|X,\mathbf{y})\,d\mathbf{u}\,d\boldsymbol{\theta}, \quad (41)$$

where mean $\boldsymbol{\mu}_{\mathbf{w}}$ and covariance $\Lambda_{\mathbf{w}}$ are functions of $\mathbf{u}$ and $\boldsymbol{\theta}$ (eq. 12). This suggests a Monte Carlo representation of the posterior as

$$p(\mathbf{w}|X,\mathbf{y}) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{N}\left(\mathbf{w}\;\middle|\;\boldsymbol{\mu}_{\mathbf{w}}(\mathbf{u}^{(i)},\boldsymbol{\theta}^{(i)}),\Lambda_{\mathbf{w}}(\mathbf{u}^{(i)},\boldsymbol{\theta}^{(i)})\right) \quad (42)$$

$$\mathbf{u}^{(i)},\boldsymbol{\theta}^{(i)} \sim p(\mathbf{u},\boldsymbol{\theta}|X,\mathbf{y}), \quad (43)$$

where i is the index of the samples and N is the total number of samples. We can use Gibbs sampling to alternately sample $\mathbf{u}$ and $\boldsymbol{\theta}$ from their conditional distributions given the other. The joint posterior distribution of $\mathbf{u}$ and $\boldsymbol{\theta}$ has the following proportional relationship,

$$p(\mathbf{u},\boldsymbol{\theta}|X,\mathbf{y}) \propto p(\mathbf{y}|X,\mathbf{u},\sigma^2,\delta)p(\mathbf{u}|\boldsymbol{\theta}_{drd})\text{Prior}(\boldsymbol{\theta}), \quad (44)$$

where $p(\mathbf{y}|X,\mathbf{u},\sigma^2,\delta)$ and $p(\mathbf{u}|\boldsymbol{\theta}_{drd})$ have the likelihoods given in (eq. 20) and (eq. 21), and $\text{Prior}(\boldsymbol{\theta})$ is the prior distribution for $\boldsymbol{\theta}$.

Sampling latents $\mathbf{u}|\boldsymbol{\theta}$
The first phase of Gibbs sampling is to sample $\mathbf{u}$ from the conditional distribution of $\mathbf{u}$ given $\boldsymbol{\theta}$,

$$\mathbf{u}|\boldsymbol{\theta} \sim p(\mathbf{y}|X,\mathbf{u},\sigma^2,\delta)p(\mathbf{u}|\boldsymbol{\theta}_{drd}). \quad (45)$$

This is the product of a Gaussian process prior $p(\mathbf{u}|\boldsymbol{\theta}_{drd})$ and a likelihood function $p(\mathbf{y}|X,\mathbf{u},\boldsymbol{\theta})$ that ties the latent variables $\mathbf{u}$ to the observed data. This setting meets the requirements of elliptical slice sampling (ESS), a rejection-free MCMC (Murray et al., 2009). ESS generates random elliptical loci using the Gaussian prior and then searches along these loci to find acceptable points by evaluating the data likelihood. This method takes into account strong dependencies imposed by GP covariance on the elements of the vector $\mathbf{u}$ to facilitates faster mixing. It also requires no tuning parameters, unlike alternative samplers such as Metropolis-Hastings or Hamiltonian Monte Carlo, but performs similarly to the best possible performance of a related M-H scheme. To overcome slow mixing that can result when the prior covariance is highly elongated, we apply ESS to a whitened variable using a reparametrization trick, discussed in more details in Sec. 4.3.

Sampling hyperparameters $\boldsymbol{\theta}|\mathbf{u}$
The conditional distribution for sampling $\boldsymbol{\theta}$ given $\mathbf{u}$ is

$$\boldsymbol{\theta}|\mathbf{u} \sim p(\mathbf{y}|X,\mathbf{u},\sigma^2,\delta)p(\mathbf{u}|\boldsymbol{\theta}_{drd})\text{Prior}(\boldsymbol{\theta}), \quad (46)$$

where $\boldsymbol{\theta} = \{\sigma^2, b, \rho, l, \delta\}$ contains five individual hyperparmaters. We therefore perform slice sampling for each variable conditioned on the others. We use prior distributions of the form:

$$\log(\sigma^2) \sim \mathcal{N}(m_n,\sigma_n^2), \quad b \sim \mathcal{N}(m_b,\sigma_b^2), \quad \rho \sim \Gamma(a_\rho,b_\rho), \quad l \sim \Gamma(a_l,b_l), \quad \delta \sim \Gamma(a_\delta,b_\delta). \quad (47)$$

We put a Gaussian prior on the log of σ^2 instead of σ^2. We will provide the values for these priors in Sec. 5 on synthetic experiments. To control the number of samples, we inspect burn-in of MCMC, e.g., the training error and the change of coefficient given the averaged coefficient samples.

4.3. Whitening the GP prior using reparametrization

In both Laplace approximation and MCMC frameworks, the latent vector $\mathbf{u}$ depends on the product of the conditional evidence $p(\mathbf{y}|X,\mathbf{u},\sigma^2,\delta)$ and the GP prior $p(\mathbf{u}|\boldsymbol{\theta}_{drd})$. The GP prior (which is the primary difference between our model and standard ARD) introduces strong dependencies between $\mathbf{u}$ and GP hyperparameters, resulting in a highly elliptical joint distribution. Such distributions are often problematic for both optimization and sampling. For example, if we are trying to perform Gibbs sampling on $\mathbf{u}$ and the GP length scale hyperparameter l, and the prior is strong relative to the evidence term, the samples $\mathbf{u}|l^{(i)}$ will have smoothness strongly determined by $l^{(i)}$, and the samples $l|\mathbf{u}^{(i)}$ will in turn be strongly determined by the smoothness of the current sample $\mathbf{u}^{(i)}$. In this case, mixing will be slow, and Gibbs sampling will take a long time to explore the full posterior over different values of l.

We can overcome this difficulty with a technique known as the "reparametrization trick," which involves reparameterizing the model so that the unknown variables are independent under the prior (Murray and Adams, 2010). If we have prior $P(\mathbf{u}) = \mathcal{N}(b\mathbf{1},K)$, then $\mathbf{u}$ can be described equivalently by a deterministic transformation of a standard normal random variable $\mathbf{v}$:

$$\mathbf{v} \sim \mathcal{N}(0,I), \quad \mathbf{u} = L\mathbf{v} + b\mathbf{1}, \quad (48)$$

where $K = LL^\top$ is the Cholesky factorization of prior covariance K.

This reparametrization simplifies Laplace-approximation-based inference by allowing a change of variables in (eq. 19) so that we directly maximize $p(\mathbf{y}|X,\mathbf{v},\boldsymbol{\theta})\mathcal{N}(\mathbf{v}|0,I)$ for $\mathbf{v}$. This optimization problem has better conditioning, and eliminates the computational problem of computing $\mathbf{u}^\top K^{-1}\mathbf{u}$ in the log prior, which is replaced by a simple ridge penalty of the form $\mathbf{v}^\top\mathbf{v}$.

For sampling-based inference, the reparametrization allows us to improve mixing performance because the conditionals $\mathbf{v}|\boldsymbol{\theta}$ and $\boldsymbol{\theta}|\mathbf{v}$ exhibit much weaker dependencies than $\mathbf{u}|\boldsymbol{\theta}$ and $\boldsymbol{\theta}|\mathbf{u}$. Moreover, elliptical slice sampling for $\mathbf{v}|\boldsymbol{\theta}$ is more efficient because it involves loci on a sphere instead of a highly elongated ellipsoid.

4.4. Fourier dual form

A second trick for improving the computational performance of DRD is to perform optimization of the latent variable $\mathbf{u}$ (or $\mathbf{v}$) in the Fourier domain. When the GP prior induces a high degree of smoothness in $\mathbf{u}$, the prior covariance K becomes approximately low rank, meaning that it has a small number of non-negligible eigenvalues. Because the covariance function (eq. 7) is shift-invariant, the eigenspectrum of K has a diagonal representation

in the Fourier domain, a consequence of Bochner's theorem (Stein, 1999; Lázaro-Gredilla et al., 2010). We can exploit this representation to optimize $\tilde{\mathbf{u}}$ (the discrete Fourier transform of $\mathbf{u}$) while ignoring Fourier components above a certain high-frequency cutoff, where this cutoff depends on the length scale l. This results in a lower-dimensional optimization problem. Fourier-domain representation of the latent vector $\mathbf{u}$ also simplifies the application of the reparametrization trick described above because the Cholesky factor L is now a diagonal matrix that can be computed analytically from the spectral density of the squared-exponential prior.

To summarize the joint application of the reparametrization and Fourier dual tricks in our model, they can be understood as allowing us to draw samples $\mathbf{u} \sim \mathcal{N}(b\mathbf{1}, K)$ via the series of transformations:

$$\tilde{\mathbf{v}} \sim \mathcal{N}(0, I), \qquad \textit{whitened Fourier domain sample} \tag{49}$$

$$\tilde{\mathbf{u}} = L\tilde{\mathbf{v}} + \tilde{\mathbf{b}}, \qquad \textit{transformed Fourier domain sample} \tag{50}$$

$$\mathbf{u} = B\tilde{\mathbf{u}}, \qquad \textit{inverse Fourier transform} \tag{51}$$

where $\tilde{\mathbf{b}}$ is the discrete Fourier transform of $b\mathbf{1}$, a vector of zeros except for a single non-zero element carrying the DC component, and B is the truncated (tall skinny) discrete inverse Fourier transform matrix mapping the low-frequency Fourier components represented in $\tilde{\mathbf{u}}$ to the space domain.

Note that the smoothness on $\mathbf{u}$, which controls the spatial scale of dependent sparsity, is different from the smoothing prior used in smooth-DRD to induce smoothness in the coefficients $\mathbf{w}$, although both can benefit from sparse Fourier-domain representation in cases where the relevant length scale is large.

5. Synthetic experiments

5.1. Simulated example with smooth and sparse weights

To illustrate and give intuition for the performance of the DRD estimator, we performed simulated experiments with a vector of regression weights in a one-dimensional space. We sampled a $p = 4000$ dimensional weight vector $\mathbf{w}$ from the smooth-DRD prior (see Fig. 1), with hyperparameters GP mean $b = -8$, GP length scale $l = 100$, GP marginal variance $\rho = 36$, smoothness length scale $\delta = 50$, measurement noise variance $\sigma^2 = 5$. We then sampled $n = 500$ responses $\mathbf{y} = X\mathbf{w} + \epsilon$, where X is an $n \times p$ design matrix with entries drawn i.i.d. from a standard normal distribution, and noise $\epsilon \sim \mathcal{N}(0, 5I)$.

Fig. 3 shows an example weight vector drawn from this prior, along with estimates obtained from a variety of different estimators:

- lasso (Tibshirani, 1996), using Least Angle Regression (LARS) implemented by glmnet[1];

1. https://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html

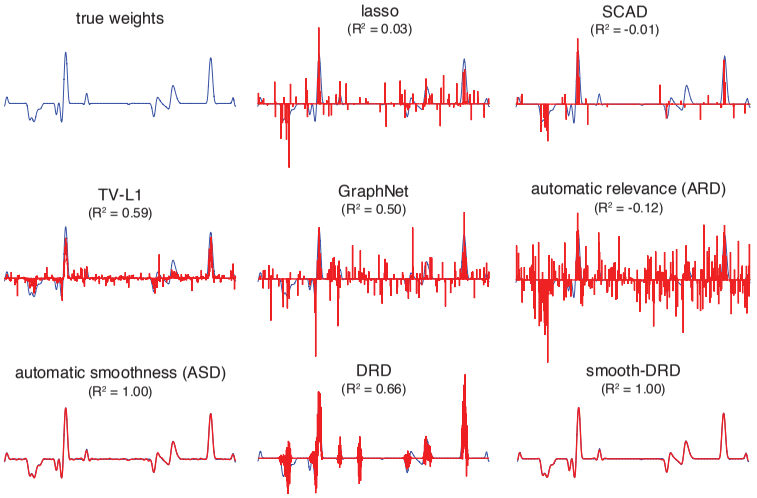


Figure 3: Example 4000-element weight vector $\mathbf{w}$ sampled from the smooth-DRD prior (upper left), and estimates obtained from different methods on a simulated dataset with $n = 500$ samples. The R^2 performance of each estimate in recovering $\mathbf{w}$ is indicated above each plot. The bottom row shows our estimators: DRD-Laplace (bottom center) and smooth-DRD-Laplace (bottom right); the other DRD and smooth-DRD estimators (not shown) achieved similar performance.

- Automatic Relevance Determination (ARD) (Neal, 1995; MacKay, 1992), implemented with the classic fixed point algorithm.
- Automatic Smoothness Determination (ASD) (Sahani and Linden, 2003), which uses numerical optimization of marginal likelihood to learn the hyperparameters of a squared exponential kernel governing $\mathbf{w}$.
- Total Variation l_1 (TV-L1) (Michel et al., 2011; Baldassarre et al., 2012; Gramfort et al., 2013), combining total variation penalty (also known as fused lasso), which imposes an l_1 penalty on the first-order differences of $\mathbf{w}$, with a standard lasso penalty.
- GraphNet (Grosenick et al., 2011), a graph-constrained elastic net, developed for spatial and temporally correlated data that yields interpretable model parameters by incorporating sparse graph priors based on model smoothness or connectivity, as well as a global sparsity inducing a prior that automatically selects important variables.
- Smoothly Clipped Absolute Deviation (SCAD) (Fan and Li, 2001), an estimator with non-convex sparsity penalty.

We computed total variation l_1 (TV-L1) and graph net (GraphNet) estimates using the Nilearn[2] package (Abraham et al., 2014). SCAD was implemented by SparseReg[3] (Zhou and Gaines, 2017). For lasso, GraphNet, TV-L1 and SCAD, we used cross-validation to set hyperparameters, whereas ARD and ASD used evidence optimization to automatically set hyperparameters. For the DRD estimators, we used evidence optimization to set hyperparameters for Laplace-approximation based estimates and used sampling to integrate over hyperparameters for MCMC-based estimates.

For the basic DRD model, which incorporates structured sparsity but not smoothness, we compared three different inference methods: (1) Laplace approximation based inference ("DRD-Laplace"); (2) Markov Chain Monte Carlo ("DRD-MCMC"); and (3) Convex relaxation based optimization ("DRD-Convex"). Lastly, for the smooth-DRD model, we used two inference methods: (4) Laplace approximation ("smooth-DRD-Laplace"); and (5) MCMC ("smooth-DRD-MCMC"). For the non-MCMC estimators, we initialized the vector of Fourier domain coefficients $\tilde{\mathbf{v}}$ (eq. 49) to values of 10^{-3} in the first iteration when learning $\mathbf{u}$. The hyper-hyperparameters in the MCMC methods (eq. 47) were set to: $m_n = -2, \sigma_n^2 = 5, m_b = -10, \sigma_b^2 = 8, a_\rho = 4, b_\rho = 5, a_l = 4, b_l = 25, a_\delta = 4, b_\delta = 25$.

Fig. 3 shows the reconstruction performance (R^2) of the true regression weight $\mathbf{w}$ for different estimators. The reconstruction performance metric for an estimate $\hat{\mathbf{w}}$ is given by $R^2 = 1 - \frac{||\mathbf{w}-\hat{\mathbf{w}}||_2^2}{||\mathbf{w}-\bar{\mathbf{w}}||_2^2}$, where $||\cdot||_2$ denotes the l_2-norm and $\bar{\mathbf{w}} = \frac{1}{p}\sum_{i=1}^{p} \mathbf{w}_i$ is the mean of vector $\mathbf{w}$. The true weight vector was sampled from the smooth DRD model. The smooth-DRD estimate achieved the best performance in terms of R^2. The ASD estimate also performed well, although the estimate was not sparse, exhibiting small wiggles where the coefficients should be zero. The standard DRD estimate recovered the support of $\mathbf{w}$ with high accuracy, but had larger error than smooth-DRD estimates due to the smoothness of the true $\mathbf{w}$. The other methods (lasso, ARD, TV-L1, GraphNet and SCAD) all had lower accuracy in recovering both the support and values of the regression weights.

To provide insight into the performance of ARD, DRD, and smooth-DRD, we plotted the inferred prior covariance of each model (Fig. 4). The DRD and smooth-DRD models were both similar to ARD in that they achieved sparsity by shrinking the prior variance of unnecessary coefficients to zero. However, unlike ARD, their inferred prior covariances both exhibited clusters of non-zero coefficients, reflecting the dependencies introduced by the latent Gaussian process. Note also that ARD and DRD covariances were both diagonal, making the weights independent given the prior variances, whereas the smooth-DRD covariance had off-diagonal structure that induced smoothness.

To quantitatively validate that our estimators succeed at identifying structured sparse and smooth structure, we performed simulated experiments using data drawn from the DRD generative model. For each experiment, we generated simulated data with $n = 500$ samples from a p-element weight vector, and varied p from 500 to 4000. We used hyperparameters GP mean $b = -8$, GP length scale $l = p/40$, GP marginal variance $\rho = 36$, smoothness length scale $\delta = p/20$, and varied measurement noise variance σ^2 between 1 and 50. The sparsity ratio for the sampled weights $\mathbf{w}$ was approximately 0.20, where we considered

2. http://nilearn.github.io/index.html
3. https://github.com/Hua-Zhou/SparseReg

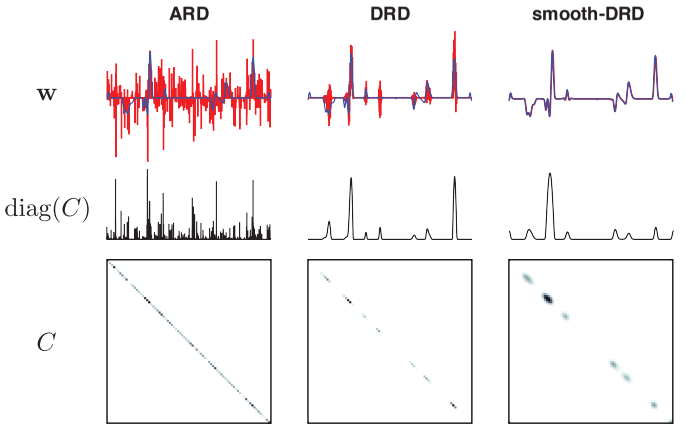


Figure 4: Estimated filter weights and prior covariances. The upper row shows the true filter (blue) and estimated ones (red); the middle row displays the diagonal of each estimated covariance matrix; and the bottom row shows the entire estimated covariance matrix for each prior.

weights with $|w_i| > 0.005$ to be non-zero. We varied training set size from $n = 100$ to 400 and kept a fixed test size of 100 samples. (We noted that even with $n = 400$ samples, the problem resides in the $n < p$ small-sample regime). We repeated each experiment 5 times.

We compared performance of DRD estimators to the above-mentioned estimators. Fig. 5 shows the reconstruction performance (R^2) of the true regression weights $\mathbf{w}$ for different estimators as a function of noise variance, training set size and dimension.

We found that Laplace and MCMC estimates for the smooth-DRD model outperformed other estimators and were approximately equally accurate, indicating that use of Laplace approximation did not noticeably harm performance relative to the fully Bayesian estimate. ASD had a good performance indicating that for these extremely smooth weights, smoothness was a more useful form of regularization than structured sparsity conferred by DRD. DRD models came next. DRD-MCMC was slightly better due to the robustness of the fully Bayesian inference. DRD-Laplace-exp and DRD-Convex employed the exponential nonlinearity when transforming $\mathbf{u}$ to the diagonal of the covariance matrix. DRD-Laplace-rec used a soft-rectifier nonlinearity which was more numerically stable. They had similar R^2 values with TV-L1 when recovering $\mathbf{w}$, but were better than all the other methods. We can also investigate the influence of each variable, i.e. noise variance, training size or dimension. When increasing the noise variance, all the R^2 values dropped; smooth-DRD-MCMC outperformed others with $\sigma^2 = 50$ indicating the power of the fully Bayesian estimate and the nontrivial effect of simultaneously imposing local sparsity and smoothness. When increasing the training size, DRD-Laplace models and DRD-Convex outperformed TV-L1

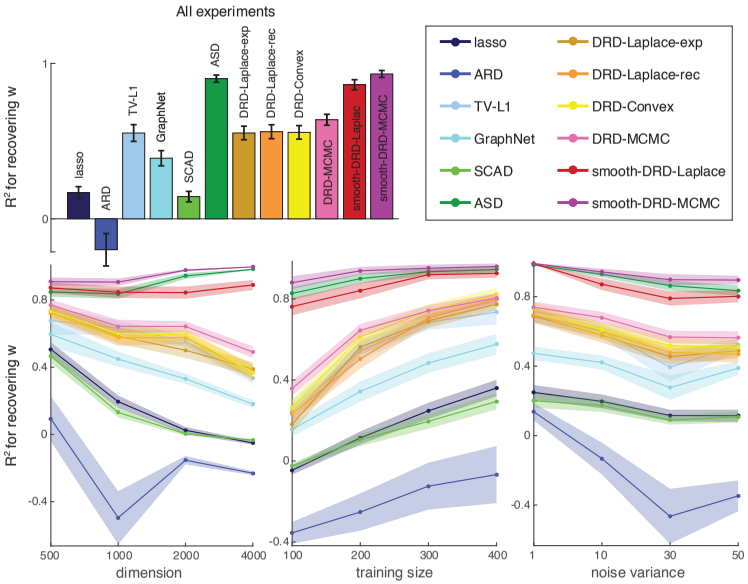


Figure 5: Comparison of performance recovering true regression weights $\mathbf{w}$ in simulated experiments as a function of dimensions p (lower left), number of samples n (lower middle), and noise variance σ^2 (lower right). Experiments were repeated five times for each of 64 combinatorial settings of four values for p, n, and σ^2. Traces show average R^2 (± 1 standard error of the mean (SEM)) as a function of each variable, and the bar plot (top row) shows average R^2 (± 1 SEM) over all $5 \times 64 = 320$ experiments.

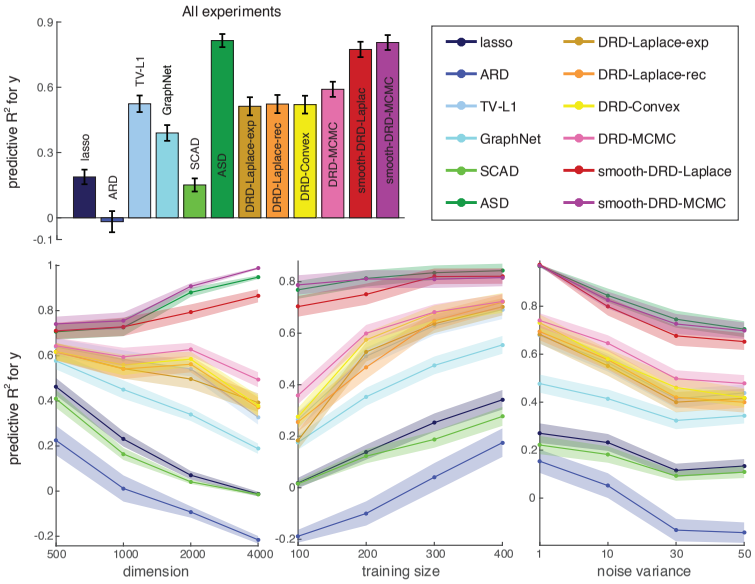


Figure 6: Comparison of performance predicting held-out responses $\mathbf{y}_{test}$ in simulated experiments as a function of dimensions p (lower left), number of samples n (lower middle), and noise variance σ^2 (lower right). Traces show average R^2 (± 1 SEM), and the bar plot (top row) shows average R^2 (± 1 SEM) over all experiments. Simulation experiments were the same as in Fig. 5.

and were comparable with DRD-MCMC, which was due to the decreasing optimizing complexity. Also surprisingly, smooth-DRD estimators achieved nearly perfect reconstruction performance over all the training sizes and all the dimensions.

Fig. 6 shows the R^2 performance for regression prediction on the test set for different estimators as a function of noise variance, training set size and dimension. The reconstruction performance for recovering the true $\mathbf{y}_{test}$ is given by $R^2 = 1 - \frac{||\mathbf{y}_{test} - \hat{\mathbf{y}}_{test}||_2^2}{||\mathbf{y}_{test} - \bar{\mathbf{y}}_{test}||_2^2}$, where $\bar{\mathbf{y}}_{test} = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \mathbf{y}_{test,i}$ is the mean of vector $\mathbf{y}_{test}$. The top-left subfigure presents the averaged R^2 values and the confidence intervals for $\hat{\mathbf{y}}_{test}$ over all runs. Similar to R^2 for $\mathbf{w}$, ASD estimate, Laplace and MCMC estimates for the smooth-DRD model outperformed other estimators.

Fig. 7 shows the AUC (Area Under the receiver operator characteristic Curve) values for different estimators as a function of noise variance, training set size and dimension. The AUC metric quantifies accuracy in recovering the binary support for $\mathbf{w}$, which is useful for assessing the effects of structured sparsity. For this metric, the smooth DRD estimators outperformed other methods, and the ASD estimator performed much worse due to its lack of sparsity. The Laplace approximation based DRD models performed slightly better than DRD-MCMC because the sparsity of MCMC estimates was diluted by averaging across multiple samples.

Overall, smooth-DRD outperformed all other methods using all metrics. This shows that combining sparsity and smoothness can provide major advantages over methods that impose only one or the other. This flexible framework for integrating structured sparsity and smoothness is one of the primary contributions of our work, in contrast to previous methods

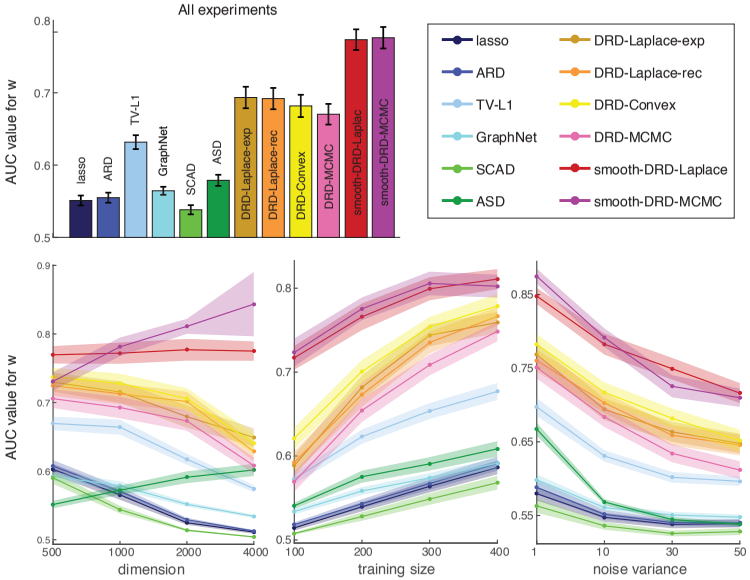


Figure 7: Comparison of performance at recovering support of regression weights in simulated experiments, quantified with area under the ROC curve (AUC), as a function of dimensions p (lower left), number of samples n (lower middle), and noise variance σ^2 (lower right). Traces show average AUC (± 1 SEM), and the bar plot (top row) shows average AUC (± 1 SEM) over all experiments. Simulation experiments were the same as in Fig. 5. Support recovery was quantified by taking all coefficients $|\mathbf{w}_i| > 0.005$ as non-zero.

in the structured sparsity literature which consider only sparsity. The code and simulated results are available online[4].

5.2. Computational complexity and optimization

We have described two basic approaches to inference for DRD: evidence optimization using the Laplace approximation and MCMC sampling-based inference. The main computational difficulty associated with Laplace-based methods is the Hessian matrix, which provides the precision matrix for the approximate Gaussian posterior distribution. This matrix costs $O(p^2)$ to store and contributes $O(p^3)$ time complexity for computation of the log-determinant. We can reduce these costs to $O(p_f^2)$ storage and $O(p_f^3)$ time, where $p_f < p$

4. https://github.com/waq1129/DRD.git

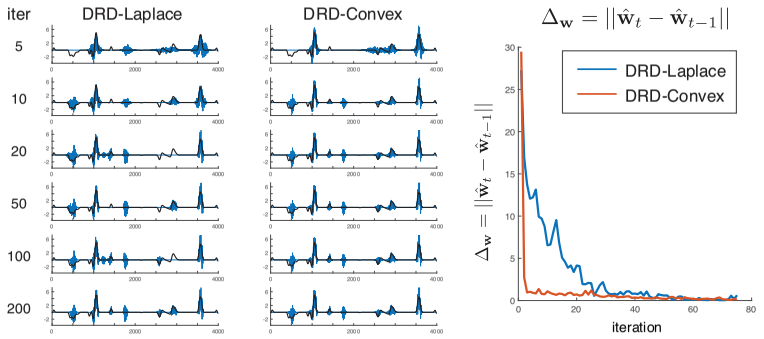


Figure 8: Comparison of the optimization of weights $\hat{\mathbf{w}}$ between DRD-Laplace and DRD-Convex. The first two columns show the weights obtained after 5, 10, 20, 50, 100, and 200 iterations with the same initialization under the two estimators, with true weights indicated in black. The third column shows the change in weights after each iteration of the standard and convex optimization algorithms over the first 80 iterations, showing that the convex algorithm made much smaller adjustments to the weights after the first few iterations and thus converged more rapidly.

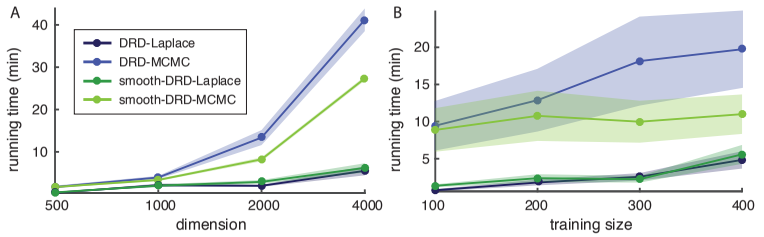


Figure 9: Running times for DRD estimators as a function of dimensions p (left) and number of samples n (right). Each point is an average across 20 simulated experiments, and the shaded area represents ± 1 SEM. For MCMC, running time was determined by the time to collect 100 posterior samples after burn-in.

is the number of non-zero Fourier coefficients in the spectral domain representation of the latent Gaussian process. This savings can be significant in problems with strong region sparsity, that is, when the zero coefficients arise in large contiguous blocks. However, in very high dimensions, the Laplace based methods may be practically infeasible due to the impossibility of storing the Hessian.

We also described a two-step convex relaxation of the Laplace method (DRD-Convex), which takes more time per iteration than the standard Laplace method (DRD-Laplace) due to the need for a two-step optimization procedure. However, we find that the DRD-Convex takes fewer iterations to converge (see Fig. 8), and in some cases proves more successful at avoiding sub-optimal local optima.

The MCMC-based inference has a time complexity of only $O(n^2 p_f)$ per sample, due to the fact that there is no need to compute the Hessian. However, MCMC-based inference is typically slower due to the need for a burn-in period and the generation of many samples from the posterior. Fig. 9 shows a comparison of running time for the two inference methods for both DRD and smooth-DRD models. For the Laplace method, we used a stopping criterion that the change in $\mathbf{w}$ was less than 0.0001. For the MCMC method, we assessed burn-in using a criterion on the relative change in $\mathbf{w}$, and then collected 100 posterior samples. Inference for the smooth-DRD model was faster than for standard DRD due to the fact that the smoothing prior effectively prunes high frequencies, reducing the dimensionality of the search space for $\mathbf{w}$. In these experiments, increasing dimension p elicited larger increases in computation time than increasing training set size n.

6. Phase transition in sparse signal recovery

Compressive sensing focuses on the recovery of sparse high-dimensional signals in settings where the number of signal coefficients p exceeds the number of measurements n. Recent work has shown that the recovery of sparse signals exhibits a phase transition between perfect and imperfect recovery as a function of the number of measurements (Ganguli and Sompolinsky, 2010; Amelunxen et al., 2014). Namely, when the measurement fraction $\gamma = n/p$ exceeds some critical value that depends on signal sparsity, the signal can be recovered perfectly with probability approaching 1, whereas for γ below this value, estimates contain errors with probability approaching 1. However, these results were derived for the case where non-zero coefficients are randomly located within the signal vector. Here we show that DRD can obtain dramatic improvements over the phase transition curve for *iid* sparse signals when the non-zero coefficients arise in clusters.

We performed simulated experiments to examine the effects of group structure on the empirical phase transition between perfect and imperfect recovery of sparse signals. The measurement equation is given by the noiseless version of the linear system we have considered so far: $\mathbf{y} = X\mathbf{w}$, where $\mathbf{w} \in \mathbb{R}^p$ is the sparse signal, $\mathbf{y} \in \mathbb{R}^n$ is the (dense) measurement vector, and here $X \in \mathbb{R}^{n \times p}$ is a (short, fat) random measurement matrix with entries drawn *iid* from a standard normal distribution. We define the sparsity of the signal as $\alpha = k/p$, where k is the number of non-zero signal coefficients in $\mathbf{w}$.

To explore the effects of group structure, we considered the signal coefficients in $\mathbf{w}$ to have 1D spatial structure and introduced a parameter g specifying the number of spatial groups or clusters into which the non-zero coefficients were divided. When $g = 1$, the non-zero coefficients formed a single contiguous block of length k, with location uniformly distributed within $\mathbf{w}$. When $g > 2$, the non-zero coefficients were divided into g blocks of size k/g, and the locations of these blocks were uniformly distributed within $\mathbf{w}$ subject to

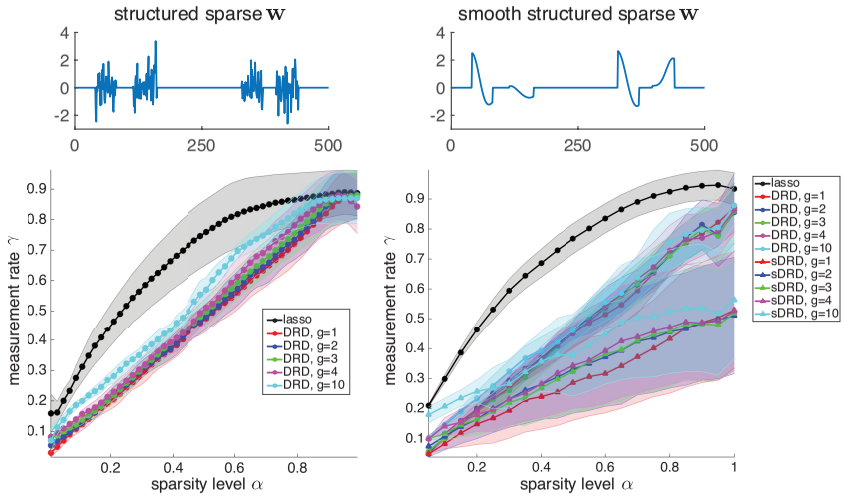


Figure 10: **Phase transitions for DRD and smooth-DRD (sDRD) estimators on signals with structured sparsity.** Top row shows example signals $\mathbf{w}$ of dimensions $p = 500$, which contain randomly positioned blocks of non-zero coefficients. Non-zero coefficients were clustered into varying numbers of groups g, and drawn either iid from a standard normal distribution (left column) or from a Gaussian with a smoothing kernel (length scale was 20) (right column), to illustrate the effects of smoothness. To compute phase transition curves, we analyzed the recovery behavior of each estimator at every point in a 2D grid of sparsity levels α and measurement rates γ. At each point, we generated 10 random signals $\mathbf{w}$, projected them noiselessly onto a random Gaussian measurement matrix X, and computed lasso and DRD estimates $\hat{\mathbf{w}}$. We then calculated the R^2 value of the estimates for each trial at every grid point (α, γ). An estimator was considered to achieve perfect recovery if all 10 trials resulted in $R^2 > 0.95$, and perfect failure if all 10 trials resulted in $R^2 \leq 0.95$; remaining points were considered to fall in the phase transition region. For each estimator, the shaded region indicates the phase transition region, and solid line indicates its center of mass along the y-axis.

the constraint that blocks remained disjoint. Once the sparsity pattern was determined, we sampled the non-zero coefficients from a standard normal distribution.

Fig. 10 shows the empirical phase transition curves for lasso and DRD estimators for sparse signals with non-zero coefficients clustered into varying numbers of groups g. These curves

show the boundary between perfect and imperfect signal recovery for different estimators in the 2D space of signal sparsity level α and measurement fraction γ. The left bottom plot shows that DRD estimators achieved perfect signal recovery for much lower measurement rates, even when non-zero coefficients were clustered into as many as 10 groups. Here DRD achieved transition to perfect recovery along the main diagonal, whereas lasso exhibited an arc-shape transition curve described previously (Ganguli and Sompolinsky, 2010; Amelunxen et al., 2014), indicating that more measurements were required to recovery signals of equal sparsity. In the right bottom plot, we generated the non-zero coefficients from a Gaussian distribution with a smoothing kernel whose length scale equaled to 20, so that non-zero coefficients were smooth as well as sparse. In these plots, we compared lasso estimates (which do not benefit from group or smooth structure) to standard DRD and smooth-DRD estimates. This reveals that smoothness allows for further reductions in measurement rates, with perfect signal recovery achievable well below that of the non-smooth DRD estimates.

7. Applications to brain imaging data

Functional magnetic resonance imaging (fMRI) measures blood oxygenation levels, which provide a proxy for neural activity in different parts of the brain. Although these measurements are noisy and indirect, fMRI is one of the primary non-invasive methods for measuring activity in human brains, and it has provided insight into the neural basis for a wide variety of cognitive abilities and functional pathologies.

A primary problem of interest in the fMRI literature is "decoding", which involves the use of linear classification and regression methods to identify the stimulus or behavior associated with measured brain activity. Decoding is a challenging statistical problem because the number of volumetric pixels or "voxels" measured with fMRI is typically far greater than the number of trials in an experiment; a full brain volume typically contains 50K voxels whereas most experiments produce only a few hundred observations.

Standard approaches to decoding have therefore tended to exploit sparsity, corresponding to the assumption that only a small set of brain voxels are relevant for decoding a particular set of stimuli (Carroll et al., 2009). However, the set of voxels useful for a specific decoding task are not randomly distributed throughout the brain, but tend to arise in clusters; if one voxel carries information useful for decoding, it is *a priori* likely that nearby voxels do too, given that voxels represent an arbitrary discretization of continuous underlying brain structures. We therefore explored brain decoding as an ideal application for evaluating the performance of our estimators.

7.1. Gambling task

We first considered the regression problem of decoding gains and losses from fMRI measurements recorded in a gambling task (Tom et al., 2007, 2011). In this experiment, event-related fMRI was administered while healthy human participants performed a decision-making task. In each trial, a gamble with a potential gain and loss (each with 50% probability) was pre-

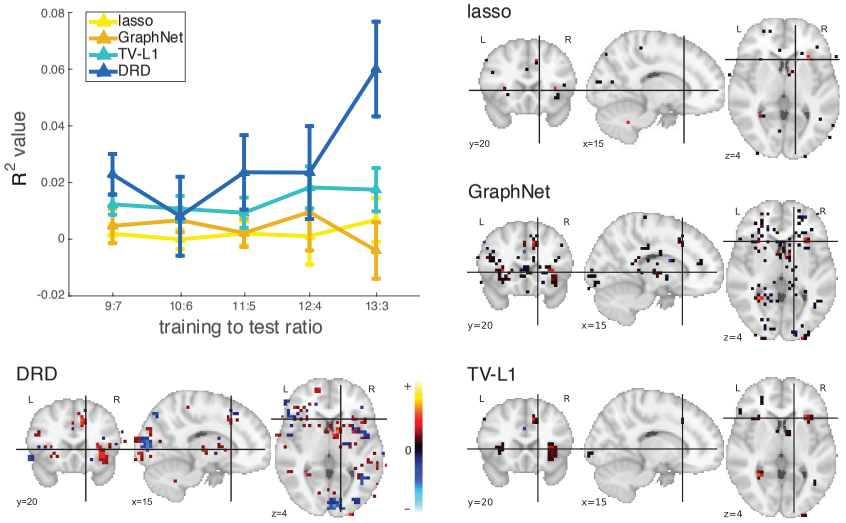


Figure 11: Top left panel shows average test R^2 values on the gambling dataset as a function of the train-test ratio for lasso, GraphNet, TV-L1 and DRD. The x-axis is the train-test split ratio and the y-axis is the R^2 criterion. The remaining panels show the estimated fMRI weight maps, overlaid on a structural fMRI image. Colors indicate the sign and magnitude of the weights (see color bar, red for positive, blue for negative, black for small). The DRD figure was obtained by cutting off small weight coefficients with a small threshold value at 0.004 (about 12% of the maximal absolute coefficient value).

sented for 3s, and the participants were instructed to decide whether to accept or reject the gamble. Experimenters varied amount of the potential gain and loss across trials. The regression task is to predict the gain of the gamble from the fMRI images recorded during the decision-making task.

After standard preprocessing, the regression dataset consisted of 16 subjects with 48 fMRI measurements per subject (resulting from 6 repeated presentations of 8 different gambles). fMRI measurements were obtained from a 3D brain volume of $40 \times 48 \times 34$ voxels, each of size $4 \times 4 \times 4$ mm, from which a subset of 33,177 valid brain voxels were used for analysis. The full dataset of 16 subjects therefore contained $n = 768$ samples in a $p = 33,177$ dimensional space.

We evaluated inter-subject prediction performance by estimating regression weights with data from a subset of the 16 subjects, and computing prediction accuracy for data from

held-out subjects. To assess the performance, we varied the train-test ratio in number of subjects from 9:7 to 13:3. We performed 10 different random train-test splits for each ratio. We used 5-fold cross-validation to set hyperparameters for all models, including DRD. For DRD models, the Laplace approximation was intractable due to the high dimensionality of the weight vector ($p = 33,177$). We therefore computed MAP estimates of the latent vector $\mathbf{u}$ conditioned on the hyperparameters, and set hyperparameters using cross-validation.

The curves in the top left panel of Fig. 11 show the performance of lasso, GraphNet, TV-L1 and DRD estimators. We found that DRD outperformed other estimators at nearly all train-test ratios, with a noticeable advantage at the largest training set size. However, we noted that the SNR of this dataset was low, making inter-subject prediction difficult and resulting in low accuracy for all methods. A non-trivial preprocessing stage, such as hyper-alignment (Chen et al., 2015), could be used to map different subjects into a shared subspace, which reduces the low SNR induced by inter-subject variability and could possibly improve performance.

Fig. 11 also shows the inferred regression weights for each estimator. The GraphNet and lasso weights had high sparsity, presumably due to the low SNR of the dataset, while TV-L1 weights exhibited small blocks of non-zero coefficients with constant value within each block, consistent with the structure expected for the TV-L1 penalty. The DRD weights were not sparse in a strict L0 sense, due to the fact that sparsity arises from soft-rectification of negative latents governing the prior variance. We therefore thresholded DRD weights for plotting purposes, revealing that the weights contributing most to prediction performance tended to cluster, as expected, although weights within each cluster were not constant. One noteworthy observation is that DRD estimate had positive (red) as well as negative weights (blue), while other estimates were largely devoid of regions with negative weights. Note that voxels in black indicate weights close to zero, which therefore contributed relatively little to readout.

7.2. Age prediction task

Next we considered the problem of predicting a subject's age from a measured map of gray-matter concentration, using data from the Open Access Series of Imaging Studies (OASIS) (Marcus et al., 2007). The OASIS dataset consisted of T1-weighted MRI scans data from 403 subjects aged 18 to 96, with 3 or 4 scans per subject. One hundred of these subjects were over 60 years of age and had been clinically diagnosed with Alzheimer's. The repeated scans provided high signal-to-noise ratio, making the dataset feasible for inter-subject analyses.

A natural regression problem for this dataset is to predict the age of subject from their anatomical MRI data. The full dataset consisted of 403 samples with a $91 \times 109 \times 91$ 3D volume and 129,081 valid voxels. To assess the performance, we varied the training ratio from 0.4 to 0.8 out of the 403 subjects, and averaged over 5 random splits for each ratio.

The curves in the top left panel of Fig. 12 show mean absolute errors between the true age and the predicted age for each estimator, evaluated on test data. The DRD and smooth-DRD estimators, which performed similarly well, achieved lower error than lasso, GraphNet, and TV-L1 estimators. The inferred regression weights (Fig. 12) reveal that the most

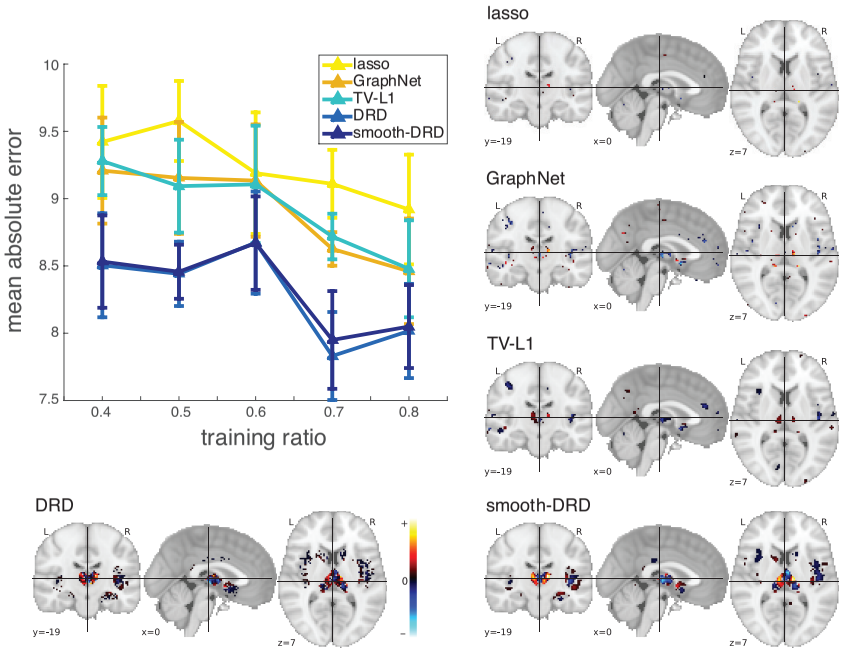


Figure 12: Top left panel shows average test mean absolute error values on the age regression dataset as a function of the training ratio for lasso, GraphNet, TV-L1, DRD and smooth-DRD. The x-axis is the ratio of the training data over the entire dataset and the y-axis is the mean absolute error criterion. The remaining panels show the estimated MRI weight maps, overlaid on a structural MRI image. Colors indicate the sign and magnitude of the weights (see color bar).

informative voxels for predicting age lie in thalamus and the basal ganglia; this is consistent with previous findings about the relationship between Alzheimer's disease progression and anatomical changes in gray matter (de Jong et al., 2008; Cho et al., 2014). Thalamus and basal ganglia were more clearly highlighted in the inferred DRD and smooth-DRD regression weights, which contained larger and more concentrated regions around these two structures. The smooth-DRD weights exhibited stronger spatial clustering than DRD weights, although regression performance was not noticeably different between the two estimators.

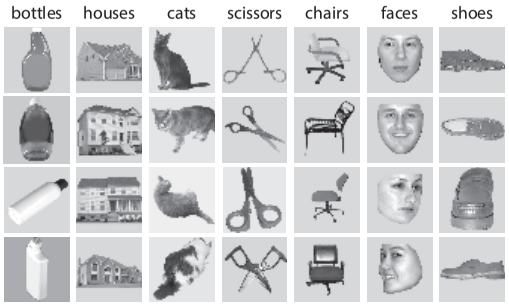


Figure 13: Examples of the stimuli for 7 categories (except for scrambled control images).

7.3. Visual recognition task

In a third application, we examined the problem of decoding faces and objects from fMRI measurements during a visual recognition task. We used a popular fMRI dataset from a study on face and object representation in human ventral temporal cortex (Haxby et al., 2001). In this experiment, 6 subjects were asked to recognize 8 different types of objects (bottles, houses, cats, scissors, chairs, faces, shoes and scrambled control images, examples in Fig. 13). Each subject participated 12 sessions of experiment. In each session, the subjects viewed images of eight object categories, with 9 full-brain measurements per category.

We assessed performance by training linear classifiers to discriminate between pairs of objects, e.g., face vs. bottle, for each of the 28 possible binary classifications among 8 objects. We trained the weights $\mathbf{w}$ for each model to linearly map fMRI measurements $\mathbf{x}$ to binary labels $y \in \{-1, +1\}$ by minimizing squared error. Note that a Bernoulli log-likelihood or logistic loss would be more appropriate for this binary classification task, but we used squared error loss because it allows for analytic marginalization over weights. We assessed decoding accuracy on test data using predicted labels $\hat{y} = \text{sign}(\mathbf{w}^\top \mathbf{x})$. We divided 12 sessions of data per subject into train-test splits of 5:7, 6:6 and 7:5. When training with data from N sessions, the training dataset consisted of 18N full-brain measurements (9 measurements per category $\times$ 2 categories). Each measurement contained 24,083 valid voxels from a $40 \times 64 \times 64$ 3D volume.

Fig. 14 shows the classification performance of lasso, GraphNet, TV-L1, DRD, and smooth-DRD estimators, averaged over 6 subjects and across the three train-test splits. The smooth-DRD estimate achieved the highest accuracy for most of the binary classifications, while the DRD estimate achieved second highest accuracy. The left column in Fig. 15 shows the regression weights estimated for the house vs. bottle classification task. The DRD and smooth-DRD weights both had significant positive regions in the parahippocampal place area (PPA), an area known to respond to images of places (Epstein et al., 1999) and negative regions in the lateral occipital complex (LOC), an area known to respond to objects (Eger

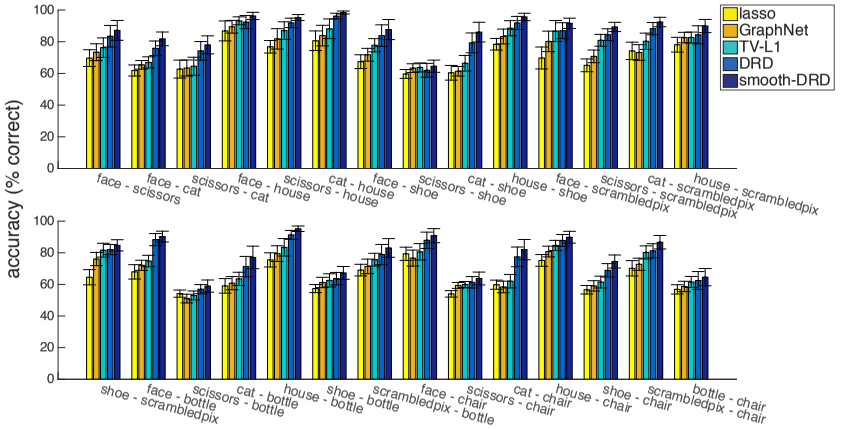


Figure 14: Classification accuracy performance for lasso, GraphNet, TV-L1, DRD and smooth-DRD, averaged across 6 subjects and three different train-test splits (5:7, 6:6 and 7:5). Error bars indicate ± 1 SEM, averaged over train-test splits. The x-axis labels indicate pairs of object categories considered for binary classification.

et al., 2008). By comparison, TV-L1 and GraphNet weights in LOC were neither strong nor clustered. The right column in Fig. 15 shows regression weights for the face vs. scrambled-pixels classification task. All methods managed to discover active responses around LOC and fusiform face area (FFA) (specialized for facial recognition) (Kanwisher et al., 1997), though DRD and smooth-DRD weights exhibited fewer isolated non-zero weights in areas far from the temporal and occipital lobes.

8. Discussion

In this paper, we introduced dependent relevance determination (DRD), a hierarchical Bayesian model for sparse, localized, and smooth regression weights. This model is appropriate for regression settings in which the regressors can be arranged geometrically as a vector, matrix, or tensor, and exhibit local dependencies within this structure (e.g., tensors of 3D brain measurements).

The DRD model takes its inspiration from the automatic relevance determination (ARD) model (Tipping, 2001), but adds a Gaussian process to introduce dependencies between prior variances of regression weights as a function of distance between their regressors. Samples from the DRD prior therefore exhibit clustering of non-zero weights. However, weights

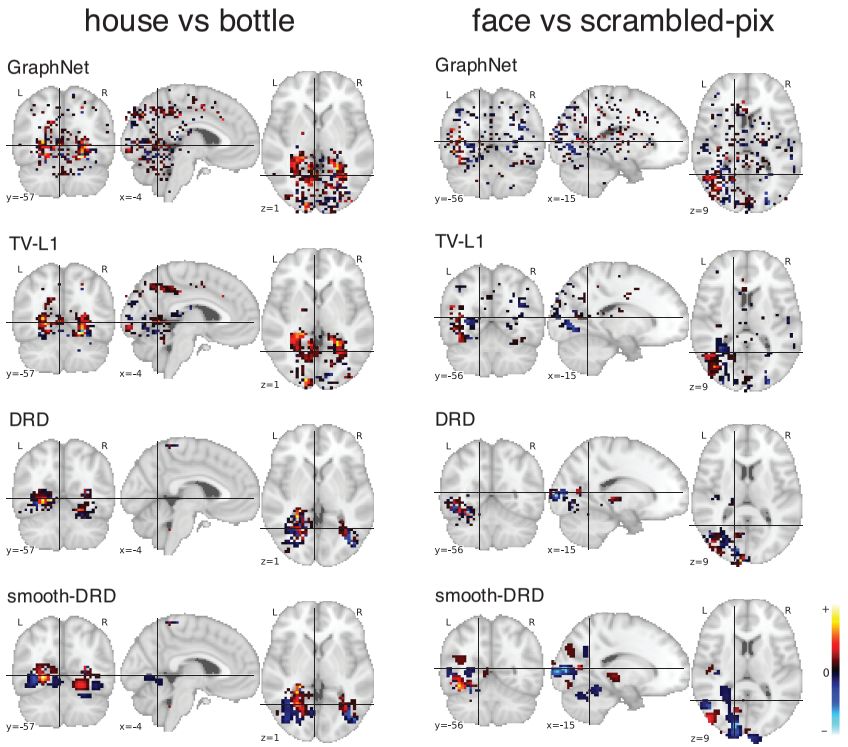


Figure 15: **Left column**: weight maps for the house vs bottle pair. **Right column**: weight maps for the face vs scrambled-pix pair. The methods shown are GraphNet, TV-L1, DRD and smooth-DRD.

sampled from the standard DRD prior are uncorrelated, meaning there is no tendency for such weight tensors to be smooth. For this reason, we introduced the smooth-DRD model, which composes the standard DRD prior with a smoothness-inducing covariance function (Sahani and Linden, 2003; Park and Pillow, 2011). Weights sampled from the resulting model tend to be sparse as well as smooth, with islands of smooth, non-zero weight features surrounded by oceans of zeros.

We described two methods for inferring the model parameters: one based on the Laplace approximation and a second based on MCMC. We proposed a novel variant of the Laplace based approach involving a two-stage convex relaxation of the log posterior.

Lastly, we carried out simulated and real application experiments to compare DRD with a variety of other methods, including lasso, SCAD, GraphNet, TV-L1, and etc. For an L_2 loss, a convex L_1 penalty leads to strong amplitude bias for lasso, while a non-convex penalty, e.g., SCAD, can overcome the strong bias. Lasso could underfit in order to get a tight support due to incorrect amplitudes of the coefficients when doing cross-validation using a L_2-based criterion. Therefore, we included SCAD apart from lasso and employed AUC as the metric for support identification. The synthetic experiments generated true weight vectors from the proposed generative model, thus aiming at validating the proposed model. We also examined phase transitions between perfect and imperfect recovery using data from a generating model different from DRD, showing that the DRD and smooth-DRD model could achieve perfect recovery with far fewer measurements when the non-zero weights in a signal were clustered. We further applied our models to three real-world brain imaging datasets. We found that DRD and smooth DRD achieved better prediction performance than previous methods, while also achieving high interpretability with regression/classification weights defined by smooth, clustered sets of voxels. Note that for the final fMRI decoding application task, a Bernoulli noise model (corresponding to a logistic loss function) would have been more appropriate than the Gaussian noise model we assumed, due to the binary nature of the outputs. Gaussian noise is a useful modeling assumption for the DRD model because it yields an analytical expression for the conditional evidence (eq. 14), which can be directly optimized for the latent process governing region sparsity. An important direction for future research will therefore be to extend DRD to incorporate Bernoulli and other likelihoods. Such extensions will need to use approximate inference methods or sampling to compute the integral over regression weights (eq. 13), but there is no conceptual barrier to incorporating region sparsity into models with binary and other non-Gaussian outputs.

The DRD and smooth-DRD models offer a powerful statistical framework for attacking problems in which sparsity is overlaid with local dependencies, a scenario that arises commonly in (for example) spatial and temporal regression problems. Recent work has shown successful application of a closely related model for capturing dependencies between sparse variables in genomic data (Engelhardt and Adams, 2014). In future work, we expect the DRD framework to find applications beyond the regression/classification setting, including structured latent factor models (Chen et al., 2015) and false discovery rate estimation (Tansey et al., 2017).

Acknowledgments

This work was supported by grants from the Sloan Foundation (JWP), McKnight Foundation (JWP), Simons Collaboration on the Global Brain (SCGB AWD1004351, JWP), and the NSF CAREER Award (IIS-1150186, JWP).

References

Alexandre Abraham, Fabian Pedregosa, Michael Eickenberg, Philippe Gervais, Andreas Mueller, Jean Kossaifi, Alexandre Gramfort, Bertrand Thirion, and Gaël Varoquaux.

Machine learning for neuroimaging with scikit-learn. *Frontiers in neuroinformatics*, 8:14, 2014.

Dennis Amelunxen, Martin Lotz, Michael B McCoy, and Joel A Tropp. Living on the edge: Phase transitions in convex programs with random data. *Information and Inference*, page iau005, 2014.

Michael R Andersen, Ole Winther, and Lars K Hansen. Bayesian inference for structured spike and slab priors. In *Advances in Neural Information Processing Systems*, pages 1745–1753, 2014.

Michael Riis Andersen, Aki Vehtari, Ole Winther, and Lars Kai Hansen. Bayesian inference for spatio-temporal spike and slab priors. *arXiv preprint arXiv:1509.04752*, 2015.

Luca Baldassarre, Janaina Mourao-Miranda, and Massimiliano Pontil. Structured sparsity models for brain decoding from fmri data. In *Pattern Recognition in NeuroImaging (PRNI), 2012 International Workshop on*, pages 5–8. IEEE, 2012.

Melissa K Carroll, Guillermo A Cecchi, Irina Rish, Rahul Garg, and A Ravishankar Rao. Prediction and interpretation of distributed neural activity with sparse models. *NeuroImage*, 44(1):112–122, 2009.

Carlos M Carvalho, Nicholas G Polson, and James G Scott. Handling sparsity via the horseshoe. In *International Conference on Artificial Intelligence and Statistics*, pages 73–80, 2009.

Po-Hsuan Cameron Chen, Janice Chen, Yaara Yeshurun, Uri Hasson, James Haxby, and Peter J Ramadge. A reduced-dimension fmri shared response model. In *Advances in Neural Information Processing Systems*, pages 460–468, 2015.

Hanna Cho, Jeong-Hun Kim, Changsoo Kim, Byoung Seok Ye, Hee Jin Kim, Cindy W Yoon, Young Noh, Geon Ha Kim, Yeo Jin Kim, Jung-Hyun Kim, et al. Shape changes of the basal ganglia and thalamus in alzheimer's disease: a three-year longitudinal study. *Journal of Alzheimer's Disease*, 40(2):285–295, 2014.

Laura W de Jong, Karin van der Hiele, Ilya M Veer, Jeanine Houwing, Rudi Westendorp, Elem Bollen, Paul W de Bruin, Huub Middelkoop, Mark A van Buchem, and Jeroen van der Grond. Strongly reduced volumes of putamen and thalamus in alzheimer's disease: an mri study. *Brain*, 131(12):3277–3285, 2008.

Evelyn Eger, John Ashburner, John-Dylan Haynes, Raymond J Dolan, and Geraint Rees. fmri activity patterns in human loc carry information about object exemplars within category. *Journal of cognitive neuroscience*, 20(2):356–370, 2008.

Barbara E Engelhardt and Ryan P Adams. Bayesian structured sparsity from gaussian fields. *arXiv preprint arXiv:1407.2235*, 2014.

Russell Epstein, Alison Harris, Damian Stanley, and Nancy Kanwisher. The parahippocampal place area: Recognition, navigation, or encoding? *Neuron*, 23(1):115–125, 1999.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

Anita C Faul and Michael E Tipping. Analysis of sparse bayesian learning. *Advances in Neural Information Processing Systems*, 14:383–389, 2002.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A note on the group lasso and a sparse group lasso. *arXiv preprint arXiv:1001.0736*, 2010.

Surya Ganguli and Haim Sompolinsky. Statistical mechanics of compressed sensing. *Physical review letters*, 104(18):188701, 2010.

Marcel V Gerven, Botond Cseke, Robert Oostenveld, and Tom Heskes. Bayesian source localization with the multivariate laplace prior. In *Advances in Neural Information Processing Systems*, pages 1901–1909, 2009.

Alexandre Gramfort, Bertrand Thirion, and Gaël Varoquaux. Identifying predictive regions from fmri with tv-l1 prior. In *Pattern Recognition in Neuroimaging (PRNI), 2013 International Workshop on*, pages 17–20. IEEE, 2013.

Logan Grosenick, Brad Klingenberg, Brian Knutson, and Jonathan E Taylor. A family of interpretable multivariate models for regression and classification of whole-brain fmri data. *arXiv preprint arXiv:1110.4139*, 2011.

Asela Gunawardana and William Byrne. Convergence theorems for generalized alternating minimization procedures. *The Journal of Machine Learning Research*, 6(Dec):2049–2073, 2005.

James V Haxby, M Ida Gobbini, Maura L Furey, Alumit Ishai, Jennifer L Schouten, and Pietro Pietrini. Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, 293(5539):2425–2430, 2001.

Daniel Hernández-Lobato and José Miguel Hernández-Lobato. Learning feature selection dependencies in multi-task learning. In *Advances in Neural Information Processing Systems*, pages 746–754, 2013.

Daniel Hernández-Lobato, José Miguel Hernández-Lobato, and Pierre Dupont. Generalized spike-and-slab priors for bayesian group feature selection using expectation propagation. *The Journal of Machine Learning Research*, 14(1):1891–1945, 2013.

Nicholas J Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.

Junzhou Huang, Tong Zhang, and Dimitris Metaxas. Learning with structured sparsity. *The Journal of Machine Learning Research*, 12:3371–3412, 2011.

Laurent Jacob, Guillaume Obozinski, and Jean-Philippe Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 433–440. ACM, 2009.

Rodolphe Jenatton, Jean-Yves Audibert, and Francis Bach. Structured variable selection with sparsity-inducing norms. *The Journal of Machine Learning Research*, 12:2777–2824, 2011.

Nancy Kanwisher, Josh McDermott, and Marvin M Chun. The fusiform face area: a module in human extrastriate cortex specialized for face perception. *The Journal of Neuroscience*, 17(11):4302–4311, 1997.

Seyoung Kim and Eric P Xing. Statistical estimation of correlated genome associations to a quantitative trait network. *PLoS genetics*, 5(8):e1000587, 2009.

Matthieu Kowalski, Kai Siedenburg, and Monika Dörfler. Social sparsity! neighborhood systems enrich structured shrinkage operators. *IEEE transactions on signal processing*, 61(10):2498–2511, 2013.

Miguel Lázaro-Gredilla, Joaquin Quiñonero-Candela, Carl Edward Rasmussen, and Aníbal R Figueiras-Vidal. Sparse spectrum gaussian process regression. *The Journal of Machine Learning Research*, 11:1865–1881, 2010.

Han Liu, Larry Wasserman, and John D Lafferty. Nonparametric regression and classification with joint sparsity constraints. In *Advances in Neural Information Processing Systems*, pages 969–976, 2009.

David JC MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Daniel S Marcus, Tracy H Wang, Jamie Parker, John G Csernansky, John C Morris, and Randy L Buckner. Open access series of imaging studies (oasis): cross-sectional mri data in young, middle aged, nondemented, and demented older adults. *Journal of cognitive neuroscience*, 19(9):1498–1507, 2007.

Vincent Michel, Alexandre Gramfort, Gaël Varoquaux, Evelyn Eger, and Bertrand Thirion. Total variation regularization for fmri-based prediction of behavior. *Medical Imaging, IEEE Transactions on*, 30(7):1328–1340, 2011.

Toby J Mitchell and John J Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

Iain Murray and Ryan P Adams. Slice sampling covariance hyperparameters of latent gaussian models. In *Advances in Neural Information Processing Systems*, pages 1732–1740, 2010.

Iain Murray, Ryan Prescott Adams, and David JC MacKay. Elliptical slice sampling. *arXiv preprint arXiv:1001.0175*, 2009.

Radford M Neal. *Bayesian learning for neural networks*. PhD thesis, University of Toronto, 1995.

Mijung Park and Jonathan W Pillow. Receptive field inference with localized priors. *PLoS computational biology*, 7(10):e1002219, 2011.

Trevor Park and George Casella. The bayesian lasso. *Journal of the American Statistical Association*, 103(482):681–686, 2008.

Maneesh Sahani and Jennifer F Linden. Evidence optimization techniques for estimating stimulus-response functions. *Advances in Neural Information Processing Systems*, pages 317–324, 2003.

Alexander Schmolck. *Smooth Relevance Vector Machines*. PhD thesis, University of Exeter, 2008.

Michael L Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer, 1999.

Wesley Tansey, Oluwasanmi Koyejo, Russell A. Poldrack, and James G. Scott. False discovery rate smoothing. *Journal of the American Statistical Association*, 0(ja):0–0, 2017. doi: 10.1080/01621459.2017.1319838.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Michael E Tipping. Sparse bayesian learning and the relevance vector machine. *The Journal of Machine Learning Research*, 1:211–244, 2001.

Michael E Tipping and Anita C Faul. Fast marginal likelihood maximisation for sparse bayesian models. In C. M. Bishop and B. J. Frey, editors, *Proceedings of the ninth international workshop on artificial intelligence and statistics*, volume 1(3), pages 1–13. Citeseer, 2003.

Sabrina M Tom, Craig R Fox, Christopher Trepel, and Russell A Poldrack. The neural basis of loss aversion in decision-making under risk. *Science*, 315(5811):515–518, 2007.

Sabrina M Tom, Craig R Fox, Christopher Trepel, and Russell A Poldrack. Mixed-gambles task, 2011. This data was obtained from the OpenfMRI database. Its accession number is ds000005.

David Wipf and Srikantan Nagarajan. A new view of automatic relevance determination. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1625–1632. MIT Press, Cambridge, MA, 2008.

Anqi Wu, Nicholas G Roy, Stephen Keeley, and Jonathan W Pillow. Gaussian process based nonlinear latent structure discovery in multivariate spike train data. In *Advances in Neural Information Processing Systems*, pages 3496–3505, 2017.

Lei Yu, Hong Sun, Jean-Pierre Barbot, and Gang Zheng. Bayesian compressive sensing for cluster structured sparse signals. *Signal Processing*, 92(1):259–269, 2012.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

Willard I Zangwill. *Nonlinear programming: a unified approach*. Prentice-Hall, 1969.

Han Zhou and Brian Gaines. Matlab sparsereg toolbox version 1.0.0. `http://hua-zhou.github.io/SparseReg/`, 2017.

Appendix A. The Hessian of the negative log-posterior in Laplace approximation for smooth-DRD

Now we derive the Hessian matrix inside of the inverse of (eq. 22).

$$H = -\frac{\partial^2}{\partial \mathbf{u} \partial \mathbf{u}^\top}\Big[\log p(\mathbf{y}|X, \mathbf{u}, \sigma^2, \delta) + \log p(\mathbf{u}|\boldsymbol{\theta}_{drd})\Big]. \tag{52}$$

The first term to take the partial derivatives of is (eq. 20):

$$\log p(\mathbf{y}|X, \mathbf{u}, \sigma^2, \delta) = -\frac{1}{2}\log|XCX^\top + \sigma^2 I| - \frac{1}{2}\mathbf{y}^\top (XCX^\top + \sigma^2 I)^{-1}\mathbf{y} + const, \tag{53}$$

and the second is (eq. 21):

$$\log p(\mathbf{u}|\boldsymbol{\theta}_{drd}) = -\frac{1}{2}(\mathbf{u} - b\mathbf{1})^\top K^{-1}(\mathbf{u} - b\mathbf{1}) - \frac{1}{2}\log|K| + const. \tag{54}$$

Define $S = XCX^\top + \sigma^2 I$, where $C = C_{drd}^{\frac{1}{2}} \Sigma C_{drd}^{\frac{1}{2}}$. Let $Z = XC_{drd}^{\frac{1}{2}}\Sigma^{\frac{1}{2}}$, then,

$$\begin{aligned} S &= XC_{drd}^{\frac{1}{2}}\Sigma C_{drd}^{\frac{1}{2}}X^\top + \sigma^2 I = ZZ^\top + \sigma^2 I, & (55)\\ \log p(\mathbf{u}|\mathbf{y}, X, \boldsymbol{\theta}) &= -\frac{1}{2}\mathbf{y}^\top S^{-1}\mathbf{y} - \frac{1}{2}\log|S| - \frac{1}{2}(\mathbf{u} - b\mathbf{1})^\top K^{-1}(\mathbf{u} - b\mathbf{1}) + const. & (56)\end{aligned}$$

The first derivative with respect to $\mathbf{u}_i$ is given by:

$$\begin{aligned} \frac{\partial}{\partial \mathbf{u}_i}\log p(\mathbf{u}|\mathbf{y}, X, \boldsymbol{\theta}) &= \frac{1}{2}\frac{\partial}{\partial \mathbf{u}_i}\left(-\mathbf{y}^\top S^{-1}\mathbf{y} - \log|S| - (\mathbf{u} - b\mathbf{1})^\top K^{-1}(\mathbf{u} - b\mathbf{1})\right) & (57)\\ &= \mathrm{Tr}\left[Z^\top S^{-1}\mathbf{y}\mathbf{y}^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_i}Z\right) - Z^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_i}Z\right)\right] & (58)\\ &\quad - \left[K^{-1}(\mathbf{u} - b\mathbf{1})\right]_i, & (59)\end{aligned}$$

where

$$\frac{\partial}{\partial \mathbf{u}_i}Z = X\left(\frac{\partial}{\partial \mathbf{u}_i}C_{drd}^{\frac{1}{2}}\right)\Sigma^{\frac{1}{2}}. \tag{60}$$

The second derivative with respect to $\mathbf{u}_j$ is given by:

$$\frac{\partial^2}{\partial \mathbf{u}_i \partial \mathbf{u}_j}\log p(\mathbf{u}|\mathbf{y}, X, \boldsymbol{\theta}) = H_1 + H_2 + H_3 - K_{ij}^{-1} = -H_{ij}, \tag{61}$$

$$H_1 = \mathrm{Tr}\left[Z^\top S^{-1}\mathbf{y}\mathbf{y}^\top S^{-1}\left(\frac{\partial^2}{\partial \mathbf{u}_i \partial \mathbf{u}_j}Z\right) - Z^\top S^{-1}\left(\frac{\partial^2}{\partial \mathbf{u}_i \partial \mathbf{u}_j}Z\right)\right], \tag{62}$$

$$H_2 = \mathrm{Tr}\left[\left(\frac{\partial}{\partial \mathbf{u}_j}Z\right)^\top S^{-1}\mathbf{y}\mathbf{y}^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_i}Z\right) - \left(\frac{\partial}{\partial \mathbf{u}_j}Z\right)^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_i}Z\right)\right], \tag{63}$$

$$H_3 = 2\mathrm{Tr}\left[-Z^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_j}Z\right)Z^\top S^{-1}\mathbf{y}\mathbf{y}^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_i}Z\right)\right. \quad (64)$$

$$-Z^\top S^{-1}\mathbf{y}\mathbf{y}^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_j}Z\right)Z^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_i}Z\right) \quad (65)$$

$$\left.+Z^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_j}Z\right)Z^\top S^{-1}\left(\frac{\partial}{\partial \mathbf{u}_i}Z\right)\right], \quad (66)$$

where

$$\frac{\partial^2}{\partial \mathbf{u}_i \partial \mathbf{u}_j}Z = X\left(\frac{\partial^2}{\partial \mathbf{u}_i \partial \mathbf{u}_j}C_{drd}^{\frac{1}{2}}\right)\Sigma^{\frac{1}{2}}. \quad (67)$$

For DRD only, we can just derive the Hessian by replacing $C_{drd}^{\frac{1}{2}}\Sigma^{\frac{1}{2}}$ with $C_{drd}^{\frac{1}{2}}$.

Appendix B. Proof of convexity of $\mathcal{L}_1(\mathbf{u})$

We ignore the scaling $\frac{1}{2}$ here for simplicity, and write,

$$\mathcal{L}_1(\mathbf{u}) = \log|X\mathrm{diag}(e^{\mathbf{u}})X^\top + \sigma^2 I| \quad (68)$$

$$= \log|\mathrm{diag}(e^{\mathbf{u}})\frac{X^\top X}{\sigma^2} + I| + const \quad (69)$$

$$= \log|\frac{X^\top X}{\sigma^2} + \mathrm{diag}(e^{-\mathbf{u}})| + \log|\mathrm{diag}(e^{\mathbf{u}})| + const. \quad (70)$$

Let $V = \frac{X^\top X}{\sigma^2}$, which is p.s.d., then

$$\begin{aligned}\frac{\partial}{\partial \mathbf{u}}\log|V + \mathrm{diag}(e^{-\mathbf{u}})| &= -\mathrm{diag}\left((\mathrm{diag}(e^{-\mathbf{u}}) + V)^{-1} \odot \mathrm{diag}(e^{-\mathbf{u}})\right)\\ \frac{\partial^2}{\partial^2 \mathbf{u}}\log|V + \mathrm{diag}(e^{-\mathbf{u}})| &= \left(V(\mathrm{diag}(e^{\mathbf{u}})V + I)^{-1}\right) \odot \left(\mathrm{diag}(e^{-\mathbf{u}}) + V\right)^{-1},\end{aligned} \quad (71)$$

where $\odot$ is the Hadamard product. Moreover, we know that

$$\begin{aligned}V(\mathrm{diag}(e^{\mathbf{u}})V + I)^{-1} &= \frac{X^\top X}{\sigma^2}(\mathrm{diag}(e^{\mathbf{u}})\frac{X^\top X}{\sigma^2} + I)^{-1}\\ &= X^\top(X\mathrm{diag}(e^{\mathbf{u}})X^\top + \sigma^2 I)^{-1}X \succeq 0.\end{aligned} \quad (72)$$

Thanks to the Schur product theorem stating that the Hadamard product of two positive semi-definite matrices is also a positive semi-definite matrix, we have $\frac{\partial^2}{\partial^2 \mathbf{u}}\log|V + \mathrm{diag}(e^{-\mathbf{u}})| \succeq 0$, thus $\log|\frac{X^\top X}{\sigma^2} + \mathrm{diag}(e^{-\mathbf{u}})|$ is convex in $\mathbf{u}$. In addition, $\log|\mathrm{diag}(e^{\mathbf{u}})|$ is also convex in $\mathbf{u}$. Therefore, $\mathcal{L}_1(\mathbf{u})$ is convex in $\mathbf{u}$.

Appendix C. Proof of boundedness and non-emptiness of $\mathcal{F}(\mathbf{u})$

We want to prove that for (eq. 35), when $||\mathbf{u}|| \to \infty$, we have $\mathcal{F}(\mathbf{u}) \to \infty$. Rewrite $\mathcal{F}$ here,

$$\mathcal{F}(\mathbf{u}) = \mathbf{z}^{k\top}\mathbf{h}(\mathbf{u}) - \mathcal{L}_{\mathbf{h}}^*(\mathbf{z}) + \mathcal{L}_1(\mathbf{u}) + \mathcal{L}_3(\mathbf{u}), \quad (73)$$

where $\mathbf{h}(\mathbf{u}) = e^{-\mathbf{u}}$, $\mathcal{L}_1(\mathbf{u}) = \frac{1}{2}\log|X\mathrm{diag}(e^{\mathbf{u}})X^\top + \sigma^2 I|$ and $\mathcal{L}_3(\mathbf{u}) = \frac{1}{2}(\mathbf{u} - b\mathbf{1})^\top K^{-1}(\mathbf{u} - b\mathbf{1})$.

1) Each element in $\mathbf{h}(\mathbf{u})$ is bounded by 0 and 1. Thus when $||\mathbf{u}|| \to \infty$, $\mathbf{z}^{k\top}\mathbf{h}(\mathbf{u})$ will be bounded.
2) K^{-1} is a positive semi-definite (p.s.d.) matrix. Thus $\mathcal{L}_3(\mathbf{u})$ is lower-bounded by 0. When $||\mathbf{u}|| \to \infty$, $\mathcal{L}_3(\mathbf{u}) \geq 0$. The upper bound is unclear.
3) Denote $\Gamma = \mathrm{diag}(e^{\mathbf{u}})$ whose diagonal values are all nonnegative. Now we want to prove when $||\mathbf{u}|| \to \infty$, we have $\mathcal{L}_1(\mathbf{u}) = \frac{1}{2}\log|X\Gamma X^\top + \sigma^2 I| \to \infty$.

First, we note that $\mathcal{L}_1(\mathbf{u})$ has a lower bound. It can be easily shown that the eigenvalues of $X\Gamma X^\top + \sigma^2 I$ should be greater than or equal to σ^2, given $X\Gamma X^\top$ is a p.s.d. matrix.

If $||\mathbf{u}|| \to \infty$, we can assume $u_1 \to \infty, ..., u_s \to \infty$ where $\mathbf{u} \in \mathbb{R}^p$ and $s \leq p$, then $e^{u_i} \to \infty$, for all $i \in \{1, ..., s\}$. For $\{u_i\}_{i=s+1}^p$, if u_i is finite, e^{u_i} will be finite; else if $u_i \to -\infty$, $e^{u_i} = 0$. Thus e^{u_i} is a finite value for all $i \in \{s+1, ..., p\}$. We can write Γ as an addition of two matrices A and B, i.e. $\Gamma = A + B$.

$$A = \begin{bmatrix} e^{u_1} & & & & & & \\ & e^{u_2} & & & & 0 & \\ & & \ddots & & & & \\ & & & e^{u_s} & & & \\ & & & & 0 & & \\ & & & & & \ddots & \\ & 0 & & & & & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & & & & & & \\ & 0 & & & & 0 & \\ & & \ddots & & & & \\ & & & 0 & & & \\ & & & & e^{u_{s+1}} & & \\ & & & & & \ddots & \\ & 0 & & & & & e^{u_p} \end{bmatrix} \quad (74)$$

The nonzero elements in A are infinite values. The nonzero elements in B are finite nonnegative values. Let $M = XBX^\top + \sigma^2 I \in \mathbb{R}^{n\times n}$. $XBX^\top$ is a p.s.d. matrix. The smallest eigenvalue of $XBX^\top$ should be nonnegative. Therefore the smallest eigenvalue of M is greater than or equal to σ^2. This implies the invertibility of M. Since M^{-1} is also positive definite, we can factorize M^{-1} into $S \in \mathbb{R}^{n\times n}$ and $S^\top$, i.e. $M^{-1} = SS^\top$. Thus, we can write

$$\mathcal{L}_1(\mathbf{u}) = \frac{1}{2}\log|XAX^\top + XBX^\top + \sigma^2 I| \quad (75)$$

$$= \frac{1}{2}\log|XAX^\top + M| \quad (76)$$

$$= \frac{1}{2}\log|XAX^\top M^{-1} + I| + \frac{1}{2}\log|M| \quad (77)$$

$$= \frac{1}{2}\log|XAX^\top SS^\top + I| + \frac{1}{2}\log|M| \quad (78)$$

$$= \frac{1}{2}\log|S^\top XAX^\top S + I| + \frac{1}{2}\log|M| \quad (79)$$

Combining $X^\top S$ to be one matrix $Z \in \mathbb{R}^{p\times n}$, we can investigate the elements in $Z^\top AZ$. $Z^\top AZ$ should be equal to $\infty * \widetilde{Z}^\top\widetilde{Z}$ where $\widetilde{Z} \in \mathbb{R}^{s\times n}$ is a trimmed Z by throwing away the rows with indices from $s+1$ to p. Therefore, nonzero eigenvalues of $\widetilde{Z}^\top\widetilde{Z}$ will turn into ∞ in $Z^\top AZ$. Zero eigenvalues will remain zero.

Let $\lambda_i \geq 0$ denote the ith eigenvalue of $Z^\top AZ$, then

$$\mathcal{L}_1(\mathbf{u}) \quad = \quad \frac{1}{2}\sum_{i=1}^{p}\log(\lambda_i + 1) + \frac{1}{2}\log|M| \tag{80}$$

Since there exists at least one eigenvalue λ_i in $Z^\top AZ$ approaching ∞, we can conclude that $\mathcal{L}_1(\mathbf{u})$ also approaches ∞ in such a case.

Accordingly, if $||\mathbf{u}|| \to \infty$, we have $\mathcal{F}(\mathbf{u}) \to \infty$, then there must exist a solution set for minimizing $\mathcal{F}(\mathbf{u})$. This validates the nonemptiness of the solution set. Furthermore, the solution set must be bounded. If it's not bounded, there must be a solution at ∞ with the minimal $\mathcal{F}(\mathbf{u})$, but this contradicts the assumption that $\mathcal{F}(\mathbf{u}) \to \infty$ when $||\mathbf{u}|| \to \infty$. Therefore, we can claim that the solution set of $\mathcal{F}(\mathbf{u})$ is bounded and nonempty.

Journal of Machine Learning Research 20 (2019) 1-56 Submitted 3/18; Revised 4/19; Published 5/19

A Particle-Based Variational Approach to Bayesian Non-negative Matrix Factorization

Muhammad A Masood masood@g.harvard.edu
Harvard John A. Paulson
School of Engineering and Applied Science
Cambridge, MA 02138, USA

Finale Doshi-Velez finale@seas.harvard.edu
Harvard John A. Paulson
School of Engineering and Applied Science
Cambridge, MA 02138, USA

Editor: Francois Caron

Abstract

Bayesian Non-negative Matrix Factorization (BNMF) is a promising approach for understanding uncertainty and structure in matrix data. However, a large volume of applied work optimizes traditional non-Bayesian NMF objectives that fail to provide a principled understanding of the non-identifiability inherent in NMF—an issue ideally addressed by a Bayesian approach. Despite their suitability, current BNMF approaches have failed to gain popularity in an applied setting; they sacrifice flexibility in modeling for tractable computation, tend to get stuck in local modes, and can require many thousands of samples for meaningful uncertainty estimates. We address these issues through a particle-based variational approach to BNMF that only requires the joint likelihood to be differentiable for computational tractability, uses a novel transfer-based initialization technique to identify multiple modes in the posterior, and thus allows domain experts to inspect a small set of factorizations that faithfully represent the posterior. On several real datasets, we obtain better particle approximations to the BNMF posterior in less time than baselines and demonstrate the significant role that multimodality plays in NMF-related tasks.

Keywords: Bayesian, Non-negative Matrix Factorization, Stein discrepancy, Non-identifiability, Transfer Learning

1. Introduction

The goal of non-negative matrix factorization (NMF) is to find a rank-R_{NMF} factorization for a non-negative data matrix X (D dimensions by N observations) into two non-negative factor matrices A and W. Typically, the rank R_{NMF} is much smaller than the dimensions and observations ($R_{\text{NMF}} \ll D, N$).

$$X \approx AW \quad | \quad X \in \mathbb{R}_+^{D\times N}, \quad A \in \mathbb{R}_+^{D\times R_{\text{NMF}}}, \quad W \in \mathbb{R}_+^{R_{\text{NMF}}\times N}$$

The linear, additive structure of these non-negative factor matrices makes NMF a popular unsupervised learning framework for discovering and interpreting latent structure in data. Each observation in the data X is approximated by an additive combination of the R_{NMF} columns of A with the combination weights given by the column of W corresponding to that observation. In this way, the basis matrix A provides a part-based representation of the data and the weights matrix W provides an R_{NMF}-dimensional latent representation of the data under this part-based representation.

The ability to easily interpret NMF solutions in this way has made them appealing in many applied areas. A few applications of NMF include understanding protein-protein interactions (Greene et al., 2008), topic modeling (Roberts et al., 2016), hyperspectral unmixing (Bioucas-Dias et al., 2012), polyphonic music transcription (Smaragdis and Brown, 2003), discovering molecular pathways from genomic samples (Brunet et al., 2004), and summarizing activations of a neural network for greater interpretability (Olah et al., 2018).

However, the analysis and interpretation of latent structure in a dataset via NMF is affected by the possibility that several non-trivially different pairs of A, W may reconstruct the data X equally well. This non-identifiability of the NMF solution space has been studied in detail in the theoretical literature (Pan and Doshi-Velez, 2016; Donoho and Stodden, 2003; Arora et al., 2012; Ge and Zou, 2015b; Bhattacharyya et al., 2016), and domain experts using NMF as a tool have noticed this issue as well. Greene et al. (2008) use ensembles of NMF solutions to model chemical interactions, while Roberts et al. (2016) conduct a detailed empirical study of multiple optima in the context of extracting topics from large corpora.

Bayesian approaches to NMF promise to characterize this parameter uncertainty in a principled manner by solving for the posterior $p(A, W|X)$ given prior $p(A, W)$ and likelihood $p(X|A, W)$ e.g. Schmidt et al. (2009); Moussaoui et al. (2006). Having a representation of uncertainty in the parameters of the factorizations can assist with the proper interpretation of the factors, allowing us to place low or high confidence on parameters of the factorization. However, computational tractability of inference limits the application of the Bayesian approach. Uncertainty estimates obtained from current Bayesian methods are often of limited use: variational approaches (e.g. Paisley et al. (2015); Hoffman and Blei (2015)) typically underestimate uncertainty and fit to a single mode; sampling-based approaches (e.g. Schmidt et al. (2009); Moussaoui et al. (2006)) also rarely switch between multiple modes and often require many thousands of samples for meaningful uncertainty estimates.

As a result of the limitations of current Bayesian approaches, domain experts tend to rely on non-Bayesian approaches to characterize uncertainty in NMF parameters. For example, Greene et al. (2008); Roberts et al. (2016); Brunet et al. (2004) all use random restarts to find multiple solutions.[1] Random restarts have no Bayesian interpretation (as they depend on the basins of attraction of each mode), but they do often find multiple optima in the objective that can be used to understand and interpret the data.

Contributions In this work, we present a transfer-learning approach that remains faithful to a principled Bayesian framework and can efficiently identify multiple, disconnected modes for any differentiable prior and likelihood model. Our transfer-learning based approach provides high-quality and diverse NMF initializations to seed a particle-based approximation to the Bayesian NMF (BNMF) posterior. We demonstrate our inference approach on two different BNMF models: first, the common exponential-Gaussian model; second, a novel

1. Random restarts involve repeating an optimization procedure with different starting points that are independently sampled.

©2019 Muhammad A Masood, Finale Doshi-Velez.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-153.html.

model that corresponds more closely with the desires of domain experts. Through our experiments, we demonstrate that:

- On a large number of real-world datasets, our particle-based posterior approximations consistently outperform baselines in terms of both posterior quality and computational running time.
- Our approach allows us to produce relatively small (less than 100 NMFs) sets of particles that belong to multiple modes of the posterior landscape, have distinct interpretations, and exhibit variability in performance on downstream tasks—all of which may be essential for a domain expert to inspect and understand the full solution space.
- Our novel practitioner-friendly BNMF model involves a new scale-fixing prior that removes many uninteresting multiple optima and captures the kinds of loss-insensitive regions that are important in many applications. Inference in this non-conjugate model is significantly more challenging than with more standard BNMF models, but our approach handles this case with ease.

2. Inference Setting

The general process of Bayesian modeling consists of three main parts. First, we must select a model (a likelihood and prior). Next, we perform inference on the model given data (under some objective). Finally, we evaluate the quality of the inference. The main innovation in this work is a novel transfer-based approach to the inference phase (Section 4). Along the way, we also introduce a novel model for BNMF that is more closely aligned to what domain experts desire from NMFs (Section 5.2).

When performing inference, we must choose how we will approximate the true posterior $p(A, W|X)$. For notational simplicity, let θ represent NMF parameters (A, W). We approximate the full BNMF posterior $p(\theta|X)$ with a discrete variational distribution $q(\theta|\theta_{1:M}, w_{1:M})$ that has M different point-masses θ_m. Each θ_m represents a different NMF solution's full set of parameters: $\theta_m = \text{vec}[A_m^T, W_m]$, and is assigned probability mass w_m. The functional form of the variational distribution is given by:

$$p(\theta|X) \approx q(\theta|\theta_{1:M}, w_{1:M}) = \sum_{m=1}^{M} w_m \delta(\theta - \theta_m) \quad \text{s.t } w_{1:M} \in \Delta^{M-1}, \quad \text{where} \quad \theta_m = \text{vec}[A_m^T, W_m] \tag{1}$$

where δ is the Dirac delta distribution and Δ^{M-1} is the probability simplex in $\mathbb{R}^M$. Particle-based approximations are attractive to domain experts because each sample represents something that they can inspect and understand.

While there exist many methods for particle-based approximations (Monte Carlo, Sequential Monte Carlo, Markov Chain Monte Carlo), these techniques often only enjoy theoretical guarantees in the limit of infinite or very large samples. Recent work in Stein discrepancy evaluation (Liu and Feng (2016); Chwialkowski et al. (2016); Gretton et al. (2006); Liu et al. (2016); Gorham and Mackey (2015); Ranganath et al. (2016); details in

Section 3) now enables us to measure the quality of an *arbitrary* finite collection as a posterior approximation.[2] As such, it opens the door to entirely new classes of particle-generation techniques, where traditional conditions, such as detailed balance, are now replaced with minimizing the Stein discrepancy $\mathbb{S}_p(q)$[3] between the true BNMF posterior $p(\theta|X)$ and the discrete approximation $q(\theta|\theta_{1:M}, w_{1:M})$:

$$q^*(\theta|\theta_{1:M}, w_{1:M}) = \text{argmin}_{q(\theta|\theta_{1:M}, w_{1:M})} \mathbb{S}_{p(\theta|X)}(q(\theta|\theta_{1:M}, w_{1:M})) \text{ s.t. } w_{1:M} \in \Delta^{M-1} \tag{2}$$

As with all variational inference problems, the problem of posterior inference is now reduced to the problem of optimizing the objective above; we are free to explore any method for producing settings $\{\theta_{1:M}, w_{1:M}\}$ to minimize the Stein discrepancy to the true posterior.

In the following, we observe that the task of minimizing this Stein discrepancy often depends on producing high-quality, diverse factorization collections $\theta_{1:M}$ and determining their associated weights $w_{1:M}$. In Section 4, we introduce a transfer-learning based approach to efficiently suggest a diverse collection of particles and optimize their associated weights. We describe traditional BNMF as well as a novel threshold-based NMF model, and discuss their merits in the context of our approach in Section 5. Experimental details including parameter choices for our approach as well as description of baselines and evaluation metrics is provided in Section 6. In Section 7, we compare our approach to more traditional particle-based approaches (MCMC), more naive ways of generating candidate particle collections, as well as directly attempting to optimize the Stein objective above. We evaluate the quality of different posterior approximations both based on their Stein discrepancies, likelihoods and reconstruction on held-out data.

3. Background

Bayesian Non-negative Matrix Factorization In BNMF, we define a prior $p(A, W)$ and a likelihood $p(X|A, W)$ and seek to characterize the posterior $p(A, W|X)$. These are related by Bayes' rule:

$$p(A, W|X) = \frac{p(X|A, W)p(A, W)}{p(X)}$$

There exist many options for the choice of prior and likelihood (e.g., exponential-Gaussian, (Paisley et al., 2015; Schmidt et al., 2009), Gamma Markov chain priors (Dikmen and Cemgil, 2009) and volume-based priors (Arngren et al., 2011)). The likelihood and prior choices are often chosen to have good computational properties (e.g. the resulting partial conjugacy of the exponential-Gaussian model). One advantage of our work is that we do not require the computationally convenient priors for inference.

Transfer learning The field of transfer learning aims to leverage models and inference applied to one problem to assist in solving related problems. It is of practical value because

2. While the popular Kullback-Leibler divergence requires comparing the ratio of probability densities or probability masses, the Stein discrepancy can be used to compare a particle-based collection defined by probability masses with a continuous target distribution.
3. This notation refers to the Stein discrepancy (a variational objective) between two distributions p and q. For a precise definition, see Section 3

there may be an abundance of data and computational resources for one problem but not another (see Pan and Yang (2010) for a survey). In this work, we shall use the solutions to BNMF from small, synthetic problems to help solve much larger NMF problems.

Stein discrepancy The Stein discrepancy $\mathbb{S}_p(q)$ is a divergence from distributions $q(\theta)$ to $p(\theta)$ that only requires sampling from the variational distribution $q(\theta)$ and evaluating the score function of the target distribution $p(\theta)$. The Stein discrepancy is computed over some class of test functions $f \in \mathcal{F}$ and satisfies the closeness property for operator variational inference (Ranganath et al., 2016): it is non-negative in general and zero only for some equivalence class of distributions $q \in \mathcal{Q}_0$. For a rich enough function class, the only distribution for which the Stein discrepancy is zero is the distribution p itself. For discrete distributions like our $q(\theta|\theta_{1:M}, w_{1:M})$ from Section 2, the approximation quality to the posterior distribution $p(\theta|X)$ of interest can be analytically computed using recent advances in Stein discrepancy evaluation with kernels (Liu and Feng, 2016; Chwialkowski et al., 2016; Gretton et al., 2006; Liu et al., 2016; Gorham and Mackey, 2015). The Stein discrepancy is related to the maximum mean discrepancy (MMD): a discrepancy that measures the worst-case deviation between expectations of functions $h \in \mathcal{H}$ under p and q (Gretton et al., 2006).

$$\text{MMD}(\mathcal{H}, q, p) := \sup_{h \in \mathcal{H}} \mathbb{E}_{\theta \sim q}[h(\theta)] - \mathbb{E}_{\theta' \sim p}[h(\theta')]$$

The Stein operator $\mathcal{T}_p$ corresponding to the distribution p is given by

$$(\mathcal{T}_p h)(x) := \frac{1}{p(x)} \langle \nabla, p(x) h(x) \rangle$$

and under its application, the function space $\mathcal{H}$ is transformed into another function space $\mathcal{T}_p(\mathcal{H}) = \mathcal{F}$. The advantage of applying this operator to the MMD equation is that expectations under p of any $f \in \mathcal{F}$ are zero, i.e. $\mathbb{E}_{\theta' \sim p} f(\theta')) = 0$ (Barbour and Brown, 1992). The Stein discrepancy is given by:

$$\mathbb{S}_p(\mathcal{H}, q) := \sup_{f \in \mathcal{T}_p(\mathcal{H})} (\mathbb{E}_{\theta \sim q} f(\theta))^2$$

Computing the Stein discrepancy is of particular interest when the distribution p is intractable. Evaluating the Stein discrepancy does not require expectations over p and the Stein operator $\mathcal{T}_p$ only depends on the unnormalized distribution via the score function $\nabla_\theta \log p(\theta)$.

In this work, we use a kernelized form of the Stein discrepancy. For every positive definite kernel $k(\theta, \theta')$, a unique Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ is defined. Chwialkowski et al. (2016) showed that the Stein operator applied to an RKHS defines a modified positive definite kernel $\mathcal{K}_p$ given by:

$$\begin{aligned} \mathcal{K}_p(\theta, \theta') = &\nabla_\theta \log p(\theta)^T \nabla_{\theta'} \log p(\theta') k(\theta, \theta') \\ &+ \nabla_{\theta'} \log p(\theta')^T \nabla_\theta k(\theta, \theta') \\ &+ \nabla_\theta \log p(\theta)^T \nabla_{\theta'} k(\theta, \theta') \\ &+ \sum_{i=1}^{d} \frac{\partial^2 k(\theta, \theta')}{\partial \theta_i \partial \theta_i'} \end{aligned} \tag{3}$$

Finally, the Stein discrepancy is simply the expectation of the modified kernel $\mathcal{K}_p$ under the joint distribution of two independent variables θ, $\theta' \sim q$.

$$\mathbb{S}_p(q) = \mathbb{E}_{\theta, \theta' \sim q}[\mathcal{K}_p(\theta, \theta')]$$

For a discrete distribution over $\theta_{1:M}$ with probability masses $w_{1:M}$ (of the form in equation 1), this can be evaluated exactly (Liu and Lee, 2016) as:

$$\begin{aligned} \mathbb{S}_p(q) &= \sum_{i,j=1}^{M} w_i w_j \mathcal{K}_p(\theta_i, \theta_j) \\ &= \mathbf{w}^T \mathbf{K} \mathbf{w} \end{aligned} \tag{4}$$

The (pure) quadratic form $\mathbf{w}^T \mathbf{K} \mathbf{w}$ is a reformulation where $\mathbf{K} \in \mathbb{R}^{M \times M}$ is the (positive definite) pairwise kernel matrix with entries $\mathbf{K}_{ij} = \mathcal{K}_p(\theta_i, \theta_j)$ and the probability masses $w_{1:M}$ are embedded into a vector $\mathbf{w} \in \mathbb{R}^{M \times 1}$. Our particle-based variational objective (equation 2) simplifies to the form in equation 4. In Section 4, we will provide a method for estimating $\theta_{1:M}$ and $w_{1:M}$ for the BNMF problem.

4. Approach

In this Section, we describe our transfer-based inference. As noted in Section 2, creating a particle-based posterior involves two distinct parts: creating a collection of candidate NMFs $\theta_{1:M}$, and then optimizing their weights $w_{1:M}$. We introduce a novel transfer-based approach that uses state-of-the-art (non-Bayesian) algorithms to efficiently generate the candidate NMFs $\theta_{1:M}$ (Section 4.1). Given $\theta_{1:M}$, we optimize the weights $w_{1:M}$ via standard convex optimization tools to minimize Stein discrepancy. (See Algorithm 1 for the full algorithm.) In Section 7, we compare our approach for generating candidate NMFs and weights to other baselines, including those that use traditional methods for particle generation (MCMC), other ways of creating candidate NMFs (and then again using a convex optimization on the weights), and gradient-based optimization of the objective.

4.1. Learning factorization parameters $\theta_{1:M}$ via Transfer Learning

A natural approach to finding the factorization parameters $\theta_{1:M}$ is to optimize for them directly via the variational objective (equation 2), however, as we shall see in Section 7, the direct approach tends to get stuck in poor local optima and is computationally expensive. Since the quality of the variational approximation is determined solely by the value of the variational objective under a given set of parameters $\theta_{1:M}$, $w_{1:M}$, we are free to employ any technique that produces a suitable collection $\theta_{1:M}$.

We observe that to minimize the Stein discrepancy, we will need solutions that are both high-quality and diverse. Random restarts have been previously used to find multiple solutions in general (Gendreau and Potvin, 2010) and for NMF in particular (Greene et al., 2008; Roberts et al., 2016; Brunet et al., 2004). These restarts can take advantage of specialized (non-Bayesian) optimization algorithms for NMF (Lee and Seung, 2001; Lin, 2007; Hsieh and Dhillon, 2011) that are widely used in applied settings to produce single factorization parameters θ_m from some initialization; there also exist algorithms to speed up

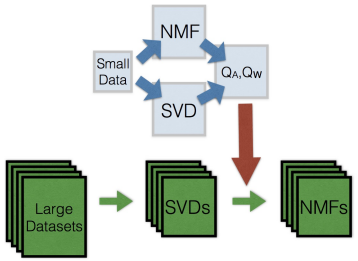


Figure 1: A schematic of the transfer learning procedure for NMF: A small dataset is used to learn transformation matrices Q_A, Q_W. We then apply these transformation matrices to multiple larger datasets (with any number of dimensions or observations) using its SVD to obtain a transfer-based initialization.

convergence of these methods (Salakhutdinov et al., 2002; Wild et al., 2004; Xue et al., 2008; Boutsidis and Gallopoulos, 2008). However, these random restarts do not take advantage of any structure of NMF; for each new NMF instance they propose random initializations from scratch. As such, many initializations may converge to the same mode—a waste of computational effort—while missing other modes (especially when the number of restarts is small).

In this Section, we introduce a transfer-based technique (which we will call Q-Transform) to speed-up, as compared to random restarts, the process of finding a diverse set of factorizations from high-density regions of the posterior. Our initializations are determined by identifying the low-rank subspace of the data (via singular value decomposition (SVD)) and then transforming it in specific ways. Figure 1 shows a schematic illustrating the idea: we generate subspace transformation matrices Q_A, Q_W from a number of small, synthetic datasets and then apply those transformations to the dataset of interest. These transformations serve as more intelligent initializations—compared to random restarts—from which to apply NMF algorithms to obtain a more diverse collection of high-quality NMFs. Because our initializations are almost always already decent NMFs, convergence is also computationally faster. To explain our Q-Transform procedure, we first define the subspace transformation matrices, then describe the method for generating transformation matrices Q_A, Q_W using synthetic data, and finally discuss how to apply them to real data sets (transfer learning).

Subspace transformations Q_A, Q_W relating SVD and NMF A low dimensional approximation for the data X can be obtained via the top R_{SVD} vectors of the SVD $A_{\text{SVD}}, W_{\text{SVD}}$. An NMF A, W of rank R_T (which may be different from R_{SVD}) also leads to an approximation of the data. The NMF factors are interpretable due to the non-negativity constraint whereas the SVD factors typically violate non-negativity. However, both approaches describe low dimensional subspaces that can be used to understand and approximate the data. These subspaces are the same when $R_{\text{SVD}} = R_T$ and the NMF is exact (i.e. $X = AW$; corresponds to Type I non-identifiability in Pan and Doshi-Velez (2016). Under

Algorithm 1 Particle-based Variational Inference for BNMF using Q-Transform

Input: Data $\{X\}$, Rank $\{R_{\text{NMF}}\}$, # Factorizations M
Step 1: Perform M repetitions of Algorithm 2 to get matrices $\{Q_A^m, Q_W^m\}_{m=1}^M$ or re-use them if previously constructed
Step 2: Apply Q-Transform (Algorithm 3) to get Initializations $\{A_0^m, W_0^m\}_{m=1}^M$
Step 3: Apply NMF algorithm to get Factorizations $\{A^m, W^m\}_{m=1}^M$
Step 4: Apply Algorithm 5 using a given BNMF model to get weights $\{w^m\}_{m=1}^M$ for approximate posterior
Output: Discrete NMF Posterior $\{w^m, A^m, W^m\}_{m=1}^M$

these conditions, there exist transformation matrices Q_A, Q_W to obtain the non-negative basis and weights exactly in terms of the singular value decomposition matrices:

$$\text{If} \quad X = A_{\text{SVD}} W_{\text{SVD}} = AW \quad \text{then}$$
$$A = A_{\text{SVD}} Q_A \quad W = Q_W W_{\text{SVD}}$$

When the data X is not an exact NMF but rather a perturbation of it (i.e. $X = AW + \epsilon$), the singular subspace of the matrix is bounded by Wedin's theorem (Wedin, 1972). We therefore still expect that there exist transformation matrices $Q_A \in \mathbb{R}^{R_{\text{SVD}} \times R_T}, Q_W \in \mathbb{R}^{R_T \times R_{\text{SVD}}}$ to yield approximations of the NMF factorizations that can be expressed in terms of the singular value decomposition matrices.

$$A_Q = A_{\text{SVD}} Q_A \approx A \quad W_Q = Q_W W_{\text{SVD}} \approx W$$

Our transfer-based strategy will involve identifying candidate matrices $A_Q \in \mathbb{R}^{D \times R_T}, W_Q \in \mathbb{R}^{R_T \times N}$ such that $A_{\text{SVD}} Q_A$ and $Q_W W_{\text{SVD}}$ are likely to be good initializations for an NMF of the data X. (Note that we assume that computing the SVD to obtain A_{SVD} and W_{SVD} from the data X is straight-forward.) We will describe the details for using these initializations below, but first we describe how we might create a collection of candidate transformation matrices Q_A, Q_W.

Generating transformations Q_A, Q_W for NMF initialization. To generate candidate transformations, we note that if we have already computed an NMF A, W for a dataset X, the appropriate transforms Q_A, Q_W can be computed by relating the SVD factors $A_{\text{SVD}}, W_{\text{SVD}}$ to A, W (e.g. via linear least squares). We propose to generate candidate transforms by using random restarts on small, synthetic datasets X_s that follow some generative model for NMF, where we can run (non-Bayesian) NMF algorithms quickly and solve for Q_A, Q_W (Algorithm 2). Multiple pairs of transformation matrices can be obtained by repeating Algorithm 2 with different random initializations to compute NMF of the synthetic data X_s, as well as by generating multiple synthetic datasets (see Section 8.1 for experiments and discussion of alternate generation procedures). Since the transformations Q_A, Q_W act on the inner dimensions (columns of A_{SVD} and rows of W_{SVD}), we emphasize that they can be applied to new datasets with any number of dimensions D and number of observations N.

Algorithm 2 Generate Q-Transform Matrices

Input: Synthetic Data $\{X_s\}$, SVD Dimension $\{R_{\text{SVD}}\}$, Transfer Dimension $\{R_T\}$
$A_{\text{SVD}}, W_{\text{SVD}} \leftarrow$ Compute top R_{SVD} SVD of X_s
$A_{\text{NMF}}, W_{\text{NMF}} \leftarrow$ Compute rank-R_T NMF of X_s using random initialization
$Q_A = \arg\min_Q \|A_{\text{NMF}} - A_{\text{SVD}}Q\|_F$ via linear least squares
$Q_W = \arg\min_Q \|W_{\text{NMF}} - QW_{\text{SVD}}\|_F$ via linear least squares
Output: Q_A, Q_W

Algorithm 3 Apply Q-Transform

Input: Real Data $\{X\}$, SVD Rank $\{R_{\text{SVD}}\}$, NMF Rank $\{R_{\text{NMF}}\}$, Transformation Matrices $\{Q_A, Q_W\}$
$A_{\text{SVD}}, W_{\text{SVD}} \leftarrow$ Compute top R_{SVD} SVD of X
$\tilde{A}_0 = A_{\text{SVD}}Q_A, \quad \tilde{W}_0 = Q_W W_{\text{SVD}}$
$A_0, W_0 \leftarrow$ Apply non-negativity and fix dimensions: Algorithm 4 $(\tilde{A}_0, \tilde{W}_0, R_{\text{NMF}})$
Output: A_0, W_0

Creating initializations for a new dataset. Given the top SVD factors of a new dataset $A_{\text{SVD}}, W_{\text{SVD}}$, we apply the Q-Transform (Algorithm 3) which multiplies SVD factors by the Q_A, Q_W matrices and adjusts entries of $A_{\text{SVD}}Q_A$ and $Q_W W_{\text{SVD}}$ to ensure non-negativity and correct dimensions using Algorithm 4 to obtain initializations A_0, W_0 that can be used as input for any standard (non-Bayesian) NMF algorithm (e.g. Cichocki and Phan (2009), Févotte and Idier (2011)). Algorithm 4 ensures that all values in the initialization A_0, W_0 are non-negative as well as provides a way to pad the initialization if the size R_T of the transforms Q_A, Q_W are smaller than the desired NMF rank R_{NMF}. The latter is an important point: in Section 8.3 we find that it is often the first few dimensions of the transformation that contain transferable information, and the rest provide little benefit. This observation also allows us to use transforms of some rank R_T on problems with a range of desired NMF ranks R_{NMF}. Finally, running the algorithm gives us a set of factorization parameters $\theta_m = \text{vec}[A_m^T, W_m]$ that we may (or may not) ultimately decide to keep in our approximation of the true posterior.

In the experiments in Section 7, we find that knowledge from these transformations Q_A, Q_W can be transferred to real datasets by re-using them to relate the top SVD factors of other datasets to high quality, approximately non-negative factorizations.[4]

4.2. Learning weights $w_{1:M}$ given parameters $\theta_{1:M}$

To infer the weights corresponding to a given factorization collection $\theta_{1:M}$, we minimize the Stein discrepancy (Algorithm 5) subject to the simplex constraint on the weights. This process involves first computing the pairwise kernel matrix[5] $\mathbf{K}$ using the kernel $\mathcal{K}_p$ in equation 3. The objective function is convex and can be solved using standard convex optimization solvers. Given point-masses $\theta_{1:M}$, this framework can be employed to infer weights

4. Code and demonstrations at https://github.com/dtak/Q-Transfer-Demo-public-/
5. As the kernel $\mathcal{K}_p$ is positive definite, $\mathbf{K}$ is also positive definite.

Algorithm 4 Initialization Adjustment

Input: Approximation matrices $\{A_Q, W_Q\}$, NMF Rank $\{R_{\text{NMF}}\}$
$\tilde{A}_0 \leftarrow$ Absolute Value(A_Q)
$\tilde{W}_0 \leftarrow$ Absolute Value(W_Q)
Transfer Rank R_T = # Columns of A_Q
if NMF Rank R_{NMF} > Transfer Rank R_T **then**
  $r = R_{\text{NMF}} - R_T$
  Pad $\tilde{A}_0, \tilde{W}_0$ with matrices $M_{D\times r}$ and $M_{N\times r}$ having small random entries so that initializations are the correct dimensions and matrices $M_{D\times r}, M_{N\times r}$ have little effect of the product $\tilde{A}_0\tilde{W}_0$.
  $A_0 \leftarrow [\tilde{A}_0, M_{D\times k}]$
  $W_0 \leftarrow [\tilde{W}_0^T, M_{N\times k}]^T$
else if NMF Rank R_{NMF} < Transfer Rank R_T **then**
  Pick the top R_{NMF} columns of $\tilde{A}_0$ and rows of $\tilde{W}_0$
  $A_0 \leftarrow \tilde{A}_0[:, 0 : R_{\text{NMF}}]$
  $W_0 \leftarrow \tilde{W}_0[0 : R_{\text{NMF}}, :]$
end if

Algorithm 5 Kernelized Stein inference for discrete approximations to posterior

Input: Particles $\theta_{1:M}$, Score $\nabla_\theta \log p(\theta)$, RKHS $\mathcal{H}$ defined by kernel k
Step 1: Compute pairwise kernel matrix $\mathbf{K_{i,j}} = \mathcal{K}_p(\theta_i, \theta_j)$ (from equation 3)
Step 2: Find probability masses that minimize the Stein discrepancy for the given point-masses: $\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathbf{w}^T\mathbf{K}\mathbf{w} \quad \text{s.t.} \quad \mathbf{w} \in \Delta^{M-1}$ via standard convex optimization.
Output: Probability masses $\mathbf{w}^*$

for discrete approximations to any posterior for which the score function $\nabla_\theta \log p(\theta)$ can be computed.

5. BNMF Models

Section 4 outlined a general procedure for producing a particle-based approximation to the BNMF posterior using transfer learning. In Section 7, we compare our approach to other particle-based approaches for BNMF. However, before going to the results, we first describe the two BNMF models (below) as well as our experimental procedure (Section 6).

The first model we shall use in our experiments is the commonly-used exponential-Gaussian model. This model is computationally convenient to use (e.g. Schmidt et al. (2009) derive a Gibbs sampler for this model) but the scale-flexible prior allows for multiple optima that are essentially the same factorization, and the Gaussian likelihood severely penalizes solutions of differing quality even when all solutions may be far from perfect reconstructions. These properties make this popular model less desirable from the perspective of a domain expert seeking to understand their data. The second model is a novel threshold-based model with a scale-fixing prior that at once removes scale ambiguities and allows for the kinds of likelihood ambiguities that practitioners expect—in particular, when the NMF is already an approximation of the data, solutions with different absolute likelihoods but whose relative

differences are small compared to the magnitude of the likelihood may be considered similar by a practitioner (Roberts et al., 2016). Our threshold-based likelihood model allows the practitioner to choose what levels of error are effectively the same for their purposes.

Before continuing, we emphasize again that our transfer-based inference approach can be applied to *any* BNMF model; in this paper we demonstrate our approach on the following two models because together they include a standard model often-used in the machine learning community and a novel model of interest to the practitioner community. Importantly, because our inference approach decouples the process of model choice, particle generation, and particle weighting, we use the same particle generation process (non-Bayesian optimization algorithms using the Frobenius objective) for both models. In Section 7, we demonstrate empirically that this particle generation process is robust enough; that is, we do not require processes tuned to each model.

5.1. Exponential-Gaussian Model for BNMF

The commonly used exponential-Gaussian BNMF model uses a Gaussian likelihood and exponential priors for the basis and weights matrices:

$$p_{\mathcal{N}}(X|A,W) = \prod_{d,n} \mathcal{N}(X_{d,n}, (AW)_{d,n}, \sigma_X^2)$$

$$p(A) = \prod_{d=1}^{D}\prod_{r=1}^{R} p(A_{d,r}), \quad A_{d,r} \sim \text{Exp}(\lambda_{d,r})$$

$$p(W) = \prod_{n=1}^{N}\prod_{r=1}^{R} p(W_{r,n}), \quad W_{r,n} \sim \text{Exp}(\lambda_{r,n})$$

As derived in Schmidt et al. (2009), the combination of exponential priors and Gaussian likelihoods results in element-wise conjugate parameter updates; in general, this model enjoys relatively straightforward inference approaches.

That said, as noted above, the exponential-Gaussian has several drawbacks from the perspective of a domain expert seeking to interpret their data via NMF. First, especially in settings where the model is misspecified (which will almost always be the case), the reconstruction error of even the best factorization may be relatively large. Even so, the Gaussian likelihood will tend to make the posterior highly peaked around the MAP solution—and exclude factorizations of only slightly worse (relative approximation) quality with respect to the overall error. However, domain experts may have found those factorizations interesting, as they have about the same relative error. Second, the exponential prior allows for some amount of uncertainty simply due to scale, which is typically uninteresting for domain experts. In the following, we introduce a model that addresses both of these shortcomings; because our transfer-based inference approach does not require conjugacy, we will be able to efficiently compute approximate posteriors for such more complex models.

5.2. Threshold-based, Scale-Fixing Model for BNMF

The procedure described in Algorithm 1 for finding a discrete approximation to the BNMF posterior does not depend on any special properties (such as conjugacy) and only requires

the joint density $p(X, W, A)$ to be differentiable in order to make inference tractable. Such flexibility is important as different applied domains use different notions of factorization quality: squared Euclidean distance is commonly used in hyperspectral unmixing (Bioucas-Dias et al., 2012), Kullback-Leibler divergence in image analysis (Lee and Seung, 2001) and Itakura-Saito divergence in music analysis (Févotte et al., 2009).

A common theme in many applied domains is that small differences in factorization quality may not be important if all factorizations have some large level of approximation error. In such cases, domain experts may be interested in all of these solutions (Roberts et al., 2016). At the same time, solutions that are different only in scale are likely uninteresting. Below, we present a novel prior and likelihood that reflect these application-specific preferences of practitioners in a Bayesian framework. In particular, our model class allows domain experts to take any application-specific notion of a high-quality factorization—conjugate or not—and put it into a Bayesian context.

Likelihood: Soft Insensitive Loss Function (SILF) over NMF objectives We define a likelihood that is maximum (and flat) in the region of high quality factorizations and decays as factorization quality decreases. To do so, we use the soft insensitive loss function (SILF) (Chu et al., 2004): a loss function defined over the real numbers $\mathbb{R}$, where the loss is negligible in some region around zero defined by the insensitivity threshold ϵ, and grows linearly outside that region (see figure 2). A quadratic term depending on the smoothness parameter β, makes the transition between the two main regions smooth. This transition region has length 2β, making smaller values of β correspond to sharper transitions between the flat and linear loss regions. We adapt the SILF from (Chu et al., 2004) to only be defined over the non-negative numbers $\mathbb{R}_+$ (as is typical with NMF objectives) and define it as:

$$\text{SILF}_{\epsilon,\beta}(y) = \begin{cases} 0 & 0 \leq y \leq (1-\beta)\epsilon \\ \frac{(y-(1-\beta))^2}{4\beta\epsilon} & (1-\beta)\epsilon \leq y \leq (1+\beta)\epsilon \\ y-\epsilon & y \geq (1+\beta)\epsilon \end{cases}$$

To form the likelihood, we apply the SILF loss to an NMF objective $f_X(A, W)$ to give:

$$P(X|W,A) = \frac{1}{Z} e^{(-C \times \text{SILF}_{\epsilon,\beta}(f_X(A,W)))} \tag{5}$$

We emphasize that the SILF-based likelihood allows the domain expert to use an NMF objective $f_X(A, W)$ that is best suited to their task and can specify a threshold under that objective for identifying high-quality factorizations. Once an NMF objective is chosen, the domain expert can easily choose appropriate parameters for the SILF-based likelihood since the parameters (insensitivity factor ϵ and smooth transition factor β) are interpretable and the likelihood can be visually inspected (as a one-dimensional function of a chosen NMF objective) to validate parameter choices.

Prior: uniform over basis and unambiguous in factorization scaling Often, domain experts do not have specific notions of what the prior over factorizations should be. However, prior distributions can have a large effect. These effects are undesirable if the prior was chosen for computational convenience rather than based on some true knowledge about the problem. Another concern is that NMF has a scaling and permutation ambiguity

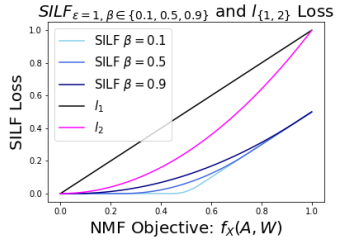


Figure 2: A comparison of SILF loss and commonly used l_1, l_2 loss functions. The SILF insensitivity parameter ϵ is set to 0.5, and the smooth transition factor β is varied. Small values of β lead to sharp transition in the SILF loss profile whereas the transition is less abrupt for large values of β. In contrast, other popular loss functions such as l_1 or l_2 do not have insensitive regions, and in the case of NMF, treat the objective function as the sole guide for factorization quality.

that is uninteresting in practice:

$$AW = \underbrace{ASP}_{\widetilde{A}}\underbrace{(SP)^{-1}W}_{\widetilde{W}} \quad \text{where S is a positive diagonal matrix, P is a permutation matrix} \tag{6}$$

Depending on the priors chosen, this ambiguity can add redundancy to the posterior distribution.

To facilitate exploration of the space of distinct high-quality factorizations, we propose an NMF prior that eliminates redundancy due to scale and is also uniform over the space of factorizations. Specifically, we let each column of the basis matrix A_r be generated by a symmetric Dirichlet distribution with parameter $\alpha = 1$. This prior determines a unique scale of the factorization and is uniform over the basis matrix A for that scaling. For W, we use a prior where each entry $W_{r,n}$ is i.i.d from an exponential distribution with parameter $\lambda_{r,n}$. The exponential distribution has support over all $\mathbb{R}_+$ ensuring that any weights matrix W corresponding to a column-stochastic basis matrix A is a valid parameter setting under our model, and that the posterior is proper.

$$p(A) = \prod_{r=1}^{R} p(A_r), \quad A_r \sim \text{Dir}(\mathbf{1}_D)$$

$$p(W) = \prod_{n=1}^{N}\prod_{r=1}^{R} p(W_{r,n}), \quad W_{r,n} \sim \text{Exp}(\lambda_{r,n})$$

6. Experimental Setup

In this Section, we provide details of our experimental settings and parameter choices, and describe our baseline algorithms and datasets. Our experiments are performed on a wide

array of benchmark NMF datasets as well as on Electronic Health Records (EHR) data of patients with Autism Spectrum Disorder (ASD) that is of interest to the medical community (see quantitative and qualitative results in Section 7).

6.1. Model, Evaluation, and Inference Settings

Model: exponential-Gaussian model parameters: We set the standard deviation σ_X to be equal to the empirical standard deviation of a reference NMF. The exponential parameter was set to one for each entry in the basis and weights matrices ($\lambda_{d,r} = \lambda_{r,n} = 1$).

Model: SILF model parameters: While any objective can be put into the SILF likelihood, in the following, we used the squared Frobenius objective $f_X(A, W) = \|X - AW\|_F^2$. To set the threshold parameter ϵ for each dataset, we use an empirical approach where we find a collection of 50 high-quality factorizations under default settings of scikit-learn (Pedregosa et al., 2011). The objective function is evaluated for each of them $\{f_i\}_{i=1}^{50}$ and $\epsilon = 1.2 \max_i f_i$. We set the remaining SILF likelihood sensitivity parameters $\beta = 0.1$, $C = 2$. For the prior, we identically set the exponential parameter for each entry: $\lambda_{r,n} = 1$.

Inference: Generating Q-transform matrices for transfer: For the Q-Transform initializations, we set the transfer rank and SVD rank $R_T = R_{\text{SVD}} = 3$. We generated twenty sets of synthetic data $X_s \in \mathbb{R}_+^{12\times 12}$ using non-negative matrices of rank R_T with truncated Gaussian noise. For each synthetic dataset, we find five pairs of transformation matrices through random restarts. In *all* our experiments, the *same* set of $M_{\max} = 100$ pairs of transformation matrices $\{Q_A^m, Q_W^m\}_{m=1}^{100}$ are applied to each of the real datasets.

Inference: Solver for inferring weights $w_{1:M}$: The optimization for the weights $w_{1:M}$ (Step 2 in Algorithm 5) is carried out using the Splitting Conic Solver (SCS) in the convex optimization package CVXPY (Diamond and Boyd, 2016).

Inference and evaluation: Stein discrepancy base RKHS and parameters: The Stein discrepancy for our variational objective requires a function space to optimize over. This optimization over the function space has an analytical solution when a Reproducing Kernel Hilbert Space (RKHS) is used. Gorham and Mackey (2017) show that the Inverse Multiquadric (IMQ) kernel is a suitable kernel choice for Stein discrepancy calculations as it detects non-convergence to posterior[6] for $c > 0$ and $b \in (-1, 0)$.

$$k_{\text{IMQ}}(\theta_i, \theta_j) = (\|\theta_i - \theta_j\|^2 + c^2)^b$$

Since the length scales of the basis and weights matrix differ, we define a kernel via a linear combination of two IMQ kernels defined separately over the basis A and weights W.

$$k([A_1, W_1], [A_2, W_2]) = \frac{1}{2\gamma_A}(\|A_1 - A_2\|^2 + c_A^2)^{b_A} + \frac{1}{2\gamma_W}(\|W_1 - W_2\|^2 + c_W^2)^{b_W} \tag{7}$$

Here $\gamma_A = (c_A^2)^{b_A}$ and similarly $\gamma_W = (c_W^2)^{b_W}$ are scaling factors that ensure the kernel takes values between 0 and 1. In general, across our datasets, the Dirichlet prior on the

6. Gorham and Mackey (2017) also prove that popular Gaussian and Matern kernels fail to detect non-convergence when the dimensionality of its inputs is greater than 3.

basis matrix induces a small length scale for A and a larger length scale for the weights W. We uniformly set $c_A = 1 \times 10^{-2}$, $c_W = 1 \times 10^3$ and $b_A = b_W = -0.5$ across all our datasets.

We note that choosing sensible values for these parameters—and validating them—is important. Kernel parameters that induce length scales that are too small or too large give rise to a similarity measure that either considers all factorizations completely dissimilar or completely similar respectively. In our experiments, our kernel choice gives rise to a similarity measure that distinguishes across collections of factorizations obtained from different algorithms. Our kernel similarity analysis shows agreement with difference between factorizations as measured by the Frobenius distances between basis and weights matrices (see figures 24, 25, 26 in Appendix). The range in kernel similarity values and its agreement with alternative measures indicates that our parameter choices for the kernel are reasonable and fairly robust.[7]

Evaluation: Measuring computational time In experiments, we keep track of the time taken (initialization and optimization) to produce each of the $M_{\max} = 100$ factorizations. We sample collections of size $M = \{5, 25, 50\}$ from these factorizations and report the total time taken to produce the factorizations in the collection alongside reporting the Stein Discrepancies for the approximate BNMF posteriors.

For the baselines below, the reported runtimes correspond to time taken to generate NMFs $\{\theta_m\}_{m=1}^M$ in the approximate posterior. For initialization approaches this corresponds to the time taken to generate the initialization and subsequent optimization time. To allow for a transparent comparison of the performance of these initialization approaches with MCMC and gradient-based algorithms, we report runtimes at various points in the duration of the MCMC chain and for the gradient-based algorithms. For more details on measuring computational time, see Appendix E in supplementary materials.

6.2. Baselines

In the previous Section, we described the implementation details for our transfer-based inference approach. In this Section, we describe implementation details for three classes of baselines for our experiments: MCMC, which represents standard practice for generating particle-based posteriors; gradient-based approaches which directly minimize the Stein variational objective, which represent our main competitors; and alternate initialization approaches, which represent simpler ablations on our approach.

Markov Chain Monte Carlo baselines MCMC approaches involve sampling from a Markov Chain whose stationary distribution is the posterior of interest, and are often considered the gold-standard for approximating posterior distributions (as opposed to variational methods). That said, for a finite sample size, MCMC will still be approximate—and thus we must still evaluate its quality with respect to the Stein objective. In this work, we consider two different MCMC baselines:

- **Hamiltonian Monte Carlo (HMC)** Our HMC was initialized with an NMF obtained using the default settings of scikit-learn (Pedregosa et al., 2011) (warm start), and adaptively selects the step-size using the procedure outlined in Neal et al. (2011).

7. Factorizations across our different datasets have different scales but the kernel parameters were fixed across all datasets.

We run the chain for a total of 10000 samples and at various intermediate points thin it to $M = \{5, 25, 50\}$ factorizations and compute the Stein discrepancy using Algorithm 5. We repeat this experiment three times to capture variability in the performance of the HMC.

For our scale-fixing prior in Section 5.2, we needed to simulate Hamiltonian dynamics as defined on the manifold of the simplex. To do this, we incorporate a reparametrization trick (Betancourt, 2012; Altmann et al., 2014) to sample under the column-stochastic (simplex) constraints of the basis matrix A, and a mirroring trick (Patterson and Teh, 2013) for sampling from the positive orthant for the weights matrix W.

- **Gibbs Sampling**. Only the exponential-Gaussian model admits a conjugate form for straight-forward Gibbs sampling. For experiments using the exponential-Gaussian, we use the same number of samples and thinning factor as with HMC for a Gibbs sampler. Similarly to the HMC baseline, the Gibbs sampler was also initialized with an NMF obtained using the default settings of scikit-learn (Pedregosa et al., 2011) (warm start).

Gradient-based baselines Gradient-based baselines optimize the collection of factorizations directly via gradient descent on the Stein variational objective. They represent the class of inference approaches most similar to ours. Gradient-based approaches typically require fixing the size of the collection. In our experiments, we set the size of this collection to be equal to $M = 5$. Due to the large memory requirement of running this algorithm with automatic differentiation using autograd (Maclaurin et al., 2015), we were unable to run these algorithms for larger M. We impose scaling and non-negativity constraints after every gradient step (for a total of 2000 steps) and keep track of the Stein discrepancy in relation to the algorithm's runtime. The experiment is repeated three times to capture variability in its performance over multiple iterations. We use the following three algorithms:

- **SVGD**: Stein Variational Gradient Descent is a functional gradient descent algorithm (Liu and Wang, 2016) that optimizes a collection of particles (factorizations) to approximate the posterior. We replace the RBF kernel from the original work with the more principled IMQ-based kernel defined in equation 7.
- **SVGD-Q** is a variant were we initialize SVGD with the Q-Transform.
- **DSGD**: Direct Stein Gradient Descent is a variant where we replace the functional gradient descent of SVGD with the gradient of the Stein discrepancy (using automatic differentiation (Baydin et al., 2015; Maclaurin et al., 2015)).

Initialization-based baselines Our Q-transform approach can be thought of as an initialization approach: we provide a way of creating a collection of particles that we believe are likely to be representative of the posterior. Our main algorithm can be run with any process for creating the collection (step 2 of Algorithm 1). Our final set of baselines considers other alternatives to creating the collection.

- **Random restarts** Our random restart initializations for NMF in scikit-learn (Pedregosa et al., 2011) set each entry of the factors A, W as independent, coming

from a truncated standard normal distribution. These entries are all scaled by $\eta = \sqrt{\frac{1}{R_{\text{NMF}}} \sum_{D,N} X_{d,n}}$ and are given by: $A^0_{d,k}, W^0_{k,n} \sim \eta|\mathcal{N}(0,1)|$.

- **NNDSVDar** NNDSVDar is a variant of a popular initialization technique called Nonnegative Double Singular Value Decomposition (NNDSVD) which was introduced by Boutsidis and Gallopoulos (2008). It is based on approximating the SVD expansion with non-negative matrices. Since the NNDSVD algorithm is deterministic, this only gives a single initialization. The NNDSVDar variant of this initialization replaces the zeros in the NNDSVD initialization with small random values. We use the scikit-learn initialization for NNDSVDar which uses a randomized SVD algorithm (Halko et al., 2011), and note that it introduces some additional variability in the initializations.

6.3. Datasets

Our datasets cover a range of different types and can be divided into three main categories (count data, grayscale face images and hyperspectral images). The ranks for hyperspectral data are chosen according to ground truth values. In the 20-Newsgroups data, we select articles from 16 newsgroups (hence the rank 16) and for other datasets we pick a rank that corresponds to explaining at least 70 percent of the variance in the data (as measured by the SVD). Table 1 provides a description of each dataset as well as the rank used and a citation. The Autism dataset is of interest to the medical community for understanding disease subtypes in the Autism spectrum and is not publicly available. The remaining datasets are public and are considered standard benchmark datasets for NMF. In our experiments, we hold out ten percent of the observations and report performance on both provided and held-out observations.

Table 1: Datasets for NMF

Dataset	Dimension	Observations	Rank	Description
20-Newsgroups	1000	8926	16	Newspaper articles (20NG, 2013)
Autism	2862	5848	20	Patient visits (Doshi-Velez et al., 2014)
LFW	1850	1288	10	Grayscale Faces Images (LFW, 2017)
Olivetti Faces	4096	400	10	Grayscale Faces Images (Samaria, 1994)
Faces CBCL	361	2429	10	Grayscale Faces Images (CBCL, 2000)
Faces BIO	6816	1514	10	Grayscale Faces Images (Jesorsky et al., 2001)
Hubble	100	2046	8	Hyperspectral Image (Nicolas Gillis, 1987)
Salinas A	204	7138	6	Hyperspectral Image (SalinasA, 2015)
Urban	162	10404	6	Hyperspectral Image (Zhu et al., 2014)

7. Results

In this Section, we compare computational time and Stein discrepancy values for variational posteriors obtained through different algorithms. For the exponential-Gaussian model, our approach using Q-Transform is either the most-competitive or second in performance to the Gibbs sampler for this model. Under the SILF model, we find that our approach for BNMF posterior approximation using transfer learning (Q-Transform) consistently produces the highest quality posterior approximations in the shortest amount of time (see Section 6.1 for details on runtime calculation). Inspection of factorization parameters from Q-Transform reveals that the parameter uncertainty captured by the BNMF posterior approximation has meaningful consequences for interpreting and utilizing these factorizations.

In the supplement, we provide an in-depth look at our results. We report on quality metrics for both the training data (figures 18, 19, 15 and 16) as well as held-out data (figures 20 and 17); we report on multiple metrics for measuring diversity of factorizations obtained from different algorithms (figures 24, 25, 26, 21, 22 and 23). Overall, these results support the notion that the Stein discrepancy is lowest for algorithms with the most diverse collection of high-quality factorizations.

7.1. Exponential-Gaussian Model Results

In figure 3, we show the performance of our algorithm and other competing baselines across our various datasets. Overall, we note that the best approximate posteriors are produced in the shortest time either by our Q-Transform algorithm or the Gibbs sampler for this model. Using random restarts for initialization yields approximate posteriors with similar Stein discrepancies to our approach but typically takes more time. The gradient-based approaches (even Q-SVGD which is initialized with Q-Transform) rarely do well, often plateauing at much higher discrepancies.

While the likelihood term in this model is invariant to (redundant) scalings[8], a limitation is that the prior (chosen for computational convenience) is dependent on the scaling. We find that this is an undesirable feature because the posterior landscape includes infinite redundant scalings and therefore requires greater effort from the inference procedure to find appropriate scalings of factorizations. Another concern is that the likelihood model is not directly expressible in terms of whatever properties might be of interest to a practitioner. To address our concerns regarding the exponential-Gaussian model, we focus for the remainder of this work on the threshold-based model with scale-fixing prior.

7.2. SILF-based Model Results

In figure 4, we show the performance of our algorithm and other competing baselines across our various datasets. Recall that the Stein discrepancy variational objective involves terms that consider both the quality of the factorizations (as given by the score function $\nabla_\theta \log p(\theta)$) and their similarity (as given by the base RKHS kernel $k(\theta_i, \theta_j)$). The NNDSVDar initializations and thinned HMC samples lead to factorizations that are high-quality but often not diverse (see diversity analysis in supplementary material: figures 24, 25, 26). The SVGD and the DSGD are generally the worst performing algorithms. These methods are often unable to find factorization parameters that meet the quality criteria of the SILF likelihood (see quality analysis in supplementary material: figures 18 and 19). This is understandable because even using simple gradient-based approaches to find a single high-quality NMF turns out to be difficult, hence the existence of a literature on specialized algorithms for performing NMF. Our Q-transform algorithm and random restarts are able to find sam-

8. Basis and weights matrices can be multiplied by any positive diagonal matrix and its inverse (respectively) to yield 'new' factors that identically reconstruct the data but differ in scale

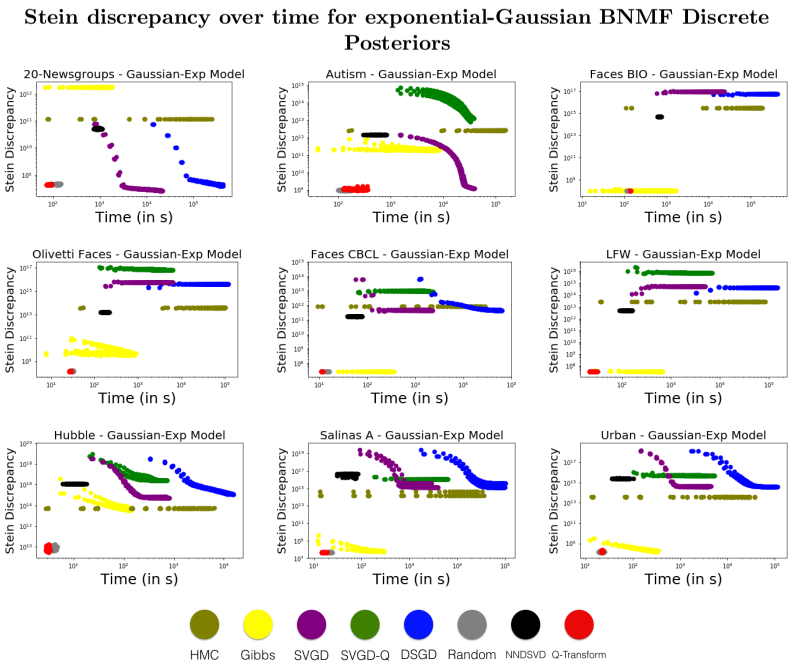


Figure 3: For each dataset we show the quality of the BNMF approximate posterior ($M = 5$) and the corresponding runtime of Q-Transform and the other baselines. Across multiple datasets, we see that the best discrete posteriors to the exponential-Gaussian BNMF (lowest Stein discrepancy) are produced either using the Q-Transform initializations (in red) for the Gibbs sampler (in yellow).

ples that are both high-quality and diverse, thus achieving the lowest Stein discrepancies; however, our Q-transform algorithm does so in the shortest time.

Figures 31 and 32 in the Appendix show results for $M = \{25, 50\}$ where Q-Transform continues to have a runtime advantage over other baselines. Additionally, for some datasets (Olivetti Faces, LFW and Faces BIO) Q-Transform also produces higher quality of the posterior approximations. Variational posteriors constructed using thinned samples from HMC significantly lack diversity as the Stein discrepancies for collections of size 5, 25 and 50 are comparable. This indicates that the HMC chain only explores a small region of the posterior distribution and can be confirmed through the diversity analysis in the Appendix (figures 24, 25, 26). Sminchisescu et al. (2007) notes that in high dimensional spaces, we expect there to be many ridges of probability as there are likely to be some directions in which the posterior density decays sharply. Alternatively, there may be several isolated modes with no connecting regions of high probability making it particularly challenging for the HMC chain to avoid getting stuck in a local mode of the BNMF posterior.

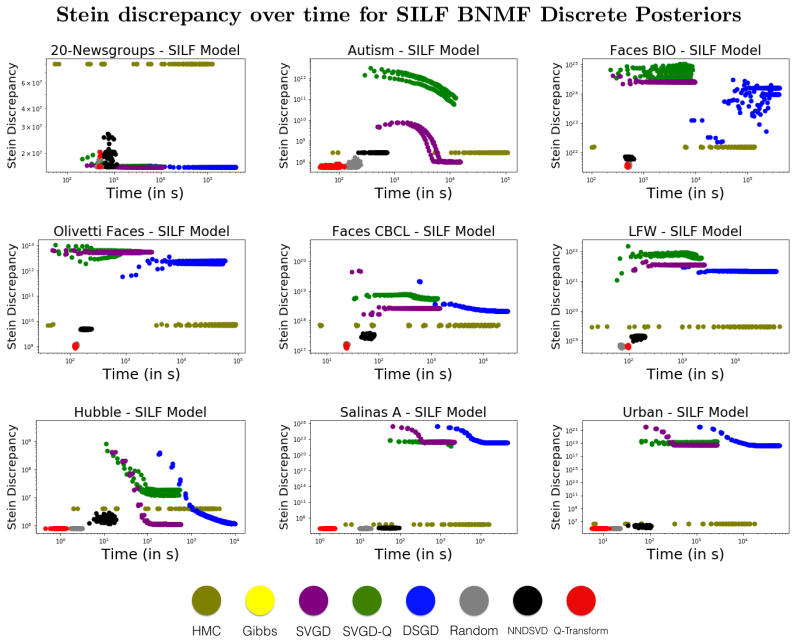


Figure 4: For each dataset we show the quality of the BNMF approximate posterior ($M = 5$) and the corresponding runtime of Q-Transform and the other baselines. Across multiple datasets, we see that the best discrete posteriors to BNMF (lowest Stein discrepancy) are produced in the least time using the Q-Transform initializations (in red).

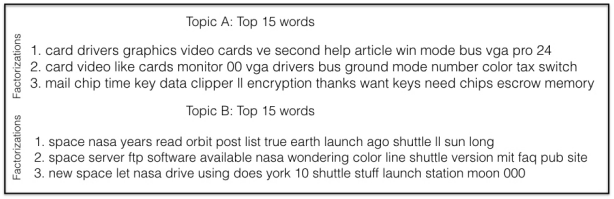


Figure 5: The top 15 words for topic A (computers/electronics) and topic B (space) shows that different factorizations provide an emphasis on different terms. In topic A, the top word from factorization 1 and 2 is 'card', but it does not appear in the top 15 words of factorizations 3. Instead a similar term 'chip' is emphasized in Factorization 3. In topic B, the terms 'space' and 'nasa' appear in all three factorizations but factorization 2 is the only one with digital terms like 'ftp', 'server','site' and 'faq'. In contrast factorization 1 and 3 both contain more physical terms like 'sun', 'moon','launch'.

7.2.1. Interpretation and Utilization of Posterior Estimates

BNMF posteriors can provide insight into the non-identifiability present within a particular dataset. Different factorizations may explain the data as a whole equally well, but do it through dictionary elements that have different interpretations, or can be used to understand specific parts of the data better than other factorizations. We show visual examples of diversity in the top words of the 20 Newsgroups BNMF posteriors and examples of how performance in downstream tasks for the 20 Newsgroups and Autism dataset is dependent on the posterior samples. Our analysis yields meaningful insights that could not be gained through a single factorization.

20-Newsgroups Our BNMF of 20-Newsgroups was a rank 16 decomposition of posts from 4 categories. In figure 6, we show the held-out AUC of a classifier trained to predict those categories based on the weights matrix W from each factorization in our variational posterior. Even though all of these factorizations have essentially equivalent reconstruction (see figure 19 in supplementary material), there exists a significant variation in the performance of these NMFs on the prediction tasks. The best performing NMF for one category is generally not the best (or even one of the top performing) NMFs for other categories. This observation may be valuable to a practitioner intending to use the NMF for some downstream task: different samples explain different patterns in the data. In figure 5, we see that this is indeed true: even after alignment,[9] distinct NMF factorizations have top words that indicate different emphasis across topics.

Autism Spectrum Disorder (ASD) In addition to core autism symptoms, Doshi-Velez et al. (2014) describe three major subtypes in autism spectrum disorder: those with higher rates of neurological disorder, those with higher rates of autoimmune disorders, and those with higher rates of psychiatric disorders. In figure 7, we show the number of topics that

9. We compare topics after finding the permutation of columns that best aligns them by solving the bipartite graph matching problem. We minimize the cost given by the angle between topics.

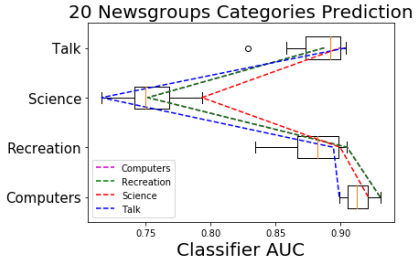


Figure 6: Classifiers trained on feature vectors from different factorizations yield variability in prediction performance (as measured by AUC). The dotted lines show the factorization that produces the best performing classifier for each category. The factorization (blue dotted line) that predicts the 'Talk' category best is actually one of the worst performing factorizations for the 'Science' category. This variability in performance demonstrates that no single factorization gives the best latent representation for the overall prediction task.

contain key terms corresponding to these areas (expressive language disorder, epilepsy, asthma, and attention deficit disorder) across different factorizations in the variational posterior obtained via Q-Transform. The large variation suggests that different factorizations in the particle-based posterior are spending different amount of modeling effort across these known factors; knowing that such uncertainty exists is essential for clinicians who may be trying to interpret topics to understand patterns in autism spectrum disorder.

On the same set of patients, we can also ask whether we can predict the onset of certain medical issues in the subsequent patient trajectory. We train a classifier on the weights of the NMFs to predict the onset of these medical issues. Similar to the category prediction results in 20-Newsgroups, figure 8 shows that there is a large variability (around 0.1 in AUC) in the performance of classifiers trained on the weights matrices of different factorizations on the prediction task. No single factorization has the best performance across the different prediction tasks.

7.3. Extension: BNMF in the presence of missing data

In the presence of missing data, there is perhaps an even greater need to understand the uncertainty in factorization parameters for NMF. The factorization space of a fully observed dataset forms a subset of the factorization space in the presence of missing data. Our particle-based approach to BNMF posterior approximation can be applied to the missing data setting by making some minor adjustments to the experimental settings.

The multiplicative update algorithm for NMF (Lee and Seung, 2001) can be adjusted so that the update equations for factorization parameters only consider the observed data. We use an implementation of this modification to the multiplicative update algorithm[10] to find a completion of the data X, compute the SVD subspace and then apply our Q-Transform initializations. Figure 9 demonstrates that our approach to BNMF can be extended to the

10. https://github.com/scikit-learn/scikit-learn/pull/8474/commits/a838f94c8c832aaf57140f23bd8c8a14daec2626

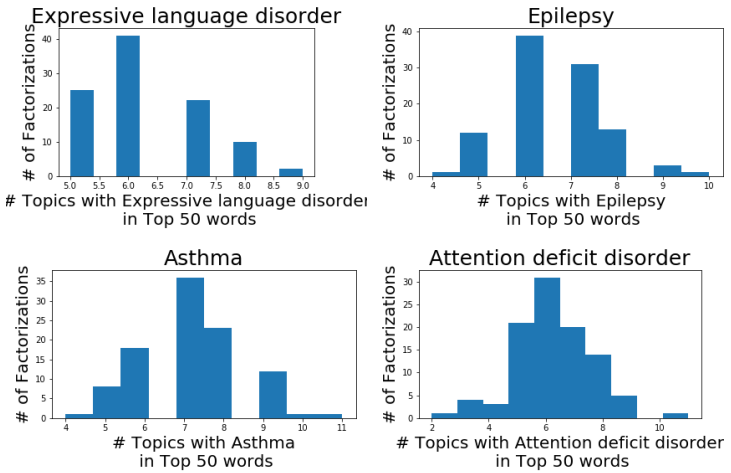


Figure 7: We explore top words in the topics relating to key terms of interest to clinicians and discover that different NMFs place varying amount of emphasis on different terms. Such variability is of interest to clinicians who may be trying to interpret topics to understand patterns in ASD.

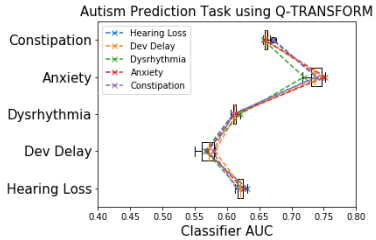


Figure 8: Classifiers trained on weights matrix of $M_{\max} = 100$ different factorizations to predict the presence of certain medical codes in a patient's trajectory exhibit significant variability in prediction on a test set (as measured by AUC). Different factorizations lead to top predictors for the onset of different medical issues.

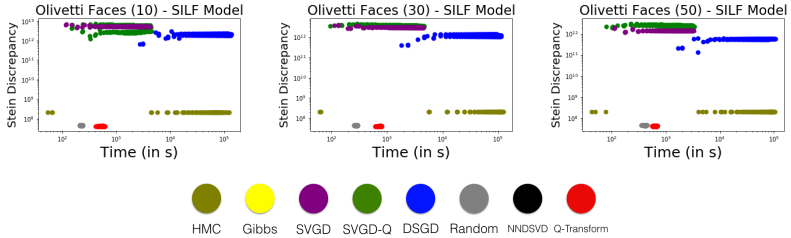


Figure 9: Under different percentages of missingness in the Olivetti Faces dataset (10%, 30%, 50%), the quality of the BNMF approximate posterior and the corresponding runtime of Q-Transform and the other baselines is shown. The best discrete posterior approximations to BNMF are produced using the Q-Transform initializations (in red).

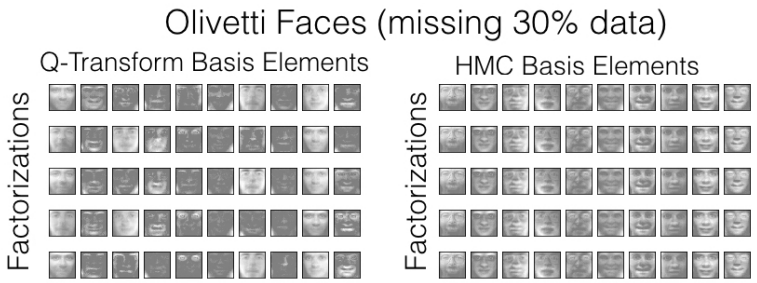


Figure 10: Sample factorizations from the variational posterior using Q-Transforms show that a diverse range of basis elements can be use to approximate the data. However, HMC samples seem to be identical indicating that HMC was only exploring a very small region of the posterior space.

case where the data matrix X is partially observed. For the Olivetti Faces dataset with varying degrees of missingness, the Q-Transform approach to BNMF consistently finds posterior approximations that are significantly better (as measured by Stein discrepancy) than other baselines whereas for a given M, the runtime is second-lowest.

Figure 10 shows sample factorizations from the variational posterior using Q-Transform and HMC samples. To allow for comparison, we have aligned the positions of the basis (dictionary) elements to a reference factorization using the bipartite matching algorithm. It is clear from looking at the Q-Transform factorizations that a diverse range of dictionaries can be used to approximate the data well whereas the HMC chain only explores one set of dictionary elements. Interestingly, the diversity of solutions obtained using Q-Transform have visually interpretable differences, i.e. these are not simply perturbations of some ground truth basis elements. Some of the basis elements look like faces and some of them look like different shadow or lighting configurations. In contrast, the factorization samples

from HMC have basis elements that look identical. This indicates that the HMC has explored a limited region of the posterior space.

8. Discussion: When is Q-Transform successful?

Our ability to extract *transferable* low-rank transformation matrices from an SVD and an instance of NMF indicates that there exist similarities across different NMF problems. In this Section we seek to develop a better intuition behind the success of the Q-Transform initializations at exploiting these similarities. In this Section, we provide discussion and smaller-scale experiments to shed light on when, why, and how our Q-transform approach is successful.

8.1. Q-Transform Generating Process

In our approach, we generated candidate Q-Transform matrices (Algorithm 2) by applying random restarts to small, synthetic data sets. We focused on this approach because small datasets are much faster to train, and with synthetic data sets, we can know at least one ground truth NMF and level of noise. However, there are obviously a large number of choices for the data used to generate candidate Q-Transform matrices.

In figure 11, we present results with a variety of different methods for generating candidates. In all cases, the source data was of small dimension ($X_S \in \mathbb{R}^{15\times 15}$), and the target data was larger ($X_T \in \mathbb{R}^{500\times 500}$). The target data had a true non-negative rank of 10 and factors were generated with i.i.d. entries from a standard normal. In all these experiments we set the transfer rank to be $R_T = R_{\text{SVD}} = 3$. We explored six ways of generating candidates from the source data:

- Uniform data: Generating a dataset X_S where each entry is i.i.d. with a uniform distribution in [0,1]; then apply random restarts to find candidate transforms.
- Simple sub-sample data: Generating dataset X_S by uniformly selecting 15 rows and columns of the target data X_T; then apply random restarts to find candidate transforms
- Column-projection data: Generating dataset X_S by sub-sampling 15 columns of X_T and applying a random projection into $\mathbb{R}^{15}$ for each each column; then apply random restarts to find candidate transforms.
- Dirichlet factors: Generating factors A, W with each column of A, W from a Dirichlet distribution (with concentration parameter α set to 1); let $X_S = AW$+Gaussian Noise; then apply random restarts to find candidate transforms.
- Uniform factors: Generating factors A, W with each entry i.i.d. from a uniform distribution in [0,1]; let $X_S = AW$ + Gaussian Noise; then apply random restarts to find candidate transforms.
- Gaussian factors: Generating factors A, W with each entry i.i.d. from a standard normal distribution; let $X_S = AW$ + Gaussian Noise; then apply random restarts to find candidate transforms.

The methods that produced the source data from some true NMF factors produced candidate transformations that resulted in the highest quality initializations on the target data (figure 11). In settings where a practitioner deals with a collection of similar NMF datasets (e.g. music analysis, hyper spectral images), there may be more clever ways in which the NMF solution spaces corresponding to a real dataset may yield more appropriate Q-Transforms specific to that type of data. Finally, we find in figure 12 that the performance of Gaussian factors method does not vary with the rank of the synthetic data (the transfer rank is still held fixed).

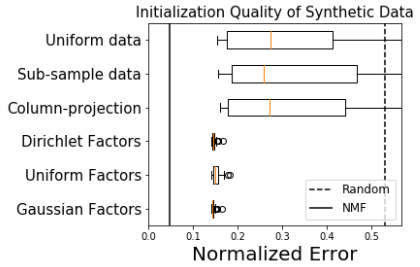


Figure 11: For different synthetic data X_S generating procedures, we show the initialization quality obtained via the Q-Transform matrices on a target data X_T. Dirichlet, Uniform, Gaussian have significantly superior performance compared to Sub-sample, Column-projection and Uniform data. For comparison, we show the quality of NMF solutions (solid line) and random initializations (dashed line).

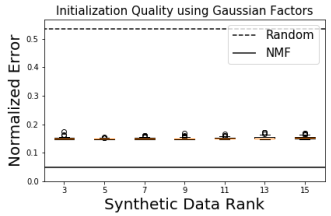


Figure 12: Using Gaussian factors for the synthetic data generation process with different ranks does not appear to change the quality of the Q-Transform initialization quality on the target data X_T. This indicates that this generating procedure is not sensitive to the rank in order to produce high quality (close to true NMF solution) initializations using Q-Transform. For comparison, we show the quality of NMF solutions (solid line) and random initializations (dashed line).

8.2. The Q-Transform Initialization versus Noise

In Section 4, we sought high-quality initializations because they generally require less time to converge. On synthetic target data $X_T = AW + \epsilon N_o$ (D = N = 500, R = 20) we explore the effect of increasing noise (ϵ) on the quality of our transfer-based NMF initializations and the time taken to converge. Specifically: are there noise regimes in which the Q-transform method works better, and noise regimes in which it does not?

We normalize the norm of the noise matrix to be equal to the norm of the data $\|N_o\| = \|AW\|$ so that the contribution of signal AW and noise ϵN_o to the data is equal when $\epsilon = 1$. We continue to use the same 100 pairs of Q_A, Q_W matrices. We compare the performance of Q-Transform over random restarts in terms of initialization quality (ratio of the reconstruction error from Q-Transform to the reconstruction error from random restart)

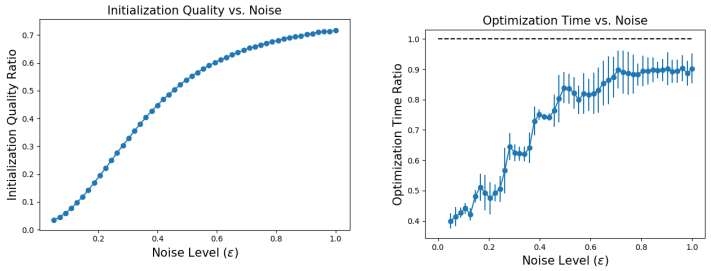


Figure 13: In the low-noise regime, the reconstruction error of Q-Transform initializations is significantly less than random restart initializations. This relative advantage gets smaller as the noise level increases. Similarly, the time taken to converge is significantly shorter than the random restart approach under the low noise scenario and continues to increase with noise. As expected, at high noise levels there exists no additional advantage to the Q-Transform approach (the optimization time ratio approaches 1).

and time to convergence (ratio of time taken using Q-Transform initialization to time taken using random restart). In both metrics, the Q-Transform has an advantage over random restarts for values of the noise ϵ smaller than 1, and the advantage is greatest for smallest noise. Figure 13 shows that the advantage of Q-Transform initializations is highest in a low noise regime and decreases as the noise increases. This behavior makes sense because as noise increases, the data is no longer truly low rank.

8.3. Selecting ranks

We emphasize that there are two distinct ranks that need to be chosen when applying our technique. The first is the rank of the factorization R_{NMF}. There exist multiple approaches for choosing this rank, e.g. Tan and Févotte (2009); Alquier and Guedj (2017), and they can be applied to our approach (as well as any other NMF algorithm).

The second is choosing the transfer rank R_T. The transformation dimensions R_T and R_{SVD} determine the dimensions of transformation matrices Q_A, Q_W which map basis vectors defining the top SVD subspace of dimension R_{SVD} to a set of R_T non-negative basis vectors that approximate the same subspace. The full initialization for NMF is obtained by either padding the initialization with small entries ($R_T < R_{\text{NMF}}$) or removing extra columns and rows of the factor matrices ($R_T > R_{\text{NMF}}$). (For simplicity, we consider the case where the transfer rank and SVD rank are equal $R_T = R_{\text{SVD}}$ and the resulting transformation matrices Q_A, Q_W are square.)

The choice of the transfer rank R_T is specific to our algorithm, and in figure 14 we investigate how well our transfer learning performs for different choices of transfer rank R_T. In the experiment, we extract a set of 100 transformation matrices Q_A, Q_W for transfer dimensions $R_T = R_{\text{SVD}} = \{1, 2, \ldots, 10\}$ using synthetic source data ($D = N = 15$). Once constructed, we applied the transformation matrices to a 500×500 target dataset X_T of rank $R = 10$. We find that even though the dataset X_T has rank 10, the rank 10 transformation

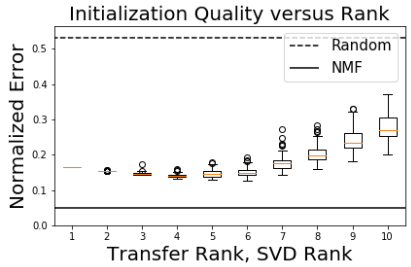


Figure 14: On a synthetic target dataset ($D = N = 500, K_{\text{NMF}} = 10$), we apply Q-Transform initializations using varying transfer ranks and SVD ranks $R_T = R_{\text{SVD}} = \{1, 2, \ldots, 10\}$. We see that for a range of low rank values, the Q-Transform initializations are high quality, but at larger values the quality of initializations gets worse. The dotted line shows the quality of random initializations and the solid line shows the quality NMF solutions. The reconstruction errors are normalized by the norm of the data.

matrices found using the 15×15 synthetic source dataset are unable to successfully transfer to this new dataset. We see that the error initially decreases, but then increases as the transfer rank increases. This result suggests that the top directions of variation hold the most transferable information across NMF problems.

8.4. Sign Convention for SVD

In considering when Q-Transform is successful, we note that there exists an intrinsic ambiguity in the sign of the singular vectors of X: changing the sign of any column of A_{SVD} and corresponding row of W_{SVD} gives a valid SVD. For Q-Transform to work, we must apply a consistent resolution of the sign ambiguity (e.g. from Bro et al. (2008)). This ensures that learned transformations Q_A, Q_W map in a consistent way to SVD decompositions of new datasets.

9. Related Work

There is a large body of work on inference for BNMF. Sampling-based approaches include Gibbs sampling (Schmidt et al., 2009), Hamiltonian Monte Carlo (Schmidt and Mohamed, 2009), and reversible jump variants (Schmidt and Mørup, 2010). All of these have trouble escaping local modes (Masood et al., 2016), and are often constrained to a limited class of tractable distributions. Variational approaches to BNMF have successfully yielded interpretable factorizations (Bertin et al., 2009; Cemgil, 2009; Paisley et al., 2014; Hinrich and Mørup, 2018) but also typically only capture one mode and rely on mean-field or other modeling assumptions to make inference tractable. We note that in many cases, priors of convenience—for example, exponential distributions—can induce a single dominant mode, even when that was not the intent of the practitioner.

Closer to the goals of our work, Gershman et al. (2012) develop a non-parametric approach to variational inference that provides flexibility in modeling the number of Gaussian components required to approximate a posterior. However, the isotropic covariance in the model makes it unsuitable for applying it to BNMF. With regard to the inference process, our Q-Transform approach to finding multiple optima is most similar to Ročková and George (2016) and Paatero and Tapper (1994), who use rotations to find solutions to a single matrix factorization problem that are sparse and non-negative respectively. In contrast, we use rotations to find multiple non-negative solutions, and also demonstrate how these rotations can be *re-used* for transfer learning.

More broadly, recent work on NMF has involved theoretical work on non-identifiability with new algorithms that can provably recover the NMF under certain assumptions (Li and Liang, 2017; Bhattacharya et al., 2016; Ge and Zou, 2015a). However, these assumptions are often difficult to check and may indeed be violated in practice; Bayesian methods typically provide more flexibility in modeling and assumptions.

All of the works above typically assume some desired factorization rank. There also exists work on models that automatically detect the rank—through automatic relevance determination for NMF (Tan and Févotte, 2009) or more recently, via a rank-adaptive prior Alquier and Guedj (2017). These works are complementary to ours, in that those techniques could be combined with our transfer-based approach of generating candidates of whatever rank those algorithms determine is appropriate.

The ability of Stein discrepancies to assess the quality of any collection of particles (Gorham and Mackey, 2015) has resulted in large recent interest in other ways to create collections of samples (Oates et al., 2017; Liu and Wang, 2016). Liu et al. (2016) and Chwialkowski et al. (2016) showed that kernelized Stein discrepancy could be computed analytically in Reproducing Kernel Hilbert Spaces (RKHS); Pu et al. (2017) and Feng et al. (2017) use neural networks instead. Ranganath et al. (2016) establish the Stein discrepancy as a valid variational objective. To our knowledge, Stein discrepancy-based posterior approximation has not been applied to NMF, and yet, we see that it allows us to leverage existing non-Bayesian approaches to characterize these multi-modal posteriors. In our work, the Dirichlet prior on the columns of the basis matrix A is important to ensure that we avoid a known saddle point of the zero factorization (from likelihood term) that yields a corresponding zero for the score function.

10. Conclusion

In this work, we presented a novel transfer learning-based approach to posterior estimation in BNMF. Simply creating collections of factorizations via random restarts on our Q-Transform initializations, and then weighing them, produces diverse collections that approximate the posterior well (the NNDSVDar-based methods fail to produce diverse collections for posterior estimation). In contrast, the functional gradient descent of SVGD and direct gradients of Stein discrepancy (DSGD) perform worse to the collection-based approaches, requiring more time and also limiting the user to specify in advance the number of factorizations. Hamiltonian Monte Carlo also suffers from difficulties in exploring the posterior space, something random initializations are well suited to. Our transfer learning approach consistently produces the highest quality posterior approximations.

Through Q-Transform, we introduce a way to speed-up the process of finding multiple diverse NMFs. The discovery that Q-Transform matrices can transfer from synthetic to multiple real datasets is exciting and also suggests interesting questions for further research. For example, what is the theoretical nature of the similarities between principal eigenspaces of different non-negative matrices and the relation between their SVD and NMF bases? And, how does the synthetic data generation process used to obtain Q-Transform matrices impact the initializations and the effectiveness of the Q-Transform algorithm in general?

More broadly, our qualitative results demonstrate that even relatively simple models, such as NMF, can have multiple optima that are comparable under the objective function but have large variation in how well they explain different portions of the data—or how they perform on different downstream tasks. Thus, it is important to be able to compute these posteriors efficiently.

Acknowledgments

We would like to thank anonymous reviewers for their insightful comments and suggestions. Additionally, we thank Andreas Krause, Mike Hughes, Melanie Pradier, Nick Foti, Omer Gottesman and other members of the Dtak lab at Harvard for many helpful conversations and feedback. MAM and FDV acknowledge support from AFOSR FA 9550-17-1-0155.

References

20NG. The 20 newsgroups text dataset scikit-learn 0.19.1 documentation. `http://scikit-learn.org/stable/datasets/twenty_newsgroups.html`, July 2013. (Accessed on 01/23/2018).

Pierre Alquier and Benjamin Guedj. An oracle inequality for quasi-Bayesian Nonnegative Matrix Factorization. *Mathematical Methods of Statistics*, 26(1):55–67, 2017.

Yoann Altmann, Nicolas Dobigeon, and Jean-Yves Tourneret. Unsupervised post-nonlinear unmixing of hyperspectral images using a Hamiltonian Monte Carlo algorithm. *IEEE Transactions on Image Processing*, 23(6):2663–2675, 2014.

Morten Arngren, Mikkel N Schmidt, and Jan Larsen. Unmixing of hyperspectral images using Bayesian Non-Negative Matrix Factorization with volume prior. *Journal of Signal Processing Systems*, 65(3):479–496, 2011.

Sanjeev Arora, Rong Ge, Ravindran Kannan, and Ankur Moitra. Computing a Non-Negative Matrix Factorization–provably. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 145–162. ACM, 2012.

Andrew D Barbour and Timothy C Brown. Stein's method and point process approximation. *Stochastic Processes and their Applications*, 43(1):9–31, 1992.

Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *arXiv preprint arXiv:1502.05767*, 2015.

Nancy Bertin, Roland Badeau, and Emmanuel Vincent. Fast Bayesian NMF algorithms enforcing harmonicity and temporal continuity in polyphonic music transcription. In *Applications of Signal Processing to Audio and Acoustics, 2009. WASPAA'09. IEEE Workshop on*, pages 29–32. IEEE, 2009.

Michael Betancourt. Cruising the simplex: Hamiltonian Monte Carlo and the Dirichlet distribution. In *AIP Conference Proceedings 31st*, volume 1443, pages 157–164. AIP, 2012.

Chiranjib Bhattacharya, Navin Goyal, Ravindran Kannan, and Jagdeep Pani. Non-Negative Matrix Factorization under Heavy Noise. In *International Conference on Machine Learning*, pages 1426–1434, 2016.

Chiranjib Bhattacharyya, IISC ERNET, Navin Goyal, COM Ravindran Kannan, and COM Jagdeep Pani. Non-Negative Matrix Factorization under heavy noise. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1426–1434, 2016.

José M Bioucas-Dias, Antonio Plaza, Nicolas Dobigeon, Mario Parente, Qian Du, Paul Gader, and Jocelyn Chanussot. Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, 5(2):354–379, 2012.

Christos Boutsidis and Efstratios Gallopoulos. SVD based initialization: A head start for Non-Negative Matrix Factorization. *Pattern Recognition*, 41(4):1350–1362, 2008.

Rasmus Bro, Evrim Acar, and Tamara G Kolda. Resolving the sign ambiguity in the Singular Value Decomposition. *Journal of Chemometrics*, 22(2):135–140, 2008.

Jean-Philippe Brunet, Pablo Tamayo, Todd R Golub, and Jill P Mesirov. Metagenes and molecular pattern discovery using Matrix Factorization. *PNAS*, 101(12):4164–4169, 2004.

CBCL. Home — poggio lab. `http://poggio-lab.mit.edu/`, 2000. (Accessed on 01/23/2018).

Ali Taylan Cemgil. Bayesian inference for Non-Negative Matrix factorisation models. *Computational Intelligence and Neuroscience*, 2009, 2009.

Wei Chu, S Sathiya Keerthi, and Chong Jin Ong. Bayesian support vector regression using a unified loss function. *IEEE transactions on neural networks*, 15(1):29–44, 2004.

Kacper Chwialkowski, Heiko Strathmann, and Arthur Gretton. A kernel test of goodness of fit. *arXiv preprint arXiv:1602.02964*, 2016.

Andrzej Cichocki and Anh-Huy Phan. Fast local algorithms for large scale Non-Negative Matrix and Tensor Factorizations. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 92(3):708–721, 2009.

Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

Onur Dikmen and A Taylan Cemgil. Unsupervised single-channel source separation using Bayesian NMF. In *Applications of Signal Processing to Audio and Acoustics, 2009. WASPAA'09. IEEE Workshop on*, pages 93–96. IEEE, 2009.

David Donoho and Victoria Stodden. When does Non-Negative Matrix Factorization give a correct decomposition into parts? In *Advances in neural information processing systems*, page None, 2003.

Finale Doshi-Velez, Yaorong Ge, and Isaac Kohane. Comorbidity clusters in autism spectrum disorders: an electronic health record time-series analysis. *Pediatrics*, 133(1):e54–e63, 2014.

Yihao Feng, Dilin Wang, and Qiang Liu. Learning to draw samples with amortized Stein Variational Gradient Descent. *arXiv preprint arXiv:1707.06626*, 2017.

Cédric Févotte and Jérôme Idier. Algorithms for Non-Negative Matrix Factorization with the β-divergence. *Neural computation*, 23(9):2421–2456, 2011.

Cédric Févotte, Nancy Bertin, and Jean-Louis Durrieu. Non-Negative Matrix Factorization with the Itakura-Saito divergence: With application to music analysis. *Neural computation*, 21(3):793–830, 2009.

Rong Ge and James Zou. Intersecting faces: Non-Negative Matrix Factorization with new guarantees. In *International Conference on Machine Learning*, pages 2295–2303, 2015a.

Rong Ge and James Zou. Intersecting faces: Non-Negative Matrix Factorization with new guarantees. In *International Conference on Machine Learning*, page X. ICML, 2015b.

Michel Gendreau and Jean-Yves Potvin. *Handbook of metaheuristics*, volume 2. Springer, 2010.

Samuel Gershman, Matt Hoffman, and David Blei. Nonparametric variational inference. *arXiv preprint arXiv:1206.4665*, 2012.

Jackson Gorham and Lester Mackey. Measuring sample quality with Stein's method. In *Advances in Neural Information Processing Systems*, pages 226–234, 2015.

Jackson Gorham and Lester Mackey. Measuring sample quality with kernels. *arXiv preprint arXiv:1703.01717*, 2017.

Derek Greene, Gerard Cagney, Nevan Krogan, and Pádraig Cunningham. Ensemble Non-Negative Matrix Factorization methods for clustering protein–protein interactions. *Bioinformatics*, 2008.

Arthur Gretton, Karsten M Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J Smola. A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520, 2006.

Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

Jesper Løve Hinrich and Morten Mørup. Probabilistic Sparse Non-negative Matrix Factorization. In *International Conference on Latent Variable Analysis and Signal Separation*, pages 488–498. Springer, 2018.

Matthew D Hoffman and David M Blei. Structured stochastic Variational Inference. In *Artificial Intelligence and Statistics*, 2015.

Cho-Jui Hsieh and Inderjit S Dhillon. Fast coordinate descent methods with variable selection for Non-Negative Matrix Factorization. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1064–1072. ACM, 2011.

Oliver Jesorsky, Klaus J Kirchberg, and Robert W Frischholz. Robust face detection using the hausdorff distance. In *International conference on audio-and video-based biometric person authentication*, pages 90–95. Springer, 2001.

Daniel D Lee and H Sebastian Seung. Algorithms for Non-Negative Matrix Factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

LFW. Lfw face database : Main. `http://vis-www.cs.umass.edu/lfw/`, May 2017. (Accessed on 01/23/2018).

Yuanzhi Li and Yingyu Liang. Provable Alternating Gradient Descent for Non-negative Matrix Factorization with Strong Correlations. *arXiv preprint arXiv:1706.04097*, 2017.

Chih-Jen Lin. Projected gradient methods for Non-Negative Matrix Factorization. *Neural computation*, 19(10):2756–2779, 2007.

Qiang Liu and Yihao Feng. Two Methods for wild Variational Inference. *arXiv preprint arXiv:1612.00081*, 2016.

Qiang Liu and Jason D Lee. Black-box importance sampling. *arXiv preprint arXiv:1610.05247*, 2016.

Qiang Liu and Dilin Wang. Stein Variational Gradient Descent: A general purpose Bayesian inference algorithm. In *Advances in Neural Information Processing Systems*, pages 2370–2378, 2016.

Qiang Liu, Jason Lee, and Michael Jordan. A kernelized Stein discrepancy for goodness-of-fit tests. In *International Conference on Machine Learning*, pages 276–284, 2016.

Dougal Maclaurin, David Duvenaud, and Ryan P Adams. Autograd: Reverse-mode differentiation of native python. In *ICML workshop on Automatic Machine Learning*, 2015.

Arjumand Masood, Weiwei Pan, and Finale Doshi-Velez. An empirical comparison of sampling quality metrics: A case study for Bayesian Non-Negative Matrix Factorization. *arXiv preprint arXiv:1606.06250*, 2016.

Saïd Moussaoui, David Brie, Ali Mohammad-Djafari, and Cédric Carteret. Separation of Non-Negative mixture of Non-Negative sources using a Bayesian approach and MCMC sampling. *Signal Processing, IEEE Transactions on*, 54(11):4133–4145, 2006.

Radford M Neal et al. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2:113–162, 2011.

Robert J. Plemmons Nicolas Gillis. Hubble telescope hyperspectral image data, 1987. URL `https://sites.google.com/site/nicolasgillis/code`.

Chris J Oates, Mark Girolami, and Nicolas Chopin. Control functionals for Monte Carlo integration. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(3):695–718, 2017.

Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. The building blocks of interpretability. *Distill*, 2018. doi: 10.23915/distill.00010. https://distill.pub/2018/building-blocks.

Pentti Paatero and Unto Tapper. Positive Matrix Factorization: A Non-Negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2): 111–126, 1994.

John Paisley, David M Blei, and Michael I Jordan. Bayesian Non-Negative Matrix Factorization with stochastic variational inference. *Handbook of Mixed Membership Models and Their Applications. Chapman and Hall/CRC*, 2015.

John William Paisley, David M Blei, and Michael I Jordan. Bayesian Non-Negative Matrix Factorization with Stochastic Variational Inference., 2014.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

Weiwei Pan and Finale Doshi-Velez. A characterization of the non-uniqueness of Non-Negative Matrix Factorizations. *arXiv preprint arXiv:1604.00653*, 2016.

Sam Patterson and Yee Whye Teh. Stochastic gradient Riemannian Langevin dynamics on the probability simplex. In *Advances in Neural Information Processing Systems*, pages 3102–3110, 2013.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

Yunchen Pu, Zhe Gan, Ricardo Henao, Chunyuan Li, Shaobo Han, and Lawrence Carin. Stein Variational Autoencoder. *arXiv preprint arXiv:1704.05155*, 2017.

Rajesh Ranganath, Dustin Tran, Jaan Altosaar, and David Blei. Operator Variational Inference. In *Advances in Neural Information Processing Systems*, pages 496–504, 2016.

Margaret E Roberts, Brandon M Stewart, and Dustin Tingley. Navigating the local modes of big data. *Computational Social Science*, page 51, 2016.

Veronika Ročková and Edward I George. Fast Bayesian factor analysis via automatic rotations to sparsity. *Journal of the American Statistical Association*, 111(516):1608–1622, 2016.

Ruslan Salakhutdinov, Sam Roweis, and Zoubin Ghahramani. On the convergence of bound optimization algorithms. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 509–516. Morgan Kaufmann Publishers Inc., 2002.

SalinasA. Multispec — home. `https://engineering.purdue.edu/~biehl/MultiSpec/`, June 2015. (Accessed on 01/23/2018).

Ferdinando S. Samaria. The database of faces (olivetti), 1994. URL `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html`.

Mikkel N Schmidt and Shakir Mohamed. Probabilistic Non-Negative Tensor Factorization using Markov Chain Monte Carlo. In *Signal Processing Conference, 2009 17th European*, pages 1918–1922. IEEE, 2009.

Mikkel N Schmidt and Morten Mørup. Reversible jump MCMC for Bayesian NMF. In *Proc. NIPS Workshop on Monte Carlo Methods for Modern Applications*, 2010.

Mikkel N Schmidt, Ole Winther, and Lars Kai Hansen. Bayesian Non-Negative Matrix Factorization. In *Independent Component Analysis and Signal Separation*, pages 540–547. Springer, 2009.

Paris Smaragdis and Judith C Brown. Non-Negative Matrix Factorization for polyphonic music transcription. In *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pages 177–180. IEEE, 2003.

Cristian Sminchisescu, Max Welling, and G Hinton. Generalized darting Monte Carlo. In *AISTATS*, pages 516–523. Citeseer, 2007.

Vincent YF Tan and Cédric Févotte. Automatic relevance determination in Nonnegative Matrix Factorization. In *SPARS'09-Signal Processing with Adaptive Sparse Structured Representations*, 2009.

Per-Åke Wedin. Perturbation bounds in connection with Singular Value Decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

Stefan Wild, James Curry, and Anne Dougherty. Improving Non-Negative Matrix Factorizations through structured initialization. *Pattern recognition*, 37(11):2217–2232, 2004.

Yun Xue, Chong Sze Tong, Ying Chen, and Wen-Sheng Chen. Clustering-based initialization for Non-Negative Matrix Factorization. *Applied Mathematics and Computation*, 205(2): 525–536, 2008.

Feiyun Zhu, Ying Wang, Shiming Xiang, Bin Fan, and Chunhong Pan. Structured sparse method for hyperspectral unmixing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 88:101–118, 2014.

Appendix A: Quality of factorizations

We measure the quality of factorizations in terms of the log of the joint likelihood (figures 18 and 15) as well the Frobenius NMF objective (figures 19 and 16). We see that both quality measures are in agreement with each other. We also assess the quality of factorizations via the reconstruction error of heldout data (figures 20 and 17).

Overall, Q-Transform, Random, NNDSVDar, HMC and (for the exponential-Gaussian model) Gibbs produce high quality factorizations. The initialization-based approaches all work well as they use specialized NMF algorithms that are designed to find high quality factorizations. HMC was given a warm start with a high likelihood initialization and the chain continues to stay in high likelihood regions. The remaining gradient-based approaches for optimizing a collection of particles (SVGD and DSGD) fail to produce high quality factorizations. This is indicative of the need for specialized NMF algorithms designed to work with the constraints and structure of the NMF problem and highlight how difficult it is to apply a naive gradient descent approach for finding NMFs.

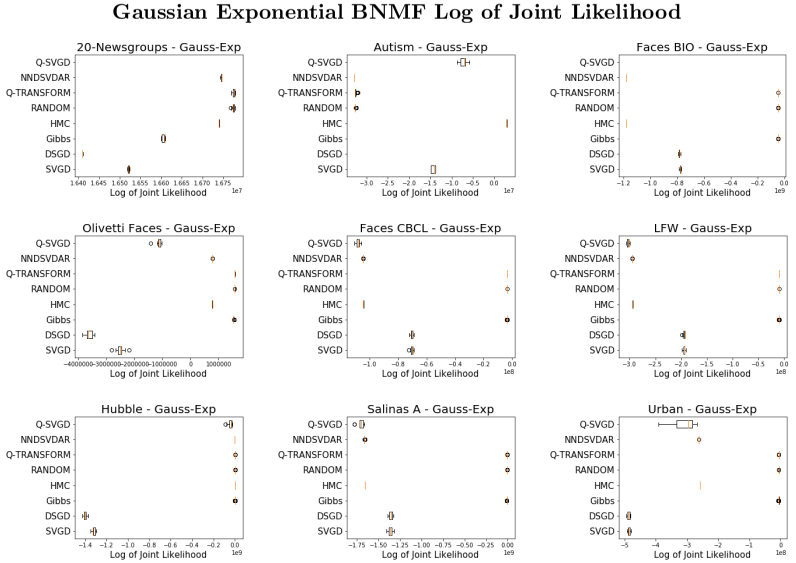


Figure 15: The joint likelihood of factorizations shows that SVGD and DSGD generally produce the worst quality factorizations. The remaining algorithms produce higher quality factorizations.

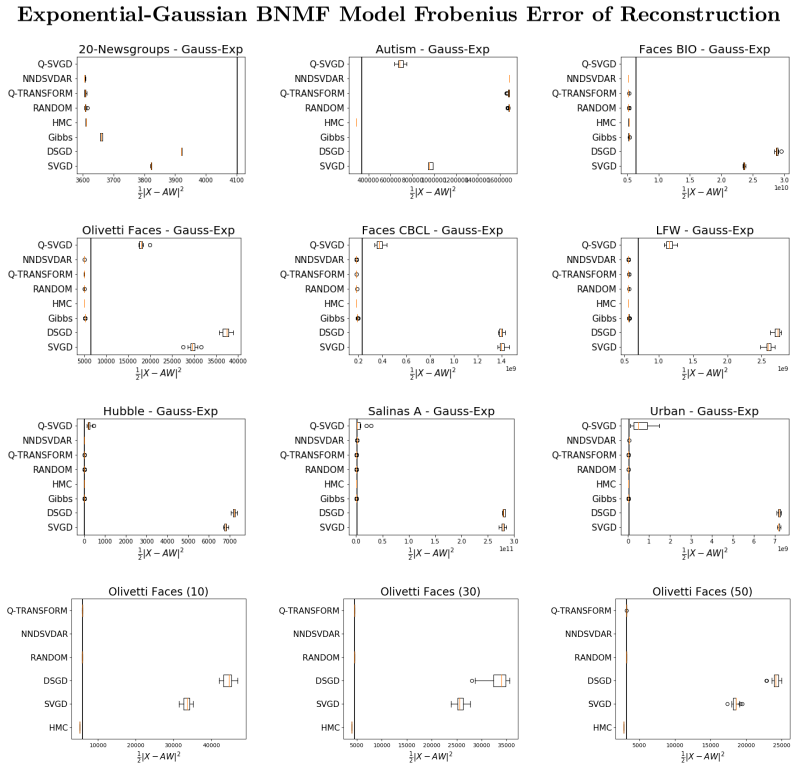


Figure 16: The reconstruction error of the factorizations shows that SVGD and DSGD are typically unable to find factorization parameters that meet the threshold quality (black line) for useful factorizations. The other approaches consistently produce factorizations that meet this minimum quality requirement.

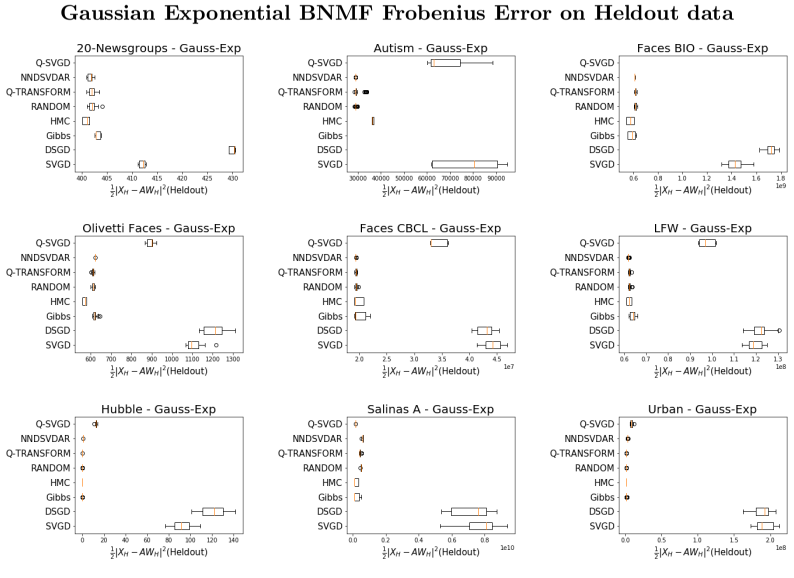


Figure 17: The reconstruction error of the factorizations shows that SVGD and DSGD are typically unable to find factorization parameters that result in low error on heldout data. The other approaches consistently produce factorizations which generalize better and give low error on heldout data.

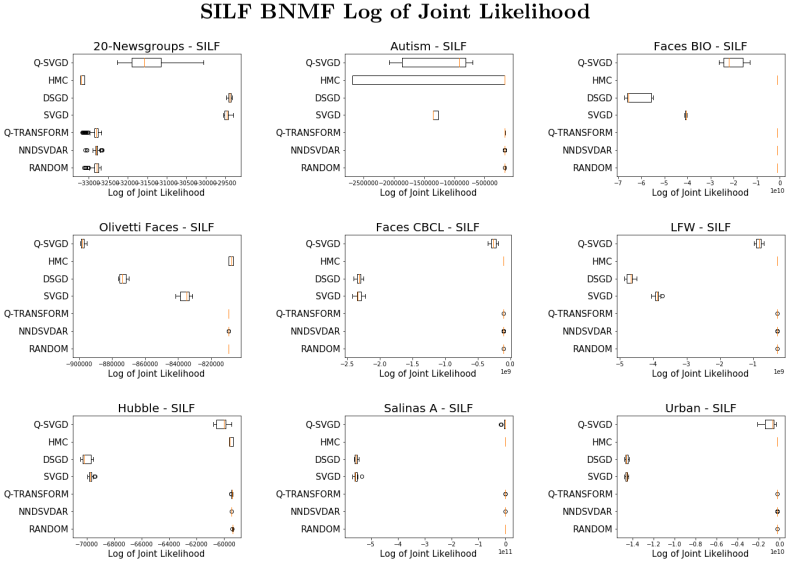


Figure 18: The joint likelihood of factorizations shows that SVGD and DSGD generally produce the worst quality factorizations. HMC, NNDSVDar, Random and Q-Transform produce high quality factorizations.

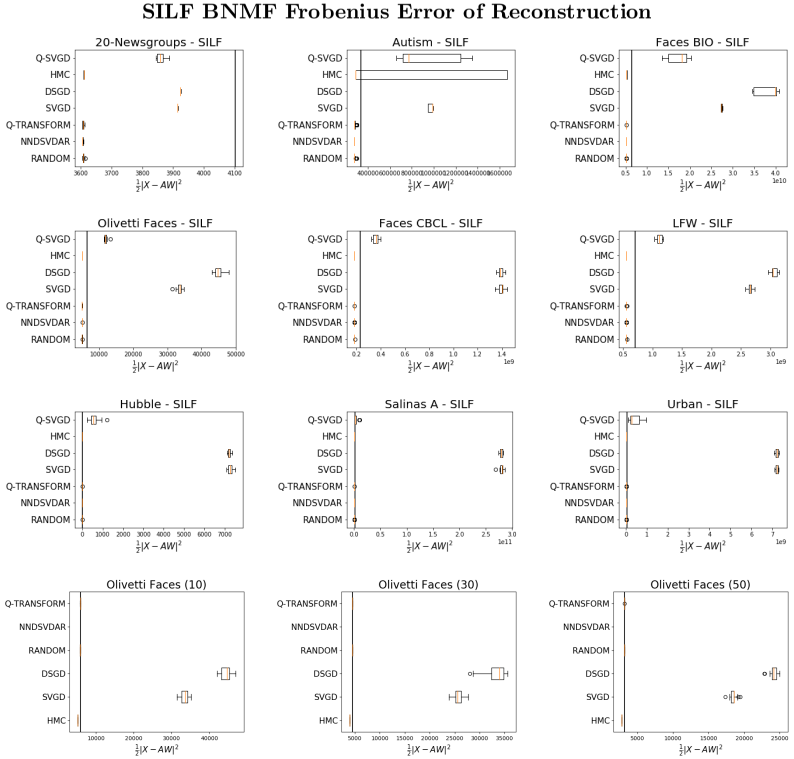


Figure 19: The reconstruction error of the factorizations shows that SVGD and DSGD are typically unable to find factorization parameters that meet the threshold quality (black line) for useful factorizations. The other approaches consistently produce factorizations that meet this minimum quality requirement.

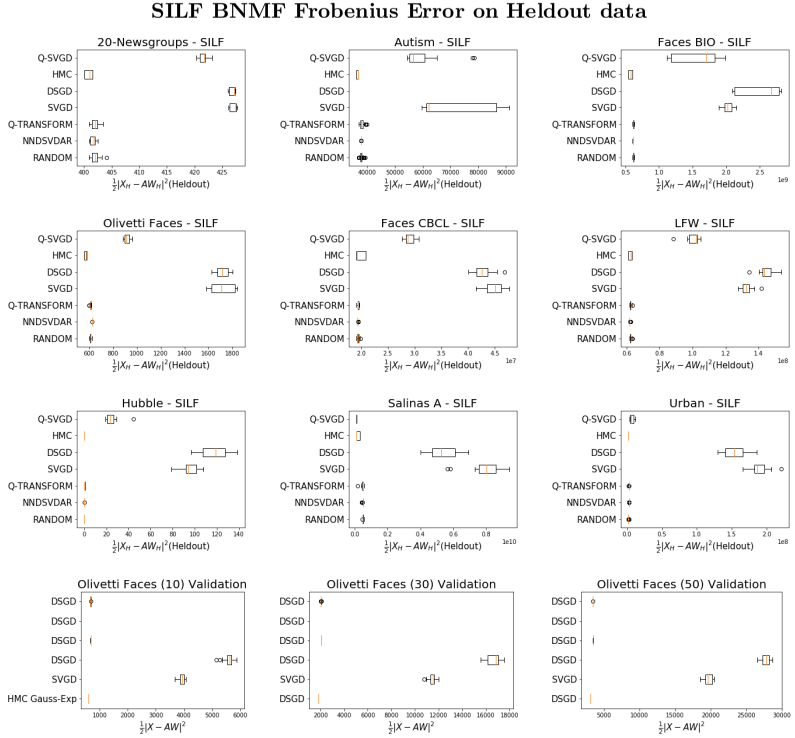


Figure 20: The reconstruction error of the factorizations shows that SVGD and DSGD are typically unable to find factorization parameters that result in low error on heldout data. The other approaches consistently produce factorizations which generalize better and give low error on heldout data. The matrix completion error on the missing data variant of the Olivetti faces dataset also gives similar results.

Appendix B: Diversity of factorizations

Similarity of factorizations is measured by the kernel (equation 7) for the base RKHS (figures 24 and 21) used in evaluating the Stein discrepancy and by pairwise distances between basis matrices (figures 25 and 22) and weights matrices (figures 26 and 23)[11]. Generally, the HMC chain exhibits the least exploration of the factorization space. Remaining algorithms, particularly Q-Transform exhibits higher amounts of diversity in the factorization space. The diversity metrics indicate that SVGD and DSGD give diverse factorizations but the quality metrics indicate that these factorizations are poor quality (do not correspond to high likelihood regions of the posterior) therefore such diversity is of little interest.

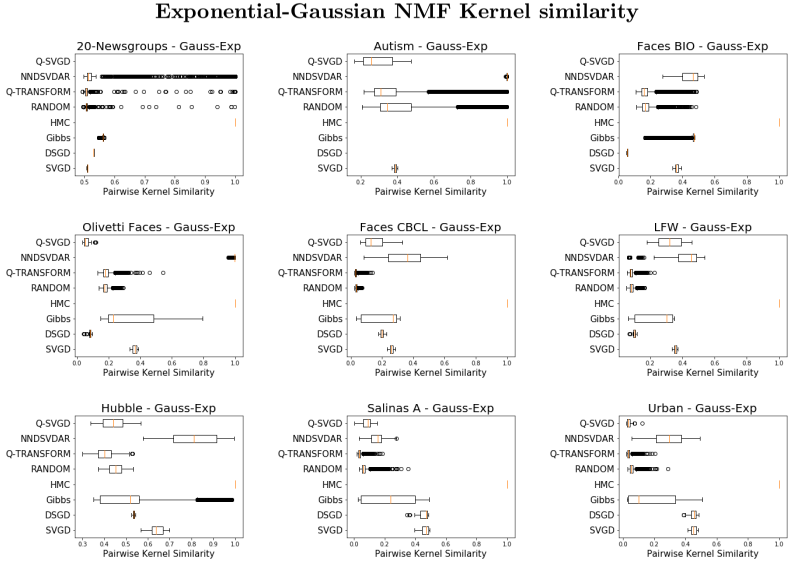


Figure 21: The kernel similarity indicates that factorization collections obtained by HMC are most similar indicating that the HMC chain is only exploring a small region of the posterior. In many cases NNDSVDar factorizations are also very similar. Q-Transform and Random are the only algorithms that produce factorizations of high quality that are not similar.

11. In the exponential-Gaussian model, we adjust the scalings of the factorizations so that we can meaningfully compare pairwise distances between basis and weights matrices.

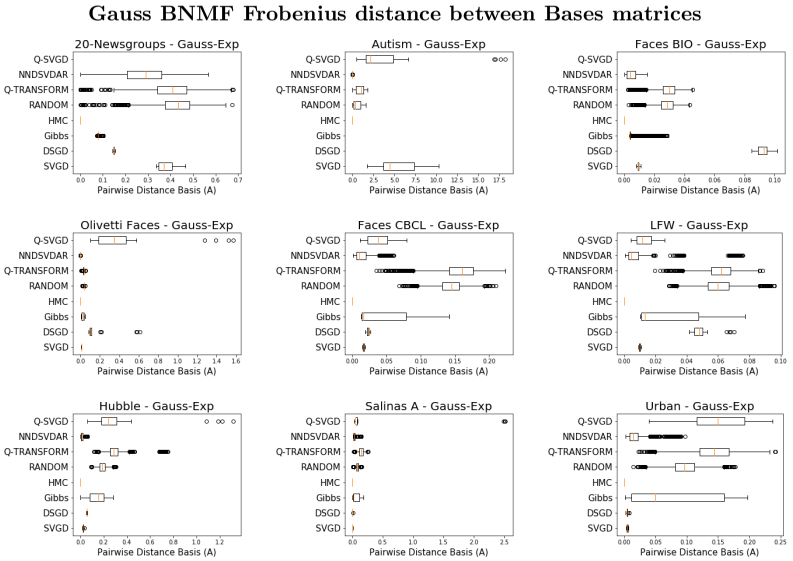


Figure 22: The pairwise distance between basis matrices shows that factorization collections obtained by HMC are most similar indicating that the HMC chain is only exploring a small region of the posterior. In many cases NNDSVDar, Q-SVGD, DSGD and SVGD factorizations are also very similar. Q-Transform and Random produce basis matrices that are more distinct than the Gibbs sampler.

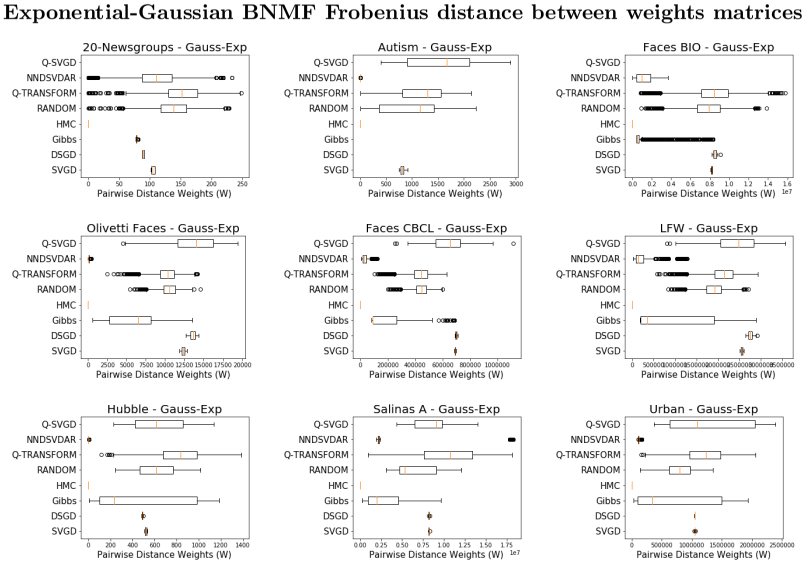


Figure 23: The pairwise distance between weights matrices shows that factorization collections obtained by DSGD, SVGD and Q-SVGD lead to diverse weights matrices. The remaining methods lead to similar levels of diversity within the factorization collections.

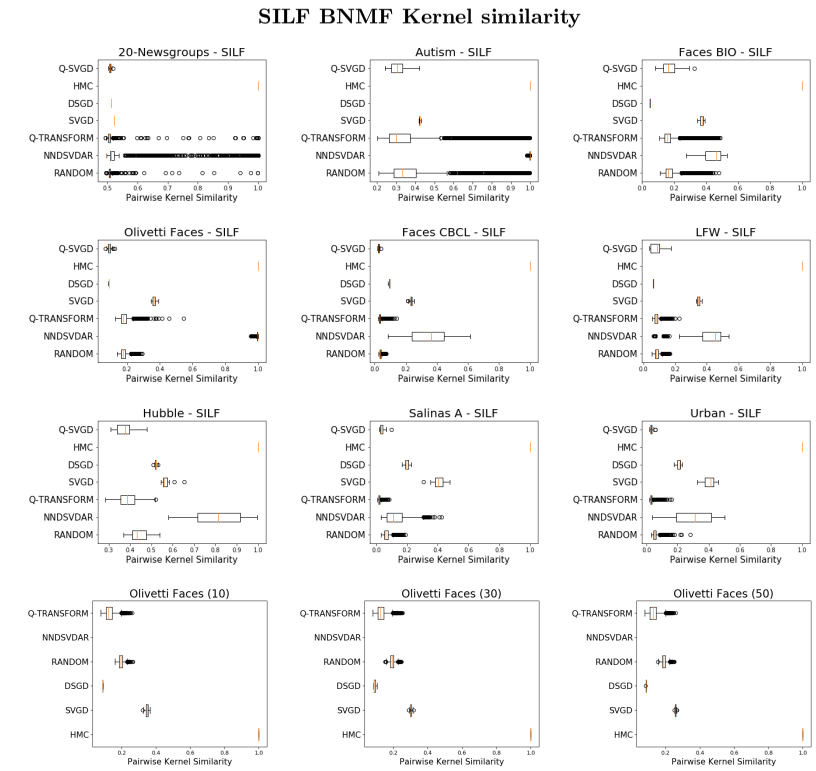


Figure 24: The kernel similarity indicates that factorization collections obtained by HMC are most similar indicating that the HMC chain is only exploring a small region of the posterior. In many cases NNDSVDar factorizations are also very similar.

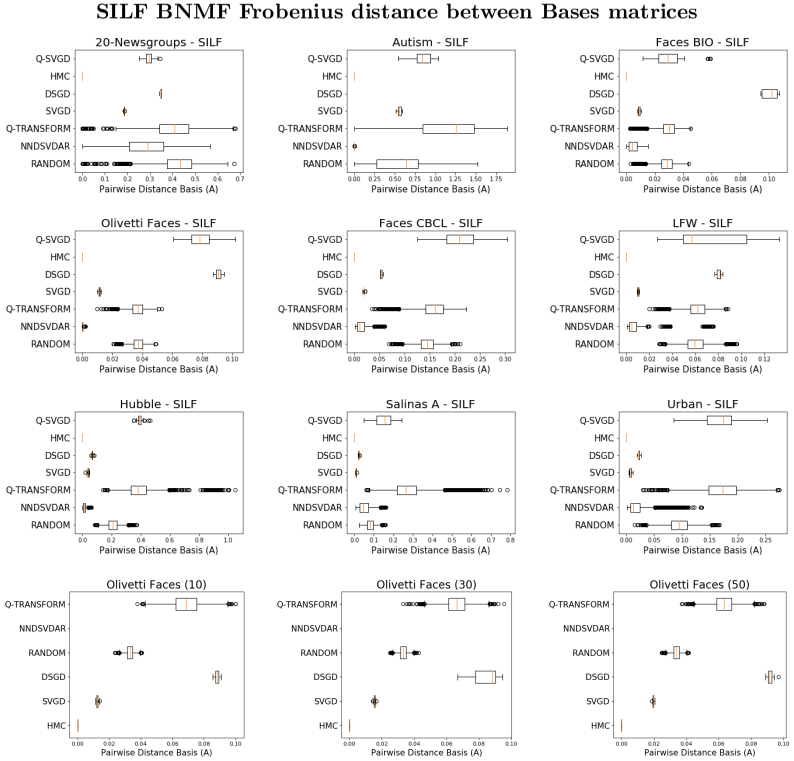


Figure 25: The pairwise distance between basis matrices shows that factorization collections obtained by HMC are most similar indicating that the HMC chain is only exploring a small region of the posterior. In many cases NNDSVDar factorizations are also very similar. *Q*-Transform and Random are the only algorithms that produce factorizations of high quality that are different.

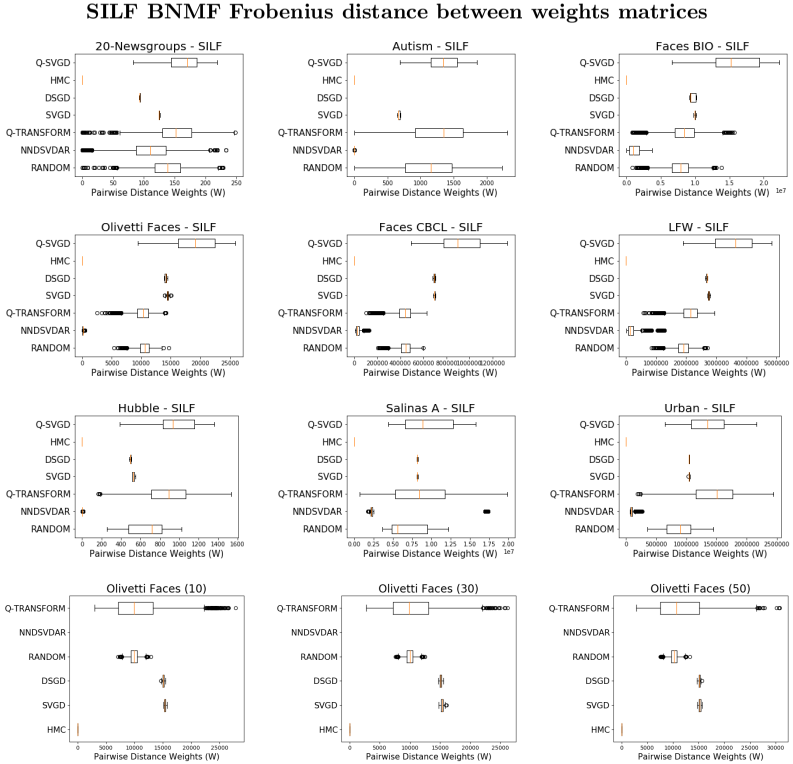


Figure 26: The pairwise distance between weights matrices shows that factorization collections obtained by HMC are most similar indicating that the HMC chain is only exploring a small region of the posterior. In many cases NNDSVDar factorizations are also very similar. *Q*-Transform and Random are the only algorithms that produce factorizations of high quality that are different.

Appendix C: Baseline Performance on Prediction Tasks

We show the performance of other baseline algorithms on the prediction tasks for the 20-Newsgroups and Autism datasets (see figures 6 and 8 in main text for reference)

Autism Prediction Task: Performance of baseline algorithms

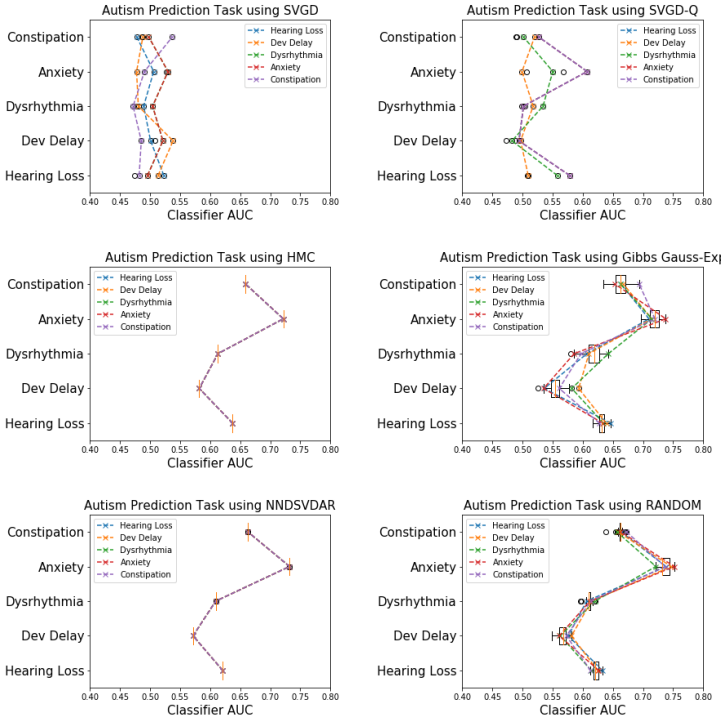


Figure 27: Variability in the prediction task on the Autism dataset shows that SVGD factorizations yield poor prediction (that are slightly improved upon by Q-SVGD), HMC and NNDSVDar factorizations predictions are high but not diverse, and Gibbs and random restarts yields performance similar to Q-Transform

20 Newsgroups Prediction Task: Performance of baseline algorithms

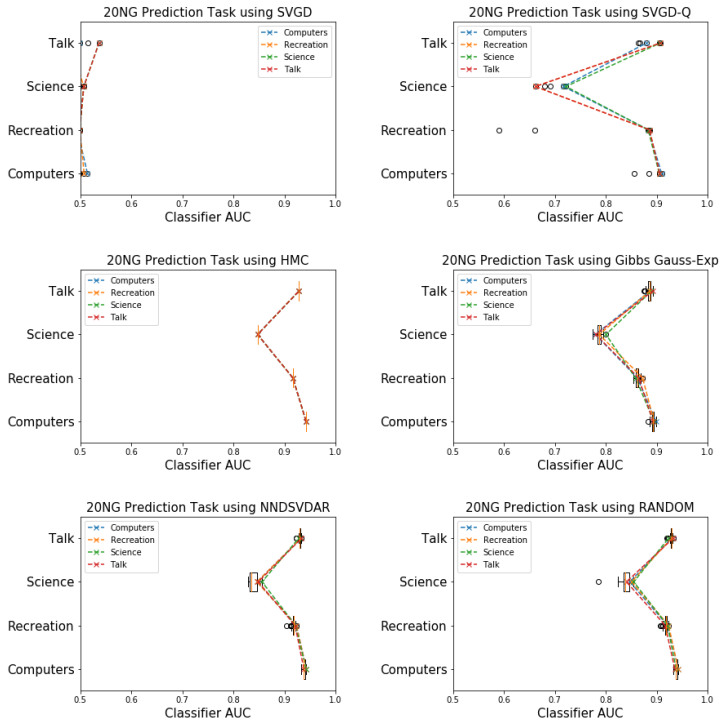


Figure 28: Variability in the prediction task on the 20 Newsgroups dataset shows that SVGD factorizations yield poor prediction (that is drastically improved upon by Q-SVGD), HMC factorizations are not diverse, NNDSVDar, Gibbs and random restarts yields some variability in prediction that is similar to Q-Transform

Appendix D: Discrete posteriors for $M = \{25, 50\}$

We provide results on the quality of discrete BNMF posteriors for the SILF and exponential-Gaussian models for $M = 25$ and $M = 50$.

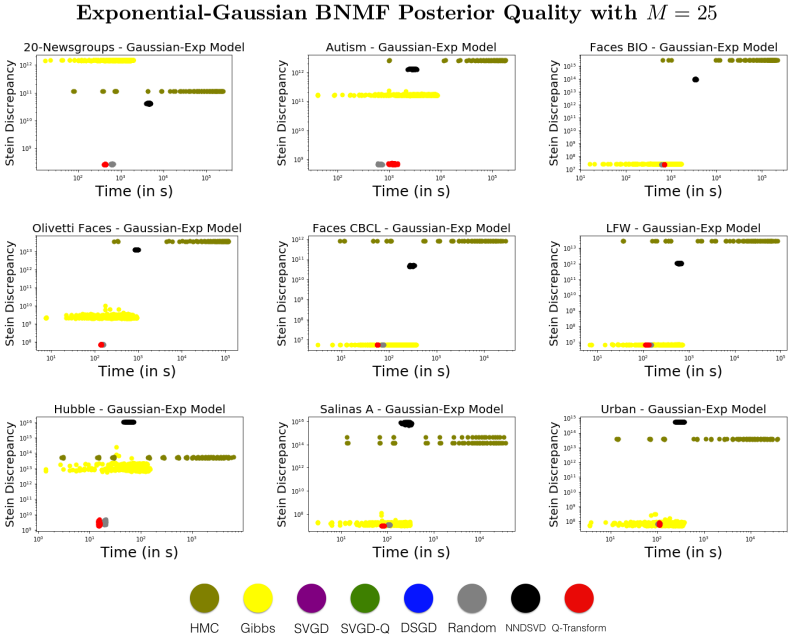


Figure 29: For each dataset we show the quality of the BNMF approximate posterior ($M = 25$) and the corresponding runtime of Q-Transform and the other baselines. Across multiple datasets, we see that the best discrete posteriors to BNMF (lowest Stein discrepancy) are produced in the least time using the Q-Transform initializations (in red).

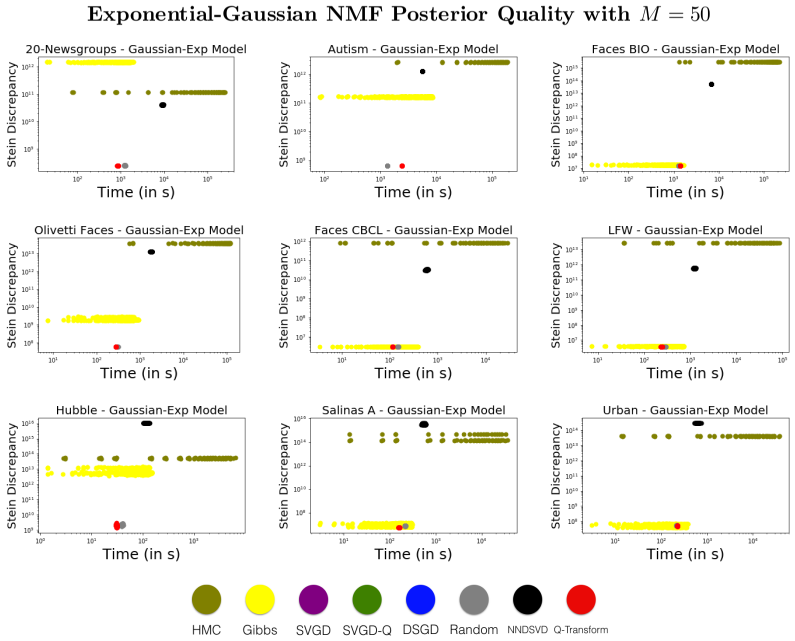


Figure 30: For each dataset we show the quality of the BNMF approximate posterior ($M = 50$) and the corresponding runtime of Q-Transform and the other baselines. Across multiple datasets, we see that the best discrete posteriors to BNMF (lowest Stein discrepancy) are produced in the least time using the Q-Transform initializations (in red).

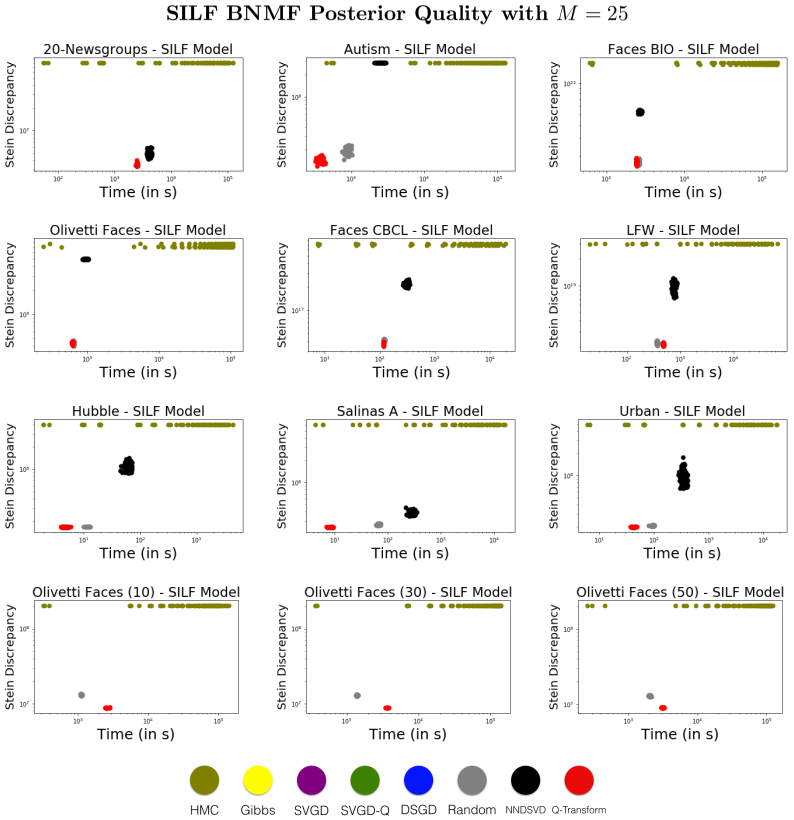


Figure 31: For each dataset we show the quality of the BNMF approximate posterior ($M = 25$) and the corresponding runtime of Q-Transform and the other baselines. Across multiple datasets, we see that the best discrete posteriors to BNMF (lowest Stein discrepancy) are produced in the least time using the Q-Transform initializations (in red).

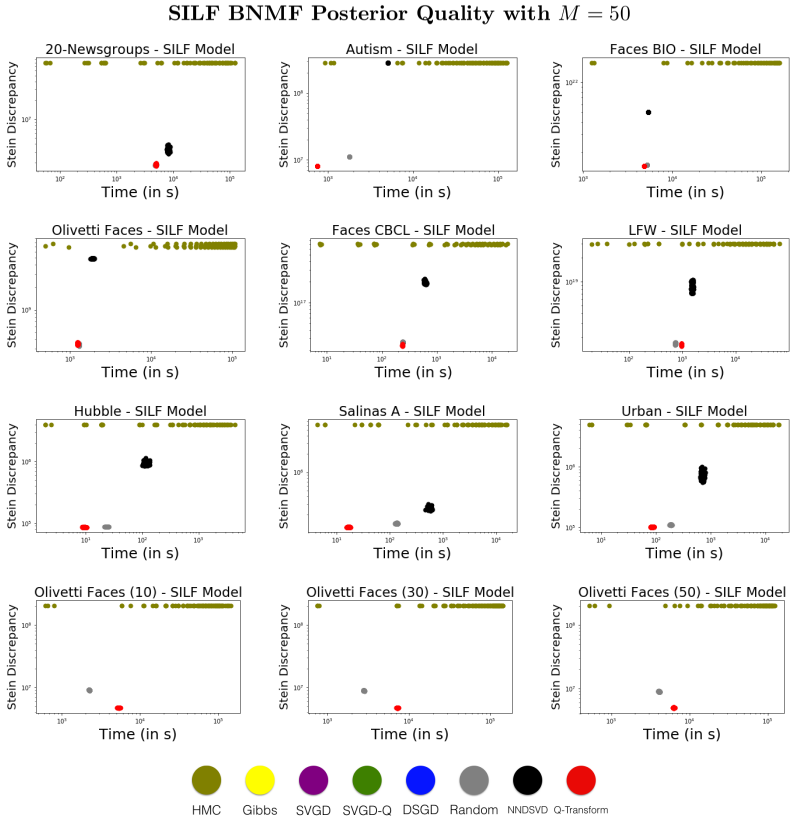


Figure 32: For each dataset we show the quality of the BNMF approximate posterior ($M = 50$) and the corresponding runtime of Q-Transform and the other baselines. Across multiple datasets, we see that the best discrete posteriors to BNMF (lowest Stein discrepancy) are produced in the least time using the Q-Transform initializations (in red).

Appendix E: Runtime Calculation

Here we provide full details on how our runtimes were calculated. Recall that both our transfer-based algorithm and the baselines all can be considered as having two phases: (1) Producing candidate particles $\{\theta_m\}_{m=1}^M$, and (2) Calculating the optimal weights $\{w_m\}_{m=1}^M$ to minimize the Stein discrepancy. Importantly, to give all the methods their best performance, we calculate optimal weights for *all* the methods, including those such as MCMC that produce unweighted collections.

Runtime for generating candidate particles Thus, all the methods only differ in how the candidate particles $\{\theta_m\}_{m=1}^M$ are found. To focus on the parts that differ, in our figures, we only report on the computational time required to produce these candidate particles. Below we describe the computations for obtaining the candidate particles:

- MCMC Methods: We keep track of the time from the initialization of the chain through each element added to the chain. At various points, we thin the current chain to the desired number of particles and record the time elapsed since initialization.
- Gradient-based Optimization: We keep track of the time taken from initialization through all gradient updates. At various points, we output the current particles and record the time elapsed since initialization.
- Initialization Approaches: For each factorization θ_m, we keep track of the time taken from initialization through optimization using NMF solvers. In the case of Q-Transform, we also include the time taken to find the transferable Q matrices (less than 0.5 seconds for all 100 pairs of Q-Transform matrices!), even though that cost is shared by all the datasets because we re-use the same matrices. The reported runtime is the sum of the time taken for the initialization procedure and for running the NMF solver.

Runtime for optimizing weights We do not report the time required to weight the particles because it is the same for all methods. In figure 33 we see the time to optimize the weights depends only on the number of particles and not their parameter values or quality.

Runtime for calculating weights: NMF with different parameters

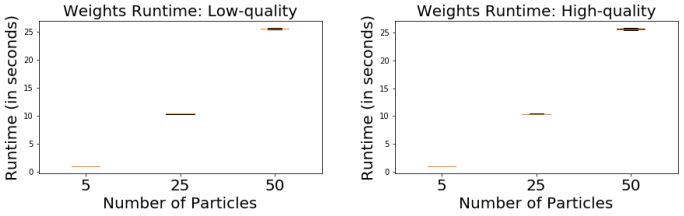


Figure 33: The runtime for calculating optimal weights (left: low-quality random parameters, right: "good" parameters obtained from NMF solvers) does not depend on the parameters; both have the same runtime costs and depend only on collection size.

Note for efficient batch weight optimization To produce the confidence intervals in our experiments, we had to run our approaches many times. To speed up these repetitions, we observed that the weight optimization step first requires computing the pairwise kernel matrix $\mathbf{K}_{ij} = \mathcal{K}_p(\theta_i, \theta_j)$ (from equation 3) and given $\mathbf{K}$, running a solver to find the weights $\{w_m\}_{m=1}^M$. When computing the weights for many subsets of a collection of factorizations (as is the case with the initialization based approaches), we can avoid recomputing terms in the kernel matrix multiple times by calculating the pairwise kernel matrix $\mathbf{K}_{\max}$ for a large collection of $M_{\max}$ factorizations. Subsequently, the matrix $\mathbf{K}_M$ for a given subset of size M (from this large collection) can be determined by simply choosing the relevant columns and rows of $\mathbf{K}_{\max}$. We use this approach to compute quickly compute weights for any M-sized collection sampled from a larger set of $M_{\max}$ factorizations. Note that this efficiency in experimental design trick does not affect our reported numbers, which include only the time to generate the candidate particles, but may help others achieve faster computational times if they are replicating our results.

Journal of Machine Learning Research 20 (2019) 1-38 Submitted 6/18; Revised 2/19; Published 5/19

Best Arm Identification for Contaminated Bandits

Jason Altschuler JASONALT@MIT.EDU
Victor-Emmanuel Brunel VEBRUNEL@MIT.EDU
Alan Malek AMALEK@MIT.EDU
Massachusetts Institute of Technology
77 Massachusetts Ave, Cambridge, MA 02139

Editor: Csaba Szepesvári

Abstract

This paper studies active learning in the context of robust statistics. Specifically, we propose a variant of the Best Arm Identification problem for *contaminated bandits*, where each arm pull has probability ε of generating a sample from an arbitrary contamination distribution instead of the true underlying distribution. The goal is to identify the best (or approximately best) true distribution with high probability, with a secondary goal of providing guarantees on the quality of this distribution. The primary challenge of the contaminated bandit setting is that the true distributions are only partially identifiable, even with infinite samples. To address this, we develop tight, non-asymptotic sample complexity bounds for high-probability estimation of the first two robust moments (median and median absolute deviation) from contaminated samples. These concentration inequalities are the main technical contributions of the paper and may be of independent interest. Using these results, we adapt several classical Best Arm Identification algorithms to the contaminated bandit setting and derive sample complexity upper bounds for our problem. Finally, we provide matching information-theoretic lower bounds on the sample complexity (up to a small logarithmic factor).

Keywords: multi-armed bandits, best arm identification, robust statistics, contamination model, partial identifiability

1. Introduction

Consider Pat, an aspiring machine learning researcher deciding between working in a statistics, mathematics, or computer science department. Pat's sole criterion is salary, so Pat surveys current academics with the goal of finding the department with highest median income. However, some subset of the data will be inaccurate: some respondents obscure their salaries for privacy, some convert currency incorrectly, and some do not read the question and report yearly instead of monthly salary, etc. How should Pat target his or her surveys, in an adaptive (online) fashion, to find the highest paying department with high probability in the presence of contaminated data?

In this paper, we study the Best Arm Identification (BAI) problem for multi-armed bandits where observed rewards are not completely trustworthy. The multi-armed bandit problem has received extensive study in the last three decades (Lai and Robbins, 1985; Bubeck and Cesa-Bianchi, 2012). We study the *fixed confidence*, or (α, δ)-PAC, BAI problem, in which the learner must identify an α-suboptimal arm with probability at least $1 - \delta$, using as few samples as possible. Most BAI algorithms for the fixed-confidence setting assume i.i.d. rewards from distributions with relatively strict control on the tails, such as boundedness or more generally sub-Gaussianity (Jamieson et al., 2014). However, for Pat, the data are neither i.i.d. nor from a distribution with controllable tails. How can we model this data, and how can we optimally explore these arms?

To answer the first question, we turn to robust statistics, which has studied such questions for over fifty years. In a seminal paper, Huber (1964) introduced the contamination model, which we adapt to the multi-armed bandit model by proposing the *Contaminated Best Arm Identification problem* (CBAI). This is formally defined in Section 2, but is informally described as follows. There are $k \geq 2$ arms, each endowed with a fixed base distribution F_i and arbitrary contamination distributions $G_{i,t}$ for $t \geq 1$. We place absolutely no assumptions on $G_{i,t}$. When arm i is pulled in round t, the learner receives a sample that with probability $1 - \varepsilon$ is drawn from the arm's true distribution F_i, and with the remaining probability ε is drawn from an arbitrary contamination distribution $G_{i,t}$. The goal is to identify the arm whose true distribution has the highest (or approximately highest) median. The median is the goal rather than the mean since the distributions are not assumed to have finite first moments; and even if they do, the true distributions' means are impossible to estimate from contaminated samples since the contaminations may be arbitrary. A key point is that suboptimality of the arms is based on the quality of the underlying true distributions F_i, not the contaminated distributions $\tilde{F}_{i,t}$ of the observed samples. Note that existing BAI algorithms, fed with samples from $\tilde{F}_{i,t}$, will not necessarily work.

This contamination model nicely fits Pat's problem: samples are usually trustworthy, but sometimes they are completely off and cannot be modeled by a distribution with controlled tails. Additionally, the nature of contamination changes with the respondent, and hence $G_{i,t}$ should be considered as time varying, which completely breaks the usual i.i.d. data assumption. Finally, Pat wants to determine the department with highest *true* median salary, not highest *contaminated* median (or mean) salary.

The contaminated bandit setup also naturally models many other situations that the classical bandit setup cannot, such as any bandit problem where data may be subject to measurement or recording error with some probability. For example, consider the canonical problem of optimal experiment design where we are measuring drug responses, but in a setting where samples can be corrupted or where test results can be incorrectly measured or recorded. Another example is testing new software features, where yet-unfixed bugs may distort responses but will be fixed before release. More scenarios are discussed in Subsection 1.2 when comparing to other models in previous works, and in Subsection 2.1 after the formal definition of the CBAI problem.

Importantly, the CBAI problem is nontrivially harder than the BAI problem since there are no consistent estimators for statistics (including the mean or median) of F_i if the contaminations are allowed to be arbitrary. Under some mild technical assumptions on F_i, the contamination can cause the median of $\tilde{F}_{i,t}$ to be anywhere in an $\Theta(\varepsilon)$-neighborhood of the median of F_i, and hence we can only determine the median of F_i up to some unavoidable estimation bias, U_i, of order ε (see Section 2 for details). This leads us to generalize our study to the more abstract *Partially Identifiable Best Arm Identification* (PIBAI) problem (defined formally in Section 3), which includes CBAI as a special case.

©2019 Jason Altschuler, Victor-Emmanuel Brunel, Alan Malek.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-395.html.

This PIBAI problem can be seen as an active-learning version of the classical problem of estimation under partial identifiability, which has been studied for most of the last century in the econometrics literature (Marschak and Andrews, 1944; Manski, 2009). A canonical problem in this field is trying to learn the age distribution of a population by only asking in which decade each subject was born; clearly the median age can only be learned up to certain unidentifiability regions.

1.1. Our Contributions

To the best of our knowledge, this is the first paper to consider the Best Arm Identification problem with arbitrary contaminations. This contaminated bandit setup models many practical problems that the classical bandit setup cannot. However, developing algorithms for this setup requires overcoming the challenge of partial identifiability of the arms' true distributions. Indeed, it is not hard to show that the adversary's ability to inject arbitrary contaminations can render "similar" underlying distributions indistinguishable, even with access to infinite samples.

We analyze this CBAI problem under three models of the adversary's power: the *oblivious* adversary chooses all contamination distributions a priori; the *prescient* adversary may choose the contamination distributions as a function of all realizations (past and future) of the true rewards and knowledge of when the learner will observe contaminated samples; and the *malicious* adversary may, in addition, correlate when the learner observes contaminated samples with the true rewards. Subsection 2.1 gives formal definitions of these adversarial settings, as well as motivating examples and applications for each.

Our technical contributions can be divided into three parts:

(i) we prove tight, non-asymptotic sample complexity bounds for estimation of the first two robust moments (median and median absolute deviation) from contaminated samples,

(ii) we use the statistical results in (i) to develop efficient algorithms for the CBAI problem and provide sample complexity upper bounds for the fixed-confidence setting, and

(iii) we prove matching information-theoretic lower bounds showing that our algorithms have optimal sample complexity (up to small logarithmic factors).

We elaborate on each of these below.

Contribution (i). These concentration inequalities are the main technical contributions of the paper and may be of independent interest to the robust statistics community. We consider estimating statistics of a single arm from contaminated samples and show that although estimation of standard moments (mean, variance) is impossible, estimation of robust moments (median, median absolute deviation) is possible. Specifically, for each of the three adversarial models, we show that with probability at least $1-\delta$, the empirical median of the contaminated samples lies in a region around the true median of width $U_i + E_{n,\delta}$, where U_i is some unavoidable bias that depends on quantiles of F_i and the power of the adversary, n is the number of samples, and $E_{n,\delta}$ is a confidence-interval term that decreases at the optimal $\sqrt{\frac{\log 1/\delta}{n}}$ rate. Our results neatly capture the effect of the adversary's power by deriving different U_i for each scenario, thereby precisely quantifying the hardness of

the three different adversarial settings. We also present non-asymptotic sample complexity guarantees for estimation of the second robust moment, often called the Median Absolute Deviation (MAD), under all three adversarial settings. The MAD is a robust measure of the spread of F_i and controls the width U_i of the median's unidentifiability region.

Contribution (ii). We show that, surprisingly, several classical BAI algorithms are readily adaptable to the PIBAI problem. This suggests a certain inherent robustness of these classical bandit algorithms. We first present these algorithms for an abstract version of the PIBAI problem so that our algorithmic results may be easily transferrable to other application domains with different statistics of interest or different contamination models. We then combine these general results with our statistical results from (i) to obtain PAC algorithms for CBAI, the problem this paper focuses on. We give fixed-confidence sample complexity guarantees that mirror the sample complexity guarantees for BAI in the classical stochastic multi-armed bandit setup. The main difference is that BAI sample complexities depend on the suboptimality "gaps" $\Delta_i := p_{i^*} - p_i$ between the statistics of the optimal arm i^* and each suboptimal arm i, whereas our CBAI sample complexities depend on the suboptimality "*effective gaps*" $\tilde{\Delta}_i := (p_{i^*} - U_{i^*}) - (p_i + U_i) = \Delta_i - (U_{i^*} + U_i)$, which account for the unavoidable estimation uncertainties in the most pessimistic way. We also show how to apply the MAD estimations results from (i) to obtain guarantees on the quality of the underlying distribution of the selected arm.

Contribution (iii). We prove matching information-theoretic lower bounds (up to a small logarithmic factor) on the sample complexity of CBAI via a reduction to classical lower bounds for the stochastic multi-armed bandit problem. We argue that for CBAI the effective gap $\tilde{\Delta}_i$ is the right analog of the traditional gap since it appears in matching ways in both our upper and lower bounds.

1.2. Previous Work

The Best Arm Identification problem in the fixed-confidence setting has a long history, starting from work by Bechhofer et al. (1968) and Lai and Robbins (1985). Recent interest from the learning theory community was sparked by the seminal paper by Even-Dar et al. (2002), which proposed several algorithms obtaining instance-adaptive sample complexity bounds. Since then, there has been significant work on the algorithmic side (see work by Kalyanakrishnan et al., 2012; Gabillon et al., 2012; Karnin et al., 2013; Jamieson et al., 2013, 2014). Concurrently, a parallel line of work has focused on improving lower bounds, starting with the 2-armed setting (Chernoff, 1972; Anthony and Bartlett, 2009), extending to the multi-armed setting (Mannor and Tsitsiklis, 2004), and, more recently, continuing with more finely tuned lower bounds that include properties of the arm distributions aside from the gaps (Chen and Li, 2015; Kaufmann et al., 2016; Garivier and Kaufmann, 2016).

In the cumulative regret setting, the online learning literature has considered both stochastic bandits with mild tail assumptions (for example, Bubeck et al. (2013) only assumed the existence of a $(1+\varepsilon)$ moment) and algorithms that obtain near-optimal regret guarantees if the environment is stochastic or adversarial (Bubeck and Cesa-Bianchi, 2012). The partial monitoring problem, where the learner only knows the loss up to some subset (Bartók et al., 2014), is also loosely similar to the CBAI problem in the sense that both problems feature partial identification. We note, however, that minimizing cumulative

regret is not a reasonable goal in our contamination setup since the contaminations can be arbitrary and even unbounded. (Furthermore, even when boundedness is assumed, minimizing cumulative regret can still be a poor criteria; see below for concrete examples.)

The existing literature closest to our work studies the BAI problem in settings more general than i.i.d. arms, for example stochastic but non-stationary distributions (Allesiardo et al., 2017; Allesiardo and Féraud, 2017) or arbitrary rewards where each arm converges to a limit (Jamieson and Talwalkar, 2016; Li et al., 2016). However, neither setting fits the contamination model or allows for arbitrary perturbations.

A related contamination model is studied by Seldin and Slivkins (2014); however, they make a boundedness assumption which makes their setup drastically different from ours.[1] Specifically, they consider cumulative regret minimization in a setting where rewards are bounded in $[0, 1]$ and contaminations can be arbitrary. Because of this $[0, 1]$ boundedness, contamination can move the means by at most ε, which is why it is reasonable that they base their algorithms on the contaminated means. However, the best contaminated mean is *not* a reasonable proxy for the best mean without the $[0, 1]$ boundedness assumption. First off, the contaminated distributions may not even have finite first moments. Moreover, when rewards are bounded but only within a large range, the contaminated mean can deviate from the true mean by ε times the size of that range. This can lead to an error bound that is extremely loose—sometimes to the point of being useless for prediction or estimation—compared to the tight bound given by quantiles (see Lemma 1) which our algorithms achieve (see Subsection 4.2).

For instance, consider the adaptive survey example mentioned earlier. There might be a wide range of (true) academic salaries, say between zero and a million dollars. As such, contaminating an ε fraction of the data could move the mean by roughly $10^6 \cdot \varepsilon$; even for a reasonable value of $\varepsilon = 0.1$, this error bound of roughly 10^5 is so large that it may drown out the actual information that the survey was trying to investigate. On the other hand, the median may be moved a significantly smaller amount since the (true) distribution of academic salaries might have reasonably narrow quantiles around the median (for example, being somewhat bell-shaped is sufficient). This performance improvement is intuitively explained by the fact that most academic salaries are not at the extremes of 0 or a million dollars, but rather are fairly regular around the median. An identical phenomenon occurs in the canonical problem of optimal experiment design where we are measuring drug responses but samples can be corrupted or where test results can be incorrectly measured or recorded. Indeed, if the measurements of interests are, say, blood pressure or weight, the values might only be bounded within a large range, but the true distribution might have reasonably narrow quantiles around the median.

We note that the need for robust bandit algorithms is further motivated by the recent work of Jun et al. (2018), which shows that in a similar model, an adversary can make certain classical bandit algorithms perform very poorly by injecting only a small amount of contamination into the observed samples.

This paper also makes connections between several long bodies of work. The contamination model (Huber, 1964) has a long history of more than fifty years in robust statistics; for examples, see work by Hampel (1974); Maronna and Yohai (1976); Rousseeuw and Leroy (2005); Hampel et al. (2011). Contamination models and malicious errors also have a long history in computer science, including the classical work of Valiant (1985); Kearns and Li (1993) and a recent burst of results on algorithms that handle estimation of means and variances (Lai et al., 2016), efficient estimation in high dimensions (Diakonikolas et al., 2018), PCA (Cherapanamjeri et al., 2017), and general learning (Charikar et al., 2017) in the presence of outliers or corrupted data. Finally, the partial identification literature from econometrics also has a rich history (Marschak and Andrews, 1944; Horowitz and Manski, 1995; Manski, 2009; Romano and Shaikh, 2010; Bontemps et al., 2012).

1. Another difference is that our setup and results also work for more powerful types of contaminating adversaries.

1.3. Notation

We denote the Dirac measure at a point $x \in \mathbb{R}$ by δ_x, the Bernoulli distribution with parameter $p \in [0, 1]$ by $\mathrm{Ber}(p)$, and the uniform distribution over an interval $[a, b]$ by $\mathrm{Unif}([a, b])$. The interval $[a - b, a + b]$ is denoted by $[a \pm b]$, the set of non-negative real numbers by $\mathbb{R}_{\geq 0}$, the set of positive integers by $\mathbb{N}$, and the set $\{1, \dots, n\}$ by $[n]$ for $n \in \mathbb{N}$. We abbreviate "with high probability" by "w.h.p." and "cumulative distribution function" by "cdf".

Let F be a cdf. We denote its left and right quantiles, respectively, by $Q_{L,F}(p) := \inf\{x \in \mathbb{R} : F(x) \geq p\}$ and $Q_{R,F}(p) := \inf\{x \in \mathbb{R} : F(x) > p\}$; the need for this technical distinction arises when F is not strictly increasing. We denote the set of medians of F by $m_1(F) := [Q_{L,F}(\frac{1}{2}), Q_{R,F}(\frac{1}{2})]$. When F has a unique median, we overload $m_1(F)$ to be this point rather than a singleton set containing it. For shorthand, we often write $m_1(X)$ for a random variable X to denote $m_1(F)$, where F is the law of X. When F has a unique median, we denote the median absolute deviation (MAD) of F by $m_2(F) := m_1(|X - m_1(F)|)$ where $X \sim F$. When $m_2(F)$ is unique, we define $m_4(F) := m_1(||X - m_1(F)| - m_2(F)|)$. Note that $m_1(F)$, $m_2(F)$, and $m_4(F)$ are robust analogues of centered first (mean), second (variance), and fourth (kurtosis) moments, respectively.

The empirical median of a (possibly random) sequence $x_1, \dots, x_n \in \mathbb{R}$ is denoted by $\hat{m}_1(x_1, \dots, x_n)$: if n is odd, this is the middle value; and if n is even, it is the average of the middle two values. The empirical MAD $\hat{m}_2(x_1, \dots, x_n)$ is then defined as $\hat{m}_1(|x_1 - \hat{m}_1(x_1, \dots, x_n)|, \dots, |x_n - \hat{m}_1(x_1, \dots, x_n)|)$.

1.4. Outline

Section 2 formally defines the CBAI problem and the power of the adversary, describes several motivating applications and examples, and discusses the primary challenge of the problem: partial identifiability. Section 3 presents our algorithmic results for the abstract setting of best arm identification under partial identifiability (the PIBAI problem). We state these algorithms in this general setting so that they may be easily transferrable later to other application domains with different statistics of interest or different contamination models. Section 4 then specializes to the CBAI problem. In order to implement the aforementioned general-purpose PIBAI algorithms for CBAI, concentration results are needed for estimation of medians given contaminated samples. These statistical results are developed in Subsection 4.1, and then used to derive upper bounds on the sample complexity of our CBAI algorithms in Subsection 4.2. Subsection 4.3 gives matching information-theoretic lower bounds on the sample complexity, showing that our algorithms are optimal

(up to small logarithmic factors). This answers the primary question the paper sets out to solve: understanding the complexity of identifying the arm with best median given contaminated samples. Section 5 then turns to our secondary goal: providing guarantees on the distribution of the selected arm. This goal requires additionally estimating the second robust moment. Subsection 5.1 develops the necessary statistical results, which are then used in Subsection 5.2 provide our algorithmic results. Section 6 concludes and discusses several open problems.

2. Formal Setup

Here we formally define the *Contaminated Best Arm Identification problem (CBAI)*. Let $k \geq 2$ be the number of arms, $\varepsilon \in (0, \frac{1}{2})$ be the contamination level, $\{F_i\}_{i\in[k]}$ be the true but unknown distributions, and $\{G_{i,t}\}_{i\in[k],t\in\mathbb{N}}$ be arbitrary contamination distributions. This induces contaminated distributions $\tilde{F}_{i,t}$, samples from which are equal in distribution to $(1 - D_{i,t})Y_{i,t} + D_{i,t}Z_{i,t}$, where $D_{i,t} \sim \text{Ber}(\varepsilon)$, $Y_{i,t} \sim F_i$, and $Z_{i,t} \sim G_{i,t}$. Note that if all of these random variables are independent, then each $\tilde{F}_{i,t}$ is simply equal to the contaminated mixture model $(1-\varepsilon)F_i + \varepsilon G_{i,t}$. However, we generalize by also considering the setting when the $Y_{i,t}, D_{i,t}, Z_{i,t}$ are not all independent, which allows an adversary to further obfuscate samples by adapting the distributions of the $D_{i,t}$ and $Z_{i,t}$ based on the realizations of the $Y_{i,t}$ (that is, by coupling these random variables); see below for details.

At each iteration t, a CBAI algorithm chooses an arm $I_t \in [k]$ to pull and receives a sample $X_{I_t,t}$ distributed according to the corresponding contaminated distribution $\tilde{F}_{I_t,t}$. After T iterations (a possibly random stopping time that the algorithm may choose), the algorithm outputs an arm $\hat{I} \in [k]$. For $\alpha \geq 0$ and $\delta \in (0, 1)$, the algorithm is said to be (α, δ)-*PAC* if, with probability at least $1 - \delta$, $\hat{I}$ has median within $\alpha + U$ of the optimal; that is,

$$\mathbb{P}\left(m_1(F_{\hat{I}}) \geq \max_{i\in[k]} m_1(F_i) - (\alpha + U)\right) \geq 1 - \delta, \tag{1}$$

where U is the unavoidable uncertainty term in median estimation that is induced by partial identifiability (see Subsection 2.2 for a discussion of U, and see Subsection 4.1 for an explicit computation of this quantity). Thus, the goal is to find an algorithm achieving the PAC guarantee in (1) with small expected sample complexity T.

2.1. Power of the Adversary

As is typical in online learning problems, it is important to define the power of the adversary since this affects the complexity of the resulting problem. Interestingly, CBAI is still possible even when we grant the adversary significant power. We consider three settings, presented in increasing order of adversarial power. The key differences between these different types of adversaries are twofold: (1) whether they can choose the contaminated distributions "presciently" based on all other realizations $\{Y_{i,t}, D_{i,t}\}_{i\in[k],t\geq 1}$ both past and future; and (2) whether they can "maliciously" couple the distributions of each $D_{i,t}$ with the corresponding $Y_{i,t}$, subject only to the constraint that the marginals $D_{i,t} \sim \text{Ber}(\varepsilon)$ and $Y_{i,t} \sim F_i$ stay correct.

- **Oblivious adversary.** For all $i \in [k]$, the triples $\{(Y_{i,t}, D_{i,t}, Z_{i,t})\}_{t\geq 1}$ are independent, and for all $t \geq 1$, $Y_{i,t} \sim F_i$, $D_{i,t} \sim \text{Ber}(\varepsilon)$, and $Y_{i,t}$ and $D_{i,t}$ are independent.
- **Prescient adversary.** Same as the oblivious adversary except that the contaminations $Z_{i,t}$ may depend on everything else (i.e. $\{Y_{j,s}, D_{j,s}, Z_{j,s}\}_{j\in[k],s\geq 1}$). Formally, for all $i \in [k]$, the pairs $\{(Y_{i,t}, D_{i,t})\}_{t\geq 1}$ are independent, and for all $t \geq 1$, $Y_{i,t} \sim F_i$, $D_{i,t} \sim \text{Ber}(\varepsilon)$, $Y_{i,t}$ and $D_{i,t}$ are independent, and $Z_{i,t}$ may depend on all $\{Y_{j,s}, D_{j,s}, Z_{j,s}\}_{j\in[k],s\geq 1}$.
- **Malicious adversary.** Same as the prescient adversary except that $Y_{i,t}$ and $D_{i,t}$ do not need to be independent. Formally, for all $i \in [k]$, the pairs $\{(Y_{i,t}, D_{i,t})\}_{t\geq 1}$ are independent, and for all $t \geq 1$, $Y_{i,t} \sim F_i$, $D_{i,t} \sim \text{Ber}(\varepsilon)$, and $Z_{i,t}$ may depend on all $\{Y_{j,s}, D_{j,s}, Z_{j,s}\}_{j\in[k],s\geq 1}$.

We note that for all three of these adversarial settings, we do not require independence between the outputs $(Y_{i,t}, D_{i,t}, Z_{i,t})$ across the arms for each fixed t.

The oblivious adversary is well-motivated, as many real-world problems where contaminations may occur fit naturally into this model. Indeed, this setting encompasses any bandit problem where data may be subject to measurement or recording error with some probability, such as measuring drug responses for clinical trials (Lai and Robbins, 1985), conducting surveys (Martin and Miron, 1992), or testing new software features with yet-unfixed bugs as mentioned in the introduction. The theory we develop extends naturally and with little modification for the prescient and malicious adversaries. Moreover, these different settings allow for the modeling of many other real-world scenarios in which the contamination models are not as simple. For instance, all data may already exist but not yet be released to the learning algorithm. If adversaries have access to all the data from the beginning, they may be able to contaminate samples depending on all the data—this is captured by the prescient setting. The malicious setting includes, for example, scenarios in which adversaries (for example, hackers) can successfully contaminate a given sample depending on the amount of effort they put into it, and the adversaries can try harder to contaminate certain samples depending on their values.

Perhaps surprisingly, we will show that the sample complexity is the same for both oblivious and prescient adversaries. Indeed for these two settings, we will prove our upper bounds for the more powerful setting of prescient adversaries, and our lower bounds for the less powerful setting of oblivious adversaries. We also note that the rate for malicious adversaries is only worse by at most a "factor of 2"; see Section 4 for a precise statement.

2.2. The Challenge of Partial Identifiability of the Median

In the introduction, we emphasized the point that a contaminating adversary can render different underlying distributions of an arm statistically indistinguishable. That is, even with infinite samples, it is impossible to estimate statistics of an arm's true distribution exactly. Consider, for example, the problem of estimating the median of a single arm with true distribution F against an oblivious adversary, which is the weakest of our three adversarial settings. Define S to be the set of all distributions F' for which there exists adversarially chosen distributions G and G' such that $(1 - \varepsilon)F + \varepsilon G = (1 - \varepsilon)F' + \varepsilon G'$. How large is this set S, and in particular, how far can the medians of distributions in S be from the median of F?

The following simple example shows that S is non-trivial (and thus in particular contains more than just F). Let F be the uniform distribution on the interval $[-1,1]$, and let G be the uniform distribution on $[-1-c,-1]\cup[1,1+c]$, where $c=\varepsilon(1-\varepsilon)^{-1}$. The contaminated distribution $\tilde{F}:=(1-\varepsilon)F+\varepsilon G$ is the uniform distribution on $[-1-c,1+c]$. However, for any $p\in[-c,c]$, $\tilde{F}$ is also equal to $(1-\varepsilon)F(\cdot-p)+\varepsilon G_p$, where G_p is the uniform distribution over $[-1-c,1+c]\setminus[-1+p,1+p]$. We conclude that F is statistically indistinguishable from any of the translations $\{F(\cdot-p)\}_{p\in[-c,c]}$. Hence, S is non-trivial and contains distributions with medians at least $\Omega(c)$ away from $m_1(F)$. Even in this simple setting, infinite samples only allow us to identify the median of F at most up to the unidentifiability region $[-c,c]$.

In fact, this toy example captures the correct dependence on ε of how far the median of the contaminated distribution can be shifted, as formalized by the following simple lemma.

Lemma 1 *For any $\varepsilon\in(0,\frac{1}{2})$, any distribution F, and any median $m\in m_1(F)$,*

$$\sup_{\substack{\text{distribution } G,\\ \tilde{m}\in m_1((1-\varepsilon)F+\varepsilon G)}} |\tilde{m}-m| = \max\left\{Q_{R,F}\left(\frac{1}{2(1-\varepsilon)}\right)-m, m-Q_{L,F}\left(\frac{1-2\varepsilon}{2(1-\varepsilon)}\right)\right\}.$$

Proof The fact that the left hand side is no smaller than the right hand side is straightforward: to shift the median to the right (resp. left), let $\{G_n\}_{n\in\mathbb{N}}$ be a sequence of distributions which are Dirac measures at n (resp. $-n$).

We now prove the reverse direction, the "$\leq$" inequality for any fixed distribution G. For shorthand, denote the contaminated distribution by $\tilde{F}:=(1-\varepsilon)F+\varepsilon G$, and denote its left and right medians by $\tilde{m}_L:=Q_{L,\tilde{F}}(\frac{1}{2})$ and $\tilde{m}_R:=Q_{R,\tilde{F}}(\frac{1}{2})$, respectively. Since every median of $\tilde{F}$ lies within $[\tilde{m}_L,\tilde{m}_R]$, it suffices to show that $\tilde{m}_L\geq Q_{L,F}(\frac{1-2\varepsilon}{2(1-\varepsilon)})$ and $\tilde{m}_R\leq Q_{R,F}(\frac{1}{2(1-\varepsilon)})$. We bound $\tilde{m}_L$ presently, as bounding $\tilde{m}_R$ follows by a similar argument or by simply applying the $\tilde{m}_L$ bound to $F(-\cdot)$. By definition of $\tilde{m}_L$, for all $t>0$, $\frac{1}{2}\leq\tilde{F}(\tilde{m}_L+t)=(1-\varepsilon)F(\tilde{m}_L+t)+\varepsilon G(\tilde{m}_L+t)\leq(1-\varepsilon)F(\tilde{m}_L+t)+\varepsilon$, where the last step is because $G(\tilde{m}_L+t)\leq 1$ since G is a distribution. Rearranging yields $F(\tilde{m}_L+t)\geq\frac{1-2\varepsilon}{2(1-\varepsilon)}$, which implies that $\tilde{m}_L\geq Q_{L,F}(\frac{1-2\varepsilon}{2(1-\varepsilon)})$. ■

Finally, we note that for some simple contamination models, partial identifiability in CBAI is not a problem for identifying the arm with the best median. For example, if all arms are contaminated with a common distribution, then although the true medians are only partially identifiable, the contaminated medians are ordered in the same way as the true medians, albeit perhaps not strictly. However, for general (arbitrary) contaminations, such an ordering invariance is clearly not guaranteed.

3. Algorithms for Best Arm Identification Under Partial Identifiability

Our algorithms for CBAI actually work for the more general problem of best arm identification in *partially identified* settings. Specifically, consider any setting where the statistic (for example, the median or mean) which measures the goodness of an arm can be estimated only up to some unavoidable error term due to lack of identifiability. The main result of this section is informally that certain BAI algorithms for the classical stochastic multi-armed

bandit setting can be adapted with little modification to such partially identified settings. Perhaps surprisingly, this suggests a certain innate robustness of these existing classical BAI algorithms.

We present our algorithms in this section for this slightly more abstract problem of *Partially Identifiable Best-Arm-Identification* problem (PIBAI), which we define formally below. This abstraction allows our algorithmic results to later be transferred easily to other variants of the Best Arm Identification problem with different contamination models or different statistics of interest. We will show in Section 4 how to adapt this to the CBAI problem, the main focus of the paper, after first developing the necessary statistical results.

We now formally define the setup of PIBAI. Let $k\geq 2$ be the number of arms. For each arm $i\in[k]$, consider a family of distributions $\mathcal{D}_i=\{D_i(p_i,G)\}_{G\in\mathcal{G}}$ where p_i is a real-valued measure of the quality of arm i and G is a nuisance parameter in some abstract space $\mathcal{G}$. We let $i^*:=\arg\max_{i\in[k]}p_i$ be the best arm. We assume the existence of non-negative unavoidable biases $\{U_i\}_{i\in[k]}$ satisfying

(i) even from infinitely many independent samples $X_t, t=1,2,\ldots$ with $X_t\sim D_i(p_i,G_t)$ for some unknown, possibly varying $G_t\in\mathcal{G}$ $(t\geq 1)$, it is impossible to estimate p_i more precisely than the region $[p_i\pm U_i]$, and

(ii) there exists some estimator that, for any $\alpha>0$ and $\delta\in(0,1)$, uses $n_{\alpha,\delta}=O(\alpha^{-2}\log\frac{1}{\delta})$ i.i.d. samples[2] from D_i to output an estimate $\hat{p}_i$ satisfying

$$\mathbb{P}\Big(\hat{p}_i\in[p_i\pm(U_i+\alpha)]\Big)\geq 1-\delta. \tag{2}$$

The PIBAI problem is then precisely the standard fixed-confidence stochastic bandit problem using these distributions where in each iteration t, the algorithm chooses an arm $I_t\in[k]$ and receives a sample from $D_{I_t}(p_{I_t},G_t)$ for some unknown $G_t\in\mathcal{G}$.

By the partial identifiability property (i), it is clear that even given infinite samples, it is impossible to distinguish between the optimal arm i^* and any suboptimal arm $i\neq i^*$ satisfying $p_i+U_i\geq p_{i^*}-U_{i^*}$. Therefore, we assume henceforth the statistically possible setting in which the *effective gaps* $\tilde{\Delta}_i:=(p_{i^*}-U_{i^*})-(p_i+U_i)$ are strictly positive for each suboptimal arm $i\neq i^*$.

For any $\alpha\geq 0$, arm i is said to be *α-suboptimal* if $\tilde{\Delta}_i\leq\alpha$. Moreover, for any $\alpha\geq 0$ and $\delta\in(0,1)$, a PIBAI algorithm is said to be *(α,δ)-PAC* if it outputs an arm $\hat{I}$ that is α-suboptimal with probability at least $1-\delta$. That is,

$$\mathbb{P}\big(\tilde{\Delta}_{\hat{I}}\leq\alpha\big)\geq 1-\delta,$$

where the above probability is taken over the possible randomness of the samples, the estimator from (ii), and the PIBAI algorithm.

2. Using the same techniques as presented in this paper, one can also consider PIBAI for general $n_{\alpha,\delta}\neq O(\alpha^{-2}\log\frac{1}{\delta})$ and compute the resulting sample complexities. However, for simplicity of presentation, we assume that $n_{\alpha,\delta}=O(\alpha^{-2}\log\frac{1}{\delta})$ since anyways this is the natural (and optimal) quantity for many estimation problems such as estimating a median from contaminated samples (see Corollaries 14 and 15), or using Chernoff bounds to estimate the mean of a $[0,1]$-supported distribution for classical stochastic MAB, etc.

for $i \in [k]$ **do**
  Sample arm i for $n_{\alpha/2,\delta/k}$ times and produce estimate $\hat{p}_i$
end
Output $\hat{I} := \max_{i\in[k]} \hat{p}_i$.

Algorithm 1: Simple uniform exploration algorithm for PIBAI.

We now present algorithms for the PIBAI problem. First, in Subsection 3.1, we present a simple algorithm that performs uniform exploration among all arms. Next, in Subsection 3.2, we obtain more refined, instance-adaptive sample complexity bounds by adapting the Successive Elimination algorithm of (Even-Dar et al., 2006). All algorithms use as a blackbox an estimator satisfying the above property (ii) of the PIBAI problem.

3.1. Simple Algorithm

A simple (α, δ)-PAC PIBAI algorithm is the following: pull each of the k arms $n_{\alpha/2,\delta/k}$ times to create estimates $\hat{p}_i$ and output the arm $\hat{I} := \max_{i\in[k]} \hat{p}_i$ with the highest estimate. Pseudocode is given in Algorithm 1.

Theorem 2 *For any $\alpha > 0$ and $\delta \in (0,1)$, Algorithm 1 is an (α,δ)-PAC* PIBAI *algorithm with sample complexity $O(kn_{\alpha/2,\delta/k}) = O\left(\frac{k}{\alpha^2}\log\frac{k}{\delta}\right)$.*

Proof By (2) and a union bound, we have that with probability at least $1-\delta$, all estimates $\hat{p}_i \in [p_i \pm (U_i + \frac{\alpha}{2})]$. Whenever this occurs,

$$\tilde{\Delta}_{\hat{I}} = (p_{i^*} - U_{i^*}) - (p_{\hat{I}} + U_{\hat{I}}) \le (\hat{p}_{i^*} + \tfrac{\alpha}{2}) - (\hat{p}_{\hat{I}} - \tfrac{\alpha}{2}) \le \alpha,$$

implying that $\hat{I}$ is α-suboptimal. The sample complexity bound is clear. ■

3.2. Instance-adaptive Algorithms

The sample complexity of the simple Algorithm 1 is not adaptive to the difficulty of the actual instance: an arm is sampled $n_{\alpha/2,\delta/k}$ times even if it is far from α-suboptimal (meaning that $\tilde{\Delta}_i \gg \alpha$) and could potentially be eliminated much more quickly. In this subsection, we obtain such an instance-adaptive sample complexity by modifying the Successive Elimination algorithm of (Even-Dar et al., 2006), which was originally designed for the classical multi-armed bandit problem.

Pseudocode is given in Algorithm 2. At each round r, Algorithm 2 gets a single new sample from each remaining arm in order to update its estimate $\hat{p}_{i,r}$. Then, for the next round $r+1$, it only keeps the arms i whose estimates $\hat{p}_{i,r}$ are α-close to the best estimate, where the threshold α is updated at each round. The algorithm and analysis are almost identical to the original. As such, proof details are deferred to Appendix C. The main difference is that in the proof of correctness, we show the event $\{|\hat{p}_{i,r} - p_i| \le U_i + \alpha_{r,6\delta/(\pi^2 k r^2)},\ \forall r \in [R],\ \forall i \in S_r\}$ occurs with probability at least $1-\delta$. This ensures that in each round r, each estimate $\hat{p}_{i,r}$ is accurate enough to use for the elimination step.

$S \leftarrow [k]$, $r \leftarrow 1$
while $|S| > 1$ **do**
  Sample each arm $i \in S$ once and produce $\hat{p}_{i,r}$ from all r past samples of it
  $S \leftarrow \{i \in S : \hat{p}_{i,r} \ge \max_{j\in S} \hat{p}_{j,r} - 2\alpha_{r,6\delta/(\pi^2 k r^2)}\}$
  $r \leftarrow r+1$
end
Output the only arm left in S

Algorithm 2: Adaptation of Successive Elimination algorithm for PIBAI. Here, $\alpha_{r,\delta} := \sqrt{\frac{c\log\frac{1}{\delta}}{r}}$, where c is a universal constant satisfying $n_{\alpha,\delta} \le c\alpha^{-2}\log\frac{1}{\delta}$ (see Equation 2).

Note that Algorithm 2 returns the optimal arm w.h.p. (without knowing the smallest effective gap), unlike Algorithm 1 above which only returns a near-optimal arm w.h.p.

Theorem 3 *Let $\delta \in (0,1)$. With probability at least $1-\delta$, Algorithm 2 outputs the optimal arm after using at most $O\left(\sum_{i\neq i^*} \frac{1}{\tilde{\Delta}_i^2}\log\left(\frac{k}{\delta\tilde{\Delta}_i}\right)\right)$ samples.*

Moreover, as noted in Remark 9 of (Even-Dar et al., 2006), Algorithm 2 is easily modified (by simply terminating early) to be an (α,δ)-PAC algorithm with sample complexity

$$O\left(\frac{N_\alpha}{\alpha^2}\log\left(\frac{N_\alpha}{\delta}\right) + \sum_{i\in[k]:\,\tilde{\Delta}_i>\alpha} \frac{1}{\tilde{\Delta}_i^2}\log\left(\frac{k}{\delta\tilde{\Delta}_i}\right)\right),$$

where N_α is the number of α-suboptimal arms with $\tilde{\Delta}_i \le \alpha$.

Remark 4 *In the multi-armed bandit literature, the sample complexity of the Successive Elimination algorithm (tight up to a logarithmic factor) was improved upon by (Karnin et al., 2013)'s Exponential-Gap Elimination (EGE) algorithm (tight up to a doubly logarithmic factor). A natural idea is to analogously improve upon the* PIBAI *guarantee in Theorem 3 by adapting the EGE algorithm. However, this approach does not work. The same holds for adapting the Median Elimination algorithm of (Even-Dar et al., 2006) and the PRISM algorithm of (Jamieson et al., 2013). The reason for the inadaptability of these algorithms—in contrast to the easy adaptability of the Successive Elimination algorithm above—is that these algorithms heavily rely on the "additive property of suboptimality" for* BAI*:*

> *"If arm i has Δ_i suboptimality gap w.r.t. the optimal arm i^*, and if arm j has $\Delta_j^{(i)}$ suboptimality gap w.r.t. arm i, then arm j has suboptimality gap $\Delta_j = \Delta_i + \Delta_j^{(i)}$ w.r.t. the optimal arm i^*."*

The critical point is that PIBAI *does not have this property since errors propagate from adding the uncertainties U_i in the suboptimality gaps: that is, $(p_{i^*} - U_{i^*}) - (p_j + U_j) \neq [(p_{i^*} - U_{i^*}) - (p_i + U_i)] + [(p_i - U_i) - (p_j + U_j)]$. Because of this, in order to return an α-suboptimal arm for* PIBAI*, we must ensure that all arms that are more than α-suboptimal*

are eliminated before the optimal arm is eliminated (if it ever is). This is in contrast to BAI *in the classical multi-armed bandit setup: there, it is not problematic if the best (or even currently best) arm is eliminated at round r, so long as the best arm in consecutive rounds r and $r+1$ changes at most by a small amount.*[3] *We stress that this nuance is not merely a technicality but actually fundamental to the correctness proofs for* PIBAI.

4. Finding the Best Median

The previous section provided algorithms for the general PIBAI problem; in this section, these algorithms are adapted to the special case of the CBAI problem, which is the focus of the paper. This requires an estimator of the median from contaminated samples that concentrates at an exponential rate[4]. Subsection 4.1 develops these statistical results, which are of potential independent interest to the robust statistics community. Subsection 4.2 then combines these results with the results of the previous section to conclude algorithms for CBAI, and gives upper bounds on their sample complexity. Finally, Subsection 4.3 gives information-theoretic lower bounds showing that the sample complexities of these algorithms are optimal (up to a small logarithmic factor).

4.1. Estimation of Median from Contaminated Samples

In this subsection, we develop the statistical results needed to obtain algorithms for the CBAI problem. Specifically, we obtain tight, non-asymptotic sample-complexity bounds for median estimation from contaminated samples. We do this for all three adversarial models for the contamination (oblivious, prescient, and malicious). These results may be of independent interest to the robust statistics community.

The subsection is organized as follows. First, Subsection 4.1.1 studies the concentration of the empirical median under the most general distributional assumptions and provides, for all three adversarial settings, tight upper and lower bounds on the sample complexity of median estimation from contaminated samples. Next, in Subsection 4.1.2 we obtain more algorithmically useful guarantees by specializing these results to a family of distributions where the cdfs increase at least linearly in a neighborhood of the median, a very common assumption in the robust estimation literature. For this family of distributions, we explicitly compute for all three adversarial settings the unavoidable bias terms mentioned in Section 2 for median estimation from contaminated samples. It is worth emphasizing that these results precisely quantify the effect of the three different adversarial strengths on the complexity of the problem of median estimation in the contamination model.

Throughout this subsection, we will only consider a single arm, and its true distribution will be denoted by F. For brevity of the main text, all proofs are deferred to Appendix B.

3. In particular, at most $2^{-r}\alpha$ for most of these aforementioned algorithms, since this implies that the final arm $\hat{I}$ is at most $\sum_{r=1}^{\infty} 2^{-r}\alpha = \alpha$-suboptimal.
4. For the formal statement, see (ii) in the definition of PIBAI in Section 3.

4.1.1. General Concentration Results for Median Estimation

Lemma 1 implies that some control of the quantiles of F is necessary to obtain guarantees for estimation of $m_1(F)$ from contaminated samples. We begin by considering F in the following family of distributions, which we stress makes the most general such assumption.

Definition 5 *For any $\bar{t} \in (0, \frac{1}{2})$ and any non-decreasing function $R : [0, \bar{t}] \to \mathbb{R}_{\geq 0}$, define $\mathcal{H}_{\bar{t},R}$ to be the family of all distributions F satisfying*

$$R(t) \geq \max\left\{ Q_{R,F}\left(\frac{1}{2}+t\right) - m, m - Q_{L,F}\left(\frac{1}{2}-t\right)\right\} \tag{3}$$

for any $t \in [0, \bar{t}]$ and any median $m \in m_1(F)$.

In words, the function R dictates the the maximal deviation from the median that a cdf F can have for all quantiles in a small neighborhood $[\frac{1}{2} - \bar{t}, \frac{1}{2} + \bar{t}]$ around the median. Since R is an arbitrary non-decreasing function, Definition 5 gives the most general possible bound on these quantiles.

Note that if $\varepsilon > \bar{\varepsilon}(\bar{t}) := \frac{2\bar{t}}{1+2\bar{t}}$, it is impossible to control the deviation of the contaminated median from the true median for $F \in \mathcal{H}_{\bar{t},R}$. Indeed, the requirement $\varepsilon \leq \bar{\varepsilon}(\bar{t}) := \frac{2\bar{t}}{1+2\bar{t}}$ is equivalent to $\bar{t} \geq \frac{\varepsilon}{2(1-\varepsilon)}$, which, by Lemma 1, is the largest possible deviation from the $\frac{1}{2}$-quantile.

Combining Lemma 1 with the definition of R in Definition 5 immediately yields the following tight bound on how far the contaminated median can be moved from the true median for any $F \in \mathcal{H}_{\bar{t},R}$.

Corollary 6 *For any $\bar{t} \in (0, \frac{1}{2})$, any $\varepsilon \in (0, \bar{\varepsilon}(\bar{t}))$, and any non-decreasing function $R : [0, \bar{t}] \to \mathbb{R}_{\geq 0}$,*

$$\sup_{\substack{F \in \mathcal{H}_{\bar{t},R},\, m \in m_1(F),\\ \text{distribution } G,\, \tilde{m} \in m_1((1-\varepsilon)F + \varepsilon G)}} |\tilde{m} - m| = R\left(\frac{\varepsilon}{2(1-\varepsilon)}\right)$$

We now turn to estimation results. At this point it is necessary to distinguish between the three adversarial settings for the contamination. We begin with guarantees for the oblivious and prescient adversarial settings, which—somewhat surprisingly—turn out to have the same rates. In particular, we show that with probability at least $1-\delta$, the estimation error is bounded above by $R(\frac{\varepsilon}{2(1-\varepsilon)} + O(\sqrt{\frac{\log 1/\delta}{n}}))$. Note that if R is Lipschitz, then this quantity is bounded above by the unavoidable uncertainty term $R\left(\frac{\varepsilon}{2(1-\varepsilon)}\right)$ (see Corollary 6) plus an error term that decays quickly with $n^{-1/2}$ rate and has sub-Gaussian tails in δ. This confidence-interval term is of optimal order since it is tight even for estimating the mean (and thus median) of a Gaussian random variable with known unit variance from *uncontaminated* samples.

Lemma 7 *Let $\bar{t} \in (0, \frac{1}{2})$, $\varepsilon \in (0, \bar{\varepsilon}(\bar{t}))$, and $F \in \mathcal{H}_{\bar{t},R}$. Let $Y_i \sim F$ and $D_i \sim Ber(\varepsilon)$, for $i \in [n]$, all be drawn independently. Let $\{Z_i\}_{i\in[n]}$ be arbitrary random variables possibly*

depending on $\{Y_i, D_i\}_{i\in[n]}$*, and define* $X_i = (1-D_i)Y_i + D_iZ_i$*. Then, for any confidence level* $\delta \in (0,1)$ *and number of samples* $n \geq 2\left(\bar{t} - \frac{\varepsilon}{2(1-\varepsilon)}\right)^{-2}\log\frac{2}{\delta}$ *, we have*

$$\mathbb{P}\left(\sup_{m\in m_1(F)} |\hat{m}_1(X_1,\dots,X_n) - m| \leq R\left(\frac{\varepsilon}{2(1-\varepsilon)} + \sqrt{\frac{2\log(2/\delta)}{n}}\right)\right) \geq 1-\delta.$$

Note that because the class $\mathcal{H}_{\bar{t},R}$ only assumes control on the $[\frac{1}{2}\pm\bar{t}]$ quantiles, the minimum sample complexity n must grow as $\bar{t}$ approaches $\frac{\varepsilon}{2(1-\varepsilon)}$, since by Lemma 1 this is the largest quantile deviation that the contaminated median can be moved from the true median.

We now turn to malicious adversaries and derive analogous tight, non-asymptotic sample complexity bounds for median estimation from contaminated samples. We show that it possible to obtain estimation accuracy of $R(\varepsilon)$ in this malicious adversarial setting (Lemma 8). Note that this is weaker than the the accuracy of $R(\frac{\varepsilon}{2(1-\varepsilon)})$ obtained above against oblivious and prescient adversaries. However, we also show that this dependency is tight and unavoidable (Lemma 9). Moreover, the $O(\sqrt{\frac{\log 1/\delta}{n}})$ error term in Lemma 8 is tight for the same reason as it was in Lemma 7; see the discussion there. Finally, we remark that the upper bound on ε and the lower bound on the sample complexity n in Lemma 8 are exactly the analogues of the corresponding bounds in Lemma 7; the only difference is that malicious adversaries can force the contaminated distributions to have medians at roughly $F^{-1}(\frac{1}{2}\pm\varepsilon)$, resulting in our need for control of these further quantiles.

Lemma 8 *Let* $\bar{t} \in (0,\frac{1}{2})$*,* $\varepsilon \in (0,\bar{t})$*, and* $F \in \mathcal{H}_{\bar{t},R}$*. Let* (Y_i, D_i)*, for* $i \in [n]$*, be drawn independently with marginals* $Y_i \sim F$ *and* $D_i \sim Ber(\varepsilon)$*. Let* $\{Z_i\}_{i\in[n]}$ *be arbitrary random variables possibly depending on* $\{Y_i, D_i\}_{i\in[n]}$*, and define* $X_i = (1-D_i)Y_i + D_iZ_i$*. Then for any confidence level* $\delta \in (0,1)$ *and number of samples* $n \geq 2(\bar{t}-\varepsilon)^{-2}\log\frac{3}{\delta}$*,*

$$\mathbb{P}\left(\sup_{m\in m_1(F)} |\hat{m}_1(X_1,\dots,X_n) - m| \leq R\left(\varepsilon + \sqrt{\frac{2\log(3/\delta)}{n}}\right)\right) \geq 1-\delta.$$

Lemma 9 *Let* $\bar{t} \in (0,\frac{1}{2})$*,* $\varepsilon \in (0,\bar{t})$*,* $\delta \in (0,1)$*,* $n \geq \frac{1}{2}\varepsilon^{-2}\log\frac{1}{\delta}$*, and* $R : [0,\bar{t}] \to \mathbb{R}_{\geq 0}$ *be any strictly increasing function. Then there exists a distribution* $F \in \mathcal{H}_{\bar{t},R}$ *and a joint distribution on* (D,Y,Z) *with marginals* $D \sim Ber(\varepsilon)$ *and* $Y \sim F$*, such that* F *has unique median and*

$$\mathbb{P}\left(|\hat{m}_1(X_1,\dots,X_n) - m_1(F)| \geq R\left(\varepsilon - \sqrt{\frac{\log(1/\delta)}{2n}}\right)\right) \geq 1-\delta,$$

where each $\{(D_i,Y_i,Z_i)\}_{i\in[n]}$ *is drawn independently from the joint distribution, and* $X_i := (1-D_i)Y_i + D_iZ_i$*.*

4.1.2. Improved Concentration under Quantile Control

Note that if a cdf F changes little around its median, then the neighboring quantiles are harder to distinguish from each other, and thus it is more difficult to estimate the median. In order to obtain more algorithmically useful rates for median estimation from contaminated samples, we now introduce a more specific class of cdfs F that increase at least linearly in a

neighborhood around the median. This ensures that F is not "too flat" in this neighborhood, which we stress is a very standard assumption for median estimation.

Definition 10 *For any* $\bar{t} \in (0,\frac{1}{2})$ *and* $B > 0$*, let* $\mathcal{F}_{\bar{t},B}$ *be the family of distributions* F *that have a unique median and satisfy*

$$|F(x_1) - F(x_2)| \geq \frac{1}{Bm_2(F)}|x_1 - x_2| \tag{4}$$

for all $x_1, x_2 \in \left[Q_{L,F}(\frac{1}{2}-\bar{t}), Q_{R,F}(\frac{1}{2}+\bar{t})\right]$*.*

We make a few remarks about the definition. i) Requiring the right-hand side of Equation 4 to scale inversely in the median absolute deviation (MAD) $m_2(F)$ ensures closure of $\mathcal{F}_{\bar{t},B}$ under scaling; see below. We also mention that $m_2(F)$ is a robust measure of the spread of F (it is the "median moment" analogue of variance), and controls the width of the median's unidentifiability region. ii) If $F \in \mathcal{F}_{\bar{t},B}$, then $F \in \mathcal{H}_{\bar{t},R}$ for $R(t) = Bm_2(F)t$. iii) Distributions in $\mathcal{F}_{\bar{t},B}$ are not required to have densities, nor even be continuous. iv) The family $\mathcal{F}_{\bar{t},B}$ has many natural and expected properties, such as closure under scaling and translation. These properties are gathered in Lemma 26, which is deferred to Appendix A for brevity of the main text.

Remark 11 (Examples) *Most common distributions belong to* $\mathcal{F}_{\bar{t},B}$ *for some values of the parameters* $\bar{t}$ *and* B*. Moreover, by Lemma 26, if a distribution is in* $\mathcal{F}_{\bar{t},B}$*, then all scaled and translated versions are as well. A short list includes: i) the Gaussian distribution, for any* $\bar{t} \in (0,\frac{1}{2})$ *and any* $B \geq \frac{q_{3/4}}{\phi(q_{1/2+\bar{t}})}$*, where* ϕ *is the standard Gaussian density and* q_α *is the corresponding* α*-quantile; ii) the uniform distribution on any interval, for any* $\bar{t} \in (0,\frac{1}{2})$ *and any* $B \geq 4$*; and iii) any distribution* F *with positive density* F'*, for any* $\bar{t} \in (0,\frac{1}{2})$ *and any* $B \geq \left(m_2(F)\min_{x\in[Q_{L,F}(\frac{1}{2}-\bar{t}),Q_{R,F}(\frac{1}{2}+\bar{t})]} F'(x)\right)^{-1}$*.*

We now apply the estimation results for $\mathcal{H}_{\bar{t},R}$ to estimation for $\mathcal{F}_{\bar{t},B}$. By Lemma 26, whenever $F \in \mathcal{F}_{\bar{t},B}$, then $F \in \mathcal{H}_{\bar{t},R}$ for $R(t) = tBm_2(F)$. Corollary 6 immediately yields a bound on the size of the contamination region in terms of the quantity

$$U_{\varepsilon,B,m_2} := Bm_2\frac{\varepsilon}{2(1-\varepsilon)}.$$

Corollary 12 *For any* $\bar{t} \in (0,\frac{1}{2})$*, any* $\varepsilon \in (0,\bar{\varepsilon}(\bar{t}))$*, and any* $F \in \mathcal{F}_{\bar{t},B}$*,*

$$\sup_{\substack{\text{distribution } G,\\ \tilde{m}\in m_1((1-\varepsilon)F+\varepsilon G)}} |\tilde{m} - m_1(F)| \leq U_{\varepsilon,B,m_2(F)}.$$

Remark 13 (Tightness of Corollary 12) *For any* $a > 0$*, let* F *be the uniform distribution over* $[0,a]$*. Then* $m_1(F) = \frac{a}{2}$ *and* $m_2(F) = \frac{a}{4}$*, and thus* $F \in \mathcal{F}_{\bar{t},B}$ *for any* $\bar{t} \in (0,\frac{1}{2})$ *and* $B = 4$*. Now for any* $\varepsilon \in (0,\bar{\varepsilon}(\bar{t})) \subset (0,\frac{1}{2})$*, let* G *be the uniform distribution over*

$[a, \frac{a}{1-\varepsilon}]$. *Then* $\tilde{F} = (1-\varepsilon)F + \varepsilon G$ *is the uniform distribution over* $[0, \frac{a}{1-\varepsilon}]$ *and has median* $m_1(\tilde{F}) = \frac{a}{2} + \frac{\varepsilon}{2(1-\varepsilon)}a = m_1(F) + U_{\varepsilon,B,m_2(F)}$. *An identical argument on* $F(-\cdot)$ *and* $G(-\cdot)$ *yields an example where the median decreases by* $-U_{\varepsilon,B,m_2(F)}$, *so the bound in Corollary 12 is tight.*

Now we turn to median estimation bounds for $\mathcal{F}_{\bar{t},B}$. Since $R(t) = tBm_2(F)$ is clearly Lipschitz, the discussion preceding Lemmas 7 and 8 about the optimality of the following error bounds applies. That is, the error decomposes into the sum of the unavoidable uncertainty term $U_{\varepsilon,B,m_2(F)}$ plus a confidence-interval term decaying at a $n^{-1/2}$ rate and with sub-Gaussian tails in δ, both of which are optimal. The $\bar{t}$ terms in the sample complexity bounds appear for the analogous reasons as in Lemma 7; see the discussion there. For brevity, we omit the proofs of the following corollaries since they follow immediately from Lemmas 7, 8, and 26.

Corollary 14 *Consider the same setup as in Lemma 7, except with the added restriction that* $F \in \mathcal{F}_{\bar{t},B}$. *Then, for any confidence level* $\delta \in (0,1)$, *error level* $E > 0$, *and number of samples* $n \geq 2\max\left(\frac{B^2 m_2^2(F)}{E^2}, \left(\bar{t} - \frac{\varepsilon}{2(1-\varepsilon)}\right)^{-2}\right)\log\frac{2}{\delta}$, *we have*

$$\mathbb{P}\left(|\hat{m}_1(X_1,\dots,X_n) - m_1(F)| \leq U_{\varepsilon,B,m_2(F)} + E\right) \geq 1-\delta.$$

Corollary 15 *Consider the same setup as in Lemma 8, except with the added restriction that* $F \in \mathcal{F}_{\bar{t},B}$. *Then for any confidence level* $\delta \in (0,1)$, *error level* $E > 0$, *and number of samples* $n \geq 2\max\left(\frac{B^2 m_2^2(F)}{E^2}, (\bar{t}-\varepsilon)^{-2}\right)\log\frac{3}{\delta}$,

$$\mathbb{P}\left(|\hat{m}_1(X_1,\dots,X_n) - m_1(F)| \leq U^{(\text{MALICIOUS})}_{\varepsilon,B,m_2(F)} + E\right) \geq 1-\delta.$$

In Corollary 15 above, we have defined

$$U^{(\text{MALICIOUS})}_{\varepsilon,B,m_2} := Bm_2\varepsilon,$$

which is a tight bound on the uncertainty in median estimation that a malicious adversary can induce (see Lemma 9).

4.2. Algorithms

In this subsection, we apply the algorithms developed in Section 3 for the general problem of PIBAI to the special case of CBAI. In order to implement the estimator required by part (ii) of the definition of PIBAI (see Section 3), we make use of the guarantees proved in Subsection 4.1 for median estimation from contaminated samples. Nearly matching information-theoretic lower bounds are provided in the following subsection and show that the algorithms below have optimal sample complexity (up to a small logarithmic factor).

We will only assume that the arms' distributions are in $\mathcal{F}_{\bar{t},B}$ and have robust second moments uniformly bounded[5] by some $\bar{m}_2$; this will allow us to perform median estimation

5. It is typically necessary to have bounds on higher order moments in order to control the error of estimating lower order moments, see e.g. (Bubeck et al., 2013).

from contaminated samples. For simplicity, the parameters of this family are assumed to be known to the algorithm beforehand[6]. Note, however, that it is not necessary to know ε, but only an upper bound ε_0 on ε; indeed ε can be replaced with ε_0 in all our algorithms. It is very standard in the robust statistics literature to assume knowledge of such an upper bound ε_0 on the amount of contamination. The parameter $\bar{t}$ can then be arbitrarily chosen by the algorithms, so long as it is bigger than $\varepsilon_0/(2(1-\varepsilon_0))$ (or ε_0 in the malicious case) and smaller than $1/2$.

The sample complexities we prove below are in terms of the *effective gaps* $\tilde{\Delta}_i := (m_1(F_{i^*}) - U_{i^*}) - (m_1(F_i) + U_i)$ of the suboptimal arms $i \neq i^*$, where $i^* := \arg\max_{i\in[k]} m_1(F_i)$ is the arm with the highest median. Here U_i is the unavoidable uncertainty term for median estimation from contaminated samples under the given adversarial setting. In Subsection 4.1, we explicitly computed $U_i = U_{\varepsilon,B,m_2(F_i)}$ for oblivious and prescient adversaries (Corollary 14) and $U_i = U^{(\text{MALICIOUS})}_{\varepsilon,B,m_2(F_i)}$ for malicious adversaries (Corollary 15). We emphasize that the strength of the adversarial setting is encapsulated in the corresponding U_i.

First, we discuss how the simple Algorithm 1 yields an (α,δ)-PAC algorithm for CBAI without modification. In this setting, $\hat{p}_i$ is the empirical median of the payoffs of arm i. The quantity $n_{\alpha,\delta}$ is equal to $2\max(\frac{B^2\bar{m}_2^2}{\alpha^2}, (\bar{t} - \frac{\varepsilon}{2(1-\varepsilon)})^{-2})\log\frac{2}{\delta}$ against oblivious and prescient adversaries (Corollary 14) and is equal to $2\max(\frac{B^2\bar{m}_2^2}{\alpha^2}, (\bar{t}-\varepsilon)^{-2})\log\frac{3}{\delta}$ against malicious adversaries (Corollary 15). Theorem 2 along with the observation that the constant term in the sample complexity $n_{\alpha,\delta}$ only introduces a term that is negligible in the overall sample complexity of Algorithm 1, immediately yields the following result.

Theorem 16 *Let* $F_i \in \mathcal{F}_{\bar{t},B}$ *with* $m_2(F) \leq \bar{m}_2$ *for each arm* $i \in [k]$, *and let the adversary be oblivious, prescient, or malicious. For any* $\alpha > 0$ *and* $\delta \in (0,1)$, *Algorithm 1 is an* (α,δ)*-PAC* CBAI *algorithm with sample complexity* $O(kn_{\alpha/2,\delta/k}) = O\left(\frac{k}{\alpha^2}\log\frac{k}{\delta}\right)$.

Algorithm 1 does not adapt to the difficulty of the problem instance. A natural approach is to try to apply the Successive Elimination Algorithm to BAI directly. Unfortunately, this algorithm needs concentration inequalities even for small sample sizes, whereas our guarantees are only valid when the sample size is greater than a certain threshold. Hence, we slightly modify the Successive Elimination Algorithm to obtain additional samples in an initial exploration phase; the number of samples required in this initial exploration phase is dictated by our median estimation results form Subsection 4.1. We similarly modify later rounds to take additional samples in order to produce updated median estimates at the desired certainty levels. Pseudocode is given in Algorithm 3.

6. The classes of distributions that we define have two parameters that would be hard or even impossible to estimate in practice, namely B and $\bar{m}_2$. In classical setups in which algorithms need to learn means instead of medians, it is common to assume that there is a second moment σ^2 which is known, or at least bounded by a known constant. This is because in practice σ^2 is hard to estimate from the data. Note that if the distributions F_i are known to have bounded support, $\bar{m}_2$ can be chosen as the size of the domain. It seems that knowing B is the price to pay to go from the classical to the robust setup. Indeed, even in the non-contaminated case, it is well known that the median of a distribution can be estimated at a parametric rate only when the distribution is not too flat around its median. In that case, the estimation rate depends on how "*non-flat*" the distribution is, which is exactly $Bm_2(F)$ in the present setup. In a parametric setup, i.e., when the distributions of the arms $F_i, i = 1,\dots,K$ are known up to some location and/or scale parameters, B would be known since it is translation and scale invariant (see Lemma 26).

$S \leftarrow [k]$, $r \leftarrow 1$
Sample each arm $\lceil N \log(\frac{\pi^2 k}{2\delta}) \rceil$ times
while $|S| > 1$ **do**
  Sample each arm $i \in S$ for $1 + \lceil 2N \log(\frac{r+1}{r}) \rceil$ times and produce $\hat{p}_{i,r}$ from all past samples
  $S \leftarrow \{i \in S : \hat{p}_{i,r} \geq \max_{j \in S} \hat{p}_{j,r} - 2\alpha_{r,6\delta/(\pi^2 k r^2)}\}$
  $r \leftarrow r + 1$
end
Output the only arm left in S

Algorithm 3: Adaptation of successive elimination algorithm (see Algorithm 2) for CBAI. For oblivious and prescient adversaries, $N := 2(\bar{t} - \frac{\varepsilon}{2(1-\varepsilon)})^{-2}$; and for malicious adversaries $N := 2(\bar{t} - \varepsilon)^{-2}$. In all settings, $\alpha_{r,\delta} := \sqrt{\frac{2B\bar{m}_2^2 \log \frac{3}{\delta}}{r}}$.

Theorem 17 *Let $\delta \in (0,1)$, let $F_i \in \mathcal{F}_{\bar{t},B}$ with $m_2(F) \leq \bar{m}_2$ for each arm $i \in [k]$, and let the adversary be either oblivious, prescient, or malicious. With probability at least $1-\delta$, Algorithm 3 outputs the optimal arm after using at most $\tilde{O}\left(\sum_{i \neq i^*} \left(\frac{1}{\tilde{\Delta}_i^2} + N\right) \log\left(\frac{k}{\delta \tilde{\Delta}_i}\right)\right)$ samples.*

The proof follows from Theorem 3 and is deferred to Appendix C. Like Algorithm 2, Algorithm 3 can be stopped earlier in order to obtain an (α, δ)-PAC guarantee.

4.3. Lower Bounds

Here we provide information-theoretic lower bounds on the sample complexity of the CBAI problem that match, up to small logarithmic factors, the algorithmic upper bounds proved above in Subsection 4.2 for each of the three adversarial settings. The main insight is that we can reduce hard instances of the BAI problem to hard instances of the CBAI problem. In this way, we can leverage the sophisticated lower bounds already developed in the classical multi-armed bandit literature.

Let $i^* := \arg\max_{i \in [k]} m_1(F_i)$ denote the optimal arm. As in Section 3, $\tilde{\Delta}_i := (m_1(F_{i^*}) - U_{i^*}) - (m_1(F_i) + U_i)$ denotes the effective gap for suboptimal arms $i \neq i^*$, and U_i denotes the unavoidable uncertainty term for median estimation. We emphasize that since the power of the adversary is encapsulated in the U_i, and therefore also the effective gaps $\tilde{\Delta}_i$, we can address all three adversarial settings simultaneously by proving a lower bound in terms of the effective gaps.

In Section 3, we argued that the PIBAI problem is impossible when there exists suboptimal arms with non-positive effective gaps; the CBAI problem also has this property. Indeed, if $i \neq i^*$ satisfies $\tilde{\Delta}_i \leq 0$, then there exist distributions G_{i^*} and G_i such that the resulting contaminated distributions $\tilde{F}_i := (1-\varepsilon)F_i + \varepsilon G_i$ and $\tilde{F}_{i^*} := (1-\varepsilon)F_{i^*} + \varepsilon G_{i^*}$ have equal medians. Moreover, since the distributions F are arbitrary (CBAI makes no parameteric assumptions), it is impossible to determine whether arm i or i^* has a higher true median. Thus, any CBAI algorithm, even with infinite samples, cannot succeed with probability more than $\frac{1}{2}$. Therefore, we only consider the setting where all effective gaps $\tilde{\Delta}_i$ are strictly positive.

The results in this section lower bound the sample complexity of any CBAI algorithm in terms of the effective gaps. We will focus on lower bounds for the function class $\mathcal{F}_{\bar{t},B}$ and provide lower bounds matching the upper bounds for CBAI in Section 4.2.

Theorem 18 *Consider* CBAI *against an oblivious, prescient, or malicious adversary. There exists a positive constant B such that for any number of arms $k \geq 2$, any confidence level $\delta \in (0, \frac{3}{20})$, any suboptimality level $\alpha \in (0, \frac{1}{6})$, any contamination level $\varepsilon \in (0, \frac{1}{15})$, any regularity level $\bar{t} \in (0, \frac{1}{10})$, and any effective gaps $\{\tilde{\Delta}_i\}_{i \in [k] \setminus i^*} \subset (0, \frac{1}{3})$, there exists a* CBAI *instance with $F_1, \ldots, F_k \in \mathcal{F}_{\bar{t},B}$ for which any (α, δ)-PAC algorithm has expected sample complexity*

$$\mathbb{E}[T] = \Omega\left(\sum_{i \in [k] \setminus \{i^*\}} \frac{1}{\max(\tilde{\Delta}_i, \alpha)^2} \log \tfrac{1}{\delta}\right),$$

where $i^* = \arg\max_{i \in [k]} m_1(F_i)$ is the optimal arm.

Taking the limit as $\alpha \to 0$ in Theorem 18 immediately yields the following lower bound on $(0,\delta)$-PAC algorithms.

Corollary 19 *Consider the same setup as in Theorem 18. There exists a* CBAI *instance $F_1, \ldots, F_k \in \mathcal{F}_{\bar{t},B}$ for which any $(0,\delta)$-PAC algorithm has expected sample complexity*

$$\mathbb{E}[T] = \Omega\left(\sum_{i \in [k] \setminus \{i^*\}} \frac{1}{\tilde{\Delta}_i^2} \log \tfrac{1}{\delta}\right).$$

Proof Sketches. We now sketch the proofs of Theorem 18 and Corollary 19. Full details are deferred to Appendix D.remove for brevity of the main text.

The key idea in the proof is to "lift" hard BAI instances to hard CBAI problem instances. Specifically, let $\{P_i\}_{i \in [k]}$ be distributions for arms in a BAI problem, let $\{U_i\}_{i \in [k]}$ be the corresponding unavoidable uncertainties for median estimation of P_i in CBAI , and let i^* denote the best arm. We say $\{P_i\}_{i \in [k]}$ admits an "$(\varepsilon, \mathcal{F}_{\bar{t},B})$ *CBAI-lifting*" to $\{F_i\}_{i \in [k]}$ if (i) $F_i \in \mathcal{F}_{\bar{t},B}$ for each i, (ii) there exists some adversarial distributions $\{G_i\}_{i \in [k]}$ such that each $P_i = (1-\varepsilon)F_i + \varepsilon G_i$; and (iii) the effective gaps $\tilde{\Delta}_i$ for the $\{F_i\}_{i \in [k]}$ are equal to the gaps Δ_i in the original BAI problem. Intuitively, such a lifting can be thought of as choosing F_{i^*} to have the smallest possible median and the other F_i to have the largest possible median, which still being consistent with P_i and remaining in the uncertainty region corresponding to the U_i.

Armed with this informal definition, we now outline the central idea behind the reduction. Let $\{P_i\}_{i \in [k]}$ be a "hard" BAI instance with best arm i^*, and assume it admits an $(\varepsilon, \mathcal{F}_{\bar{t},B})$-CBAI-lifting to $\{F_i\}_{i \in [k]}$. Consider running a CBAI algorithm $\mathcal{A}$ on the samples obtained from $\{P_i\}$; we claim that if $\mathcal{A}$ is (α, δ)-PAC, then it must output an α-suboptimal arm for the original BAI problem. Indeed, by (ii), the samples from the BAI instance have the same law as the samples that would be obtained from the CBAI problem with $\{F_i\}$, and by (iii), the effective gaps in the CBAI problem are equal to the gaps in the original problem. Therefore, $\mathcal{A}$ must have sample complexity for this problem that is no smaller than the sample complexity of the best BAI algorithm.

Note that the $\{F_i\}$ in the above lifting need only *exist*, as we do not explicitly use these distributions but instead simply run $\mathcal{A}$ on samples from the original bandit instance. Ensuring the existence of such a lifting is the main obstacle, due the following technical yet important nuance: most BAI lower bounds are constructed from Bernoulli or Gaussian distributions, neither of which is compatible with the reduction described above. The problem with Bernoulli arms is that they do not have liftings to $\mathcal{F}_{\bar{t},B}$: any resulting F_i would not satisfy Equation 4 and thus cannot be in $\mathcal{F}_{\bar{t},B}$. Gaussian arms run into a different problem: it is not clear how to shift Gaussian arms up or down far enough to change the median by exactly the maximum uncertainty amount U_i. We overcome this nuance by considering arms with *smoothed Bernoulli distribution* $\mathrm{SBer}(p)$, which we define to be the uniform mixture between a Bernoulli distribution with parameter p and a uniform distribution over $[0,1]$. Indeed, this distribution is $(\varepsilon, \mathcal{F}_{\bar{t},B})$-CBAI-liftable: unlike the Bernoulli distribution, it is smooth enough to have liftings in $\mathcal{F}_{\bar{t},B}$; and unlike the Gaussian distribution, the median of the appropriate lifting of $\mathrm{SBer}(p)$ is exactly U_i away from the median of $\mathrm{SBer}(p)$. Both of these facts are simple calculations; see Appendix D for details.

The only ingredient remaining in the proof is to prove that there are hard instances for BAI with SBer-distributed arms, which follows easily from Lemma 1 and Remark 5 in Kaufmann et al. (2016).

5. Quality Guarantees for the Selected Arm

In the previous Section 4, we developed algorithms for CBAI and proved that they select the best (or an approximately best) arm with high probability. However, in many applications, it is desirable to also provide guarantees on the quality of the selected arm such as "with probability at least 80%, a new random variable $Y \sim F_{\hat{I}}$ is at least 10."

Such a quality guarantee could be accomplished directly using the machinery developed earlier in Subsection 4.1 by, for example, estimating the 0.2 quantile in addition to the median (the 0.5 quantile). However, this approach has two problems. First, if we would like to obtain such a guarantee for multiple probability levels (such as $55\%, 60\%, \ldots$, etc.) we would have to perform a separate estimation for each of the corresponding quantiles. Second, and more importantly, estimation of a quantile q from contaminated samples requires control of quantiles in a $\frac{\varepsilon}{2(1-\varepsilon)}$ neighborhood of q (this follows by an identical argument as in Lemma 1), which can severely restrict the range of quantiles that are possible to estimate.

We circument both of these problems by estimating the *median absolute deviation* (MAD) in addition to the median. The MAD describes the scale of the tails of a distribution away from the median and is the appropriate analogue to variance (which may not exist for F and, we stress, is not estimable from contaminated samples) for this contamination model setting.

This section is organized as follows. Subsection 5.1 develops these statistical results for estimation of the MAD from contaminated samples. Subsection 5.2 then uses these results to prove that the CBAI algorithms presented earlier in Subsection 4.2—with no additional modifications—can also provide quality guarantees for the selected arm. That is, we show that in addition to outputting the best (or an approximately best) arm, these algorithms also provide quality guarantees to a certain precision "for free" without needing extra samples.

5.1. Estimation of Second Robust Moment from Contaminated Samples

In this subsection, we develop the statistical results needed to obtain quality guarantees for the selected arm in the CBAI problem. Specifically, we consider only a single arm and obtain non-asymptotic sample-complexity bounds for estimation of the second robust moment (the MAD) from contaminated samples. We do this for all three adversarial models for the contamination (oblivious, prescient, and malicious). These results may be of independent interest to the robust statistics community.

The subsection is organized as follows. First, Subsection 5.1.1 introduces a class of distributions for which the MAD is estimable from contaminated samples. The few assumptions we make are common in the statistics literature, and we give multiple examples showing that many common distributions are included in this class. Subsection 5.1.2 then proves finite-sample guarantees for MAD estimation from contaminated samples for this class of distributions and for all three adversarial contamination settings.

5.1.1. A Class of Distributions with Estimable Second Robust Moment

Like the median, the MAD is not fully identifiable from contaminated samples (see the discussion in Section 2.2). However, we can estimate the MAD up to a reasonable region of uncertainty with only a few additional assumptions.

Definition 20 *For any $\bar{t} \in (0, \frac{1}{2})$, $B > 0$, $\bar{m}_2 > 0$, and $\kappa \geq 0$, let $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ be the family of distributions F with unique robust moments $m_1(F)$, $m_2(F)$, and $m_4(F)$ satisfying:*

(i) (4) *holds for all $x_1, x_2 \in \left[Q_{L,F}(\frac{1}{2}-\bar{t}), Q_{R,F}(\frac{1}{2}+\bar{t})\right] \cup [m_1(F) \pm 2m_2(F)]$.*

(ii) $m_2(F) \leq \bar{m}_2$.

(iii) $m_2(F) \leq \kappa m_4(F)$.

Let us make a few remarks on this definition. i) We emphasize that our CBAI algorithms already have optimality guarantees when the arm distributions are in $\mathcal{F}_{\bar{t},B}$ (see Section 4.2); the additional assumption that the arm distributions are in $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ allows us to also obtain quantile guarantees on the arm returned by the algorithm (see Section 5.2). ii) Property (iii) of the definition requires a bound on the fourth robust moment, the strength of which is dictated by the parameter κ. Informally, higher robust moments are necessary to control the error of estimation of lower robust moments, analogous to how bounds on variance (resp. kurtosis) are typically necessary for estimation of a distribution's mean (resp. variance). iii) Note that distributions in $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ are not required to have densities, nor even be continuous. iv) The family $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ satisfies some natural and expected properties, such as closure under translation. These properties are gathered and proved in Lemma 26, which is deferred to Appendix A for brevity of the main text.

Remark 21 (Examples) *Many common distributions (e.g. the normal distribution, the uniform distribution, the Cauchy distribution, etc.) belong to $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ for some values of the parameters $\bar{t}, B, \bar{m}_2$, and κ. For instance, the uniform distribution on the interval $[a,b]$ is in $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ for any $\bar{t} \in (0, \frac{1}{2})$, any $B \geq 4$, any $\kappa \geq 2$, and any $\bar{m}_2 \geq \frac{b-a}{4}$. Another example is that the Cauchy distribution with scale parameter a belongs to $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ for any $\bar{t} \in (0, \frac{1}{2})$,*

any $B \geq \pi(1+\max(1,\tan(\pi\bar{t})^2))$*, and* $\kappa \geq \sqrt{3}-1$*, and any* $\bar{m}_2 \geq a$*. We recall that a Cauchy distribution with scale parameter* $a>0$ *has density* $\frac{1}{\pi a}\frac{1}{1+\left(\frac{x-x_0}{a}\right)^2}, x\in\mathbb{R}$*, where* $x_0 \in \mathbb{R}$ *is a location parameter. For such a distribution, it is easy to check that* $m_1(F)=x_0$*,* $m_2(F)=a$ *and* $m_4(F)=a(\sqrt{3}-1)$*. Note that neither the mean nor the variance exists for the Cauchy distribution.*

5.1.2. Concentration Results for Estimation of Second Robust Moment

We are now ready to present finite-sample guarantees on the speed of convergence of the empirical MAD $\hat{m}_2$ to the underlying distribution's MAD m_2 when given contaminated samples. For brevity, proofs are deferred to Appendix B.3.

We first present results for the oblivious and prescient adversarial settings. As before, the estimation error decomposes into the sum of two terms: a bias term reflecting the uncertainty the adversary can inject given her contamination level, and a confidence-interval term that shrinks with an optimal $n^{-1/2}$ rate and has sub-Gaussian tails in δ.

Lemma 22 *Let* $\bar{t}\in(0,\frac{1}{2})$*,* $\varepsilon\in(0,\min(\bar{\varepsilon}(\bar{t}),\frac{1}{B}))$*, and* $F\in\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$*. Let* $Y_i\sim F$ *and* $D_i\sim Ber(\varepsilon)$*, for* $i\in[n]$*, all be drawn independently. Let* $\{Z_i\}_{i\in[n]}$ *be arbitrary random variables possibly depending on* $\{Y_i,D_i\}_{i\in[n]}$*, and define* $X_i=(1-D_i)Y_i+D_iZ_i$*. Then, for any confidence level* $\delta>0$*, error level* $E>0$*, and number of samples* $n\geq 2\max\left(\frac{16\kappa^2B^2\bar{m}_2^2}{E^2},\left(\min(\bar{t},\frac{1}{B})-\frac{\varepsilon}{2(1-\varepsilon)}\right)^{-2}\right)\log\frac{4}{\delta}$*,*

$$\mathbb{P}\left(|\hat{m}_2(X_1,\dots,X_n)-m_2(F)|\leq(1+2\kappa)U_{\varepsilon,B,m_2(F)}+E\right)\geq 1-\delta.$$

Similar to median estimation above, it is possible to estimate the MAD even in the malicious adversarial setting.

Lemma 23 *Let* $\bar{t}\in(0,\frac{1}{2})$*,* $\varepsilon\in(0,\min(\bar{t},\frac{1}{B}))$*, and* $F\in\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$*. Let* (Y_i,D_i)*, for* $i\in[n]$*, be drawn independently with marginals* $Y_i\sim F$ *and* $D_i\sim Ber(\varepsilon)$*. Let* $\{Z_i\}_{i\in[n]}$ *be arbitrary random variables possibly depending on* $\{Y_i,D_i\}_{i\in[n]}$*, and define* $X_i=(1-D_i)Y_i+D_iZ_i$*. Then, for any confidence level* $\delta>0$*, error level* $E>0$*, and number of samples* $n\geq 2\max\left(\frac{16\kappa^2B^2\bar{m}_2^2}{E^2},\left(\min(\bar{t},\frac{1}{B})-\varepsilon\right)^{-2}\right)\log\frac{6}{\delta}$*,*

$$\mathbb{P}\left(|\hat{m}_2(X_1,\dots,X_n)-m_2(F)|\leq(1+2\kappa)U^{(\text{MALICIOUS})}_{\varepsilon,B,m_2(F)}+E\right)\geq 1-\delta.$$

5.2. Algorithmic Guarantees

We now apply these statistical results for MAD estimation to show that the CBAI algorithms presented earlier in Subsection 4.2—with no additional modifications—can also provide quality guarantees for the arm they select. That is, we show that in addition to outputting the best (or an approximately best) arm, these algorithms also provide quality guarantees to a certain precision "for free" without needing extra samples.

For simplicity, we will focus on error guarantees for Algorithm 1. A nearly identical (yet technically hairier) argument yields a similar guarantee for our adaptation of the Successive Elimination algorithm for CBAI (Algorithm 3).

We will make the assumption that the arms distributions are in $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ so that the MAD estimation results proved above will apply. The parameters of this family are assumed to be known to the algorithm beforehand. Distributions in $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ have the following property that their lower tails are controlled by their median and MAD.

Lemma 24 *Let* $F\in\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ *and* $Y\sim F$*. Then simultaneously for all* $t\in[0,\bar{t}]$*,*

$$\mathbb{P}\Big(Y\geq m_1(F)-tBm_2(F)\Big)\geq\tfrac{1}{2}+t.$$

Proof By item (6) of Lemma 26, we have that for all $t\in[0,\bar{t}]$, $t=\frac{1}{2}-(\frac{1}{2}-t)\geq\frac{1}{Bm_2(F)}(F^{-1}(\frac{1}{2})-F^{-1}(\frac{1}{2}-t))$. Rearranging yields $F^{-1}(\frac{1}{2}-t)\geq m_1(F)-Bm_2(F)t$ and completes the proof. ■

With this lemma in hand, we turn to guarantees for the arm returned by Algorithm 1. The following guarantees hold for all adversarial settings since the adversarial strength is encapsulated in the definition of the unavoidable uncertainty terms $U_{\hat{i}}$.

Theorem 25 *Let* $F_1,\dots,F_k\in\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$*. Let* $\varepsilon\in(0,\min(\frac{2\bar{t}}{1+2\bar{t}},\frac{1}{B}))$ *and* $\bar{U}:=U_{\varepsilon,B,\bar{m}_2}$ *if the adversary is oblivious or prescient, or let* $\varepsilon\in(0,\min(\bar{t},\frac{1}{B}))$ *and* $\bar{U}:=U^{(\text{MALICIOUS})}_{\varepsilon,B,\bar{m}_2}$ *if the adversary is malicious. Consider using Algorithm 1 for* CBAI *exactly as in Theorem 2. Let* $\hat{I}$ *denote the arm it outputs, let* Y *be a random variable whose conditional distribution on* $\hat{I}$ *is* $F_{\hat{I}}$*, and let* $\hat{m}_1$ *and* $\hat{m}_2$ *denote the empirical median and MAD, respectively, from the samples it has seen from* $F_{\hat{I}}$*. Then, simultaneously for all* $t\in[0,\bar{t}]$*,*

$$\mathbb{P}\Big(Y\geq(\hat{m}_1-tB\hat{m}_2)-\big((\tfrac{1}{2}+2\kappa tB)\alpha+(1+(1+2\kappa)Bt)\bar{U}\big)\Big)\geq\tfrac{1}{2}+t-\tfrac{3\delta}{k}.$$

Proof By definition of Algorithm 1, $\hat{I}$ is sampled $n_{\alpha/2,\delta/k}=2\max(\frac{4B^2\bar{m}_2^2}{\alpha^2},(\bar{t}-\frac{\varepsilon}{2(1-\varepsilon)})^{-2})\log\frac{2k}{\delta}$ times against oblivious and prescient adversaries, and $n_{\alpha/2,\delta/k}=2\max\left(\frac{4B^2\bar{m}_2^2}{\alpha^2},(\bar{t}-\varepsilon)^{-2}\right)\log\frac{3k}{\delta}$ times against malicious adversaries. Therefore—by Corollary 14 and Lemma 22 for oblivious and prescient adversaries, or similarly by Corollary 15 and Lemma 23 for malicious adversaries—we have by a union bound that with probability at least $1-\frac{3\delta}{k}$, both of the following inequalities hold:

$$\begin{aligned}&m_1(F_{\hat{I}})\geq\hat{m}_1-\bar{U}-\tfrac{\alpha}{2},\ \text{ and}\\&m_2(F_{\hat{I}})\leq\hat{m}_2+(1+2\kappa)\bar{U}+2\kappa\alpha.\end{aligned}$$

Whenever this event occurs, we have

$$m_1(F_{\hat{I}})-tBm_2(F_{\hat{I}})\geq(\hat{m}_1-tB\hat{m}_2)-\big((\tfrac{1}{2}+2\kappa tB)\alpha+(1+(1+2\kappa)Bt)\bar{U}\big).$$

Applying Lemma 24 and taking a union bound completes the proof. ■

Note that the bound inside the probability term in Theorem 25 can be far less conservative than the crude bound $\hat{m}_1-tB\bar{m}_2$ if ε and α are small.

6. Conclusion

In this paper, we proposed the Best Arm Identification problem for contaminated bandits (CBAI). This setup can model many practical applications that cannot be modeled by the classical bandit setup. On the way to efficient algorithms for CBAI, we developed tight, non-asymptotic sample-complexity bounds for estimation of the first two robust moments (median and median absolute deviation) from contaminated samples. These results may be of independent interest, perhaps as ingredients for adapting other online learning techniques to similar contaminated settings.

We formalized the contaminated bandit setup as a special case of the more general partially identifiable bandit problem (PIBAI), and presented ways to adapt celebrated Best Arm Identification algorithms for the classical bandit setting to this PIBAI problem. The sample complexity is essentially changed only by replacing the suboptimality "gaps" with suboptimality "effective gaps" to adjust for the challenge of partial identifiability. We then showed how these algorithms apply to the special case of CBAI by making use of the aforementioned guarantees for estimating the median from contaminated samples. We complemented these results with nearly matching information-theoretic lower bounds on the sample complexity of CBAI, showing that (up to a small logarithmic factor) our algorithms are optimal. This answers the question this paper set out to solve: determining the complexity of finding the arm with highest median given contaminated samples. Finally, we used our statistical results on estimation of the second robust moment from contaminated samples to show that without additional samples, our algorithms can also output quality guarantees on the selected arm.

Our paper suggests several potential directions for future work.

- Our upper and lower bounds on the sample complexity of CBAI are off by a logarithmic factor in the number of arms. It is an interesting open question to close this gap, especially since many of the tricks for doing this in the classical BAI setup do not seem to extend to this partially identifiable setting (see Remark 4). One concrete possibility is to obtain a tighter upper bound by adapting the algorithm of (Jamieson et al., 2014) to the CBAI problem. This would require proving a non-asymptotic version of the Law of the Iterated Logarithm for empirical medians, which would be of independent interest.

- We have developed algorithms for online learning in the presence of partial identifiability. How far does this toolkit extend? In particular, do our results apply to more complicated or more general feedback structures such as partial monitoring (see e.g. (Bartók et al., 2014)) or graph feedback (see e.g. (Alon et al., 2017))?

- The contaminated bandit setup models many real-world problems that cannot be modeled by the classical bandit setup. Is it applicable and approachable to formulate other classical online-learning problems in similar contamination setups?

- More abstractly, we think that problems at the intersection of online learning and robust statistics are not only mathematically rich, but also are increasingly relevant, given the recent influx of active learning tasks with data that are not completely trustworthy. It may be valuable to use techniques from one of the fields to approach problems in the other, as we did here.

Acknowledgments

We are grateful to the anonymous reviewers for their insightful comments. We also thank Marco Avella Medina and Philippe Rigollet for helpful discussions. JA is supported by NSF Graduate Research Fellowship 1122374.

Appendix A. Properties of $\mathcal{F}_{\bar{t},B}$ and $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$

The following lemma lists some simple properties of $\mathcal{F}_{\bar{t},B}$ (defined in Definition 10) and $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$ (defined in Definition 20), which we use often throughout the paper.

For shorthand, we denote $I_{F,\bar{t}} := \left[Q_{L,F}(\frac{1}{2}-\bar{t}), Q_{R,F}(\frac{1}{2}+\bar{t})\right]$ for a distribution F and a real number $\bar{t} \in (0, \frac{1}{2})$.

Lemma 26 *If $F \in \mathcal{F}_{\bar{t},B}$, then:*

1. *For any $a \neq 0$ and $b \in \mathbb{R}$, the distribution $F(a \cdot + b)$ is also in $\mathcal{F}_{B,\bar{t}}$.*
2. *F is strictly monotonically increasing in $I_{F,\bar{t}}$.*
3. *$Q_{L,F}(t) = Q_{R,F}(t)$, for all $t \in F(I_{F,\bar{t}}) = [\frac{1}{2} \pm \bar{t}]$.*
4. *$Q_{L,F}(F(x)) = x$ for all $x \in I_{F,\bar{t}}$ and we write F^{-1} for $Q_{L,F} = Q_{R,F}$.*
5. *The left and right quantiles of F are equal in the interval $F(I_{F,\bar{t}}) = [\frac{1}{2}-\bar{t}, \frac{1}{2}+\bar{t}]$.*
6. *For any $u_1, u_2 \in F(I_{F,\bar{t}}) = [\frac{1}{2}-\bar{t}, \frac{1}{2}+\bar{t}]$,*

$$|u_1 - u_2| \geq \frac{1}{Bm_2(F)} |F^{-1}(u_1) - F^{-1}(u_2)|.$$

Moreover if we also have $F \in \mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$, then:

7. *For any $a \neq 0$ and $b \in \mathbb{R}$, the distribution $F(a \cdot + b)$ is in $\mathcal{F}_{B,\bar{t},a\bar{m}_2,\kappa}$.*
8. *Let H be the distribution of $|Y - m|$, where $Y \sim F$. Then $H \in \mathcal{F}_{\bar{t}_H,B_H}$, where $B_H := \kappa B$ and $\bar{t}_H := \min(\frac{1}{2}, \frac{2}{B})$.*

Proof When proved in order, all of these statements follow easily from the definition of the function class and the earlier statements. The only part requiring effort is item 8, which we

now verify. Fix any $r_1, r_2 \in I_{H,\bar{t}_H}$, where without loss of generality $r_1 \geq r_2 \geq 0$. Then

$$\begin{aligned}
H(r_1) - H(r_2) &= \mathbb{P}(|Y - m_1(F)| \leq r_1) - \mathbb{P}(|Y - m_1(F)| \leq r_2) \\
&= [\mathbb{P}(Y \leq m_1(F) + r_1) - \mathbb{P}(Y \leq m_1(F) + r_2)] \\
&\quad + [\mathbb{P}(Y < m_1(F) - r_2) - \mathbb{P}(Y < m_1(F) - r_1)] \\
&\geq F(m_1(F) + r_1) - F(m_1(F) + r_2) \\
&\geq \frac{|r_1 - r_2|}{Bm_2(F)} \qquad (5) \\
&\geq \frac{|r_1 - r_2|}{\kappa B m_4(F)} \\
&= \frac{|r_1 - r_2|}{B_H m_2(H)}.
\end{aligned}$$

The only step requiring justification is the inequality in (5): this is evident by (4) if $r_1, r_2 \in [m_1(F) \pm 2m_2(F)]$, but this condition must be checked. Once we show this condition is met, we are immediately done.

Therefore, it is now sufficient to prove that $r_1, r_2 \in [m_1(F) \pm 2m_2(F)]$. Since $r_1, r_2 \geq 0$, it suffices to show that the largest value in $I_{H,\bar{t}_H}$, namely $Q_{R,H}(\frac{1}{2} + \bar{t}_H)$, is at most $2m_2(F)$. And to show this, it suffices to show $\frac{1}{2} + \bar{t}_H < H(2m_2(F))$. We show this last inequality presently: by an similar argument as in the first few lines of the above display,

$$\begin{aligned}
H(2m_2(F)) - \tfrac{1}{2} &= H(2m_2(F)) - H(m_2(F)) \\
&= [\mathbb{P}(Y \leq m_1(F) + 2m_2(F)) - \mathbb{P}(Y \leq m_1(F) + m_2(F))] \\
&\quad + \mathbb{P}(Y \in [m_1(F) - 2m_2(F), m_1(F) - m_2(F))) \\
&> F(m_1(F) + 2m_2(F)) - F(m_1(F) + m_2(F)) \qquad (6) \\
&\geq \frac{1}{B} \qquad (7) \\
&\geq \bar{t}_H,
\end{aligned}$$

where (6) is because F is monotonically increasing in $[m_1(F) - 2m_2(F), m_1(F) + 2m_2(F)]$ by property (i) in the definition of $\mathcal{F}_{\bar{t},B,\bar{m}_2,\kappa}$, and (7) is due to (4). This completes the proof. ■

Appendix B. Proofs for Estimation from Contaminated Samples

Throughout, we denote the indicator random variable for an event $\mathcal{E}$ by $\mathbb{1}\{\mathcal{E}\}$.

B.1. Estimation of Median for Oblivious and Prescient Adversaries

Proof of Lemma 7 For shorthand, let $\hat{m} := \hat{m}_1(X_1, \ldots, X_n)$ and let $m \in m_1(F)$ be any median of F. For each $i \in [n]$, define the indicator random variable

$$L_i := \mathbb{1}\left\{(D_i = 1) \text{ or } \left(D_i = 0 \text{ and } Y_i \geq Q_{R,F}\left(\tfrac{1}{2(1-\varepsilon)} + a\right)\right)\right\},$$

where $a := \frac{\sqrt{\log(2/\delta)}}{(1-\varepsilon)\sqrt{2n}} < \sqrt{\frac{2\log(2/\delta)}{n}}$. By independence of D_i and Y_i, L_i has mean

$$\mathbb{E}[L_i] = \varepsilon + (1-\varepsilon)\left(1 - \left(\tfrac{1}{2(1-\varepsilon)} + a\right)\right) = \tfrac{1}{2} - (1-\varepsilon)a.$$

Moreover, the $\{L_i\}_{i\in[n]}$ are independent, and thus by Hoeffding's inequality,

$$\mathbb{P}\left(\hat{m} \geq Q_{R,F}\left(\frac{1}{2(1-\varepsilon)} + a\right)\right) \leq \mathbb{P}\left(\sum_{i=1}^n L_i \geq \frac{n}{2}\right) \leq \exp\left(-2n(1-\varepsilon)^2 a^2\right) = \frac{\delta}{2}.$$

Therefore, with probability at least $1 - \frac{\delta}{2}$,

$$\hat{m} - m < Q_{R,F}\left(\frac{1}{2(1-\varepsilon)} + a\right) - Q_{R,F}\left(\frac{1}{2}\right) \leq R\left(\frac{\varepsilon}{2(1-\varepsilon)} + a\right),$$

where the final inequality is due to (3), which we may invoke since $\frac{1}{2} + \frac{\varepsilon}{2(1-\varepsilon)} + a \leq \frac{1}{2} + \bar{t}$ by our choice of n. An identical argument (or by symmetry with $-F$) yields the analogous result for the lower tail of $\hat{m}$, namely that $m - \hat{m} < R(\frac{\varepsilon}{2(1-\varepsilon)} + a)$ with probability at least $1 - \frac{\delta}{2}$. The lemma statement follows by a union bound. ■

B.2. Estimation of Median for Malicious Adversaries

We now turn to providing the proofs of Lemma 8 and 9. Below, it will be convenient to denote by $y_{(k)}$ the k-th order statistic of a (possibly random) sequence $y_1, \ldots, y_n \in \mathbb{R}$.

Informally, the proof of Lemma 8 proceeds by (i) showing that $\hat{m}_1(X_1, \ldots, X_n)$ is deterministically bounded within the order statistics $Y_{(\lfloor \frac{n}{2} \rfloor \pm \sum_{i=1}^n D_i)}$ (Lemma 27), (ii) applying Hoeffding's inequality to argue that w.h.p., at most $\sum_{i=1}^n D_i \approx \varepsilon n$ samples are contaminated, and (iii) reusing the techniques of Lemma 7 to argue that w.h.p., the order statistics of $Y_{(\lfloor \frac{n}{2} \rfloor \pm \varepsilon n)}$ are within the desired error range from $m_1(F)$. Since each of these steps is tight up to a small amount of slack, the proof of the converse Lemma 9 proceeds essentially by just showing that each of these steps occurs also in the opposite direction w.h.p.

The following lemma will be helpful for step (i). Its proof is straightforward by induction on s and is thus omitted.

Lemma 27 *Let $x_i := d_i y_i + (1 - d_i) z_i$, where $y_1 \leq \cdots \leq y_n$ and $z_1, \ldots, z_n$ are arbitrary real-valued sequences, and $d_1, \ldots, d_n$ is an arbitrary binary-valued sequence satisfying $s := \sum_{i=1}^n d_i < \frac{n}{2}$. Then*

$$y_{(\lfloor \frac{n}{2} \rfloor - s)} \leq \hat{m}_1(x_1, \ldots, x_n) \leq y_{(\lceil \frac{n}{2} \rceil + s)}.$$

Proof of Lemma 8 Define for shorthand $a := \sqrt{\frac{\log(3/\delta)}{2n}}$. By Hoeffding's inequality, the event $E := \{\sum_{i=1}^n D_i \leq (\varepsilon + a)n\}$ occurs with probability at least $\mathbb{P}(E) \geq 1 - \frac{\delta}{3}$. Since $\varepsilon + a \leq \bar{t} < \frac{1}{2}$ by our choice of n, Lemma 27 implies that, whenever E occurs,

$$Y_{(\lfloor \frac{n}{2} \rfloor - \lfloor (\varepsilon+a)n \rfloor)} \leq \hat{m}_1(X_1, \ldots, X_n) \leq Y_{(\lceil \frac{n}{2} \rceil + \lfloor (\varepsilon+a)n \rfloor)}$$

also occurs. Now, define for each $i \in [n]$ the indicator random variable $L_i := \mathbb{1}\left\{Y_i > Q_{R,F}(\frac{1}{2}+\varepsilon+2a)\right\}$. Then $\mathbb{E}[L_i] \le \frac{1}{2}-\varepsilon-2a$, so by Hoeffding's inequality,

$$\mathbb{P}\left(Y_{\left(\lceil \frac{n}{2}\rceil + \lfloor(\varepsilon+a)n\rfloor\right)} > Q_{R,F}(\tfrac{1}{2}+\varepsilon+2a)\right) \le \mathbb{P}\left(\sum_{i=1}^n L_i \ge (\tfrac{1}{2}-\varepsilon-a)n\right) \le \exp(-2na^2) = \frac{\delta}{3}.$$

An identical argument (or simply by symmetry on $F(-\cdot)$) also yields that $Y_{\left(\lfloor \frac{n}{2}\rfloor - \lfloor(\varepsilon+a)n\rfloor\right)} < Q_{L,F}(\frac{1}{2}-\varepsilon-2a)$ with probability at most $\frac{\delta}{3}$. We conclude by a union bound that with probability at least $1-\delta$,

$$Q_{L,F}\left(\tfrac{1}{2}-(\varepsilon+2a)\right) \le \hat{m}_1(X_1,\dots,X_n) \le Q_{R,F}\left(\tfrac{1}{2}+(\varepsilon+2a)\right).$$

Whenever this occurs, we have by virtue of (3) that $\sup_{m\in m_1(F)} |\hat{m}_1(X_1,\dots,X_n) - m| \le R(\varepsilon+2a)$, since both $\frac{1}{2}\pm(\varepsilon+2a) \in [\frac{1}{2}\pm\bar{t}]$ by our choice of n. ■

Proof of Lemma 9 Consider any distribution F with unique median 0 satisfying $R(t) = Q_{R,F}(\frac{1}{2}+t) = -Q_{L,F}(\frac{1}{2}-t)$ for each $t \in \bar{t}$. Such an F can be constructed by starting with a Dirac measure δ_0 at zero and then pushing mass to the tails as far as Equation 3 allows. Next, consider the joint distribution on (D,Y,Z) where $Y \sim F$, the conditional distribution of D given Y is $\mathrm{Ber}(2\varepsilon \cdot \mathbb{1}\{Y \le 0\})$ and $Z \sim \delta_{Q_{R,F}(\frac{1}{2}+\varepsilon)}$. The marginal of D is easily seen to be correct, since

$$\mathbb{P}(D=1) = \mathbb{P}(D=1|Y\le 0)\cdot\mathbb{P}(Y\le 0) + \mathbb{P}(D=1|Y>0)\cdot\mathbb{P}(Y>0) = 2\varepsilon\cdot\tfrac{1}{2} = \varepsilon.$$

Let $a := \sqrt{\frac{\log(1/\delta)}{2n}}$, and note that $a < \varepsilon$ by our choice of n. For each $i \in [n]$, define the indicator random variable

$$L_i := \mathbb{1}\left\{(Y_i > R(\varepsilon-a)) \text{ or } (Y_i \le 0 \text{ and } D_i = 1)\right\}.$$

Each of these has mean

$$\mathbb{E}[L_i] \le 1 - (\tfrac{1}{2}+(\varepsilon-a)) + \tfrac{1}{2}(2\varepsilon) = \tfrac{1}{2}+a.$$

Therefore, by Hoeffding's inequality, we conclude that

$$\begin{aligned}\mathbb{P}\Big(|\hat{m}_1(X_1,\dots,X_n) - m_1(F)| < R(\varepsilon-a)\Big) &\le \mathbb{P}\Big(\hat{m}_1(X_1,\dots,X_n) < m_1(F) + R(\varepsilon-a)\Big)\\ &\le \mathbb{P}\left(\sum_{i=1}^n L_i \ge \frac{n}{2}\right) \le \exp\left(-2a^2n\right) = \delta.\end{aligned}$$

■

B.3. Estimation of Second Robust Moment

Here, we prove Lemma 22. This is done by decomposing the MAD estimation error into two terms, each of which resembles the error between a true median (of a distribution related to F) and an empirical median of contaminated samples, and then applying the median estimation guarantees from the previous section. We begin by proving several helpful lemmas.

Lemma 28 *For any (possibly random) sequence $x_1,\dots,x_n \in \mathbb{R}$ and any $c \in \mathbb{R}$,*

$$\left|\hat{m}_1(|x_1+c|,\dots,|x_n+c|) - \hat{m}_1(|x_1|,\dots,|x_n|)\right| \le |c|$$

holds almost surely, over the possible randomness of the sequence.

Proof For any fixed vector $x \in \mathbb{R}^n$, the function $c \mapsto \hat{m}_1(|x_1+c|,\dots,|x_n+c|)$ is 1-Lipschitz since it is the composition of the following 1-Lipschitz functions with respect to the L_∞ norm: adding $c\vec{1}$, taking entrywise absolute values, and taking an order statistic. ■

We also give the following distributional version of Lemma 28.

Lemma 29 *Consider any real-valued random variable X and any $c \in \mathbb{R}$. Assume that $m_1(|X|)$ and $m_1(|X+c|)$ are unique. Then*

$$\left|m_1(|X+c|) - m_1(|X|)\right| \le |c|.$$

Proof Assume without loss of generality that $c \ge 0$; the case $c < 0$ follows by an identical argument or by symmetry. For shorthand, let $m := m_1(|X|)$ and let $\tilde{m} := m_1(|X+c|)$. First, we show that $\tilde{m} \le m+c$, i.e., we show that $\mathbb{P}(|X+c| \le m+c) \ge \frac{1}{2}$. This is straightforward: $\mathbb{P}(|X+c| \le m+c) = \mathbb{P}(-m-2c \le X \le m) \ge \mathbb{P}(-m \le X \le m) = \mathbb{P}(|X| \le m) \ge \frac{1}{2}$, where we the first step is by non-negativity of m and the last step is by definition of m. A nearly identical argument shows $m \le \tilde{m}+c$, namely $\mathbb{P}(|X| \le \tilde{m}+c) = \mathbb{P}(-\tilde{m}-c \le X \le \tilde{m}+c) = \mathbb{P}(-\tilde{m} \le X \le \tilde{m}+2c) \ge \mathbb{P}(-\tilde{m} \le X \le \tilde{m}) = \mathbb{P}(|X| \le \tilde{m}) \ge \frac{1}{2}$. We therefore conclude that $|\tilde{m}-m| \le c$. ■

Lemma 30 *For any distribution F with unique $m_2(F)$ and $m_4(F)$,*

$$m_4(F) \le 2m_2(F).$$

Proof Let $Y \sim F$. By Lemma 29,

$$m_4(F) = m_1\left(\big||Y-m_1(F)| - m_2(F)\big|\right) \le m_1\left(|Y-m_1(F)|\right) + m_2(F) = 2m_2(F).$$

■

We are now ready to prove Lemma 22.

Proof of Lemma 22 For shorthand, denote $\hat{m} := \hat{m}_1(X_1, \ldots, X_n)$, and let H denote the distribution of $|Y - m_1(F)|$ where $Y \sim F$. Combining the fact that $m_2(F) = m_1(H)$ with Lemma 28 and the triangle inequality, the MAD estimation error is bounded above by

$$\begin{aligned}
&\left|\hat{m}_2(X_1, \ldots, X_n) - m_2(F)\right| \\
&= \left|\hat{m}_1\left(|X_1 - \hat{m}|, \ldots, |X_n - \hat{m}|\right) - m_1(H)\right| \\
&\leq \left|\hat{m}_1\left(|X_1 - m_1(F)|, \ldots, |X_n - m_1(F)|\right) - m_1(H)\right| + \left|\hat{m} - m_1(F)\right|. \qquad (8)
\end{aligned}$$

By Corollary 14 and our choice of n, the second error term in (8) is bounded above by $U_{\varepsilon,B,m_2(F)} + \frac{E}{2}$ with probability at least $1 - \frac{\delta}{2}$. To control the first error term in (8), we apply Corollary 14 with the distribution H in lieu of F and the contaminations $\tilde{Z}_i := |Z_i - m_1(F)|$ in lieu of Z_i. Thus, by combining item 8 in Lemma 26 with Corollary 14, this first error term is bounded above by $U_{\varepsilon,\kappa B,m_2(H)} + \frac{E}{2}$ with probability at least $1 - \frac{\delta}{2}$ whenever we have at least $2\max\left(4B^2\kappa^2 m_2^2(H)E^{-2}, (\min(\frac{1}{2}, \frac{1}{B}) - \frac{\varepsilon}{2(1-\varepsilon)})^{-2}\right)\log\frac{4}{\delta}$ samples, which is satisfied because of our choice of n and the inequality $m_2(H) = m_4(F) \leq 2m_2(F)$ from Lemma 30. Therefore, a union bound implies that, with probability at least $1 - \delta$, the MAD estimation error is at most $U_{\varepsilon,B,m_2(F)} + U_{\varepsilon,\kappa B,m_2(H)} + E$. By another application of the inequality $m_2(H) \leq 2m_2(F)$, this is bounded above by $(1 + 2\kappa)U_{\varepsilon,B,m_2(F)} + E$, as desired. ■

The proof of Lemma 23—the analogous result to Lemma 22 but for the *malicious* adversarial setting—is omitted since it is identical to the proof of Lemma 22 with the uses of Corollary 14 replaced by uses of Corollary 15.

Appendix C. Proofs for Adaptation of the Successive Elimination Algorithm

We first define some notation. Let c be a constant such that $n_{\alpha,\delta} \leq \frac{c}{\alpha^2}\log\frac{1}{\delta}$ for all $\alpha > 0$ and $\delta \in (0,1)$; such a constant clearly exists by definition of PIBAI (see Equation 2). Let R denote the number of total rounds in Algorithm 2 before termination, and for each round $r \in [R]$, let S_r denote the set of all arms still in S when entering round r. For succintness, we also denote $\delta_r := \frac{6\delta}{\pi^2 k r^2}$.

Proof of Theorem 3 Without loss of generality, assume that the best arm is $i^* = 1$. Define the event $E := \{|\hat{p}_{i,r} - p_i| \leq U_i + \alpha_{r,\delta_r}, \forall r \geq 1, \forall i \in S_r\}$. Recall that $\hat{p}_{i,r}$ is the estimate of the measure of quality for arm i after r samples, and that arm i will stop being pulled once $i \notin S_r$. For analysis purposes, consider consider virtual estimates $\hat{p}_{i,r}$ that would be obtained if we continued to pull the eliminated arms indefinitely, so that $\hat{p}_{i,r}$ is defined for all $i \in [k]$ and $r \in \mathbb{N}$. By a union bound,

$$\mathbb{P}\left(E^C\right) \leq \sum_{i=1}^{k}\sum_{r=1}^{\infty}\mathbb{P}\left(|\hat{p}_{i,r} - p_i| > U_i + \alpha_{r,\delta_r}\right) \leq \sum_{i=1}^{k}\sum_{r=1}^{\infty}\delta_r = \sum_{i=1}^{k}\sum_{r=1}^{\infty}\frac{6\delta}{\pi^2 k r^2} = \delta,$$

where above we have used (2) and the famous Basel identity.

We conclude from the above that E occurs with probability at least $1 - \delta$. Henceforth let us assume that E occurs. A simple induction argument shows that $1 \in S_r$ for each round $r \in [R]$, which implies that Algorithm 2 returns the optimal arm. Indeed, at each round r for which $1 \in S_r$, E guarantees that for all $j \in S_r$,

$$\hat{p}_{1,r} \geq p_1 - U_1 - \alpha_{r,\delta_r} = \tilde{\Delta}_j + p_j + U_j - \alpha_{r,\delta_r} > p_j + U_j - \alpha_{r,\delta_r} \geq \hat{p}_{j,r} - 2\alpha_{r,\delta_r}.$$

and so by definition of Algorithm 2, the optimal arm 1 is not eliminated at round r.

Now, still assuming that E occurs, let us bound the sample complexity T. For each $i \in [k]$, denote by T_i the number of times that arm i is pulled. Clearly, $T = \sum_{i=1}^{k} T_i$. Moreover, since arm 1 is never eliminated so long as E occurs, $T \leq 2\sum_{i=2}^{k} T_i$. For each $i \geq 2$, arm i is eliminated no later than the first round r in which $\hat{p}_{i,r} < \hat{p}_{1,r} - 2\alpha_{r,\delta_r}$ which, by E, is satisfied as soon as $p_i + U_i + \alpha_{r,\delta_r} < p_1 - U_1 - 3\alpha_{r,\delta_r}$, or equivalently $\tilde{\Delta}_i > 4\alpha_{r,\delta_r}$. We conclude that arm i is eliminated in the first round r where

$$\tilde{\Delta}_i > 4\alpha_{r,\delta_r} = 4\sqrt{\frac{c\log(\frac{\pi^2 k r^2}{6\delta})}{r}},$$

which occurs for $r \leq C\frac{1}{\tilde{\Delta}_i^2}\log\left(\frac{k}{\delta\tilde{\Delta}_i}\right)$ for some universal constant $C > 0$. Hence,

$$T \leq \sum_{j=2}^{k} T_j = O\left(\sum_{j=2}^{k}\frac{1}{\tilde{\Delta}_i^2}\log\left(\frac{k}{\delta\tilde{\Delta}_i}\right)\right).$$

■

Proof of Theorem 17 The proof is nearly identical to that of Theorem 3. The main difference in the proof of correctness is that at each round $r \geq 0$, if any arm is still in S_r, it has been pulled at least

$$\begin{aligned}
N\log\left(\frac{\pi^2 k}{2\delta}\right) + 2N\sum_{t=1}^{r}\log\left(\frac{t+1}{t}\right) + r &= N\log\left(\frac{\pi^2 k(r+1)^2}{2\delta}\right) + r \\
&\geq \left(\frac{2B^2\bar{m}_2^2}{\alpha_{r,\delta_r}^2} + N\right)\log\frac{3}{\delta_r} \\
&\geq \max\left(\frac{2B^2\bar{m}_2^2}{\alpha_{r,\delta_r}^2}, N\right)\log\frac{3}{\delta_r}
\end{aligned}$$

times. Thus, we may apply Corollary 14 (for oblivious or prescient adversaries) or Corollary 15 (for malicious adversaries) to obtain the identical guarantees as needed in the proof of Theorem 3 for estimating the median up to accuracy α_{r,δ_r} with probability at least $1 - \delta_r$. Hence, the $(0,\delta)$-PAC guarantee follows easily by identical reasoning. The only difference in the sample complexity proof is that it we must keep track of the additional draws

$$\sum_{i=1}^{k}\left[\left\lceil N\log\left(\frac{\pi^2 k}{2\delta}\right)\right\rceil + \sum_{t=1}^{T_i}\left\lceil 2N\log\left(\frac{t+1}{t}\right)\right\rceil\right] \leq \sum_{i=1}^{k}\left[(T_i+1) + N\log\left(\frac{\pi^2 k(T_i+1)^2}{2\delta}\right)\right],$$

where T_i is the number of times that arms i is pulled. By what was shown in Theorem 3, the first term $\sum_{i=1}^{k}(T_i+1) = O(\sum_{i\neq i^*}\frac{1}{\Delta_i^2}\log(\frac{k}{\delta\Delta_i}))$, and the second term $\sum_{i=1}^{k} N\log(\frac{\pi^2 k(T_i+1)^2}{2\delta}) = O(N\sum_{i\neq i^*}\log(\frac{k}{\delta\Delta_i}\log(\frac{k}{\delta\Delta_i}))) = O(N\sum_{i\neq i^*}\log(\frac{k}{\delta\Delta_i}))$. ■

Appendix D. Proofs for Lower Bounds

In this section, we make the proof sketch in Subsection 4.3 formal. The proof is broken into two parts. First, we exhibit hard instances for BAI in which the arms all have smoothed Bernoulli distributions. Second, we reduce this instance into a lower bound instance for CBAI.

Throughout, we adopt the standard assumption in the multi-armed bandit literature (Mannor and Tsitsiklis, 2004) that all algorithms we consider have a stopping time T which is almost surely finite.

D.1. BAI Lower Bound

Here, we prove a gap-dependent lower bound for the BAI problem that we will make use of in the following subsection. Mannor and Tsitsiklis (2004) gave the first such lower bound by exhibiting hard instances for BAI using Bernoulli-distributed arms. However, our CBAI reduction will not work with Bernoulli distributions (see the proof sketch in Subsection 4.3), and so here we exhibit hard instances for BAI using instead a distribution that will work for our CBAI reduction, namely the smoothed Bernoulli distribution.

Recall that we define the smoothed Bernoulli distribution with parameter $p \in (0,1)$, denoted by $\mathrm{SBer}(p)$, to be the uniform mixture of the Bernoulli distribution with parameter p and the uniform distribution on $[0,1]$ with weights $\frac{1}{2}$ and $\frac{1}{2}$. It is easy to see that for $p, q \in (0,1)$, the Kullback-Leibler divergence between $\mathrm{SBer}(p)$ and $\mathrm{SBer}(q)$ is given by

$$\begin{aligned}\mathsf{KL}(\mathrm{SBer}(p), \mathrm{SBer}(q)) &= \frac{1}{2}\mathsf{KL}(\mathrm{Ber}(p), \mathrm{Ber}(q)) \\ &= \frac{1}{2}\left(p\log\left(\frac{p}{q}\right) + (1-p)\log\left(\frac{1-p}{1-q}\right)\right).\end{aligned}$$

In particular, a second-order Taylor expansion yields that, for all $\eta \in (0, \frac{1}{2})$, there exists some positive constant C_η such that the inequality

$$\mathsf{KL}(\mathrm{SBer}(p), \mathrm{SBer}(q)) \leq C_\eta (p-q)^2 \tag{9}$$

holds for all $p, q \in [\eta, 1-\eta]$.

With Equation 9 in hand, the following lemma becomes a direct consequence of Lemma 1 and Remark 5 from Kaufmann et al. (2016).

Lemma 31 *Let $k \geq 2$, $\eta \in (0, \frac{1}{2})$, and $p_1, \ldots, p_k \in [\eta, 1-\eta]$. Consider the instance of* BAI *where arm $i \in [k]$ has distribution $SBer(p_i)$. Then, for any $\alpha > 0$ and any $\delta \in (0, \frac{3}{20})$, any (α, δ)-PAC* BAI *algorithm must use at least the following number of samples in expectation:*

$$\mathbb{E}[T] \geq \frac{C_\eta}{4}\left(\sum_{i\in[k]\setminus\{i*\}} \frac{1}{\max(\Delta_i, \alpha)^2}\log\left(\frac{1}{2.4\delta}\right)\right),$$

where $i^* := \arg\max_{i\in[k]} p_i$, and $\Delta_i := p_{i^*} - p_i$ for each $i \in [k]\setminus\{i^*\}$.

D.2. CBAI Lower Bound

In this subsection, we show how to prove the lower bounds for CBAI using the technique sketched in Subsection 4.3. In particular, we will show how to prove Theorem 18 by "CBAI-lifting" the MAB instances we proved were hard in Lemma 31. The proof of Corollary 19 then follows immediately by letting $\alpha \to 0$; since an algorithm that returns the *best* arm with probability at least $1-\delta$, is by definition a $(0,\delta)$-PAC algorithm.

Proof of Theorem 18 Fix any $k \geq 2$, $\delta \in (0, \frac{3}{20})$, $\alpha \in (0, \frac{1}{6})$, $\varepsilon \in (0, \frac{1}{15})$, and $\bar{t} \in (0, \frac{1}{10})$. With $\eta = \frac{1}{3}$, Lemma 31 proves that for any $p_1, \ldots, p_k \in [\frac{1}{3}, \frac{2}{3}]$, the BAI instance with $\tilde{F}_i := \mathrm{SBer}(p_i)$ distributed-arms has the property that any (α, δ)-PAC BAI algorithm must use at least

$$\Omega\left(\sum_{i\in[k]\setminus\{i^*\}} \frac{1}{\max(\Delta_i, \alpha)^2}\log\frac{1}{\delta}\right) \tag{10}$$

samples in expectation, where $i^* := \arg\max_{i\in[k]} p_i$ and $\{\Delta_i\}_{i\in[k]\setminus\{i^*\}}$ are the gaps w.r.t. the arm distributions $\{\mathrm{SBer}(p_i)\}_{i\in[k]}$. Without loss of generality, let us assume $1 = \arg\max_{i\in[k]} p_i$ (arm 1 is the best). At this point we now treat the different adversarial settings separately, since the lower bounds and thus also the liftings differ.

D.2.1. Lifting for Oblivious and Prescient Adversaries

Define the following distributions:

$$\begin{aligned}F_1 :&= \frac{1-2\varepsilon}{2(1-\varepsilon)}\mathrm{Ber}(r) + \frac{1}{2(1-\varepsilon)}\mathrm{Unif}([0,1]), \quad \text{and} \\ F_i :&= \frac{1-2\varepsilon}{2(1-\varepsilon)}\mathrm{Ber}(q_i) + \frac{1}{2(1-\varepsilon)}\mathrm{Unif}([0,1]) \quad \forall i \in \{2,\ldots,k\},\end{aligned}$$

where $r := \frac{p_1}{1-2\varepsilon}$ and $q_i := \frac{p_i - 2\varepsilon}{1-2\varepsilon}$. It is not hard to see that

$$\begin{aligned}\tilde{F}_1 &:= \mathrm{SBer}(p_1) = (1-\varepsilon)F_1 + \varepsilon\delta_0, \quad \text{and} \\ \tilde{F}_i &:= \mathrm{SBer}(p_i) = (1-\varepsilon)F_i + \varepsilon\delta_1 \quad \forall i \in \{2,\ldots,k\}.\end{aligned}$$

In other words, *samples generated from $SBer(p_i)$ are equal in distribution to samples generated from the above contaminated mixture model* of $(1-\varepsilon)F_i$ and ε times a Dirac measure.

Next, a simple calculation shows that $m_1(\tilde{F}_1) = p_1$, $m_1(F_1) = p_1 + \varepsilon$, and $m_2(F_1) = \frac{1-\varepsilon}{2}$. Similarly, for any $i \in \{2,\ldots,k\}$, we have $m_1(\tilde{F}_i) = p_i$, $m_1(F_i) = p_i - \varepsilon$, and $m_2(F_i) = \frac{1-\varepsilon}{2}$. Moreover, for all $i \in [k]$, we have that $F_i \in \mathcal{F}_{B,\bar{t}}$ for $B = 4$ and any $\bar{t} < \frac{1}{2(1-\varepsilon)}\min(p_i + \varepsilon, 1-$

$(p_i+\varepsilon)) \le \frac{1}{2}\cdot\frac{15}{14}\cdot\min(\frac{1}{3}, 1-(\frac{2}{3}+\frac{1}{15})) = \frac{1}{7}$. We conclude that for each $i\in[k]$, the change in median between the distribution F_i and the contaminated distribution $\tilde{F}_i$ is equal to

$$|m_1(\tilde{F}_i) - m_1(F_i)| = \varepsilon = Bm_2(F_i)\frac{\varepsilon}{2(1-\varepsilon)} = U_{\varepsilon,B,m_2(F_i)}.$$

Therefore, we conclude that running any (α,δ)-approximate CBAI algorithm $\mathcal{A}$ on the samples obtained from the above BAI instance will result in $\mathcal{A}$ returning arm $\hat{I}$ satisfying $m_1(F_i) \ge m_1(F_1) - (2U_{\varepsilon,B,\frac{1-\varepsilon}{2}} + \alpha)$ with probability at least $1-\delta$. Whenever this event occurs, we have by the above calculations that $p_{\hat{i}} \ge p_1 - \alpha$. Therefore, $\mathcal{A}$ returned an α approximate arm with probability at least $1-\delta$ for this hard BAI instance. We conclude from the lower bound in (10) that $\mathcal{A}$ must use at least $\Omega\left(\sum_{i\in[k]\setminus\{i^*\}}\frac{1}{\max(\tilde{\Delta}_i,\alpha)^2}\log\frac{1}{\delta}\right)$ samples in expectation, where $\{\tilde{\Delta}_i\}_{i\in[k]\setminus\{i^*\}}$ are the effective gaps w.r.t. the arm distributions $\{F_i\}_{i\in[k]}$.

D.2.2. Lifting for Malicious Adversaries

The idea is similar to the oblivious and prescient case done above. The difference is that malicious adversaries can shift quantiles further (see Corollary 15) and so we must exhibit a lifting that exactly matches this larger shift. Consider the distributions over the CBAI arms

$$\begin{aligned} F_1 &:= \tfrac{1}{2}\mathrm{Ber}(p_1+2\varepsilon) + \tfrac{1}{2}\mathrm{Unif}([0,1]), \quad \text{and} \\ F_i &:= \tfrac{1}{2}\mathrm{Ber}(p_i-2\varepsilon) + \tfrac{1}{2}\mathrm{Unif}([0,1]) \quad \forall i\in\{2,\dots,k\}. \end{aligned}$$

We now present the malicious CBAI adversarial strategy. For the optimal arm 1, define the joint distribution J_1 over (Y_1,Z_1,D_1) where $Y_1\sim F_1$, $Z_1\sim\delta_0$, and the conditional distribution of D_1 given Y_1 is $\mathrm{Ber}(\varepsilon(\frac{p_1}{2}+\varepsilon)^{-1}\mathbb{1}\{Y_1=1\})$. Similarly, for each suboptimal arm $i\in\{2,\dots,k\}$, define the joint distribution J_i over (Y_i,Z_i,D_i) to be $Y_i\sim F_i$ and $Z_i\sim\delta_1$, and the conditional distribution of D_i given Y_i is $\mathrm{Ber}(\varepsilon(\frac{1-p_1}{2}+\varepsilon)^{-1}\mathbb{1}\{Y_i=0\})$. It is simple to see that for each arm $i\in[k]$, the marginals are correct under each J_i. Indeed, a simple conditioning calculation yields

$$\begin{aligned} \mathbb{P}(D_1=1) &= \mathbb{P}(D_1=1|Y_1=1)\mathbb{P}(Y_1=1) + \mathbb{P}(D_1=1|Y_1\neq 1)\mathbb{P}(Y_1\neq 1) \\ &= \varepsilon(\tfrac{p_1}{2}+\varepsilon)^{-1}\cdot\tfrac{1}{2}(p_1+2\varepsilon) + 0 \\ &= \varepsilon. \end{aligned}$$

An identical argument shows that the marginal distribution of D_i is also equal to $\mathrm{Ber}(\varepsilon)$ for each suboptimal arm $i\in\{2,\dots,k\}$. Now, for each arm $i\in[k]$, denote by C_i the corresponding contaminated distributions induced by $(1-D_i)Y_i+D_iZ_i$ where $(Y_i,Z_i,D_i)\sim J_i$. It is not hard to see that

$$\begin{aligned} \tilde{F}_1 &:= \mathrm{SBer}(p_1) = C_1, \quad \text{and} \\ \tilde{F}_i &:= \mathrm{SBer}(p_i) = C_i, \quad \forall i\in\{2,\dots,k\} \end{aligned}$$

In other words, *samples generated from SBer(p_i) are equal in distribution to the samples generated by the malicious* CBAI *adversary's distribution C_i.*

A simple calculation shows that for the optimal arm, $(F_1)^{-1}(\frac{1}{2}) = p_1+2\varepsilon$ and $(\tilde{F}_1)^{-1}(\frac{1}{2}) = (F_1)^{-1}(\frac{1}{2}-\varepsilon) = p_1$. Note further that F_1 has cdf satisfying $F_1(s) = \frac{1}{2}(1-p_1-2\varepsilon)+\frac{1}{2}s$ for

all $s\in[0,1)$, and that $F_1(1)=1$. Thus, for any $x_1,x_2\in[Q_{L,F_1}(\frac{1}{2}-\bar{t}), Q_{R,F_1}(\frac{1}{2}+\bar{t})]$, we have that $|F(x_1)-F(x_2)| = \frac{1}{2}|x_1-x_2|$ since $[\frac{1}{2}\pm\bar{t}] \subseteq [0.4, 0.6]$ is contained within the interval $[\frac{1}{2}(p_1+2\varepsilon), 1-\frac{1}{2}(p_1+2\varepsilon)] \supseteq [\frac{1}{2}(\frac{2}{3}+2\cdot\frac{1}{15}), 1-\frac{1}{2}(\frac{2}{3}+2\cdot\frac{1}{15})] = [0.4,0.6]$. By definition, this implies that $F_1\in\mathcal{F}_{\bar{t},B_1}$ with $B_1m_2(F_1)=2$. (Note that B_1 is a finite constant since $p_1\in(\frac{1}{3},\frac{2}{3})\subset(0,1)$ implies $m_2(F_i)>0$.) A completely identical calculation shows that each suboptimal arm $i\in\{2,\dots,k\}$ satisfies $(F_i)^{-1}(\frac{1}{2}) = p_i-2\varepsilon$, $(\tilde{F}_i)^{-1}(\frac{1}{2}) = (F_i)^{-1}(\frac{1}{2}+\varepsilon) = p_i$, and $F_i\in\mathcal{F}_{\bar{t},B_i}$ with $B_im_2(F_i)=2$. Therefore, we conclude that for each $i\inf[k]$, the difference in medians between the distributions F_i and $\tilde{F}_i$ is equal to

$$|m_1(F_i) - m_1(\tilde{F}_i)| = 2\varepsilon = B_im_2(F_i)\varepsilon = U^{(\text{MALICIOUS})}_{\varepsilon,B_i,m_2(F_i)},$$

which is exactly equal to the largest possible uncertainty.An identical reduction argument as in the oblivious and prescient adversary finishes the proof.

■

References

Robin Allesiardo and Raphaël Féraud. Selection of learning experts. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1005–1010. IEEE, 2017.

Robin Allesiardo, Raphaël Féraud, and Odalric-Ambrym Maillard. The non-stationary stochastic multi-armed bandit problem. *International Journal of Data Science and Analytics*, (4):267–283, 2017.

Noga Alon, Nicolo Cesa-Bianchi, Claudio Gentile, Shie Mannor, Yishay Mansour, and Ohad Shamir. Nonstochastic multi-armed bandits with graph-structured feedback. *SIAM Journal on Computing*, 46(6):1785–1826, 2017.

Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University Press, 2009.

Gábor Bartók, Dean P Foster, Dávid Pál, Alexander Rakhlin, and Csaba Szepesvári. Partial monitoring – classification, regret bounds, and algorithms. *Mathematics of Operations Research*, (4):967–997, 2014.

Robert Eric Bechhofer, Jack Kiefer, and Milton Sobel. *Sequential identification and ranking procedures: with special reference to Koopman-Darmois populations*. University of Chicago Press, 1968.

Christian Bontemps, Thierry Magnac, and Eric Maurin. Set identified linear models. *Econometrica*, (3):1129–1155, 2012.

Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, (1):1–122, 2012.

Sébastien Bubeck, Nicolo Cesa-Bianchi, and Gábor Lugosi. Bandits with heavy tail. *IEEE Transactions on Information Theory*, (11):7711–7717, 2013.

Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Symposium on Theory of Computing (STOC)*, pages 47–60. ACM, 2017.

Lijie Chen and Jian Li. On the optimal sample complexity for best arm identification. *arXiv preprint arXiv:1511.03774*, 2015.

Yeshwanth Cherapanamjeri, Prateek Jain, and Praneeth Netrapalli. Thresholding based Efficient Outlier Robust PCA. *arXiv preprint arXiv:1702.05571*, 2017.

Herman Chernoff. *Sequential analysis and optimal design*. SIAM, 1972.

Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robustly learning a gaussian: Getting optimal error, efficiently. In *Symposium on Discrete Algorithms (SODA)*, pages 2683–2702. SIAM, 2018.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. PAC bounds for multi-armed bandit and markov decision processes. In *Conference on Computational Learning Theory (COLT)*, pages 255–270. Springer, 2002.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research (JMLR)*, (Jun):1079–1105, 2006.

Victor Gabillon, Mohammad Ghavamzadeh, and Alessandro Lazaric. Best arm identification: A unified approach to fixed budget and fixed confidence. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3212–3220, 2012.

Aurélien Garivier and Emilie Kaufmann. Optimal best arm identification with fixed confidence. In *Conference on Learning Theory (COLT)*, pages 998–1027, 2016.

Frank R Hampel. The influence curve and its role in robust estimation. *Journal of the American Statistical Association*, (346):383–393, 1974.

Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust statistics: the approach based on influence functions*. John Wiley & Sons, 2011.

Joel L Horowitz and Charles F Manski. Identification and robustness with contaminated and corrupted data. *Econometrica*, pages 281–302, 1995.

Peter J Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, pages 73–101, 1964.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 240–248, 2016.

Kevin Jamieson, Matthew Malloy, Robert Nowak, and Sebastien Bubeck. On finding the largest mean among many. *arXiv preprint arXiv:1306.3917*, 2013.

Kevin Jamieson, Matthew Malloy, Robert Nowak, and Sébastien Bubeck. lilucb: An optimal exploration algorithm for multi-armed bandits. In *Conference on Learning Theory (COLT)*, pages 423–439, 2014.

Kwang-Sung Jun, Lihong Li, Yuzhe Ma, and Jerry Zhu. Adversarial attacks on stochastic bandits. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3644–3653, 2018.

Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. PAC subset selection in stochastic multi-armed bandits. In *International Conference on Machine Learning (ICML)*, pages 655–662, 2012.

Zohar Karnin, Tomer Koren, and Oren Somekh. Almost optimal exploration in multi-armed bandits. In *International Conference on Machine Learning (ICML)*, pages 1238–1246, 2013.

Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. On the complexity of best-arm identification in multi-armed bandit models. *The Journal of Machine Learning Research (JMLR)*, (1):1–42, 2016.

Michael Kearns and Ming Li. Learning in the presence of malicious errors. *SIAM Journal on Computing*, (4):807–837, 1993.

Kevin A Lai, Anup B Rao, and Santosh Vempala. Agnostic estimation of mean and covariance. In *Foundations of Computer Science (FOCS)*, pages 665–674. IEEE, 2016.

Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, (1):4–22, 1985.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *arXiv preprint arXiv:1603.06560*, 2016.

Shie Mannor and John N Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research (JMLR)*, (Jun):623–648, 2004.

Charles F Manski. *Identification for prediction and decision*. Harvard University Press, 2009.

Ricardo A Maronna and Víctor J Yohai. Robust estimation of multivariate location and scatter. *Journal of the American Statistical Association*, 1976.

Jacob Marschak and William H Andrews. Random simultaneous equations and the theory of production. *Econometrica*, pages 143–205, 1944.

Margaret Martin and Straf Miron. *Principles and practices for a federal statistical agency*. National Academies Press, 1992.

Joseph P Romano and Azeem M Shaikh. Inference for the identified set in partially identified econometric models. *Econometrica*, (1):169–211, 2010.

Peter J Rousseeuw and Annick M Leroy. *Robust regression and outlier detection*. John Wiley & Sons, 2005.

Yevgeny Seldin and Aleksandrs Slivkins. One practical algorithm for both stochastic and adversarial bandits. In *International Conference on Machine Learning (ICML)*, pages 1287–1295, 2014.

Leslie G Valiant. Learning disjunction of conjunctions. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 560–566, 1985.

Journal of Machine Learning Research 20 (2019) 1-6 Submitted 7/18; Revised 5/19; Published 5/19

AffectiveTweets: a Weka Package for Analyzing Affect in Tweets

Felipe Bravo-Marquez FBRAVO@DCC.UCHILE.CL
Department of Computer Science, University of Chile & IMFD, Santiago, Chile

Eibe Frank EIBE@WAIKATO.AC.NZ
Bernhard Pfahringer BERNHARD@WAIKATO.AC.NZ
Department of Computer Science, University of Waikato, Hamilton, New Zealand

Saif M. Mohammad SAIF.MOHAMMAD@NRC-CNRC.GC.CA
National Research Council Canada, Ottawa, ON, Canada

Editor: Antti Honkela

Abstract

AffectiveTweets is a set of programs for analyzing emotion and sentiment of social media messages such as tweets. It is implemented as a package for the Weka machine learning workbench and provides methods for calculating state-of-the-art affect analysis features from tweets that can be fed into machine learning algorithms implemented in Weka. It also implements methods for building affective lexicons and distant supervision methods for training affective models from unlabeled tweets. The package was used by several teams in the shared tasks: EmoInt 2017 and Affect in Tweets SemEval 2018 Task 1.

Keywords: Twitter, Sentiment Analysis, Emotion Analysis, Affective Computing, Lexicon Induction, Distant Supervison

1. Introduction

Social media platforms such as Twitter are widely used to express affective states, often in relation to particular entities, topics or events. We use the term "affect" to encompass sentiment, emotion, mood, and other related mental states.

The automatic detection of affect from tweets[1] has become a popular task due to its importance in commerce, politics, and sports (Liu, 2012). Most successful solutions to this problem, including its many variations, are based on supervised machine learning models trained on manually annotated examples (Mohammad et al., 2018).

The process of building an affect detection system for tweets comprises three main components: 1) the training dataset on which the system is trained, 2) the function used for mapping tweets, provided as strings, into fixed size vectors of feature-value pairs, and 3) the machine learning algorithm used for training the model for detecting affect.

Replicating an affect detection system reported in a scientific publication can be a challenging endeavor. These systems usually include complex hand-engineered features and specific parameter setups for the learning algorithm (Mohammad et al., 2018). Unfortunately, these aspects are not always described in sufficient detail.

1. We use the terms "social media post" and "tweet" interchangeably.

©2019 Felipe Bravo-Marquez, Eibe Frank, Bernhard Pfahringer, and Saif M. Mohammad.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v20/18-450.html.

We present AffectiveTweets, a software for easily prototyping affect analysis systems for social media posts using Weka (Hall et al., 2009), a widely-used open source machine learning workbench implemented in Java.

2. The AffectiveTweets Package

The software is implemented as a Weka[2] package that can be installed with the Weka package manager. It can be accessed through Weka's GUIs or the command line interface. It is licensed under the GNU General Public License, Version 3 and hosted on Github.[3] Instructions for installing, using, testing, and contributing to the software are given on the project's website.[4]

The main functionalities of the package are implemented as Weka filters: tools enabling data transformations for the underlying machine learning task. AffectiveTweets provides three types of filters: 1) tweet-level filters, 2) word-level filters, and 3) distant supervision filters.

Other resources provided by the package are: training datasets, affective lexicons (lists of words annotated by affect), pre-trained word embeddings, and wrappers to the Twitter-specific tokenizer and POS tagger implemented in the *TweetNLP* library (Gimpel et al., 2011).

The package was made available as the official baseline system for the WASSA-2017 Shared Task on Emotion Intensity (EmoInt) (Mohammad and Bravo-Marquez, 2017) and for SemEval 2018 task 1 (Affect in Tweets) (Mohammad et al., 2018). It was used by 5 teams in the former task and by 15 teams in the latter.

2.1. Tweet-level Filters

Tweet-level filters in the package act on string attributes containing the text of tweets and calculate features suitable for further processing with the learning algorithms implemented in Weka. Many of these features have been used by state-of-the-art systems (Kiritchenko et al., 2014; Mohammad et al., 2018). The filters are described in the following.

The *TweetToSparseFeatureVector* filter calculates several sparse features for every tweet: word n-grams (adding a NEG prefix to words occurring in negated contexts), character n-grams, POS tags, and Brown word clusters. The scope of a negation is determined by a simple heuristic: from the occurrence of a negator word up until a punctuation mark or the end of the sentence. A list of negator words such as *no, not, won't* and *never* is provided with the package.

The *TweetToLexiconFeatureVector* filter calculates features by aggregating word-level affect scores taken from a fixed list of well-known affect lexicons. This list includes AFINN (Årup Nielsen, 2011), the Sentiment140 lexicon (Kiritchenko et al., 2014), and others. Users are encouraged to cite the publications describing any of the lexicons or resources used with the package.[5]

2. https://www.cs.waikato.ac.nz/ml/weka/
3. https://github.com/felipebravom/AffectiveTweets
4. https://affectivetweets.cms.waikato.ac.nz/
5. The BibTex entries of these publications are given at the package's website.

The *TweetToInputLexiconFeatureVector* filter allows users to calculate features from a tweet using their own affective lexicons in ARFF format.[6] The features are calculated by adding or counting the affect associations of the words matching the given lexicons. All numeric and nominal attributes from each lexicon are considered. Numeric scores are added and nominal labels are counted.

The *TweetToSentiStrengthFeatureVector* filter calculates positive and negative sentiment intensities for a tweet using the *SentiStrength* lexicon-based method (Thelwall et al., 2012).

The *TweetToEmbeddingsFeatureVector* filter calculates a tweet-level feature representation using pre-trained word embeddings (word vectors) by applying one of the following aggregation schemes: average of word embeddings; addition of word embeddings; or concatenation of the first k word embeddings in the tweet. The concatenation scheme is suitable for training deep neural networks with the *WekaDeeplearning4j* package (Lang et al., 2019).

2.2. Word-level Filters

The word-level filters allow users to build their own affective lexicons, which can later be used to calculate tweet-level features via the *TweetToInputLexiconFeatureVector* filter.

The *PMILexiconExpander* filter calculates the Pointwise Mutual Information (PMI) semantic orientation score for each word in a corpus of tweets annotated by sentiment (Turney, 2002). The score is calculated by subtracting the PMI of the target word with a negative sentiment from the PMI of the target word with a positive sentiment.

The *TweetCentroid* filter calculates word distributional vectors from a corpus of unlabeled tweets by treating words as the centroid of the tweet vectors in which they appear (Bravo-Marquez et al., 2015). The resulting word vectors can be labeled with an affective lexicon in ARFF format using the *LabelWordVectors* filter, making it possible to train a word-level affective classifier on these words. This classifier can be used to expand the original affective lexicon to the words from the provided corpus that are not covered by the lexicon. Word embeddings trained with neural networks can also be used in a similar way using filters provided by the *WekaDeepLearning4j* package.

2.3. Distant Supervision Filters

Distant supervision models are heuristic labeling functions used for automatically creating training data from unlabeled corpora. These models have been widely adopted for training affective models for tweets because large amounts of unlabeled tweets can easily be obtained through the use of the Twitter API. The filters described below take a collection of unlabeled tweets and a polarity lexicon of positive and negative words in ARFF format as input.

The *LexiconDistantSupervision* filter implements the most popular distant supervision approach for Twitter affective analysis: If a word from the lexicon is found, the tweet is labeled with the word's polarity. Tweets with both positive and negative words are discarded. The filter allows the removal of the lexicon word from the tweet's content. A list of positive and negative emoticons is used as the default lexicon (Read, 2005).

Two additional distant supervision methods that generate positive and negative instances by averaging tweet vectors annotated according to a given polarity lexicon are

6. `https://waikato.github.io/weka-wiki/arff_stable/`

also provided. These methods have shown to be superior to the emoticon-based approach (Bravo-Marquez et al., 2016a),(Bravo-Marquez et al., 2016b).

The *ASA* filter implements the Annotate-Sample-Average (ASA) (Bravo-Marquez et al., 2016a) distant supervision method. It generates synthetic labeled instances by sampling a number of tweets containing at least one word from the lexicon with the desired polarity, and averaging the feature vectors of the sampled tweets.

The *PTCM* filter implements the Partitioned Tweet Centroid Model (PTCM) (Bravo-Marquez et al., 2016b). This method uses the tweet centroid model for calculating word vectors and a polarity lexicon for training a word-level classifier. The classifier is deployed on tweets represented by the original tweet-level representation used to calculate the word vectors. The model increases the number of labeled instances by partitioning the tweets associated with each word into smaller, disjoint subsets and calculating one centroid per partition, which is labeled according to the lexicon.

3. Related Tools and Benchmark

There is no existing tool for building affective models for tweets implementing all the functionalities provided by AffectiveTweets, but some related tools exist. The Stanford *CoreNLP* software allows training recursive deep models on a sentiment treebank (Socher et al., 2013). *SentiStrength* (Thelwall et al., 2012) and *Vader* (Hutto and Gilbert, 2014) are both lexicon-based systems that calculate sentiment intensity scores for social media messages. The *R* package *TM* (Meyer et al., 2008) for text mining has a plugin for sentiment classification based on an affective lexicon. *The Natural Language Toolkit* (NLTK) (Bird and Loper, 2004) provides a sentiment analysis module implementing the Vader method and the Bing Liu's lexicon.[7]

For demonstration, we benchmark *AffectiveTweets* against similar and equivalent models built using the NLTK sentiment analysis module and *Scikit-learn* (Pedregosa et al., 2011) on the dataset from the "SemEval 2013 Sentiment Analysis in Twitter Message Polarity Classification" task (Nakov et al., 2013). Classification results on the testing partition and execution times are shown in Table 1. Instructions for reproducing these experiments are given on the project's website.

Features	Implementation	Kappa	F_1	Time (seconds)
Word n-grams	Scikitlearn + NLTK	0.42	0.64	30.7
Word n-grams	AffectiveTweets	0.45	0.66	13.0
Word n-grams + Liu Lexicon	Scikitlearn + NLTK	0.48	0.68	13.4
Word n-grams + Liu Lexicon	AffectiveTweets	0.48	0.68	27.4
Liu Lexicon + Vader	Scikitlearn + NLTK	0.41	0.63	8.9
Liu Lexicon + SentiStrength	AffectiveTweets	0.40	0.63	31.9
Word n-grams + Liu Lexicon + Vader	Scikitlearn + NLTK	0.51	0.70	16
Word n-grams + Liu Lexicon + SentiStrength	AffectiveTweets	0.49	0.69	68.5
Word n-grams + All lexicons + SentiStrength	AffectiveTweets	0.52	0.71	74.6

Table 1: Benchmark results. The word n-grams include features for $n = 1, 2, 3$ and 4. Each model consists of a logistic regression trained on the corresponding features. Execution time is averaged over 10 repetitions.

7. `http://www.nltk.org/api/nltk.sentiment.html`

4. Acknowledgments

Felipe Bravo-Marquez was funded by Millennium Institute for Foundational Research on Data.

References

Finn Årup Nielsen. A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. In *Proceedings of the 1st Workshop on Making Sense of Microposts (#MSM2011)*, pages 93–98, May 2011.

Steven Bird and Edward Loper. NLTK: The natural language toolkit. In *The Companion Volume to the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics*, pages 214–217, Barcelona, Spain, July 2004. Association for Computational Linguistics.

Felipe Bravo-Marquez, Eibe Frank, and Bernhard Pfahringer. From unlabelled tweets to Twitter-specific opinion words. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 743–746, 2015.

Felipe Bravo-Marquez, Eibe Frank, and Bernhard Pfahringer. Annotate-sample-average (ASA): A new distant supervision approach for Twitter sentiment analysis. In *Proceedings of 22nd European Conference on Artificial Intelligence*, pages 498–506, 2016a.

Felipe Bravo-Marquez, Eibe Frank, and Bernhard Pfahringer. From opinion lexicons to sentiment classification of tweets and vice versa: A transfer learning approach. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 145–152, 2016b.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 42–47, 2011.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

Clayton J. Hutto and Eric Gilbert. VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM*, pages 216–225, 2014.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, 50:723–762, 2014.

Steven Lang, Felipe Bravo-Marquez, Christopher Beckham, Mark Hall, and Eibe Frank. Wekadeeplearning4j: A deep learning package for weka based on deeplearning4j. *Knowledge-Based Systems*, 2019.

Bing Liu. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2012.

David Meyer, Kurt Hornik, and Ingo Feinerer. Text mining infrastructure in R. *Journal of Statistical Software*, 25(5):1–54, 2008.

Saif M. Mohammad and Felipe Bravo-Marquez. WASSA-2017 shared task on emotion intensity. In *Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA)*, 2017.

Saif M. Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. Semeval-2018 Task 1: Affect in tweets. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, 2018.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. Semeval-2013 task 2: Sentiment analysis in twitter. In *Proceedings of the seventh international workshop on Semantic Evaluation Exercises*, pages 312–320, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

Jonathon Read. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *Proceedings of the ACL Student Research Workshop*, pages 43–48, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642. Association for Computational Linguistics, 2013.

Mike Thelwall, Kevan Buckley, and Georgios Paltoglou. Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology*, 63(1):163–173, 2012.

Peter D. Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 417–424. Association for Computational Linguistics, 2002.